MVT

A GAUSS/DLL Implementation of Alan Genz's Fortran Packages for Computing Multivariate t CDF

Thierry Roncalli

Groupe de Recherche Opérationnelle, Crédit Lyonnais, France

June 2001

1 Introduction

MVT is a GAUSS library for computing multivariate t cdf. It is based on the Fortran packages mvt.f and mvtdstpack.f written by Alan Genz. The Fortran source code is available from his web page:

`www.sci.wsu.edu/math/faculty/genz/homepage`

The Fortran packages contain different subroutines to compute t-probabilities. The algorithms of these subroutines are described in the following articles:

Genz, A. and F. Bretz [1999], Numerical computation of the multivariate t-probabilities with applications to power calculation of multiple contrasts, *Journal of Statistical Computation and Simulation*, **63(4)**, 361-378 (available from `www.sci.wsu.edu/math/faculty/genz/homepage`)

Genz, A. and F. Bretz [2000], Comparison of methods for the computation of multivariate t-probabilities, submitted (available from `www.sci.wsu.edu/math/faculty/genz/homepage`)

Remark 1 *Any research use of the* **MVT** *library should cite the two previous articles. Moreover, it is submitted with no performance guarantees.*

2 Installation

2.1 Transferring the files

1. The file *mvt.zip* is a zipped archive file. Copy this file under the root directory of GAUSS, for example **C:\GAUSS**.

2. Unzip it with archive mode. It is automatically recognized by WinZip. With Unzip or PKunzip, use the -d flag

   pkunzip -d mvt.zip

   Directories will then be created and files will be copied over them:

target_path	*readme.mvt*
target_path\dlib	Dynamic link libraries
target_path\lib	Library file
target_path\mvt	Examples and tutorial files
target_path\mvt\dlls	Fortran/C source code files
target_path\src	GAUSS source code files

2.2 Getting started

Gauss 3.2+ for OS/2 or Windows NT/95 is required to use the **MVT** routines.

2.3 The file *readme.mvt*

The file *readme.mvt* contains last minute information on the **MVT** procedures. Please read it before using them.

2.4 Setup

In order to use these procedures, the **MVT** library must be active. This is done by including **MVT** in the `LIBRARY` statement at the top of your program:

library mvt;

To reset global variables in subsequent executions of the program, the following instruction should be used:

MVTset;

If you plan to make any right-hand reference to the global variables, you will also need the statement:

#include *target_path***src****mvt.ext;**

The **MVT** library uses two dynamic link libraries *mvt1.dll* and *mvt2.dll*. You have to declare these DLLs with the following command:

dlibrary mvt1.dll,mvt2.dll;

Nevertheless, if you use the `MVTset` command at the top of your program, it is done automatically.

The **MVT** version number is stored as a global variable:

_mvt_ver	3×1 matrix where the first element indicates the major version number, the second element the minor version number, and the third element the revision number

2.5 Using Online Help

MVT library supports Windows Online Help. Before using the browser, you have to verify that the **MVT** library is activated by the `library` command.

3 What is MVT?

MVT is a GAUSS library for computing multivariate t cdf. **MVT** contains the procedures whose list is given below. See the command reference part for a full description.

- **cdfbvt**: Computes bivariate t cdf.
- **cdfmvt**: Computes multivariate t cdf.
- **cdfmvtnc**: Computes non-central multivariate t cdf.
- **mvtset**: Resets the global variables and loads the DLL files.

Remark 2 *I have translated the Fortran packages mvt.f and mvtdstpack.f written by Alan Genz into C with f2c. There exist others procedures than the previous ones that may be used directly:*

GAUSS procedure	*DLL wrapper function*	*Fortran subroutine*
_cdfbvt	*dllmvbvt*	*mvbvt*
_cdfmvtnc_QRSVN	*dllqrsvnmvt*	*mvtdst*
_cdfmvt_RAN	*dllranmvt*	*ranmvt*
_cdfmvt_KRO	*dllkromvt*	*kromvt*
_cdfmvt_SAD	*dllsadmvt*	*sadmvt*
_cdfmvt_SPH	*dllsphmvt*	*sphmvt*

4 Command reference

The following global variables and procedures are defined in **MVT**. They are the *reserved words* of **MVT**.

