Universiteit van Amsterdam

Amsterdam School of Economics

**Computer class NLIST—Assignment 1A**

# Generating multinormal and multi-Student r.v.'s

In this assignment, you learn about:

- How `R` draws normal pseudo-random variables
- Simulating and plotting Brownian motions in one and two dimensions
- How to draw a bivariate normal vector with prescribed correlation (by a method that is in fact a special case of the Cholesky method)
- Plotting a scattergram
- Multi-Student distributions; a multivariate Student$(\vec{\mu}, \mathbf{\Sigma}, k)$ random variable results by taking $\vec{T} = \vec{\mu} + \vec{Z}/\sqrt{U/k}$, where vector $\vec{Z} \sim \mathrm{N}(\vec{0}, \mathbf{\Sigma})$ and scalar $U \sim \chi^2(k)$ are independent
- The concept of 'tail dependence', see e.g. here
- The difference in tail dependence between multinormal and multi-Student r.v.'s, see for example here
- Drawing multinormal samples using the Cholesky method
- Estimating and computing multinormal and multi-Student stop-loss premiums and Expected Shortfalls

## 1 Drawing independent normal random variables; Brownian motion

To draw iid Uniform(0,1) pseudo-random variables $U_1, U_2, \ldots$ is easy, using `runif`, or in Matlab, `unifrnd`. As is known, if $\Phi$ is the standard normal cdf, then $\Phi^{-1}(U) \sim \mathrm{N}(0,1)$ (inversion method, MART Sec. 3.9.1), because

$$\Pr[\Phi^{-1}(U) \leq x] = \Pr[U \leq \Phi(x)] = \Phi(x).$$

`R`'s `rnorm` function can be demonstrated to use a variant of this method to draw N(0,1) r.v.'s. Consider the output of the following:

```
set.seed(1); nor  <- qnorm(runif(5))
set.seed(1); nor1 <- rnorm(3)
nor; nor1; nor[c(1,3,5)] - nor1
```

First we set the random seed, to ensure that both times we start at the same position in the long stream of 'random' numbers generated by the software. As you see, the odd-numbered elements of `nor` are those of `nor1`. In fact, they are not exactly equal; they differ in about the 9th decimal. This is because `R` combines consecutive pairs of random numbers for more precision. In fact, `runif()` has a rather small *resolution* (the number of different values produced) of $2^{32} \approx 4.3 \times 10^9$.

In the help-file for random number generation, accessed through `?RNG`, you will find that the length of the *period* after which the stream of random numbers repeats itself, that is, the minimal number $p$ such that for the random numbers $r_i = r_{i+p}$ for all integer $i$, is equal to the Mersenne prime $2^{19937} - 1$. More on this can be found here.

A simple way of describing what happens in normal random number generation is that `runif` randomly draws decimals $D_i$, with $D_i \in \{0, 1, \ldots, 9\}$ to generate a drawing from the integers 0 to $10^9 - 1$. Let's write it as $D_1 D_2 \ldots D_9$, for example 141592653. To get a value in $[0, 1)$, it is divided by $10^9$, resulting in $0.D_1 D_2 \ldots D_9 = 0.141592653$. Drawing one million such numbers gives many that already occurred earlier.

The second uniform random number is $0.D_{10} D_{11} \ldots D_{18}$, and the first number delivered by the `rnorm` function is $\Phi^{-1}(0.D_1 D_2 \ldots D_9 D_{10} D_{11} \ldots D_{18})$. Now the probability of drawing a number that was encountered before is much smaller.

$Q_1$ To illustrate this, run the following script (this takes about 30 seconds):

```
set.seed(1)
sum(duplicated(runif(1e6))) ## = 120
sum(duplicated(rnorm(1e8))) ## = 0
```

First of all, consult `?duplicated` to find out how to interpret these numbers.

Let $N_k$ denote the number of different numbers resulting from the first $k$ drawings when drawingly randomly from the set $0, \ldots, m-1$, for $k = 1, \ldots, n$. We know that the probability of drawing an old number when $t$ of $m$ numbers have already occurred is $t/m$, so conditionally,

$$\Pr[N_k = t \mid N_{k-1} = t] = t/m = 1 - \Pr[N_k = t + 1 \mid N_{k-1} = t].$$

So for the conditional mean,

$$\mathrm{E}[N_k \mid N_{k-1}] = N_{k-1} + 1 - N_{k-1}/m.$$

From the tower rule we see that, writing $f = 1 - 1/m$,

$$\mathrm{E}[N_k] = \mathrm{E}[\mathrm{E}[N_k \mid N_{k-1}]] = f\,\mathrm{E}[N_{k-1}] + 1.$$

