Methods for the Computation of Multivariate t-Probabilities *

Alan Genz †
Department of Mathematics
Washington State University
Pullman, WA 99164-3113
alangenz@wsu.edu

Frank Bretz
LG Bioinformatik
University of Hannover
Hannover, Germany
bretz@ifgb.uni-hannover.de

Abstract

This paper compares methods for the numerical computation of multivariate t-probabilities for hyper-rectangular integration regions. Methods based on acceptance-rejection, spherical-radial transformations and separation-of-variables transformations are considered. Tests using randomly chosen problems show that the most efficient numerical methods use a transformation developed by Genz (1992) for multivariate normal probabilities. These methods allow moderately accurate multivariate t-probabilities to be quickly computed for problems with as many as twenty variables. Methods for the non-central multivariate t-distribution are also described.

Key Words: multivariate t-distribution, non-central distribution, numerical integration, statistical computation.

1 Introduction

A common problem in many statistics applications is the numerical computation of the multivariate t (MVT) distribution function (see Tong, 1990) defined by

$$\mathbf{T}(\mathbf{a},\mathbf{b},\mathbf{\Sigma},\nu) = \frac{\Gamma(\frac{\nu+m}{2})}{\Gamma(\frac{\nu}{2})\sqrt{|\mathbf{\Sigma}|(\nu\pi)^m}} \int_{a_1}^{b_1}\int_{a_2}^{b_2}\cdots\int_{a_m}^{b_m} (1+\frac{\mathbf{x}^t\mathbf{\Sigma}^{-1}\mathbf{x}}{\nu})^{-\frac{\nu+m}{2}} d\mathbf{x} \tag{1}$$

$$\equiv \frac{2^{1-\frac{\nu}{2}}}{\Gamma(\frac{\nu}{2})} \int_0^\infty s^{\nu-1} e^{-\frac{s^2}{2}} \mathbf{\Phi}(\frac{s\mathbf{a}}{\sqrt{\nu}}, \frac{s\mathbf{b}}{\sqrt{\nu}}, \mathbf{\Sigma}) ds. \tag{2}$$

The second form for the MVT distribution function (Cornish, 1954) uses the multivariate Normal (MVN) distribution function, defined by

$$\mathbf{\Phi}(\mathbf{a},\mathbf{b},\mathbf{\Sigma}) = \frac{1}{\sqrt{|\mathbf{\Sigma}|(2\pi)^m}} \int_{a_1}^{b_1}\int_{a_2}^{b_2}\cdots\int_{a_m}^{b_m} e^{-\frac{1}{2}\mathbf{x}^t\mathbf{\Sigma}^{-1}\mathbf{x}} d\mathbf{x}.$$

*Revised version in *J. Comp. Graph. Stat* **11** (2002), pp. 950–971
†Partially supported by NATO Collaborative Research Grant CRG 940139

This definition of the MVT distribution function is also used in the definition of the non-central MVT (NCMVT),

$$\mathbf{T}(\mathbf{a},\mathbf{b},\mathbf{\Sigma},\nu,\boldsymbol{\delta}) = \frac{2^{1-\frac{\nu}{2}}}{\Gamma(\frac{\nu}{2})}\int_0^\infty s^{\nu-1}e^{-\frac{s^2}{2}}\mathbf{\Phi}(\frac{s\mathbf{a}}{\sqrt{\nu}}-\boldsymbol{\delta},\frac{s\mathbf{b}}{\sqrt{\nu}}-\boldsymbol{\delta},\mathbf{\Sigma})ds. \tag{3}$$

In all of these definitions $\mathbf{x} = (x_1, x_2, ..., x_m)^t$, $\mathbf{\Sigma}$ is an $m \times m$ symmetric positive definite covariance matrix and $-\infty \le a_i < b_i \le \infty$, for $i = 1, \ldots, m$. In the NCMVT case, the non-centrality vector $\boldsymbol{\delta}$ has components that satisfy $-\infty < \delta_i < \infty$. The purpose of this paper is to compare different methods for the numerical computation of MVT probabilities. We also discuss some methods for NCMVT probabilities.

There is reliable and efficient software available for computing $\mathbf{T}$ for $m = 1$, so we assume $m > 1$. The simplest traditional methods use acceptance-rejection sampling. Other methods for $m > 1$ use algorithms developed by Somerville (1997, 1998 and 1999), and Genz and Bretz (1999). We consider acceptance-rejection sampling, Somerville and related methods, the method of Genz and Bretz, and other methods that have not been carefully considered for MVT and NCMVT problems. In Section 2 we provide brief descriptions for various methods, in Section 3 we describe algorithms for implementing the methods and we report test results for the methods.

2 The Methods

All of the methods that we consider begin with a Cholesky decomposition of $\mathbf{\Sigma}$ in the form $\mathbf{\Sigma} = CC^t$, where C is a lower triangular $m \times m$ matrix. This is followed by the change of variables $\mathbf{x} = C\mathbf{y}$, so that $\mathbf{x}^t\mathbf{\Sigma}^{-1}\mathbf{x} = \mathbf{y}^t\mathbf{y}$, $d\mathbf{x} = |C|d\mathbf{y} = \sqrt{|\mathbf{\Sigma}|}d\mathbf{y}$, and therefore

$$\begin{aligned}\mathbf{T}(\mathbf{a},\mathbf{b},\mathbf{\Sigma},\nu) &= \frac{\Gamma(\frac{\nu+m}{2})}{\Gamma(\frac{\nu}{2})\sqrt{(\nu\pi)^m}}\int_{\mathbf{a}\le C\mathbf{y}\le\mathbf{b}}(1+\frac{\mathbf{y}^t\mathbf{y}}{\nu})^{-\frac{\nu+m}{2}}d\mathbf{y} &(4)\\ &\equiv \frac{2^{1-\frac{\nu}{2}}}{\Gamma(\frac{\nu}{2})}\int_0^\infty s^{\nu-1}e^{-\frac{s^2}{2}}\frac{1}{\sqrt{(2\pi)^m}}\int_{\frac{s\mathbf{a}}{\sqrt{\nu}}\le C\mathbf{y}\le\frac{s\mathbf{b}}{\sqrt{\nu}}}e^{-\frac{\mathbf{y}^t\mathbf{y}}{2}}d\mathbf{y}ds, &(5)\end{aligned}$$

and

$$\mathbf{T}(\mathbf{a},\mathbf{b},\mathbf{\Sigma},\nu,\boldsymbol{\delta}) = \frac{2^{1-\frac{\nu}{2}}}{\Gamma(\frac{\nu}{2})}\int_0^\infty s^{\nu-1}e^{-\frac{s^2}{2}}\frac{1}{\sqrt{(2\pi)^m}}\int_{\frac{s\mathbf{a}}{\sqrt{\nu}}-\boldsymbol{\delta}\le C\mathbf{y}\le\frac{s\mathbf{b}}{\sqrt{\nu}}-\boldsymbol{\delta}}e^{-\frac{\mathbf{y}^t\mathbf{y}}{2}}d\mathbf{y}ds. \tag{6}$$

Genz and Bretz (1999) introduced additional transformations for $\mathbf{T}$ as defined by equation (4). These transformations, which effect a separation of the variables, will be used for some of the methods to be described in the following sections. First, let $K_\nu^{(m)} = \frac{\Gamma(\frac{\nu+m}{2})}{\Gamma(\frac{\nu}{2})(\nu\pi)^{\frac{m}{2}}}$, and notice that $(1+\frac{\sum_{j=1}^m y_j^2}{\nu}) = (1+\frac{y_1^2}{\nu})(1+\frac{y_2^2}{\nu+y_1^2})\cdots(1+\frac{y_m^2}{\nu+\sum_{j=1}^{m-1}y_j^2})$. Now let $y_i = u_i\sqrt{\frac{\nu+\sum_{j=1}^{i-1}y_j^2}{\nu+i-1}}$ (which can equivalently be written $y_i = u_i\sqrt{\prod_{j=1}^{i-1}\frac{\nu+j-1+u_j^2}{\nu+j}}$). Then a little algebra (see Genz and Bretz, 1999, for some details) shows that

$$\mathbf{T}(\mathbf{a},\mathbf{b},\mathbf{\Sigma},\nu) = \int_{\mathbf{a}\le C\mathbf{y}(\mathbf{u})\le\mathbf{b}}\frac{K_\nu^{(1)}}{(1+\frac{u_1^2}{\nu})^{\frac{1+\nu}{2}}}\cdots\frac{K_{\nu+m-1}^{(1)}}{(1+\frac{u_m^2}{\nu+m-1})^{\frac{m+\nu}{2}}}d\mathbf{u}. \tag{7}$$

2.1 Acceptance-Rejection

If we denote the indicator function by $\mathbf{I}(e)$ (with $\mathbf{I}(e) = 1$ if e is true; otherwise $\mathbf{I}(e) = 0$), then a simple acceptance-rejection (AR) algorithm for the MVT problem, using equation (7), uses

$$\mathbf{T}(\mathbf{a}, \mathbf{b}, \mathbf{\Sigma}, \nu) \approx \frac{1}{N} \sum_{k=1}^{N} \mathbf{I}(\mathbf{a} \leq C\mathbf{y}(\mathbf{u}_k) \leq \mathbf{b}), \tag{8}$$

where $\{\mathbf{u}_k\}$ is random with components $u_{i,k} \sim t_{\nu+i-1}$, and we define the univariate t-distribution function by $t_\nu(u) = K_\nu^{(1)} \int_{-\infty}^{u} (1 + \frac{s^2}{\nu})^{-\frac{1+\nu}{2}} ds$.

A simple AR algorithm for the NCMVT problem, based on equation (6), uses

$$\mathbf{T}(\mathbf{a}, \mathbf{b}, \mathbf{\Sigma}, \nu, \boldsymbol{\delta}) \approx \frac{1}{N} \sum_{k=1}^{N} \mathbf{I}(\frac{s_k \mathbf{a}}{\nu} - \boldsymbol{\delta} \leq C\mathbf{y}_k \leq \frac{s_k \mathbf{b}}{\nu} - \boldsymbol{\delta}),$$

where $\{\mathbf{y}_k\}$ is random with components $y_{i,k} \sim N(0,1)$, and where $\{s_k\}$ is random with $s_k \sim \chi_\nu$, and we define $\chi_\nu(u) = \frac{2^{1-\frac{\nu}{2}}}{\Gamma(\frac{\nu}{2})} \int_0^u s^{\nu-1} e^{-\frac{s^2}{2}} ds$.