`cdfbvt, _cdfbvt, cdfmvt, _cdfmvt_kro, _cdfmvt_ran, _cdfmvt_sad, _cdfmvt_sph, cdfmvtnc, _cdfmvtnc_qrsvn, _mvt_abs_eps, _mvt_algr, _mvt_rel_eps, _mvt_max_pts, mvtset, _mvt_ver`

The default global control variables are

`_mvt_abs_eps`	0
`_mvt_algr`	1
`_mvt_rel_eps`	0.005
`_mvt_max_pts`	0

cdfbvt

■ Purpose

Computes bivariate t cdf.

■ Format

cdf = cdfbvt(x,y,rho,nu);

■ Input

x	$M \times N$ matrix, upper integration bound for variable 1
y	$M \times N$ matrix, upper integration bound for variable 2
rho	$M \times N$ matrix, correlation coefficient
nu	scalar, number of degrees of freedom

■ Output

cdf	$M \times N$ matrix, cdf value

■ Remark

This procedure uses the Fortran subroutine `MVBVT` written by Alan Genz for the Fortran package MVTDSTPACK. The algorithm is based on the method of Dunnet and Sobel [1954].

The matrices (`x`, `y` and `rho`) may be $E \times E$ conformable.

■ Reference

Dunnet, C.W. and M. Sobel [1954], A bivariate generalization of Student's t-distribution, with tables for certain special cases, *Biometrika*, **41**, 153-169

■ Source

src/mvt.src

cdfmvt

Purpose
Computes multivariate t cdf.

Format
{cdf,err,retcode} = cdfmvt(x,rho,nu);

Input

x	$M \times N$ matrix, upper integration bounds
rho	$N \times N$ matrix, correlation matrix
nu	scalar, number of degrees of freedom

Output

cdf	$M \times 1$ vector, cdf value
err	$M \times 1$ vector, estimated absolute error with 99% confidence level
retcode	$M \times 1$ vector, return code
	0 if normal completion (err ¡ _mvt_abs_eps)
	1 if err ¿ _mvt_abs_eps
	2 if $N < 1$ or $N > N^+$

Globals

_mvt_abs_eps	absolute error tolerance (default = 0.000)
_mvt_algr	scalar, algorithm (default = 1)
	1 for the QRSVN algorithm ($N^+ = 100$)
	2 for the RAN algorithm ($N^+ = 20$)
	3 for the KRO algorithm ($N^+ = 20$)
	4 for the SAD algorithm ($N^+ = 20$)
	5 for the SPH algorithm ($N^+ = 50$)
_mvt_max_pts	scalar, maximum number of function values allowed (default = 0 —¿ _mvt_max_pts = $1000 \times N$)
_mvt_rel_eps	scalar, relative error tolerance (default = 0.005)

Remark
This procedure uses the different Fortran subroutines written by Alan Genz for the Fortran packages MVT and MVTDSTPACK:

Algorithm	Description
QRSVN	Quasi-Monte Carlo with symmetrization and prioritization, applied to the $\chi - \Phi$ formulation (GENZ and BRETZ [2000])
RAN	Crude Monte-Carlo scheme with simple antithetic variates and weighted results on restart (GENZ and BRETZ [1999])
KRO	Multidimensional integration scheme with randomized Korobov rules (GENZ and BRETZ [1999])
SAD	Subregion adaptive integration scheme (GENZ and BRETZ [1999])
SPH	Modified version of the Monte Carlo method proposed by DEAK [1990] (GENZ and BRETZ [1999])

References
DEAK, I. [1990], Random number generators and simulation, Akademiai Kiado, Budapest

GENZ, A. and F. BRETZ [2000], Comparison of methods for the computation of multivariate t-probabilities, *submitted*

GENZ, A. and F. BRETZ [1999], Numerical computation of the multivariate t-probabilities with applications to power calculation of multiple contrasts, *Journal of Statistical Computation and Simulation*, **63(4)**, 361-378

Source
src/mvt.src

cdfmvtnc

Purpose

Computes non-central multivariate t cdf.

