Simulating Ultra High-Dimensional Multivariate Data Using the bigsimr R Package

A.G. Schissler A. Knudson

Monday, June 15, 2020

Abstract

In this era of Big Data, it is critical to realistically simulate data to conduct informative Monte Carlo studies. This is problematic when data are inherently multivariate while at the same time are (ultra-) high dimensional. This situation appears frequently in observational data found on online and in high-throughput biomedical experiments (e.g., RNA-sequencing). Due to the difficulty in simulating realistic correlated data points, researchers often resort to simulation designs that posit independence — greatly diminishing the insight into the empirical operating characteristics of any proposed methodology. Major challenges lie in the computational complexity involved in simulating these massive random vectors. We propose a fairly general, scalable procedure to simulate high-dimensional multivariate distributions with pre-specified marginal characteristics and dependency characteristics. As a motivating example, we use our methodology to study large-scale statistical inferential procedures applied to cancer-related RNA-sequencing data sets. The proposed algorithm is implemented as the `bigsimr` R package.

Contents

1 Introduction **3**

1.1 Ultra High Dimensional multivariate modeling and simulation desired properties 3

2 Background and notation **4**

2.1 Gaussian copulas 4

2.2 Measures of dependency 5

2.3 Marginal-dependent bivariate correlation bounds 5

3 Simulation algorithm **6**

3.1 Algorithm description 6

3.2 Simulation Algorithm 7

3.2.1 Inputs 7

3.2.2 Algorithm 7

4 The `bigsimr` R package **7**

4.1 Basic use 7

4.1.1 Specifying marginals 8

4.1.2 Specify correlation 8

4.1.3 Small sample . . . 8
4.1.4 Scaling up N . . . 10
4.1.5 Evaluation . . . 11

5 Monte Carlo evaluation and Comparsions to other software 11

5.1 Other multivariate simulation software . . . 11
5.2 Simulation I: Bivariate simulation using rank-based correlation . . . 11
5.2.1 Continuous example: Bivariate Exponential . . . 11
5.2.2 Discrete example: Bivariate Geometric . . . 12
5.3 Simulation II: Ultra HD multivariate simulation with identical margins and exchangeable rank-based correlation. . . . 12
5.3.1 UHD Continuous example: Multivariate Gamma . . . 12
5.3.2 UHD Discrete example: Multivariate Negative Binomial . . . 12
5.4 Simulation III: Ultra HD multivariate simulatino with heterogenous margins and exchangeable rank-based correlation. . . . 12
5.4.1 UHD Heteregeous continuous example: Multivariate Gamma . . . 12
5.4.2 UHD Heteregeousdiscrete example: Multivariate Negative Binomial . . . 12

6 Some applications 12

6.1 Simulating Ultra High Dimensional RNA-seq data . . . 12
6.2 Ultra High-Dimensional simulation-based correlation hypothesis testing under . . . 16
6.3 Simulation-based computation of correlation bounds . . . 16
6.4 Simulation-based joint probability calculations . . . 17

7 Conclusion/Discussion 17

8 Supplementary Materials 18

9 Acknowledgement(s) 18

10 Disclosure statement 18

11 Funding 18

12 Nomenclature/Notation 18

13 Notes 18

13.0.1 List of supported distributions . . . 18

1 Introduction

Massive high-dimensional data sets are commonplace in many areas of scientific inquiry. As new methods are developed for these data, a fundamental challenge lies in designing and conducting simulation studies to assess operating characteristics of proposed methodology, such as false positive rates, statistical power, and robustness — often in comparison to existing methods. Another application lies in the discovering the sampling distribution of test statistic under hypothesized null models in complex scenarios. Such Monte Carlo studies in the high-dimensional setting often become difficult to conduct with existing algorithms and tools, or require expertise in computing on clusters. This is particularly true when simulating massive multivariate, non-normal distributions.

Along those lines, simulating high-dimension dependent negative binomial data motivates this work — in pursuit of modeling RNA-sequencing data (Wang et al., 2009; Conesa et al., 2016) derived from breast cancer patients. This laboratory procedure results in count data with infinite support, since RNA-sequencing platforms measure gene expression by enumerating the number of reads aligned to genomic regions. Often researchers posit a negative binomial model (refs) as counts are often over (or under) expressed that a Poisson model would suggest. Yet due to inherent biological processes, gene expression data exhibits correlation (coexpression) across genes (Efron, 2007; Schissler et al., 2018). RNA-sequencing analysis has garnered much interest in the statistical literature (refs). Often researchers simulate independently or from their proposed model in order to conduct Monte Carlo studies (refs). An alternative strategy is to think *generatively* about the data collection process and scientific domain knowledge and design simulations with probabilistic models that reflect those processes. In our motivating example, we'll study the consequences correlation in the simulation study replicating test statistics from an illustrative, classical large-scale hypothesis testing study (see Efron, 2004).

1.1 Ultra High Dimensional multivariate modeling and simulation desired properties

With our application in mind and seeking a general-purpose algorithm with broad applications, our purposed methodology should possess the following properties (adapted from criteria from Nikoloulopoulos (2013)):

- P1: Wide range of dependence, allowing both positive and negative dependence
- P2: Flexible dependence, meaning that the number of bivariate marginals is (approximately) equal to the number of dependence parameters.
- P3: Flexible marginal modeling, generating heterogeneous data — possibly from differing probability families.

Moreover, the simulation method must scale to high dimensions:

- S1: Procedure must scale to high dimensions, computable in a reasonable amount time.
- S2: Procedure must scale to high dimensions while maintaining accuracy.

As others Madsen and Birkes (2013) have noted, however, simulating dependent data can be challenging. A central issue lies characterizing dependency between components in the high-dimensional random vector. The choice of correlation in typical practice usually relates to the eventual analytic goal and distributional assumptions of the data (e.g. non-normal, discrete, finite support, etc). Here we propose a scheme that ignores the analytic goal — and, instead, aims to match the empirically-derived estimated correlation data and marginal characteristics. For normal data, the Pearson product-moment correlation describes the dependency completely. As we will see, however, simulating arbitrary random vectors with a given Pearson correlation matrix is computationally intense (Chen, 2001, Xiao (2017)), violating property **S1**. On the other hand, an analyst may consider the use of non-parametric correlation measures, such as Spearman's ρ and Kendall's τ, particularly when modeling non-normal data as in our application. Adding to the complexity is the well-known, marginal-dependent constraints on the possible bivariate correlation (a consequence of the bounds of bivariate distribution functions — the *Frechet-Hoeffding bounds* (Nelsen, 2007)). Denote the pointwise upper bound as $M(y_i, y_{i'})$ These bounds give strict restraints on the possible bounds and cannot be overcome through algorithm design. Yet not all multivariate simulation approaches obtain these bounds [for example REF]. Computationally, algorithms that rely on serial computation scale poorly to high dimensions (refs - hierarchy) and fail to make use of modern high-performance computing including parallelization and graphical processing unit acceleration (Li et al., 2019).