2.2 Spherical-Radial Transformation Methods

These methods use a transformation to a spherical-radial (SR) coordinate system. Let $\mathbf{y} = r\mathbf{z}$, with $||\mathbf{z}||_2 = 1$, so that $\mathbf{y}^t\mathbf{y} = r^2$ and $d\mathbf{y} = r^{m-1} dr d\mathbf{z}$. Deák (1980-90) used this transformation as the basis for several methods for MVN problems, and the methods described in this section can be considered as generalizations of Deák's methods. After the SR transformation, the MVT problem becomes

$$\begin{aligned} \mathbf{T}(\mathbf{a}, \mathbf{b}, \mathbf{\Sigma}, \nu) &= \frac{\Gamma(\frac{m}{2})}{2\pi^{\frac{m}{2}}} \int_{||\mathbf{z}||_2=1} \frac{2\Gamma(\frac{\nu+m}{2})}{\Gamma(\frac{m}{2})\Gamma(\frac{\nu}{2})\nu^{\frac{m}{2}}} \int_{\mathbf{a} \leq rC\mathbf{z} \leq \mathbf{b}} \frac{r^{m-1}}{(1 + \frac{r^2}{\nu})^{\frac{\nu+m}{2}}} dr d\mathbf{z} \\ &\equiv \frac{\Gamma(\frac{m}{2})}{2\pi^{\frac{m}{2}}} \int_{||\mathbf{z}||_2=1} F(\mathbf{a}, \mathbf{b}, C, \nu, \mathbf{z}) d\mathbf{z}, \end{aligned}$$

where $F(\mathbf{a}, \mathbf{b}, C, \nu, \mathbf{z})$ can be written in the form,

$$F(\mathbf{a}, \mathbf{b}, C, \nu, \mathbf{z}) = \frac{2\Gamma(\frac{\nu+m}{2})}{\Gamma(\frac{m}{2})\Gamma(\frac{\nu}{2})\nu^{\frac{m}{2}}} \int_{\rho_l(\mathbf{z})}^{\rho_u(\mathbf{z})} \frac{r^{m-1}}{(1 + \frac{r^2}{\nu})^{\frac{\nu+m}{2}}} dr.$$

We assume that the F values can be quickly computed using standard statistical software. If we let $\mathbf{v} = C\mathbf{z}$, the limits for the r-variable integration are given by

$$\rho_l(\mathbf{z}) = \max\{0, \max_{v_i>0}\{a_i/v_i\}, \max_{v_i<0}\{b_i/v_i\}\} \quad \text{and} \quad \rho_u(\mathbf{z}) = \max\{0, \min\{\min_{v_i>0}\{b_i/v_i\}, \min_{v_i<0}\{a_i/v_i\}\}\}.$$

For a given radial direction, $\mathbf{z}$, the ρ limits are the distances from the origin to the two points where the vector with direction $\mathbf{z}$ intersects the boundary of the integration region.

A Monte-Carlo algorithm for the MVT problem uses

$$\mathbf{T}(\mathbf{a}, \mathbf{b}, \mathbf{\Sigma}, \nu) \approx \frac{1}{N} \sum_{k=1}^{N} F(\mathbf{a}, \mathbf{b}, C, \nu, \mathbf{z}_k),$$

where the $\mathbf{z}_k$ points are uniformly random from U_m, the surface of the m-sphere. Sets of these points can easily be generated from sets of points uniform on C_{m-1}, the unit hypercube $[0,1]^{m-1}$, using the transformation (see Fang and Wang, 1994), from a point $\mathbf{w} \in C_{m-1}$ to a point $\mathbf{z} \in U_m$, defined by

$$\begin{aligned} z_{m-2i+2}(\mathbf{w}) &= \sin(2\pi w_{m-2i+1})\sqrt{1-w_{m-2i}^{\frac{2}{m-2i}}}\prod_{k=1}^{i-1} w_{m-2k}^{\frac{1}{m-2k}} \\ z_{m-2i+1}(\mathbf{w}) &= \cos(2\pi w_{m-2i+1})\sqrt{1-w_{m-2i}^{\frac{2}{m-2i}}}\prod_{k=1}^{i-1} w_{m-2k}^{\frac{1}{m-2k}} \end{aligned}$$

for $i = 1, 2, \ldots l$, where $l = \lfloor \frac{m}{2} \rfloor - 1$, and ending with

$$z_2(\mathbf{w}) = \sin(2\pi w_1)\prod_{k=1}^{l} w_{m-2k}^{\frac{1}{m-2k}} \quad \text{and} \quad z_1(\mathbf{w}) = \cos(2\pi w_1)\prod_{k=1}^{l} w_{m-2k}^{\frac{1}{m-2k}},$$

when m is even, or ending with

$$z_3(\mathbf{w}) = (2w_1 - 1)\prod_{k=1}^{l} w_{m-2k}^{\frac{1}{m-2k}},$$

$$z_2(\mathbf{w}) = 2\sin(2\pi w_2)\sqrt{w_1(1-w_1)}\prod_{k=1}^{l} w_{m-2k}^{\frac{1}{m-2k}} \quad \text{and} \quad z_1(\mathbf{w}) = 2\cos(2\pi w_2)\sqrt{w_1(1-w_1)}\prod_{k=1}^{l} w_{m-2k}^{\frac{1}{m-2k}},$$

when m is odd. This transformation has a constant Jacobian, so a Monte-Carlo algorithm for the MVT problem based on uniform C_{m-1} points uses

$$\mathbf{T}(\mathbf{a}, \mathbf{b}, \mathbf{\Sigma}, \nu) \approx \frac{1}{N}\sum_{k=1}^{N} F(\mathbf{a}, \mathbf{b}, C, \nu, \mathbf{z}(\mathbf{w}_k)), \tag{9}$$

with all $w_{i,k} \sim Uniform(0,1)$. This method can be also used for integration regions that are not hyper-rectangles. All that is needed for other regions is an efficient method for computing the intersection points for the boundary of the integration region for each radial direction. Lohr (1990) has described this type of generalization for MVN problems.

The use of various types of antithetic variates, as described by Deák (1990) for improving the convergence of SR MVN methods, can improve the convergence of this type of method. Let Z be an $m \times m$ uniformly random (with Haar measure, see Stewart, 1980) orthogonal matrix with columns $\{\mathbf{z}_j\}$, and define

$$S_n(Z) = \frac{1}{2^n\binom{m}{n}}\sum_{\mathbf{s}}\sum_{1\le j_1<\ldots<j_n\le m} F(\mathbf{a}, \mathbf{b}, C, \nu, \frac{\sum_{l=1}^{n} s_l \mathbf{z}_{j_l}}{\sqrt{n}}),$$

where $\mathbf{s} = (s_1, s_2, ..., s_n) = (\pm 1, ..., \pm 1)$ and the outer sum is taken over the 2^n possible sign combinations for the components of $\mathbf{s}$. The sample points used by $S_n(Z)$ are very evenly spread over the surface of the unit m-sphere. For MVN problems, Deák found that the larger values for the parameter n (which must satisfy $n \le m$) can provide values for S_n with significantly smaller variances. But the larger the n value, the higher the computational cost for S_n, so these two features of the S_n sums must be balanced, for practical computations. Deák recommended values of $n = 1, 2$ or 3 for typical computations. MVT estimates based on S_n are obtained using

$$\mathbf{T}(\mathbf{a}, \mathbf{b}, \mathbf{\Sigma}, \nu) \approx \frac{1}{M}\sum_{k=1}^{M} S_n(Z_k). \tag{10}$$

One problem with the SR transformation algorithms is that $F(\mathbf{a}, \mathbf{b}, C, \nu, \mathbf{z})$ as a function of $\mathbf{z}$, although continuous, is not very smooth, because of sharp corners of the integration region defined by $\mathbf{a} \le rC\mathbf{z} \le \mathbf{b}$.

This problem resulted in approximations with large variation and slower convergence for SR algorithms for MVN problems (see Genz, 1992). For some combinations of $\mathbf{a}$ and $\mathbf{b}$, (e.g. if $\mathbf{0} < \mathbf{a} < \mathbf{b}$) many F values will be zero, and this can cause further reductions in the efficiency of the SR algorithms.

SR methods for the NCMVT problem can easily be constructed by combining a numerical integration method for the s variable (in equation (3)) with an SR method for the inner MVN integral. Using the SR transformation for the inner integral we have

$$\begin{aligned}\mathbf{T}(\mathbf{a},\mathbf{b},\mathbf{\Sigma},\nu,\boldsymbol{\delta}) &= \frac{2^{1-\frac{\nu}{2}}}{\Gamma(\frac{\nu}{2})}\int_0^\infty s^{\nu-1}e^{-\frac{s^2}{2}}\frac{\Gamma(\frac{m}{2})}{2\pi^{\frac{m}{2}}}\int_{||\mathbf{z}||_2=1}\frac{2^{\frac{m}{2}-1}}{\Gamma(\frac{m}{2})}\int_{\frac{s\mathbf{a}}{\sqrt{\nu}}-\boldsymbol{\delta}\le rC\mathbf{z}\le\frac{s\mathbf{a}}{\sqrt{\nu}}-\boldsymbol{\delta}} r^{m-1}e^{-\frac{r^2}{2}}dr d\mathbf{z} ds \\ &\equiv \frac{2^{1-\frac{\nu}{2}}}{\Gamma(\frac{\nu}{2})}\int_0^\infty s^{\nu-1}e^{-\frac{s^2}{2}}\frac{\Gamma(\frac{m}{2})}{2\pi^{\frac{m}{2}}}\int_{||\mathbf{z}||_2=1} G(\mathbf{a},\mathbf{b},C,\nu,\boldsymbol{\delta},\mathbf{z},s)d\mathbf{z}ds,\end{aligned}$$

with G appropriately defined. A simple Monte-Carlo algorithm for the NCMVT problem uses

$$\mathbf{T}(\mathbf{a},\mathbf{b},\mathbf{\Sigma},\nu,\boldsymbol{\delta}) \approx \frac{1}{N}\sum_{k=1}^{N} G(\mathbf{a},\mathbf{b},C,\nu,\mathbf{z}_k,s_k),$$

where $\{\mathbf{z}_k\}$ are uniformly random from the surface of m-sphere and $s_k \sim \chi_\nu$.

