Where do Multivariate Normal Samples Come from?

Paul E. Johnson[1] [2]

[1] Department of Political Science

[2] Center for Research Methods and Data Analysis, University of Kansas

2017

What I learned on my Winter Vacation

- These notes accompany the longer essay
- That essay uses some LyX/LaTeX tricks that were new to me, such as the bold upright symbol for matrices in math. If you want to know how that's done, I'll share the LyX document to you.
- The next thing I want to learn is how to make transpose symbols look nicer

Overview: The 5 step process for manufacturing MVN

The paper says a 5 step algorithm receives the user's requested mean vector $\boldsymbol{\mu}$ and a variance-covariance matrix $\boldsymbol{\Sigma}$.

1. Calculate the eigen decomposition of $\boldsymbol{\Sigma}$.
2. Check that $\boldsymbol{\Sigma}$ is positive definite by inspecting the eigenvalues.
    1. If an eigenvalue is intolerably negative, terminate with an error message.
    2. Tolerably negative eigenvalues are reset to 0.
3. Create a scaling matrix, $\mathbf{S}$. The two programs differ in this stage. R uses the eigen decomposition while Stata uses Cholesky roots.
4. Create a candidate $n \times p$ matrix of random vectors by drawing from $N(0, 1)$.
5. Apply $\mathbf{y} = \boldsymbol{\mu} + \mathbf{S}\mathbf{x}$ to rescale the candidate random draws.

MVN

- $\mathbf{x} \sim MVN(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ is a draw from the multivariate normal distribution, $MVN(\boldsymbol{\mu}, \boldsymbol{\Sigma})$

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_p \end{bmatrix} \sim MVN(\boldsymbol{\mu}, \boldsymbol{\Sigma}) = MVN\left(\begin{bmatrix} \mu_1 \\ \mu_2 \\ \vdots \\ \mu_p \end{bmatrix}, \begin{bmatrix} \sigma_1^2 & \sigma_{12} & & \sigma_{1p} \\ \sigma_{12} & \sigma_2^2 & & \sigma_{2p} \\ & & \ddots & \\ \sigma_{1p} & \sigma_{2p} & & \sigma_p^2 \end{bmatrix} \right). \tag{1}$$

- The output from one draw is a vector of values, not a single number.

$$(1.1, 2.4, -3, 2.2\ldots)^T \tag{2}$$

- A data set might be a collection of those draws, each one representing a person's answers on a survey.

Draw a sample from $N(\mu, \sigma^2)$

- A univariate "standard" normal draw $x_i \sim N(0,1)$
- Most programs can draw that. I could teach you where that comes from, lets don't worry about it now.
- If you want a draw from $N(5, 100)$, you calculate

$$y_i = 5 + 10 \times x_i \tag{3}$$

- In words: multiply by the square root of the variance and add the desired expected value. for $N(\mu, \sigma^2)$

$$y = \mu + \sigma \cdot x. \tag{4}$$

- Little Puzzle. Suppose you want draws that appear as if they are $N(0, 9)$. Is it correct to write

$$x_i = 0 + \sqrt{9} u_i ? \tag{5}$$

- hint: The square root of 9 is not unique

Generalize that to $MVN(\boldsymbol{\mu}, \Sigma)$

- The process is going to be the same, except it is scaled up to draw a vector of candidate values filled with $N(0,1)$ draws.

$$\mathbf{y} = \boldsymbol{\mu} + \mathbf{S}\mathbf{x}. \tag{6}$$

- p candidate values $N(0,1)$ values are in $\mathbf{x}$, multiply them by something, add something

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_p \end{bmatrix} = \begin{bmatrix} \mu_1 \\ \mu_2 \\ \vdots \\ \mu_p \end{bmatrix} + \begin{bmatrix} s_{11} & s_{12} & & s_{1p} \\ s_{21} & s_{22} & & s_{2p} \\ & & \ddots & \\ s_{p1} & s_{p2} & & s_{pp} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_p \end{bmatrix}.$$

- The big puzzle is where do you get that matrix full of s's so that the resulting random draw has variance $\boldsymbol{\Sigma}$. And what properties it should have.

