



*The World's Largest Open Access Agricultural & Applied Economics Digital Library*

**This document is discoverable and free to researchers across the globe due to the work of AgEcon Search.**

**Help ensure our sustainability.**

**Give to AgEcon Search**

AgEcon Search

<http://ageconsearch.umn.edu>

[aesearch@umn.edu](mailto:aesearch@umn.edu)

*Papers downloaded from **AgEcon Search** may be used for non-commercial purposes and personal study only. No other use, including posting to another Internet site, is permitted without permission from the copyright owner (not AgEcon Search), or as allowed under the provisions of Fair Use, U.S. Copyright Act, Title 17 U.S.C.*

*No endorsement of AgEcon Search or its fundraising activities by the author(s) of the following work or their employer(s) is intended or implied.*

The Stata Journal (2018)  
18, Number 4, pp. 826–843

# Calculations involving the multivariate normal and multivariate $t$ distributions with and without truncation

Michael J. Grayling  
Hub for Trials Methodology Research  
MRC Biostatistics Unit  
Cambridge, UK  
mjg211@cam.ac.uk

Adrian P. Mander  
Hub for Trials Methodology Research  
MRC Biostatistics Unit  
Cambridge, UK  
adrian.mander@mrc-bsu.cam.ac.uk

**Abstract.** In this article, we present a set of commands and Mata functions to evaluate different distributional quantities of the multivariate normal distribution and a particular type of noncentral multivariate  $t$  distribution. Specifically, their densities, distribution functions, equicoordinate quantiles, and pseudo-random vectors can be computed efficiently, in either the absence or the presence of variable truncation.

**Keywords:** st0542, mvnrmalden, pmvnormal, invmvnormal, rmvnormal, mvt-den, mvt, invmvt, rmvt, tmvnormalden, tmvnormal, invtmvnormal, rtmvnormal, tmvtden, tmvt, invtmvt, rtmvt, multivariate normal, multivariate  $t$ , truncated distribution

## 1 Introduction

The normal distribution plays an important role in statistics because much collected data are or are assumed to be normally distributed. While the central limit theorem tells us that for any random sample  $X_1, \dots, X_n$  of independent and identically distributed random variables with expected value  $\mu$  and variance  $\sigma^2$ ,  $\sqrt{n}(\sum_{i=1}^n X_i/n - \mu)$  converges in distribution to  $N(0, \sigma^2)$  as  $n \rightarrow \infty$  (Patel and Read 1996). Moreover, the  $t$  distribution, and its use in cases of normally distributed data of small sample size and of unknown variance, is also important (Ireland 2010). Consequently, functions such as `normal()` and `t()` are available for working with these distributions in Stata.

However, a considerable amount of statistical analysis performed is not univariate but multivariate. As a result, the generalization of the normal and  $t$  distributions to higher dimensions—the multivariate normal (MVN) and multivariate  $t$  (MVT) distributions—is of increasing significance. Their use is seen through introductions created for statisticians (Tong 1990; Kotz, Balakrishnan, and Johnson 2004), natural scientists (Piegorisch and Bailer 1997; Blæsild and Granfeldt 2002; Mazza and Benaïm 2014; Feigelson and Babu 2012), and social scientists (Stevens 2016; Howell 2012).

The complexity of these distributions makes computational analysis almost certainly necessary; thus, much research into their numerical analysis has been conducted (Genz and Bretz 2009). Many programming languages now have efficient code avail-

able for working with densities, evaluating distribution functions, finding quantiles, and generating pseudo-random variables (see, for example, [Genz et al. \[2018\]](#)). However, in Stata no code currently exists for working with the MVT distribution, and there are limitations to the support for the MVN distribution. Specifically, the `drawnorm` command allows users to generate pseudo-random samples from an MVN distribution, and the `lnmvnormalden()` function evaluates its density. [Cappellari and Jenkins \(2006\)](#) provided `mvnp` for the evaluation of cumulative MVN probabilities of any dimension through use of the Geweke–Hajivassiliou–Keane simulator ([Geweke 1989](#); Hajivassiliou and McFadden 1998; [Keane 1994](#)). This simulator is also used in the Mata function `ghk()` ([Gates 2006](#)). In Stata 15, `mvnormal()` and several similar Mata commands were released to facilitate the computation of the MVN distribution function over any range of integration. However, to the best of our knowledge, no command is presently available for easily determining equicoordinate quantiles of the MVN distribution.

Furthermore, no commands are currently available to perform calculations involving the MVN or MVT distribution or their univariate counterparts in the presence of variable truncation. In certain scenarios, it is not uncommon for one or more variables in an MVN or MVT setting to be bound to some particular finite interval. For example, the truncated normal distribution is used in modeling censored data through the tobit model ([Tobin 1958](#)).

In this article, we present several commands and Mata functions for performing key calculations for any nondegenerate case of the MVN or MVT distribution and for a particular type of noncentral multivariate  $t$  (NCMVT) distribution in either the presence or the absence of variable truncation. We proceed by briefly summarizing the algorithms underpinning these commands, detailing their syntax, and finally demonstrating their use through several examples.

## 2 Methods

### 2.1 Multivariate normal distribution

We begin by considering a  $k$ -dimensional random variable  $\mathbf{X} = (X_1, \dots, X_k)^\top$  with a (nondegenerate) MVN distribution. We denote this by  $\mathbf{X} \sim N_k(\boldsymbol{\delta}, \boldsymbol{\Sigma})$ , where  $\boldsymbol{\delta} \in \mathbb{R}^k$  is a location parameter and  $\boldsymbol{\Sigma} \in \mathbb{M}^{k \times k}$  is a positive-definite covariance matrix. Many of the steps below revolve around the evaluation of  $\Phi_k(\mathbf{a}, \mathbf{b}, \boldsymbol{\delta}, \boldsymbol{\Sigma}) = \mathbb{P}(a_1 \leq X_1 \leq b_1, \dots, a_k \leq X_k \leq b_k)$ , for  $\mathbf{a} = (a_1, \dots, a_k)^\top$  and  $\mathbf{b} = (b_1, \dots, b_k)^\top$  with  $-\infty \leq a_i < b_i \leq \infty$  for  $i = 1, \dots, k$ . Specifically,

$$\begin{aligned} \Phi_k(\mathbf{a}, \mathbf{b}, \boldsymbol{\delta}, \boldsymbol{\Sigma}) &= \int_{a_1}^{b_1} \dots \int_{a_k}^{b_k} \phi_k(\mathbf{x}, \boldsymbol{\delta}, \boldsymbol{\Sigma}) \, d\mathbf{x} \\ &= \int_{a_1}^{b_1} \dots \int_{a_k}^{b_k} \frac{1}{\sqrt{(2\pi)^k |\boldsymbol{\Sigma}|}} \exp \left\{ -\frac{1}{2} (\mathbf{x} - \boldsymbol{\delta})^\top \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\delta}) \right\} \, d\mathbf{x} \end{aligned}$$

for  $\mathbf{x} = (x_1, \dots, x_k)^\top \in \mathbb{R}^k$  and where  $|\boldsymbol{\Sigma}| = \det(\boldsymbol{\Sigma})$ .