Second formulations of the MVT and NCMVT problems, using the SR coordinate system, reverse the order of the radial and spherical integrations. The result for the MVT problem is

$$\begin{aligned}\mathbf{T}(\mathbf{a},\mathbf{b},\mathbf{\Sigma},\nu) &= \frac{2\Gamma(\frac{\nu+m}{2})}{\Gamma(\frac{m}{2})\Gamma(\frac{\nu}{2})\nu^{\frac{m}{2}}}\int_0^\infty \frac{r^{m-1}}{(1+\frac{r^2}{\nu})^{\frac{\nu+m}{2}}}\frac{\Gamma(\frac{m}{2})}{2\pi^{\frac{m}{2}}}\int_{||\mathbf{z}||_2=1,\mathbf{a}\le rC\mathbf{z}\le\mathbf{b}} d\mathbf{z}dr \\ &\equiv \frac{2\Gamma(\frac{\nu+m}{2})}{\Gamma(\frac{m}{2})\Gamma(\frac{\nu}{2})\nu^{\frac{m}{2}}}\int_0^\infty \frac{r^{m-1}}{(1+\frac{r^2}{\nu})^{\frac{\nu+m}{2}}}H(\mathbf{a},\mathbf{b},C,\nu,r)dr,\end{aligned}$$

where H is appropriately defined. Algorithms for the general MVT and NCMVT problems using these formulations can suffer from convergence problems similar to those for the first formulations. However, Somerville (1997-99) has found the MVT second formulation to be useful for some confidence interval computation applications.

2.3 Separation-of-Variables Methods

Genz and Bretz (1999) continued the separation of variables (SV) transformation that results in equation (7) with additional transformations (similar to those used by Genz, 1992, for MVN problems), to produce methods based on a complete transformation of the original integration to the unit hypercube $[0,1]^m$. The first step is to notice that the lower triangular structure of C allows a separation of the integration limits in equation (7), so that

$$\mathbf{T}(\mathbf{a},\mathbf{b},\mathbf{\Sigma},\nu) = \int_{\tilde{a}_1}^{\tilde{b}_1}\frac{K_\nu^{(1)}}{(1+\frac{u_1^2}{\nu})^{\frac{1+\nu}{2}}}\cdots\int_{\tilde{a}_m(y_1,\ldots,y_{m-1})}^{\tilde{b}_m(y_1,\ldots,y_{m-1})}\frac{K_{\nu+m-1}^{(1)}}{(1+\frac{u_m^2}{m+\nu-1})^{\frac{m+\nu}{2}}}d\mathbf{u}; \tag{11}$$

with $\tilde{a}_i = \sqrt{\frac{\nu+i-1}{\nu+\sum_{j=1}^{i-1}y_j^2}}(a_i - \sum_{j=1}^{i-1}c_{i,j}y_j)/c_{i,i}$, and $\tilde{b}_i = \sqrt{\frac{\nu+i-1}{\nu+\sum_{j=1}^{i-1}y_j^2}}(b_i - \sum_{j=1}^{i-1}c_{i,j}y_j)/c_{i,i}$.

Next, let $u_i = t_{\nu+i-1}^{-1}(z_i)$, and then

$$\mathbf{T}(\mathbf{a},\mathbf{b},\mathbf{\Sigma},\nu) = \int_{t_\nu(\tilde{a}_1)}^{t_\nu(\tilde{b}_1)}\cdots\int_{t_{\nu+m-1}(\tilde{a}_m(t_\nu^{-1}(z_1),\ldots,t_{\nu+m-2}^{-1}(z_{m-1})))}^{t_{\nu+m-1}(\tilde{b}_m(t_\nu^{-1}(z_1),\ldots,t_{\nu+m-2}^{-1}(z_{m-1})))} d\mathbf{z}.$$

Finally, let $z_i = d_i + (e_i - d_i)w_i$, so $dz_i = (e_i - d_i)dw_i$, with

$$\begin{aligned} d_i(w_1, ..., w_{i-1}) &= t_{\nu+i-1}(\tilde{a}_i(t_\nu^{-1}(z_1(w_1)), ..., t_{\nu+i-2}^{-1}(z_{i-1}(w_{i-1})))), \\ e_i(w_1, ..., w_{i-1}) &= t_{\nu+i-1}(\tilde{b}_i(t_\nu^{-1}(z_1(w_1)), ..., t_{\nu+i-2}^{-1}(z_{i-1}(w_{i-1})))). \end{aligned}$$

Then

$$\mathbf{T}(\mathbf{a}, \mathbf{b}, \mathbf{\Sigma}, \nu) = (e_1 - d_1) \int_0^1 (e_2(w_1) - d_2(w_1)) \cdots \int_0^1 (e_m(w_1, \ldots, w_{m-1}) - d_m(w_1, \ldots, w_{m-1})) \int_0^1 d\mathbf{w}.$$

The innermost integral has value equal to one, so the number of integration variables has been reduced to $m-1$, and standard multidimensional numerical integration methods can be used for the transformed $\mathbf{T}$. A simple Monte-Carlo algorithm for the MVT problem uses

$$\mathbf{T}(\mathbf{a}, \mathbf{b}, \mathbf{\Sigma}, \nu) \approx \frac{1}{N} \sum_{k=1}^{N} \prod_{i=1}^{m} (e_i(\mathbf{w}_k) - d_i(\mathbf{w}_k)) \tag{12}$$

with $w_{i,k} \sim Uniform(0, 1)$.

Schervish (1984) originally suggested that the computation of MVN probabilities should be easier for numerical integration methods if the variables are reordered (and appropriate rows and columns of $\mathbf{\Sigma}$ are permuted) so that the innermost integrals have the larger integration intervals. This sorting heuristic often has the effect that the innermost integrals have expected value closer to one, thereby reducing the overall variation in the integrand. Gibson, Glasbey and Elston (1992), who independently developed a Monte-Carlo method similar to the one developed by Genz (1992) for MVN probabilities, suggested an improved prioritization of the variables. With this technique, the variables are sorted so that the innermost integrals have the largest expected integration intervals. This is more complicated than just sorting the integration limits and permuting the respective rows and columns of $\mathbf{\Sigma}$, because the Cholesky factor C must be computed dynamically during the sorting of the variables. This method uses $\mathbf{a}$, $\mathbf{b}$ and $\mathbf{\Sigma}$ in the sorting process, and it should therefore further increases the likelihood that the innermost integrals have values close to one and improve the convergence of the numerical integration methods.

The Gibson, Glasbey and Elston method can be generalized to MVT problems, which use the SV transformations, in the following manner, using the MVT definition given by equation (11). The first (outermost) integration variable is chosen by selecting a variable i where $\min_{1 \le i \le m}\{t_\nu(b_i/\sqrt{\sigma_{i,i}}) - t_\nu(a_i/\sqrt{\sigma_{i,i}})\}$ is achieved. The limits and rows and columns of $\mathbf{\Sigma}$ for variables 1 and i are interchanged. Then the first column of the Cholesky decomposition C of $\mathbf{\Sigma}$ is computed using $c_{1,1} = \sqrt{\sigma_{1,1}}$ and $c_{i,1} = \sigma_{i,1}/c_{1,1}$, for $i = 2, \ldots, m$, and we set

$$y_1 = u_1 = \frac{K_\nu^{(1)} \int_{a_1}^{b_1} s(1 + \frac{s^2}{\nu})^{-\frac{\nu+1}{2}} ds}{t_\nu(b_1) - t_\nu(a_1)}.$$

Given this (expected) value for y_1, the second integration variable is chosen by selecting a variable i where

$$\min_{2 \le i \le m} \left\{ t_{\nu+1}\Big(\sqrt{\frac{\nu+1}{\nu+y_1^2}} \frac{b_i - c_{i,1}y_1}{\sqrt{\sigma_{i,i} - c_{i,1}^2}}\Big) - t_{\nu+1}\Big(\sqrt{\frac{\nu+1}{\nu+y_1^2}} \frac{a_i - c_{i,1}y_1}{\sqrt{\sigma_{i,i} - c_{i,1}^2}}\Big) \right\}$$

is achieved. The integration limits, rows and columns of $\mathbf{\Sigma}$, and rows of C for variables 2 and i are interchanged. Then the second column of C is computed using $c_{2,2} = \sqrt{\sigma_{2,2} - c_{2,1}^2}$ and $c_{i,2} = (\sigma_{i,2} - c_{2,1}c_{i,1})/c_{2,2}$, for $i = 3, \ldots, m$, and we compute the expected value for u_2 using

$$u_2 = \frac{K_{\nu+1}^{(1)} \int_{\tilde{a}_2}^{\tilde{b}_2} s(1 + \frac{s^2}{\nu+1})^{-\frac{\nu+2}{2}} ds}{t_{\nu+1}(\tilde{b}_2) - t_{\nu+1}(\tilde{a}_2)},$$

and we set $y_2 = u_1\sqrt{\frac{\nu+y_1^2}{\nu+1}}$. At stage j, given the expected values for $y_1, y_2, \ldots, y_{j-1}$, the jth integration variable is chosen by selecting a variable i, where

$$\min_{j\le i\le m}\Big\{t_{\nu+j-1}\Big(\sqrt{\frac{\nu+j-1}{\nu+\sum_{k=1}^{j-1}y_k^2}}\frac{b_i-\sum_{k=1}^{j-1}c_{i,k}y_k}{\sqrt{\sigma_{i,i}-\sum_{k=1}^{j-1}c_{i,k}^2}}\Big)-t_{\nu+j-1}\Big(\sqrt{\frac{\nu+j-1}{\nu+\sum_{k=1}^{j-1}y_k^2}}\frac{a_i-\sum_{k=1}^{j-1}c_{i,k}y_k}{\sqrt{\sigma_{i,i}-\sum_{k=1}^{j-1}c_{i,k}^2}}\Big)\Big\}$$

is achieved. The integration limits, rows and columns of $\mathbf{\Sigma}$, and rows of C for variables j and i are interchanged. Then the jth column of C is computed using $c_{j,j} = \sqrt{\sigma_{j,j}-\sum_{k=1}^{j-1}c_{j,k}^2}$ and $c_{i,j} = (\sigma_{i,j} - \sum_{k=1}^{j-1}c_{j,k}c_{i,k})/c_{j,j}$, for $i = j+1,\ldots,m$, and we let

$$u_j = \frac{K_{\nu+j-1}^{(1)}\int_{\tilde a_j}^{\tilde b_j} s(1+\frac{s^2}{\nu+j-1})^{-\frac{\nu+j}{2}}ds}{t_{\nu+j-1}(\tilde b_j)-t_{\nu+j-1}(\tilde a_j)}.$$

and set $y_j = u_j\sqrt{\frac{\nu+\sum_{i=1}^{j-1}y_i^2}{\nu+j-1}}$. The complete $m-1$ stage process has overall cost $O(m^3)$, which is not significant compared to the rest of the computation cost for the methods discussed here, and is therefore a relatively cheap preconditioning step that can be used with the algorithms. This sorting technique was included in the implementations for some of the algorithms used for the tests reported later in this paper.