Generalize that to $MVN(\boldsymbol{\mu}, \Sigma)$ ...

$$E[\mathbf{y}] = E[\boldsymbol{\mu} + \mathbf{S}\mathbf{x}] = \boldsymbol{\mu} + \mathbf{S}E[\mathbf{x}]. \tag{7}$$

- And the variance matrix of $\mathbf{y}$ is

$$Var[\mathbf{y}] = \mathbf{S}Var(x)\mathbf{S}^T. \tag{8}$$

- Because $Var(\mathbf{x}) = \mathbf{I}$ and because $\mathbf{S}^T\mathbf{S}$ is symmetric,

$$Var[\mathbf{y}] = \mathbf{S}\mathbf{S}^T = \mathbf{S}^T\mathbf{S}. \tag{9}$$

- So if our goal is to generate $y \sim N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, then we need $\boldsymbol{\Sigma} = \mathbf{S}\mathbf{S}^T = \mathbf{S}^T\mathbf{S}$.
- In a sense, $\mathbf{S}$ is thus one kind of “square root” of a matrix.

If we are given a user's requested , can we use that to create the matrix we need?

- Not always. There may be no matrix "square root".
- User might supply $\boldsymbol{\Sigma}$ that's not actually a covariance matrix.
- What do we know about $\boldsymbol{\Sigma}$?
  - Symmetric, $\sigma_{ij} = \sigma_{ji}$
  - Positive Definite (PD) or Positive Semidefinite (PSD).
- This is either esoteric and difficult to understand or simple and obvious. Here is my simple and obvious explanation.
- Here's the formula for the MVN:

$$f(\mathbf{x}) = \frac{1}{(2\pi)^{p/2}|\boldsymbol{\Sigma}|^{1/2}} e^{\frac{-1}{2}(\mathbf{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})}. \tag{10}$$

If we are given a user's requested , can we use that to create the matrix we need? ...

- The basic idea of the normal is that as a point moves away from the mean, as $\mathbf{x}-\boldsymbol{\mu}$ grows greater, we are less and less likely to observe that point. For this to be true, we need this to be positive. Otherwise, probability would grow as $\mathbf{x}-\boldsymbol{\mu}$ grows.
$$(\mathbf{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu}) > 0. \tag{11}$$
- This seems obvious to me. Suppose $\boldsymbol{\Sigma}=I$, so this reduces to the (Pythagorean) distance between $\mathbf{x}$ and $\boldsymbol{\mu}$
$$(\mathbf{x}-\boldsymbol{\mu})^T(\mathbf{x}-\boldsymbol{\mu}) = (x_1-\mu_1)^2+(x_2-\mu_2)^2+\ldots+(x_p-\mu_p)^2 \tag{12}$$
- That's a sum of squared things, it has to be 0 or greater.
- The $\boldsymbol{\Sigma}^{-1}$ matrix puts weights on the deviations, but it should not be able to turn the distance between $\mathbf{x}$ and $\boldsymbol{\mu}$ into a negative number. The closest two things can be is the same location.
- There's a theorem that says if $\boldsymbol{\Sigma}$ is positive definite, then $\boldsymbol{\Sigma}^{-1}$ is also PD

Check if a matrix is PD: Eigen Decomposition

- We insist that $\mathbf{\Sigma}$ is symmetric.
- We need a way to find out if it is PD. This turns out to be tricky because of numerical rounding error.
  - In pencil and paper math, there is a theorem says that a matrix is PD if and only if all of its eigenvalues are positive.
- Recognizing that eigenvalues are approximated in computers, R and Stata use the same approach. A tolerance value is used as follows

$$\lambda_j < -tol\,|\lambda_1|. \tag{13}$$

If λ_j is so far negative that it is below $-tol|\lambda_1|$, then we conclude the matrix is not PD.

Background info on the eigenvalues and eigenvectors

- The eigenvalues (λ_j) and eigenvectors $\mathbf{v}_j$ of $\boldsymbol{\Sigma}$ are paired in this weird way.

$$\boldsymbol{\Sigma}\mathbf{v}_j = \lambda_j \mathbf{v}_j \tag{14}$$

The vector $\mathbf{v}_j$ that can be thought of as a rescaled thing coming out of $\boldsymbol{\Sigma}$. Er, the scaling effect of $\boldsymbol{\Sigma}$ can be equaled by a simple multiplicative rescaling by λ_j.

- Eigenvector matrix: collect the eigenvectors, side by side:

$$\mathbf{V} = \begin{bmatrix} v_{11} & & v_{1p} \\ v_{21} & \vdots & v_{2p} \\ & & \\ v_{p1} & & v_{pp} \end{bmatrix} \tag{15}$$

Background info on the eigenvalues and eigenvectors ...

