Better approximations to cumulative normal functions

Graeme West
Financial Modelling Agency and
Programme in Advanced Mathematics of Finance,
School of Computational & Applied Mathematics,
University of the Witwatersrand, Private Bag 3, Wits 2050, South Africa
graeme@finmod.co.za

1. The need for High Precision Cumulative Normal Functions

Espen Haug relates a story to me of how his book (Haug 1998) has received a rather scathing review at the Amazon website by one reader; and the underlying reason for the problem is in actual fact the inaccuracy of the cumulative normal approximation in his book, this inaccuracy is in turn inherited by the bivariate cumulative approximation. As a consequence, option prices where the bivariate cumulative is used can be negative, under not absurd inputs!

It is important to remember that in most if not all approximations, the n-variate cumulative function will use the $n-1$-variate. It makes sense a priori to have a high precision univariate cumulative normal, but it makes even more sense if we are going to use the bivariate cumulative normal, as—besides needing to be satisfactory in its own right—this will rely on the univariate that we have chosen. And, if we use a trivariate cumulative then one will require a high precision bivariate cumulative function.

2. Univariate Cumulative Normal

The Cumulative Standard Normal Integral is the function:

$$N(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{-\frac{X^2}{2}} dX \qquad (1)$$

As is well known, a closed form solution does not exist for this integral, so a numerical approximation needs to be implemented. Most common is an approximation which involves an exponential and a fifth degree polynomial, given in (Abramowitz & Stegun 1974), and repeated in (Hull 2002, §12.9) and (Haug 1998, Appendix A), for example. (Some other shorter and less accurate approximations are available, but the above

Thanks to Axel Vogt for some testing of the accuracy of some function implementations, and Espen Haug for suggesting this topic.

algorithm is hardly complicated.) This function is used by most option exchanges for futures option pricing and margining, and hence may be preferred to better methods, in order to maintain consistency with the results from the exchange.

However, another option is one that first appears in (Hart 1968). This algorithm uses high degree rational functions to obtain the approximation. This function is accurate to double precision throughout the real line.

We can compare the performance of the excel NORMSDIST function, the (Abramowitz & Stegun 1974) function (AS henceforth) and the Hart function. The poor reputation of the NORMSDIST function is probably unwarranted, with this and the AS function materially the same for inputs in the range $[-6, +6]$, which of course is more than adequate for all purposes. (Anybody working so far in the tails of a normal distribution in financial mathematics is probably working with the wrong distribution anyway, so whether or not the results that are returned there are accurate is probably moot.) In Figure 1 we see the relative values of these functions ('f relative to g' at a point x means that we are graphing the value $\frac{f(x)-g(x)}{g(x)}$). Here we see the consistency of NORMSDIST until -6 (as all the functions involved are symmetric, we only plot for negative real numbers), after which its behaviour is rather mysterious.

A vb version of the Hart function is in Figure 2.[1]

As pointed out in (Acklam 2004), having such a double precision function has some rather pleasant spin-offs. For example, the Moro transform to find inverse cumulative normals is well known. Having the ability to generate normally distributed variables from a (quasi) random uniform sample is clearly important in work involving Monte Carlo experiments, and the Moro transformation is fast and accurate to about 10 decimal places.[2] Given a function that can compute the normal cumulative distribution function to double precision, the Moro approximation (and, in fact, ANY initial approximation) of the inverse normal cumulative distribution function can be refined to full machine precision, by a fairly straightforward application of Newton's method. In fact, higher degree methods such as Newton's second order method (sometimes called the Newton-Bailey method) or a third order method known as Halley's method will be the fastest, and are very amenable here, because the Gaussian function is so easily differentiated over and over—see (Acklam 2002) and (Acklam 2004).

3. Bivariate Cumulative Normal

The cumulative bivariate normal distribution is the function

$$N_2(x, y, \rho) = \frac{1}{2\pi\sqrt{1-\rho^2}} \int_{-\infty}^{x} \int_{-\infty}^{y} \exp\left[\frac{-(X^2 - 2\rho XY + Y^2)}{2(1-\rho^2)}\right] dY\, dX$$

Again, approximations are required. The most common algorithm is that of (Drezner 1978), which appears in both (Hull 2002, Appendix 12C) and in (Haug 1998, Appendix A.2), for example.