Several methods are available for evaluating such integrals. For example, the Mata function `mvnormalcv()` uses numerical quadrature for this task. In our code, we use a quasi-Monte Carlo integration algorithm over a randomized lattice after performing separation of variables (see [Genz \[1992\]](#); [Genz and Bretz \[2002; 2009\]](#) for further details). Moreover, we employ variable reordering to improve efficiency as proposed by [Gibson, Glasbey, and Elston \(1994\)](#). This is the approach recommended for such calculations by [Genz and Bretz \(2009\)](#). The procedure is implemented in our command `pmvnormal`. The algorithm used requires specification of the number of shifts and the number of samples to use in the Monte Carlo integration. It also needs a value for the Monte Carlo confidence factor. However, the default values have been shown to perform consistently well, and thus little user alteration is required in practice. Increasing the number of shifts or samples theoretically reduces the Monte Carlo error of the integration but also increases the run time of the commands.

In the above,  $\phi_k(\mathbf{x}, \boldsymbol{\delta}, \boldsymbol{\Sigma})$  is the probability density function for  $\mathbf{X}$ . We later present a command, `mvnormalden`, to directly compute the above density for any user-specified choices of  $\mathbf{x}$ ,  $\boldsymbol{\delta}$ , and  $\boldsymbol{\Sigma}$ .

Note that those with access to Stata 15 should use `mvnormalcv()` for evaluating  $\Phi_k(\mathbf{a}, \mathbf{b}, \boldsymbol{\delta}, \boldsymbol{\Sigma})$  because its execution time will be smaller. Additionally, as noted earlier, `lnmvnormalden()` could also be used for density calculations. The purpose of our commands is to provide a framework for these tasks with a consistent syntax to our commands performing previously unsupported derivations and a means of evaluating the MVN distribution function with earlier versions of Stata.

Determining equicoordinate quantiles of the MVN distribution is also routinely of interest in multivariate analysis. Determining the value,  $q$ , of the (100)*pth* equicoordinate quantile of  $X$  requires us to solve the following equation:

$$p = \Phi_k\{(-\infty, \dots, -\infty)^\top, (q, \dots, q)^\top, \boldsymbol{\delta}, \boldsymbol{\Sigma}\}$$

Here we achieve this with the command `invmvnormal`, allowing either `mvnormalcv()` or the algorithm implemented in `pmvnormal` to be used to evaluate the right-hand side of the above equation. We then use modified interval bisection to search for the correct value of  $q$ .

Finally, to draw pseudo-random vectors with distribution  $N_k(\boldsymbol{\delta}, \boldsymbol{\Sigma})$ , we use the fact that for  $\mathbf{R}$  with  $\mathbf{R}\mathbf{R}^\top = \boldsymbol{\Sigma}$ ,  $\mathbf{x} = \boldsymbol{\delta} + \mathbf{R}\mathbf{z}$  has the desired distribution when  $\mathbf{z} = (z_1, \dots, z_k)^\top$  with  $z_i \sim N(0, 1)$ . The  $z_i$  can be generated using `rnormal()`. Several methods for generating such an  $\mathbf{R}$  are available, as discussed for example by [Johnson \(1987\)](#). Here we allow for eigen, singular-value, and Cholesky decomposition of  $\boldsymbol{\Sigma}$ . Our command for this pseudo-random generation is `rmvnormal`, and `drawnorm` provides similar functionality for the density and distribution function evaluations discussed above. However, in this case, our command provides additional options and a consistent syntax.

## 2.2 MVT distribution

We now consider a  $k$ -dimensional random variable  $\mathbf{X} = (X_1, \dots, X_k)^\top$  with a (nondegenerate) NCMVT distribution. We denote this by  $\mathbf{X} \sim T_k(\boldsymbol{\delta}, \boldsymbol{\Sigma}, \nu)$ , where  $\boldsymbol{\delta} \in \mathbb{R}^k$  is now a vector of noncentrality parameters,  $\boldsymbol{\Sigma} \in \mathbb{M}^{k \times k}$  is a positive-definite scale matrix, and  $\nu \in \mathbb{R}^+$  is a degrees-of-freedom parameter. There are several ways by which one can define an associated NCMVT integral. Here we consider what has been referred to as the location-shifted NCMVT (Genz and Bretz 2009). See also Kotz and Nadarajah (2004) for further details. This NCMVT distribution arises, for example, as the Bayesian posterior distribution for the regression coefficients in a linear regression (Genz et al. 2018). Note that in the case when  $\boldsymbol{\delta} = \mathbf{0}$ , the central MVT distribution is obtained.

Our integral of interest is

$$\begin{aligned} \Psi_k(\mathbf{a}, \mathbf{b}, \boldsymbol{\delta}, \boldsymbol{\Sigma}, \nu) &= \int_{a_1}^{b_1} \dots \int_{a_k}^{b_k} \psi_k(\mathbf{x}, \boldsymbol{\delta}, \boldsymbol{\Sigma}, \nu) \, d\mathbf{x} \\ &= \int_{a_1}^{b_1} \dots \int_{a_k}^{b_k} \frac{\Gamma\{(\nu+k)/2\}}{\Gamma(\nu/2)\sqrt{(\nu\pi)^k|\boldsymbol{\Sigma}|}} \left\{ 1 + \frac{1}{\nu}(\mathbf{x} - \boldsymbol{\delta})^\top \boldsymbol{\Sigma}^{-1}(\mathbf{x} - \boldsymbol{\delta}) \right\}^{-(\nu+k)/2} d\mathbf{x} \end{aligned}$$

and can be evaluated using `mvt`. `mvtDen` can evaluate the density  $\psi_k(\mathbf{x}, \boldsymbol{\delta}, \boldsymbol{\Sigma}, \nu)$ .