We know that $N_1 \equiv 1$ so $\mathrm{E}[N_1] = 1$. So by induction we see that for $n = 2, 3, \ldots,$

$$\mathrm{E}[N_n] = f\,\mathrm{E}[N_{n-1}] + 1 = f(\underbrace{f^{n-2} + f^{n-3} + \cdots + 1}_{\text{induction assumption}}) + 1 = f^{n-1} + f^{n-2} + \cdots + 1 = \frac{1 - f^n}{1 - f}.$$

Check if the outcome 120 of `duplicated(runif(1e6))` is consistent with the assumption that `runif()` randomly produces $m = 2^{32}$ different values uniformly.

In this post you find that the actual resolution of `rnorm()` is 'somewhere in the 2^50's', that is, roughly between $10^{15}$ and $10^{18}$. Check if the number of duplicates for `rnorm` is consistent with $m = 10^{15}, 10^{16}, 10^{17}, 10^{18}$.

Hint: computing $n - \frac{1-f^n}{1-f}$ gives inaccurate results here because $1 - f^n = 0$ numerically. Use

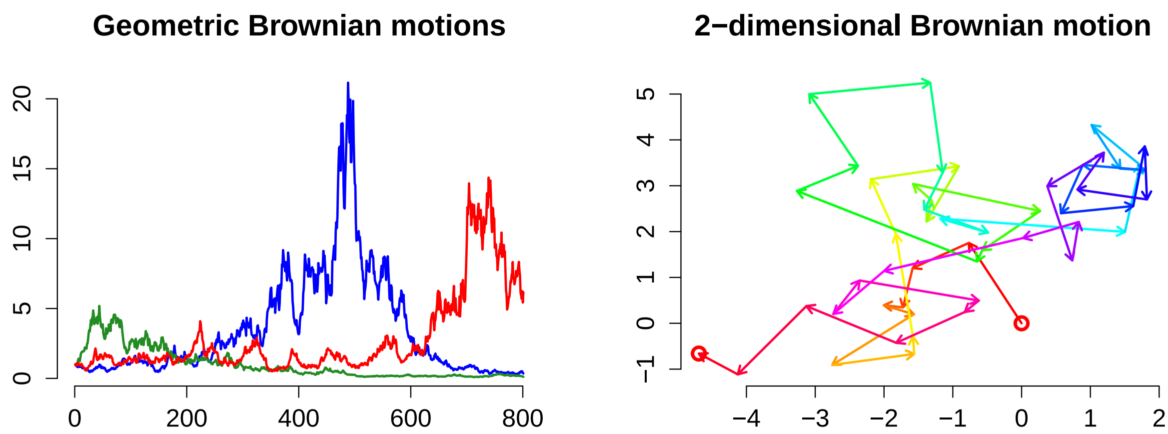
$$\left(1 - \frac{1}{m}\right)^n = 1 - \frac{n}{m} + \frac{n^2}{2m^2} + O\left(\left(\frac{n}{m}\right)^3\right) \text{ for small } \frac{n}{m}$$

(binomial expansion) to derive that a good approximation to the result is $n - \frac{1-f^n}{1-f} \approx \frac{n^2}{2m}$. $\square$

**Example: Brownian motion** A Brownian motion, or Wiener process, is a stochastic process $B(t)$ with stationary independent normally distributed increments $B(t+h) - B(t)$. In the left hand panel of the plot below, we depict $\mathrm{e}^{B(0)}, \mathrm{e}^{B(1)}, \ldots, \mathrm{e}^{B(800)}$ if $B(0) = 0$ and $B(1) \sim \mathrm{N}(0, \frac{1}{100})$, using the `cumsum` function to make cumulative sums. By taking exponents, we create three instances of *geometric* Brownian motions. On the right hand side, we create a two-dimensional Brownian motion, connecting successive points $(x_i, y_i)$ to $(x_{i+1}, y_{i+1})$ by colored arrows and starting at the origin.

```
par(mfrow=c(1,2), lwd=2, bty="n")
set.seed(3); y <- c(0,cumsum(rnorm(800)))/10
plot(exp(y), col="blue", type="l", ylab="", xlab="", lwd=2,
     main="Geometric Brownian motion")
set.seed(4); y <- c(0,cumsum(rnorm(800)))/10
lines(exp(y), col="forestgreen", lwd=2)
set.seed(5); y <- c(0,cumsum(rnorm(800)))/10
lines(exp(y), col="red")

n<-50; set.seed(9)
x <- cumsum(c(0,rnorm(n))); y <- cumsum(c(0,rnorm(n)))
plot(x, y, type="n", xlab="", ylab="", main="2-dimensional Brownian motion")
arrows(x[1:n], y[1:n], x[1+(1:n)], y[1+(1:n)], col=rainbow(n), lwd=2, length=.08)
points(0,0,col="red",cex=1.5,lwd=3)
points(x[n+1],y[n+1],col="red",cex=1.5,lwd=3)
```




