# Contents

**5 Random vectors and objects** **3**
5.1 Generalizations of one-dimensional methods . . . . . . . . . . . . 4
5.2 Multivariate normal and *t* . . . . . . . . . . . . . . . . . . . . . . 7
5.3 Multinomial . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 12
5.4 Dirichlet . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 14
5.5 Multivariate Poisson and other distributions . . . . . . . . . . . 17
5.6 Copula-marginal sampling . . . . . . . . . . . . . . . . . . . . . . 19
5.7 Random points on the sphere . . . . . . . . . . . . . . . . . . . . 23
5.8 Random matrices . . . . . . . . . . . . . . . . . . . . . . . . . . 27
5.9 Example: Classification error rates . . . . . . . . . . . . . . . . . 32
5.10 Random permutations . . . . . . . . . . . . . . . . . . . . . . . . 34
5.11 Sampling without replacement . . . . . . . . . . . . . . . . . . . 39
5.12 Random graphs . . . . . . . . . . . . . . . . . . . . . . . . . . . 41
End notes . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 49
Exercises . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 53

# 5

# Random vectors and objects

In Chapter 4 we built up methods for sampling non-uniform random variables. Here we sample random vectors in $\mathbb{R}^d$ for $d \geqslant 1$. The challenge for the multidimensional case is to give the components of the random vectors the right dependence.

The three main ways to sample non-uniform random variables, namely inversion, acceptance-rejection and mixture sampling, all have natural $d$-dimensional generalizations. Unfortunately, all of these methods run into practical difficulties in the multidimensional setting. The extreme difficulty of some multidimensional sampling problems is one of the main motivations for Markov chain Monte Carlo and Sequential Monte Carlo methods in Chapters 11 and 15, respectively.

After considering generic methods we look closely at some specific multidimensional distributions for which good sampling methods are available. The most important of these are the multivariate normal, multivariate $t$, Dirichlet and multinomial distributions.

Following the methods for specific multivariate distributions, we introduce another generic method, copula-marginal sampling, which allows us to use the dependence structure from one family with marginal distributions of another. Copula-marginal sampling reduces to the QQ transformation when $d = 1$, but having $d > 1$ introduces so much extra complexity that it is fair to consider it a fourth generic method.

Finally, we consider how to sample certain objects that while not always thought of as vectors, can be represented by random vectors with dependent components. These include: points on a sphere, rotation matrices, Wishart matrices, permutations, samples without replacement and random graphs. In this chapter, we assume that $d < \infty$. Random processes, which are inherently infinite-dimensional are considered in Chapter 6.

## 5.1 Generalizations of one-dimensional methods

In this section we generalize the standard one-dimensional methods, inversion, acceptance-rejection and mixtures, to the $d$-dimensional setting. The good news is that all of these methods extend naturally to $d \geqslant 1$. The bad news is that, despite a few noteworthy successes covered later in this chapter, the methods all tend to become unwieldy.

The most straightforward approach to sampling a random vector $\boldsymbol{X}$ is to sample the components $X_1, X_2, \dots, X_d$ in order. First, we sample $X_1$ from its marginal distribution and then we sample each successive component, conditionally on the others. This sequential generation is justified by writing

$$f(\boldsymbol{x}) = \prod_{j=1}^{d} f_{j|1:(j-1)}\bigl(x_j \mid \boldsymbol{x}_{1:(j-1)}\bigr) \tag{5.1}$$

where the $j = 1$ factor is just the marginal distribution of $X_1$.

We can generalize inversion to $d$-dimensional problems, via sequential inversion, a conceptually very simple method based on (5.1). In ***sequential inversion***, we sample $\boldsymbol{U} = (U_1, \dots, U_d) \sim \mathbf{U}(0,1)^d$ and then we set

$$\begin{aligned} x_1 &= F^{-1}_{1|\varnothing}(u_1), \quad \text{and,} \\ x_j &= F^{-1}_{j|1:(j-1)}\bigl(u_j \mid \boldsymbol{x}_{1:(j-1)}\bigr), \quad \text{for} \quad j = 2, \dots, d. \end{aligned} \tag{5.2}$$

Here $F_{1|\varnothing}$ is the marginal CDF of $X_1$. We can sample the components of $X$ in any order that has tractable conditional CDFs.

**Example 5.1.** Let $\boldsymbol{X}$ have the planar density function

$$f(\boldsymbol{x}) = \begin{cases} x_1 + x_2, & \boldsymbol{x} \in [0,1]^2 \\ 0, & \text{else.} \end{cases}$$

The marginal density of $X_1$ is $f_1(x_1) = \int_0^1 f(x_1, x_2)\,\mathrm{d}x_2 = x_1 + 1/2$. Therefore the marginal CDF of $X_1$ is $F_1(x_1) = \int_0^{x_1} f_1(v)\,\mathrm{d}v = (x_1^2 + x_1)/2$ for $0 \leqslant x_1 \leqslant 1$. The conditional density of $X_2$ given that $X_1 = x_1$ is

$$f_{2|1}(x_2 \mid x_1) = \frac{x_1 + x_2}{\int_0^1 x_1 + x_2\,\mathrm{d}x_2} = \frac{x_1 + x_2}{x_1 + 1/2}.$$

Integrating this density, the conditional CDF of $X_2$ given $X_1 = x_1$ is

$$F_{2|1}(x_2 \mid x_1) = \frac{x_1 x_2 + x_2^2/2}{x_1 + 1/2}.$$

Both CDFs are of the form $Ax^2 + Bx$ and to solve $Ax^2 + Bx = u$ we will take $x = (\sqrt{B^2 + 4Au} - B)/(2A)$. Now we are ready to do sequential inversion. We sample $\boldsymbol{U} \sim \mathbf{U}(0,1)^2$. Then we put

$$X_1 = F_1^{-1}(U_1) = \sqrt{1/4 + 2U_1} - 1/2, \quad \text{and}$$

$$X_2 = F_{2|1}^{-1}(U_2 \mid X_1) = \sqrt{X_1^2 + U_2(2X_1+1)} - X_1,$$

after some simplification.

Like one-dimensional inversion, sequential inversion gives us very good control over the simulation and it is well suited to more balanced sampling methods such as stratification. The difficulty with sequential inversion is that outside of simple settings like Example 5.1 we may find that $F_{j|1:(j-1)}(x_j \mid \boldsymbol{x}_{1:(j-1)})$ is not so easy to invert. Sequential inversion is practical when we have good ways of handling the many conditional distributions that arise. If the inversion for $X_j$ requires expensive setup time for each possible $\boldsymbol{x}_{1:(j-1)} \in \mathbb{R}^{j-1}$ and that work cannot be reused, then sampling $\boldsymbol{X}$ this way will be very slow.

Some multivariate distributions are easy to sample by sequential inversion. We will find sequential inversion samplers for the multivariate normal distribution in §5.2 and for the multinomial distribution in §5.3.

The sequential approach can be applied using any of our one-dimensional methods to get $X_j$ given $X_k = x_k$ for $1 \leqslant k < j$. We could mix them, using inversion for some components, special transformations for some others and acceptance-rejection sampling for the rest. Methods that use conditional acceptance-rejection are now quite well developed. We will need variance reduction ideas from later chapters in order to study them. We postpone discussion of those techniques until Chapter 15 on sequential Monte Carlo.

Sequential inversion generates $\boldsymbol{X}$ by a transformation of $\boldsymbol{U}$. As in the one-dimensional case there are some useful special purpose (non-generic) transformations. The Box-Muller method of §4.6 uses a transformation from $\mathbf{U}(0,1)^2$ to $\mathcal{N}(0, I_2)$. We can extend sequential inversion, and other transformation methods, by using them to make proposals in acceptance-rejection sampling.

Acceptance-rejection sampling also generalizes in a straightforward way to the case with $d \geqslant 1$. To sample from a density $f(\boldsymbol{x})$ on $\mathbb{R}^d$ we find another density $g(\boldsymbol{x})$ such that we can sample $\boldsymbol{X} \sim g$ and can prove that $f(\boldsymbol{x}) \leqslant cg(\boldsymbol{x})$ for a known constant $c < \infty$. We can use Algorithm 4.5 as written, changing the scalar random variables there to vectors. Theorem 4.2 there also applies to the multidimensional case upon replacing intervals of the form $(-\infty, x]$ by sets $\prod_{j=1}^d (-\infty, x_j]$. As in the one dimensional case, we have to generate $c$ samples from $g$ for each delivered sample from $f$, on the average.

The geometry behind acceptance-rejection sampling also extends to $d$ dimensions. Let

$$\mathcal{S}_c(f) = \{(\boldsymbol{x}, z) \mid 0 \leqslant z \leqslant cf(\boldsymbol{x}),\ \boldsymbol{x} \in \mathbb{R}^d\}$$

be a $d+1$-dimensional solid region lying below a multiple of $f(\boldsymbol{x})$ and with its last coordinate nonnegative. If $(\boldsymbol{X}, Z) \sim \mathbf{U}(\mathcal{S}_c(f))$ then $\boldsymbol{X} \sim f$. Conversely if $\boldsymbol{X} \sim f$ and $Z \mid \boldsymbol{X} = \boldsymbol{x} \sim \mathbf{U}(0, cf(\boldsymbol{x}))$ then $(\boldsymbol{X}, Z) \sim \mathbf{U}(\mathcal{S}_c(f))$. In other words, Theorems 4.3 and 4.4 apply to the multidimensional case.

In the one-dimensional case, the geometric point of view established that we could use unnormalized densities $\tilde{f}$ and $\tilde{g}$. The same holds in $d$ dimensions using generalizations of Theorems 4.3 and 4.4. Just as in Theorem 4.5, we may get $\boldsymbol{X} \sim f$ by sampling candidates $\boldsymbol{Y}$ from a density $g$ proportional to $\widetilde{g}$, and accepting them with probability $A(\boldsymbol{Y}) = \widetilde{f}(\boldsymbol{Y})/(c\widetilde{g}(\boldsymbol{Y}))$ where $\widetilde{f}$ is proportional to $f$ and $\widetilde{f}(\boldsymbol{y}) \leqslant c\widetilde{g}(\boldsymbol{y})$ holds for all $\boldsymbol{y}$.

**Example 5.2.** Let $f$ be the uniform distribution in the unit ball $B_d = \{\boldsymbol{x} \in \mathbb{R}^d \mid \|\boldsymbol{x}\|^2 \leqslant 1\}$ and let $g$ be $\mathbf{U}[-1,1]^d$. By using unnormalized densities $\widetilde{f} \propto 1_{\boldsymbol{x}\in B_d}$ and $\widetilde{g} \propto 1_{\boldsymbol{x}\in[-1,1]^d}$ we find that acceptance-rejection sampling simplifies to drawing candidates $\boldsymbol{Y} \sim g$ and delivering as $\boldsymbol{X}$ those for which $\|\boldsymbol{Y}\| \leqslant 1$. The acceptance probability $1/c$ is

$$\frac{\mathbf{vol}(B_d)}{2^d} = \frac{\pi^{d/2}}{2^d\Gamma(1+d/2)}.$$

For $d = 2$ we get an acceptance probability of $\pi/4 \doteq 0.785$. This is high enough that uniform sampling within the unit circle this way is competitive. For $d \geqslant 9$ the acceptance ratio is below 1% and for $d \geqslant 23$ the ratio is below $10^{-9}$.

The bad performance of acceptance-rejection sampling in Example 5.2 for large $d$ is a well-known consequence of high-dimensional geometry. Square and circular regions have a mismatch that grows strongly with dimension.

The high-dimensional setting magnifies mismatches between distributions in more ways than Example 5.2 shows. We can see the effects very clearly in some simple examples. The examples that show the effect most clearly are ones where we would not have to use acceptance-rejection sampling.

As a first simple example, when $f$ and $g$ are both products of $d$ univariate densities then we have $c = \sup_{\boldsymbol{x}} f(\boldsymbol{x})/g(\boldsymbol{x}) = \prod_{j=1}^d c_j$ where $c_j = \sup_x f_j(x)/g_j(x) \geqslant 1$. If each $c_j > 1 + \epsilon$ then $c$ grows exponentially with $d$.

As a second simple example, we can expect to have at least some mismatch between the covariance in $f$ and that in $g$. Exercise 5.3 shows that, even in the Gaussian case such mismatches grow rapidly more severe as $d$ increases. For general distributions it could be worse.

Using equation (5.1) and an equivalent form for $g$, we find that

$$\frac{f(\boldsymbol{x})}{g(\boldsymbol{x})} = \prod_{j=1}^{d} \frac{f_{j\mid 1:(j-1)}(x_j \mid \boldsymbol{x}_{1:(j-1)})}{g_{j\mid 1:(j-1)}(x_j \mid \boldsymbol{x}_{1:(j-1)})}.$$

If we were to bound each factor in this ratio, then we might anticipate a constant $c$ growing exponentially with $d$ unless we were so lucky that the factor ratios approached 1 quickly.

Inversion becomes quite complicated to implement in high dimensions. Acceptance-rejection sampling can get a little harder to do because finding $c = \sup_{\boldsymbol{x}} f(\boldsymbol{x})/g(\boldsymbol{x})$ now requires $d$-dimensional optimization. But the more critical problem is that we have trouble keeping $c$ small enough to make the algorithm efficient.

We can make acceptance-rejection work in high dimensions if the connection between $f$ and $g$ is inherently low-dimensional. For example in §5.7 we look at radially symmetric distributions. Using acceptance-rejection for the radius and a uniform distribution on the unit sphere has a constant $c$ that comes from the one-dimensional part generating the radius.

The third generic method for non-uniform random variables is mixture sampling. Let

$$F(\boldsymbol{x}) = \sum_{k=1}^{K} \pi_k F_k(\boldsymbol{x})$$

where $\pi_k \geqslant 0$ and $\sum_{k=1}^{K} \pi_k = 1$ and $F_k$ are distributions on $\mathbb{R}^d$ from which we can sample. As there, we first sample a discrete random variable $Z$ with $\mathbb{P}(Z = k) = \pi_k$, and then conditionally on $Z = k$ we sample $\boldsymbol{X} \sim F_k$. Continuous mixing distributions $F(\boldsymbol{x}) = \int F(\boldsymbol{x} \mid \boldsymbol{y}) g(\boldsymbol{y}) \,\mathrm{d}\boldsymbol{y}$ are also useful. We sample $\boldsymbol{Y} \sim g$ and then $\boldsymbol{X} \sim F(\cdot \mid \boldsymbol{Y})$.

In one dimension, some automatic sampling methods decompose an almost arbitrary density into pieces suitable for mixture sampling. Those methods generalize in principal, but become unworkable in high dimensions.

The rectangle-wedge-tail method for $d$ dimensions generalizes, but the intervals we used for the domain of $\boldsymbol{X}$ become rectangular regions. The number of rectangular regions needed to cover $\mathbb{R}^d$ properly grows rapidly with $d$ as does the cost of pre-computing $\pi_k$ for each of those regions. There are many more kinds of tail regions, which are unbounded in some but not all of the $d$ dimensions. Finally the wedge regions are more complicated than for $d = 1$. The result is too cumbersome.

The ziggurat method uses horizontal regions that for $d$-dimensional sampling take a form like $\{\boldsymbol{x} \in \mathbb{R}^d \mid a \leqslant f(\boldsymbol{x}) < b\} \times [a, b]$. That is, we have to sample the region between two contours of $f$. In one dimension such a region is just an interval but in $d$ dimensions it is more complicated.

Similarly, the adaptive rejection sampling can in principle extend to multidimensional $\boldsymbol{X}$. But if we bound $\log f(\boldsymbol{x})$ by a piecewise planar function we then have to sample piecewise exponential densities defined not over intervals, but over polygonal regions of $\mathbb{R}^d$.

Thus while mixture sampling is applicable to special problems in the multivariate setting it is not easily adapted to automated generation.

## 5.2 Multivariate normal and $t$

The multivariate normal distribution on $\mathbb{R}^d$ is one of the most important multivariate distributions and, fortunately, one of the easiest to sample from.

The ***multivariate normal distribution*** is defined by a mean vector $\mu \in \mathbb{R}^d$ and a positive semi-definite variance-covariance matrix $\Sigma \in \mathbb{R}^{d\times d}$. The distribution is written $\mathcal{N}(\mu, \Sigma)$. When $\boldsymbol{X} \sim \mathcal{N}(\mu, \Sigma)$ and $\Sigma$ is invertible, then $\boldsymbol{X}$ has the density function

$$\varphi(\boldsymbol{x}; \mu, \Sigma) = \frac{e^{-\frac{1}{2}(\boldsymbol{x}-\mu)'\Sigma^{-1}(\boldsymbol{x}-\mu)}}{(2\pi)^{d/2}|\Sigma|^{1/2}}, \quad \boldsymbol{x} \in \mathbb{R}^d.$$

If $\Sigma$ is not invertible, then $\boldsymbol{X}$ has a singular distribution which does not have a density function. Instead, for some $k \in \{1, \dots, d\}$ we have $\mathbb{P}(X_k = \alpha_0 + \sum_{j=1, j\neq k}^{d} \alpha_j X_j) = 1$ and in this sense component $X_k$ is redundant. Sometimes we remove redundant components and other times we retain them in order to work with simpler formulas.

The multivariate normal distribution has many convenient properties that we will use:

**MVN-1** If $\boldsymbol{X} \sim \mathcal{N}(\mu, \Sigma)$, then $AX + b \sim \mathcal{N}(A\mu + b, A\Sigma A^{\mathsf{T}})$.

**MVN-2** Split $\boldsymbol{X}$ into two disjoint subvectors $\boldsymbol{X}_1 = (X_1, X_2, \dots, X_r)^{\mathsf{T}}$ and $\boldsymbol{X}_2 = (X_{r+1}, \dots, X_d)^{\mathsf{T}}$, and similarly partition the parameters

$$\mu = \begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix} \quad \text{and} \quad \Sigma = \begin{pmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{pmatrix}.$$

Then $\boldsymbol{X}_j \sim \mathcal{N}(\mu_j, \Sigma_j)$ for $j = 1, 2$.

**MVN-3** $\boldsymbol{X}_1$ and $\boldsymbol{X}_2$ above are independent if and only if $\Sigma_{12} = 0$.

**MVN-4** If $\Sigma_{22}$ is not singular then the conditional distribution of $\boldsymbol{X}_1$ given that $\boldsymbol{X}_2 = \boldsymbol{x}_2$ is $\mathcal{N}(\mu_1 + \Sigma_{12}\Sigma_{22}^{-1}(\boldsymbol{x}_2 - \mu_2), \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21})$.

It is easy to sample $\boldsymbol{Z} \sim \mathcal{N}(0, I_d)$. In that case the components of $\boldsymbol{Z}$ are uncorrelated, which for the normal distribution means that they are independent. We take independent components $Z_j \sim \mathcal{N}(0,1)$ for $j = 1, \dots, d$. Either inversion or Box-Muller would be suitable. To sample $\boldsymbol{X} \sim \mathcal{N}(\mu, \Sigma)$, we find a matrix $C$ such that $CC^{\mathsf{T}} = \Sigma$. Then we set

$$\boldsymbol{X} = \mu + C\boldsymbol{Z}, \quad \text{where} \quad \boldsymbol{Z} \sim \mathcal{N}(0, I_d). \tag{5.3}$$

Equation (5.3) includes the singular normal distributions, which arise when the matrix $C$ has rank smaller than $d$.

From (5.3), we see that the task in sampling the multivariate normal distribution reduces to finding an appropriate matrix $C$. The choice of $C$ is not unique. If $C$ works and $\widetilde{C} = CQ$, for an orthogonal matrix $Q$, then $\widetilde{C}\widetilde{C}^{\mathsf{T}} = CQQ^{\mathsf{T}}C^{\mathsf{T}} = CC^{\mathsf{T}}$, and so $\widetilde{C}$ works too. There are numerical methods for finding $C$ given $\Sigma$. In special cases we can find closed-form solutions.

A covariance matrix $\Sigma$ is a symmetric positive semi-definite matrix, and so it has a ***spectral decomposition***

$$\Sigma = P\Lambda P^{\mathsf{T}} \tag{5.4}$$

where $\Lambda$ is a diagonal matrix in which $\Lambda_{ii} \geqslant 0$ is the $i$'th eigenvalue of $\Sigma$ and $P$ is the matrix whose $i$'th column is the $i$'th eigenvector of $\Sigma$. Then we may take

$$C = P\Lambda^{1/2}P^{\mathsf{T}}$$

where $\Lambda^{1/2} = \mathrm{diag}(\Lambda_{ii}^{1/2})$. Code for the eigendecomposition (5.4) is widely available.

When $\Sigma$ is singular, the matrix $C$ can be chosen to have dimension $d \times r$ where $r$ is the rank of $\Sigma$. We can implement this choice by simply ignoring the eigenvalue-eigenvector pairs of $\Sigma$ for which $\Lambda_{ii} = 0$. For singular, or nearly numerically singular $\Sigma$, it can be advantageous to use the singular value decomposition (SVD), $\Sigma = U\mathrm{diag}(\sigma_1, \dots, \sigma_d)V^\mathsf{T}$ to find a suitable matrix $C$. Here $U = V = P$ and $\mathrm{diag}(\sigma_1, \dots, \sigma_d) = \Lambda$, so the decomposition is the same as the spectral one. The SVD is useful here as an alternative computation for numerically challenging covariance matrices. In particular, it enforces $\sigma_i \geqslant 0$. We take $C = U\mathrm{diag}(\sqrt{\sigma_1}, \dots, \sqrt{\sigma_d})U^\mathsf{T}$.

When $\Sigma$ is positive definite, then it is possible to choose $C = L$ where $L$ is a lower triangular matrix satisfying $LL^\mathsf{T} = \Sigma$. This is the ***Cholesky decomposition***. The Cholesky decomposition is easy to program but can run into numerical trouble when $\Sigma$ is singular or nearly so. Fortunately, the resulting $LL^\mathsf{T}$ is reliable even when $L$ is not (Trefethen and Bau, 1997, Lecture 23). The eigendecomposition is often preferred, but for some special matrices we can do the Cholesky decomposition in closed form. The covariance of Brownian motion in §6.4 is one example.

The use of a lower triangular matrix $C$ in (5.3) generates components $X_j$ in increasing order of $j$. It implements the sequential inversion algorithm (5.2), if $Z_i = \Phi^{-1}(U_i)$ for $\boldsymbol{U} \sim \mathbf{U}(0,1)^d$. Some implementations of the Cholesky decomposition return an upper triangular matrix $R = L^\mathsf{T} = C^\mathsf{T}$, so that $R^\mathsf{T}R = \Sigma$. In such cases we take $C = R^\mathsf{T}$, and failing to transpose $R$ would be an error that could be hard to detect from simulation output. It is a good practice to compute $\Sigma - CC^\mathsf{T}$ before using a matrix $C$ in a simulation.

For some specially structured covariances $\Sigma$ we don't have to use numerical methods to find $C$. Closed-form special cases are especially valuable when $d$ is extremely large because the work in finding the spectral or Cholesky matrix $C$ takes $O(d^3)$ time.

**Example 5.3** (Symmetric nonnegative correlation)**.** Suppose for $d \geqslant 2$ that $\boldsymbol{X} \sim \mathcal{N}(0, \Sigma)$ where $\Sigma_{jk} = 1$ for $j = k$ and $\Sigma_{jk} = \rho$ for $j \neq k$ with $0 \leqslant \rho \leqslant 1$. Since $\rho \geqslant 0$, there is a representation with $X_j = \sqrt{\rho}Z_0 + \sqrt{1-\rho^2}Z_j$ and $Z_0, \dots, Z_d$ independent $\mathcal{N}(0,1)$. We can directly verify that $\mathrm{Cov}(\boldsymbol{X}) = \Sigma$. Implicitly we are using the matrix

$$C = \begin{pmatrix} \sqrt{\rho}\,\mathbf{1}_d & \sqrt{1-\rho^2}\, I_d \end{pmatrix} \in \mathbb{R}^{(d+1)\times d}$$

with $\boldsymbol{Z} \sim \mathcal{N}(0, I_{d+1})$.

When we want $\rho < 0$ then we cannot use the method in Example 5.3. Values of $\rho$ below $-1/(d-1)$ are impossible because then $\Sigma$ is not positive semi-definite. But all values $\rho \geqslant -1/(d-1)$ can be obtained by the following method.