Format

{cdf,err,retcode} = cdfmvtnc(x,rho,nu,delta);

Input

x	$M \times N$ matrix, upper integration bounds
rho	$N \times N$ matrix, correlation matrix
nu	scalar, number of degrees of freedom
delta	$N \times 1$ vector, non-centrality parameters

Output

cdf	$M \times 1$ vector, cdf value
err	$M \times 1$ vector, estimated absolute error with 99% confidence level
retcode	$M \times 1$ vector, return code
	0 if normal completion (err ¡ _mvt_abs_eps)
	1 if err ¿ _mvt_abs_eps
	2 if $N > 100$ or $N < 1$
	3 if the correlation matrix is not positive semi-definite

Globals

_mvt_abs_eps	absolute error tolerance (default = 0.000)
_mvt_max_pts	scalar, maximum number of function values allowed (default = 0 —¿ _mvt_max_pts = 1000 × N)
_mvt_rel_eps	scalar, relative error tolerance (default = 0.005)

Remark

This procedure uses the subroutine MVTDST written by Alan Genz for the Fortran package MVTDST-PACK. The algorithm is based on the method QRSVN (quasi-Monte Carlo with symmetrization and prioritization, applied to the $\chi - \Phi$ formulation) of Genz and Bretz [2000].

Reference

Genz, A. and F. Bretz [2000], Comparison of methods for the computation of multivariate t-probabilities, *submitted*

Source

src/mvt.src

5 Tutorial

- In this first example, we compare the accuracy of the **MVT** procedures `cdfmvt` and `cdfmvtnc` with the GAUSS procedures `cdftc` and `cdftnc`.

```
new;
library mvt;

mvtset;

cls;

output file = mvt1.out reset;

/*
** The case N = 1
**
*/


x = seqa(-3,0.5,13);
rho = 1;
nu = 2;

_mvt_algr = 1;
{cdf1,err,retcode} = cdfmvt(x,rho,nu);
cdf2 = 1 - cdftc(x,nu);

print ''     Upper    cdfmvt      cdft     diff.'';
print ''========================================'';
call printfm(x~cdf1~cdf2~(cdf1-cdf2),1,''%lf''~10~3);
print;
print;


/*
** The case N = 1 (non-centered)
**
*/

delta = 1.5;

_mvt_algr = 1;
{cdf1,err,retcode} = cdfmvtnc(x,rho,nu,delta);
cdf2 = cdftnc(x,nu,delta);

print ''     Upper  cdfmvtnc    cdftnc     diff.'';
print ''========================================'';
call printfm(x~cdf1~cdf2~(cdf1-cdf2),1,''%lf''~10~3);
print;
print;

output off;
```

```
     Upper    cdfmvt      cdft     diff.
========================================
    -3.000     0.048     0.048    -0.000
    -2.500     0.065     0.065     0.000
    -2.000     0.092     0.092     0.000
    -1.500     0.136     0.136    -0.000
    -1.000     0.211     0.211    -0.000
    -0.500     0.333     0.333     0.000
     0.000     0.500     0.500     0.000
     0.500     0.667     0.667    -0.000
     1.000     0.789     0.789     0.000
     1.500     0.864     0.864     0.000
     2.000     0.908     0.908    -0.000
     2.500     0.935     0.935     0.000
     3.000     0.952     0.952     0.000


     Upper  cdfmvtnc    cdftnc     diff.
========================================
    -3.000     0.002     0.002    -0.000
    -2.500     0.003     0.003     0.000
    -2.000     0.005     0.005     0.000
```

```
   -1.500     0.008     0.008    -0.000
   -1.000     0.014     0.014    -0.000
   -0.500     0.029     0.029     0.000
    0.000     0.067     0.067     0.000
    0.500     0.152     0.152    -0.000
    1.000     0.287     0.287     0.000
    1.500     0.436     0.436     0.000
    2.000     0.566     0.566    -0.000
    2.500     0.666     0.666    -0.000
    3.000     0.740     0.740    -0.000
```