To meet the desired properties in the face of the challenging setting, we present a general-purpose, scalable multivariate simulation algorithm. The crux of the method lies in the construction of a Gaussian copula [refs]. The idea that many multivariate and marginally heterogeneous distribution can be constructed in such a manner is well known [refs]. This article's contribution lies in it's application to high-dimensional data through a high-performance implementation (`bigsimr`) and speed-focused algorithm design. The algorithm design relies on useful properties of non-standard correlation measures, namely Kendall's τ and Spearman's ρ. We'll show that quality of simulation does not require the use of the standard Pearson correlation matrix, even for normally distributed data. The purpose of our method is to generate random vectors that match exactly specified any possible Kendall's τ or Spearman's ρ, and approximately Pearson product-moment correlation.

The study proceeds with more details on our motivating example in RNA-sequencing breast cancer data. Then we describe and justify our simulation methodology, followed by extensive Monte Carlo studies under various distributional assumptions — emphasizing normal and non-normal scenarios. In these studies, we evaluate the accuracy and speed of our approach in comparison to other readily available software. After evaluating *in silico*, we revisit our motivating example by employing our method and compare our simulations to empirically-derived statistics. Finally, we'll make concluding remarks regarding the method's utility and future directions.

2 Background and notation

2.1 Gaussian copulas

We present an general-purpose, scalable multivariate simulation algorithm. The crux of the method is construction of a Gaussian copula. This idea is well known [refs]. A copula is a distribution function on $[0,1]^d$ describing a random vector with standard uniform marginals. Moreover, for any random vector $\mathbf{X} = (X_1, \ldots, X_d)$ with cumulative distribution function (CDF) F and marginal CDFs F_i there is a copula function $C(u_1, \ldots, u_d)$ so that

$$F(x_1, \ldots, x_d) = \mathbb{P}(X_1 \le x_1, \ldots, X_d \le x_d) = C(F_1(x_1), \ldots, F_d(x_d)), \;\; x_i \in \mathbb{R}, i = 1, \ldots, d.$$

A Gaussian copula is the case where all marginal CDFs F_i are the standard normal cdf, Φ. The Gaussian copula is the one that corresponds to a multivariate normal distribution with standard normal marginal distributions and covariance matrix $\mathbf{R}$. (Since the marginals are standard normal, this $\mathbf{R}$ is also the correlation matrix). If $F_{\mathbf{R}}$ is the CDF of such multivariate normal distribution, then the corresponding Gaussian copula $C_{\mathbf{R}}$ is defined through

$$F_{\mathbf{R}}(x_1, \ldots, x_d) = C_{\mathbf{R}}(\Phi(x_1), \ldots, \Phi(x_d)), \tag{1}$$

where $\Phi(\cdot)$ is the standard normal CDF. Note that the copula $C_{\mathbf{R}}$ is simply the CDF of the random vector $(\Phi(X_1), \ldots, \Phi(X_d))$, where $(X_1, \ldots, X_d) \sim N_d(\mathbf{0}, \mathbf{R})$.

Sklar's Theorem (ref) guarantee's that any random vector with computational feasible inverse CDFs (P4 above) and obtainable correlation matrix (within the Frechet bounds) can be obtained via transformations involving copula functions. Namely, to simulate a random vector $\mathbf{Y} = (Y_1, \ldots, Y_d)$, where $Y_i = F_i^{-1}(U_i)$,

$i = 1, \ldots, d$, we can construct a Gaussian copula $\mathbf{U} = (U_1, \ldots, U_d)$ viz $U_i = \Phi(X_i)$, $i = 1, \ldots, d$. When an Y_i is discrete, some care must be taken to define F_i. Letting

$$F_i^{-1} = inf\{y : F_i(y) \geq u\} \tag{2}$$

ensures that $Y_i \sim F_i$.

Simulation of a general multivariate random vector $\mathbf{T}$ with margins F_i based on this copula is quite simple. Ensuring that one obtains a certain dependence among the marginal distributions, however, proves challenging in our proposed Gaussian-copula-based scheme. Two core issues: 1) marginal characteristics induce bounds on the possible bivariate correlations and 2) monotone transformations deform the Pearson correlation which no closed form expression exists in general (ref? Chen2001). The Frechet bounds are well-known.

2.2 Measures of dependency

The population correlation coefficient, commonly called the Pearson (product-moment) correlation coefficient, describes the linear association between two random variables X and Y and is given by

$$\rho(X, Y) = \frac{E(XY) - E(X)E(Y)}{[var(X)var(Y)]^{1/2}} \tag{3}$$

As Madsen and Birkes (2013) and Mari and Kotz (2001) discuss, for a bivariate normal (X, Y) random vector the Pearson correlation adequately describes the dependency between the components. ρ's utility in non-normal or non-linear associations is lacking (Mari and Kotz (2001)). Rank-based (ordinal) approaches performance better in these settings, such as Spearman's (denoted ρ_s) and Kendall's τ. Define

$$\rho_s(X, Y) = 3\left[P\left[(X_1 - X_2)(Y_1 - Y_3) > 0\right] - P\left[(X_1 - X_2)(Y_1 - Y_3) < 0\right]\right] \tag{4}$$

where $(X_1, Y_1) \overset{d}{=} (X, Y), X_2 \overset{d}{=} X, Y_3 \overset{d}{=} Y$ with X_2 and Y_3 are independent of one other and of (X_1, Y_1). For continuous marginals, the measure works well due to zero probability of ties. For discrete marginals, however, one could perform a rescaled version of ρ_s for random variables X, Y with pmfs (or pdfs) $p(x)$ and $q(y)$, respectively.

$$\rho_{RS}(X, Y) = \frac{\rho_s(X, Y)}{\left[\left[1 - \sum_x p(x)^3\right]\left[1 - \sum_y q(y)^3\right]\right]^{1/2}} \tag{5}$$

Kendall's τ is the probability of concordant pairs minus the probability of discordant pairs and is given compactly as

$$\tau(X, Y) = \frac{\rho_s(X, Y)}{\left[\left[1 - \sum_x p(x)^3\right]\left[1 - \sum_y q(y)^3\right]\right]^{1/2}} \tag{6}$$

2.3 Marginal-dependent bivariate correlation bounds

The pairwise correlation between two correlated random variables cannot in general obtain the full range of possible values, $[-1, -1]$. The range, called the Frechet(-Hoeffding) bounds, is a well-known function of the marginal distributions and are given by (Barbiero and Ferrari, 2017):

$$\rho^{max} = \rho\left(F_1^{-1}(U), F_2^{-1}(U)\right), \quad \rho^{min} = \rho\left(F_1^{-1}(U), F_2^{-1}(1-U)\right) \tag{7}$$

where U is a uniform random variable in $(0,1)$, and F_1^{-1}, F_2^{-1} are the inverse cdf of random variables X_1 and X_2, respectively. For discrete random variables, define F^{-1} as in Equation (2).