A more accurate pair of approximations was developed in (Drezner & Wesolowsky 1989). A first function is given on pg. 103 of that paper. However, it is then acknowledged that for ρ near to ± 1, there will be inaccuracies, and so another algorithm is provided on pg. 105. We will call these algorithms DW1 and DW2 respectively.

The DW2 method is single precision. (Genz 2004) has provided a modification of that algorithm which is near double precision. Again, we have a vb version of the FORTRAN code of Genz. Adaptation was needed because the algorithm calculated the complementary probability i.e. the probability that $X \geq x$, $Y \geq y$, given the correlation coefficient. The algorithm has been adapted to return the more usual (in mathematical finance anyway!) probability that $X \leq x$, $Y \leq y$.

In (Ağca & Chance 2003) the speed of the various bivariate approximations is considered.

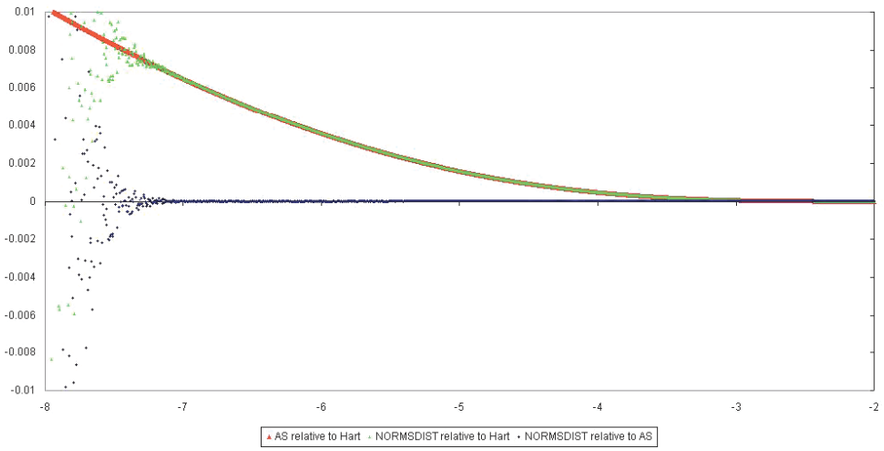

Figure 1: Relative values of the three univariate normal approximations.

4. Option Pricing Disasters

4.1 Problems with the univariate

The probably esoteric advantage mentioned in §2 of using the double precision algorithm—in order to 'tone' our cumulative normal inverse approximations—pales into insignificant when we analyse the example provided by (the critic of) Espen Haug.

The example that Espen tells me of is a partial-time-start-barrier option (Haug 1998, §2.10.3) i.e. an up-and-out call type A with asset price $S = 75$, strike price $X = 85$, barrier $H = 95$, time to maturity $t_1 = 0.35$ years, time to maturity $t_2 = 0.5$ years, risk free rate 10%, cost of carry 5%, and volatility 4%, with continuous monitoring of the barrier. The option price returned by Espen's software is -0.0393. For a volatility of 3%, it returns a value of -3860.5652, and as $\sigma \downarrow 0$, so $V \downarrow -\infty$.

```
Function Cumnorm(x As Double) As Double
  XAbs = Abs(x)
  If XAbs > 37 Then
    Cumnorm = 0
  Else
    Exponential = Exp(-XAbs ^ 2 / 2)
    If XAbs < 7.07106781186547 Then
      build = 3.52624965998911E-02 * XAbs + 0.700383064443688
      build = build * XAbs + 6.37396220353165
      build = build * XAbs + 33.912866078383
      build = build * XAbs + 112.079291497871
      build = build * XAbs + 221.213596169931
      build = build * XAbs + 220.206867912376
      Cumnorm = Exponential * build
      build = 8.83883476483184E-02 * XAbs + 1.75566716318264
      build = build * XAbs + 16.064177579207
      build = build * XAbs + 86.7807322029461
      build = build * XAbs + 296.564248779674
      build = build * XAbs + 637.333633378831
      build = build * XAbs + 793.826512519948
      build = build * XAbs + 440.413735824752
      Cumnorm = Cumnorm / build
    Else
      build = XAbs + 0.65
      build = XAbs + 4 / build
      build = XAbs + 3 / build
      build = XAbs + 2 / build
      build = XAbs + 1 / build
      Cumnorm = Exponential / build / 2.506628274631
    End If
  End If
  If x > 0 Then Cumnorm = 1 - Cumnorm
End Function
```

Figure 2: A double precision univariate normal function. (All variables which are not declared here are set as private variables elsewhere.)