- The second example considers bivariate t distributions.

```
new;
library mvt;

mvtset;

cls;

output file = mvt2.out reset;

/*
** The case N = 2
**
*/


x = rndn(10,2);
let rho[2,2] = 1 0.25 0.25 1;
nu = 2;

_mvt_algr = 1;
{cdf1,err,retcode} = cdfmvt(x,rho,nu);
cdf2 = cdfbvt(x[.,1],x[.,2],rho[1,2],nu);

print ''    var. 1    var. 2    cdfmvt    cdfbvt     diff.'';
print ''===================================================='';
call printfm(x~cdf1~cdf2~(cdf1-cdf2),1,''%lf''~10~3);
print;
print;


nu = 100000; /* Asymtotic case --> BVN */

cdf1 = cdfbvt(x[.,1],x[.,2],rho[1,2],nu);
cdf2 = cdfbvn(x[.,1],x[.,2],rho[1,2]);

print ''    var. 1    var. 2    cdfbvt    cdfbvn     diff.'';
print ''===================================================='';
call printfm(x~cdf1~cdf2~(cdf1-cdf2),1,''%lf''~10~3);

output off;
```

```
    var. 1    var. 2    cdfmvt    cdfbvt     diff.
==================================================
   -0.700    -2.201     0.042     0.042     0.000
   -0.145     1.232     0.391     0.391     0.000
    0.187    -0.124     0.296     0.296     0.000
    0.124     0.184     0.347     0.347     0.000
   -0.359     1.833     0.346     0.346     0.000
    0.196     0.233     0.371     0.371     0.000
    1.022    -0.768     0.216     0.216     0.000
    0.831    -0.594     0.246     0.246     0.000
    0.996    -1.852     0.081     0.081     0.000
   -1.941     0.351     0.067     0.067    -0.000


    var. 1    var. 2    cdfbvt    cdfbvn     diff.
==================================================
   -0.700    -2.201     0.007     0.007     0.000
   -0.145     1.232     0.412     0.412    -0.000
    0.187    -0.124     0.298     0.298    -0.000
    0.124     0.184     0.354     0.354    -0.000
   -0.359     1.833     0.354     0.354    -0.000
    0.196     0.233     0.381     0.381    -0.000
    1.022    -0.768     0.203     0.203     0.000
    0.831    -0.594     0.242     0.242     0.000
    0.996    -1.852     0.030     0.030     0.000
   -1.941     0.351     0.022     0.022     0.000
```