We would again like to determine equicoordinate quantiles, this time solving the following equation:

$$p = \Psi_k\{(-\infty, \dots, -\infty)^\top, (q, \dots, q)^\top, \boldsymbol{\delta}, \boldsymbol{\Sigma}, \nu\}$$

Later, we describe the command `invmvt`, which performs this calculation. In both `mvt` and `invmvt`, the algorithm used for numerical integration is a direct modification of that described above for the MVN distribution.

Finally, to draw pseudo-random vectors following a location shifted  $T_k(\boldsymbol{\delta}, \boldsymbol{\Sigma}, \nu)$  distribution, we use the fact that if  $\mathbf{y} \sim N_k(\mathbf{0}, \boldsymbol{\Sigma})$  and  $v \sim \chi_\nu^2$ , then

$$\mathbf{x} = \boldsymbol{\delta} + \mathbf{y} \sqrt{\frac{\nu}{v}}$$

has the desired distribution. We draw pseudo-random vectors for  $\mathbf{y}$  using the algorithm described earlier for the MVN distribution, with pseudo-random numbers  $v$  acquired through `rchi2()`. The command for this is `rmvt`.

## 2.3 Truncated MVN and MVT distributions

We additionally consider truncated versions of the MVN and NCMVT distributions discussed above. We denote these by  $\mathbf{X} \sim TN_k(\boldsymbol{\delta}, \boldsymbol{\Sigma}, \mathbf{l}, \mathbf{u})$  and  $\mathbf{X} \sim TT_k(\boldsymbol{\delta}, \boldsymbol{\Sigma}, \nu, \mathbf{l}, \mathbf{u})$ , respectively. Here  $\mathbf{l} = (l_1, \dots, l_k)^\top \in \mathbb{R}^k$  and  $\mathbf{u} = (u_1, \dots, u_k)^\top \in \mathbb{R}^k$ , with  $l_i < u_i$  for  $i = 1, \dots, k$ , are the lower and upper truncation points for  $\mathbf{X}$ . That is, we restrict such that  $l_i \leq X_i \leq u_i$  for  $i = 1, \dots, k$ . The distribution functions for these variables are

$$\begin{aligned}
\Phi'_k(\mathbf{a}, \mathbf{b}, \boldsymbol{\delta}, \boldsymbol{\Sigma}, \mathbf{l}, \mathbf{u}) &= \int_{\max(a_1, l_1)}^{\min(b_1, u_1)} \cdots \int_{\max(a_k, l_k)}^{\min(b_k, u_k)} \phi'_k(\mathbf{x}, \boldsymbol{\delta}, \boldsymbol{\Sigma}, \mathbf{l}, \mathbf{u}) \, d\mathbf{x} \\
&= \int_{\max(a_1, l_1)}^{\min(b_1, u_1)} \cdots \int_{\max(a_k, l_k)}^{\min(b_k, u_k)} \frac{\phi_k(\mathbf{x}, \boldsymbol{\delta}, \boldsymbol{\Sigma})}{\Phi_k(\mathbf{l}, \mathbf{u}, \boldsymbol{\delta}, \boldsymbol{\Sigma})} \, d\mathbf{x} \\
\Psi'_k(\mathbf{a}, \mathbf{b}, \boldsymbol{\delta}, \boldsymbol{\Sigma}, \nu, \mathbf{l}, \mathbf{u}) &= \int_{\max(a_1, l_1)}^{\min(b_1, u_1)} \cdots \int_{\max(a_k, l_k)}^{\min(b_k, u_k)} \psi'_k(\mathbf{x}, \boldsymbol{\delta}, \boldsymbol{\Sigma}, \nu, \mathbf{l}, \mathbf{u}) \, d\mathbf{x} \\
&= \int_{\max(a_1, l_1)}^{\min(b_1, u_1)} \cdots \int_{\max(a_k, l_k)}^{\min(b_k, u_k)} \frac{\psi_k(\mathbf{x}, \boldsymbol{\delta}, \boldsymbol{\Sigma}, \nu)}{\Psi_k(\mathbf{l}, \mathbf{u}, \boldsymbol{\delta}, \boldsymbol{\Sigma}, \nu)} \, d\mathbf{x}
\end{aligned}$$

with  $\phi'_k(\mathbf{x}, \boldsymbol{\delta}, \boldsymbol{\Sigma}, \mathbf{l}, \mathbf{u})$  and  $\psi'_k(\mathbf{x}, \boldsymbol{\delta}, \boldsymbol{\Sigma}, \nu, \mathbf{l}, \mathbf{u})$  being their respective densities. Our commands to evaluate these quantities are `tmvnormal`, `tmvt`, `tmvnormalden`, and `tmvtiden`.

We further provide the commands `invtmvnormal` and `invtmvt` to determine the values of  $q$  such that

$$p = \Phi'_k\{(-\infty, \dots, -\infty)^\top, (q, \dots, q)^\top, \boldsymbol{\delta}, \boldsymbol{\Sigma}, \mathbf{l}, \mathbf{u}\}$$

or

$$p = \Psi'_k\{(-\infty, \dots, -\infty)^\top, (q, \dots, q)^\top, \boldsymbol{\delta}, \boldsymbol{\Sigma}, \nu, \mathbf{l}, \mathbf{u}\}$$

Finally, we use rejection sampling in the commands `rtmvnormal` and `rtmvt` to generate random variables with a  $TN_k(\boldsymbol{\delta}, \boldsymbol{\Sigma}, \mathbf{l}, \mathbf{u})$  or  $TT_k(\boldsymbol{\delta}, \boldsymbol{\Sigma}, \nu, \mathbf{l}, \mathbf{u})$  distribution. For example, in `rtmvnormal`, we make a random draw from  $N_k(\boldsymbol{\delta}, \boldsymbol{\Sigma})$ . If  $l_i \leq X_i \leq u_i$  for  $i = 1, \dots, k$ , then we retain this draw; otherwise, we reject it and repeat the process.

### 3 Commands

Note that all the commands presented here are written as ado-files, using Mata for computational efficiency wherever possible. This allows all desired calculations to be performed easily from within Stata. We conveniently provide corresponding Mata functions for each of the commands given the matrix-based nature of this work.