3 Simulation algorithm

3.1 Algorithm description

This section describes the method for simulating a random vector $\mathbf{Y}$ with Y_i components for $i = 1, 2, \ldots, d$. Each Y_i has a specified marginal distribution function F_i and its inverse. To characterize dependency, every pair (Y_i, Y_j) has either a specified Pearson correlation (3), (rescaled) Spearman correlation (5) , or Kendall's τ (Equation (6)). The method only approximately matches the Pearson correlation in general, whereas the rank-based methods are exact.

The method is best understand as a **parallelized Gaussian copula** (see Equation (1). We shall see that constructing continuous joint distributions that match a target Spearman or Kendall's correlations computes easily when employing Gaussian copulas, since this measures are invariant under the monotone transformations involved [refs]. To do this, we take advantage of a closed form relationship [ref?] between Kendall's τ and Pearson's correlation coefficient for bivariate normal random variables:

$$r_{Pearson} = sin\left(\tau_{Kendall} \times \frac{\pi}{2}\right), \tag{8}$$

and similarly for Spearman's ρ (Kruskal, 1958),

$$\rho_{Pearson} = 2 \times sin\left(\rho_{Spearman} \times \frac{\pi}{6}\right). \tag{9}$$

For discrete marginals, achieving a target Spearman correlation under this scheme is possible by using components from Equation (5) to further adjust the input correlation matrix. Let the unscaled Spearman correlation coefficients be $\rho_s(Y_i, Y_{i'})$ for two marginal distributions and divide the target correlation by the product in the denominator of Equation (5). Let these adjustment factors be denoted as $a_i = \left[1 - \sum_y p_i(y)^3\right]^{1/2}$ and specifically rescale the target Spearman correlation matrix by

$$\rho_{rs}(Y_i, Y_{i'}) = \frac{\rho_s(Y_i, Y_{i'})}{a_i \times a_{i'}}. \tag{10}$$

In a similar fashion, we rescale Kendall's τ to adjust the input correlation matrix. The conversion formula is given by

$$\rho_{rs}(Y_i, Y_{i'}) = \frac{\rho_s(Y_i, Y_{i'})}{a_i \times a_{i'}}. \tag{11}$$

In contrast the rank-based correlations, matching specified Pearson correlation coefficients exactly is computational intense in this scheme. In general, there is no closed form correspondence and involving computing or approximating $\binom{d}{2}$ integrals of the form $EY_iY_j = \int\int y_i y_j f_{X|r}(F_i^{-1}(\Phi(z_i)), F_j^{-1}(\Phi(z_j))dy_i dy_j$, for $i, j = 1, 2, \ldots, d$. For accurate numeric approximation of these integrals, the functions must be evaluated hundreds of times. Others have used efficient Monte Carlo integration schemes (see Chen (2001)), but scale poorly to large dimension in reasonable times (property **S2**). Despite all this, if one does desire to characterize dependency using Pearson correlations, we often see in practice — and it is theoretically justified under certain conditions (Song (2000)) — that simply using the target Pearson correlation matrix as the initial conditions to our proposed algorithm will lead to approximate matching in the resultant distribution.

3.2 Simulation Algorithm

Putting the together the facts provided in the equations above, we come the following proposed simulation algorithm to produce a random vector $\mathbf{Y}$ with specified Spearman's correlation and marginal distributions. Note that all computational steps can be parallelized as each operation can be done or either the d marginals or the $\binom{d}{2}$ pairs for the correlation values. Even the generation of multivariate normal random vectors is parallelized through optimized matrix multiplication/decomposition routines.

3.2.1 Inputs

(1) Marginal characteristics including the same of distributional family and parameter values, denotes as F_i for marginal component random variable Y_i for $i = 1, \ldots, d$.
(2) A target (specified) Spearman's correlation matrix $\mathbf{R_{Spearman}}$ with each $\rho_{Spearman}$ within the Frechet limits. Alternatively, the rescaled Spearman's correlation (see Equation (11)) can be provided.
(3) An error tolerance ϵ for finding the nearest positive definite matrix a transformed correlation matrix (see step iii below). Alternatively, a maximum number of iterations can be supplied.

3.2.2 Algorithm

(i) Compute Spearman's ρ_{rs} (see Equation (11)) from the specified $\rho_{spearman}$ for each pair of marginal random variables $(Y_i, Y_{i'})$ with $i \neq i'$ when either marginal is discrete. In such large scale computations, it may be that numerically ρ_{rs} is be larger/smaller than the Frechet bounds (or even ± 1). To guard against this, set $\rho_{rs} = min(\rho_{rs}, M)$, where M is the upper Frechet bound (see [Section Introduction]) and similarly set $\rho_{rs} = max(\rho_{rs}, W)$, where W is the lower Frechet bound for this pair of distributions. Gather these ρ_{rs}'s into a new input correlation matrix $\mathbf{R_{rs}}$.

(ii) Convert $\mathbf{R_{Spearman}}$ into $\mathbf{R_{Pearson}}$ via $\rho_{Pearson} = 2 \times sin\left(\rho_{rs} \times \frac{\pi}{6}\right)$.

(iii) Finally, ensure that $\mathbf{R_{Pearson}}$ is positive definite, by finding the nearest positive definite correlation matrix $\mathbf{R}$ — using the `R` routine `nearPD` in the `Matrix` package — within an error tolerance of ϵ.

(iv) Generate $\mathbf{X} = (X_1, \ldots, X_d) \sim N_d(\mathbf{0}, \mathbf{R})$;

(v) Transform $\mathbf{X}$ to $\mathbf{U} = (U_1, \ldots, U_d)$ viz $U_i = \Phi(X_i)$, $i = 1, \ldots, d$;

(vi) Return $\mathbf{Y} = (Y_1, \ldots, Y_d)$, where $Y_i = F_i^{-1}(U_i)$, $i = 1, \ldots, d$;

4 The `bigsimr` R package

Short description and key features, including GPU acceleration.