- Interesting property. The Eigenvalues are “orthogonal” to one another.

$$\mathbf{v}_1^T \cdot \mathbf{v}_2 = 0 \tag{16}$$

- We usually scale eigenvectors these so that $\mathbf{v}_i^T \cdot \mathbf{v}_i = 1$,
- Which implies this interesting property

$$\mathbf{V}^T\mathbf{V} = \mathbf{I} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{17}$$

- And thus the inverse of the orthogonal matrix $\mathbf{V}$ is very easy to calculate: it is the transpose

$$\mathbf{V}^T = \mathbf{V}^{-1} \tag{18}$$

Background info on the eigenvalues and eigenvectors ...

- Anyway, given the vector of eigenvalues $(\lambda_1, \lambda_2, \ldots, \lambda_p)$, in pencil and paper math, $\lambda_j > 0 \, \forall j$.
- Numerically, we say if λ_j is just a little bit negative, we will act as if that is roundoff error, and it is re-assigned as 0.

If all eigenvalues are greater than 0, Cholesky root

- Cholesky works for SQUARE, PD matrices only.

$$\mathbf{\Sigma} = \mathbf{R^T} \times \mathbf{R} = \begin{bmatrix} r_{11} & 0 & 0 & 0 & 0 \\ r_{12} & r_{22} & 0 & 0 & 0 \\ r_{13} & r_{23} & r_{33} & 0 & 0 \\ & & & \ddots & 0 \\ r_{1p} & r_{1p} & r_{3p} & & r_{pp} \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & \cdots & r_{1p} \\ & r_{22} & r_{23} & & r_{2p} \\ & & r_{33} & & \\ & & & \ddots & \\ & & & & r_{pp} \end{bmatrix}. \quad (19)$$

- If you require the diagonal elements are positive, this is a UNIQUE triangular square root
- We usually see people using the lower triangular part of the Cholesky as the scaling matrix

$$\mathbf{S} = \mathbf{R}^T \quad (20)$$

- Other square roots exist, as we see next
- Recall Cholesky is not guaranteed to exist if $\mathbf{\Sigma}$ is not strictly PD. SPD is not sufficient.

If an eigenvalue is equal to 0, it is not PD, must use Eigen decomposition

$$\begin{aligned}
\mathbf{\Sigma V} &= \mathbf{V} diag(\boldsymbol{\lambda}) \\
\mathbf{\Sigma V V}^T &= \mathbf{V} diag(\boldsymbol{\lambda}) \mathbf{V}^T \\
\mathbf{\Sigma} &= \mathbf{V} diag(\boldsymbol{\lambda}) \mathbf{V}^T.
\end{aligned} \tag{21}$$

- The function $diag$ places a vector's values along the diagonal:

$$diag(\boldsymbol{\lambda}) = \begin{bmatrix} \lambda_1 & 0 & 0 & 0 \\ 0 & \lambda_2 & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & \lambda_p \end{bmatrix}. \tag{22}$$

If an eigenvalue is equal to 0, it is not PD, must use Eigen decomposition ...

- Since $\lambda_j \geq 0$, a real-valued square root of each eigenvalue exists, and we can write this as a product using $diag(\boldsymbol{\lambda})^{1/2}$:

$$diag(\boldsymbol{\lambda}) = \begin{bmatrix} \sqrt{\lambda_1} & 0 & 0 & 0 \\ 0 & \sqrt{\lambda_2} & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & \sqrt{\lambda_p} \end{bmatrix} \begin{bmatrix} \sqrt{\lambda_1} & 0 & 0 & 0 \\ 0 & \sqrt{\lambda_2} & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & \sqrt{\lambda_p} \end{bmatrix} = di \tag{23}$$

- This allows us to revise (21) into a format that helps us to see that we have square root of $\boldsymbol{\Sigma}$:

$$\begin{aligned} \boldsymbol{\Sigma} &= \mathbf{V} diag(\boldsymbol{\lambda})^{1/2}\ diag(\boldsymbol{\lambda})^{1/2} \mathbf{V}^T. \\ & \mathbf{V} diag(\boldsymbol{\lambda})^{1/2} (\mathbf{V} diag(\boldsymbol{\lambda})^{1/2})^T \end{aligned} \tag{24}$$

If an eigenvalue is equal to 0, it is not PD, must use Eigen decomposition ...

- The scaling matrix will be

$$\mathbf{S} = \mathbf{V} diag(\boldsymbol{\lambda})^{1/2} \tag{25}$$

because $\mathbf{S}\mathbf{S}^T = \mathbf{S}^T\mathbf{S} = \boldsymbol{\Sigma}$.