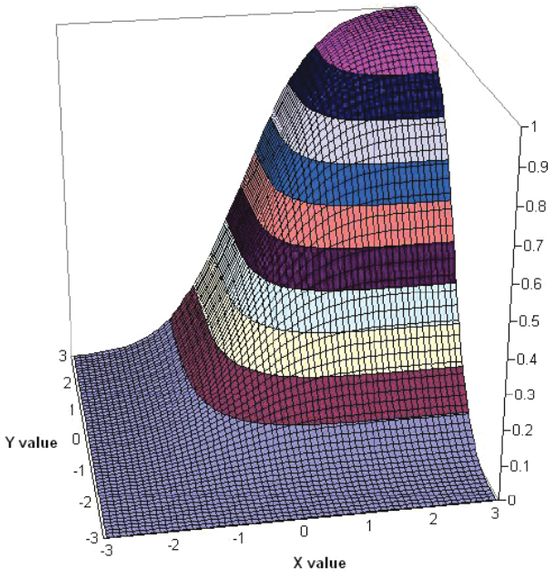

Figure 3: The bivariate cumulative normal function, $\rho = 50\%$.

Why does this come about? We created code for this and the other option pricing formulae mentioned here, where in addition to the parameters of the option, one needs to specify which univariate and which bivariate approximation one is using (the bivariate itself calling the specified univariate when required). Thus, one can dig down into each factor of the option price and isolate the problem.

We fix a volatility of 3%. Using the code notation of Espen, the option price involves the calculation of the value $N_2(g_4, -e_4, \rho)$. We have $g_4 = 7.54255645241296$, $e_4 = 12.7827258096518$ and $\rho = 0.25$. Using the bivariate method of (Drezner 1978) and the AS and Hart univariate method, one obtains values of 5.24808418944644E-10 and 0 respectively. (The execution of the Drezner function involves several recursive calls to itself, which includes calls to the relevant univariate function.) While the difference may not appear material, this value is subsequently multiplied by $(h/S)^{2\mu} = 504727548721.962$ in the option price!

In actual fact, the problem is the evaluation of $N(0)$. While we all know that the answer is 0.5, the AS code doesn't. It returns the value 0.5000000010279300. While of course this is correct to the claimed 6 decimal places, it is the root of the problem. If you add in the AS algorithm a clause which instructs the function that if the input is 0, to return 0.5 exactly, and exit, the problem goes away! Of course, this is not exactly a very appealing solution: it should worry. But the Hart algorithm does return 0.5 to double precision.

A similar problem can be created for the Bjerksund and Stensland formula (Bjerksund & Stensland 2002). In both cases the problem can be resolved by using the corrected AS algorithm or the Hart algorithm.

The rule of thumb is that as soon as our option becomes exotic, and we need an n variate cumulant, the cumulant functions of lower order should be double precision. But even that might not be enough, as we will see now.

4.2 Problems with the bivariate: $\rho = \pm 1$

A problem that arises with the Drezner bivariate function is the failure to account for the case where $\rho = \pm 1$. This observation is important, because even though the initial correlation might not be equal to ± 1, the algorithm of Drezner might make a function call where the correlation is indeed equal to ± 1.