The most computationally extensive lies in the second step of our algorithm. We use a optimized and parallelized multivariate normal simulator within the `R` package `mvnfast`. The rest of the code is also parallelized, but these steps run rapidly in serial computation except for extremely large dimension.

4.1 Basic use

short description

4.1.1 Specifying marginals

As stated earlier, to generate multivariate data, we need a list of marginals (and their parameters), and a correlation structure (matrix). The marginal distributions can be built up as a list of lists, where each sublist contains the information for the target distribution.

The things to point out here are that in each sublist (marginal), the first item is an unnamed character string with the R name of the distribution *without a letter prefix*. E.g. instead of `rnorm`, we pass in just `"norm"`. The second thing to note is that the remaining items are *named* arguments that go along with the distribution. A full list of built-in distributions is found in the appendix.

4.1.2 Specify correlation

The next step is to define a correlation structure for the multivariate distribution. This correlation matrix can either come from observed data, or we can set it ourselves, or we can generate a random correlation matrix via `bigsimr::rcor`. Let's create a simple correlation matrix where all off-diagonal elements are 0.5. Since we have 3 marginals, we need a 3×3 matrix.

```
     [,1] [,2] [,3]
[1,]  1.0  0.5  0.5
[2,]  0.5  1.0  0.5
[3,]  0.5  0.5  1.0
```

Finally we can generate a random vector with our specified marginals and correlation structure. The last argument, `type`, is looking to know what kind of correlation matrix it is receiving. Right now it can handle Pearson, Spearman, or Kendall.

4.1.3 Small sample

On my machine, there is no dedicated GPU, so I would see the following warning message once per session.

Taking a look at our random vector, we see that it is 10 rows and 3 columns, one column for each marginal.

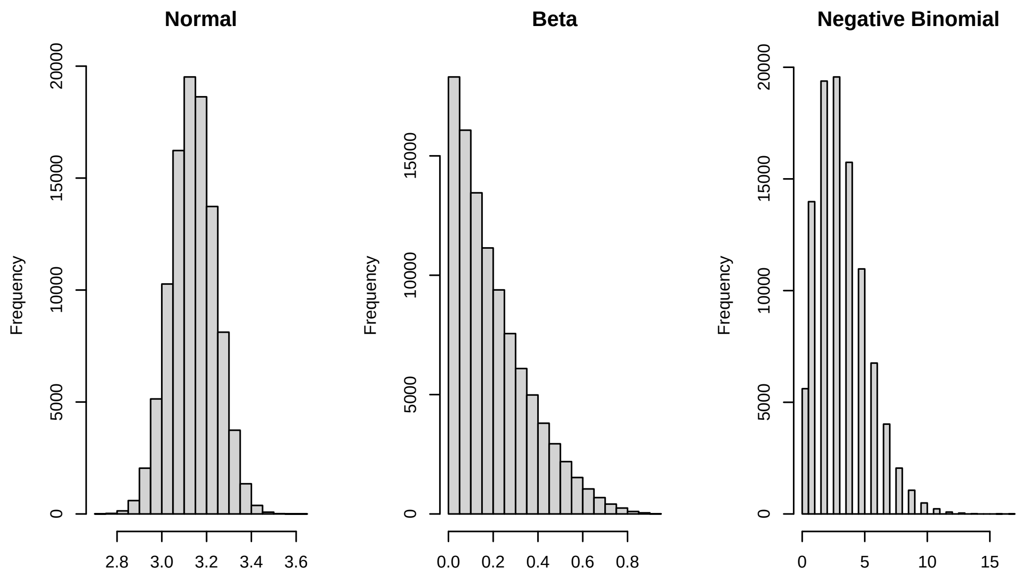
```
        [,1]     [,2] [,3]
  [1,] 3.044 0.095446    3
  [2,] 3.266 0.149347    4
  [3,] 3.210 0.368758    7
  [4,] 3.096 0.273881    1
  [5,] 3.262 0.167132    8
  [6,] 3.186 0.297752    4
  [7,] 3.244 0.378209    2
  [8,] 3.092 0.061396    2
  [9,] 3.153 0.388307    5
 [10,] 3.191 0.017892    2
 [11,] 3.021 0.143984    1
 [12,] 2.977 0.306241    2
 [13,] 3.327 0.311243    6
 [14,] 3.087 0.166009    3
 [15,] 3.146 0.462549    2
 [16,] 3.130 0.045254    3
 [17,] 3.215 0.341481    3
 [18,] 3.153 0.481414    9
 [19,] 3.311 0.536290    6
```

```
[20,] 3.054 0.183528    4
[21,] 3.017 0.011316    1
[22,] 3.195 0.293363    1
[23,] 3.017 0.077518    1
[24,] 3.024 0.041316    1
[25,] 3.272 0.330607    6
[26,] 3.218 0.206071    5
[27,] 2.984 0.001637    2
[28,] 3.156 0.114774    3
[29,] 3.269 0.089939    3
[30,] 3.103 0.177539    3
[31,] 3.147 0.138347    6
[32,] 3.157 0.093564    4
[33,] 3.318 0.356667    2
[34,] 3.111 0.231386    4
[35,] 3.174 0.659264    2
[36,] 3.218 0.293820    4
[37,] 3.264 0.240420    6
[38,] 3.072 0.255088    3
[39,] 3.131 0.168695    0
[40,] 3.280 0.491158    6
[41,] 3.189 0.238835    5
[42,] 3.017 0.011214    1
[43,] 3.145 0.022872    1
[44,] 3.111 0.144358    1
[45,] 3.236 0.085572    3
[46,] 3.221 0.309900    7
[47,] 2.955 0.014524    3
[48,] 3.069 0.011569    2
[49,] 2.970 0.075424    2
[50,] 3.265 0.233258    5
[51,] 3.125 0.223936    2
[52,] 3.118 0.100138    3
[53,] 3.042 0.113373    2
[54,] 2.993 0.335292    3
[55,] 3.073 0.020944    1
[56,] 3.178 0.256723    4
[57,] 3.242 0.679650    7
[58,] 3.204 0.228560    5
[59,] 3.325 0.254273    9
[60,] 2.918 0.109870    2
[61,] 3.106 0.134519    5
[62,] 3.035 0.022232    1
[63,] 3.238 0.029679    3
[64,] 3.163 0.159671    3
[65,] 3.250 0.265043    5
[66,] 3.139 0.109471    6
[67,] 2.952 0.069780    3
[68,] 3.092 0.103947    6
[69,] 3.163 0.196133    6
[70,] 2.993 0.079026    1
[71,] 2.990 0.043229    2
[72,] 3.057 0.018904    2
[73,] 2.985 0.009260    1
```

```
 [74,] 3.224 0.778705    6
 [75,] 3.231 0.238377    4
 [76,] 3.083 0.014756    2
 [77,] 3.175 0.262330    1
 [78,] 3.141 0.106700    3
 [79,] 3.047 0.007255    2
 [80,] 3.182 0.334552    2
 [81,] 3.184 0.303953    5
 [82,] 3.033 0.182557    2
 [83,] 3.114 0.227299    4
 [84,] 3.066 0.057184    5
 [85,] 3.077 0.069102    4
 [86,] 2.866 0.004398    1
 [87,] 2.920 0.100838    2
 [88,] 3.219 0.175613    6
 [89,] 3.080 0.094636    1
 [90,] 3.145 0.076677    4
 [91,] 3.176 0.243460    4
 [92,] 3.005 0.230508    2
 [93,] 3.056 0.315420    5
 [94,] 3.162 0.524840    5
 [95,] 3.332 0.685218    7
 [96,] 3.120 0.209939    0
 [97,] 3.184 0.174008    2
 [98,] 3.037 0.037463    2
 [99,] 3.064 0.013679    1
[100,] 3.098 0.101754    3
```