**Example 5.4** (Symmetric correlation)**.** Suppose for $d \geqslant 2$ that $\boldsymbol{X} \sim \mathcal{N}(0, \Sigma)$ with $\Sigma_{jj} = 1$ and $\Sigma_{jk} = \rho$ for $j \neq k$ where $-1/(d-1) \leqslant \rho \leqslant 1$. Now take $X_j = aZ_j + b\bar{Z}$ where $\bar{Z} = \frac{1}{d}\sum_{j=1}^d Z_j$ and $\boldsymbol{Z} \sim \mathcal{N}(0, I_d)$. Having $b < 0$ will induce negative correlation. Solving $\mathbb{E}(X_1^2) = 1$ and $\mathbb{E}(X_1X_2) = \rho$ together leads to quadratic equations whose solutions include $a = \sqrt{1-\rho}$ with $b = -(\sqrt{1-\rho} + \sqrt{1+\rho(d-1)}\,)$. Therefore

$$X_j = \sqrt{1-\rho}\,Z_j - \bigl(\sqrt{1-\rho} + \sqrt{1+\rho(d-1)}\,\bigr)\bar{Z}$$

works. That is, $C = \sqrt{1-\rho}\,I_d - \bigl(\sqrt{1-\rho} + \sqrt{1+\rho(d-1)}\,\bigr)\mathbf{1}_d\mathbf{1}_d^{\mathsf{T}}/d$.

## Conditional sampling

One of the great conveniences of the multivariate normal distribution is that the conditional distribution of $\boldsymbol{X}_u$ given $\boldsymbol{X}_v$ remains in the multivariate normal family where $u$ and $v$ are subsets of $\{1,\dots,d\}$. Let $\mu_u$ and $\mu_v$ be the corresponding subvectors of $\mu$ and define $\Sigma_{uv}$ to be the submatrix of $\Sigma$ whose rows are in $u$ and columns are in $v$.

We assume that $\Sigma_{vv}$ is invertible. This is reasonable because to condition on $\boldsymbol{X}_v$ we just have to condition on its non-redundant components. Then the conditional distribution of $\boldsymbol{X}_u$ given that $\boldsymbol{X}_v = \boldsymbol{x}_v$ is

$$\mathcal{N}\bigl(\,\mu_u + \Sigma_{uv}\Sigma_{vv}^{-1}(\boldsymbol{x}_v - \mu_v),\,\Sigma_{uu} - \Sigma_{uv}\Sigma_{vv}^{-1}\Sigma_{vu}\bigr). \tag{5.5}$$

To use equation (5.5) as it stands requires finding a matrix square root $C_{u|v}$ for which

$$C_{u|v}C_{u|v}^{\mathsf{T}} = \Sigma_{u|v} \equiv \Sigma_{uu} - \Sigma_{uv}\Sigma_{vv}^{-1}\Sigma_{vu}$$

holds. The work to find $C_{u|v}$ scales as $O(d_u^3)$ where $d_u$ is the cardinality of $u$, unless there is some special structure to exploit. If we have a square root $CC^{\mathsf{T}} = \Sigma$, then we can use $C$ in a shortcut sampler for $\boldsymbol{X}_u$ given $\boldsymbol{X}_v = \boldsymbol{x}_v$ as follows:

$$\begin{aligned}\boldsymbol{X} &\leftarrow \mu + C\boldsymbol{Z}, \quad \boldsymbol{Z} \sim \mathcal{N}(0,I)\\ \boldsymbol{X}_u &\leftarrow \boldsymbol{X}_u + \Sigma_{uv}\Sigma_{vv}^{-1}(\boldsymbol{x}_v - \boldsymbol{X}_v)\end{aligned} \tag{5.6}$$

In equation (5.6) we still need to be able to invert $\Sigma_{vv}$. That is inexpensive when $d_v = d - d_u$ is small.

Conditional sampling of the multivariate normal distribution lets us generate the components of $\boldsymbol{X}$ in any order we like. We can generate them one at a time or in larger groups. At each step after the first, $u$ has the new indices we're generating and $v$ includes all the previously generated ones.

## Conditioning on linear combinations

The multivariate normal distribution even allows us to generate a linear combination of $X_j$ such as $T = X_1 + \cdots + X_d$ and then later sample the entries $X_j$ or even some other linear combinations of $X$ conditionally on the ones previously generated. This can be a big advantage if we want to compute $\mathbb{E}(f(\boldsymbol{X}))$ and $f$ is such that $f(\boldsymbol{X}) = 0$ for certain values of $T$. Generating $T$ first would sometimes save us most of the cost of generating $\boldsymbol{X}$ and all of the cost of computing $f(\cdot)$.

In other settings we might benefit from sampling $T$ by advanced methods from later chapters, such as stratification or randomized quasi-Monte Carlo. Once we have $T$, we fill in the rest of $X$ by sampling conditionally on $T$.

Suppose that we have generated $r$ linear combinations of $\boldsymbol{X}$. Write them as $\boldsymbol{T} = \Theta\boldsymbol{X} \in \mathbb{R}^r$ for $\Theta \in \mathbb{R}^{r\times d}$ of rank $r < d$. Now we want to sample from the conditional distribution of $\boldsymbol{X}$ given $\Theta\boldsymbol{X}$. The variance of $\Theta\boldsymbol{X}$ is $\Theta\Sigma\Theta^{\mathsf{T}}$ which we assume has full rank $r$. Then

$$\begin{pmatrix}\boldsymbol{X}\\ \boldsymbol{T}\end{pmatrix} = \begin{pmatrix}\boldsymbol{X}\\ \Theta\boldsymbol{X}\end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix}\mu\\ \Theta\mu\end{pmatrix}, \begin{pmatrix}\Sigma & \Sigma\Theta^{\mathsf{T}}\\ \Theta\Sigma & \Theta\Sigma\Theta^{\mathsf{T}}\end{pmatrix}\right).$$

Therefore the conditional distribution of $\boldsymbol{X}$ given that $\boldsymbol{T} = \boldsymbol{t}$ is

$$\mathcal{N}\big(\,\mu + \Sigma\Theta^{\mathsf{T}}(\boldsymbol{t} - \Theta\mu),\, \Sigma - \Sigma\Theta^{\mathsf{T}}(\Theta\Sigma\Theta^{\mathsf{T}})^{-1}\Theta\Sigma\,\big). \tag{5.7}$$

Though (5.7) is a cumbersome expression, it still represents a great simplification. Most joint distributions do not lead to closed forms for conditioning on arbitrary linear combinations of the variables. With the multivariate normal distribution and arbitrary problem-specific $\Theta$ and $\Sigma$, we can apply a sequence of numerical matrix operations to get the conditional samples we need.

Finding a square root for the covariance matrix in (5.7) could take $O(d^3)$ work and this might be expensive. It might be much more expensive than taking a square root of $\Sigma$ if the problem is one where $\Sigma$ has a special simplifying structure (e.g., diagonality). In such cases conditional sampling might benefit from the shortcut in (5.6).

First we compute $\boldsymbol{T}$ somehow, perhaps by using a variance reduction method to sample from $\mathcal{N}(\Theta\mu, \Theta\Sigma\Theta^{\mathsf{T}})$. Then given $\boldsymbol{T} = \boldsymbol{t}$, we sample

$$\begin{aligned}&\boldsymbol{X} \sim \mathcal{N}(\mu, \Sigma), \quad \text{and set,}\\ &\boldsymbol{X} \leftarrow \boldsymbol{X} + \Sigma\Theta^{\mathsf{T}}(\Theta\Sigma\Theta^{\mathsf{T}})^{-1}(\boldsymbol{t} - \Theta\boldsymbol{X}).\end{aligned} \tag{5.8}$$

Notice that we generate $r + d$ random variables for $\boldsymbol{T}$ and $\boldsymbol{X}$, in order to deliver the $d$ components of $\boldsymbol{X}$. We have to solve a set of linear equations to get $(\Theta\Sigma\Theta^{\mathsf{T}})^{-1}(\boldsymbol{t} - \Theta\boldsymbol{X})$ but it is only $r \times r$ so the cost is $O(r^3)$ which may be a great savings when $r \ll d$. We may need to find a square root of $\Sigma$, but if so the same root works for any matrix $\Theta$.

## Multivariate $t$

The multivariate normal is closely related to the multivariate $t$ distribution. The ***multivariate*** $t$ ***distribution*** with center $\mu$, scale matrix $\Sigma$ and $\nu$ degrees of freedom, denoted $\mathrm{t}(\mu, \Sigma, \nu)$, has a representation

$$\boldsymbol{X} = \mu + \frac{\Sigma^{1/2}\boldsymbol{Z}}{\sqrt{W/\nu}} \tag{5.9}$$

where $Z \sim \mathcal{N}(0, I)$ in $d$ dimensions, independently of $W \sim \chi^2_{(\nu)}$. Here $\Sigma^{1/2}$ denotes any $d$ by $d$ matrix $C$ with $CC^{\mathsf{T}} = \Sigma$. The distribution can be sampled directly from its defining representation (5.9). In the limit $\nu \to \infty$, the distribution of $\boldsymbol{X}$ converges to $\mathcal{N}(\mu, \Sigma)$. The case with $\nu = 1$ is sometimes called the ***multivariate Cauchy*** distribution.

The $\mathrm{t}(\mu, \Sigma, \nu)$ density is

$$f(\boldsymbol{x}; \mu, \Sigma, \nu) = C_{\mu,\Sigma,\nu}\bigl(1 + (\boldsymbol{x} - \mu)^{\mathsf{T}}\Sigma^{-1}(\boldsymbol{x} - \mu)\bigr)^{-(\nu+d)/2} \tag{5.10}$$

where

$$C_{\mu,\Sigma,\nu} = \frac{1}{|\Sigma|^{1/2}}\frac{1}{(\nu\pi)^{d/2}}\frac{\Gamma((\nu+d)/2)}{\Gamma(\nu/2)}.$$

The standard multivariate $t$ density, with $\mu = 0$ and $\Sigma = I$, is proportional to $(1 + \|\boldsymbol{x}\|^2)^{-(\nu+d)/2}$.

The density (5.10) has ellipsoidal contours centered around $\mu$. The tails of the $t$ distribution are heavier than those of the multivariate normal distribution. This makes it suitable for some applications in which the normal distribution is thought to be useful apart from light tails. Examples include importance sampling (§9.1) and generating candidates for acceptance-rejection sampling (§4.7).

The marginal distribution of $(X_j - \mu_j)/\sqrt{\Sigma_{jj}}$ is $t_{(\nu)}$. Notice that when $\Sigma = I_d$ the components of $\boldsymbol{X}$ are not independent. For $\Sigma = I_d$, $\boldsymbol{X} - \mu$ has a radially symmetric distribution, around the origin. The only radially symmetric distributions around the origin with independent components are of the form $\mathcal{N}(0, \sigma^2 I_d)$. See Exercise 5.4 for the variance and correlations of $\boldsymbol{X}$.

## 5.3 Multinomial

The vector $\boldsymbol{X}$ has the multinomial distribution $\mathrm{Mult}(m, p_1, \dots, p_d)$ if

$$\mathbb{P}(X_1 = x_1, \dots, X_d = x_d) = \frac{m!}{x_1!x_2!\cdots x_d!}\prod_{j=1}^{d} p_j^{x_j}$$

for integers $x_j \geqslant 0$ satisfying $\sum_{j=1}^{d} x_j = m$. The parameters are an integer $m \geqslant 0$ and probabilities $p_j \geqslant 0$ that sum to 1. The multinomial distribution can be interpreted as giving the number $X_j$ of balls landing in each of $d$ cells, when $m$ balls are dropped independently and each one has probability $p_j$ of landing in cell $j$.

Sometimes we write $\boldsymbol{X} \sim \mathrm{Mult}(m, \boldsymbol{p})$ where $\boldsymbol{p} = (p_1, \dots, p_d)$. We include the degenerate case $m = 0$ with $\mathbb{P}(\boldsymbol{X} = (0, \dots, 0)) = 1$. The parameter $\boldsymbol{p}$ belongs to the set

$$\Delta^{d-1} = \Bigl\{(p_1, \dots, p_d) \mid p_j \geqslant 0, \sum_{j=1}^{d} p_j = 1\Bigr\} \tag{5.11}$$

which is known as the ***unit simplex*** in $d$ dimensions. The index on $\Delta$ is $d - 1$ because the $d$ components lie in a $d - 1$-dimensional subspace.

**Algorithm 5.1** Sample $\boldsymbol{X} \sim \mathrm{Mult}(m, p_1, \dots, p_d)$

```
given $m \in \mathbb{N}_0$, $d \in \mathbb{N}$ and $\boldsymbol{p} = (p_1, \dots, p_d) \in \Delta^{d-1}$
$\ell \leftarrow m$, $S \leftarrow 1$
for $j = 1$ to $d$ do
    $X_j \sim \mathrm{Bin}(\ell, p_j/S)$
    $\ell \leftarrow \ell - X_j$
    $S \leftarrow S - p_j$
deliver $\boldsymbol{X}$
```

If rounding errors yield $S < 0$ then $S$ should be replaced by 0. Any $p_j/S > 1$ should be replaced by 1. If $\ell = 0$, the remaining components of $\boldsymbol{X}$ can be set to 0.

The $\mathrm{Mult}(1, p_1, \dots, p_d)$ distribution, in which just one ball is dropped, may be sampled by selecting $\boldsymbol{Y} = (0, \dots, 0, 1, 0, \dots, 0)$ with probability $p_j$ where the single 1 is in the $j$'th position. Then a $\mathrm{Mult}(m, p_1, \dots, p_d)$ random variable $\boldsymbol{X}$ can be generated as $\boldsymbol{X} = \sum_{k=1}^{m} \boldsymbol{Y}_k$ where $\boldsymbol{Y}_k$ are IID $\mathrm{Mult}(1, p_1, \dots, p_d)$ random variables. That is, we sample via

```
$\boldsymbol{X} \leftarrow (0, \dots, 0)$            // length $d$
for $k = 1$ to $m$ do
    $J \sim p$                       // i.e., $\mathbb{P}(J = j) = p_j$
    $X_j \leftarrow X_j + 1$
```

with $\boldsymbol{Y}_k$ implicitly given by the categorical random variable $J$. When we have sufficient storage available and $p_j = n_j/N$ for integers $n_j \geqslant 0$ summing to $N$ then the array sampling method in §4.6 is a convenient way to sample $J$.

For large $m$ it is more efficient to generate the components of $\boldsymbol{X}$ in order. The conditional distribution of $X_j$ given $X_1, \dots, X_{j-1}$ is a binomial. Conditional sampling of the multinomial distribution is described in Algorithm 5.1.

As with the normal distribution, the components $X_j$ can be generated in any order, not necessarily in the order of their indices. For example, it might be faster to generate $X_j$ before $X_k$ when $p_j > p_k$. Exercise 5.5 makes a comparison.

It is not convenient to work with arbitrary linear combinations of multinomial components, as we did for the multivariate normal distribution. But we can easily work with sums of subsets of multinomial components. Generating sums over subsets allows us to develop a recursive sampling scheme for the multinomial distribution. The recursive scheme may be faster. It can also be used in some variance reduction techniques such as stratified sampling.

Let $Y = \sum_{j \in u} X_j$ where $\boldsymbol{X} \sim \mathrm{Mult}(m, p_1, \dots, p_d)$ and $u \subset \{1, \dots, d\}$. Then $Y \sim \mathrm{Bin}(m, \sum_{j \in u} p_j)$. Conditionally on $Y = y$, the vectors $(X_j)_{j \in u}$ and $(X_j)_{j \notin u}$ are independent multinomials. Their count parameters are $y$ and $m - y$ respectively. Their probability vectors are renormalized versions of $(p_j)_{j \in u}$ and

**Algorithm 5.2** Recursively sample $\boldsymbol{X} \sim \mathrm{Mult}(m, p_1, \dots, p_d)$

```
given $m \in \mathbb{N}_0$, $d \in \mathbb{N}$, $r \in \{1, \dots, d\}$ and $\boldsymbol{p} = (p_1, \dots, p_d) \in \Delta^{d-1}$
if $d = 1$ then
    deliver $\boldsymbol{X} = (m)$
$q \leftarrow p_1 + \cdots + p_r$
$Y \sim \mathrm{Bin}(m, q)$
$(X_1, \dots, X_r) \sim \mathrm{Mult}(Y, p_1/q, \dots, p_r/q)$
$(X_{r+1}, \dots, X_d) \sim \mathrm{Mult}(m - Y, p_{r+1}/(1-q), \dots, p_d/(1-q))$
deliver $\boldsymbol{X} = (X_1, \dots, X_d)$
```

This algorithm generates a multinomial random vector. The two multinomial samples can be generated by Algorithm 5.1. If this algorithm is to be used recursively, then we have to select $r$ at each level, and possibly reorder the components. If $m = 0$, then we can deliver $\boldsymbol{X} = (0, \dots, 0)$.

$(p_j)_{j \notin u}$ respectively. If $u = \{1, \dots, r\}$, then

$$\begin{aligned} Y = \sum_{j=1}^{r} X_j &\sim \mathrm{Bin}(m, p_1 + \cdots + p_r) \\ (X_1, \dots, X_r) \mid Y = y &\sim \mathrm{Mult}(y, p_1/p_{1:r}, \dots, p_r/p_{1:r}) \\ (X_{r+1}, \dots, X_d) \mid Y = y &\sim \mathrm{Mult}(m - y, p_{r+1}/p_{(r+1):d}, \dots, p_d/p_{(r+1):d}), \end{aligned}$$

where $p_{1:r} = \sum_{j=1}^{r} p_j$ and $p_{(r+1):d} = \sum_{j=r+1}^{d} p_j$.

Algorithm 5.2 uses this decomposition to recursively sample the multinomial distribution. It provides a divide-and-conquer approach to generating $\boldsymbol{X}$.

## 5.4 Dirichlet

Many random vectors are internally normalized and so belong to the unit simplex

$$\Delta^{d-1} = \Bigl\{ (x_1, \dots, x_d) \mid x_j \geqslant 0, \sum_{j=1}^{d} x_j = 1 \Bigr\}.$$

Compositional data, such as the proportions by weight of $d$ mineral types in a rock sample are of this type. Also, any point $\boldsymbol{x} \in \Delta^{d-1}$ is a probability on the set $\{1, \dots, d\}$. Indeed this simplex was part of the parameter space for the multinomial distribution. Applications that require us to generate random probability distributions with finite support, involve sampling in $\Delta^{d-1}$.

The Dirichlet distribution with parameter $\alpha \in (0, \infty)^d$, denoted $\mathrm{Dir}(\alpha)$, has probability density function proportional to

$$D(\alpha)^{-1} \prod_{j=1}^{d} x_j^{\alpha_j - 1}, \quad \boldsymbol{x} \in \Delta^{d-1}, \tag{5.12}$$

where the normalizing constant is $D(\alpha) = \prod_{j=1}^d \Gamma(\alpha_j)/\Gamma(\sum_{j=1}^d \alpha_j)$. Because $\Delta^{d-1}$ is a $d-1$-dimensional subset of $\mathbb{R}^d$, this density is nonstandard. A standard probability density function may be specified for any $d-1$ components. For example, the density of the first $d-1$ components is

$$D(\alpha)^{-1}\prod_{j=1}^{d-1} x_j^{\alpha_j-1}\Bigl(1-\sum_{j=1}^{d-1} x_j\Bigr)^{\alpha_d-1}, \tag{5.13}$$

for

$$\boldsymbol{x}_{1:(d-1)} \in \Bigl\{(x_1,\dots,x_{d-1}) \mid x_j \geqslant 0, \sum_{j=1}^{d-1} x_j \leqslant 1\Bigr\}.$$

The symmetry in equation (5.12) often makes it easier to work with than the proper density (5.13).

There are two noteworthy special cases of $\mathrm{Dir}(\alpha)$. First, when every $\alpha_j = 1$, then $\boldsymbol{X} \sim \mathbf{U}(\Delta^{d-1})$. Second, when $d = 2$, then equation (5.13) yields $X_1 \sim \mathrm{Beta}(\alpha_1, \alpha_2)$ and $X_2 = 1 - X_1 \sim \mathrm{Beta}(\alpha_2, \alpha_1)$.

It is especially simple to sample $\boldsymbol{X} \sim \mathrm{Dir}(\alpha)$. The method is as follows:

$$\begin{aligned} &Y_j \sim \mathrm{Gam}(\alpha_j), \quad j = 1,\dots,d, \quad \text{then}\\ &X_j = \frac{Y_j}{\sum_{k=1}^d Y_k}, \quad j = 1,\dots,d. \end{aligned} \tag{5.14}$$

The method (5.14) is mildly redundant, using $d$ random variables to sample in a $d-1$-dimensional space. But the simplicity of (5.14) makes it very attractive.

Using (5.14), we can show that each component $X_j \sim \mathrm{Beta}(\alpha_j, \sum_{k\neq j}\alpha_k)$. Let $Y_{-j} \equiv \sum_{k\neq j} Y_k \sim \mathrm{Gam}(\sum_{k\neq j}\alpha_k)$. Then $X_j = Y_j/(Y_j + Y_{-j})$, which matches our representation of the beta distribution in terms of independent Gamma random variables. As a result, the Dirichlet distribution may be considered a multivariate beta distribution.

When there are three components, then $\boldsymbol{x}$ is inside the equilateral triangle with corners $e_1 = (1,0,0)$, $e_2 = (0,1,0)$ and $e_3 = (0,0,1)$. We can plot that triangle in the plane. Figure 5.1 shows the triangle $\Delta^2$ along with samples from 6 different Dirichlet distributions. The triangles in the figure are annotated with $\alpha_j$ appearing at corner $e_j$. It is visually clear that the points tend to lie near the corners $e_j$ with the largest $\alpha_j$ and this is natural given the sampling equation (5.14). On closer inspection we see that a large value of $\alpha_j$ acts to push points away from the side opposite corner $j$, rather than attract them to corner $j$. These sound equivalent, but to see the distinction, compare $\mathrm{Dir}((7,7,7))$ with $\mathrm{Dir}((1,1,1))$ and $\mathrm{Dir}((0.2,0.2,0.2))$ in Figure 5.1.

The case with all $\alpha_j = 1$ is special because it leads to $\boldsymbol{X} \sim \mathbf{U}(\Delta^{d-1})$. This distribution can be sampled by the ***uniform spacings*** method. Let $U_1,\dots,U_{d-1}$ be independent $\mathbf{U}(0,1)$ random variables and let their order statistics be $U_{(1)} \leqslant U_{(2)} \leqslant \cdots \leqslant U_{(d-1)}$. We extend them with $U_{(0)} = 0$ and $U_{(d)} = 1$.

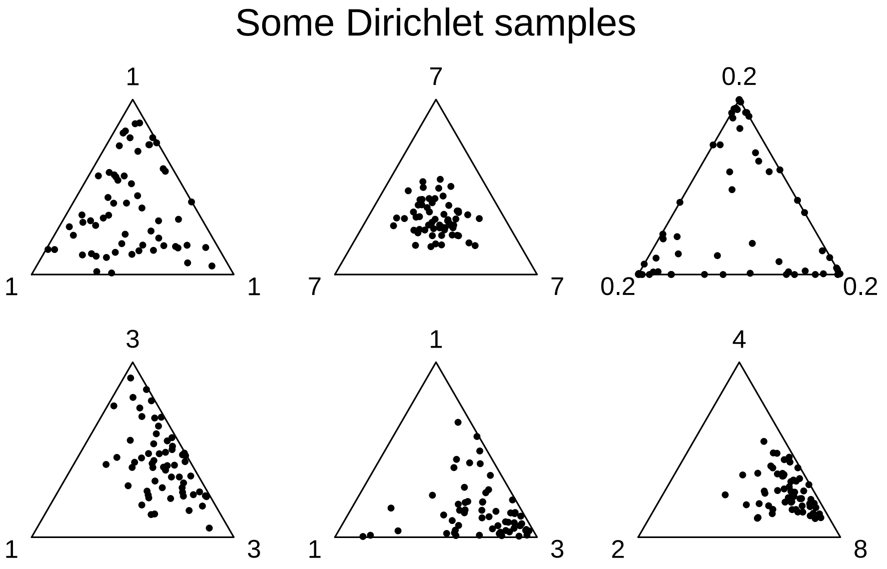

Figure 5.1: This figure shows samples of size 60 from the Dir($\alpha$) distribution for 6 different vectors $\alpha \in (0,\infty)^3$. The corner in each sample is labeled with its corresponding component $\alpha_j$, $j = 1, 2, 3$. Small values of $\alpha$ lead to points near the boundary. The points tend to land closest to the corners with the large $\alpha$ values.