Let us consider the call on the minimum option pricing formula of (Stulz 1982) with $\sigma_1 = 40.00\%$, $\sigma_2 = 25.00\%$, $\rho = -1.00\%$, $S_1 = S_2 = X = 100$, $\tau = 2$, $r = 8.00\%$. Once again the option price will include the calculation of the quantity $N_2(a, b, \rho)$, where $a = -4.9065389333868\,E-17$, $b = 0.275771644662754$ and $\rho = -0.01$. The Drezner algorithm now performs a very extensive recursive evaluation, which critically relies on which octant of three dimensional space (a, b, ρ) lies. And here, the question of which octant a point on one of the xy, xz and yz planes is assigned to is crucial. Essentially, the problem is familiar: the code does not recognise that 'in reality' $a = 0$, so the above point is on the yz plane, and should be assigned to an octant where $x \geq 0$, rather than an octant where $x < 0$. As a consequence the code now fails to execute because a value of $\rho = 1$ is used. The Drezner algorithm always involves a normalisation of a and b by division by $\sqrt{2(1-\rho^2)}$, and so a division by 0 occurs. If, in this particular case, we manually override a with a value of 0, the Drezner algorithm executes properly (because a function value with $\rho = 1$ never occurs), returning a value of 0.302786942980365.

The general solution is not to try to manipulate the definition of the octants, but rather to build in traps for the limiting cases. Note that in the sense of a limit

$$N_2(x, y, 1) = N(\min(x, y)) \tag{3}$$

$$N_2(x, y, -1) = \begin{cases} 0 & \text{if } y \leq -x \\ N(x) + N(y) - 1 & \text{if } y > -x \end{cases} \tag{4}$$

So, we can build in a test if $|\rho| = 1$, and if so, the algorithm executes as above and the function exits.

It is not even sufficient in the Drezner algorithm to test directly if $|\rho| = 1$, because to machine precision this can be false, while to the same precision it is true that $\sqrt{2(1-\rho^2)} = 0$. So the latter needs to be the criterion for testing.

4.3 Problems with the bivariate: negative option values

Taking the care mentioned in §4.1 turns out to be insufficient. The DW1 algorithm fails to avoid the problem of negative option values even when the univariate is double precision, and even when the correlation coefficient is quite far away from ± 1. We can find inputs where the call on the minimum formula of (Stulz 1982) will return negative values even for $\rho \approx -70\%$. For example, with $S_1 = 85$, $S_2 = 60$, $\sigma_1 = 40.00\%$, $\sigma_2 = 25.00\%$, $X = 100$, $\rho = -70.00\%$, $r = 8.00\%$ and $\tau = 2$ years using the DW1 algorithm (with an underlying Hart univariate) gives an option premium of -0.0038939. When using the DW2 algorithm the value improves to 0.0180211. The price using the Genz algorithm is 0.0180005. See Figure 4.

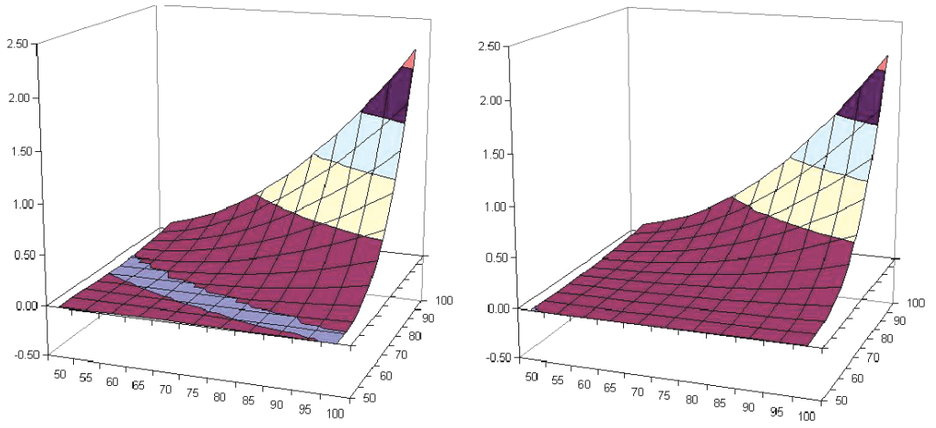

Figure 4: The price of a call on the minimum, using the DW1 bivariate cumulant and the DW2 bivariate cumulant. The underlying univariate is the Hart function in both cases. $\rho = -70\%$.