A second set of MVT SV methods are based on the combination of equation (2) with MVN SV methods using

$$\mathbf{T}(\mathbf{a},\mathbf{b},\mathbf{\Sigma},\nu) = \frac{2^{1-\frac{\nu}{2}}}{\Gamma(\frac{\nu}{2})}\int_0^\infty s^{\nu-1}e^{-\frac{s^2}{2}}\int_{\hat a_1(s)}^{\hat b_1(s)} e^{-\frac{y_1^2}{2}}\cdots\int_{\hat a_m(s,y_1,\ldots,y_{m-1})}^{\hat b_m(s,y_1,\ldots,y_{m-1})} e^{-\frac{y_m^2}{2}}d\mathbf{y}ds, \tag{13}$$

with $\hat a_i(s,y_1....,y_{i-1}) = \frac{s}{\sqrt{\nu}}(a_i-\sum_{j=1}^{i-1}c_{i,j}y_j)/c_{i,i}$, and $\hat b_i(s,y_1....,y_{i-1}) = \frac{s}{\sqrt{\nu}}(b_i-\sum_{j=1}^{i-1}c_{i,j}y_j)/c_{i,i}$. Then

$$\mathbf{T}(\mathbf{a},\mathbf{b},\mathbf{\Sigma},\nu) = \frac{2^{1-\frac{\nu}{2}}}{\Gamma(\frac{\nu}{2})}\int_0^\infty s^{\nu-1}e^{-\frac{s^2}{2}}(\hat e_1(s,\mathbf{w})-\hat d_1(s,\mathbf{w}))\int_0^1\cdots(\hat e_m(s,\mathbf{w})-\hat d_m(s,\mathbf{w}))\int_0^1 d\mathbf{w}ds,$$

with $\hat z_i = \hat d_i + (\hat e_i - \hat d_i)w_i$, and

$$\begin{aligned}\hat d_i(s,\mathbf{w}) &= \Phi(\hat a_m(s,\Phi^{-1}(\hat z_1(w_1)),...,\Phi^{-1}(\hat z_{i-1}(w_{i-1})))),\\ \hat e_i(s,\mathbf{w}) &= \Phi(\hat b_m(s,\Phi^{-1}(\hat z_1(w_1)),...,\Phi^{-1}(\hat z_{i-1}(w_{i-1})))).\end{aligned}$$

The last $\mathbf{w}$ component integral has value one, so the $\mathbf{T}$ integral is determined as an m-dimensional integral. A simple Monte-Carlo algorithm for this formulation of the MVT problem uses

$$\mathbf{T}(\mathbf{a},\mathbf{b},\mathbf{\Sigma},\nu) \approx \frac{1}{N}\sum_{k=1}^N\prod_{i=1}^m(\hat e_i(s_k,\mathbf{w})-\hat d_i(s_k,\mathbf{w})) \tag{14}$$

with $w_{i,k} \sim Uniform(0,1)$, for $i = 1,2,\ldots,m-1$, and $s_k \sim \chi_\nu$. An inverse χ distribution function can be used to generate the required s_k's from $Uniform(0,1)$ random numbers, via the formula $s_k = \chi^{-1}(w_{m,k})$, so each term in the Monte-Carlo sum for $\mathbf{T}$ requires m $Uniform(0,1)$ random numbers.

The Gibson, Glasbey and Elston technique can also be used as a preconditioning step for this method. In order to simplify our implementation of this preconditioning, we use $\sqrt{\nu}$ for the expected value for s, instead of the theoretical $\sqrt{2}\Gamma((\nu+1)/2)/\Gamma(\nu/2)$ (which approaches $\sqrt{\nu}$ for large ν). This value for s cancels the $\sqrt{\nu}$'s in the integration limits $\hat{\mathbf{a}}$ and $\hat{\mathbf{b}}$, so the preconditioning step for this method used the original Gibson, Glasbey and Elston technique with input $\mathbf{a}$, $\mathbf{b}$, and $\mathbf{\Sigma}$ (and $\boldsymbol{\delta}$ for NCMVT problems) and completes the preconditioning step as if the problem were an MVN problem.

2.4 An Example

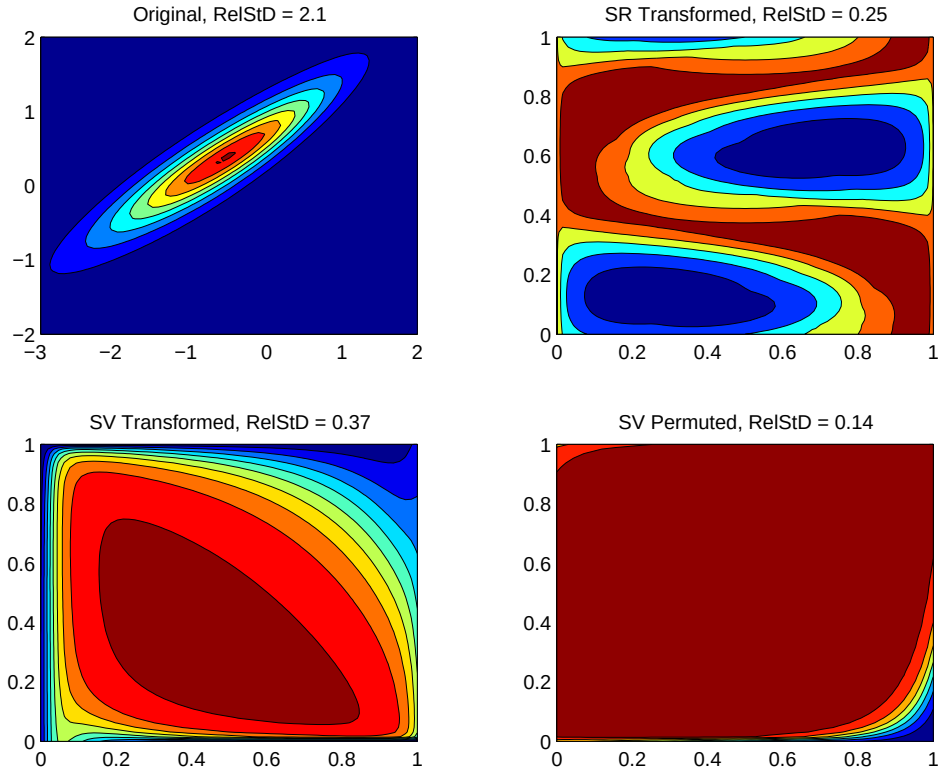

Figure 1: Example Graphs

We now illustrate some of the methods described in this section with an example. Consider the three-dimensional problem where $\mathbf{a} = (-3, -2, -1)^t$, $\mathbf{b} = (2, 2, 2)^t$, $\nu = 5$ and

$$\mathbf{\Sigma} = \begin{bmatrix} 1 & 12/13 & -3/5 \\ 12/13 & 1 & -4/5 \\ -3/5 & -4/5 & 1 \end{bmatrix}, \quad \text{with } C = \begin{bmatrix} 1 & 0 & 0 \\ 12/13 & 5/13 & 0 \\ -3/5 & -16/25 & 12/15 \end{bmatrix}.$$

The SV integration region is determined by $\mathbf{a} \le C\mathbf{y} \le \mathbf{b}$:

$$-3 \le y_1 \le 2, \quad \frac{-2 - 12y_1/13}{5/13} \le y_2 \le \frac{2 - 12y_1/13}{5/13}, \quad \frac{-1 + 3y_1/5 + 16y_2/25}{12/25} \le y_3 \le \frac{2 + 3y_1/5 + 16y_2/25}{12/25},$$

with $y_1 = u_1$, $y_2 = u_2\sqrt{\frac{5+u_1^2}{6}}$, $y_3 = u_3\sqrt{\frac{5+u_1^2}{6}\frac{6+u_2^2}{7}}$, so

$$\begin{aligned} \mathbf{T}(\mathbf{a}, \mathbf{b}, \mathbf{\Sigma}, 5) &= \frac{\Gamma(\frac{8}{2})}{\Gamma(\frac{5}{2})\sqrt{(5\pi)^3}} \int\limits_{\mathbf{a} \le C\mathbf{y} \le \mathbf{b}} (1 + \frac{\mathbf{y}^t\mathbf{y}}{5})^{-\frac{8}{2}} d\mathbf{y} \\ &= \int_{-3}^{2} \frac{K_5^{(1)}}{(1 + \frac{u_1^2}{5})^{\frac{6}{2}}} \int_{\tilde{a}_2(u_1)}^{\tilde{b}_2(u_1)} \frac{K_6^{(1)}}{(1 + \frac{u_2^2}{6})^{\frac{7}{2}}} \int_{\tilde{a}_3(u_1,u_2)}^{\tilde{b}_3(u_1,u_2)} \frac{K_7^{(1)}}{(1 + \frac{u_1^2}{7})^{\frac{8}{2}}} d\mathbf{u}, \end{aligned}$$

with $\tilde{a}_2(u_1) = \sqrt{\frac{6}{5+u_1^2}}(-\frac{2}{5} - \frac{12}{5}u_1)$, $\tilde{b}_2(u_1) = \sqrt{\frac{6}{5+u_1^2}}(\frac{26}{5} - \frac{12}{5}u_1)$,
$\tilde{a}_3(u_1, u_2) = \sqrt{\frac{6}{5+u_1^2}\frac{7}{6+u_2^2}}(-\frac{25}{12} + \frac{5}{4}u_1 + \frac{4}{3}u_2\sqrt{\frac{6}{5+u_1^2}})$, and $\tilde{b}_3(u_1, u_2) = \sqrt{\frac{6}{5+u_1^2}\frac{7}{6+u_2^2}}(\frac{25}{6} + \frac{5}{4}u_1 + \frac{4}{3}u_2\sqrt{\frac{6}{5+u_1^2}})$.

Then we use $d_1 = t_5(-3)$, $e_1 = t_5(2)$, $z_1 = d_1 + (e_1 - d_1)w_1$, $u_1 = t_5^{-1}(z_1)$, $d_2 = t_6(\tilde{a}_2(u_1))$, $e_2 = t_6(\tilde{b}_2(u_1)$, $z_2 = d_2 + (e_2 - d_2)w_2$, $u_2 = t_6^{-1}(z_2)$, $d_3 = t_7(\tilde{a}_3(u_1, u_2))$, $e_3 = t_7(\tilde{b}_3(u_1, u_2))$, so

$$\mathbf{T}(\mathbf{a}, \mathbf{b}, \mathbf{\Sigma}, 5) = (e_1 - d_1) \int_0^1 (e_2 - d_2) \int_0^1 (e_3 - d_3) dw_2 dw_1.$$

In Figure 1 we provide four graphs. The top left graph is a contour graph for the original density function for this example, with the last variable integrated out. The top right graph is a contour graph for the SR integrand F (see equation 9). The bottom left graph is a contour graph for the SV integrand $(e_1 - d_1)(e_2(w_1) - d_2(w_1))(e_3(w_1, w_2) - d_3(w_1, w_2))$. The bottom right graph also shows the SV integrand for the example in this section but variables 1 and 3 have been interchanged. These graphs show the integrands that would be sampled by Monte-Carlo methods that we have described for the transformed problems. The "RelStD" quantities given for each graph are relative standard errors (standard error divided by the mean) for the data used to plot each graph.