#### 3.1 Syntax

`mvnormalden`, `x(numlist)` `mean(numlist)` `sigma(string)` [`logdensity`]

`pmvnormal`, `lower(numlist)` `upper(numlist)` `mean(numlist)` `sigma(string)`  
 [`shifts(#)` `samples(#)` `alpha(real)`]

`invmvnormal`, `p(real)` `mean(numlist)` `sigma(string)` [`tail(string)` `itermax(#)`]  
`tolerance(real)` `integrator(string)` `shifts(#)` `samples(#)`]

```

rmvnormal, mean(numlist) sigma(string) [ n(#) method(string) ]

mvtdden, x(numlist) delta(numlist) sigma(string) [ df(real) logdensity ]

mvt, lower(numlist) upper(numlist) delta(numlist) sigma(string) [ df(real)
shifts(#) samples(#) alpha(real) ]

invmvt, p(real) delta(numlist) sigma(string) [ df(real) tail(string)
itermax(#) tolerance(real) shifts(#) samples(#) ]

rmvt, delta(numlist) sigma(string) [ df(real) n(#) method(string) ]

tmvnormalden, x(numlist) mean(numlist) sigma(string)
lowertruncation(numlist) uppertruncation(numlist) [ logdensity
integrator(string) shifts(#) samples(#) ]

tmvnormal, lower(numlist) upper(numlist) mean(numlist) sigma(string)
lowertruncation(numlist) uppertruncation(numlist) [ integrator(string)
shifts(#) samples(#) ]

invtmvnormal, p(real) mean(numlist) sigma(string) lowertruncation(numlist)
uppertruncation(numlist) [ tail(string) itermax(#) tolerance(real)
integrator(string) shifts(#) samples(#) ]

rtmvnormal, mean(numlist) sigma(string) lowertruncation(numlist)
uppertruncation(numlist) [ n(#) method(string) ]

tmvtdden, x(numlist) delta(numlist) sigma(string) lowertruncation(numlist)
uppertruncation(numlist) [ df(real) logdensity shifts(#) samples(#) ]

tmvt, lower(numlist) upper(numlist) delta(numlist) sigma(string)
lowertruncation(numlist) uppertruncation(numlist) [ df(real) shifts(#)
samples(#) ]

```

```
invtmvt, p(real) delta(numlist) sigma(string) lowertruncation(numlist)
uppertruncation(numlist) [df(real) tail(string) itermax(#)
tolerance(real) shifts(#) samples(#)]
```

```
rtmvt, delta(numlist) sigma(string) lowertruncation(numlist)
uppertruncation(numlist) [df(real) n(#) method(string)]
```

### 3.2 Options

The above options are as follows:

**x**(*numlist*) specifies the vector of quantiles at which a density is sought (see *x* above). **x**() is required.

**p**(*real*) specifies a desired probability, denoted by *p* above. *real* must be strictly between 0 and 1. **p**() is required.

**delta**(*numlist*) specifies the vector of noncentrality parameters  $\delta$  of an NCMVT or a truncated NCMVT distribution. Note that this may not be these distributions' expected value. **delta**() is required.

**lower**(*numlist*) specifies the vector of lower limits, denoted by *a* above. Use . to indicate a value is  $-\infty$ . **lower**() is required.

**upper**(*numlist*) specifies the vector of upper limits, denoted by *b* above. Use . to indicate a value is  $+\infty$ . **upper**() is required.

**mean**(*numlist*) specifies the location parameter  $\delta$  of an MVN or a truncated MVN distribution. In the case of an MVN distribution, this is the expected value of the distribution. **mean**() is required.

**sigma**(*string*) specifies the matrix  $\Sigma$  described above. This must be symmetric positive definite. In the case of an MVN distribution, this is the variables covariance matrix. This is not the case for the NCMVT distribution or for truncated distributions. **sigma**() is required.

**lowertruncation**(*numlist*) specifies the vector of lower truncation limits denoted by *l* above. Use . to indicate a value is  $-\infty$ . **lowertruncation**() is required.

**uppertruncation**(*numlist*) specifies the vector of upper truncation limits denoted *u* above. Use . to indicate a value is  $+\infty$ . **uppertruncation**() is required.

**logdensity** specifies that the log of the evaluated density should be returned.

**df**(*real*) specifies the degrees-of-freedom parameter,  $\nu$ , of an NCMVT or a truncated NCMVT distribution. The default is **df**(1).



`tail(string)` specifies which quantile should be computed. For `invmvnormal`, `lower` gives the  $q$  such that

$$p = \Phi_k\{(-\infty, \dots, -\infty)^\top, (q, \dots, q)^\top, \boldsymbol{\delta}, \boldsymbol{\Sigma}\}$$

upper such that

$$p = \Phi_k\{(q, \dots, q)^\top, (\infty, \dots, \infty)^\top, \boldsymbol{\delta}, \boldsymbol{\Sigma}\}$$

and both such that

$$p = \Phi_k\{(-q, \dots, -q)^\top, (q, \dots, q)^\top, \boldsymbol{\delta}, \boldsymbol{\Sigma}\}$$

Analogous statements hold for `invmvt`, `invtmvnormal`, and `invtmvt`.

`itermax(#)` specifies the maximum allowed number of iterations in the implemented modified interval bisection algorithm. `#` must be strictly positive. The default is `itermax(1000000)`.

`tolerance(real)` specifies the desired tolerance for termination of the implemented modified interval bisection algorithm. `#` must be strictly positive. The default is `tolerance(0.000001)`.

`integrator(string)` specifies the method for evaluating required MVN integrals. `string` must be either `pmvnormal` (to use `pmvnormal.ado`) or `mvnormal` (to use `mvnormal()` or `mvnormalcv()` as released in Stata 15). The default is `integrator(mvnormal)`.

`n(#)` specifies the number of random vectors to generate from a chosen distribution. `#` must be a strictly positive integer. The default is `n(1)`.

`method(string)` specifies the method to use for generating matrix  $\mathbf{R}$  with  $\mathbf{R}\mathbf{R}' = \boldsymbol{\Sigma}$ . `string` must be `eigen`, `svd`, or `chol`.

`shifts(#)` specifies the number of shifts of the quasi-Monte Carlo integration algorithm to use. `#` must be a strictly positive integer. In `invmvnormal`, `invtmvnormal`, `tmvnormalden`, and `tmvnormal`, this will have an effect only if `integrator()` is set to `pmvnormal`. The default is `shifts(12)`.

`samples(#)` specifies the number of samples in each shift of the quasi-Monte Carlo integration algorithm to use. `#` must be a strictly positive integer. As above, in `invmvnormal`, `invtmvnormal`, `tmvnormalden`, and `tmvnormal`, this will have an effect only if `integrator()` is set to `pmvnormal`. The default is `samples(1000)`.

`alpha(real)` specifies the value of the Monte Carlo confidence factor to use in estimating the error in the returned value of the integral when using `pmvnormal` or `mvt`. `real` must be strictly positive. The default is `alpha(3)`.