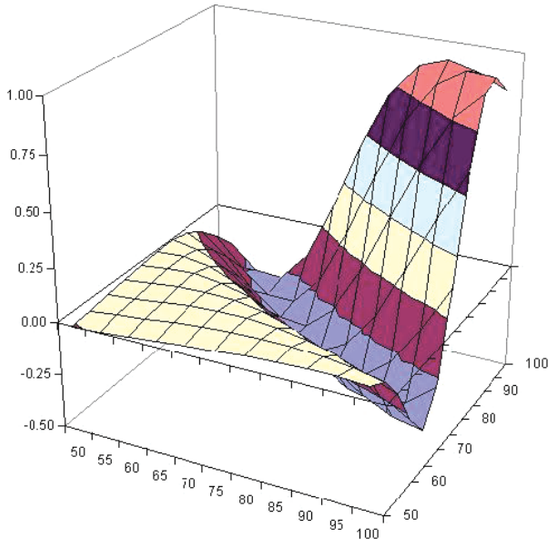

Figure 5: The price of a call on the minimum, using the DW1 bivariate cumulant. The underlying univariate is the Hart function. $\rho = -97\%$.

This valley of negative option values is exacerbated as $\rho \to -1$—see Figure 5.

4.4 Problems with the bivariate: underflow

Remarkably, I have stumbled on problems with the DW2 algorithm too. This algorithm is in Figure 6.

```
Function Bivarcumnorm(a As Double, b As Double, r As Double) As Double

  Dim i As Integer
  Dim x As Variant, W As Variant
  Dim h1 As Double, h2 As Double
  Dim LH As Double, h12 As Double
  dim h3 As Double, h5 As Double, h6 As Double, h7 As Double
  Dim r1 As Double, r2 As Double, r3 As Double, rr As Double
  Dim AA As Double, ab As Double

  x = Array(0.04691008, 0.23076534, 0.5, 0.76923466, 0.95308992)
  W = Array(0.018854042, 0.038088059, 0.0452707394, 0.038088059, 0.018854042)
  h1 = a
  h2 = b
  h12 = (h1 * h1 + h2 * h2) / 2
  If Abs(r) >= 0.7 Then
    r2 = 1 - r * r
    r3 = Sqr(r2)
    If r < 0 Then h2 = -h2
    h3 = h1 * h2
    h7 = Exp(-h3 / 2)
    If Abs(r) < 1 Then
      h6 = Abs(h1 - h2)
      h5 = h6 * h6 / 2
      h6 = h6 / r3
      AA = 0.5 - h3 / 8
      ab = 3 - 2 * AA * h5
      LH = 0.13298076 * h6 * ab * (1 - Cumnorm(h6)) _
              - Exp(-h5 / r2) * (ab + AA * r2) * 0.053051647
      For i = 1 To 5
        r1 = r3 * x(i)
        rr = r1 * r1
        r2 = Sqr(1 - rr)
        LH = LH - W(i) * Exp(-h5 / rr) * (Exp(-h3 / (1 + r2)) / r2 / h7 - 1 - AA * rr)
      Next i
    End If
    Bivarcumnorm = LH * r3 * h7 + Cumnorm(Min(h1, h2))
    If r < 0 Then
      Bivarcumnorm = Cumnorm(h1) - Bivarcumnorm
    End If
  Else
    h3 = h1 * h2
    If r <> 0 Then
      For i = 1 To 5
        r1 = r * x(i)
        r2 = 1 - r1 * r1
        LH = LH + W(i) * Exp((r1 * h3 - h12) / r2) / Sqr(r2)
      Next i
    End If
    Bivarcumnorm = Cumnorm(h1) * Cumnorm(h2) + r * LH
  End If
End Function
```

Figure 6: A visual basic version of the DW2 function, modified to return the usual rather than complementary probabilities.

An obvious coding failsafe strategy is always to check that whenever division occurs in any of your algorithms, that division cannot be by 0. An obvious example where this immediately bears fruit is having option pricing formulae that correctly return the intrinsic value of the option at maturity, rather than not executing. Recall that any formula of Black-Scholes type will have one or more divisions by the square root of the annualised term left to expiry in their execution.