3 Algorithms and Tests

The assumed input for the algorithms that are described in this section will consist of the list $\mathbf{a}$, $\mathbf{b}$, $\mathbf{\Sigma}$, ν, $\boldsymbol{\delta}$, ϵ and N_{max}. The vector $\boldsymbol{\delta}$ is assumed to be $\mathbf{0}$ for MVT problems. The parameters ϵ, an absolute error tolerance parameter, and N_{max}, a limit on the number of simulations or integrand values allowed, are introduced because of the nature of the MVT algorithms, where, for many practical problems, we cannot expect to feasibly compute high accuracy MVT values. We expect a reliable algorithm to produce output consisting of an approximation $\hat{\mathbf{T}}$ and an approximate error $\hat{\epsilon}$ satisfying $|\mathbf{T} - \hat{\mathbf{T}}| < \hat{\epsilon}$ most of the time, and with $\hat{\epsilon} \leq \epsilon$ in those cases where N_{max} is large enough to allow algorithm termination.

3.1 Monte Carlo Algorithm Tests

All of the methods described in the previous can be implemented as MC methods that use only $Uniform(0, 1)$ random numbers. This includes the methods that use χ random numbers, which can be obtained from $Uniform(0, 1)$ numbers via the inverse χ distribution, and the methods that use sequences of random orthogonal matrices, because each orthogonal matrix can be generated from a sequence of $\frac{m(m-1)}{2}$ $Uniform(0, 1)$, (see Fang and Wang, 1994). Therefore all of the MC methods discussed so far, can be viewed as MC methods for integrals of the form

$$I(f) = \int_0^1 \int_0^1 \cdots \int_0^1 f(\mathbf{v}) d\mathbf{v},$$

where f is appropriately chosen, and, for most of the algorithms, $\mathbf{v}$ has length m or $m - 1$, but could have length $m(m-1)/2$ for the methods that use random orthogonal matrices. The standard error can be used to provide an error estimate for all of the algorithms for these methods. To standardize notation, we use $\hat{\mathbf{T}}_N$ to denote an approximation to $\mathbf{T}$ obtained using N simulation or random integrand values $\{f(\mathbf{v}_k)\}_{k=1}^N$,with $\hat{\mathbf{T}}_N = \frac{1}{N} \sum_{k=1}^N f(\mathbf{v}_k)$. The standard error σ_N^2, for $\hat{\mathbf{T}}_N$, is defined using $\sigma_N^2 = \frac{1}{N(N-1)} \sum_{k=1}^N (f(\mathbf{v}_k) - \hat{\mathbf{T}}_N)^2$. We will use the error estimate $\hat{\epsilon} = 3\sigma_N$ in order to provide an approximate confidence level of 99% for our algorithms.

When the input for a particular problem is given, there is usually no way of knowing how large N needs to be before we have $3\sigma_N < \epsilon$, so some type of iterative algorithm is required. We have implemented all of our MC algorithms in the following iterative form.

Monte Carlo Algorithm

1. **Input** $\mathbf{a}$, $\mathbf{b}$, $\mathbf{\Sigma}$, ν, $\boldsymbol{\delta}$, ϵ, and N_{max};.
2. **Set** $\bar{N} = N_{min}$;

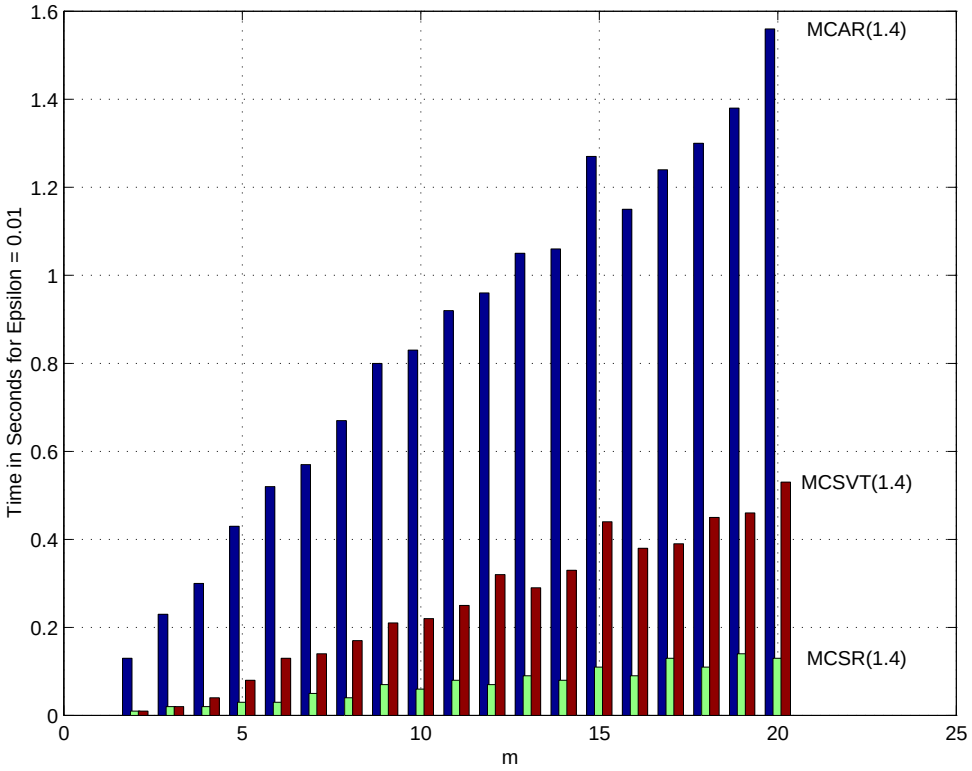

Figure 2: Average Monte Carlo MVT Algorithm Times, $\epsilon = 10^{-2}$

3. **Compute** $\hat{\mathbf{T}}_{\bar{N}}$ and $\sigma_{\bar{N}}$;
4. **Set** $\bar{\mathbf{T}} = \hat{\mathbf{T}}_{\bar{N}}$, $\bar{\sigma} = \sigma_{\bar{N}}$, $\hat{\epsilon} = 3\bar{\sigma}$;
5. **Do While** ($\hat{\epsilon} > \epsilon$ **and** $\bar{N} < N_{max}$)
   (a) **Set** $N = \max(\min(\lceil \bar{N}((\hat{\epsilon}/\epsilon)^2 - 1) \rceil, N_{max} - \bar{N}), 25)$;
   (b) **Compute** $\hat{\mathbf{T}}_N$ and σ_N;
   (c) **Set** $\bar{N} = \bar{N} + N$, $\bar{\mathbf{T}} = \bar{\mathbf{T}} + \bar{\sigma}^2(\hat{\mathbf{T}}_N - \bar{\mathbf{T}})/(\bar{\sigma}^2 + \sigma_N^2)$, $\bar{\sigma}^2 = \bar{\sigma}^2\sigma_N^2/(\bar{\sigma}^2 + \sigma_N^2)$, $\hat{\epsilon} = 3\bar{\sigma}$;
   End Do
6. **Output** $\bar{\mathbf{T}}$, $\bar{N}$, $\hat{\epsilon}$.

This is usually a two iteration algorithm. We initially use a relatively small N_{min} (we set $N_{min} = 100$ for all MC algorithms) to estimate the variance σ_f^2 for f, assuming $\sigma_f \approx \sigma_{N_{min}}\sqrt{N_{min}}$. Then we try to pick an N so that the next iteration completes the calculation. We choose the smallest N satisfying $N + N_{min} \le N_{max}$ and $3\bar{\sigma} = 3\sigma_{N+N_{min}} \approx 3\sigma_f/\sqrt{N + N_{min}} \le \epsilon$. The algebraic rearrangement of this inequality provides the heuristic formula used at step (5a). In practice, we sometimes require more than two iterations, which is why we have implemented the algorithm in the form given. The updating formulas for $\bar{\mathbf{T}}$ and $\bar{\sigma}$ are algebraically equivalent to $\bar{\mathbf{T}} = \sum_{i=1}^{k} \frac{\hat{\mathbf{T}}_{N_i}}{\sigma_{N_i}^2} / \sum_{i=1}^{k} \frac{1}{\sigma_{N_i}^2}$ and $\bar{\sigma}^2 = 1/\sum_{i=1}^{k} \frac{1}{\sigma_{N_i}^2}$, if the algorithm requires k iterations, with sample sizes $N_1, N_2, \ldots, N_k$, so that the final $\bar{N} = \sum_{i=1}^{k} N_i$. This type of weighted combination of values for iterative MC algorithms was used by Lepage (1978). If $\sigma_{N_i}^2 \approx \sigma_f^2/N_i$, then $\bar{\mathbf{T}} \approx \sum_{i_1}^{k} \hat{\mathbf{T}}_{N_i}/\bar{N}$, and $\bar{\sigma}^2 \approx \sigma_f^2/\bar{N}$, which provides some motivation for the choice of weighting.