### 3.3 Mata functions

Mata functions are provided with a naming convention that adds the suffix ‘`_mata`’ to the command names above. All inputs are listed in the same order as the commands

and have the exact same name and interpretation. Input types are specified in the obvious manner. That is, `delta()` and `lower()` must be real vectors, `df()` a real scalar, `method()` a string, and similarly for other inputs. Thus, we omit their specification here. However, as an example,

```
mvtden.mata(real vector x, real vector delta, real matrix Sigma, real
  scalar df, string logdensity)
```

## 4 Examples

### 4.1 Density comparison

To demonstrate the use of `mvnnormalden`, `mvtden`, `tmvnnormalden`, and `tmvtden`, we plot and compare densities in the case with

$$\delta = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \quad \Sigma = \begin{pmatrix} 1 & 0.5 \\ 0.5 & 1 \end{pmatrix}, \quad \nu = 1, \quad l = \begin{pmatrix} -1.5 \\ -1.5 \end{pmatrix}, \quad u = \begin{pmatrix} 1.5 \\ 1.5 \end{pmatrix}$$

We first loop over a grid of values in the region  $[-3, 3]^2$ . At each point, we compute the density of an MVN, an MVT, a truncated MVN, and a truncated MVT distribution with the above parameters:

```
. matrix Sigma      = J(2, 2, 0.5) + 0.5*I(2)
. matrix mvnnormalden = J(101, 101, 0)
. matrix mvtden      = J(101, 101, 0)
. matrix tmvnnormalden = J(101, 101, 0)
. matrix tmvtden      = J(101, 101, 0)
. local i = 1
. foreach x1 of numlist -3(0.06)3 {
2.   local j = 1
3.   foreach x2 of numlist -3(0.06)3 {
4.     quietly mvnnormalden, x(`x1' `x2') mean(0, 0) sigma(Sigma)
5.     matrix mvnnormalden[`i' `j'] = r(density)
6.     quietly mvtden, x(`x1' `x2') delta(0, 0) sigma(Sigma) df(1)
7.     matrix mvtden[`i' `j'] = r(density)
8.     quietly tmvnnormalden, x(`x1' `x2') mean(0, 0) sigma(Sigma)
> lowertruncation(-1.5, -1.5) uppertruncation(1.5, 1.5)
9.     matrix tmvnnormalden[`i' `j'] = r(density)
10.    quietly tmvtden, x(`x1' `x2') delta(0, 0) sigma(Sigma)
> lowertruncation(-1.5, -1.5) uppertruncation(1.5, 1.5) df(1)
11.    matrix tmvtden[`i' `j'] = r(density)
12.    local `j++'
13.  }
14.  local `i++'
15. }
```

We then pass our results to `twoway contour` to produce figure 1:

```
. quietly svmat mvnnormalden
. quietly svmat mvtden
. quietly svmat tmvnormalden
. quietly svmat tmvtden
. generate row = _n
. quietly reshape long mvnnormalden mvtden tmvnormalden tmvtden, i(row) j(col)
. local opt " aspect(2) ccuts(0 0.03 0.06 0.09 0.12 0.15 0.18 0.21)
> xtitle("x1") ytitle("x2")"
. twoway contour mvnnormalden row col, nodraw saving(g1,replace) `opt`
(file g1.gph saved)
. twoway contour mvtden row col, nodraw saving(g2,replace) `opt`
(file g2.gph saved)
. twoway contour tmvnormalden row col, nodraw saving(g3,replace) `opt`
(file g3.gph saved)
. twoway contour tmvtden row col, nodraw saving(g4,replace) `opt`
(file g4.gph saved)
. graph combine g1.gph g2.gph g3.gph g4.gph, common imargin(0 0 0 0) rows(2)
> cols(2)
```

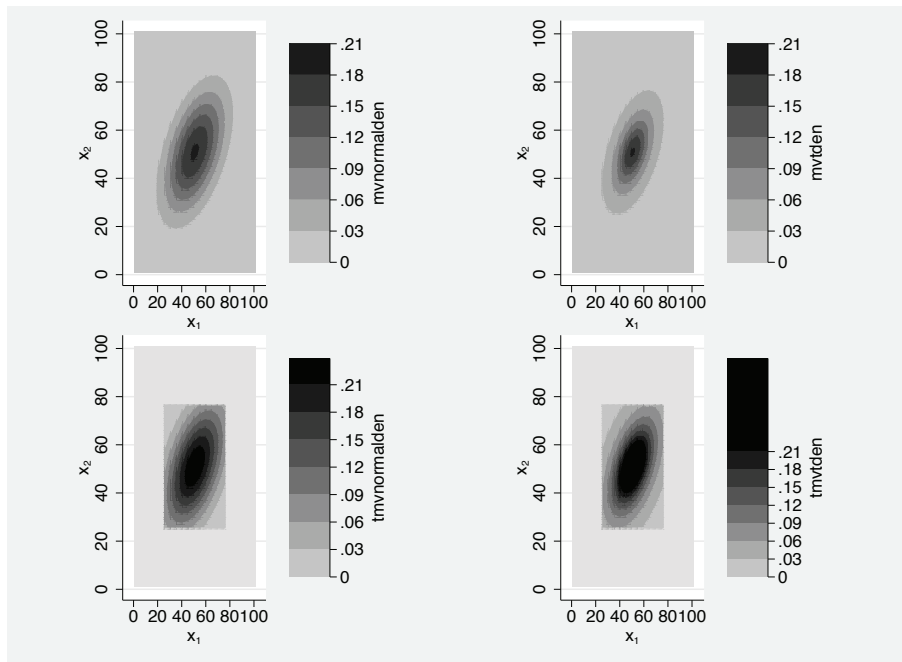


Figure 1. Densities of an MVN (`mvnnormalden`), an MVT (`mvtden`), a truncated MVN (`tmvnormalden`), and a truncated MVT distribution (`tmvtden`) are shown

As expected, we observe that the density of the MVT distribution with  $\nu = 1$  degree of freedom is more homogeneous than that of the MVN distribution. This explains

why tail probabilities are larger for MVT distributions with low degrees of freedom. Truncation increases the density at each point in the region  $[-1.5, 1.5]^2$  compared with the corresponding nontruncated variable.