The $100,000 Question

How can the two completely different scaling matrices lead to equally good MVN draws?

$$\mathbf{S} = \begin{bmatrix} r_{11} & 0 & 0 & 0 \\ r_{12} & r_{22} & 0 & 0 \\ & & \ddots & 0 \\ r_{1p} & r_{1p} & & r_{pp} \end{bmatrix}, \text{or} \begin{bmatrix} \sqrt{\lambda_1}v_{11} & \sqrt{\lambda_2}v_{12} & & \sqrt{\lambda_p}v_{1p} \\ \sqrt{\lambda_1}v_{21} & \sqrt{\lambda_2}v_{22} & & \sqrt{\lambda_p}v_{2p} \\ & & \ddots & \\ \sqrt{\lambda_1}v_{p1} & \sqrt{\lambda_2}v_{p2} & & \sqrt{\lambda_p}v_{pp} \end{bmatrix}, \tag{26}$$

- I accept that because
  $\mathbf{S}^T\mathbf{S} = \mathbf{S}\mathbf{S}^T = \mathbf{\Sigma}$, using either matrix
  Generally speaking, matrix square roots are not unique.
  Theorem: Given a symmetric $\mathbf{\Sigma}$ for which a square root S exists ($\mathbf{S}^T\mathbf{S} = \mathbf{\Sigma}$), and given $\mathbf{V}$ is orthonormal, then $\mathbf{VS}$ is also a square root.

The \$100,000 Question ...

Proof. Recall $\mathbf{V}\mathbf{V}^T = \mathbf{I}$.

$$(\mathbf{VS})^T(\mathbf{VS}) \quad = \quad \mathbf{S}^T\mathbf{V}^T\mathbf{VS} = \mathbf{S}^T\mathbf{S}. \tag{27}$$

Here's a question: what is the statistical distribution of an empirically standardized variable

- In stats 101, they say calculate the mean and standard deviation and then calculate
$$\frac{x_i - \bar{x}}{s} \tag{28}$$
- That result has empirical mean 0 and empirical standard deviation 1.
- Did you ever notice that a column of those scores is no longer iid?
- In Stata and R, there is a way to use that same idea to standardize a whole matrix of MVN draws, so the observed mean is $\mathbf{0}$ and the observed variance is $\mathbf{I}$. From there, we can re-scale that so it has observed mean $\boldsymbol{\mu}$ and variance whatever you say, $\boldsymbol{\Sigma}$.
- The Stata documentation cautions users that the return from that is not MVN, but rather it is simply data with sufficient statistics that match the user's request (`corr2data`)
- But what is it?

Afterthought: Variance of X

- I got this far without using the QR or SVD decompositions. They are used in the paper
- I'll throw in an example here to show that these are used to calculate $\mathbf{X}^T\mathbf{X}$ efficiently.

The QR Decomposition

- A data matrix X might be prone to roundoff error when used in calculations. The proneness to error is summarized by its condition value, the ratio of its greatest and smallest singular values (SVD) defined below.
- Golub and Van Loan point out that if the condition number of $\mathbf{X}$ is κ, then the condition number of $\mathbf{X}^T\mathbf{X}$ is $\boldsymbol{\kappa}^2$.
  - Hence, calculations for linear models do not (any longer) actually form the product $\mathbf{X}^T\mathbf{X}$.
- How can we get covariance without forming that matrix, you ask?
- The theoretical quantity $(\mathbf{X}^T\mathbf{X})$ can be calculated in a much more numerically accurate way with the QR decomposition.
- The "thin" version of the $\mathbf{QR}$ decomposition is

$$\mathbf{X} = \mathbf{QR}. \tag{29}$$

The QR Decomposition ...

- $\mathbf{R}$ is an upper triangular matrix.

$$\mathbf{R} = \begin{bmatrix} r_{11} & r_{12} & \dots & r_{1p} \\ 0 & r_{22} & & r_{2p} \\ 0 & 0 & \ddots & \\ 0 & 0 & 0 & r_{pp} \end{bmatrix} \tag{30}$$

This turns out to be a numerically more accurate version of the Cholesky triangle and Cholesky calculations today usually rely on the QR decomposition.

The QR Decomposition ...