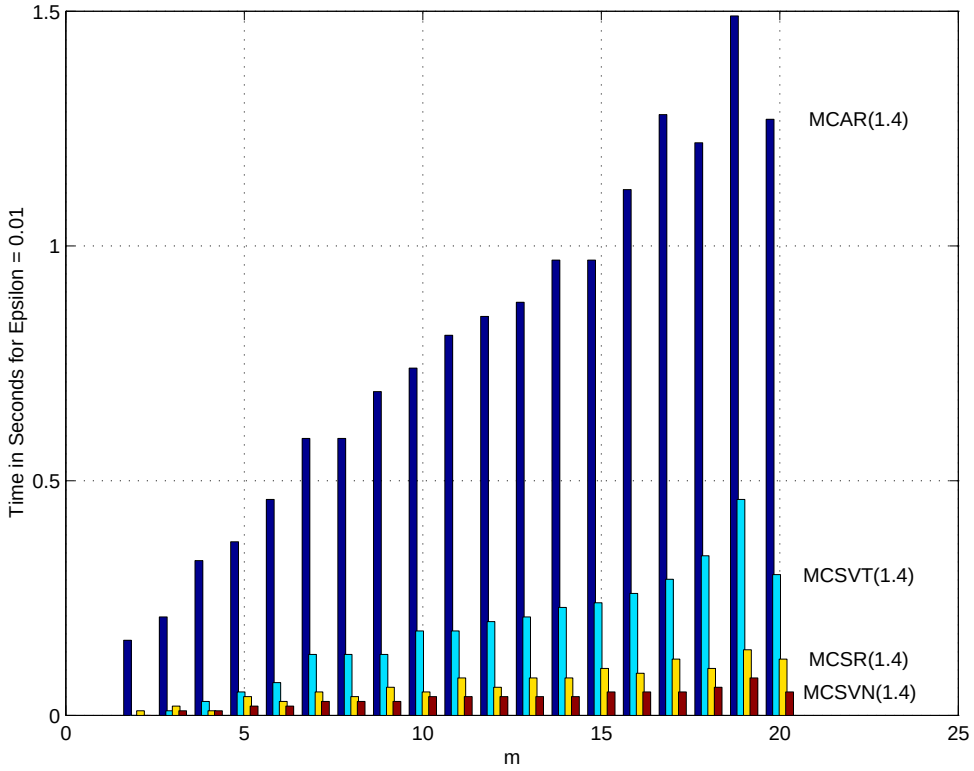

Figure 3: Average Monte Carlo Symmetrized MVT Algorithm Times, $\epsilon = 10^{-2}$

The type of testing that we use consists of applying several algorithms to a set of randomly selected MVT problems, $\{\mathbf{T}(\mathbf{a}_j, \mathbf{b}_j, \mathbf{\Sigma}_j, \nu_j)\}_{j=1}^{L}$, for selected values of ϵ, m and L, and with N_{max} chosen large enough to allow the algorithms to terminate. We generate random covariance matrices $\mathbf{\Sigma}_j$ which are random correlation matrices using the algorithm described by Marsaglia and Olkin (1984). We generate random limit vectors $\mathbf{a}_j$ with components $a_{i,j} = -3v_{i,j}\sqrt{m}$, $\mathbf{b}_j$ with components $b_{i,j} = 3w_{i,j}\sqrt{m}$, and random $\nu_j = \max(1, \lfloor 10u_j\sqrt{m} \rfloor)$, where all $v_{i,j}$, $w_{i,j}$ and u_j are $Uniform(0,1)$ random numbers. The distributions that we have chosen for $\mathbf{b}_j$ and ν_j were picked after some experimentation to produce average MVT values in the range .2-.8 for m in the range 2-20. We compute the average time taken by each method for each m, and the average, denoted by $\bar{c}$, for $\log(|\mathbf{T}_j - \hat{\mathbf{T}}_j|)/\log(\epsilon)$ (the ratio of the number of correct digits produced to the number of digits requested). The "correct" $\mathbf{T}_j$ values used for our tests were determined using our most efficient algorithm (QSMVN, described later in this section) with input error tolerance set at $\epsilon/10$. We also count the number of times $\hat{\epsilon}_j > \epsilon$ and the average of $\log_{10}(|\mathbf{T}_j - \hat{\mathbf{T}}_j|) - \log_{10}(\epsilon)$ (the number of wrong digits) for these events. A 433 MHZ DEC Alpha workstation was used for the computations, with all algorithms implemented in double precision in FORTRAN.

We first present some test results, for $m = 2, 3, \ldots, 20$, for three methods which will be denoted MCAR, MCSR, MCSVT. These methods use simple Monte Carlo algorithms based on equations (8), (9) and (12), respectively. The results, obtained using $L = 100$ samples, are given in Figure 2. The number given in parentheses next to the name of each method is the $\bar{c}$ statistic for that method, averaged over all of the m values. All of the methods were reliable, terminating typically with a results that had approximately 2.8 correct decimal digits. We also tested the algorithms based on equation (10, with $n = 1$), denoted by MCSR1, and equation (14), denoted by MCSVN, but the results were similar to (with MCSR1 results slightly better than) the results for MCSVN at this accuracy level.

A standard method for improving for overall performance of MC algorithms is to use simple antithetic

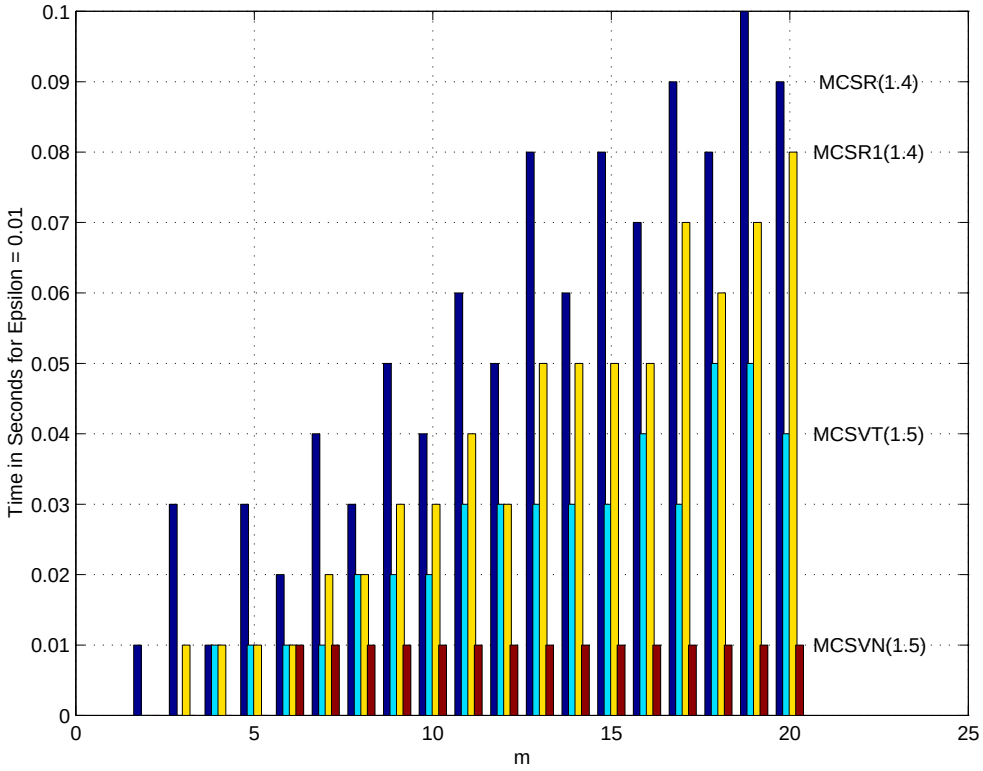

Figure 4: Average Monte Carlo Prioritized MVT Algorithm Times, $\epsilon = 10^{-2}$

variates. This method replaces $f(\mathbf{v}_k)$ by $(f(\mathbf{v}_k) + f(\mathbf{1} - \mathbf{v}_k))/2$, with $\mathbf{1} = (1, 1, \ldots, 1)'$, in the MC sum. In this case we redefine $\hat{\mathbf{T}}_N$ and σ_N to be given by

$$\hat{\mathbf{T}}_N = \frac{2}{N} \sum_{k=1}^{N/2} \frac{f(\mathbf{v}_k) + f(\mathbf{1} - \mathbf{v}_k)}{2} \quad \text{and} \quad \sigma_N^2 = \frac{2}{N(N/2-1)} \sum_{k=1}^{N/2} (\frac{f(\mathbf{v}_k) + f(\mathbf{1} - \mathbf{v}_k)}{2} - \hat{\mathbf{T}}_N)^2.$$

The methods that use equation (10) are already symmetrized. We tested the other algorithms, but with this "symmetrized" modification included, at the $\epsilon = 0.01$ accuracy level and the results (again with $L = 100$ samples) are given in Figure 3. The times are clearly lower for the symmetrized algorithms, with more significant improvements for the MCSR and MCSVN methods. The times for then MCAR method also showed some improvement, but they were still significantly higher than the times for the other methods. The clear loser at this accuracy level is the MCAR method (acceptance-rejection). We will not present any further results for this method, although we have carried out additional tests of MCAR and obtained similar results, when compared with other methods. We do not recommend this method for the efficient computation of MVT probabilities.

For a final test at this accuracy level we added the variable prioritization preconditioning step, described near the end of Section 2.3, to the symmetrized algorithms. Some of the results (based on 100 samples) are given in Figure 4. The times are even lower for all of the prioritized algorithms, with the most significant improvement for the MCSVN method. All of the tests described in the rest of this paper used symmetrized algorithms with prioritization.

In order to more carefully compare the MC methods we carried out another test at a higher accuracy level, using the methods MCSVT, MCSVN, MCSR, MVSR1 and MCSR2 (based on equation (10) with $n = 2$). Some of the results are given in Figure 5. All of the SR test results appear to exhibit the overall

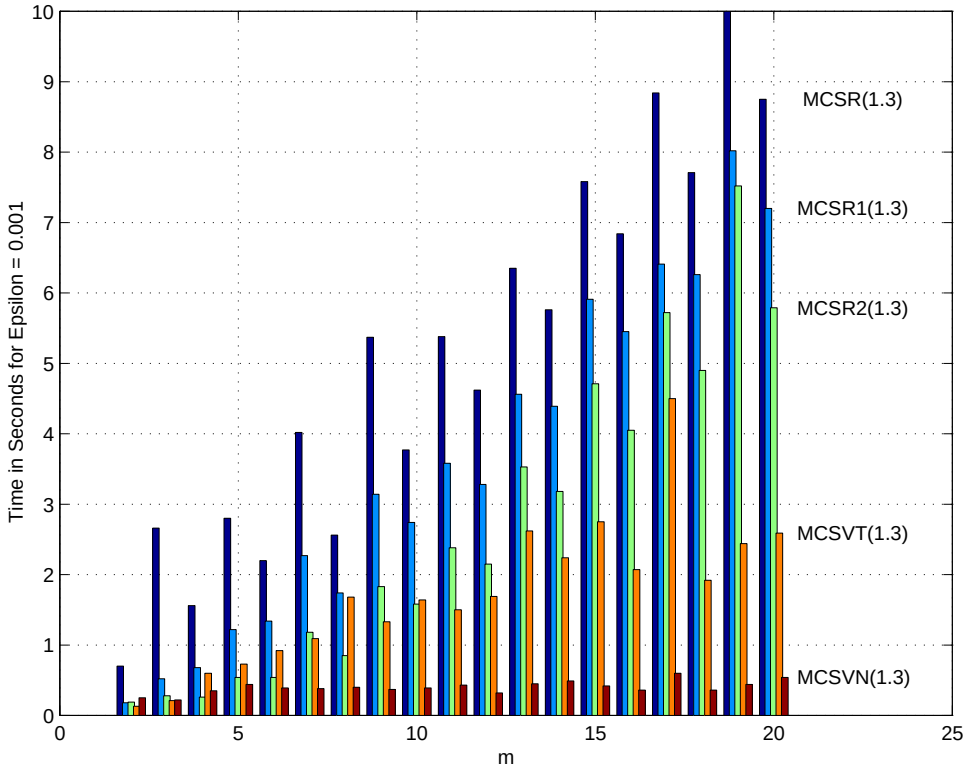

Figure 5: Average Monte Carlo MVT Algorithm Times, $\epsilon = 10^{-3}$

increase in time taken as function of dimension, but with times for the odd m larger than expected. This occurs because the F values needed for these methods require the evaluation of a series (determined from a succession of integrations by parts) which ends with an extra t_ν value when m is odd, and the computation of this t_ν value is what requires increases the time when m is an odd integer.