Now if we define $\boldsymbol{X}$ by

$$X_j = U_{(j)} - U_{(j-1)}, \quad j = 1, \dots, d \tag{5.15}$$

we get $\boldsymbol{X} \sim \mathbf{U}(\Delta^{d-1})$. This computation avoids using the logarithm like we would have to if we normalized $\log(U_j)$. It also uses just $d-1$ random variables to sample within $\Delta^{d-1}$ which is an advantage for some variance reduction methods when $d$ is small. Sorting takes $O(d\log(d))$ work, so for large enough $d$, uniform spacings will be slower than normalizing exponential random variables.

**Example 5.5** (Uniform distribution on a simplex)**.** For $d \geqslant 1$, let $\boldsymbol{z}_1, \dots, \boldsymbol{z}_{d+1} \in \mathbb{R}^d$ be in general position, meaning that they do not lie within a $d-1$-dimensional hyperplane. These points form the vertices of a $d$-dimensional simplex

$$\left\{ \sum_{j=1}^{d+1} u_j \boldsymbol{z}_j \mid 0 \leqslant u_j \leqslant 1,\ \sum_{j=1}^{d+1} u_j = 1 \right\}$$

(for $d = 2$, a triangle). To sample uniformly within this simplex we take

$$\boldsymbol{X} = \sum_{j=1}^{d+1} X_j \boldsymbol{z}_j, \quad \text{where} \quad \boldsymbol{X} \sim \mathbf{U}(\Delta^d) = \mathrm{Dir}((1, 1, \dots, 1)).$$

The Dirichlet distribution is not very flexible. It has only $d$ parameters. The mean $\mathbb{E}(\boldsymbol{X}) = \alpha/\sum_{j=1}^d \alpha_j$ uses up $d-1$ parameters, leaving just the normalizing constant $\sum_{j=1}^d \alpha_j$ to describe how close $\boldsymbol{X}$ is to $\mathbb{E}(\boldsymbol{X})$. There are not enough parameters to represent $d(d-1)/2$ correlations among components of $\boldsymbol{X}$ or even $d$ different variances. The components of $\boldsymbol{X}$ are nearly independent, with a mild negative correlation stemming from their normalization. If we need to model positive correlation within $\Delta^{d-1}$, or a strong negative correlation, then we cannot do so with a Dirichlet distribution.

The logistic-normal distribution on $\Delta^{d-1}$ incorporates a correlation matrix, making it more flexible than the Dirichlet. Under the ***logistic-normal distribution***

$$(\log(X_1/X_d), \dots, \log(X_{d-1}/X_d)) \sim \mathcal{N}(\mu, \Sigma)$$

for $\mu \in \mathbb{R}^{d-1}$ and a positive definite $\Sigma \in \mathbb{R}^{(d-1)\times(d-1)}$. To generate $\boldsymbol{X}$ from this distribution we sample $\boldsymbol{Y} \sim \mathcal{N}(\mu, \Sigma)$. Then $X_j = e^{Y_j}/D$ for $j = 1, \dots, d-1$ and $X_d = 1/D$ where $D = 1 + \sum_{j=1}^{d-1} e^{Y_j}$ normalizes the vector. We can extend $\boldsymbol{Y}$ to a point $\widetilde{\boldsymbol{Y}} = (\boldsymbol{Y}, 0)$ in $d$ dimensions with a $d$'th component that is always 0. Then $\boldsymbol{X} = \exp(\widetilde{\boldsymbol{Y}})/\sum_{j=1}^d \exp(\widetilde{Y}_j)$ componentwise.

## 5.5 Multivariate Poisson and other distributions

The normal distribution generalizes to the multivariate normal as described in §5.2, and we also saw a multivariate $t$ distribution there. It is natural then to seek out multivariate generalizations of all the useful one-dimensional distributions. But Kotz et al. (2000) include at least 12 different bivariate Gamma distributions. Thus, while we may safely refer to 'the multivariate normal', we should speak not of 'the multivariate gamma' but 'a multivariate gamma', and then be specific about which one is intended. As a rule, we cannot put the word "multivariate" in front of an arbitrary distribution's name and expect there to be a unique best choice.

There are far too many multivariate distributions to cover them all. One reason for the large number is that generalizing one property of a univariate distribution may be incompatible with generalizing another. In this section we consider a small number of distributions, emphasizing those that can be sampled in an easy and interpretable way.

There is a well known ***bivariate Poisson distribution***. To sample it, generate $Z_j \sim \mathrm{Poi}(\lambda_j)$ independently for $j = 1, 2, 3$, and then let $X_1 = Z_1 + Z_3$ and $X_2 = Z_2 + Z_3$. Each $X_j \sim \mathrm{Poi}(\lambda_j + \lambda_3)$, for $j = 1, 2$, because sums of independent Poisson random variables are Poisson. If $\lambda_3 > 0$, then $X_1$ and $X_2$ are positively correlated, due to the common contribution of $Z_3$.

We can generalize the bivariate case to obtain a ***multivariate Poisson distribution***. Suppose that $Z_j \sim \mathrm{Poi}(\lambda_j)$ are independent, for $j = 1, \dots, p$ and let

$$\begin{pmatrix} X_1 \\ \vdots \\ X_d \end{pmatrix} = \begin{pmatrix} A_{11} & A_{12} & \dots & A_{1p} \\ \vdots & \vdots & \ddots & \vdots \\ A_{d1} & A_{d2} & \dots & A_{dp} \end{pmatrix} \begin{pmatrix} Z_1 \\ \vdots \\ Z_p \end{pmatrix}, \tag{5.16}$$

where $A_{ij} \in \{0, 1\}$. That is $\boldsymbol{X} = A\boldsymbol{Z}$ where $A$ is a binary matrix and $\boldsymbol{Z}$ is a vector of independent Poisson random variables with mean vector $\lambda \in (0, \infty)^p$. We may think of $X_j$ as counting flaws of type $j$ and $Z_k$ as the number of occurrences of problem $k$, where $A_{jk} = 1$ if and only if problem $k$ causes flaw $j$.

The expected value of $\boldsymbol{X}$ is $\mu = A\lambda$ and each $X_j \sim \mathrm{Poi}(\mu_j)$. The variable $X_j$ has a contribution from each of the $Z_k$ with $A_{jk} = 1$. If $X_j$ and $X_\ell$ share no $Z_k$ in common, that is $A_{jk}A_{\ell k} = 0$ for all $k$, then $X_j$ and $X_\ell$ are independent. If $A_{jk}A_{\ell k} = 1$ for some $k$, then $X_j$ and $X_\ell$ have a positive dependence stemming from the common contribution $Z_k$.

This multivariate Poisson distribution cannot have any negatively correlated components. If negatively correlated Poisson random variables are required then the copula sampling method described in §5.6 provides one way to do this.

The summation method above can be used to create multivariate distributions in other families of random variables. If $Z_k$ are independent $\mathrm{Gam}(\theta_k)$ random variables, for $k = 1, \dots, p$ and $\boldsymbol{X} = A\boldsymbol{Z}$ for $A \in \{0, 1\}^{d\times p}$ then $\boldsymbol{X}$ has nonnegatively correlated components $X_j \sim \mathrm{Gam}\left(\sum_{k=1}^p A_{jk}\theta_k\right)$. The construction requires a family of distributions in which sums of independent variables remain in that family, though usually with a new parameter. In the case of the gamma distribution, we get the desired property if the random variables being summed share the same scale parameter.

The Marshall-Olkin ***bivariate exponential*** distribution has

$$\mathbb{P}(X_1 > x_1, X_2 > x_2) = \exp\left(-\lambda_1 x_1 - \lambda_2 x_2 - \lambda_3 \max(x_1, x_2)\right) \tag{5.17}$$

for $x_j \geqslant 0$ where $\lambda_k \geqslant 0$ and $\min(\lambda_1, \lambda_2) + \lambda_3 > 0$. What is special about this distribution is that both $X_1$ and $X_2$ are exponentially distributed, while

$$\mathbb{P}(X_1 > s_1 + t,\ X_2 > s_2 + t \mid X_1 > t,\ X_2 > t) = \mathbb{P}(X_1 > s_1, X_2 > s_2) \tag{5.18}$$

which generalizes the memoryless property $\mathbb{P}(X > s + t \mid X > t) = \mathbb{P}(X > s)$ of the univariate exponential distribution. It is the only distribution with exponential margins that satisfies (5.18). It also has the property that $\min(X_1, X_2)$ is exponentially distributed; not all bivariate exponential distributions have an exponentially distributed minimum.

The Marshall-Olkin bivariate exponential distribution has a 'shock model' interpretation. Let $Z_k \sim \mathrm{Exp}(1)/\lambda_k$ for $k = 1, 2, 3$ be independent and suppose that $X_1 = \min(Z_1, Z_3)$ and $X_2 = \min(Z_2, Z_3)$. Then $\boldsymbol{X}$ has the bivariate exponential distribution with tail probabilities given by (5.17). If $X_j$ is the lifetime of item $j$, then $Z_1$ is the time to a shock that is fatal for item 1, $Z_2$ is the time to a shock that is fatal for item 2, and $Z_3$ is the time to a shock that kills both. This distribution has the property that $0 < \mathbb{P}(X_1 = X_2) < 1$, which is unusual in bivariate distributions, but well suited to components subject to a common failure cause.

More generally, suppose that there are $p$ different shock events at independent times $Z_k \sim \mathrm{Exp}(1)/\lambda_k$ for $k = 1, \dots, p$. Let $A \in \{0,1\}^{d\times p}$ be a fixed matrix, with the interpretation that $A_{jk} = 1$ if event $k$ is fatal to item $j = 1, \dots, d$. Letting

$$X_j = \min_{1\leqslant k\leqslant p} \{Z_k \mid A_{jk} = 1\}$$

gives a random vector $\boldsymbol{X} \in [0,\infty)^d$ whose margins are exponentially distributed and whose bivariate margins have the distribution (5.17). This is the Marshall-Olkin multivariate exponential distribution. The marginal distribution of $X_j$ is exponential with rate $\nu_j = \sum_{k=1}^p A_{jk}\lambda_k$. For a proper distribution, we need $\min_{1\leqslant j\leqslant d} \nu_j > 0$.

There is also some benefit in treating a set of independent Poisson random variables as a random vector that we then sample recursively. Suppose that $X_j \sim \mathrm{Poi}(\lambda_j)$ independently for $j = 1, \dots, d$. If $d$ is very large and most of the $\lambda_j$ are small, then straightforward sampling will consume a lot of effort generating components $X_j$ that turn out to be 0.

We can speed things up using the fact that if $Y \sim \mathrm{Poi}(\nu)$ and $Z \sim \mathrm{Poi}(\gamma)$ are independent then the distribution of $Y$ given $Y+Z = n$ is $\mathrm{Bin}(n, \nu/(\nu+\gamma))$. For $0 \leqslant y \leqslant n$ the value of $\mathbb{P}(Y = y \mid Y + Z = n)$ is

$$\frac{(e^{-\nu}\nu^y/y!)(e^{-\gamma}\gamma^z/z!)}{e^{-\nu-\gamma}(\nu+\gamma)^n/n!} = \binom{n}{y}\Bigl(\frac{\nu}{\nu+\gamma}\Bigr)^y\Bigl(1-\frac{\nu}{\nu+\gamma}\Bigr)^{n-y}$$

after some cancellation and rearranging the factors. Much the same argument shows that the distribution of $(X_1, X_2, \dots, X_d)$ given that $X_{1:d} \equiv \sum_{j=1}^d X_j = m$ is $\mathrm{Mult}(m; p_1, \dots, p_d)$ where $p_j = \lambda_j/\sum_{k=1}^d \lambda_k$.

To use this approach, we first sample $M \sim \mathrm{Poi}(\sum_{k=1}^d \lambda_k)$ then generate $\boldsymbol{X} \sim \mathrm{Mult}(M, p_1, \dots, p_d)$. We gain most from the speedup if we can compute sums of $\lambda_j$ in closed form without computing each $\lambda_j$ and summing them. For example if $\lambda_j = c\int_{j-1/2}^{j+1/2} x^{-a}\,\mathrm{d}x$ for $a > 1$ and $c > 1$, then we can sample a consecutive range $X_{r:s} = \sum_{j=r}^s X_j \sim \mathrm{Poi}(\lambda_{r:s})$ for $\lambda_{r:s} = c\int_{r-1/2}^{s+1/2} x^{-a}\,\mathrm{d}x = c((r-1/2)^{1-a} - (s-1/2)^{1-a})/(a-1)$. Similarly the ratios of $\lambda$ over subranges needed by the multinomial algorithm are available in closed form for these $\lambda_j$.

## 5.6 Copula-marginal sampling

A final generic method is the copula-marginal method. It may be viewed as another generalization of the one-dimensional CDF inversion method. The way it appears in applications is as a generalization of the $QQ$ transformation (4.3).

Let $\boldsymbol{X} \sim F$, where the marginal distribution of $X_j$ is $F_j(x_j)$, and suppose for simplicity that every $F_j$ is a continuous distribution. Then the random vector $\boldsymbol{U} = (F_1(X_1), \dots, F_d(X_d))$ has a multivariate uniform distribution: every component $U_j$ has the $\mathbf{U}(0,1)$ distribution, although the $U_j$ are not in general independent. This multivariate uniform distribution is known as the copula of $F$. We'll call it $C$. The procedure for copula-marginal sampling is as follows:

$$\begin{aligned} &\boldsymbol{U} \sim C, \quad \text{then} \\ &X_j = F_j^{-1}(U_j), \quad j = 1, \dots, d. \end{aligned} \tag{5.19}$$

Copula sampling is fully general because of Sklar's theorem (Sklar, 1959), from which we learn that for any distribution $F$, there is a copula $C$ for which (5.19) will deliver $\boldsymbol{X} \sim F$. If all of the $F_j$ are continuous distributions, then $C$ is unique, but otherwise not.

Sampling from copulas is not necessarily any easier than sampling from other multivariate distributions. The marginal distributions are defined through $d$ one-dimensional curves, but the copula is inherently $d$-dimensional. So equation (5.19) just pushes the difficulty of sampling into the copula.

**Example 5.6** (Independence copula)**.** The independence copula is $\mathbf{U}(0,1)^d$. That is $C(\boldsymbol{u}) = \prod_{j=1}^d u_j$ and $X_j = F_j^{-1}(U_j)$ are independent.

At the other extreme from the independence copulas are monotonic copulas. In these, there is one single $U \sim \mathbf{U}(0,1)$ and then each $U_j$ is either $U$ or $1-U$. When the $F_j$ are continuous, then either $X_j = F_j^{-1}(F_k(X_k))$ (if $U_j = U_k$) or $X_j = F_j^{-1}(1 - F_k(X_k))$ (if $U_j = 1 - U_k$) and so there is a deterministic relationship among the $d$ variables.

**Example 5.7** (Comonotonic copula)**.** The comonontonic copula describes a deterministic monotone dependence among all $d$ components of $\boldsymbol{X}$. It has $C(u_1, \dots, u_d) = \min(u_1, \dots, u_d)$. As a result $\mathbb{P}(U_1 = U_2 = \cdots = U_d) = 1$, and $X_j = F_j^{-1}(U)$ for all $j$ and one $U \sim \mathbf{U}(0,1)$.

**Example 5.8** (Countermonotonic copula)**.** For $d = 2$ the countermonotonic copula has $U_1 = 1 - U_2$. It describes perfect deterministic negative dependence between two random variables. It can be sampled via $X_1 = F_1^{-1}(U)$ and $X_2 = F_2^{-1}(U)$ for $U \sim \mathbf{U}(0,1)$.

There are some classical families of copulas, described for example in Nelsen (2006), but for our purposes here, we will not sample from them. Instead, we will use copulas to combine the dependence structure from one multivariate distribution with the marginal distributions of one or more others. Suppose that we can sample $\boldsymbol{Y} \sim G$ where $G$ is a continuous distribution on $\mathbb{R}^d$ and that we want to sample a distribution with $d$ given continuous marginal distributions $F_1, \dots, F_d$. Then taking $X_j = F_j^{-1}(G_j(Y_j))$, where $G_j$ is the marginal distribution of $Y_j$ gives a random vector $\boldsymbol{X}$ with the marginal distributions $F_j$ and the same copula as $G$.

**Example 5.9** (Gaussian copula)**.** Given a correlation matrix $R \in \mathbb{R}^{d \times d}$ and $d$ CDFs $F_1, \dots, F_d$, the ***Gaussian copula*** method takes

$$\boxed{\boldsymbol{U} \sim \mathbf{U}(0,1)^d, \quad \boldsymbol{Y} = R^{1/2}\Phi^{-1}(\boldsymbol{U}), \quad \text{and} \quad X_j = F_j^{-1}(\Phi(Y_j)), \quad j = 1, \dots, d.}$$

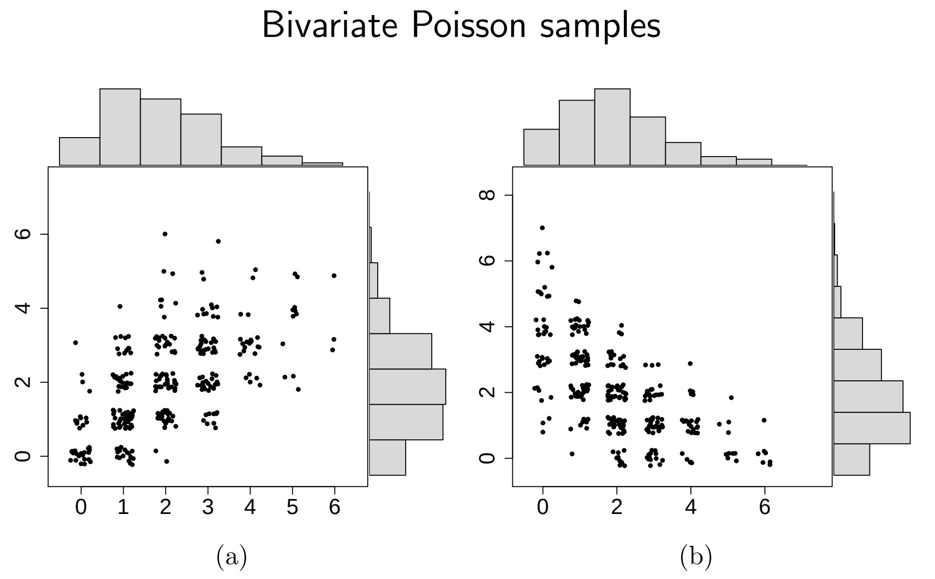

Figure 5.2: This figure shows samples from two different bivariate distributions. The 300 sample values are integers but they have been randomly jittered in the plots. The sample marginal distributions are shown as histograms. For panel (a) both of the marginal distributions are Poi(2). They are joined via a Gaussian copula with correlation $\rho = 0.7$. Panel (b) is the same except that $\rho = -0.7$.

The matrix root satisfies $R^{1/2}R^{\mathsf{T}/2} = R$. This is sometimes called the NORTA (normal to anything) method and is also known as the Nataf transformation after Nataf (1962). It is a rough and ready technique based on the computational convenience of the multivariate normal distribution. It should be used with caution unless we have reason to believe that the appropriate copula for $\boldsymbol{X}$ is close to the Gaussian one. Sampling with the $t$ copula, as described below, is an alternative with quite different assumptions.

When we combine a copula from one family of distributions with margins of another, we may fail to preserve any of the nice properties that held in either family, such as closure under forming sums or taking minima. As such there is an aesthetic objection to copula-marginal sampling. On the other hand, the property we desire most might be incompatible with the standard multivariate distributions. For example, Figure 5.2 shows two samples from a distribution with Poisson margins and the Gaussian copula. One has a negative dependence, which would not have been possible with multivariate Poisson distribution given by (5.16).

The vector $\boldsymbol{X}$ is a component by component transformation of $\boldsymbol{Y} \sim \mathcal{N}(0, R)$. The correlation matrix of $\boldsymbol{Y}$ is $R$ but $\boldsymbol{X}$ generally has some other correlation matrix. Indeed the marginal distributions $F_j$ might not have finite variances and in that case $\mathrm{Corr}(\boldsymbol{X})$ is not even defined. When each $F_j$ is a continuous distribution, then the rank correlations of $\boldsymbol{X}$ are the same as those of $\boldsymbol{Y}$. The ***rank correlation*** of $X_j$ and $X_k$ is the ordinary Pearson correlation of $F_j(X_j)$ and $F_k(X_k)$.

For a Gaussian random vector, the Pearson correlation and the rank correlation are related via

$$\rho_{\mathrm{rank}} = \frac{2}{\pi}\arcsin(\rho) \tag{5.20}$$

(McNeil et al., 2005, Theorem 5.26) and so to get a desired rank correlation $\rho_{\mathrm{rank}}$ for $X_j$ and $X_k$ we take $R_{jk} = \rho_{jk} = \sin(\pi\rho_{\mathrm{rank}}/2)$.

If we have a set of desired marginal distributions and a desired rank correlation matrix $R$, then the Gaussian copula will generate random vectors from one of the joint distributions that matches these criteria. If we replace the desired rank correlation by a desired ordinary correlation matrix, then the situation is more complicated. For instance with $F_1 = \mathrm{Exp}(1)$ and $F_2 = \mathbf{U}(0,1)$ a rank correlation of 1 is possible for $X_1$ and $X_2$, but a Pearson correlation of 1 is not possible. See Exercise 5.11. In cases like this no distribution with the desired correlation matrix and marginal distributions exists. There are $d$-dimensional distributions with well-defined correlations and continuous marginal distributions that cannot be sampled via the Gaussian copula. There is an example on page 50 of the end notes.

When we want to closely match an empirical correlation structure, from a sample $\boldsymbol{Y}_1, \dots, \boldsymbol{Y}_n \in \mathbb{R}^d$, then we may form

$$Z_{ij} = \Phi^{-1}\Bigl(\frac{W_{ij} - 1/2}{n}\Bigr) \tag{5.21}$$

where

$$W_{ij} = \sum_{\ell=1}^{n} \mathbb{1}_{Y_{\ell j} \leqslant Y_{ij}}$$

is the rank of $Y_{ij}$ among $Y_{1j}, \dots, Y_{nj}$, and estimate $\Sigma$ by the sample correlation matrix $\widehat{\Sigma}$, of $\boldsymbol{Z}_i$. When there are no ties in the data,

$$\widehat{\Sigma} = \frac{1}{\widetilde{n}}\sum_{i=1}^{n} \boldsymbol{Z}_i\boldsymbol{Z}_i^{\mathsf{T}}, \quad \widetilde{n} = \sum_{i=1}^{n} \Phi^{-1}\left(\frac{i-1/2}{n}\right)^2 \doteq n. \tag{5.22}$$

Researchers in insurance and finance have long known about one flaw with the Gaussian copula. It has a ***tail independence*** property. Suppose that $\boldsymbol{X} \sim \mathcal{N}(\mu, \Sigma)$ for $d \geqslant 2$, with $\Sigma$ of full rank, and let $\sigma_j = \sqrt{\Sigma_{jj}}$ for $j = 1, \dots, d$. Then for any two distinct indices $j, k \in \{1, \dots, d\}$,

$$\lim_{u \to 1-} \mathbb{P}\bigl(X_j > F_j^{-1}(u) \mid X_k > F_k^{-1}(u)\bigr) = 0. \tag{5.23}$$

Property (5.23) describes an asymptotic independence of extreme events in the two variables. If $X_j$ represents the loss from one insured and $X_k$ represents the loss from another, then (5.23) implies that when an extremely large loss $X_k$ takes place then the chance that $X_j$ is comparably large is negligible. Where two losses could both be large, as in some insurance problems or in finance where multiple counterparties may be bankrupted by the same event, the Gaussian copula gives a false sense of security.

The limit (5.23) still holds if $\mathrm{Corr}(X_j, X_k) = 0.999$. A strong correlation in the center of the distribution may have a surprisingly weak presence in the tails. In a given problem it may be difficult to know what value the limit in (5.23) should have, because data on extreme events is by definition rare. But the value 0 given by the Gaussian copula is a dangerous default.