We can simulate many more samples and then check the histogram of each margin, as well as the estimated correlation between the columns.

4.1.4 Scaling up N



4.1.5 Evaluation

```
       [,1]   [,2]   [,3]
[1,] 1.0000 0.4989 0.4963
[2,] 0.4989 1.0000 0.4929
[3,] 0.4963 0.4929 1.0000
```

We can see that even with 100,000 samples, the estimated correlation of the simulated data is not exactly the same as the target correlation. This can be explained by the fact that some correlations are simply not possible due to the discrete nature of certain distributions. Another possibility is that the copula algorithm is biased and needs correction.

5 Monte Carlo evaluation and Comparsions to other software

Before applying our methodology to real data simulation, we conduct several Monte Carlo studies to investigate method performance in comparison to other existing implementations. We focus the numerical experiments on assessing how well the procedure scales to high dimension with respect to reasonable computation times (property S1 above) and accurately matching marginal and dependency parameters. The simulations will proceed in increasing complexity — leading up to the setting in our motivating example. We begin by exploring simple exchangeable (constant) correlation structures under a large number of simulation replicates while applying the algorithm above to first continuous then discrete marginal distributions. We conclude the simulation studies with 10^5 replicated data sets with sample sizes and distributional characteristics corresponding the estimates from the motivating example data.

The CPU-based computation times we report were from runs using eight dual-threaded 3.9GHz Xeon Gold processors on a linux workstation, allowing for parallel instructions over 16 cores. We chose 15 cores for our experiments.

5.1 Other multivariate simulation software

Describe the competing software and algorithms briefly.

- `copula`
- `Genord`
- `Multiord`
- `nortaRA` only does Pearson matching.

5.2 Simulation I: Bivariate simulation using rank-based correlation

5.2.1 Continuous example: Bivariate Exponential

Both Spearman's and Kendall's measures of dependency are invariant under strictly monotone transformations (ref). This in turn provides exact simulation of continuous marginals F_i $i = 1, \ldots, d$ since their corresponding quantile function are increasing monotonically over their support. The results below provide computations times for $N = 10^5$ under increasingly large dimension d. For simplicity we assume multivariate Normal with varying parameters and to additionally demonstrate the utility of Kendall's τ even in the Gaussian setting (as opposed to the nature choice of Pearson's correlation). To layer on complexity, we now assume a multivariate gamma distribution with fixed marginal parameters.

Compare speed and accuracy.

```
## computation times normal
```

5.2.2 Discrete example: Bivariate Geometric

Compare speed and accuracy.

5.3 Simulation II: Ultra HD multivariate simulation with identical margins and exchangeable rank-based correlation.

5.3.1 UHD Continuous example: Multivariate Gamma

Compare speed and accuracy.

5.3.2 UHD Discrete example: Multivariate Negative Binomial

Compare speed and accuracy.

5.4 Simulation III: Ultra HD multivariate simulatino with heterogenous margins and exchangeable rank-based correlation.

5.4.1 UHD Heteregeous continuous example: Multivariate Gamma

Compare speed and accuracy.

5.4.2 UHD Heteregeousdiscrete example: Multivariate Negative Binomial

Compare speed and accuracy.

6 Some applications

6.1 Simulating Ultra High Dimensional RNA-seq data

We apply our methodology to simulate RNA-sequencing data sets based on samples derived from breast cancer patients' tumors (the BRCA data set in TCGA). The data are freely available as part of the BRCA data set within the The Cancer Genome Atlas data warehouse and were downloaded on BLAH using the Harvard's Broad Institute's Firehouse interface. For each of the 1093 patients, the gene expression (abundance of messenger-RNA) from 20501 genes were counted using RNA-sequencing. The resultant data set can be represented as a matrix of size 1093 x 20501 with discrete values aligned to HUGO gene symbols for each patient. In this illustrative case study, we model the gene-wise marginal distributions as heterogeneous negative binomial with pmf given by Equation (12) with two parameters, p and r (OR SWITCH TO mu and delta). Modeling RNA-seq counts as negative binomially distributed random variables is commonplace (e.g., Zhao et al. (2018)) since overdispersion is often observed for most expressed genes. We begin by estimating the marginal negative binomial parameters for each group using a simple method of moments approach by matching the negative binomial parameters with the sample mean and variance. Table/Figure XXX summarizes the estimated means, variances, and NB parameters as described in Equation (12). Notably some genes show underdispersion compared to a Poisson random variable. For simplicity we exclude these cases, but nothing in the later described simulation method prevents using a different marginal distribution in those cases (such as a Maxwell distribution REF Sellers).