3.2 Quasi-Monte Carlo Algorithm Tests

Tests by Beckers and Haegemans (1992), and by Genz (1993), for MVN problems demonstrated that the performance of MC MVN methods could usually be improved if the sets of (pseudo-)random numbers used by the MC methods were replaced by appropriate sets of quasi-random numbers. If we wish to construct Quasi-Monte Carlo(QMC) MVT methods, we need only replace the $Uniform(0,1)$ random numbers in our MC methods by appropriately chosen sets of $Quasi(0,1)$ random numbers. All of our MC methods are implemented as methods that use only $Uniform(0,1)$ random numbers, However, simple QMC methods do not provide the statistically robust (standard)error estimates that MC methods do provide, so we decided to use randomized QMC algorithms. The QMC methods that we have implemented use approximations to $I(f)$ in the form

$$\hat{\mathbf{T}}_{N,P} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{2P}\sum_{j=1}^{P}(f(|2\{\mathbf{p}_j + \mathbf{w}_i\} - \mathbf{1}|) + f(\mathbf{1} - |2\{\mathbf{p}_j + \mathbf{w}_i\} - \mathbf{1}|)). \tag{15}$$

In this definition, $\{\mathbf{x}\}$ denotes the vector obtained by taking the fractional part of each of the components of $\mathbf{x}$, $w_{k,i} \sim Uniform(0,1)$ and $\mathbf{p}_j, j = 1,2,\ldots,P$ is a set of quasi-random points. If the dimension of $\mathbf{v}$ is m or $m-1$, we use good lattice point sets (see Sloan and Joe, 1994, and Hickernell, 1998) for the

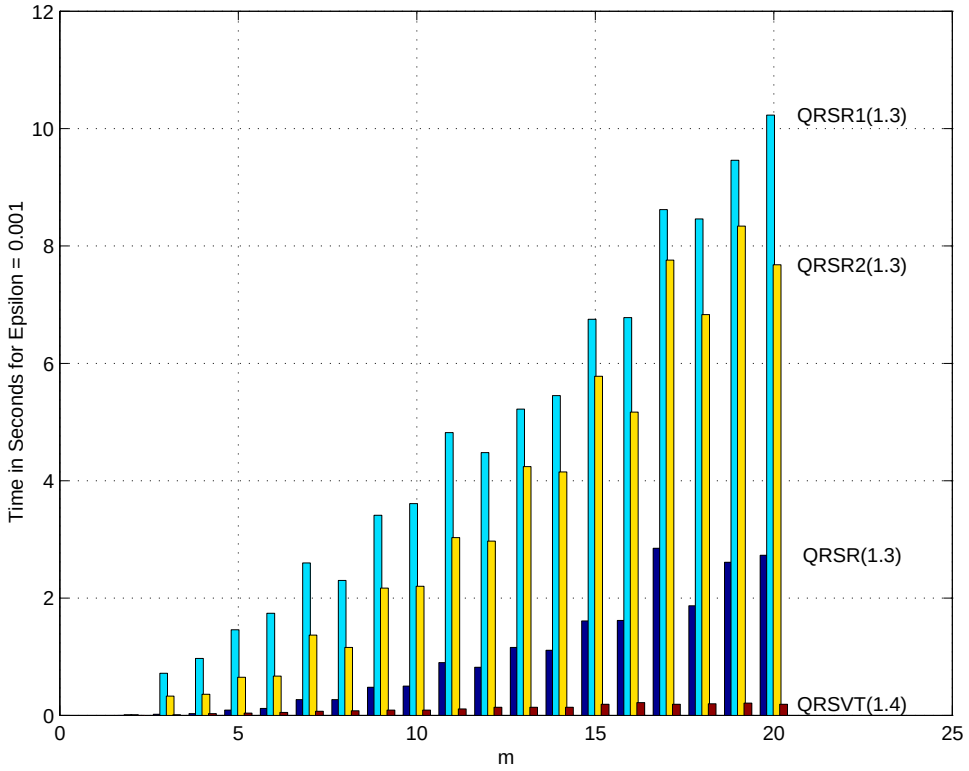

Figure 6: Average Quasi-Monte Carlo MVT Algorithm Times, $\epsilon = 10^{-3}$

required quasi-random point sets. If the dimension of $\mathbf{v}$ is $\frac{m(m-1)}{2}$, we use "Richtmeyer sequence" point sets (see Davis and Rabinowitz, 1984, pp. 482–483) for the dimensions greater than 100. The Richtmeyer points are defined by $p_{k,j} = \{j\sqrt{q_{k-100}}\}$, for $k = 101, 102, \ldots, \frac{m(m-1)}{2}$, $j = 1, 2, \ldots, P$, where q_i is the i^{th} prime number (starting with $q_1 = 2$). The "periodizing" transformation $|2\mathbf{v} - \mathbf{1}|$ is included because the quasi-Monte Carlo rules that we use have better convergence properties for periodic integrands. If we let $Q_P(\mathbf{w}) = \frac{1}{2P}\sum_{j=1}^{P}(f(|2\{\mathbf{p}_j + \mathbf{w}\} - \mathbf{1}|) + f(\mathbf{1} - |2\{\mathbf{p}_j + \mathbf{w}\} - \mathbf{1}|))$, then the standard error σ_N for the approximation (15) can be determined using $\sigma_N^2 = \frac{1}{N(N-1)}\sum_{i=1}^{N}(Q_P(\mathbf{w}_i) - \hat{\mathbf{T}}_{N,P})^2$. We have implemented the quasi-Monte Carlo methods using the following algorithm.

Quasi-Monte Carlo Algorithm

1. **Input** $\mathbf{a}$, $\mathbf{b}$, $\boldsymbol{\Sigma}$, ν, $\boldsymbol{\delta}$, ϵ, and N_{max};.
2. **Set** $N = N_{min}$, $i = 0$;
3. **Compute** $\hat{\mathbf{T}}_{\bar{N}}$ and $\sigma_{\bar{N}}$;
4. **Set** $\bar{N} = 2NP_0$, $\bar{\mathbf{T}} = \hat{\mathbf{T}}_{N,P_0}$, $\bar{\sigma} = \sigma_N$, $\hat{\epsilon} = 3\bar{\sigma}$;
5. **Do While** ($\hat{\epsilon} > \epsilon$ **and** $\bar{N} < N_{max}$)
   (a) **Set** $i = i + 1$;
   (b) **Compute** $\hat{\mathbf{T}}_{N,P_i}$ and σ_N.
   (c) **Set** $\bar{N} = \bar{N} + 2NP_i$, $\bar{\mathbf{T}} = \bar{\mathbf{T}} + \bar{\sigma}^2(\hat{\mathbf{T}}_{N,P_i} - \bar{\mathbf{T}})/(\bar{\sigma}^2 + \sigma_N^2)$, $\bar{\sigma}^2 = \bar{\sigma}^2\sigma_N^2/(\bar{\sigma}^2 + \sigma_N^2)$, $\hat{\epsilon} = 3.5\bar{\sigma}$;

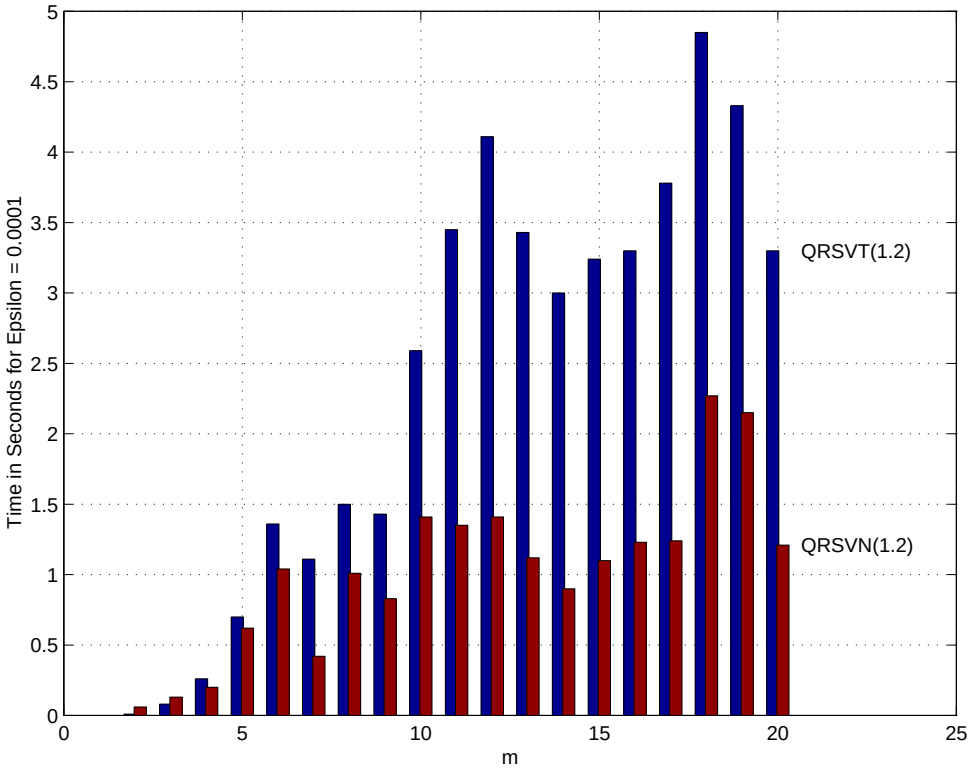

Figure 7: Average Quasi-Monte Carlo MVT Algorithm Times, $\epsilon = 10^{-4}$

End Do

6. **Output** $\bar{\hat{\mathbf{T}}}$, $\bar{N}$, $\hat{\epsilon}$.

The sequence $P_0, P_1, \ldots$ is a sequence of primes starting with $P_0 = 31$, with $P_{i+1} \approx 3P_i/2$. All of our QMC algorithms use $N_{min} = 8$, with $\hat{\epsilon} = 3.5\bar{\sigma}$ (using 3.5 instead of 3 because of the smaller N_{min}).