**Example 5.10** ($t$ copula sampling)**.** Given a correlation matrix $R \in \mathbb{R}^{d\times d}$, a degrees of freedom parameter $\nu > 0$, and $d$ CDFs $F_1, \dots, F_d$, $t$ copula sampling takes

$$\boldsymbol{Y} \sim \mathrm{t}(\mu, \nu, R), \quad \text{and} \quad X_j = F_j^{-1}(T_\nu(Y_j)), \quad j = 1, \dots, d,$$

where $T_\nu$ is the CDF of the $t_{(\nu)}$ distribution. This method avoids the tail independence property of Gaussian copula sampling.

Removing tail independence gives a strong advantage for the $t$ copula over the Gaussian one, even for normally distributed $\boldsymbol{X}$ and perhaps much more so when the marginal distributions of $\boldsymbol{X}$ have long tails. But we should still be cautious. The $t$ copula gives a tail dependence for very large $X_1$ and $X_2$. By symmetry of the $t$ distribution, the very same dependence occurs in the upper tail of $-\boldsymbol{X}$. It may be more realistic to believe that large upward movements for two stocks will show a different tail dependence than large downward movements.

Continuing the previous discussion, there is also a tail dependence issue for $(X_1, -X_2)$ describing extreme upward movements in one stock coinciding with downward movements in another. Those would be hazardous for an investor who held one stock and had a short position in the other. There is no special reason to think that the tail dependence for opposite movements should match either the one for upward or downward movements. With $d$ random variables there are $d(d-1)/2$ different bivariate tail dependencies to consider. Even if we tune them all, the copula we use will have consequences for the behavior of triples $(X_1, X_2, X_3)$. Because combinations of rare events are necessarily even rarer, choosing a copula to match all forms of tail behavior would require some extrapolation beyond empirical data.

Sampling from a copula model is easier than knowing which one to use. There is more discussion of copula sampling in the bibliographic notes.

## 5.7 Random points on the sphere

Here we look at sampling random points from the sphere in dimension $d$, and some closely related sampling problem of sampling spherically and elliptically symmetric distributions. We give special attention to the uniform distribution but also consider non-uniform points in the sphere.

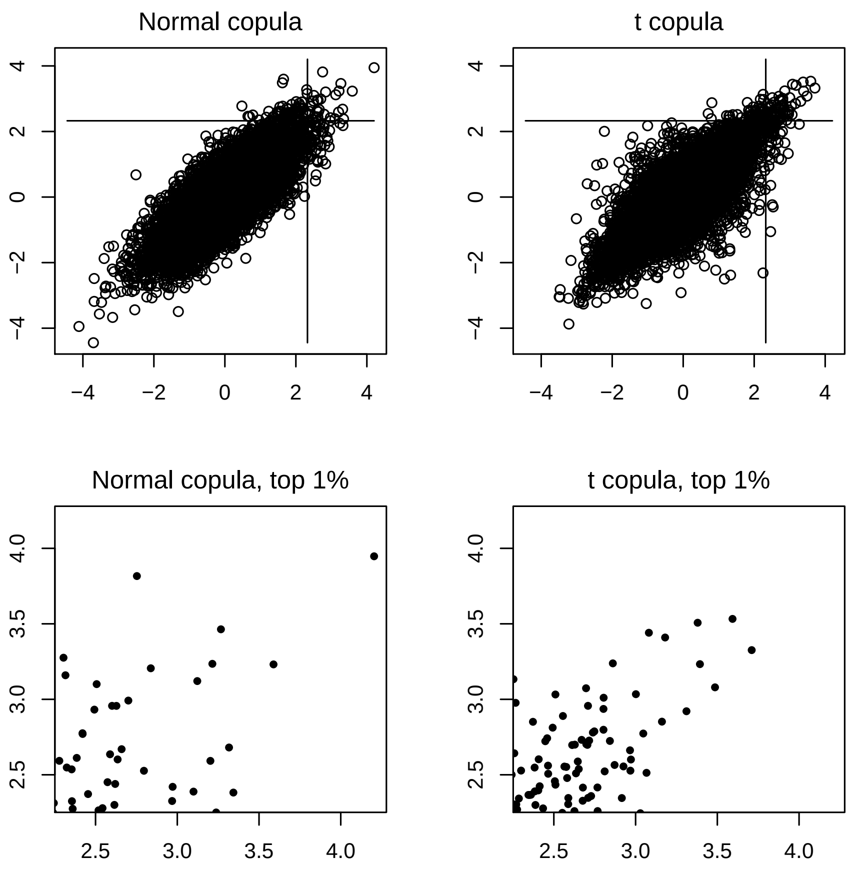

Figure 5.3: The upper left panel shows a sample of 10,000 points from a bivariate unit normal distribution with $\rho = 0.8$. The upper right panel shows a sample with normal marginal distributions but a copula from the bivariate $t$ distribution on $\nu = 5$ degrees of freedom with the same correlation as the normal data. The bottom two panels zoom in on the top 1% of the distribution for the upper panels.

The unit sphere in $d$ dimensions is

$$S^{d-1} = \{\boldsymbol{x} \mid \|\boldsymbol{x}\| = 1\}.$$

The conceptually simplest way to generate $\boldsymbol{X} \sim \mathbf{U}(S^{d-1})$ is to take $\boldsymbol{X} = \boldsymbol{Z}/\|\boldsymbol{Z}\|$ where $\boldsymbol{Z} \sim \mathcal{N}(0, I_d)$. This method works because the probability density function of $\boldsymbol{z}$ is $\varphi(\boldsymbol{z}; 0, I_d) = (2\pi)^{-d/2} \exp(-\|\boldsymbol{z}\|^2/2)$ which is constant on spherical contours. In the Box-Muller method of §4.6 we took a random point on the unit circle $S^1$ and attached a $\sqrt{\chi^2_{(2)}}$ radius to it to sample from $\mathcal{N}(0, I_2)$. Here we reverse the process, using Gaussian variables to sample on the general $d$-dimensional hypersphere.

For $d=2$ and $d=3$, there are convenient alternative samplers that use only $d-1$ random variables. For $d=2$ we let

$$U \sim \mathbf{U}(0,1) \quad \text{and} \quad \boldsymbol{X} = (\cos(2\pi U), \sin(2\pi U)).$$

For $d=3$, we take

$$\begin{aligned}&\boldsymbol{U} \sim \mathbf{U}(0,1)^2, \quad R = \sqrt{U_1(1-U_1)}, \quad \theta = 2\pi U_2, \quad \text{and then,}\\ &\boldsymbol{X} = \bigl(1-2U_1,\, 2R\cos(\theta),\, 2R\sin(\theta)\bigr).\end{aligned}$$

The first component of $\boldsymbol{X}$ for $d=3$ illustrates Archimedes' hat box theorem: the height above/below the equator for a random point on the unit sphere has the $\mathbf{U}(-1,1)$ distribution. By reducing the dimension by one, these methods make stratified sampling methods (see §8.4) more effective. For generalizations to $d>3$ see the chapter end notes.

With uniformly distributed points on a sphere we can now sample from any spherically symmetric distribution we want, as long as we have a way to sample the desired distribution for the radius $R = \|\boldsymbol{X}\|$.

**Example 5.11** (Density $\propto \exp(-\|\boldsymbol{x}\|)$)**.** When $\boldsymbol{X} \sim f(\boldsymbol{x}) \propto \exp(-\|\boldsymbol{x}\|)$ then

$$\mathbb{P}(r \leqslant \|\boldsymbol{X}\| \leqslant r + \mathrm{d}r) \propto \exp(-r) r^{d-1}\mathrm{d}r,$$

which we recognize as the $\mathrm{Gam}(d)$ distribution (§4.8). The factor $r^{d-1}$ comes from the $d-1$-dimensional volume of $S^{d-1}(r) = \{\boldsymbol{x} \mid \|\boldsymbol{x}\| = r\}$. Therefore we may take

$$\boldsymbol{Z} \sim \mathcal{N}(0, I_d), \quad R \sim \mathrm{Gam}(d), \quad \text{and} \quad \boldsymbol{X} = R\,\frac{\boldsymbol{Z}}{\|\boldsymbol{Z}\|}. \tag{5.24}$$

Note that when $d \geqslant 2$ then individual components $|X_j|$ are not exponentially distributed.

**Example 5.12** (Density $\propto \|\boldsymbol{x}\|^k \mathbf{1}_{\{\|\boldsymbol{x}\| \leqslant 1\}}$, $k > -d$)**.** Here the density of $R = \|\boldsymbol{X}\|$ is proportional to $r^{d+k-1}\mathbf{1}_{0<r<1}$, the $\mathrm{Beta}(d+k, 1)$ distribution (Example 4.29). Therefore we may take

$$\boldsymbol{Z} \sim \mathcal{N}(0, I_d), \quad R \sim \mathrm{Beta}(d+k, 1), \quad \text{and} \quad \boldsymbol{X} = R\,\frac{\boldsymbol{Z}}{\|\boldsymbol{Z}\|}. \tag{5.25}$$

For $k=0$ we get the uniform distribution within the ball $B_d = \{\boldsymbol{x} \in \mathbb{R}^d \mid \|\boldsymbol{x}\| \leqslant 1\}$. For $0 > k > -d$ we get an unbounded density, with a singularity at $\boldsymbol{x} = 0$.

In the previous two examples we were able to write $f(\boldsymbol{x}) \propto h(\|\boldsymbol{x}\|)$ and recognize the density proportional to $r^{d-1}h(r)$. When we do not know the resulting radial distribution we may still be able to apply acceptance-rejection sampling (see §4.7). For that we must find a density function $g(r) = c_g \widetilde{g}(r)$ on

$[0, \infty)$ that we know how to sample from and a constant $c$ for which we know $r^{d-1}h(r) \leqslant c\widetilde{g}(r)$. We do not need to know the value of $c_g$.

In addition to choosing a radial distribution for spherically symmetric random vectors, we can also apply linear transformations. We did this in §5.2 to sample multivariate normal and $t$ random vectors and in Example 5.5 to sample uniformly in the simplex. Now suppose that $\boldsymbol{X} \sim \mathbf{U}(B_d)$ and let $\mu \in \mathbb{R}^d$ and matrix $C \in \mathbb{R}^{d\times d}$ be invertible. Then $\boldsymbol{Y} = \mu + C\boldsymbol{X}$ is uniformly distributed in the ellipsoidal ball

$$\mathcal{E}(\mu, \Sigma) = \bigl\{\boldsymbol{y} \in \mathbb{R}^d \mid (\boldsymbol{y} - \mu)^{\mathsf{T}}\Sigma^{-1}(\boldsymbol{y} - \mu) \leqslant 1\bigr\}$$

where $\Sigma = CC^{\mathsf{T}}$. Note that $\mu + C\boldsymbol{X}$ does not generally have the uniform distribution on the surface of $\mathcal{E}(\mu, \Sigma)$ when $\boldsymbol{X} \sim \mathbf{U}(S^{d-1})$.

We may want a non-uniform distribution on the unit sphere to model phenomena with preferred directionality. There are ad hoc ways to sample. We could for example take $\boldsymbol{X} \sim \mathcal{N}(\mu, \Sigma)$ and deliver $\boldsymbol{X}/\|\boldsymbol{X}\|$.

A commonly used non-uniform distribution on $S^{d-1}$ is the ***von Mises-Fisher*** distribution which has probability density function

$$f(\boldsymbol{x}; \mu, \kappa) \propto \widetilde{f}(\boldsymbol{x}; \mu, \kappa) = \exp(\kappa\mu^{\mathsf{T}}\boldsymbol{x}).$$

The parameters are $\kappa \geqslant 0$ and $\mu \in S^{d-1}$. For $\kappa > 0$ the distribution has a favored direction $\mu$. Larger values of $\kappa$ yield even stronger concentration of $\boldsymbol{X}$ around $\mu$.

The vector $\boldsymbol{X}$ has a component parallel to $\mu$ and a component orthogonal to $\mu$. Since $f$ is unaffected by the component of $\boldsymbol{X}$ orthogonal to $\mu$, that component can be generated by sampling the uniform distribution in $\mathbb{R}^{d-1}$ and appropriately scaling it. The main difficulty is to sample $W \equiv \mu^{\mathsf{T}}\boldsymbol{X}$.

Combining Ulrich (1984), Wood (1987) and Hoff (2009) we get the following algorithm:

$$\begin{aligned}
&W \sim h, \quad h(w) \propto (1-w^2)^{(d-3)/2}\exp(\kappa w) \quad \text{on } (-1,1),\\
&\boldsymbol{V} \sim \mathbf{U}(S^{d-2}), \quad \text{and}\\
&\boldsymbol{X} \leftarrow \begin{pmatrix}\mu & B\end{pmatrix}\begin{pmatrix} W \\ \sqrt{1-W^2}\boldsymbol{V}\end{pmatrix} = W\mu + \sqrt{1-W^2}B\boldsymbol{V}
\end{aligned}$$

where $B \in \mathbb{R}^{d\times(d-1)}$ is any matrix for which $A = \begin{pmatrix}\mu & B\end{pmatrix}$ is orthogonal. The matrix $A$ can be precomputed by running the Gram-Schmidt algorithm on $\begin{pmatrix}\mu & \widetilde{B}\end{pmatrix}$ where $\widetilde{B} \sim \mathcal{N}(0, I_d, I_{d-1})$. When $A = \begin{pmatrix}\mu & B\end{pmatrix} = I_d$ we get samples from $f(\cdot; \mu, \kappa)$ with $\mu = \begin{pmatrix}1 & 0 & \cdots & 0\end{pmatrix}^{\mathsf{T}}$.

The variable $W$ can be sampled by acceptance-rejection. This involves precomputation of

$$\begin{aligned}
b &\leftarrow (-2\kappa + \sqrt{4\kappa^2 + (d-1)^2})/(d-1)\\
x_0 &\leftarrow (1-b)/(1+b)
\end{aligned}$$

$$c \leftarrow \kappa x_0 + (d-1)\log(1-x_0^2),$$

and then each sample uses

$$\begin{aligned} Z &\sim \mathrm{Beta}((d-1)/2,(d-1)/2) \\ W &\leftarrow (1-(1+b)Z)/(1-(1-b)Z) \\ U &\sim \mathbf{U}(0,1) \end{aligned}$$

where $W$ is accepted if and only if $\kappa W + (d-1)\log(1-x_0W) - c \geqslant \log(U)$.

## 5.8 Random matrices

Many problems require that we generate a random matrix $X \in \mathbb{R}^{m\times d}$. The dimensions of $X$ are chosen so that we may think of it as $m$ random vectors in $\mathbb{R}^d$. We may need to generate $n$ independent copies of this matrix but for this section we focus on generating just one of them.

The elements of $X$ can always be rearranged into an $md$ by 1 vector, and then we can use methods for sampling random vectors. We turn to random matrix methods when there is some special structure to the problem that makes it easier than just sampling a large vector. If there is too much simplifying structure, such as independence between rows (or between columns) then the problem simplifies still further to generation of independent random vectors row by row or column by column. So random matrix methods are interesting when the problem has less simplifying structure than a collection of independent random vectors has, but more structure than an arbitrary $md$-dimensional dependence.

### Matrix normal distribution

The matrix normal distribution is often used as a model for data matrices that have correlations in both their rows and their columns. In genomic applications, columns might correspond to correlated genes, while the rows might correspond to per subject measurements with unwanted correlations due to imperfections in laboratory processing. In econometrics, the rows may correspond to sampling times, the columns to commodities and the entries to prices.

We write $X \sim \mathcal{N}(\mu,\Gamma,\Sigma)$ for matrices $\mu \in \mathbb{R}^{m\times d}$, $\Gamma \in \mathbb{R}^{m\times m}$ and $\Sigma \in \mathbb{R}^{d\times d}$ when the components $X_{ij}$ of $X$ have a multivariate normal distribution with $\mathbb{E}(X_{ij}) = \mu_{ij}$ and $\mathrm{Cov}(X_{ij}, X_{k\ell}) = \Gamma_{ik}\Sigma_{j\ell}$. The matrices $\Gamma$ and $\Sigma$ are assumed to be positive semi-definite. The matrix normal distribution needs one more constraint to be properly identified. Replacing $\Gamma$ by $c\Gamma$ and $\Sigma$ by $c^{-1}\Sigma$ for $c>0$ does not change the distribution. If we suppose that $\mathrm{tr}(\Gamma) = n$, then the model is identified and of course other normalizations may suit a given problem better.

For an arbitrary normal distribution on $md$ components we would require an $md$ by $md$ covariance matrix with $md(md+1)/2$ parameters in it. The matrix normal distribution uses only $m(m+1)/2 + d(d+1)/2$ parameters for the covariance matrix. For large $m$ and $d$ that is an enormous simplification.

Suppose that $X \sim \mathcal{N}(\mu, \Sigma, \Gamma)$, and $A$ and $B$ are nonrandom matrices for which the product $AXB^\mathsf{T}$ is well defined. Then

$$AXB^\mathsf{T} \sim \mathcal{N}(A\mu B^\mathsf{T}, A\Sigma A^\mathsf{T}, B\Gamma B^\mathsf{T}). \tag{5.26}$$

We can easily generate a matrix $Z \in \mathbb{R}^{m\times d}$ of independent $\mathcal{N}(0,1)$ entries, which we write as $Z \sim \mathcal{N}(0, I_m, I_d)$. By choosing $A$ and $B$ with $AA^\mathsf{T} = \Sigma$ and $BB^\mathsf{T} = \Gamma$ we can sample from $\mathcal{N}(\mu, \Sigma, \Gamma)$ by taking

$$\begin{aligned} &Z \sim \mathcal{N}(0, I_m, I_d), \quad \text{and}\\ &X = \mu + AZB^\mathsf{T}. \end{aligned} \tag{5.27}$$

## Random orthogonal matrices

The set of orthogonal matrices of order $d$ is

$$\mathbb{O}_d = \{Q \in \mathbb{R}^{d\times d} \mid Q^\mathsf{T}Q = I_d\}.$$

Multiplying a list of vectors by an orthogonal matrix preserves all of their lengths. It follows that interpoint distances are preserved and so then are all angles between vectors. The operation of multiplying by an orthogonal matrix is a rigid motion sometimes called a rotation. In other usages the term rotation is restricted to $Q$ with $\det(Q) = 1$. For $Q \in \mathbb{O}_d$ we have $\det(Q) = \pm 1$.

There is a unique uniform distribution on $\mathbb{O}_d$, known as the Haar measure, that we denote by $\mathbf{U}(\mathbb{O}_d)$. If $Q \sim \mathbf{U}(\mathbb{O}_d)$ and $A \subset \mathbb{O}_d$ then $\mathbb{P}(Q \in A) = \mathbf{vol}(A)/\mathbf{vol}(\mathbb{O}_d)$. This uniform distribution is invariant under multiplication on the left or right. That is, if $Q \sim \mathbf{U}(\mathbb{O}_d)$ and $Q_*$ is some fixed matrix in $\mathbb{O}_d$, then both $Q_*Q \sim \mathbf{U}(\mathbb{O}_d)$ and $QQ_* \sim \mathbf{U}(\mathbb{O}_d)$ hold. For instance, let $Q_*^\mathsf{T}A = \{Q_*^\mathsf{T}P \mid P \in A\}$. Then

$$\mathbb{P}(Q_*Q \in A) = \mathbb{P}(Q_*^\mathsf{T}Q_*Q \in Q_*^\mathsf{T}A) = \frac{\mathbf{vol}(Q_*^\mathsf{T}A)}{\mathbf{vol}(\mathbb{O}_d)} = \frac{\mathbf{vol}(A)}{\mathbf{vol}(\mathbb{O}_d)} = \mathbb{P}(Q \in A)$$

because multiplying each element of $A$ by $Q_*^\mathsf{T}$ rotates that set without changing its volume.

If $Q \sim \mathbf{U}(\mathbb{O}_d)$, then the first column of $Q$ is a uniformly distributed orthogonal vector as studied in §5.7. We can generate it by sampling a Gaussian random vector $\boldsymbol{Z}_1 \sim \mathcal{N}(0, I_d)$ and normalizing it to $\boldsymbol{Z}_1/\|\boldsymbol{Z}_1\|$. The second column is uniformly distributed over the unit sphere, subject to being orthogonal to the first one. We can sample a random vector, $\boldsymbol{Z}_2 \sim \mathcal{N}(0, I_d)$, obtain $\widetilde{\boldsymbol{Z}}_2 = \boldsymbol{Z}_2 - \boldsymbol{Z}_1\boldsymbol{Z}_2^\mathsf{T}\boldsymbol{Z}_1/\|\boldsymbol{Z}_1\|$ (orthogonal to $\boldsymbol{Z}_1$) and take $\widetilde{\boldsymbol{Z}}_2/\|\widetilde{\boldsymbol{Z}}_2\|$ as the second column of $Q$.

If we continue, generating all $d$ columns of $Q$ as Gaussian vectors, orthogonalizing them and standardizing them, we will arrive at $Q \sim \mathbf{U}(\mathbb{O}_d)$. This procedure is simply Gram-Schmidt orthogonalization applied to a matrix $Z \in \mathbb{R}^{d\times d}$ of independent $\mathcal{N}(0,1)$ random variables.

The Gram-Schmidt orthogonalization of $Z$ yields a decomposition $Z = QR$ where $Q \in \mathbb{O}_d$ and $R$ is an upper triangular $d$ by $d$ matrix with positive diagonal entries. We can use any $QR$ decomposition of $Z$ (Stewart, 1980), so long as the diagonal entries of $R$ are positive. Many computing environments contain $QR$ decompositions, but they don't all yield nonnegative diagonal entries for $R$. Let $S = S(R) = \mathrm{diag}(s_1, \dots, s_d)$ where $s_j = -1$ if $R_{jj} < 0$ and $S_j = 1$ otherwise. Then $Z = \widetilde{Q}\widetilde{R}$ where $\widetilde{Q} = QS$ and $\widetilde{R} = SR$ is a usable $QR$ decomposition of $Z$. This result leads us to a simple procedure for sampling uniform random rotations:

$$\begin{aligned}
&Z \sim \mathcal{N}(0, I_d, I_d)\\
&(Q, R) \leftarrow \mathrm{QRdecomp}(Z)\\
&\text{for } j = 1, \dots, d:\\
&\qquad s_j \leftarrow \begin{cases} 1, & R_{jj} \geqslant 0\\ -1, & \text{else}\end{cases}\\
&\widetilde{Q} \leftarrow \begin{pmatrix} s_1\boldsymbol{q}_1 & s_2\boldsymbol{q}_2 & \cdots & s_d\boldsymbol{q}_d\end{pmatrix} \sim \mathbf{U}(\mathbb{O}_d)
\end{aligned} \tag{5.28}$$

where $\boldsymbol{q}_j$ is the $j$'th column of $Q$ and QRdecomp$(\cdot)$ computes the QR decomposition of its argument. It is necessary to check the sign of $R_{jj}$ in (5.28) to get a proper uniform distribution. We should not get $R_{jj} = 0$, but even if that happens due to roundoff, we do not take $s_j = 0$.

If we need a uniform random projection of $n$-dimensional space onto $k$-dimensional space for $k \leqslant n$, then we need a matrix $P \in \mathbb{R}^{n\times k}$ where each column is $\mathbf{U}(S^{n-1})$ and those columns are orthogonal. The projection of $x \in \mathbb{R}^n$ is then $P^\mathsf{T}x$. The set of such matrices $P$ is the ***Stiefel manifold***

$$V_{n,k} = \{P \in \mathbb{R}^{n\times k} \mid P^\mathsf{T}P = I_k\}.$$

There is a uniform distribution on $V_{n,k}$. We can sample $Q \sim \mathbf{U}(\mathbb{O}_d)$ and then take the first $k$ columns of $Q$ to be $P \sim \mathbf{U}(V_{n,k})$. There is no need to generate all $n$ columns. We can simply take $Z \sim \mathcal{N}(0, I_n, I_k)$, solve $Z = QR$, multiply the $j$'th column of $Q$ by the sign of $R_{jj}$ as at (5.28) to get $\widetilde{Q} \in \mathbb{R}^{n\times k}$ and then deliver $P = \widetilde{Q}$.