Intergene correlation (Schissler et al., 2019).

$$g(n) = \mathbb{P}(Y = n) = \frac{\Gamma(n+r)}{\Gamma(r)n!} p^r (1-p)^n, \ \ n \in \mathbb{N}_0. \tag{12}$$

```
## full workflow simulating RNA-seq data
allDat <- readRDS( file = "~/Downloads/complete_processed_tcga2stat_RNASeq2_with_clinical.rds" )
lastClinical <- which( names(allDat) == 'tumorsize' )
brca <- allDat[ allDat$disease == "BRCA", (lastClinical+1):ncol(allDat) ]
## remove naively the RSEM adjustment
brca <- round(brca, 0)
ncol(brca) ## num of genes 20501
[1] 20501
## compute the median expression! avoid the 0s
brcaMedian <- apply(brca, 2, median)
## retain top (1-probs)*100% highest expressing genes for illustration
myProb <- 0.95
cutPoint <- quantile( x = brcaMedian, probs = myProb )
## genesToKeep <- names( brcaMedian ) [ which(brcaMedian >= cutPoint) ]
genesToKeep <- names( brcaMedian ) [ which(brcaMedian >= cutPoint) ]
brca <- brca[ , genesToKeep ]
## ncol(brca) / 20501
(d <- length(genesToKeep))
[1] 1026
```

The data are overdispersed:

```
logVarOverMean <- function( x ) {
    log ( var(x) / mean (x) )
}
## check whether NB makes sense

summaryBRCA <- brca %>%
    summarize_all(  list(~ logVarOverMean( . ) ) )
summaryBRCA <- as.data.frame( t(as.data.frame( summaryBRCA )) )
names(summaryBRCA) <- "logVarOverMean"
## sum(summaryBRCA$logVarOverMean < 0)  ## no underdispersed genes
ggplot(data = summaryBRCA, mapping = aes(x = logVarOverMean)) +
  geom_histogram(color = "white", bins = 20)
```

We see that the all genes studies display overdispersion (variance greater than mean) compared to a Poisson model for the gene counts. The genes are highly heterogeneous even when dealing with highly expressed genes after filtering. The facts taken together motivate the negative binomial model for the marginal distributions. The genes are slightly more variable within the deceased group (expected with smaller sample size and potentially due to biomedical considerations).

Next we perform some essential pre-processing of the data. Since our goal is to simulate meaningful multi-variate constructions, we must first find a subset of 20501 genes that are expressed. In order words, we'll filter out the low expressing genes to allow greater range of possible d-variate correlations (see Nikoloulopoulos and Karlis (2010) for details OR SHOULD I EXPLAIN THIS MORE CAREFULLY?). Notably, many RNA-seq analytic workflows filter out low expressing genes (see Conesa et al. (2016)) and this, admittedly, is usually done in an *ad hoc* fashion. In this study, we prefer an inclusion criterion that is theoretically motivated. Based on the results in Section BLAH, we filter genes with a sample mean less than 12. In this way, we

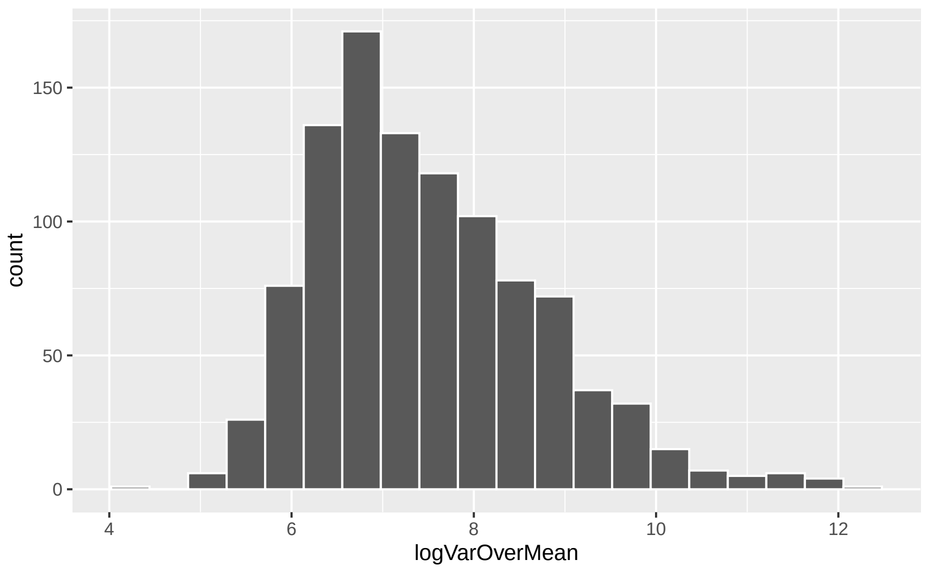


Figure 1: The genes display extreme overdispersion. Using group-specific sample means and variances for each of the d=‘r d‘ genes, we display the distributions oflog of variance to mean ratio.

remove 4762 (23.2%) of the genes, retaining 15,739 genes for further analysis. The filtered low-expressing genes will be simulated as independent negative binomial random variables to form a complete simulated transcriptome.

RNA-sequencing data derived from breast cancer patients (n=1093), with 20501 genes measured (from TCGA). We filter to genes with average expression greater than 10000 counts (d = 4777) across both groups. Consider basic differentially expressed genes (DEG) analysis between surviving (n0 = 941) and deceased (n1 = 152) patients. This setting can be described as large-scale simultaneous hypothesis testing, under correlation. We should evaluate existing and new methodology using simulations that reflect dependency among variables. But multivariate simulation tools often scale poorly to high dimension or do not model the full range of dependency. Perform a two-sample t test for each 4777 genes on the observed RNA-seq counts (*empirical*). Call a “Z score” for the i^{th} gene $Z_i = \Phi^{-1}(t_{\approx 272}(t_i))$.

Our method requires some measure of that the covariance structure is pre-specified. In practice, however, high-dimensional covariance estimation is not that easy. Indeed there is much recent interest in this area (cite a few recent high visibility articles — see the GPU-NORTA pub). Here we do not provide a comprehensive review on the topic. Instead we only seek to use covariance estimator that only guarantees positive-semidefiniteness while maintaining adequate estimation properties. To this end we chose to use the condition-number-regularized (CONDREG) approach of Won et al. (2013). This covariance estimation procedure restricts the ratio of the largest eigenvalue to the smallest eigenvalue to improve numerical results. But the method employs a penalized Gaussian likelihood. And so, rather than estimating the covariance on the discrete counts directly, we first perform a $log_2(x+1)$ transformation to better suit this modeling assumption. CONDREG requires a parameter, κ_{max}, that controls the largest condition number of the resultant estimated covariance matrix. Here we aim to allow a large number of nonzero pairwise correlations and so arbitrary set $\kappa_{max} = 10^4$. The routine ran without issue on a MacBook Pro carrying BLAH in XXX units of time.

Most pairwise correlations are nearly zero with XXX % less than $|0.1|$. Yet there are dense subsets of correlated genes. And even small correlations among many variables can disrupt the operating characteristics of commonly used statistical inferential procedures Schissler et al. (2018).