## 4.2 Goodness-of-fit tests for multivariate normality

Stata provides, through `mvtest normality`, a set of statistical tests of multivariate normality. Here we explore the results of applying these tests to random variables drawn using `rmvt` and `rtmvt`. Specifically, we consider a bivariate MVT distribution with

$$\delta = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \quad \Sigma = \begin{pmatrix} 1 & 0.5 \\ 0.5 & 1 \end{pmatrix}$$

We also examine results for a variable that is a mixture of two truncated MVT distributions, each with the above value for  $\Sigma$ , but one with

$$\delta = \begin{pmatrix} -2 \\ -2 \end{pmatrix}, \quad l = \begin{pmatrix} -2.5 \\ -2.5 \end{pmatrix}, \quad u = \begin{pmatrix} -1.5 \\ -1.5 \end{pmatrix}$$

and the other

$$\delta = \begin{pmatrix} 2 \\ 2 \end{pmatrix}, \quad l = \begin{pmatrix} 1.5 \\ 1.5 \end{pmatrix}, \quad u = \begin{pmatrix} 2.5 \\ 2.5 \end{pmatrix}$$

We additionally consider several possible values for  $\nu$ . We draw  $n$  samples from the MVT distribution, for multiple values of  $n$ , 1,000 times to compute an average  $p$ -value from the Doornik–Hansen omnibus test. This process is also performed for the truncated MVT distributions, except  $n/2$  samples are drawn from each and then combined into a single dataset, so the overall sample size is retained at  $n$ . We achieve this with the following code:

```
. set seed 2
. matrix Sigma = J(2, 2, 0.5) + 0.5*I(2)
. matrix gof_pvaluesum_mvn = J(4, 5, 0)
. matrix gof_pvaluesum_mix = J(4, 5, 0)
. local i = 1
. foreach n of numlist 10 25 50 100 {
2.   local j = 1
3.   foreach df of numlist 2 5 10 50 100 {
4.     foreach rep of numlist 1/1000 {
5.       rmvt, delta(0, 0) sigma(Sigma) df(`df') n(`n') method(chol)
6.       matrix randsamp = r(rmvt)
7.       quietly svmat randsamp
8.       quietly mvtest normality randsamp1 randsamp2
9.       matrix gof_pvaluesum_mvn[`i', `j'] = gof_pvaluesum_mvn[`i', `j']
> + r(p_dh)
10.      drop randsamp1 randsamp2
11.      rtmvt, delta(-2, -2) sigma(Sigma) df(`df') n(`n') method(chol)
> lowertruncation(-2.5, -2.5) uppertruncation(-1.5, -1.5)
12.      matrix randsamp = r(rtmvt)
13.      rtmvt, delta(2, 2) sigma(Sigma) df(`df') n(`n') method(chol)
> lowertruncation(1.5, 1.5) uppertruncation(2.5, 2.5)
```

```

14.          matrix randsamp = (randsamp \ r(rtmvt))
15.          quietly svmat randsamp
16.          quietly mvtest normality randsamp1 randsamp2
17.          matrix gof_pvaluesum_mix[`i`, `j`] = gof_pvaluesum_mix[`i`, `j`]
> + r(p_dh)
18.          drop randsamp1 randsamp2
19.      }
20.      local `j++`
21.  }
22.  local `i++`
23. }

. matrix gof_summary_mvn = gof_pvaluesum_mvn/1000
. matrix gof_summary_mix = gof_pvaluesum_mix/1000

```

Observe that we have used Cholesky decomposition of  $\Sigma$  in generating these random variables. For most distributions, the choice of decomposition will have little effect on the speed and quality of the generated random variables. However, when the dimension  $k$  is large, this choice should be made more carefully.

The results of the Doornik–Hansen tests are summarized in table 1. We can see that, as expected, increasing the value of  $\nu$  for the MVT distribution typically increases the average  $p$ -value for each value of  $n$  because the resultant distribution is closer to that of a corresponding MVN distribution. While the pattern is slightly less clear as we increase  $n$ , this generally reduces the average  $p$ -value because more data allow the test to more precisely categorize the distribution of the generated variables.

Table 1. Average *p*-values from the Doornik–Hansen omnibus test across 1,000 replicate generated datasets are shown as a function of the sample size *n* and the degrees of freedom  $\nu$ . Results are displayed for a particular MVT distribution and a variable that is a mixture of two truncated MVT distributions as described in the text.

Mixture of MVT distributions					
	$\nu = 2$	$\nu = 5$	$\nu = 10$	$\nu = 50$	$\nu = 100$
$n = 10$	0.2948	0.4069	0.4540	0.4851	0.4882
$n = 25$	0.1040	0.3200	0.4217	0.5066	0.5109
$n = 50$	0.0063	0.1502	0.3281	0.5065	0.5090
$n = 100$	0.0003	0.0574	0.2326	0.4656	0.5046

Mixture of truncated NCMVT distributions					
	$\nu = 2$	$\nu = 5$	$\nu = 10$	$\nu = 50$	$\nu = 100$
$n = 10$	0.6612	0.6502	0.6398	0.6402	0.6483
$n = 25$	0.6634	0.6466	0.6377	0.6138	0.6053
$n = 50$	0.4281	0.4581	0.4720	0.4712	0.4446
$n = 100$	0.1439	0.1942	0.2300	0.2443	0.2406

For the datasets that are a mixture of the truncated NCMVT variables, the average *p*-values are often larger than the corresponding value for the MVT distribution. In no instance is the Doornik–Hansen omnibus test significant on average at the 5% level. That is, in no case does it on average correctly reject the null hypothesis of multivariate normality. This is likely due to the chosen values for  $\delta$ ,  $\mathbf{l}$ , and  $\mathbf{u}$  in the two truncated MVT distributions. When the two simulated datasets are combined, they likely resemble an MVN distribution with  $\delta = \mathbf{0}$ .