**Example 5.13** (Random variance-covariance matrix)**.** A variance covariance matrix can be written $V = Q\Lambda Q^\mathsf{T}$, where $Q \in \mathbb{R}^{n\times n}$ is an orthogonal matrix and $\Lambda = \mathrm{diag}(\lambda_1, \dots, \lambda_n)$ for $\lambda_j \geqslant 0$. We can get a random variance-covariance matrix with fixed spectrum $\lambda_1 \geqslant \lambda_2 \geqslant \dots \geqslant \lambda_n \geqslant 0$ by taking $Q \sim \mathbf{U}(\mathbb{O}_d)$ and $V \leftarrow Q\Lambda Q^\mathsf{T}$.

There are also non-uniform matrix distributions on $\mathbb{O}_d$ and $V_{n,k}$. See page 51 of the chapter end notes.

## Wishart and inverse-Wishart matrices

The Wishart distribution plays a prominent role in multivariate analysis, and also in Bayesian statistics. Suppose that $\boldsymbol{X}_i \sim \mathcal{N}(0, \Sigma)$ are independent random vectors in $\mathbb{R}^d$ for $i = 1, \dots, n$. Let $W = \sum_{i=1}^n \boldsymbol{X}_i \boldsymbol{X}_i^\mathsf{T} \in \mathbb{R}^{d\times d}$. This sum of squares and cross-products matrix has the ***Wishart distribution*** $\mathcal{W}_d(\Sigma, n)$. The parameter $\Sigma$ is a $d\times d$ symmetric positive definite matrix, and the parameter $n$ is called the degrees of freedom. When $d = 1$, the Wishart distribution reduces to the $\sigma^2\chi^2_{(n)}$ distribution (with $1 \times 1$ matrix $\Sigma = (\sigma^2)$). There are other widely used notations for the Wishart distribution: the ordering and placement of the parameters $\Sigma$, $d$ and $n$ vary, and $d$ may be omitted because it can be deduced from $\Sigma$.

Just as the $\chi^2$ distribution can be defined for non-integer degrees of freedom, the $\mathcal{W}_d(\Sigma, \nu)$ distribution exists for non-integer degrees of freedom $\nu > d-1$. We will only consider Wishart distributions with nonsingular $\Sigma$. The probability density function of $W \sim \mathcal{W}_d(\Sigma, \nu)$ is

$$\frac{|W|^{(\nu-d-1)/2}}{c_{\nu,\Sigma}} \exp\Bigl(-\frac{1}{2}\mathrm{tr}(\Sigma^{-1}W)\Bigr), \quad \text{positive definite symmetric } W \in \mathbb{R}^{d\times d}.$$

A rough interpretation is as follows. We may interpret the determinant $|W|$ as a generalization of the absolute value. Similarly $\mathrm{tr}(AB)$ is a componentwise dot product of matrices $A$ and $B$. So the Wishart density is an exponentially downweighted power of $|W|$, yielding a matrix analogue of the Gamma distribution, as we might expect from its origin via Gaussian sums of squares and cross products. We will not use the normalizing constant, but for completeness it is

$$c_{\nu,\Sigma} = 2^{\nu d/2}\pi^{d(k-1)/4}|\Sigma|^{\nu/2}\prod_{j=1}^d \Gamma\Bigl(\frac{\nu+1-j}{2}\Bigr).$$

If $W \sim \mathcal{W}_d(\Sigma, \nu)$ then $\mathbb{E}(W) = \nu\Sigma$.

Usually sample sums of squares and cross-products are defined after subtracting out a mean vector. If $\boldsymbol{X}_i \sim \mathcal{N}(\mu, \Sigma)$ (independently) then

$$W = \sum_{i=1}^n (\boldsymbol{X}_i - \bar{\boldsymbol{X}})(\boldsymbol{X}_i - \bar{\boldsymbol{X}})^\mathsf{T} \sim \mathcal{W}_d(\Sigma, n-1),$$

for $\bar{\boldsymbol{X}} = (1/n)\sum_{i=1}^n \boldsymbol{X}_i$. The effect of centering is to reduce the degrees of freedom to $n-1$. This result gives us the distribution of the usual estimate of $\Sigma$,

$$\widehat{\Sigma} = \frac{W}{n-1} \sim \frac{\mathcal{W}_d(\Sigma, n-1)}{n-1}. \tag{5.29}$$

From the Wishart's defining property, we find that if $W \sim \mathcal{W}_d(\Lambda, \nu)$ and $C \in \mathbb{R}^{k\times d}$, then

$$CWC^\mathsf{T} \sim \mathcal{W}_k(C\Lambda C^\mathsf{T}, \nu). \tag{5.30}$$

Using (5.30) with $\Lambda = I$, we may interpret (5.29) as $\widehat{\Sigma} \sim C\mathcal{W}_d(I, n-1)C^\mathsf{T}/(n-1)$ where $CC^\mathsf{T} = \Sigma$, and so (5.30) reduces to the familiar result $s^2 \sim \sigma^2\chi^2_{(n-1)}/(n-1)$ when $d = 1$.

By applying (5.30) with $\Lambda = I$ we find that if $W \sim \mathcal{W}_d(I, \nu)$ and $C \in \mathbb{R}^{k\times d}$, then $CWC^\mathsf{T} \sim \mathcal{W}_k(CC^\mathsf{T}, \nu)$. As a result, we can sample from an arbitrary Wishart distribution $\mathcal{W}_d(\Sigma, \nu)$, using a matrix square root $C$ of $\Sigma$ (see §5.2), and a random $W \sim \mathcal{W}_d(I, \nu)$.

The Wishart can be generated directly from its definition, but that will be expensive if $\nu$ is a large integer, and impossible if $\nu$ is not an integer. If $\nu > d - 1$ then there is a shortcut, using a lower triangular matrix $T$ based on a decomposition due to Bartlett (1933). Let $T \in \mathbb{R}^{d\times d}$ have independent components as follows:

$$T_{ij} \sim \begin{cases} 0, & 1 \leqslant i < j \leqslant d \\ \mathcal{N}(0,1), & 1 \leqslant j < i \leqslant d \\ \sqrt{\chi^2_{(\nu-i+1)}}, & 1 \leqslant i = j \leqslant d. \end{cases} \tag{5.31}$$

Then $TT^\mathsf{T} \sim \mathcal{W}_d(I_d, \nu)$. As a result $W = (CT)(CT)^\mathsf{T} \sim \mathcal{W}_d(\Sigma, \nu)$ for $\Sigma = CC^\mathsf{T}$. For non-integer $\nu$, we use $\chi^2_{(\nu-i+1)} = 2\,\mathrm{Gam}((\nu - i + 1)/2)$.

Bayesian applications often use the inverse of a Wishart random matrix. If $\nu > d - 1$ and $\Sigma \in \mathbb{R}^{d\times d}$ is symmetric and positive definite then $W \sim \mathcal{W}_d(\Sigma, \nu)$ is invertible with probability one and then $W^{-1}$ has the ***inverse-Wishart distribution*** $\mathcal{IW}_d(\Sigma, \nu)$. We can sample $W \sim \mathcal{W}_d(\Sigma, \nu)$ and deliver $W^{-1}$. Or we can generate the matrix $T$ at (5.31), form $X = (CT)^{-1}$ where $CC^\mathsf{T} = \Sigma$, and deliver $W = X^\mathsf{T} X$.

The expected value of an inverse-Wishart matrix exists if $\nu - d - 1 > 0$ and then it is $\mathbb{E}(W) = \Sigma^{-1}/(\nu - d - 1)$. It is often convenient to write $W \sim \mathcal{IW}_d(\Lambda^{-1}, \nu)$ for $\Lambda = \Sigma^{-1}$ so that $\mathbb{E}(W) = \Lambda/(\nu - d - 1)$ and $W$ is no longer inversely related to the matrix parameter.

A Bayesian approach to Gaussian data $\boldsymbol{X}_i \sim \mathcal{N}(\mu, \Sigma)$, for $i = 1, \dots, n$ begins by placing a prior distribution on $\mu$ and $\Sigma$ and then finds the posterior distribution of $(\mu, \Sigma)$ given $\boldsymbol{X}_i$ for $i = 1, \dots, n$. The normal-inverse-Wishart distribution makes a very convenient prior distribution for $(\mu, \Sigma)$ because the posterior distribution of $(\mu, \Sigma)$ is also in that family. The statement is given in Proposition 5.1 below using the notation introduced next.

The pair $\mu \in \mathbb{R}^d$ and $\Sigma \in \mathbb{R}^{d\times d}$ (symmetric positive definite) follow the ***normal-inverse-Wishart*** distribution $(\mu, \Sigma) \sim \mathcal{NIW}_d(\mu_0, \kappa, \Lambda^{-1}, \nu)$ if $\Sigma \sim \mathcal{IW}_d(\Lambda^{-1}, \nu)$ and $\mu \mid \Sigma \sim \mathcal{N}(\mu_0, \Sigma/\kappa)$. The new parameter is $\kappa > 0$. It does not have to be an integer, but when it is, we may interpret $\mu$ (given $\Sigma$) as having the distribution of an average of $\kappa$ independent prior observations from $\mathcal{N}(\mu_0, \Sigma)$.

If $\boldsymbol{X}_1, \dots, \boldsymbol{X}_n \sim \mathcal{N}(\mu, \Sigma)$ and $(\mu, \Sigma)$ follow a normal-inverse-Wishart distribution, then the posterior distribution of $(\mu, \Sigma)$ given $\boldsymbol{X}_1, \dots, \boldsymbol{X}_n$ is also in the normal-inverse-Wishart family. The detailed description is in this proposition which parametrizes the variance using $\Lambda = \Sigma^{-1}$.

**Proposition 5.1.** *For integers $n \geqslant 1$ and $d \geqslant 1$, let $\Lambda_0$ be a $d$ by $d$ non-singular covariance matrix, $\mu_0 \in \mathbb{R}^d$, and $\nu_0, k_0 > 0$. If*

$$(\mu, \Sigma) \sim \mathcal{NIW}_d(\mu_0, \kappa_0, \Lambda_0^{-1}, \nu_0)$$

*and* $\boldsymbol{X}_1,\dots,\boldsymbol{X}_n \mid (\mu,\Sigma)$ *are independent from* $\mathcal{N}(\mu,\Sigma)$*, then*

$$(\mu,\Sigma)\mid(\boldsymbol{X}_1,\dots,\boldsymbol{X}_n)\sim\mathcal{NIW}_d\bigl(\mu_n,\kappa_n,\Lambda_n^{-1},\nu_n\bigr)$$

*where* $\nu_n=\nu_0+n$*,* $\kappa_n=\kappa_0+n$*,* $\mu_n=(\kappa_0\mu_0+n\bar{\boldsymbol{X}})/(\kappa_0+n)$*, and*

$$\Lambda_n=\Lambda_0+\sum_{i=1}^n(\boldsymbol{X}_i-\bar{\boldsymbol{X}})(\boldsymbol{X}_i-\bar{\boldsymbol{X}})^{\mathsf{T}}+\frac{\kappa_0 n}{\kappa_0+n}(\bar{\boldsymbol{X}}-\mu_0)(\bar{\boldsymbol{X}}-\mu_0)^{\mathsf{T}}.$$

*The marginal posterior distribution of* $\mu$ *is*

$$\mu\mid(\boldsymbol{X}_1,\dots,\boldsymbol{X}_n)\sim\mathrm{t}\bigl(\mu_n,\kappa_n^{-1}(\nu_n-d+1)^{-1}\Lambda_n,\nu_n-d+1\bigr).$$

*If* $\boldsymbol{X}_0$ *is another observation like* $\boldsymbol{X}_i$*, so* $\boldsymbol{X}_0,\boldsymbol{X}_1,\dots,\boldsymbol{X}_n\mid(\mu,\Sigma)\sim\mathcal{N}(\mu,\Sigma)$ *independently, then*

$$\boldsymbol{X}_0\mid(\boldsymbol{X}_1,\dots,\boldsymbol{X}_n)\sim\mathrm{t}\Bigl(\mu_n,\frac{\kappa_n+1}{\kappa_n(\nu_n-d+1)}\Lambda_n,\nu_n-d+1\Bigr).$$

*Proof.* These results are from Gelman et al. (2004, Chapter 3.6). □

Proposition 5.1 shows that we just add 1 to $\nu$ and $\kappa$ for each observation we get. We update $\mu$ by taking a weighted average of the prior mean $\mu_0$ and the sample mean $\bar{\boldsymbol{X}}$ where the relative weight on $\bar{\boldsymbol{X}}$ is proportional to $n$. It also reveals the main reason why inverse parameterization is useful for the variance: we get a simple additive update for $\Lambda=\Sigma^{-1}$. If we later get more observations $\boldsymbol{X}_{n+1},\dots,\boldsymbol{X}_{n+m}$ it is straightforward to incorporate them.

Choosing the parameters $\mu_0$, $\kappa_0$, $\Lambda_0$ and $\nu_0$ for the prior distribution can be a delicate task. If the prior $\mathcal{NIW}_d$ distribution has too much precision, such as very large $\kappa_0$ then the data will have only a little impact on the posterior distribution of $\mu$. Similarly if $\nu_0\gg d$ and $\Lambda_0/\nu_0$ is not very large the posterior distribution of $\Sigma$ will concentrate around $\Lambda_0/\nu_0$ with little impact from the data.

One choice that does not prespecify $\mu$ and $\Sigma$ with very high precision is the Jeffreys prior which in this setting is an improper distribution with density proportional to $|\Sigma|^{-(d+1)/2}$. It can be described formally as a limit of $\mathcal{NIW}_d(\mu_0,\kappa_0,\Lambda_0^{-1},\nu_0)$ as $\kappa_0\to0$, $\nu_0\to-1$ and $|\Lambda_0|\to0$. Though this limit cannot be normalized and $-1$ is not a valid prior degrees of freedom, the posterior distribution of $(\mu,\Sigma)$ given $\boldsymbol{X}_1,\dots,\boldsymbol{X}_n$ is proper if $n-1>d$, and then

$$\begin{aligned}\Sigma\mid(\boldsymbol{X}_1,\dots,\boldsymbol{X}_n)&\sim\mathcal{IW}_d(S^{-1},n-1)\quad\text{and}\\ \mu\mid(\Sigma,\boldsymbol{X}_1,\dots,\boldsymbol{X}_n)&\sim\mathcal{N}(\bar{\boldsymbol{X}},\Sigma/n),\end{aligned}$$

where $S=\sum_{i=1}^n(\boldsymbol{X}_i-\bar{\boldsymbol{X}})(\boldsymbol{X}_i-\bar{\boldsymbol{X}})^{\mathsf{T}}$ (Gelman et al., 2004, page 88).

## 5.9 Example: Classification error rates

In this section we use samples from the Wishart distribution to study Fisher's linear discriminant analysis for a famous data set of iris flower measurements.

The data have 4 size measurements on each of $n = 50$ flowers from each of three species of iris: setosa, versicolor, and virginica. The size measurements are sepal length, sepal width, petal length, and petal width, all in centimeters. For a single flower, let $\boldsymbol{X} \in \mathbb{R}^4$ contain the values (SL, SW, PL, PW), using the obvious abbreviations.

Iris setosa grows in northern coastal regions and it has very small petals. As a result, the setosa flowers are easily distinguished from the other two species. Iris virginica grows largely in inland regions to the south. Anderson (1936) presents the hypothesis that Iris versicolor is a hybrid of the other two species, whose ranges overlap, and that it is roughly 2/3 virginica and 1/3 setosa. The dimensions for virginica and versicolor have a tendency to overlap, and so telling them apart statistically is the challenge we consider, and we leave setosa aside.

Fisher's linear discriminant function is a linear combination of the features chosen to make the distributions for the two species as different as possible. For the virginica and versicolor flowers, the resulting function is

$$\begin{aligned} D(\boldsymbol{X}) &= -0.036 \times \mathrm{SL} - 0.056 \times \mathrm{SW} + 0.070 \times \mathrm{PL} + 0.124 \times \mathrm{PW} - 0.167 \\ &\equiv \boldsymbol{X}^{\mathsf{T}}\omega - \theta \end{aligned}$$

and the rule is to predict virginica if $D(\boldsymbol{X}) > 0$, and versicolor otherwise. This rule captures the feature that versicolor has smaller petals, compared to its sepals, than virginica. That is, it differs from virginica in the direction of its hypothesized other parent species, setosa. If we apply this rule to the same data that we used to build it, we find that one of the viriginica flowers is predicted to be versicolor and that two of the versicolor flowers are predicted to be virginica. In other words, the resubstitution error rates are 2% and 4% respectively.

The resubstitution error rate has several problems. It is biased towards optimistically small values, because the parameters in the discriminant function were chosen to minimize a closely related error measure. It has a large granularity, because with $n = 50$, the only possible error rates are integer multiples of 2%. Finally, it is just a single estimated value and does not come with an estimate of its own uncertainty.

If the dimensions for Iris virginica are well described by $\boldsymbol{X} \sim \mathcal{N}(\mu_{\mathrm{virg}}, \Sigma_{\mathrm{virg}})$ then we can get an error estimate via

$$\mathbb{P}\Big(D(\boldsymbol{X}) < 0 \mid \boldsymbol{X} \sim \mathcal{N}(\mu_{\mathrm{virg}}, \Sigma_{\mathrm{virg}})\Big) = \Phi\left(\frac{\mu_{\mathrm{virg}}^{\mathsf{T}}\omega - \theta}{\sqrt{\omega^{\mathsf{T}}\Sigma_{\mathrm{virg}}\omega}}\right). \tag{5.32}$$

Placing the usual sample estimates $\hat{\mu}_{\mathrm{virg}}$ and $\widehat{\Sigma}_{\mathrm{virg}}$ in equation (5.32) yields an estimated error rate of 3.49%. Finding $\mathbb{P}(D(\boldsymbol{X}) < 0)$ for $\boldsymbol{X} \sim \mathcal{N}(\hat{\mu}_{\mathrm{vers}}, \widehat{\Sigma}_{\mathrm{vers}})$ yields an estimated error rate of 2.46%, for versicolor flowers being classified as virginica. Because Fisher's discriminant function is derived assuming equal variances in the two groups, it would also have been reasonable to use pooled estimates of $\Sigma$ in this error estimate. By using separate variance estimates we are testing the model under broader assumptions than were used to construct it.

| True Species | Resubstitution Error | Estimated Error | Lower | Upper |
|---|---|---|---|---|
| Versicolor | 4% | 2.46% | 0.57% | 6.19% |
| Virginica | 2% | 3.49% | 0.99% | 8.07% |

Table 5.1: This table shows the resubstitution error, estimated error, and the limits of the 95% posterior credible interval for the error. In the first row, the errors are for versicolor wrongly labeled virginica. The second row is the reverse error.

These error rates only use the estimated mean and variance from within each species. Given a posterior distribution on $(\mu, \Sigma)$ for a species, we can construct a posterior distribution for the misclassification rate of that species. Taking a non-informative (Jeffreys) prior on $(\mu, \Sigma)$ leads to a posterior distribution with

$$\begin{aligned}\Sigma \mid (\boldsymbol{X}_1, \dots, \boldsymbol{X}_n) &\sim \frac{\mathcal{W}_4(\widehat{\Sigma}, n-1)}{n-1}, \quad \text{and}\\ \mu \mid (\Sigma, \boldsymbol{X}_1, \dots, \boldsymbol{X}_n) &\sim \mathcal{N}(\bar{\boldsymbol{X}}, \Sigma/n).\end{aligned}$$

The top panel in Figure 5.4 shows the error rate (5.32) under 10,000 samples from this posterior distribution. The bottom panel reverses the roles of the species.

The central 95% of each histogram in Figure 5.4 forms a posterior credible region for the corresponding misclassification rate. The upper and lower limits of these rates are given in Table 5.1. For example, for virginica, the resubstitution error estimate is 2%, but the error could reasonably be between 1% and 8% and indeed 3.49% may be a better estimate of the error than 2%.

Using the normal-Wishart model for the Iris data has removed the granularity issue and provided a measure of uncertainty for the estimated misclassification rates. Both of these gains hinge on the quality of a normal distribution for the flower dimensions and also on the prior model used.

It does not appear that the posterior histogram makes much of a correction for the optimism of the resubstitution error. The posterior distributions of $\mu$ and $\Sigma$ for each species are centered on the sample estimates, and the histogram of posterior rates has a mean very close to the error estimated from equation (5.32).

## 5.10 Random permutations

Random permutations are used in statistics to provide a quick and simple test of an hypothesis. They also play a role in some hashing schemes in computer science. We will use them to generate randomizations of quasi-Monte Carlo points in Chapter 17. Functions to generate uniform random permutations are widely available, but sometimes we have to write our own. Certain non-uniform permutations are also interesting and useful.

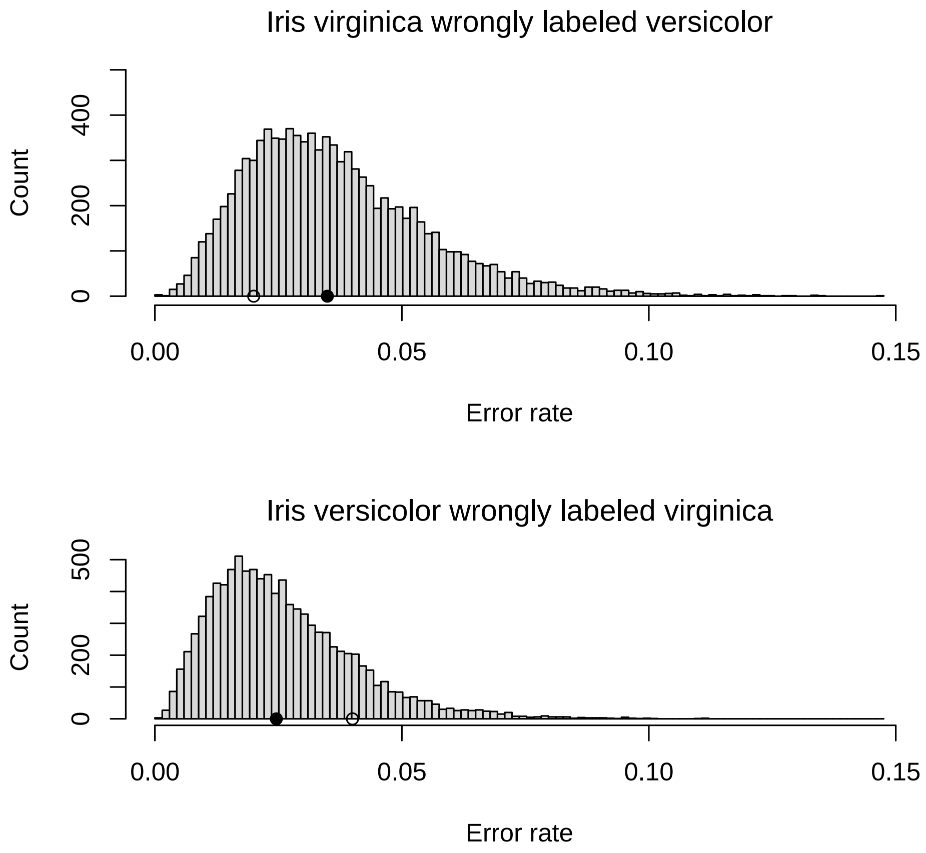

Figure 5.4: The top histogram shows 10,000 samples from the posterior distribution of the error rate for iris virginica, using a noninformative normal-inverse-Wishart prior distribution. The open reference point is the apparent error rate. The solid reference point is from the posterior mode. The bottom figure shows the same features for iris versicolor.

A permutation of elements $1,\dots,m$ is commonly written

$$\begin{pmatrix} 1 & 2 & \cdots & m \\ x_1 & x_2 & \cdots & x_m \end{pmatrix}$$

where $x_i \in \{1,\dots,m\}$ and $x_i \neq x_j$ when $i \neq j$. The idea is that $i$ gets mapped to $x_i$ under the permutation. For simplicity we represent the permutation by the vector $\boldsymbol{x} = (x_1,\dots,x_m)$. There are $m!$ permutations of $1,\dots,m$. The set of all such permutations is

$$S_m = \bigl\{(x_1,x_2,\dots,x_m) \in \{1,2,\dots,m\}^m \mid x_i \neq x_j \text{ for } 1 \leqslant i < j \leqslant m\bigr\}$$

which should not be confused with the symbol $S^d$ used for the sphere in $d+1$ dimensions.