- In the following example, we compute multivariate t cdf. Moreover, we compare the algorithms QRSVN and SPH and the GAUSS procedure `cdfmvn` for the limiting case $N = \infty$.

```
new;
library mvt;

mvtset;

cls;

output file = mvt3.out reset;

/*
** The case N > 2
**
*/

N = 5;

x = rndn(10,5);

print ''x = '';
call printfm(x,1,''%lf''~10~3);
print;

rho = diagrv(0.75*ones(N,N),1);
nu = 3;

_mvt_algr = 1;
{cdf1,err,retcode} = cdfmvt(x,rho,nu);
_mvt_algr = 2;
{cdf2,err,retcode} = cdfmvt(x,rho,nu);
_mvt_algr = 3;
{cdf3,err,retcode} = cdfmvt(x,rho,nu);
_mvt_algr = 4;
{cdf4,err,retcode} = cdfmvt(x,rho,nu);
_mvt_algr = 5;
{cdf5,err,retcode} = cdfmvt(x,rho,nu);

print;
print ''     QRSVN       RAN       KRO       SAD       SPH'';
print ''==================================================='';
call printfm(cdf1~cdf2~cdf3~cdf4~cdf5,1,''%lf''~10~4);
print;
print /flush '''';

nu = 1000;    /* Asymtotic case --> MVN */

_mvt_algr = 1;
{cdf1,err,retcode} = cdfmvt(x,rho,nu);
_mvt_algr = 5;
{cdf2,err,retcode} = cdfmvt(x,rho,nu);
cdf3 = cdfmvn(x',rho)';

print;
print ''     QRSVN       SPH    CDFMVN'';
print ''=============================='';
call printfm(cdf1~cdf2~cdf3,1,''%lf''~10~4);
print;
print;

output off;
```

```
x =
     0.102     0.059    -0.759     1.545    -0.347
     2.316    -0.741     1.440    -1.850    -0.207
     0.134    -1.182     0.001    -0.971     0.758
     0.940    -0.005     0.441    -1.145     0.143
     0.221    -0.384     0.008     1.190    -0.823
    -1.382     0.500    -1.481    -1.010     0.359
     0.136    -0.078    -1.791     0.516     0.023
     0.399    -0.474    -1.248    -0.477     1.383
     0.222     0.440    -1.072    -0.575    -1.131
    -0.048     1.623     0.021     1.427     0.688


     QRSVN       RAN       KRO       SAD       SPH
===================================================
    0.1791    0.1798    0.1790    0.1791    0.1788
    0.0644    0.0645    0.0643    0.0644    0.0651
    0.1010    0.1015    0.1014    0.1014    0.1020
    0.1400    0.1401    0.1401    0.1401    0.1404
    0.1694    0.1690    0.1689    0.1690    0.1687
    0.0601    0.0601    0.0601    0.0600    0.0601
    0.0720    0.0722    0.0721    0.0721    0.0719
    0.1088    0.1088    0.1089    0.1090    0.1084
    0.0929    0.0927    0.0927    0.0927    0.0922
    0.3632    0.3638    0.3636    0.3638    0.3653



     QRSVN       SPH    CDFMVN
===============================
    0.1677    0.1700    0.1676
    0.0290    0.0292    0.0289
    0.0753    0.0756    0.0750
    0.1173    0.1169    0.1172
    0.1559    0.1551    0.1558
    0.0313    0.0314    0.0312
    0.0359    0.0362    0.0358
    0.0834    0.0836    0.0833
    0.0673    0.0674    0.0673
    0.3712    0.3700    0.3712
```

- The last example compares central and non-central multivarariate t cdf.

```
new;
library mvt;

mvtset;

cls;

output file = mvt4.out reset;

x = 2;
delta = 0;
rho = diagrv(0.75*ones(5,5),1);

_mvt_max_pts = 50000;
_mvt_rel_eps = 0.0005;

Nu = 0;
do until Nu > 10;
  _mvt_algr = 1;
  {cdf,err,retcode} = cdfmvt(x,rho,nu);

  omat = Nu~cdf~err~retcode;
  call printfmt(omat,1);

  if Nu == 0;
    cdf = cdfmvn(x*ones(5,1),rho);
    print ftos(cdf,'' (%lf)'',10,5);;
  endif;

  print /flush '''';

  Nu = Nu + 1;
endo;

print; print;

delta = 1;

Nu = 0;
do until Nu > 10;
  {cdf,err,retcode} = cdfmvtnc(x,rho,nu,delta);

  omat = Nu~cdf~err~retcode;
  call printfmt(omat,1);
  print /flush '''';
  Nu = Nu + 1;
endo;

output off;
```

```
         0       0.93620156    0.00038848494              0  (0.93637)
         1       0.73504636    0.00031154922              0
         2       0.81692546    0.00038484975              0
         3       0.85167474    0.00041356607              0
         4       0.87108371    0.00042241636              0
         5       0.88308315    0.00031839536              0
         6       0.89146412    0.00034222936              0
         7        0.8977878    0.00032228211              0
         8       0.90220423     0.0003567237              0
         9       0.90582369    0.00036929418              0
        10       0.90899545    0.00043249478              0


         0       0.67465457    0.00019074785              0
         1       0.46674189    0.00016593729              0
         2       0.54283526    0.00022162004              0
         3       0.57726951    0.00020307979              0
         4       0.59733918    0.00027365415              0
         5       0.61056763    0.00024595136              0
         6       0.61967699    0.00027820285              0
         7       0.62677454     0.0002101323              0
         8       0.63209914    0.00019705824              0
         9       0.63635275     0.0002873238              0
        10       0.63978005     0.0002438075              0
```