$Q_2$ A one-dimensional random walk is a Markov chain whose state space is given by the integers

$i = 0, \pm 1, \pm 2, \ldots$ For some number $p$ satisfying $0 < p < 1$, the transition probabilities $P_{i,j}$ (of moving from state $i$ to state $j$) are given by

$$P_{i,i+1} = p = 1 - P_{i,i-1}.$$

Plot a realization of such a random walk of length 200 with probability of going up equal to 0.52. □

## 2 Bivariate Normal random variables

In this section, we show how to draw a sample from a bivariate normal distribution with standard marginals and known correlation. We start with $X$ and $Y$ iid $\mathrm{N}(0,1)$, and store samples in `X` and `Y`:

```
set.seed(2004); options(digits=2) ## not too many digits in output
X <- rnorm(1000); Y <- rnorm(1000)
```

We want to transform $(X, Y)$ into $(X, Y^*)$, with $Y^* = aX + bY$, choosing $a$ and $b$ in such a way that $\mathrm{Var}[Y^*] = 1$ and also that the correlation $r(X, Y^*)$ has a prescribed value.

In matrix notation, we have $\binom{X}{Y^*} = \mathbf{A}\binom{X}{Y}$ with $\mathbf{A} = \left(\begin{smallmatrix} 1 & 0 \\ a & b \end{smallmatrix}\right)$.

Recall that the correlation is given by $r(X, Y^*) = \dfrac{\mathrm{E}[XY^*] - \mathrm{E}[X]\,\mathrm{E}[Y^*]}{\sqrt{\mathrm{Var}[X]\,\mathrm{Var}[Y^*]}}$.

$Q_3$ What values for $a$ and $b$ ensure that $\mathrm{Var}[Y^*] = 1$ and $r(X, Y^*) = .8$? □

Since the old sample is no longer needed, we overwrite `Y` by the sample from $Y^*$:

```
a <- ...; b <- ...; Y <- a*X + b*Y
```

$Q_4$ What is $\mathbf{\Sigma}$, the covariance matrix of random vector $(X, Y^*)$? □

$Q_5$ Verify that $\mathbf{A}$ is the Cholesky decomposition of $\mathbf{\Sigma}$. □

$Q_6$ Using the functions `mean`, `var` and `cor`, verify if the sample means/variances/correlations resemble the theoretical values. □

To obtain a scatterplot of sample `Y` against `X`, we do

```
par(mfrow=c(1,2))     ## two plots side by side
plot(X,Y, pch="*")
```

Good times, though nice, are not very challenging. We define 'bad' times to have $X < d$ with $d = -2.2$. We separate good times from bad times by drawing a vertical red line in the plot:

```
d <- -2.2
abline(v=d, col="red")
```

Now we fill a vector of logicals, `TRUE` if $X_i < d$, else `FALSE`, as follows:

```
bad <- (X < d)
```

Next plot `Y` against `X` restricted to 'bad times', draw the same separation line, and compute the correlation restricted to bad times by doing

```
plot(X[bad],Y[bad],ylim=range(Y))
abline(v=d, col="red")              # same line as on lhs
cor(X[bad],Y[bad])   ## 0.083       # correlation << 0.8
```

In the rhs of the plot, we see that $X$ 'bad' $\Longrightarrow Y^*$ 'baddish'. But we often want to model a situation where 'tail dependence' holds: "in bad times, the worse $X$, the worse $Y^*$". Here we see that given bad times, $X$ and $Y^*$ are almost uncorrelated.