**Algorithm 5.3** Uniform random permutation of $\{1,\dots,m\}$

```
$\boldsymbol{X} \leftarrow (1,2,\dots,m)$
for $j = m,\dots,2$ do
    $k \sim \mathbf{U}\{1,\dots,j\}$
    swap $X_j$ and $X_k$
deliver $\boldsymbol{X}$
```

**Definition 5.1** (Uniform random permutation)**.** If $\boldsymbol{X} \sim \mathbf{U}(S_m)$ then $\mathbb{P}(\boldsymbol{X} = \boldsymbol{x}) = 1/m!$ for all $\boldsymbol{x} \in S_m$.

When $m$ is very small, we can sample the $\mathbf{U}(s_m)$ distribution by enumerating its $m!$ elements and sampling the $i$'th one where $i \sim \mathbf{U}(\{1,\dots,m!\})$.

For larger $m$, a simple way to generate $\boldsymbol{X}$ is to first sample $\boldsymbol{U} \sim \mathbf{U}(0,1)^m$, sort the components $U_i$, getting $U_{(1)} \leqslant U_{(2)} \leqslant \dots \leqslant U_{(m)}$ and use the reordering from the sort as a random permutation. Here $(j)$ is the index of the $j$'th smallest of the $U_i$. By symmetry, the order vector $((1),(2),\dots,(m))$ is a uniformly distributed random permutation.

Suppose for illustration that $U_1 = 0.2$, $U_2 = 0.5$, $U_3 = 0.9$, $U_4 = 0.8$, and $U_5 = 0.3$. The order vector of the $U_i$ is $(3,1,5,2,4)$ because the smallest $U_i$ is the third one, the second smallest is the first one and so on.

Functions to find the order vector are commonly available, and so this procedure gives a simple one-liner, `order`(`random`($m$)) where `order` is a function that returns the order vector of its argument, and `random`($m$) delivers $\boldsymbol{U} \sim \mathbf{U}(0,1)^m$.

It is also possible to use the rank vector, but this is less attractive as described below. The $j$'th component of the rank vector of $\boldsymbol{U}$ is the rank of $U_j$ among $U_1,\dots,U_m$. The rank vector of the example $\boldsymbol{U}$ above is $(2,4,1,5,3)$ because $U_1$ is the second smallest value, $U_2$ is the fourth smallest value and so on. So `rank`(`random`($m$)) is an alternative one-liner where `rank` returns the rank vector.

The reason to prefer the order vector to the rank vector, is that there is a chance of ties among the components of $\boldsymbol{U}$. Intuitively that probability should be small. Then again, intuition suggests that a small gathering of people is unlikely to have two who share a birthday, a probability which is famously not so small. Exercise 5.15 applies the birthday problem to ties in random numbers.

When a vector has ties, the rank is sometimes given with fractions, such as `rank`$(0.1,0.1,0.2,0.3) = (1.5,1.5,3,4)$, which is not even a permutation and could crash a program. Perhaps worse, the answer might be interpreted as $(1,1,3,4)$ which is also not a permutation but might introduce a serious error without warning. Before using a function to compute the rank or order vector, it is worth testing what happens on tied values.

Methods based on sorting independent random variables cost $O(m\log(m))$ because of the sorting step. When $m$ is large we may want to reduce this cost, as well as make ties impossible. Algorithm 5.3 samples $\boldsymbol{X} \sim \mathbf{U}(S_d)$ in time $O(m)$ by swapping random elements.

To see that Algorithm 5.3 gives a uniform distribution on permutations, let

$\boldsymbol{Y} = (Y_1, \dots, Y_m)$ be any permutation of $1, \dots, m$. Then

$$\begin{aligned}\mathbb{P}(\boldsymbol{X} = \boldsymbol{Y}) &= \mathbb{P}(X_m = Y_m) \prod_{j=1}^{m-1} \mathbb{P}\bigl(X_j = Y_j \mid X_k = Y_k,\ j < k \leqslant m\bigr)\\ &= \frac{1}{m} \prod_{j=1}^{m-1} \frac{1}{j} = \frac{1}{m!},\end{aligned}$$

as required.

**Example 5.14** ($K$-fold cross-validation)**.** We can use uniform random permutations to randomly partition a set of $n$ data items into $K$ nearly equal subsets. This partitioning is often used in statistics and machine learning to implement $K$-fold cross-validation. In that process, a prediction algorithm is trained on $K-1$ of the subsets and evaluated on the other one, called the hold-out group. Each subset takes its turn as the held-out group. The choice $K = 10$ is common.

The partition can be represented by a vector $G$ of length $n$ with $G_i = k$ when observation $i$ is in group $k$. Let $1{:}K$ be the vector $(1, 2, \dots, K)$ and $\lceil n/K \rceil$ be the smallest integer greater than or equal to $n/K$. Then a simple way to generate the group indicator vector $G$ is as follows:

```
$v \leftarrow (1{:}K, 1{:}K, \cdots, 1{:}K)$      // concatenate $\lceil n/K \rceil$ copies of $1{:}K$,
$\pi \sim \mathbf{U}(S_n)$                       // uniform random permutation,
$G_i \leftarrow v_{\pi(i)}, \quad i = 1, \dots, n.$
```

Notice that $G$ ignores the last $K\lceil n/K \rceil - n$ entries of $v$. The resulting groups are of nearly equal size. Let $\bar{n} = \lfloor n/K \rfloor$ and $r = n - K\bar{n}$. Groups $r+1, \dots, K$ have $\bar{n}$ observations each. If $r > 0$ then groups $1, \dots, r$ have $\bar{n}+1$ observations.

**Example 5.15** (Min hashing)**.** Some methods for detecting duplicated (or nearly duplicated) documents are based on random permutations. Without giving all the details, suppose that two documents are represented as sets $A = \{A_1, A_2, \dots, A_r\} \in \mathcal{D}$ and $B = \{B_1, B_2, \dots, B_s\} \in \mathcal{D}$ respectively, where $\mathcal{D}$ is a set of possible document terms, such as words or $k$-tuples of letters. We will suppose that $\mathcal{D}$ is a set of $m$ integers representing the terms. The resemblance of $A$ and $B$ is defined to be

$$R = \frac{|A \cap B|}{|A \cup B|},$$

where $|\cdot|$ denotes cardinality.

It is fairly straightforward to compute the resemblance between two given documents. But if we have many millions of documents $A^{(1)}, A^{(2)}, \dots, A^{(N)}$ and want to identify all pairs with high resemblance then the computation becomes significant. A fast approximate answer is available via random permutations.

Consider again two documents $A$ and $B$ and suppose that $A_1 < A_2 < \cdots < A_r$ and $B_1 < B_2 < \cdots < B_s$. Let $\pi$ be a uniform random permutation of $\mathcal{D}$. Then it can be shown that

$$\mathbb{P}\bigl(\min(\pi(A_1), \dots, \pi(A_r)) = \min(\pi(B_1), \dots, \pi(B_s))\bigr) = R. \tag{5.33}$$

We can estimate $R$ by Monte Carlo. A modest number of random permutations suffices to tell the difference between large resemblances, such as $R \geqslant 0.9$, and small ones, $R \leqslant 0.5$.

The problem with min hashing is that it is very expensive to generate a uniform random permutation of $m$ elements for large values like $m = 2^{64}$ used in applications. In such cases we may be willing to compromise and use permutations that are almost but not quite uniformly distributed.

The following permutations are partially derandomized versions of the uniform random permutation. They are most naturally written as permutations of $\mathcal{D} = \{0, 1, 2, \dots, m-1\}$ for an integer $m \geqslant 2$.

**Definition 5.2.** For an integer $m \geqslant 2$, a ***random digital shift*** is a permutation $\boldsymbol{X} = (X_0, X_1, \dots, X_{m-1})$ of $\{0, 1, \dots, m-1\}$ given by $X_j = j + U \mathsf{~mod~} m$ for $j = 0, \dots, m-1$ where $U \sim \mathbf{U}\{0, 1, \dots, m-1\}$.

For the random digital shift,

$$\mathbb{P}(X_j = k) = \frac{1}{m}, \quad j, k \in \{0, 1, \dots, m-1\}.$$

That is, each component of $\boldsymbol{X}$ is uniformly distributed. See Exercise 5.16. A random shift is not very good for scrambling document contents. Consider the tiny document $A = (A_1, A_2, A_3)$ where $A_2 = A_1 + 1$ and $A_3 = A_1 + 2$. Under a digital shift $\mathbb{P}(\pi(A_2) < \min(\pi(A_1), \pi(A_3))) = 1/m$ not $1/3$ as we would get from a uniform permutation. We can do much better with a random linear permutation.

**Definition 5.3.** For a prime number $p$, a ***random linear permutation*** $\boldsymbol{X} = (X_0, X_1, \dots, X_{p-1})$ of $\{0, 1, \dots, p-1\}$ is given by $X_j = U + Vj \mathsf{~mod~} p$ for $j = 0, \dots, p-1$ where $U \sim \mathbf{U}\{0, 1, \dots, p-1\}$ and $V \sim \mathbf{U}\{1, 2, \dots, p-1\}$ are independent.

For the random linear permutation,

$$\mathbb{P}(X_j = k, X_{j'} = k') = \frac{1}{p(p-1)} \quad j, j', k, k' \in \{0, 1, \dots, p-1\} \quad j \neq j',\ k \neq k'.$$

That is, pairs $(X_j, X_{j'})$ have the same distribution under random linear permutations as under uniform random permutations. See Exercise 5.17.

It is crucial for $p$ to be a prime number in the random linear permutation. Otherwise the result is not even certain to be a permutation. Recall that a prime number is an integer $p \geqslant 2$ whose only positive integer divisors are 1 and $p$. As a simple consequence, for integers $a$ and $b$,

$$ab = 0 \mathsf{~mod~} p \iff (a = 0 \mathsf{~mod~} p \quad \text{or} \quad b = 0 \mathsf{~mod~} p). \tag{5.34}$$

Equation (5.34) does not hold for non-prime moduli (e.g., $2 \times 3 = 0 \mathsf{~mod~} 6$). Proposition 5.2 uses (5.34) to verify that random linear permutations really are permutations.

**Algorithm 5.4** Uniform random sample of $n$ items out of $N$

```
$\boldsymbol{X} \leftarrow (1, 2, \dots, N)$
for $j = N, \dots, N - n + 1$ do
  $k \sim \mathbf{U}\{1, \dots, j\}$
  swap $X_j$ and $X_k$
deliver $(X_{N-n+1}, \dots, X_{N-1}, X_N)$
```

**Proposition 5.2.** *The result of a random linear permutation really is a permutation.*

*Proof.* By the properties of modular arithmetic $X_j \in \{0, \dots, p-1\}$ for $j = 0, \dots, p-1$ whatever values $U$ and $V$ take. What remains is to check that no two components of $\boldsymbol{X}$ are equal. Then we are sure that each of the $p$ values must appear exactly once. Suppose that $X_j = X_k$ for $0 \leqslant j, k < p$. Then $U + Vj = U + Vk \mod p$. By subtraction, $V(j-k) = 0 \mod p$. Therefore by (5.34) either $V = 0 \mod p$ or $j - k = 0 \mod p$. The first possibility is prohibited by the definition. The second implies $j = k$. So $X_j = X_k$ implies $j = k$. As a result, $\boldsymbol{X}$ is a permutation. $\square$

For the duplicate detection problem, $m = 2^{64}$ is not a prime. We may however use a prime number $p > m$, with a small inefficiency because $p - m$ of the allowed elements will never appear in our documents.

Though they are better than digital shift permutations, random linear permutations do not satisfy (5.33). As a result, they yield somewhat biased estimates of resemblance. The bias is reported to be acceptably small. For more details, see Broder et al. (2000).

## 5.11 Sampling without replacement

We often have to choose a subsample of $n$ items from $N$. There are of course $\binom{N}{n}$ ways to do this. When all $\binom{N}{n}$ subsets are equally probable, we call it sampling without replacement. By contrast, ordinary sampling of $n$ independent draws from $\mathbf{U}\{1, \dots, N\}$ is called sampling with replacement. The terms refer to sampling balls from an urn where we might or might not place the most recently sampled ball back into the urn. The problem is harder without replacement.

We'll suppose that the items are just the integers $1, \dots, N$. Also we can assume that $n \leqslant N/2$. When $n > N/2$ then $n' = N - n \leqslant N/2$. In that case we choose the $n'$ elements not to be sampled and reverse the list.

When $N$ is not so large, the simplest way to proceed is to permute $N$ items and return the first $n$ of them. We don't have to actually generate the whole permutation. See Algorithm 5.4. When $n \ll N$ that algorithm saves most of the swapping, though it still takes $O(N)$ storage and even $O(N)$ time to initialize $\boldsymbol{X}$.

**Algorithm 5.5** Sequential sampling of $n$ items from $N$

```
$S \leftarrow \varnothing$            // item set so far
$m \leftarrow n$            // number of items needed
for $i = 1, \dots, N$ do
    $U \sim \mathbf{U}(0,1)$
    if $U \leqslant m/(N-i+1)$ then
        $S \leftarrow S \cup \{i\}$
        $m \leftarrow m-1$
deliver $S$
```

If we don't have storage for the vector $(1, 2, \dots, N)$ then we can sample $X_i \sim \mathbf{U}\{1, \dots, N\}$ independently and reject any duplicates as they arise. At the point where we have $i-1$ unique elements, the probability that the next one sampled is new is $(N-i+1)/N$. Thus we expect to make $N/(N-i+1)$ tries to get the $i$'th element. The expected number of samples is therefore $N\sum_{i=1}^{n}(N-i+1)^{-1}$. Devroye (1986, Theorem XII.2.1) shows that the expected number of samples is at most $2n$ when $n \leqslant \lceil N/2 \rceil$. If $n \ll N$ then the expected number of samples is close to $n$.

Checking for duplicates can require a careful choice for the data struture to store the already selected items. Devroye (1986, Chapter XII) offers many suggestions.

If we have a large file of $N$ records and want to sample $n$ of them, perhaps to test out a prediction algorithm on a subset, then a simple way to proceed is via sequential sampling. By symmetry, the first record in the file belongs to the sample with probability $n/N$. Similarly, if at some point in sampling we have $N'$ items left to look at and we need $n'$ more items to reach our goal, then the next record should be included in the sample with probability $n'/N'$. Algorithm 5.5 carries out such a sequential sampling.

Sometimes we need to sample $n$ records from a data stream of $N$ records where $N$ is unknown. Suppose for example that we want to sample $n$ visitors from among those who will visit a web site in a future time period. If we took a 1% sample we might get too many to store or too few to draw a strong conclusion, depending on $N$.

Reservoir sampling is a way to get such a sample. As each item streams by, we generate $U_i \sim \mathbf{U}(0,1)$ for it. The sample we take consists of those $n$ items with the smallest values of $U_i$.

It turns out that we don't really have to generate a $U_i$ for every record. After $n$ or more items have been sampled, let $W$ be the $n$'th smallest $U_i$ that we have seen. The chance that the next $U_i$ is below $W$ is $W$. The probability that the next record accepted is $s+1$ steps into the future is $W(1-W)^{s-1}$. In other words, the waiting time for the next sampled record has a geometric distribution. We can sample it via $s \sim 1 + \lfloor \log(U)/\log(1-W) \rfloor$ where $U \sim \mathbf{U}(0,1)$.

We also don't have to maintain a list of the $n$ values $U_i$ in order to decide which record to remove. When a new record enters the list, we can simply

**Algorithm 5.6** Reservoir sampling of $n$ items from a stream

```
$S_i \leftarrow i, \quad 1 \leqslant i \leqslant n$                                  // initial list of $n$ items
$LW \leftarrow \log(\mathbf{U}(0,1))/n$                                 // $\log(W)$
$s \leftarrow 1 + \lfloor \log(\mathbf{U}(0,1))/\log(1-\exp(LW)) \rfloor$   // step size
$i \leftarrow n + s$
while not at end of stream do
  $o \leftarrow \mathbf{U}\{1,\dots,n\}$                // index of element going out
  $S_o \leftarrow i$
  $LW \leftarrow LW + \log(\mathbf{U}(0,1))/n$
  $s \leftarrow 1 + \lfloor \log(\mathbf{U}(0,1))/\log(1-\exp(LW)) \rfloor$
  $i \leftarrow i + s$
deliver $S$
```

See comments on `log1p` and numerical issues in the text. This is known as Li's reservoir sampling algorithm. As written it runs until the stream stops. The current value of $S$ could also be queried at any time.

remove one of the old ones uniformly at random, by symmetry. Finally, had we kept the $n$ values $U_i$ they would have been independently and uniformly distributed on $[0, W]$. Therefore the new $W$ is distributed as the old $W$ times the maximum of $n$ independent $\mathbf{U}(0,1)$ random variables. Algorithm 5.6 takes account of these shortcuts to generate a sample of $n$ items from $N$ where $N$ is unknown.

As Algorithm 5.6 proceeds, $W$ gets closer to 0 and so $LW$ decreases. It is possible to find that $\log(1-\exp(LW))$ evaluates to 0 and then $s \leftarrow -\infty$. In double precision this may happen for $LW$ between $-37$ and $-38$. The increment $s$ is then comparable to $1/\log(10^{-37})$ or roughly $10^{16}$. It may take an absurdly large $N$ or a very unlucky $\log(\mathbf{U}(0,1))$ to trigger this problem, but it is worth checking that $s > -\infty$ at every iteration, because $s = -\infty$ will produce an infinite loop. Many computing environments contain a function `log1p` that computes $\log(1+p)$ directly and more accurately than applying the usual function `log` to $1+p$. Replacing the updates to $s$ by

$$s \leftarrow 1 + \lfloor \log(\mathbf{U}(0,1))/\texttt{log1p}(-\exp(LW)) \rfloor$$

improves the handling of these roundoff problems.

## 5.12 Random graphs

A graph is a set $\mathcal{V}$ of vertices or nodes, together with a set $\mathcal{E}$ of edges. Each edge connects two of the nodes in $\mathcal{V}$. When there are $n$ vertices we may take $\mathcal{V} = \{1, 2, \dots, n\}$. In a directed graph edges run from $i$ to $j$ where $i, j \in \mathcal{V}$ and are denoted $(i, j)$. In an undirected graph, the edge $(i, j)$ is considered to be the same as $(j, i)$. The edges in $\mathcal{E}$ can be represented by an $n \times n$ adjacency matrix $A$. For a directed graph, $A_{ij} = 1$ if $(i, j) \in \mathcal{E}$. For an undirected graph

$A_{ij} = A_{ji} = 1$ if $(i, j) \in \mathcal{E}$. We write $G = (\mathcal{V}, \mathcal{E})$ for the graph with given sets of vertices and edges.

Random graphs are used as models for real world networks such as social networks among people, citation networks among scholarly articles and interaction networks among proteins. Random graph models may be used to explain phenomena that we see in real data. Often those phenomena can be studied mathematically from the definition of the random graph, without any sampling. Simulation becomes necessary for phenomena that are out of reach of our mathematics perhaps because we have generalized the process generating the graph. Random graph models may also be used to generate an ensemble of graphs similar to one that was observed. For instance we might simulate graphs with the same degree sequence as one we have and then compare the number of triangles in the real graph to the simulated triangle count distribution. Additionally, when we have an algorithm designed to run on graphs then it is valuable to simulate graphs as test cases. For instance, an algorithm to estimate parameters of a model for random graphs should be tested on graphs sampled from that model.

Because a graph determines its adjacency matrix and vice versa, the problem of generating a random graph amounts to generating a random binary matrix $A \in \{0, 1\}^{n \times n}$. Sometimes random graphs contain loops, which are edges with the same vertex at each end. That is $A_{ii} = 1$. Similarly, randomly generated graphs often contain duplicated edges, making $\mathcal{E}$ a multiset and some $A_{ij} > 1$. Although loops and duplicate edges are usually undesirable properties, we may prefer algorithms that generate them, especially when they scale well to very large problems. We then either tolerate the loops and duplicates or delete them.

The first random graph model is the ***Erdős-Renyi random graph***. In such a graph there are $n$ nodes and an edge exists between any two distinct nodes $i$ and $j$ with probability $p$. When $n$ is small, it is easy to simulate an Erdős-Renyi random graph directly from its definition. For $1 \leqslant i < j \leqslant n$ we sample $U_{ij} \sim \mathbf{U}(0, 1)$ and create edge $(i, j)$ if and only if $U_{ij} < p$. The left panel of Figure 5.5 shows a small Erdős-Renyi graph. For large $n$ we may sample the number $m$ of edges via $m \sim \mathrm{Bin}(n(n-1)/2, p)$ and then select those $m$ edges by sampling without replacement from the set $\{(i, j) \mid 1 \leqslant i < j \leqslant n\}$. For example we might sample edges $(i, j)$ with $i, j \sim \mathbf{U}\{1, 2, \dots, n\}$, discarding loops and duplicated edges, and continuing until $m$ edges have been obtained.

The expected number of edges in an Erdős-Renyi graph is $n(n-1)p/2$. An alternative version of the Erdős-Renyi graph has exactly $m$ edges sampled without replacement from the $\binom{n}{2}$ possible edges. These two random graph distributions are quite similar when $n$ is large and $m = n(n-1)p/2$.

Because the Erdős-Renyi graph is so simply described it is amenable to mathematical analysis. It famously has a threshold effect. If $p = c/n$ with $c > 1$ then as $n \to \infty$ a giant connected component emerges. The fraction of nodes belonging to the giant component converges to a positive limit. Nodes within the giant component tend to be quite close to each other, separated by $O(\log(n))$ links. If $c < 1$ then the network consists of a large number of quite small disconnected subgraphs.

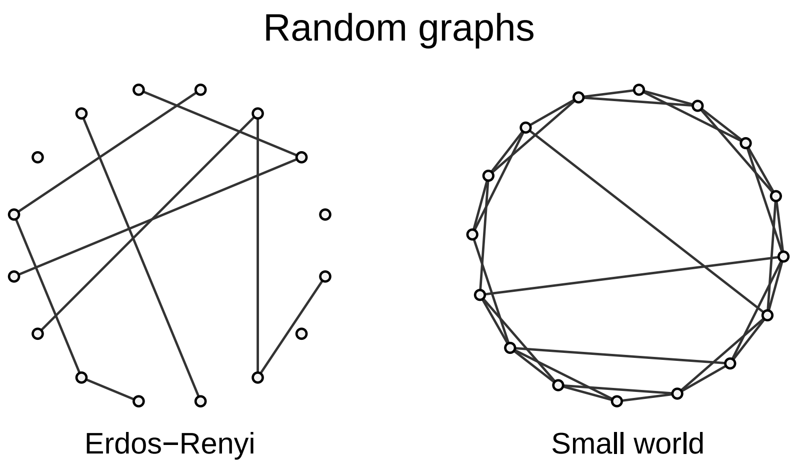

Figure 5.5: These random graphs have 16 nodes. The one on the left is an Erdős-Renyi graph with edge probability 0.1. The one on the right is a small world graph with rewiring probability 0.15.

The Erdős-Renyi graph is unrealistic in containing many fewer triangles than most real world networks with the same number of edges have. In a ***small world graph*** every node is initially connected to a small number of geographic near neighbors. They could be on a grid but are more usually placed around a circle. Then some of the edges are randomly rewired to connect to a random vertex. The right panel of Figure 5.5 shows a (small) small world graph. An alternative way to generate small world graphs simply adds some random edges without changing any of the local ones. Either way, the neighborhood structure leads to the presence of many triangles and a small number of random rewirings or insertions suffices to generate short average distances.

The degree of a vertex is the number of edges that contain it. For a directed graph there are in-degrees and out-degrees. In the Erdős-Renyi graph the degree of any single node has a distribution which approaches $\mathrm{Poi}(c)$ as $n \to \infty$. Real world networks usually have quite different degree distributions. Very often the degree distribution has a long tail. Small world graphs also have short-tailed distributions similar to the Erdős-Renyi graph.

The ***configuration model*** samples graphs in a way that preserves either a degree sequence or a degree distribution. The former is useful to compare a given graph to simulated alternatives. The latter generalizes to graphs of arbitrary size.

In the configuration model, we erase the edges in a graph leaving only $d_i$ stubs emanating from node $i$ that had degree $d_i$. These stubs are then randomly connected with each other to make a graph with the same degree sequence as the original. Figure 5.6 illustrates the process. This process might yield a defective graph, having self loops, duplicated edges or both.