- The matrix $\mathbf{Q}$ is $n \times p$ orthogonal columns

$$\mathbf{Q} = \begin{bmatrix} Q & \begin{array}{c} p\,columns \\ orthogonal \end{array} \\ & \\ n\,rows & \\ & \\ & \end{bmatrix}. \tag{31}$$

- Orthogonality implies $\mathbf{Q}^{-1}\mathbf{Q} = \mathbf{Q}^T\mathbf{Q} = \mathbf{I}$
- The $\mathbf{R}$ produced by QR is, theoretically, equivalent to the Cholesky decomposition of $(\mathbf{X}^T\mathbf{X})$, with the possible exception that the diagonal elements in some of the rows in $\mathbf{R}$ from $\mathbf{QR}$ are not positive and signs of those rows need to be reversed.

The QR Decomposition ...

- Where numeric "covariance matrices" come from: To avoid explicitly forming $(\mathbf{X}^T\mathbf{X})$, we replace $\mathbf{X}$ with $\mathbf{QR}$:

$$\mathbf{X}^T\mathbf{X} = (\mathbf{QR})^T\mathbf{QR} = \mathbf{R}^T\mathbf{Q}^T\mathbf{QR} = \mathbf{R}^T\mathbf{R}. \tag{32}$$

- If a calculation calls for $(\mathbf{X}^T\mathbf{X})^{-1}$, then, we can replace that with $(\mathbf{R}^T\mathbf{R})^{-1}$. However, we would not explicitly calculate $(\mathbf{R}^T\mathbf{R})$ and invert that product. Instead, we note, theoretically

$$(\mathbf{R}^T\mathbf{R})^{-1} = \mathbf{R}^{-1}\mathbf{R}^{-1^T} \tag{33}$$

It is necessary to calculate $\mathbf{R}^{-1}$, but that is a simpler, more stable calculation because the lower left side of $\mathbf{R}$ is full of 0's. The inverse of an upper triangular matrix will also be upper triangular, so the benefits of this simplification continue.

The SVD Decomposition

- The thin singular value decomposition (SVD) of $\mathbf{X}$ is a product of 3 matrices.

$$\mathbf{X} = \mathbf{U}\mathbf{D}\mathbf{V}^T. \tag{34}$$

$$\begin{bmatrix} \mathbf{U} & p\,columns \\ & orthogonal \\ n\,rows & \\ & \\ & \end{bmatrix} \begin{bmatrix} \delta_1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \delta_p \end{bmatrix} \begin{bmatrix} \mathbf{V}^T & p\,columns \\ p\,rows & \end{bmatrix}. \tag{35}$$

The SVD Decomposition ...

- The columns of $\mathbf{U}$ and $\mathbf{V}$ are orthogonal. That affords simplifications such as $\mathbf{U}^T\mathbf{U} = I$ and $\mathbf{V}^T = \mathbf{V}^{-1}$. The matrix $\mathbf{D}$ is a $p \times p$ diagonal matrix of the so-called “singular values”, δ_i.

$$\mathbf{D} = diag(\delta_1, \delta_2, \ldots, \delta_p) = \begin{bmatrix} \delta_1 & 0 & & 0 \\ 0 & \delta_2 & & 0 \\ & & \ddots & \\ 0 & 0 & & \delta_p \end{bmatrix} \qquad (36)$$

- To see the simplifying benefit of the SVD, replace $\mathbf{X}$ with $\mathbf{UDV}^T$.

$$\begin{aligned} \mathbf{X}^T\mathbf{X} &= (\mathbf{UDV}^T)^T\mathbf{UDV}^T = \mathbf{VDU}^T\mathbf{UDV}^T \\ &= (\mathbf{DV}^T)^T\mathbf{DV}^T = (\mathbf{VD})(\mathbf{VD})^T \qquad (37) \end{aligned}$$

The SVD Decomposition ...

- The square root of $\mathbf{X}^T\mathbf{X}$ is thus seen to be $\mathbf{VD}$(or$(\mathbf{DV}^T)^T$, depending on how you want to group things. So the SVD based candidate for a square root of $\mathbf{X}^T\mathbf{X}$ is

$$\mathbf{S} = \mathbf{V}diag(\boldsymbol{\delta}) \tag{38}$$

- The SVD approach is similar in personality to the eigenvalue decomposition of $\mathbf{X}^T\mathbf{X}$. If numerical linear algebra were "perfectly accurate", then the eigen method in equation (25), $\mathbf{V}diag(\boldsymbol{\lambda})^{1/2}$, would be identical to the SVD solution $\mathbf{V}diag(\boldsymbol{\delta})$. Consequently, we see that, on a theoretical level, the singular values are the squares of the eigen values.