In this case study in the large-scale hypothesis testing in pursuit of detecting differentially expressed genes, we aim to simulate RNA-seq counts using a multivariate negative binomial. To evaluate the simulation’s utility, we systematically compare the simulated results to the empirical results. In short we computed t-statistics and p-values for each of $d = 4777$ genes between two groups of breast cancer patients (1=deceased, 0=alive; $n_0 = 152, n_1 = 941$). Further details on the empirical results are provided in Section BLAH. We compare two aspects in the simulation design 1) Choice of correlation measure while modeling marginal distributions consistently and 2) how well does the best-performing simulation agree with the empirical results. Informed by the simulation studies in Section BLAH, we anticipate a far amount of variation at such small sample sizes and large d and so replicate the entire study 100 times.

To recap from the discussion above (Section BLAH), we estimated negative binomial parameters for each of the $d = 4777$ genes and also their Pearson, Spearman, and Kendall's correlation coefficients from n=1093 breast cancer patients, within each vital status group (0=deceased, 1=alive). We generated 100 synthetic data sets for each group, with the number of d-dimensional random vectors produced equal to the corresponding sample sizes ($n_0 = 152, n_1 = 941$). These 100 samples are used to understand uncertainty in this setting and we will use the mean simulated values to compare to the empirical (and also explore the worst-case scenarios).

Model genes as marginally negative binomial with heterogeneous gene-wise parameters. Compute sample correlations (using desired dependency measure) for the 11,407,476 genes pairs, for each group. To simulate gene expression counts, our goal is to produce a random vector $\mathbf{Y} = (Y_1, \ldots, Y_d)$ with **correlated** NB components. To do that, we start with a sequence of **independent** Poisson processes $N_i(t)$, $i = 1, \ldots, d$, where the rate of the process $N_i(t)$ is $\lambda_i = (1 - p_i)/p_i > 0$ (so that $p_i = 1/(1 + \lambda_i)$). Now, we let $\mathbf{T} = (T_1, \ldots T_d)$ have a multivariate distribution on $\mathbb{R}^d_+$ with the PDF $f_{\mathbf{T}}(\mathbf{t})$. Then, we define

We rescaled the target Spearman correlation to account for the probability of ties via Equation (10). The serial computation took approximately 10 minutes on XXX. Notably, the difference between target and adjusted matrices was very small (mean relative difference: 0.708) in this particular configuration of negative binomial parameters. For problems with smaller counts and higher probabilities of ties, the adjustment can be substantial (as observed in the simulation studies above in Section BLAH).

```
## full workflow simulating RNA-seq data

## 1. Estimate Spearman's correlation on the count data
corType <- 'spearman'
system.time( rho <- bigsimr::fastCor( brca, method = corType ) )
   user  system elapsed
  0.654   0.013   0.673
## Describe the correlations
```

```
## 2. Estimate NegBin parameters using Method of Moments
estimateNegBinMoM <- function(tmpGene, minP = 1e-5) {
    tmpMean <- mean(tmpGene)
    tmpVar <- var(tmpGene)
    ## relate to nbinom parameters
    ## See ?rbinom for details.
    p <- tmpMean / tmpVar
    if (p < minP) {p <- minP } ## maybe add noise here
    n <- ( tmpMean * p) / ( 1  - p )
    ## format for bigsimr margins
    return( list("nbinom", size = n, prob = p) )
}
brcaMargins <- apply( unname(as.matrix(brca)), 2, estimateNegBinMoM )
## describe the margins
## check for too small prob
## head( sort( unlist( lapply( brcaMargins, function(x) {x$prob} ) ) ) )
## head( sort( unlist( lapply( brcaMargins, function(x) {x$size} ) ) ) )
```

```r
## 3. Generate the simulated samples
## N <- 10
## system.time( simBRCA <- rvec(N, rho = rho[1:2, 1:2], params = brcaMargins[1:2], type = corType, adju
N <- nrow(brca)
system.time( simBRCA <- rvec(N, rho = rho, params = brcaMargins, cores = CORES, type = corType, adjustF
simBRCA[1:2, 1:2 ]
```

```r
## Describe and plot real data
GGally::ggpairs(data = as.data.frame(brca[ ,1:3] ) )

## Describe and plot simulations
GGally::ggpairs(data = as.data.frame(simBRCA[ ,1:3] ) )

## check 1st moments
trueMu <- colMeans(brca)
simMu <- colMeans(simBRCA)
qplot( x = trueMu, y = simMu ) + geom_abline(slope = 1, intercept = 0)

## check 2nd moment
trueVar <- apply( brca, 2, var)
simVar <- apply( simBRCA, 2, var)
qplot( x = trueVar, y = simVar ) + geom_abline(slope = 1, intercept = 0)

## check correlation
system.time( simRho <- bigsimr::fastCor( simBRCA, method = corType ) )
rho[1:5, 1:5]
simRho[1:5, 1:5]
trueRho <- rho[lower.tri(rho)]
simRho <- simRho[lower.tri(simRho)]
qplot( x = trueRho, y = simRho ) + geom_abline(slope = 1, intercept = 0)
```

6.2 Ultra High-Dimensional simulation-based correlation hypothesis testing under

Since many of the correlations are near zero, one could ask whether it is possible that

6.3 Simulation-based computation of correlation bounds

Using the Generate, sort, and correlate algorithm Demirtas and Hedeker (2011)

```r
## simulation based
## devtools::install_github("adknudson/bigsimr", ref="develop")
library(bigsimr)
## ?rvec

d <- 5
rho <- rcor(d)
margins <- list(
    list("nbinom", size = 5, prob = 0.3),
    list("exp", rate = 4),
    list("binom", size = 5, prob = 0.7),
    list("norm", mean = 10, sd = 3),
    list("pois", lambda = 10)
)

params = margins
rho = rho
## cores = parallel::detectCores() - 1
cores = 2
reps = 1e3
type = "spearman"

## help(package="bigsimr")
rho_bounds <- computeCorBounds(params = margins, cores = cores, type = type, reps = reps)
rho_bounds
```

6.4 Simulation-based joint probability calculations

```r
## full workflow simulating RNA-seq data
```

7 Conclusion/Discussion

- Although we recommend to use the rank-based measures
- discuss the omission (or inclusion) of high dimensional covariance estimators?
- We've shown that the multivariate NB simulations outperform the Independent sims. But the agreement is still poor. This could motivate other more appropriate models for high-expressing RNA-seq data than the negative binomial distribution. This algorithm lends itself to exploration of flexible probability models to inspire model selection in the development novel RNA-seq analytic methodology.