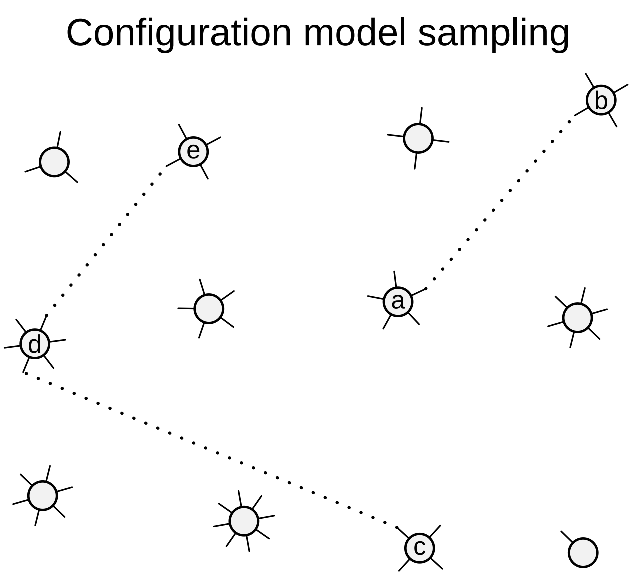

Figure 5.6: The edges of a graph are cut back to stubs emanating from each node. Pairs of stubs are then randomly connected, like $(a, b)$, $(c, d)$ and $(d, f)$ above until none remain.

Newman (2009, Section 13.2) gives a formula for the expected number of self loops in the configuration model: $(\bar{d^2} - \bar{d})/(2\bar{d})$, where $\bar{d} = (1/n)\sum_{i=1}^n d_i$ is the average degree and $\bar{d^2} = (1/n)\sum_{i=1}^n d_i^2$. Similarly the expected number of node pairs connected two or more times is $(\bar{d^2} - \bar{d})^2/(2\bar{d^2})$. These expected counts may well be small enough to neglect, although graphs with a very heavy-tailed degree distribution will be problematic.

When random degree values are used there is the possibility that the degrees sum to an odd number. We cannot pair off an odd number of stubs. We could add one stub to a randomly selected node to remove the problem. Or we could select one of the nodes to replace, for example the last one. If its degree $d_n$ is odd, then we can keep sampling $d_{n+1}, d_{n+2}, \dots$ from the degree distribution and replace $d_n$ by the first even value we draw. Similarly, if $d_n$ is even we replace it by the first odd value. This will work unless the degree distribution only supports odd values and $n$ is also odd (in which case the task is impossible).

If $\mathbb{E}(d^2) < \infty$ then the expected number of loops and self edges is $o(n)$ and it may then be reasonable to remove or ignore them. Durrett (2006) shows that these counts are asymptotically independent Poisson random variables in this case. There are however useful models with $\mathbb{E}(d^2) = \infty$. For example power laws with $\mathbb{P}(d = k) \propto (\beta + k)^{-\alpha}$ are often good descriptions of degree distributions (or just their tails) and they have infinite variance when $\alpha < 3$. Generating graphs with a prescribed degree sequence and no loops or double edges is an area of ongoing research (Blitzstein and Diaconis, 2011).

Configuration model sampling also extends to directed graphs. If vertex $i$ has $d_{i,\mathrm{out}}$ outgoing edges and $d_{i,\mathrm{in}}$ incoming edges we convert them all to stubs and then randomly connect outgoing stubs to incoming ones for the whole network. We can also sample from a degree distribution. It is necessary to enforce $\sum_{i=1}^n d_{i,\mathrm{in}} = \sum_{i=1}^n d_{i,\mathrm{out}}$ in order for the matching to be possible. A greater difficulty is deciding what bivariate degree distribution to sample from because, in particular, the appropriate copula is difficult to choose. For instance vertices with high degree might preferentially link to other such vertices, or alternatively, they might be unusually likely to link to vertices of low degree. Dyer (2010) gives graphical displays of those copulas for several large networks and they do not always resemble simple parametric forms.

The Erdős-Renyi and even the configuration models neglect some aspects of real networks. Bernoulli graphs offer much more flexibility. In a ***Bernoulli graph*** the $A_{ij}$ for $1 \leqslant i < j \leqslant n$ are independent random variables with arbitrary probabilities $P_{ij} = \mathbb{P}(A_{ij} = 1)$ and $A_{ji} = A_{ij}$. For instance, nodes are often found to belong to communities with many more edges between members of the same community and fewer cross-community edges. We can model community structure through

$$P_{ij} = \begin{cases} \rho_1, & i \text{ and } j \text{ in the same community} \\ \rho_0, & \text{else,} \end{cases}$$

for $\rho_1 \geqslant \rho_0$. More general ***stochastic block models*** split the set $\mathcal{V}$ of vertices into $k$ subsets $\mathcal{V}_r$ of cardinality $n_r$ for $r = 1, \dots, k$ and take $P_{ij} = \rho_{rs} = \rho_{sr}$ when $i \in \mathcal{V}_r$ and $j \in \mathcal{V}_s$. It is then straightforward to simulate such a graph. The number of edges from $\mathcal{V}_r$ to $\mathcal{V}_s$ has a binomial distribution, $\mathrm{Bin}(n_r n_s, \rho_{rs})$, for $r \neq s$. The number of edges within $\mathcal{V}_r$ (counting only vertices $i, j \in \mathcal{V}_r$ with $i < j$) is distributed as $\mathrm{Bin}(n_r(n_r - 1)/2, \rho_{rr})$. Once the numbers of edges are determined the actual edge locations can be sampled without replacement as in §5.11. Stochastic block models can be made much richer. The number and size of the communities can be made random as can the edge probabilities $\rho_{rs}$.

Even richer Bernoulli models take account of unequal activity levels for different vertices and use more subtle notions of similarity among vertices. They are most conveniently described using a monotone increasing squashing function $G : \mathbb{R} \to [0, 1]$. A particularly popular choice is the logistic function $G(z) = \exp(z)/(1 + \exp(z)) = 1/(1 + \exp(-z))$, though other cumulative distribution functions could be used instead.

To sample a Bernoulli random graph we have to draw $A_{ij} \sim P_{ij}$ independently for $1 \leqslant i < j \leqslant n$, for a given matrix $P$. Choosing which $P_{ij}$ to use is very difficult and problem dependent. There is not space to show all the choices, but here are a few examples to give a sense of what is done. A model that accounts for unequal activity levels has $P_{ij} = G(\mu + \alpha_i + \alpha_j + \rho_{rs})$ where $\alpha_i$ encodes the propensity of node $i$ to form links and $\rho_{rs}$ comes from the community structure. The expected total number of edges in the graph can be controlled by raising or lowering $\mu$. Even richer models take a vector $\boldsymbol{x}_i$ of features such as age, income, and interests of person $i$ in a for vertex $i$ and include $\sum_\ell \beta_\ell |x_{i\ell} - x_{j\ell}|$ inside $G(\cdot)$. When $\beta_\ell > 0$ edges are more likely to form between similar vertices while $\beta_\ell < 0$ favors edges between dissimilar vertices.

If $n$ is small enough, we may take $A_{ij} = 1_{U_{ij} \leqslant P_{ij}}$ for independent $U_{ij} \sim \mathbf{U}(0,1)$. If $n$ is very large we may not want to even form all of the $P_{ij}$. We can scale up to such problems if $P$ has a special form or if we are willing to make some approximations, or more likely, both. When $\max_{ij} P_{ij}$ is not too large we can sample $A_{ij} \sim \mathrm{Poi}(P_{ij})$. The advantage of the Poisson formulation is that we then know the distribution of sums. For example $\sum_{j=1}^n A_{ij} \sim \mathrm{Poi}(\sum_{j=1}^n P_{ij})$ and for $I, J \subset \{1, \dots, n\}$ we have $\sum_{i\in I}\sum_{j\in J} A_{ij} \sim \mathrm{Poi}(\sum_{i\in I}\sum_{j\in J} P_{ij})$. As a result we can first generate the degree of each vertex or the number of edges in a block and then recursively sample. When a sub-array of $A$ is known to sum to zero then we don't have to split it any further.

**Example 5.16** (Kronecker random graph)**.** A Kronecker random graph has a community of communities structure. The most simple version has parameters $r$ (a positive integer) and $M = \left(\begin{smallmatrix} a & b \\ b & c\end{smallmatrix}\right)$ where $a, b, c \in [0,1]$. Without loss of generality $a \geqslant c$ and ordinarily $c \geqslant b$. The $r$-fold Kronecker product of $M$ is the matrix

$$M^{(r)} = \begin{pmatrix} aM^{(r-1)} & bM^{(r-1)} \\ bM^{(r-1)} & cM^{(r-1)} \end{pmatrix} \in \mathbb{R}^{2^r \times 2^r}$$

starting with $M^{(1)} = M$. The Kronecker random graph is a Bernoulli random graph with $P = M^{(r)}$. When $a \geqslant c \geqslant b$, then the upper left corner of $M^{(r)}$ has larger values than any of the other corners. As a result the first $n/2$ vertices form a community with a concentration of internal edges. The last $n/2$ vertices form a second, somewhat weaker community, when $c < a$. Because $M^{(r-1)}$ also has this block structure we get communities within communities.

The expected number of edges in the Kronecker graph with parameter $r$ is half the sum of the off-diagonal entries in $M^{(r)}$. The sum of the entries in $M^{(r)}$ is $a + 2b + c$ times the sum of the entries in $M^{(r-1)}$ and so summing the entries in $M^{(r)}$ yields $(a + 2b + c)^r$. Similar considerations show that the diagonal of $M^{(r)}$ sums to $(a + c)^r$. As a result we find that the expected number of edges is $\mu(M, r) = ((a + 2b + c)^r - (a + c)^r)/2$.

For sparse graphs it is adequate to draw $N \sim \mathrm{Poi}(\mu(M, r))$ and then generate $N$ edges by sampling from the distribution of a single edge. That task is made slightly easier by taking the vertex set to be $\mathcal{V} = \{0, 1, \dots, n-1\}$ with indices increasing from top to bottom and left to right of $M^{(r)}$. Now we write indices $i$ and $j$ in base 2 as $i = \sum_{\ell=1}^r 2^{\ell-1} i_\ell$ and $j = \sum_{\ell=1}^r 2^{\ell-1} j_\ell$ where $i_\ell, j_\ell \in \{0, 1\}$. Given that a random edge has been added to the graph, the probability that it is in the upper left block is $a/(a + 2b + c)$ because that is the ratio of sums of matrix elements. As a result $\mathbb{P}(i_r = j_r = 0) = a/(a + 2b + c)$. Similarly $\mathbb{P}(i_r = j_r = 1) = c/(a + 2b + c)$ and $\mathbb{P}(i_r = 0, j_r = 1) = \mathbb{P}(i_r = 1, j_r = 0) = b/(a + 2b + c)$.

Whichever block of $M^{(r)}$ we find contains the edge there are four sub-blocks that contain the edge with probabilities in proportion to the entries in $M$. The index pairs $(i_\ell, j_\ell)$ are IID. So, to sample an edge we choose

$$(i_\ell, j_\ell) = \begin{cases} (0,0), & \text{with prob. } a/(a+2b+c) \\ (0,1), & \text{with prob. } b/(a+2b+c) \\ (1,0), & \text{with prob. } b/(a+2b+c) \\ (1,1), & \text{with prob. } c/(a+2b+c) \end{cases}$$

independently for $\ell = 1, \dots, r$ and then deliver $(i,j)$ where $i = \sum_{\ell=1}^{r} 2^{\ell-1} i_\ell$ and $j = \sum_{\ell=1}^{r} 2^{\ell-1} j_\ell$. For an undirected graph the presence of edge $(i,j)$ implies that of $(j,i)$. If our application requires edges $(i,j)$ to satisfy $i < j$ then we should deliver $(\min(i,j), \max(i,j))$. The Kronecker graph may include some self-loops and duplicated edges.

Bernoulli random graphs can be extended to directed graphs such as friendship graphs. There, the edge $(i,j)$ might indicate that person $i$ listed $j$ as a friend. If $(i,j) \in \mathcal{E}$ then it is more likely, though not certain, that $(j,i) \in \mathcal{E}$ too. We expect that $\mathbb{P}(A_{ij} = A_{ji} = 1) > \mathbb{P}(A_{ij} = 1)\mathbb{P}(A_{ji} = 1)$. The pair $(A_{ij}, A_{ji})$ is called a dyad. To extend Bernoulli random graphs to directed graphs, we generate dyads by independently sampling

$$(A_{ij}, A_{ji}) = \begin{cases} (0,0), & \text{with prob. } P_{ij00}, \\ (0,1), & \text{with prob. } P_{ij01}, \\ (1,0), & \text{with prob. } P_{ij10}, \\ (1,1), & \text{with prob. } P_{ij11}, \end{cases}$$

for $1 \leqslant i < j \leqslant n$. The model builder has to choose $P_{ij00}$ through $P_{ij11}$ in a way that encodes some hypothesis on the likelihood of each edge direction and the chance of reciprocation. In large graphs it is usual to have the vast majority of dyads be $(0,0)$. In that case we can sample the nonzero dyads with probability $P_{ij} = P_{ij01} + P_{ij10} + P_{ij11}$ and then choose the type of dyad with probabilities proportional to $P_{ij01}$, $P_{ij10}$ and $P_{ij11}$.

Bernoulli graphs and their generalizations to independent dyads, while flexible, still do not capture the kind of clustering seen in real graphs. Distinct vertices $i$, $j$, and $k$ form a triangle if $(i,j), (j,k), (k,i) \in \mathcal{E}$. In a Bernoulli graph, this triangle appears with probability $P_{ij}P_{jk}P_{ki}$, making triangles quite rare when the expected number of edges is small. Observed networks often contain more triangles than we would see under edge independence.

An ***exponential random graph*** model gives the probability of a graph in terms of a list of features $\phi_1(G)$, $\phi_2(G)$, …, $\phi_J(G)$ and parameters $\beta_1$ through $\beta_J$. We might choose $\phi_1$ to be the number of edges in $G$, $\phi_2$ to be the number of triangles and $\phi_j$ for $3 \leqslant j \leqslant J$ to be the number of $j-1$-stars in $G$. A $k$-star is a vertex connected to (at least) $k$ other vertices. The definitions are inclusive in that each triangle includes three 2-stars. If $\mathcal{G}$ is the set of graphs on $n$ vertices then the random graph $G$ has probability

$$p(G) = \frac{1}{Z} \exp\biggl( \sum_{j=1}^{J} \beta_j \phi_j(G) \biggr)$$

where $Z = \sum_{G\in\mathcal{G}} \exp(\beta^{\mathsf{T}}\phi(G))$ is a normalizing constant. Larger values of $\beta_j$ favor the appearance of more of the $j$'th feature type. Exponential graphs are not easily generated by direct methods. Instead Markov chain Monte Carlo methods (Chapter 11) are used. Some serious difficulties with exponential random graphs are described in the end notes.

The ***preferential attachment model*** is defined by a random process making it amenable to Monte Carlo. It may be used to model the growth of directed graphs such as those linking scientific articles to previous articles that they cite. We first describe the growth, then the initialization of such graphs. At time $t+1$ a new vertex is added to the set consisting already of $\{1, 2, \dots, t\}$. The new vertex is connected by $C = C_{t+1}$ new edges to the old vertices. In the most studied version of the model, $C = c$ with probability one, but $C$ can also be random. The $C$ new vertices point from $t+1$ to old vertices selected independently at random. The old vertex is taken to be $i$ with probability proportional to $a + d_{i,\mathrm{in}}$ for a parameter $a > 0$. Thus the probability of selecting old vertex $i$ is

$$p_i(t) = \frac{a + d_{i,\mathrm{in}}}{\sum_{i'=1}^{t} a + d_{i',\mathrm{in}}} = \frac{a + d_{i,\mathrm{in}}}{at + m}$$

where $m$ is the number of edges in the graph at time $t$. We write this as a mixture

$$p_i(t) = \frac{at}{at+m} \times \frac{1}{t} + \frac{m}{at+m} \times \frac{d_{i,\mathrm{in}}}{m}.$$

This mixture representation shows a way to sample the end-point of each new vertex. With probability $at/(at+m)$ pick vertex $T \sim \mathbf{U}\{1, \dots, t\}$ (uniformly on the existing vertices) and with the complementary probability $m/(at+m)$ pick edge $E$ uniformly from the $m$ existing edges and set $T$ to be the endpoint of edge $E$. Algorithm 5.7 describes the preferential attachment update starting from an input graph and continuing until the graph has at least $M$ edges. It keeps track of the growing graph using source and destination lists $S$ and $D$, where the $j$'th edge is $(S_j, D_j)$.

There are several choices to initialize the graph. It could be started with a single vertex having no incoming or outgoing edges. Then vertex 2 makes all of its links to this first vertex. The graph can also be started with a list of $t > 1$ vertices having no edges among them, or with a real graph such as a citation graph that we wish to generate forecasts for.

When the number $C$ of new links for each edge has finite variance then the preferential attachment process generates a graph whose degree distribution approaches a power law as $t \to \infty$ (see Newman (2009)). Specifically, the fraction of vertices with in-degree $d$ decays as $d^{-2-a/c}$ where $c = \mathbb{E}(C)$. The probability $at/(at+m)$ of choosing a random vertex converges to $a/(a+c)$ as $t \to \infty$.

Preferential attachment can also be used for undirected graphs. The new vertex $t+1$ is added to the graph along with $C$ new edges. The new edges connect vertex $t+1$ to old vertices sampled (independently) so that vertex $i$ is selected with probability proportional to $a + d_i$ where $d_i$ is the degree of vertex

**Algorithm 5.7** Preferential attachment for directed graphs

```
given $t \geqslant 1$, $M > 0$, $(S_i, D_i)$ $1 \leqslant i \leqslant m$, $F$, and $a > 0$
while $m < M$ do
  $C \sim F$                    // Number of new edges from vertex $t+1$
  for $j = 1, \dots, C$ do
    $S_{m+j} \leftarrow t+1$
    $U \sim \mathbf{U}(0,1)$
    if $U \leqslant at/(at+m)$ then
      $T \sim \mathbf{U}\{1, \dots, t\}$
      $D_{m+j} \leftarrow T$
    else
      $I \sim \mathbf{U}\{1, \dots, m\}$
      $D_{m+j} \leftarrow D_I$
  $m \leftarrow m + C$
  $t \leftarrow t + 1$
deliver $(t, m, S, D)$
```

This algorithm updates a directed graph by preferential attachment until there are at least $M$ edges. The input graph has $t$ vertices and $m$ edges $(S_i, D_i)$ with $S_i, D_i \in \{1, 2, \dots, t\}$. Each new vertex gets $C \sim F$ edges. It is ok to start with $m = 0$ and $t > 0$ unconnected vertices. When $m > 0$ we can allow $a = 0$.

$i$ and $a \geqslant 0$. Let $m$ be the number of edges among vertices $1, \dots, t$. Then we connect $t+1$ to $I \sim \mathbf{U}\{1, \dots, t\}$ with probability $at/(at+2m)$, while with probability $2m/(at+2m)$ we select an edge uniformly among the $m$ prior edges and choose $I$ randomly from that edge's two vertices, each with probability $1/2$.

## Chapter end notes

Devroye (1986) has a comprehensive treatment of sampling random vectors. Johnson et al. (1997) and Kotz et al. (2000) give encyclopedic coverage of discrete and continous multivariate distributions, respectively.

Sequential inversion is sometimes called the Hlawka-Mück method, after Hlawka and Mück (1972). Sequential inversion is the inverse of the Rosenblatt transform (Rosenblatt, 1952).

The shortcut for conditional Gaussian sampling appears in Doucet (2010) who illustrates it with an example from Kalman filtering. He attributes it to an astrophysics application in Hoffman and Ribak (1991).

The divide and conquer algorithms for the multinomial distribution are based on Fox (1999). Those algorithms mimic a Brownian bridge construction for Brownian motion that we will study in Chapter 6.

The uniform spacings method is a well-known property of uniform order statistics (David and Nagaraja, 2003). Because the uniform distribution on the simplex can also be generated via exponential random variables there are connections between uniform spacings and exponential spacings. Devroye (1986, Chapter V) presents many of those connections. In particular he presents ways to generate ordered exponential samples sequentially without sorting. For instance, if $Y_i \sim \mathrm{Exp}(1)$ independently then setting $X_{(0)} = 0$ and $X_{(i)} \leftarrow X_{(i-1)} + Y_i/(n-i+1)$ for $i = 1, \dots, n$, yields order statistics $X_{(1)}, \dots, X_{(n)}$ of $n$ independent Exp(1) random variables.

The bivariate exponential distribution is due to Marshall and Olkin (1967). They generalize it to a multivariate distribution in which up to $2^d - 1$ types of shock events exist, each of which terminates some non-empty subset of the $d$ items on test.

## Copulas

Nelsen (2006) is a comprehensive reference on copulas. The Gaussian copula method is widely used. A very early citation is Nataf (1962). The term NORTA is due to Cario and Nelson (1997) who study it in relation to their earlier ARTA (auto-regressive to anything) method for sampling time series (Cario and Nelson, 1996). Embrechts et al. (2003) survey copulas for finance and insurance. They point out that the $t$ copula does not have the tail independence property/flaw of the Gaussian copula. For more applications of copula modeling to finance and insurance see McNeil et al. (2005). A Gaussian copula model was proposed by Li (2000) for the joint distribution of time to default of multiple creditors. Some part of the financial crisis of 2008 has been attributed (Salmon, 2009) to traders who used this model.

Ghosh and Henderson (2003) look into the phenomenon in which a desired Pearson correlation matrix, in combination with uniform margins, is unattainable via the Gaussian copula. A well studied example from Li and Hammond (1975) has $d = 3$, $F_1 = F_2 = F_3 = \mathbf{U}(0, 1)$ and

$$\mathrm{Corr}(\boldsymbol{X}) = \begin{pmatrix} 1 & -0.4 & 0.2 \\ -0.4 & 1 & 0.8 \\ 0.2 & 0.8 & 1 \end{pmatrix}.$$

They show that Gaussian copula sampling cannot produce such a distribution. That such a distribution exists, and can be sampled by some other method, was proved by Ghosh and Henderson (2002). The failure to exist via the Gaussian copula is most interesting when the distribution exists.

## The hypersphere and random matrices

Muller (1959) proposed the use of normalized $\mathcal{N}(0, I)$ random vectors for sampling uniformly from the hypersphere in $d$ dimensions. There are ways to generate a random point on the hypersphere in $d$ dimensions, using only $d-1$ random variables. For large $d$ those methods are more complicated than using $\mathcal{N}(0, I)$ samples. These alternative methods have a mild advantage when equidistribution methods (like stratification and quasi-Monte Carlo) are employed, though this advantage diminishes as $d$ increases. See Fang and Wang (1994).

The matrix normal and many other matrix distributions are presented in detail by Gupta and Nagar (2000). Random orthogonal matrix sampling by Gram-Schmidt is due to Heiberger (1978), corrected by Tanner and Thisted (1982). The method had earlier been proposed by Wedderburn (1975) for rotation-based statistical testing (Langsrud, 2005), analogous to permutation tests, but Wedderburn died before that work could be published. Stewart (1980) considers QR algorithms based on the Householder transformation as a means to sample orthogonal matrices. Genz (1999) presents an approach based on butterfly transformations that is suitable for very large $d$, taking time $O(d^2\log(d))$ instead of $O(d^3)$.

Hoff (2009) presents sampling strategies for several Stiefel valued random matrices $X \in V_{n,k}$. The ***matrix von Mises-Fisher*** distribution has density $p_{\mathrm{MF}}(X;C) \propto \exp(\mathrm{tr}(C^{\mathsf{T}}X))$ for a parameter matrix $C \in V_{n,k}$. The trace of $C^{\mathsf{T}}X$ is the componentwise dot product of matrices $C$ and $X$, so $p_{\mathrm{MF}}$ reduces to the von Mises-Fisher distribution on $S^{d-1}$ when $n = d$ and $k = 1$. The ***matrix Bingham-von Mises-Fisher*** distribution Khatri and Mardia (1977) has density $p_{\mathrm{BMF}}(X;A,B,C) \propto \exp(\mathrm{tr}(C^{\mathsf{T}}X + BX^{\mathsf{T}}AX))$ where the additional parameters are symmetric matrices $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{k\times k}$. These matrices are usually taken to be symmetric. Hoff (2009) presents a Markov chain Monte Carlo sampler for $p_{\mathrm{BMF}}(X;A,B,C)$ and describes applications to Bayesian factor analysis, principal components analysis and a latent variable model for networks.