This problem occurs in the DW2 algorithm, but in quite a subtle way. Examining Figure 6, we see that a quantity h_7 is defined to be $\exp(-h_3/2)$. Later on we have a loop which involves evaluation of the quantity $\exp(-h_3/(1+r_2))/r_2/h_7$. If h_3 is large, then h_7 might evaluate as 0 to the precision of the language we are employing—that is, underflow.[3] In this case, the above fraction is of the form 0/0, and will not evaluate. However, this form suggests l'Hôpital's rule, and sure enough, the necessary calculations can be made, showing that this quantity is equal to 0, in the sense of a limit.

The closed form American option pricing model of (Bjerksund & Stensland 2002) uses the bivariate normal, and use of the DW2 algorithm without the above modification will fail if the dividend yield is non-zero but very small. For example, with a strike of 100, term of half a year, risk free rate of 10%, dividend yield of 0.10%, and volatility 20% the option price will fail for calls. At one point the bivariate function is called with a set of parameters that is reminiscent of those that we saw in §4.1, and this time the underflow explained above occurs. Note that if the dividend yield were zero, then it is known that it is sub-optimal to exercise calls early, and the model of (Bjerksund & Stensland 2002) correctly diverts to the Black-Scholes formula.

So, this modified DW2 algorithm might be the algorithm of choice. It is not as accurate as the Genz algorithm, but does not have any material inaccuracies, and is certainly a lot more compact. The modified algorithm is in Figure 7.

```
Function Bivarcumnorm(a As Double, b As Double, r As Double) As Double

  Dim i As Integer
  Dim x As Variant, W As Variant
  Dim h1 As Double, h2 As Double
  Dim LH As Double, h12 As Double
  dim h3 As Double, h5 As Double, h6 As Double, h7 As Double, h8 As Double
  Dim r1 As Double, r2 As Double, r3 As Double, rr As Double
  Dim AA As Double, ab As Double

  x = Array(0.04691008, 0.23076534, 0.5, 0.76923466, 0.95308992)
  W = Array(0.018854042, 0.038088059, 0.0452707394, 0.038088059, 0.018854042)
  h1 = a
  h2 = b
  h12 = (h1 * h1 + h2 * h2) / 2
  If Abs(r) >= 0.7 Then
    r2 = 1 - r * r
    r3 = Sqr(r2)
    If r < 0 Then h2 = -h2
    h3 = h1 * h2
    h7 = Exp(-h3 / 2)
    If Abs(r) < 1 Then
      h6 = Abs(h1 - h2)
      h5 = h6 * h6 / 2
      h6 = h6 / r3
      AA = 0.5 - h3 / 8
      ab = 3 - 2 * AA * h5
      LH = 0.13298076 * h6 * ab * (1 - Cumnorm h6)) _
               - Exp(-h5 / r2) * (ab + AA * r2) * 0.053051647
      For i = 1 To 5
        r1 = r3 * x(i)
        rr = r1 * r1
        r2 = Sqr(1 - rr)
        If h7 = 0 Then
          h8 = 0
        Else
          h8 = Exp(-h3 / (1 + r2)) / r2 / h7
        End If
        LH = LH - W(i) * Exp(-h5 / rr) * (h8 - 1 - AA * rr)
      Next i
    End If
    Bivarcumnorm = LH * r3 * h7 + Cumnorm(Min(h1, h2))
    If r < 0 Then
      Bivarcumnorm = Cumnorm(h1) - Bivarcumnorm
    End If
  Else
    h3 = h1 * h2
    If r <> 0 Then
      For i = 1 To 5
        r1 = r * x(i)
        r2 = 1 - r1 * r1
        LH = LH + W(i) * Exp((r1 * h3 - h12) / r2) / Sqr(r2)
      Next i
    End If
    Bivarcumnorm = Cumnorm(h1) * Cumnorm(h2) + r * LH
  End If
End Function
```

Figure 7: A visual basic version of the DW2 function, modified to return the usual rather than complementary probabilities, with the underflow problem resolved.