We've introduced a general-purpose high-dimensional multivariate simulation algorithm and provide a high-performance implementation called `bigsimr` (github url). The parallelized (multi-core) algorithm is simple and easy to understand as an application of Gaussian copulas. This method is largely inspired by Madsen and Birkes (2013), but with contributions with regard to scalability, implementation, and application in high-throughput biomedical data (RNA-sequencing). We advocate the use of Kendall's τ as it better captures correlation among components of non-normal, as well as non-linear patterns of association. Moreover, the matching Kendall's τ is nearly trivial due to the invariant of monotone transformations, after an adjustment to the input correlation matrix. Interestingly, our simulation studies show the use of Kendall's τ to works as well as using Pearson correlation coefficient when simulating from multivariate normal distributions.

We also show utility in our methodology through an application to differential gene expression analysis from RNA-sequencing data. The application results show that correlations indeed matter in the large-scale hypothesis testing, as many others have noted (for example, see Efron (2007), Wu and Smyth (2012)). We also hope that the application provides an example workflow — and other strategy to use in simulation design. Even the best-performing simulations we provide show a gap from the empirical distribution. To keep the demonstration straightforward, we did not attempt to match the empirical distribution of test statistics as precisely as possible. Yet our methodology could be more creatively applied to meet that goal. One could consider marginal distributions from different families, such as Poisson for a subset of genes. One could imagine finding a best fitting probability distribution among a class of distributions for each gene and this could perhaps a better fit. One could imagine additional structures/features in the data that an analyst could model to improve the correspondence with the empirical values, for example row correlations and high-dimensional covariance estimators (see Won et al. (2013)).

Mention `rslurm`

There are of course limitations to the methodology and implementation. The algorithm requires invertible marginal cdfs. This leaves some restriction to the available marginal probability distributions (DOES IT? FOR EXAMPLE?) An additional source of error is after converting dependencies (step i; $\mathbf{R}_{\mathbf{Kendall}}$)) the resultant matrix may not be positive definite (PD). Then the nearest PD matrix is used. From a practical computing standpoint, the user's computing resource provides the ultimate limit on how large a dimension can be simulated using the `bigsimr` R package. A user must consider carefully the available memory and cores when conducting a Monte Carlo experiment. The algorithm does amends itself well to parallelization and graphical processing unit acceleration (Li et al., 2019) and advanced users may take advantage of those techniques.

Future work includes developing scalable algorithms to match the Pearson correlation matrix exactly. Further, more sophisticated approaches could involve simulation algorithms that are "estimation aware" and so could explore the role of covariance estimation within the simulation. Lastly, these may reveal new approaches to estimating and evaluating high dimensional regression and Bayesian models. On the implementation side, employing graphical process unit acceleration could produce substantial speedups over our multi-core approach.

8 Supplementary Materials

We provide an open-source implementation of our methology as the `bigsimr` R package, hosted on github.

9 Acknowledgement(s)

The authors gratefully acknowledge the helpful discussions with Professor Walter W. Piegorsch and Professor Edward J. Bedrick during this project's conception.

10 Disclosure statement

The authors report no conflict of interest. DO WE?

11 Funding

FUNDING FROM EVERYONE HERE. CP3 grant.

12 Nomenclature/Notation

13 Notes

13.0.1 List of supported distributions

Mention Python?

References

Barbiero, A. and Ferrari, P. A. (2017). An R package for the simulation of correlated discrete variables. *Communications in Statistics - Simulation and Computation*, 46(7):5123–5140.

Chen, H. (2001). Initialization for NORTA: Generation of Random Vectors with Specified Marginals and Correlations. *INFORMS Journal on Computing*, 13(4):312–331.

Conesa, A., Madrigal, P., Tarazona, S., Gomez-Cabrero, D., Cervera, A., McPherson, A., Szcześniak, M. W., Gaffney, D. J., Elo, L. L., Zhang, X., and Mortazavi, A. (2016). A survey of best practices for RNA-seq data analysis.

Demirtas, H. and Hedeker, D. (2011). A practicalway for computing approximate lower and upper correlation bounds. *American Statistician*, 65(2):104–109.

Efron, B. (2004). Large-scale simultaneous hypothesis testing: The choice of a null hypothesis. *Journal of the American Statistical Association*.

Efron, B. (2007). Correlation and large-scale simultaneous significance testing. *Journal of the American Statistical Association*, 102(477):93–103.

Kruskal, W. H. (1958). Ordinal Measures of Association. *Journal of the American Statistical Association*, 53(284):814–861.

Li, X., Schissler, A. G., Wu, R., Barford, L., and Harris, Fredrick C., J. (2019). A graphical processing unit accelerated NORmal-To-Anything algorithm for high dimensional multivariate simulation. *Advances in Intelligent Systems and Computing*, pages 339–346.

Madsen, L. and Birkes, D. (2013). Simulating dependent discrete data. *Journal of Statistical Computation and Simulation*.

Mari, D. D. and Kotz, S. (2001). *Correlation and dependence*. World Scientific.

Nelsen, R. B. (2007). *An Introduction to copulas*. Springer Science & Business Media, New York, 2 edition.

Nikoloulopoulos, A. K. (2013). Copula-based models for multivariate discrete response data. In *Lecture Notes in Statistics: Copulae in Mathematical and Quantitative Finance*, pages 231–249. Springer, Heidelberg, 213 edition.

Nikoloulopoulos, A. K. and Karlis, D. (2010). Modeling multivariate count data using copulas. *Communications in Statistics: Simulation and Computation*.

Schissler, A. G., Aberasturi, D., Kenost, C., and Lussier, Y. A. (2019). A Single-Subject Method to Detect Pathways Enriched With Alternatively Spliced Genes. *Frontiers in Genetics*, 10(414).

Schissler, A. G., Piegorsch, W. W., and Lussier, Y. A. (2018). Testing for differentially expressed genetic pathways with single-subject N-of-1 data in the presence of inter-gene correlation. *Statistical Methods in Medical Research*, 27(12):3797–3813.

Song, P. X.-k. (2000). Multivariate Dispersion Models Generated from Gaussian Copula. *Scandinavian Journal of Statistics*, 27(2):305–320.

Wang, Z., Gerstein, M., and Snyder, M. (2009). RNA-Seq: A revolutionary tool for transcriptomics.

Won, J.-H., Lim, J., Kim, S.-J., and Rajaratnam, B. (2013). Condition-number-regularized covariance estimation. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 75(3):427–450.

Wu, D. and Smyth, G. K. (2012). Camera: A competitive gene set test accounting for inter-gene correlation. *Nucleic Acids Research*.

Xiao, Q. (2017). Generating correlated random vector involving discrete variables. *Communications in Statistics - Theory and Methods*.

Zhao, L., Wu, W., Feng, D., Jiang, H., and Nguyen, X. (2018). Bayesian Analysis of RNA-Seq Data Using a Family of Negative Binomial Models. *Bayesian Analysis*, 13(2):411–436.