# 3 The multivariate Student distribution

A univariate Student($k$) distribution is found by dividing $X \sim \mathrm{N}(0,1)$ by a r.v. $\sqrt{V/k}$, with $V \sim \chi^2_k$, independent of $X$. A multivariate Student distribution with mean vector $\vec{\mu}$, variance matrix $\mathbf{\Sigma}$ and $k$ degrees of freedom arises by transforming the multinormal $\vec{Z} \sim \mathrm{N}(\vec{0}, \mathbf{\Sigma})$ into

$$\vec{T} = \vec{\mu} + \vec{Z}/\sqrt{V/k},$$

with $\vec{\mu}$ an $n$-vector, $\mathbf{\Sigma}$ an $n \times n$ covariance matrix, and $V \sim \chi^2_k$, independent of $\vec{Z}$. Note that $V$ is *scalar*; the same value is used for all components of $\vec{Z}$.

$Q_7$ Let $(X,Y) \sim$ bivariate Normal with $\mathrm{E}[X] = \mathrm{E}[Y] = 0$, $\mathrm{Var}[X] = \mathrm{Var}[Y] = 1$ and $r(X,Y) = r$. Let $W$, independent of $(X,Y)$, have finite variance. Show that $r(XW, YW) = r$. Indicate why $\mathrm{Var}[W] < \infty$ is needed. □

We transform the bivariate normal random vector $(X, Y^*)$ obtained earlier as follows:

```
chi5 <- sqrt(rchisq(1000, df=5)/5) # sqrt(V/k) with k=5
X <- X/chi5; Y <- Y/chi5           # now both marginally t(5)
```

Note that the same denominator is used to inflate/deflate $X$ and $Y^*$.

$Q_8$ Again, check sample means, variance and correlation of `X,Y`. Show that population means are 0, while the variances are $k/(k-2) = 5/3$. For the variance, you will need to compute $\mathrm{E}[1/V]$ for a $\chi^2_k$ random variable $V$. Show that $r(XW, Y^*W) = 0.8$ by taking $W = \sqrt{k/V}$ above. So the sample correlation should be close to 0.8. □

$Q_9$ Repeat the statements for the normal case to get a scatterplot, both for all observations and only the 'bad' ones. Is there 'tail dependence' for the multiStudent distribution? □

Note that the (upper) tail dependence index of a pair $(X, Y)$ is the probability of an $X$-disaster, given a $Y$-disaster, both of low probability, so $\lim_{u\uparrow 1} \Pr[F_X(x) > u \mid F_Y(x) > u]$. This can be shown to be equal to 0 for bivariate normal r.v.'s (if their correlation is less than 1), positive for bivariate Student r.v.'s.

# 4 Simulating correlated multinormal random variables

To simulate for example from a VAR model in an ALM study, one must draw from *correlated* multinormal random vectors $\vec{Z} \sim \mathrm{N}(\vec{\mu}, \mathbf{\Sigma})$. This is easy using `R`, but much harder in (standard) Excel.

In `R`, simply use the `mvrnorm` function in the library `MASS`, as in the example below. This library is included in every `R`-installation, but not loaded automatically when `R` is started.

```
library(MASS)
mu <- c(1,3,5); sig2 <- c(1,2,5)
Corrmat <- rbind(c(1., .3, .3),
                 c(.3, 1., .4),
                 c(.3, .4, 1.))
Varmat <- Corrmat * sqrt(sig2 %*% t(sig2))
Z <- mvrnorm(100, mu, Varmat)
options(digits=7)
colMeans(Z); diag(cov(Z)); cor(Z)
## 1.063583 3.007684 4.963563
## 1.032778 1.923742 4.653725
##           [,1]      [,2]      [,3]
## [1,] 1.0000000 0.2990840 0.1916127
## [2,] 0.2990840 1.0000000 0.4188593
## [3,] 0.1916127 0.4188593 1.0000000
```

$Q_{10}$ If `Corrmat` is the correlation matrix and `sig2` the vector of marginal variances, verify that `Varmat` as computed above is the corresponding covariance matrix $\mathbf{\Sigma}$.

What do `colMeans(Z)`, `diag(cov(Z))` and `cor(Z)` estimate? □

**The Cholesky method** To draw from $\vec{Z} \sim \mathrm{N}(\vec{\mu}, \mathbf{\Sigma})$ for some $n$-vector $\vec{\mu}$ and some $n \times n$ variance-covariance matrix $\mathbf{\Sigma}$, the idea is to use $\vec{Z} \leftarrow \vec{\mu} + \mathbf{A}\vec{X}$, where $\vec{X}$ is a vector with elements $X_i \overset{\text{iid}}{\sim} \mathrm{N}(0,1)$, $i = 1, \dots, n$. Then $\vec{Z}$ is multinormal, and $\mathrm{E}[\vec{Z}] = \vec{\mu}$ is obvious by the linearity property of expectations. So we only have to find a matrix $\mathbf{A}$ such that $\mathrm{Var}[\vec{Z}] = \mathbf{\Sigma}$.