5. The Trivariate Cumulative Normal Function

The cumulative trivariate normal distribution is the function

$$N_3(x_1, x_2, x_3, \Sigma) = \frac{1}{(2\pi)^{3/2}\sqrt{|\Sigma|}} \int_{-\infty}^{x_1}\int_{-\infty}^{x_2}\int_{-\infty}^{x_3} \exp\left(\tfrac{1}{2}\underline{X}'\Sigma^{-1}\underline{X}\right) dX_3\, dX_2\, dX_1 \tag{5}$$

where Σ is the correlation matrix between standardised (scaled) variables X_1, X_2, X_3, and $|\cdot|$ denotes determinant. Denote by $N_3(x_1, x_2, x_3, \rho_{21}, \rho_{31}, \rho_{32})$ the function $N_3(x_1, x_2, x_3, \Sigma)$ where

$$\Sigma = \begin{bmatrix} 1 & \rho_{21} & \rho_{31} \\ \rho_{21} & 1 & \rho_{32} \\ \rho_{31} & \rho_{32} & 1 \end{bmatrix}.$$

Again, approximations are required. Code for the trivariate cumulative normal is not generally available. There are a few highly non-transparent publications, for example (Schervish 1984), but this code is known to be faulty. We have used the algorithm in (Genz 2004). This has required extensive modifications because the algorithms are implemented in Fortran, using language properties which are not readily translated. The function in (Genz 2004) returns the complementary probability, again, we have modified to return the usual probability that $X_i \le x_i$ ($i = 1, 2, 3$) given a correlation matrix. Again, it is claimed that this algorithm is double precision; high accuracy (of our vb translation) has been verified by testing against Niederreiter quasi-Monte Carlo integration (using the Matlab algorithm qsimvn.m, also at the website of Genz).

In a forthcoming paper, we will look at applying this function to the pricing of rainbow options on 3 assets, as in the work of (Johnson 1987).

FOOTNOTES & REFERENCES

1. This and all of the other vba code mentioned here is available from the author's web page (West 2004)
2. In contrast to before, the reader is now warned against the use of the built in excel function NORMSINV, which is patently absurd. Of course, this inverse function should take values in the interval (0, 1) and should map to the real line. In fact, NORMSINV returns the value $\pm$ 50000 for input values within 0.0000003 of 1 or 0 respectively. Given that such values close to 0 or 1 on occasion are provided by uniform random number generators, this approach is to be avoided.
3. This is dependent on the language used. But this example fails in C++, for example.

- Abramowitz, M. & Stegun, I. A. (1974), *Handbook of Mathematical Functions, With Formulas, Graphs, and Mathematical Tables*, Dover.
- Acklam, P. J. (2002), 'A small paper on Halley's method'. http://home.online.no/~pjacklam
- Acklam, P. J. (2004), 'An algorithm for computing the inverse normal cumulative distribution function'. http://home.online.no/~pjacklam
- Ağca, Ş. & Chance, D. M. (2003), 'Speed and accuracy comparison of bivariate normal distribution approximations for option pricing', *Journal of Computational Finance* **6**(4), 61–96.
- Bjerksund, P. & Stensland, G. (2002), 'Closed form valuation of American options'. http://www.nhh.no
- Drezner, Z. (1978), 'Computation of the bivariate normal integral', *Mathematics of Computation* **32**, 277–279.
- Drezner, Z. & Wesolowsky, G. (1989), 'On the computation of the bivariate normal integral', *Journal of Statist. Comput. Simul.* **35**, 101–107.
- Genz, A. (2004), 'Numerical computation of rectangular bivariate and trivariate normal and t probabilities', *Statistics and Computing* **14**, 151–160. http://www.sci.wsu.edu/math/faculty/genz/homepage
- Hart, J. (1968), *Computer Approximations*, Wiley. Algorithm 5666 for the error function.
- Haug, E. G. (1998), *The complete guide to option pricing formulas*, McGraw-Hill.
- Hull, J. (2002), *Options, Futures, and Other Derivatives*, fifth edn, Prentice Hall.
- Johnson, H. (1987), 'Options on the maximum or the minimum of several assets', *Journal of Financial and Quantitative Analysis* **22**, 277–283.
- Schervish, M. (1984), 'Multivariate normal probabilities with error bound', *Applied Statistics* **33**, 81–87.
- Stulz, R. M. (1982), 'Options on the minimum or the maximum of two risky assets', *Journal of Financial Economics* **XXXIII, No. 1**, 161–185.
- West, G. (2004). http://www.cam.wits.ac.za/mfinance/graeme