### 4.3 Familywise error rate using Bonferroni and Dunnett corrections

As a brief example of the usage of `pmvnormal` and `invmvnormal`, consider a statistical test of the following hypotheses:

$$H_{0i}: \mu_i \leq 0, \quad H_{1i}: \mu_i > 0 \quad i = 1, 2, 3$$

Suppose our tests are based on test statistics  $Z_i$  and  $i = 1, 2, 3$  and it is known that

$$(Z_1, Z_2, Z_3)^\top \sim N_3 \left\{ \begin{pmatrix} \mu_1 \\ \mu_2 \\ \mu_3 \end{pmatrix}, \begin{pmatrix} 1 & 0.5 & 0.5 \\ 0.5 & 1 & 0.5 \\ 0.5 & 0.5 & 1 \end{pmatrix} \right\}$$

The familywise error rate (the probability of incorrectly rejecting one or more of the null hypotheses) can be controlled to a level  $\alpha$  using the Bonferroni correction by rejecting

$H_{0i}$  if  $Z_i > \Phi_1^{-1}(1 - \alpha/3)$ . The Dunnett correction in comparison rejects if  $Z_i > r$ , where  $r$  is the solution to

$$1 - \alpha = \Phi_3 \left\{ (-\infty, \dots, -\infty)^\top, (r, \dots, r)^\top, \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 & 0.5 & 0.5 \\ 0.5 & 1 & 0.5 \\ 0.5 & 0.5 & 1 \end{pmatrix} \right\}$$

We can use `invmvnormal` to determine this value of  $r$  with the following code:

```
. set seed 3
. matrix Sigma = J(3, 3, 0.5) + 0.5*I(3)
. invmvnormal, p(0.95) mean(0, 0, 0) sigma(Sigma) tail("lower")
quantile = 2.0620839
error = 4.663e-15
flag = 0
fquantile = 0
iterations = 21
. local r = r(quantile)
```

This returns an estimate of the error in this value and also returns the quantile, which is determined to be approximately 2.06. It also provides a flag variable, which takes the value 0 if the interval bisection algorithm converged with no issues. It also lists a value of the objective function (`fquantile`) that we attempt to find the root of and the number of iterations required for convergence.

We can then verify that this value of  $r$  will control the familywise error rate to  $\alpha$  using `pmvnormal` as follows, additionally evaluating the true familywise error rate when using the Bonferroni correction:

```
. pmvnormal, lower(., ., .) upper(`r`, `r`, `r`) mean(0, 0, 0) sigma(Sigma)
integral = .94999707
error = .00005633
. pmvnormal, lower(., ., .) upper(`= invnormal(1 - 0.05/3)`,
> `= invnormal(1 - 0.05/3)`, `= invnormal(1 - 0.05/3)`) mean(0, 0, 0)
> sigma(Sigma)
integral = .95705901
error = .00003466
```

We can now see the conservatism of the Bonferroni correction: the true familywise error rate is approximately 4.3%.

#### 4.4 Orthont probabilities in the presence of truncation

The value of

$$\Phi_k\{(0, \dots, 0)^\top, (\infty, \dots, \infty)^\top, \boldsymbol{\delta}, \boldsymbol{\Sigma}\}$$

is often referred to as an orthont probability. Here, as a final example, we examine how  $\Phi'_k\{(0, \dots, 0)^\top, (\infty, \dots, \infty)^\top, \boldsymbol{\delta}, \boldsymbol{\Sigma}, (t, \dots, t)^\top, (\infty, \dots, \infty)^\top\}$  changes in  $t$  to demonstrate the usage of `tmvnormal`.

We consider again the case with

$$\boldsymbol{\delta} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}, \quad \boldsymbol{\Sigma} = \begin{pmatrix} 1 & 0.5 & 0.5 \\ 0.5 & 1 & 0.5 \\ 0.5 & 0.5 & 1 \end{pmatrix}$$

Using `pmvnormal`, we can establish that  $\Phi_3\{(0, \dots, 0)^\top, (\infty, \dots, \infty)^\top, \boldsymbol{\delta}, \boldsymbol{\Sigma}\} \approx 0.25$ .

```
. set seed 4
. matrix Sigma = J(3, 3, 0.5) + 0.5*I(3)
. pmvnormal, lower(0, 0, 0) upper(., ., .) mean(0, 0, 0) sigma(Sigma)
integral = .2500246
error = .00005905
```

The second value returned above is, as in section 4.3, an estimate of the error in the integral. As we can see, `pmvnormal` is again easily able to control this error to a small level.

Then, as stated, we can evaluate how this probability changes in  $t$  for a truncated MVN distribution as follows:

```
. matrix lowertrunc = J(110, 1, 0)
. matrix probabilities = J(110, 1, 0)
. foreach i of numlist 1/110 {
2.   matrix lowertrunc[`i', 1] = -10 + 0.1*(`i' - 1)
3.   quietly tmvnormal, lower(0, 0, 0) upper(., ., .) mean(0, 0, 0) sigma(Sigma)
> lowertruncation(`=lowertrunc[`i', 1]', `=lowertrunc[`i', 1]',
> `=lowertrunc[`i', 1]') uppertruncation(., ., .)
4.   matrix probabilities[`i', 1] = r(integral)
5. }
. matrix data = (lowertrunc, probabilities)
. quietly svmat data
. twoway (line data2 data1, yaxis(1)), xtitle(t) ytitle(P(X >= (0,0,0)|(t,t,t)))
> scheme(sj)
```

Here `twoway line` is used to produce figure 2. As expected, when  $t$  is highly negative, the value of  $\Phi'_3\{(0, \dots, 0)^\top, (\infty, \dots, \infty)^\top, \boldsymbol{\delta}, \boldsymbol{\Sigma}, (t, \dots, t)^\top, (\infty, \dots, \infty)^\top\}$  is approximately equal to that of the corresponding nontruncated distribution. This value increases up to a value of 1 when  $t \geq 0$  because at this point, truncation implies we must have that  $X_i \geq 0$  for  $i = 1, 2, 3$ .



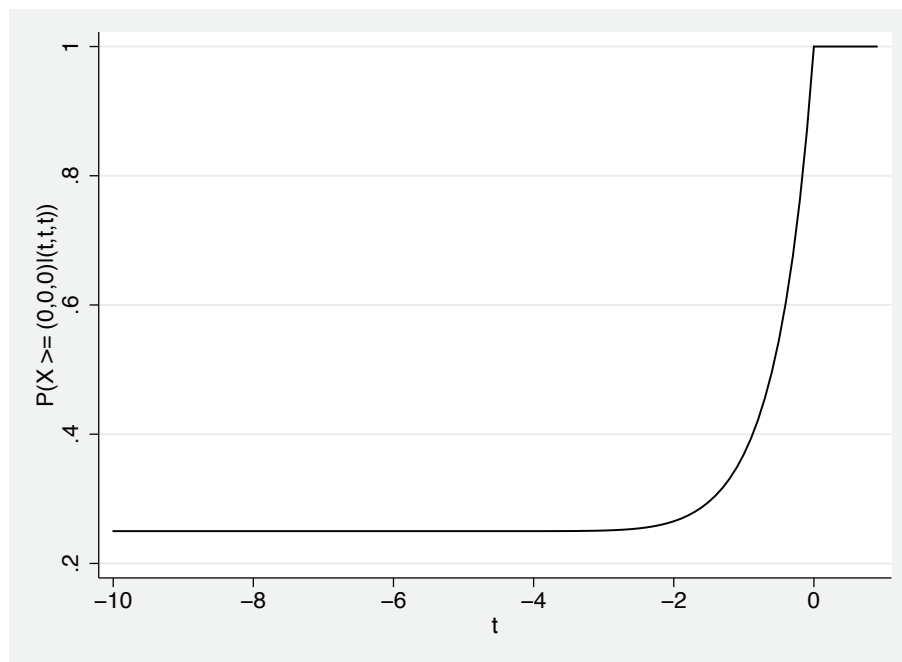


Figure 2.  $\Phi'_3\{(t, \dots, t)^\top\} \equiv \Phi'_3\{(0, \dots, 0)^\top, (\infty, \dots, \infty)^\top, \boldsymbol{\delta}, \boldsymbol{\Sigma}, (t, \dots, t)^\top, (\infty, \dots, \infty)^\top\}$  is shown as a function of  $t$  for a particular truncated MVN distribution