We first tested the quasi-Monte Carlo algorithms (with prioritization included) at the $\epsilon = 10^{-3}$ accuracy level. Some of the results (based on 100 samples) are given in Figure 6, with QRSVN results omitted because they were similar to QRSVT results. The times are significantly lower than the MC times for the QRSR, QRSVT and QRSVN algorithms, with clear winners QRSVT and QRSVN. We believe that there was no significant improvement in the QRSR1 and QRSR2 algorithm times, compared to MCSR1 and MCSR2 times because the MCSR1 and MCSR2 algorithms can themselves be considered randomized quasi-random algorithms (based on the $S_n(Z)$ rules which use evenly distributed spherical surface points), so the additional "quasi-randomization" of the Z matrices does not produce a significant improvement in algorithm performance.

We conducted an additional test of the quasi-Monte Carlo algorithms QRSVT and QRSVN at the $\epsilon = 10^{-4}$ accuracy level. The results (based on 100 samples) are given in Figure 7. The times are significantly lower for the QRSVN algorithm. We were surprised by this result, because the SVN method is based on replacing an m-dimensional problem with an $m+1$-dimensional problem. An explanation for this difference comes from the fact that each QRSVN f value requires m Φ values, $m-1$ Φ^{-1} values, and one χ^{-1} value, but each QRSVT f value requires m t values and $m-1$ t^{-1} values. The t values, using $\nu, \nu+1, \ldots, \nu+m-1$ degrees of freedom, are computed using integration by parts so that t_k uses a sum of $k/2$ terms. Some t values are also used in the t^{-1} evaluations, so the the work for the 1-dimensional distribution evaluation for one f value for QRSVT is $O(m^2)$. For the QRSVN method, the χ^{-1} time is $O(\nu)$, but the individual Φ and

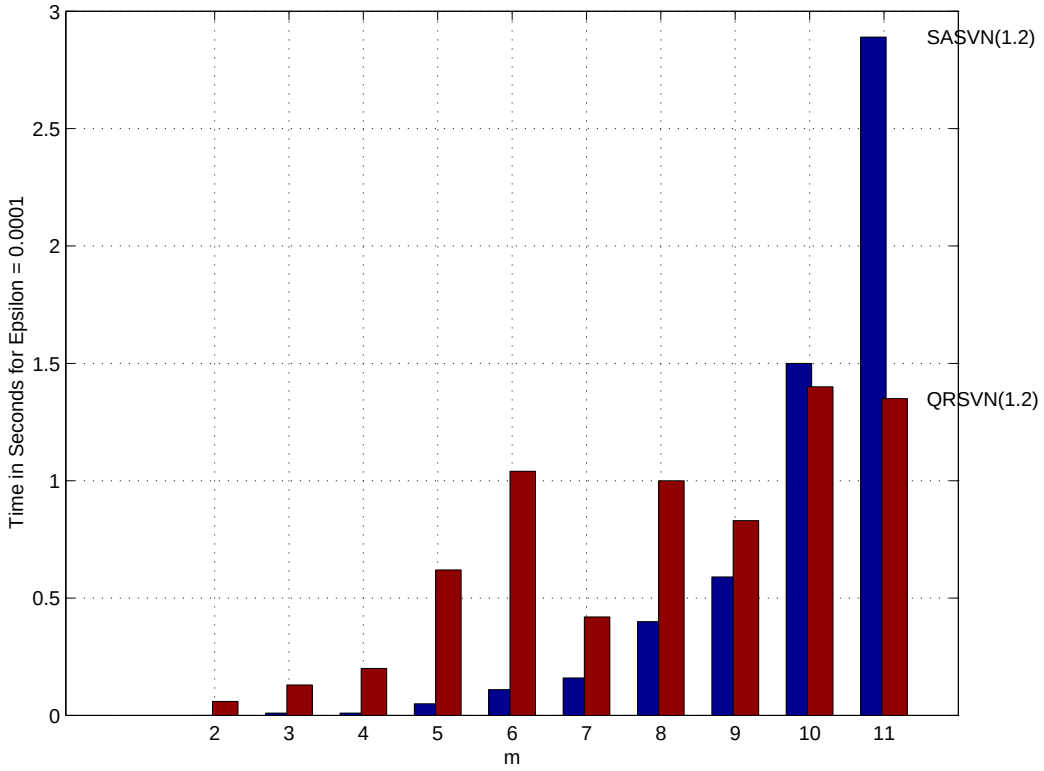

Figure 8: Average MVT Algorithm Times, $\epsilon = 10^{-4}$

Φ^{-1} times are independent of m, so the time one f value for the QRSVN method is only $O(m + \nu)$. We think these f value time complexity differences explain the increasing difference between the SVN and SVT times as m increases.

We also conducted some tests with some subregion adaptive algorithms, SASVN and SASVT, at the $\epsilon = 10^{-4}$ accuracy level. These algorithms use a subregion adaptive integration method, similar to the one that was effective for the lower dimensional MVN problems (see Genz, 1992, 1993, and Berntsen, Espelid and Genz, 1991), applied to the respective SV-Chi-Normal and SV-t formulations of the MVT problem. The results (based on 100 samples, for $m = 2, \ldots, 11$) are given in Figure 8. The times for the SASVN are significantly lower than the QRSVN for dimensions 2-8, but after that the SASVN times increase rapidly, usually exceeding the QRSVN times for $m > 10$. The SASVT times (not shown in Figure 8) exhibited similar behavior, but they were consistently larger than the SASVN times, usually exceeding the QRSVN times for $m > 8$.

These results provide strong evidence that multivariate t-probabilities can be robustly and reliably computed at low to moderate accuracy levels in less than a second of workstation time for problems with up to twenty dimensions. The symmetrization, variable prioritization, and quasi-randomization techniques all produce clearly observable improvements in algorithm performance. When moderate accuracy is required, the QRSVN method can be significantly faster than the other methods, except for $m < 10$, where the SASVN method can be faster. Software for all of the methods discussed here is available from the authors.

References

Beckers, M. and Haegemans, A. (1992) 'Comparison of Numerical Integration Techniques for Multivariate Normal Integrals', Computer Science Department preprint, Catholic University of Leuven, Belgium.

Berntsen, J., Espelid, T. O. and Genz, A. (1991) 'Algorithm 698: DCUHRE-An Adaptive Multidimensional Integration Routine for a Vector of Integrals', *ACM Transactions on Mathematical Software* **17**, pp. 452–456.

Cornish, E. A. (1954) 'The Multivariate t-Distribution Associated with a Set of Normal Sample Deviates' *Australian Journal of Physics* **7**, pp. 531–542.

Cranley, R. and Patterson, T. N. L. (1976) 'Randomization of Number Theoretic Methods for Multiple Integration', *SIAM J. Numer. Anal.* **13**, pp. 904–914.

Davis, P. J. and Rabinowitz P. (1984), *Methods of Numerical Integration*, Academic Press, New York.

Deák, I. (1980) 'Three Digit Accurate Multiple Normal Probabilities' *Numer. Math.* **35**, pp. 369–380.

Deák, I. (1986) 'Computing Probabilities of Rectangles in Case of Multinormal Distribution' *J. Statist. Comput. Simul.* **26**, pp. 101–114.

Deák, I. (1990) *Random Number Generation and Simulation*, Akadémiai Kiadó, Budapest, Chapter 7.

Fang, K.-T., and Wang, Y. (1994) *Number-Theoretic Methods in Statistics*, Chapman and Hall, London, pp. 167–170.

Genz, A. (1992) 'Numerical Computation of the Multivariate Normal Probabilities', *J. Comput. Graph. Stat.* **1**, pp. 141–150.

Genz, A. (1993), 'A Comparison of Methods for Numerical Computation of Multivariate Normal Probabilities', *Computing Science and Statistics* **25**, pp. 400–405.

Genz, A. and Kwong, K. S. (1999) 'Numerical Evaluation of Singular Multivariate Normal Distributions', submitted.

Genz, A. and Bretz, F. (1999) 'Numerical Computation of the Multivariate t Probabilities with Application to Power Calculation of Multiple Contrasts', *Journal of Statistical Computation and Simulation* **63**, pp. 361–378.

Gibson, G. J., Glasbey, C. A. and Elston, D. A. (1992) 'Monte-Carlo Evaluation of Multivariate Normal Integrals', Scottish Agricultural Statistics Service preprint, University of Edinburgh, Scotland.

Hajivassiliou, V., McFadden, D. and Rudd, O. (1996). 'Simulation of Multivariate Normal Rectangle Probabilities and Their Derivatives: Theoretical and Computational Results', *Journal of Econometrics*, **72**, pp. 85–134.

Hickernell, F. J. (1998). 'A Generalized Discrepancy and Quadrature Error Bound', *Mathematics of Computation*, **67**, pp. 299–322.

Hsu, Jason C. (1996). *Multiple Comparisons*, Chapman and Hall, London.

Joe, S. (1995). 'Approximations to Multivariate Normal Rectangle Probabilities Based on Conditional Expectations', *Journal of the American Statistical Association*, **90**, pp. 957–964.

Johnson, Mark E. (1987). *Multivariate Statistical Simulation*, Wiley, New York.

Keast, P. (1973) 'Optimal Parameters for Multidimensional Integration', *SIAM J. Numer. Anal.* **10**, pp. 831–838.

Lepage, G. Peter (1978) 'A New Algorithm for Adaptive Multidimensional Integration', *J. Computational Physics* **27**, pp. 192–203.

Lohr, S. (1990) 'Accurate Multivariate Estimation using Triple Sampling', *Ann. Statist.* **18**, pp. 1615–1633.

Marsaglia, G. and Olkin, I. (1984) 'Generating Correlation Matrices', *SIAM Journal of Scientific and Statistical Computing* **5**, pp. 470–475.

Schervish, M. (1984) 'Multivariate Normal Probabilities with Error Bound', *Applied Statistics* **33**, pp. 81–87.

Sloan, I. H., and Joe, S. (1994) *Lattice Methods for Multiple Integration*, Oxford University Press, Oxford.

Somerville, P. N. (1997) 'Multiple Testing and Simultaneous Confidence Intervals: Calculation of Constants' *Comp. Stat. & Data Analysis* **25**, pp. 217–223.

Somerville, P. N. (1998)) 'Numerical Computation of Multivariate Normal and Multivariate-t Probabilities Over Convex Regions', *J. Comput. Graph. Stat.* **7**, pp. 529–545.

Somerville, P. N. (1999) 'Critical Values for Multiple Testing and Comparisons: One Step and Step Down Procedures' *J. Stat. Plan. & Inf.* **82**, pp. 129–138.

Stewart, G. W. (1980), 'The Efficient Generation of Random Orthogonal Matrices with An Application to Condition Estimation', *SIAM J. Numer. Anal.* **17**, 403–409.

Tong, Y. L. (1990) *The Multivariate Normal Distribution*, Springer-Verlag, New York.