By definition, the $(i, j)$ element of $\mathrm{Var}[\vec{Z}]$ is

$$\mathrm{Cov}[Z_i, Z_j] = \mathrm{E}[(Z_i - \mu_i)(Z_j - \mu_j)],$$

so again by the linearity property of expectations, the covariance matrix equals

$$\mathbf{\Sigma} = \mathrm{E}[(\vec{Z} - \vec{\mu})(\vec{Z} - \vec{\mu})'] = \mathrm{E}[\mathbf{A}\vec{X}(\mathbf{A}\vec{X})'] = \mathrm{E}[\mathbf{A}\vec{X}\vec{X}'\mathbf{A}'] \overset{\text{linearity}}{=} \mathbf{A}\,\mathrm{E}[\vec{X}\vec{X}']\mathbf{A}' = \mathbf{A}\mathbf{I}\mathbf{A}' = \mathbf{A}\mathbf{A}'.$$

So every 'square root' of matrix $\mathbf{\Sigma}$, that is, a matrix $\mathbf{A}$ with $\mathbf{\Sigma} = \mathbf{A}\mathbf{A}'$, is a suitable choice. One such square root is provided by the Cholesky decomposition. Some remarks:

- The function `mvrnorm` in fact uses the eigenvalue decomposition, stating in its help-file: *"Although a Cholesky decomposition might be faster, the eigendecomposition is stabler"*
- Cholesky does not give just any square root $\mathbf{A}$ but specifically a lower triangular matrix, which was convenient for hand computations.
- Neither `chol()` nor `mvrnorm()` checks for symmetry of matrix $\mathbf{\Sigma}$. In `chol()` only the lower triangle is used, in `mvrnorm()` the upper triangle. Both return an error when the matrix $\mathbf{\Sigma}$ is not positive definite.

$Q_{11}$ Use samples of size one million to estimate $F^{-1}_{X+Y+Z}(.9999)$, being the VaR at 99.99% level of $X + Y + Z$, when $(X, Y, Z)$ is trinormal with parameters as in the script above. To get an idea of the precision reached, do this 10 times and print mean and standard deviation of the results.

Also, compute the theoretical value of this quantile. □

## 5 Multinormal, multi-Student stop-loss premiums and Expected Shortfall

$Q_{12}$ Using `mvrnorm()`, first generate `n=`$10^6$ independent drawings from a trinormal random vector $(X, Y, Z) \sim \mathrm{N}(\vec{\mu} = \vec{0}; \mathbf{\Sigma})$, with the covariance matrix $\mathbf{\Sigma}$ having ones on the diagonal and $\rho = 1/6$ outside the diagonal. Store the resulting $n \times 3$ matrix in `Z`. □

$Q_{13}$ Based on this sample, construct $V_i = X_i + Y_i + Z_i$, $i = 1, \ldots, n$. To construct the sample `V`, you might want to apply one of the functions `rowSums()` or `colSums()`, or to apply the sum function to margins of matrix `Z`. □

$Q_{14}$ From the sample `V`, estimate the 97.5% quantile $d = F_V^{-1}(0.975)$. Use the `quantile()` function. □

$Q_{15}$ Using the function `pmax()`, estimate the stop-loss premium $\mathrm{E}[(V - d)_+]$. □

*This is an estimate of the so-called Expected Shortfall (ES) of $V$ at level 97.5%. It can be described as the average excess of loss over the available capital when the latter is fixed to be sufficient with 97.5% probability.*

$Q_{16}$ What probability distribution do the random variables $V_i$ have? □

$Q_{17}$ Using formula (3.104) of MART, compute the real value of the ES of $V$ at level 97.5%. □

$Q_{18}$ Using a sample of size $10^6$, estimate the ES of $V'$ at 97.5% level if $V'$ is a sum of multi-Student random variables, also with parameters $\vec{\mu} = \vec{0}$ and $\mathbf{\Sigma}$, and with df $k = 5$. □

$Q_{19}$ A univariate Student$(k)$ random variable arises by dividing a standard normal random variable by $W = \sqrt{k/U}$, with $U \sim \chi^2(k)$. Use `integrate()`, as well as the functions `qt()` and `dt()` to find the actual value of the ES at level 97.5%. $\square$