## 5 Conclusion

The MVN and MVT distributions are extremely important in many statistical problems faced by researchers in many fields. Here we have extended Stata users' ability to work with these key distributions; we presented several commands to allow densities, pseudo-random draws, distribution functions, and equicoordinate quantiles to be computed quickly and efficiently, both with and without variable truncation. In particular, using algorithms developed by [Genz and Bretz \(2009\)](#), our commands require little input from the user to determine probabilities that the distributions fall in any range of integration. While this is possible from the MVN distribution using `mvnnormalcv()`, our commands are the first for the NCMVT distribution and for truncated MVN and NCMVT distributions.

However, while the algorithms used here for the computation of probabilities over arbitrary ranges of integration are efficient, many alternatives are available. Moreover, we have considered only one possible definition of an NCMVT distribution. Another important definition, used in the computation of the power of multiple contrast tests under a normality assumption ([Genz et al. 2018](#)), was proposed by Kshirsagar ([Kotz and Nadarajah 2004](#)). Thus, further development of the functions presented here will seek to make additional algorithms available as an option in the distribution function commands and to include an option to change between the shifted NCMVT considered here and that of Kshirsagar, available in the NCMVT and truncated NCMVT commands.

## 6 Acknowledgments

Michael J. Grayling is supported by the Wellcome Trust (grant number 099770/Z/12/Z), and Adrian P. Mander is supported by the Medical Research Council (grant number MC\_UP\_1302/2).

## 7 References

- Blæsild, P., and J. Granfeldt. 2002. *Statistics with Applications in Biology and Geology*. Boca Raton, FL: Chapman & Hall/CRC.
- Cappellari, L., and S. P. Jenkins. 2006. Calculation of multivariate normal probabilities by simulation, with applications to maximum simulated likelihood estimation. *Stata Journal* 6: 156–189.
- Feigelson, E. D., and G. J. Babu. 2012. *Modern Statistical Methods for Astronomy: With R Applications*. Cambridge: Cambridge University Press.
- Gates, R. 2006. A Mata Geweke–Hajivassiliou–Keane multivariate normal simulator. *Stata Journal* 6: 190–213.
- Genz, A. 1992. Numerical computation of multivariate normal probabilities. *Journal of Computational and Graphical Statistics* 1: 141–149.
- Genz, A., and F. Bretz. 2002. Comparison of methods for the computation of multivariate  $t$  probabilities. *Journal of Computational and Graphical Statistics* 11: 950–971.
- . 2009. *Computation of Multivariate Normal and  $t$  Probabilities*. Berlin: Springer.
- Genz, A., F. Bretz, T. Miwa, X. Mi, F. Leisch, F. Scheipl, B. Bornkamp, M. Maechler, and T. Hothorn. 2018. *mvtnorm: Multivariate normal and  $t$  distributions*. R package version 1.0-8. <https://cran.r-project.org/web/packages/mvtnorm/>.
- Geweke, J. 1989. Bayesian inference in econometric models using Monte Carlo integration. *Econometrica* 57: 1317–1339.
- Gibson, G. J., C. A. Glasbey, and D. A. Elston. 1994. Monte Carlo evaluation of multivariate normal integrals and sensitivity to variate ordering. In *Advances in Numerical Methods and Applications: Proceedings of the Third International Conference*, ed. I. T. Dimov, B. Sendov, and P. S. Vassilevski, 120–126. River Edge, NJ: World Scientific.
- Hajivassiliou, V. A., and D. L. McFadden. 1998. The method of simulated scores for the estimation of LDV models. *Econometrica* 66: 863–896.
- Howell, D. C. 2012. *Statistical Methods for Psychology*. 8th ed. Belmont, CA: Wadsworth.

- Ireland, C. R. 2010. *Experimental Statistics for Agriculture and Horticulture*. Wallingford, UK: CABI.
- Johnson, M. E. 1987. *Multivariate Statistical Simulation: A Guide to Selecting and Generating Continuous Multivariate Distributions*. Chichester, UK: Wiley.
- Keane, M. P. 1994. A computationally practical simulation estimator for panel data. *Econometrica* 62: 95–116.
- Kotz, S., N. Balakrishnan, and N. L. Johnson. 2004. *Continuous Multivariate Distributions. Volume 1: Models and Applications*. 2nd ed. New York: Wiley.
- Kotz, S., and S. Nadarajah. 2004. *Multivariate  $t$  Distributions and Their Applications*. Cambridge: Cambridge University Press.
- Mazza, C., and M. Benaïm. 2014. *Stochastic Dynamics for Systems Biology*. Boca Raton, FL: Chapman & Hall/CRC.
- Patel, J. K., and C. B. Read. 1996. *Handbook of the Normal Distribution*. Revised and expanded second ed. New York: Marcel Dekker.
- Piegorsch, W. W., and A. J. Bailer. 1997. *Statistics for Environmental Biology and Toxicology*. London: Chapman & Hall/CRC.
- Stevens, J. P. 2016. *Applied Multivariate Statistics for the Social Sciences: Analyses with SAS and IBMs SPSS*. 6th ed. New York: Routledge.
- Tobin, J. 1958. Estimation of relationships for limited dependent variables. *Econometrica* 26: 24–36.
- Tong, Y. L. 1990. *The Multivariate Normal Distribution*. New York: Springer.

**About the authors**

Michael J. Grayling is an investigator statistician at the Medical Research Council Biostatistics Unit in Cambridge, UK.

Adrian P. Mander is the Director of the Hub for Trial Methodology Research at the Medical Research Council Biostatistics Unit in Cambridge, UK.