The Wishart distribution is described in detail by Muirhead (1982). The Bartlett decomposition (5.31) is the basis of Algorithm AS 53 of Smith and Hocking (1972) for sampling from the Wishart distribution. They cite some earlier algorithms. Muirhead (1982) presents the Bartlett decomposition for integer degrees of freedom and Smith and Hocking (1972) also assume integer $n$. One reference that explicitly includes the non-integer case is Ku and Bloomfield (2010). Jones (1985) points out the advantage of inverting the triangular Bartlett matrices rather than their product when sampling the inverse Wishart distribution.

## Discrete structures

There is a good description of sampling without replacement in Devroye (1986). The reservoir sampling Algorithm 5.6 is due to Li (1994). He calls it algorithm L. In numerical comparisons it is somewhat slower than three other more complicated algorithms (M, Z, and K).

The random permutation method is due to Durstenfeld (1964). It is often called the Fisher-Yates shuffle, in reference to a permutation method done by hand for randomizing experiments.

For $m \geqslant 1$, a random permutation $\pi$ of $\{1,\dots,m\}$ is ***min-wise independent*** if, for every nonempty set $X \subseteq \{1,\dots,m\}$ and every $x \in X$ we have

$$\mathbb{P}\bigl(\min\{\pi(X)\} = \pi(x)\bigr) = \frac{1}{|X|}.$$

Min-wise independent permutations were introduced by Broder et al. (2000). They have exactly the property we need to estimate resemblance via random permutations. A min-wise independent permutation can be obtained by drawing uniformly from a set of $4^m$ permutations. For large $m$ we have $4^m \ll m!$ and so this is a small subset of permutations.

## Random graphs

Newman (2009) discusses network models including several chapters on random graph models. See also Durrett (2006). Both emphasize provable properties of random graphs. Newman's account is very intuitive and describes the history and motivations of these models. Durrett's account is more rigorous. Kolaczyk (2009) describes network models with an emphasis on statistical inference.

The basic facts about the Erdős-Renyi random graphs were established in a series of papers: Erdős and Renyi (1959, 1960, 1961). The small world graph was introduced by Watts and Strogatz (1998). The configuration model was used by Molloy and Reed (1995) to identify degree sequences that give rise to a giant component. Kronecker random graphs were introduced by Leskovec et al. (2005).

Exponential random graphs are widely used in the study of social networks. For more information see Wasserman and Faust (1994) and Snijders et al. (2006). Their very interpretable parameters are highly desirable. There is a fundamental problem with them though, going well beyond the challenges in sampling them. Even a simple model with a parameter for edges and another for two-stars (Newman, 2009, Chapter 15.2) brings out the difficulty. If the two-star parameter value is high enough then the random graph will, with overwhelming probability, have either a great many edges or very few. When an observed network has many two-stars but not an extremely large or small number of edges then no distribution in this family supplies a good description.

The term 'preferential attachment' is due to Barabási and Albert (1999) who used it for undirected graphs with $a = 0$, that is pure degree-based attachment. The directed graph model was used by Price (1976) to study the growth of citations, drawing on work by Yule (1925) who modeled the number of plant species per genus.

Dyer (2010, Chapter 6) gives a bipartite preferential model for movie ratings data and fits it to the Netflix data (Bennett and Lanning, 2007). Each new edge to arrive links either a new or old movie to a new or old viewer. The distribution of these four edge types is time dependent over the multi-year data set. Edges connecting to existing movies do so with a distribution proportional to a power of their degree times a measure of the movie's age in the system. Similarly, edges connect to existing viewers in a time and degree dependent way. By far the strongest degree dependence was for edges connecting existing customers to new movies.

## Exercises

**5.1.** Here we use mixture methods to sample the planar density function in Example 5.1 and generalizations.

**a)** Find the probability density function $f_1(\boldsymbol{x})$ on $[0,1]^2$ for which $f_1(\boldsymbol{x}) \propto x_1$. Notice that $f_1$ does not depend on $x_2$. Give a formula $\boldsymbol{X} = \phi(\boldsymbol{U})$ that transforms $\boldsymbol{U} \sim \mathbf{U}(0,1)^2$ into $\boldsymbol{X} \sim f_1$.

**b)** Write the probability density $f(\boldsymbol{x}) = x_1 + x_2$ on $[0,1]^2$ as a mixture of densities $f_1(\boldsymbol{x}) \propto x_1$ and $f_2(\boldsymbol{x}) \propto x_2$.

**c)** Let

$$f(\boldsymbol{x}) \propto \begin{cases} c_0 + \sum_{j=1}^d c_j x_j, & \boldsymbol{x} \in [0,1]^d \\ 0, & \text{else,} \end{cases}$$

be a probability density function, where $c_j \geqslant 0$ and $\sum_{j=0}^d c_j > 0$. Write $f$ as a mixture of $d+1$ component densities. Be sure to specify the mixture probabilities and give normalized versions of the individual densities that you need.

**5.2.** The planar densities in Exercise 5.1 had nonnegative coefficients $c_j$. More general planar densities may include some negative components.

**a)** Describe how to sample from $f(\boldsymbol{x}) = 2 - x_1 - x_2$ on $[0,1]^2$.

**b)** Describe how to sample from $f(\boldsymbol{x}) = 1 - x_1 + x_2$ on $[0,1]^2$.

**c)** Describe how to sample from $f(\boldsymbol{x}) \propto c_0 + \sum_{j=1}^d c_j x_j$ on $[0,1]^d$. You may assume that $\sum_{j=0}^d |c_j| > 0$ and $c_0 \geqslant \sum_{j=1}^d \max(0, -c_j)$.

**5.3.** Let $f(\boldsymbol{x}) = \mathcal{N}(0, \Sigma)$ and suppose that we consider acceptance-rejection sampling with $g = \mathcal{N}(0, \sigma^2 I_d)$. Let $\Sigma = P\Lambda P^{\mathsf{T}}$ where $P$ is an orthogonal matrix and $\Lambda = \mathrm{diag}(\lambda_1, \dots, \lambda_d)$ and $0 < \lambda_j < \infty$. In order to have $f(\boldsymbol{x})/g(\boldsymbol{x})$ bounded it is necessary to have $\sigma \geqslant \max_j \sqrt{\lambda_j}$. Prove that

$$\sup_{\boldsymbol{x}} \frac{f(\boldsymbol{x})}{g(\boldsymbol{x})} \geqslant \prod_{j=1}^d \frac{\sigma}{\sqrt{\lambda_j}}.$$

**Note:** This exercise illustrates a point from §5.1 that acceptance-rejection sampling with mismatched covariances causes a problem that is likely to grow with $d$. For more background on the multivariate normal distribution see §5.2.

**5.4.** Let $\boldsymbol{X} \sim \mathrm{t}(\mu, \nu, \Sigma)$ for $\mu \in \mathbb{R}^d$, $\nu > 2$ and a positive definite matrix $\Sigma \in \mathbb{R}^{d \times d}$. Prove that $\mathrm{Var}(X_j) = \Sigma_{jj}\nu/(\nu - 2)$ and that $\mathrm{Corr}(X_j, X_k) = \Sigma_{jk}/\sqrt{\Sigma_{jj}\Sigma_{kk}}$.

**5.5.** Let $\boldsymbol{X} \sim \mathrm{Mult}(m, \boldsymbol{p})$ where $p_j \propto 1/j^2$, for $j = 1, \dots, d$. In this exercise we determine whether it is faster to generate $X_j$ in order $j = 1, 2, \dots, d$ by Algorithm 5.1, or in the reverse order $d, d-1, \dots, 2, 1$, by simply reversing the order of the components of $\boldsymbol{p}$.

**a)** Turn in a short piece of code that implements Algorithm 5.1 consistent with the notes below.

**b)** Compare the time taken to generate $\boldsymbol{X}$ in these two orders. Do not include any time taken to reverse the order of components in either $\boldsymbol{p}$ or $\boldsymbol{X}$. The computational efficiency for generating binomial random variables depends on the algorithm used. So be sure to name your source of binomial random variables, as well as the hardware and computing environment you run that code in.

Make your comparisons for the cases listed below:

| | | Case I | Case II | Case III | Case IV |
|---|---|---|---|---|---|
| Count parameter: | $m$ | 100 | $10^6$ | 100 | $10^6$ |
| Number of bins: | $d$ | 128 | 128 | $10^6$ | $10^6$ |

Describe when the order makes a difference, and in cases where the order matters, indicate which method has the advantage, and how large that advantage is.

**Notes:** Use a general purpose binomial routine, not one tuned to the specific probabilities that arise in this problem. Don't precompute $p_j/S$ for $j = 1, \dots, d$ either. But do stop sampling once $\ell = 0$ in Algorithm 5.1. These restrictions emulate what we would find in a setting with decreasing $p$ generated on the fly by some other algorithm where each vector $p$ was used only once. Of course to get accurate timings, you may have to repeat the sampling many times.

**5.6.** For this problem, implement Algorithm 5.2. Using the same setup as in Exercise 5.5 decide whether it is better to choose $r$ to split the vector length nearly equally, or to split the probability nearly equally. In the first case we take $r = \lfloor d/2 \rfloor$. In the second, we take $r = \max\{k \mid \sum_{j=1}^{k} p_j \leqslant 1/2\}$

**5.7** (Multinomial bakeoff)**.** This exercise is for classes where all students use the same software and hardware. Write a multinomial sampler that runs as fast as possible. The instructor will test it in the class computing environment on the following settings: $d = 10^6$, $p_j \propto q_j$ for $j = 1, \dots, d$ where $q_j \sim \mathbf{U}(j^{-2}, 2j^{-2})$ independently, and $m = 10^r$ where $r \sim \mathbf{U}\{1, 2, 3, 5, 6\}$ independently of the $q_j$.

**5.8.** Give an algorithm for sampling from the uniform distribution on an equilateral triangle. Generate and plot 100 points uniformly distributed in an equilateral triangle. Give an algorithm for sampling uniformly within the triangle $ABC$ with coordinates $A = (0, 0)$, $B = (0, 1)$ and $C = (4, 5)$. Generate and plot 100 such points.

**5.9.** For $d \geqslant 1$, $\mu \in \mathbb{R}^d$ a positive definite symmetric matrix $\Sigma \in \mathbb{R}^{d\times d}$ and $r > 0$, describe how to sample uniformly within the solid $\{\boldsymbol{x} \mid (\boldsymbol{x} - \mu)^{\mathsf{T}}\Sigma(\boldsymbol{X} - \mu) \leqslant r\}$.

**5.10** (Sampling a jelly roll)**.** Several problems in statistics and machine learning have the following flavor. We suppose that the observable data are scattered noisily around some low dimensional manifold (e.g., a surface or hypersurface), and then see if we can recover that manifold from the data. To test and compare the manifold recovery algorithms, we generate data scattered around a known manifold and then see how well the manifold is recovered. First we generate points inside the manifold, then we add the scatter. This problem looks at the first step.

A widely used test case has 3 dimensional data generated inside a 2 dimensional curved surface shaped like a Swiss roll, more precisely, the jelly inside the pastry of a Swiss roll.

One way to parameterize the points of the Swiss roll is

$$\begin{aligned} x(s,t) &= t\cos(t)\\ y(s,t) &= s\\ z(s,t) &= t\sin(t), \end{aligned}$$

for $0 \leqslant s \leqslant 10$ and $3\pi/2 \leqslant t \leqslant 9\pi/2$. If we generate points $(s,t)$ uniformly on the rectangle $[0,10]\times[3\pi/2, 9\pi/2]$ then the resulting points will not be uniform within the Swiss roll.

The problem comes from the way that $t$ is sampled. We need a nonuniform distribution on $t$. Let's ignore $s$ and $y$. Then $x(t)$ and $z(t)$ are simple functions of $t$ alone and we concentrate on choosing $t$ so that $(x(t), z(t))$ are uniformly distributed along the curve $\{(t\cos(t), t\sin(t)) \mid 3\pi/2 \leqslant t \leqslant 9\pi/2\}$.

**a)** Generate 20 equispaced points that span the domain of $t$ and plot the $(x(t), z(t))$ pairs. Where do you find them most concentrated and where are they sparsest?

**b)** Let $\dot{x} = (\mathrm{d}/\mathrm{d}t)x(t)$ and $\dot{z} = (\mathrm{d}/\mathrm{d}t)z(t)$. As $t$ changes to $t + \mathrm{d}t$, the curve $(x(t), z(t))$ moves a distance $\sqrt{\dot{x}^2 + \dot{z}^2}\,\mathrm{d}t$ in the $xz$–plane. Let $S(t) = \sqrt{\dot{x}^2 + \dot{z}^2}$ be the instantaneous speed of the curve. If we sample $t$ from a density function $f(t) \propto S(t)$, on $3\pi/2 \leqslant t \leqslant 9\pi/2$, then the resulting points $(x(t), z(t))$ will be uniformly distributed along the curve. The intuition is: when the curve is moving faster, we compensate by sampling more. Find an expression for $S(t)$.

**c)** Invent a way to sample points from $f$. Prove that your method works. That is, prove you get points $t$ sampled from $f$. You can assume that such points are then uniformly distributed along the curve.

**d)** Use your method to generate 20 independent random points from $f$. Carry this operation out 9 times and make a $3\times 3$ grid of plots of sampled points.

**e)** Using a sample of 10,000 points $t \sim f$ estimate the center of gravity $(\mathbb{E}(x(t)), \mathbb{E}(z(t)))$ of the Swiss roll jelly. Give a 99% confidence interval for each component separately.

**f)** I'm now going to slice the Swiss roll using the vertical plane $\{(x,y,z) \mid x = 0, (y,z) \in \mathbb{R}^2\}$. Estimate the fraction of jelly that is on the $x < 0$ side and give a 99% confidence interval.

This problem is based on a suggestion by Patrick Perry.

**5.11.** Consider $\boldsymbol{X} = (X_1, X_2) \in (0,\infty) \times (0,1)$ with $X_1 \sim \mathrm{Exp}(1)$ and $X_2 \sim \mathbf{U}(0,1)$. Show how to sample $\boldsymbol{X}$ so that the rank correlation coefficient between $X_1$ and $X_2$ is 1. What then is the Pearson correlation coefficient? Similarly, show how to sample $\boldsymbol{X}$ so that the rank correlation coefficient between $X_1$ and $X_2$ is $-1$ and find the resulting Pearson correlation coefficient.

**5.12.** If $X_1 \sim \mathrm{Gam}(\alpha_1)$ independently of $X_2 \sim \mathrm{Gam}(\alpha_2)$ then $X_1 + X_2 \sim \mathrm{Gam}(\alpha_1 + \alpha_2)$ is independent of $X_1/(X_1 + X_2) \sim \mathrm{Beta}(\alpha_1, \alpha_2)$. Use this fact to devise a divide and conquer algorithm to generate $X_1 \sim \mathrm{Gam}(\alpha_1)$ through $X_{256} \sim \mathrm{Gam}(\alpha_{256})$ independently of each other. [Note: at this point it is not clear why we would even want to use divide and conquer for this problem. It could be slower than straightforward generation. Chapter 8 describes some sampling techniques that take advantage of divide and conquer strategies.]

**5.13.** Here we generate points from a distribution with Gaussian margins but a Marshall-Olkin copula. Let $\boldsymbol{X} \in [0,\infty)^2$ have the Marshall-Olkin bivariate exponential distribution (5.17) with parameters $\lambda_1 = \lambda_2 = 1$ and $\lambda_3 = 1/2$.

**a)** Generate and plot 1000 points $\boldsymbol{Y} \in \mathbb{R}^2$ having the same copula as $\boldsymbol{X}$ but $\mathcal{N}(0,1)$ margins.

**b)** Make QQ plots of your points from part **a** to check that the margins are Gaussian. Call the points $Y_{ij}$ for $1 \leqslant i \leqslant n$ and $j = 1,2$. Let the sorted points be $Y_{(i)j}$ where $Y_{(1)j} \leqslant Y_{(2)j} \leqslant \dots \leqslant Y_{(n)j}$ are reordered values of $Y_{1j}, \dots, Y_{nj}$ for $j = 1,2$. Then plot $Y_{(i)j}$ versus $\Phi^{-1}((i - 1/2)/n)$. The result should be two curves (for $j = 1,2$) both of which are close to the 45 degree line with a few fluctuations in the tails.

**c)** Repeat part **a** with $\lambda_3 = 10$ and then again with $\lambda_3 = 100$.

**d)** Let $\rho = \mathrm{Corr}(Y_1, Y_2)$ for the distribution in part **a**. Sample $n = 10^4$ points from that distribution and report the sample correlation $\hat\rho$.

**e)** Repeat part **d** 100 times and use $\hat\rho_1, \dots, \hat\rho_{100}$ to make a 99% confidence interval for $\mathbb{E}(\hat\rho)$.

[This is almost a 99% central limit theorem confidence interval for $\rho$. There is a slight discrepancy because $\mathbb{E}(\hat\rho)$ has a small bias as an estimate of $\rho$.]

**Note:** In practice we might face the reverse problem of choosing $\lambda_3$ to get a desired correlation $\rho$. See Exercise 8.27.

**5.14.** Describe how to sample $\boldsymbol{X} \in \mathbb{R}^d$ with a probability density function $f(\boldsymbol{x}) \propto \|\boldsymbol{x}\|^k e^{-\|\boldsymbol{x}\|}$ for $k > 0$. Are there any $k < 0$ for which your approach works?

**5.15** (Ties and random permutations). Suppose that $U_i$ are independent and uniformly distributed over $2^{32}$ values in $[0,1]$.

**a)** For $m \geqslant 2$, what is the probability that two or more of $U_1, \dots, U_m$ are equal?

**b)** Suppose that $n = 100{,}000$ random vectors $\boldsymbol{U}_i \in [0,1]^m$ are generated, for $m = 256$. What is the probability that we never see $\boldsymbol{U}_{ij} = \boldsymbol{U}_{ik}$ for $j \neq k$?

**5.16** (Random digital shift)**.** For the random digital shift, prove that $\mathbb{P}(X_j = k) = 1/m$ for all $j, k \in \{0, 1, \dots, b-1\}$.

**5.17** (Random linear permutation)**.** For the random linear permutation, give an example to show that the result is not always a permutation when $m$ is not prime. When $m$ is a prime number, prove that

$$\mathbb{P}(X_j = k, X_{j'} = k') = \frac{1}{m(m-1)}$$

when $j$ and $j'$ are two distinct values in $\{0, \dots, m-1\}$ and $k$ and $k'$ are also two distinct values in $\{0, \dots, m-1\}$.

**5.18** (Aitchison distribution)**.** The Aitchison distribution on $S^{d-1}$ has probability density function

$$f(\boldsymbol{x}; \alpha, \Sigma) \propto \prod_{j=1}^{d} x_j^{\alpha_j - 1} \exp\Bigl(-\frac{1}{2}\sum_{j=1}^{d-1}\sum_{k=j+1}^{d} \Sigma_{jk}\bigl(\log(x_j) - \log(x_k)\bigr)^2\Bigr),$$

for positive semi-definite $\Sigma \in \mathbb{R}^{d\times d}$ and $\alpha \in (0,\infty)^d$.

**a)** Show that it contains the Dirichlet distribution as a special case.

**b)** Show that it contains the logistic-normal distribution as a special case.

**c)** Devise a method of sampling from this distribution.

**5.19** (Preferential attachment I)**.** In this exercise you simulate a directed graph by preferential attachment until there are at least 10,000 vertices. The graph starts with $t = 10$ vertices among which there are $m = 0$ prior edges. Let the number of edges for each new vertex be $C = 1 + \widetilde{C}$ where $\widetilde{C} \sim \mathrm{Poi}(5)$. Take $a = 1$.

**a)** Graphically compare the observed in-degree distribution of your graph to the predicted power law (page 48). Include 3 repeats of the sampling.

**b)** In each repeat, report the fraction of the edges that connect to members of the original 10 vertices. Do those 10 vertices get nearly equal numbers of links?

**c)** If you remove those 10 vertices from the graph, do you still see a power law?

To depict data following a power law, plot the $k$'th largest value versus $k$, both on a log scale. Power law data will tend to follow a linear trend. Perfect power laws are rare. Data often show exceptions at the extremes.

**5.20** (Preferential attachment II)**.** Recent articles are more likely to be cited than older articles. The magnitude of this phenomenon can be different in each scientific literature. Define a recent vertex to be one with $\max(10, t-50) < i \leqslant t$ at the time vertex $t+1$ arrives. These are the 50 most recent vertices except that the founding vertices are not included. Simulate the model of Exercise 5.19 under each of the following changed rules.

**a)** Edge $i$ is selected with probability proportional to $10 + d_i$ if $i$ is recent and $1 + d_i$ otherwise.

**b)** Edge $i$ is selected with probability proportional to $1 + 10d_i$ if $i$ is recent and $1 + d_i$ otherwise.

**c)** Edge $i$ is selected with probability proportional to $10 + 10d_i$ if $i$ is recent and $1 + d_i$ otherwise.

For each changed rule show three simulated in-degree distributions excluding the founding articles. Describe how you sampled the

# Bibliography

Anderson, E. (1936). The species problem in Iris. *Annals of the Missouri Botanical Garden*, 23(3):457–509.

Barabási, A.-L. and Albert, R. (1999). Emergence of scaling in random networks. *Science*, 286(5439):509–512.

Bartlett, M. S. (1933). On the theory of statistical regression. *Proceedings of the Royal Society of Edinburgh*, 53:260–283.

Bennett, J. and Lanning, S. (2007). The Netflix prize. In *Proceedings of KDD cup and workshop*, volume 2007, page 35.

Blitzstein, J. and Diaconis, P. (2011). A sequential importance sampling algorithm for generating random graphs with prescribed degrees. *Internet Mathematics*, 6(4):489–522.

Broder, A., Charikar, M., Frieze, A. M., and Mitzenmacher, M. (2000). Min-wise independent permutations. *Journal of Computer and System Sciences*, 60(3):630–659.

Cario, M. C. and Nelson, B. L. (1996). Autoregressive to anything: time series input processes for simulation. *Operations Research Letters*, 19:51–58.

Cario, M. C. and Nelson, B. L. (1997). Modeling and generating random vectors with arbitrary marginal distributions and correlation matrix. Technical report, Northwestern University.

David, H. A. and Nagaraja, H. N. (2003). *Order Statistics*. Wiley, Hoboken, NJ, 3rd edition.

Devroye, L. (1986). *Non-uniform Random Variate Generation*. Springer, New York.

Doucet, A. (2010). A note on efficient conditional simulation of Gaussian distributions. Technical report, University of British Columbia.

Durrett, R. (2006). *Random Graph Dynamics.* Cambridge University Press, Cambridge.

Durstenfeld, R. (1964). Algorithm 235: Random permutation. *Communications of the ACM*, 7(7):420.

Dyer, J. S. (2010). *Visualizing and modeling joint behavior of categorical variables with a large number of levels.* PhD thesis, Stanford University.

Embrechts, P., Lindskog, F., and McNeil, A. (2003). Modelling dependence with copulas and applications to risk management. In Rachev, S. T., editor, *Handbook of Heavy Tailed Distributions in Finance*, chapter 8, pages 329–384. Elsevier, Amsterdam.

Erdős, P. and Renyi, A. (1959). On random graphs. *Publicationes Matematicae*, 6:290–297.

Erdős, P. and Renyi, A. (1960). On the evolution of random graphs. *Publications of the Mathematical Institute of the Hungarian Academy of Sciences*, 5:17–61.

Erdős, P. and Renyi, A. (1961). On the strength of connectedness of a random graph. *Acta Mathematica Hungarica*, 12(1):261–267.

Fang, K.-T. and Wang, Y. (1994). *Number Theoretic Methods in Statistics.* Chapman & Hall.

Fox, B. L. (1999). *Strategies for quasi-Monte Carlo.* Kluwer Academic, Norwell, MA.

Gelman, A., Carlin, J. B., Stern, H. S., and Rubin, D. B. (2004). *Bayesian Data Analysis.* Chapman & Hall/CRC, Boca Raton, FL, 2nd edition.

Genz, A. (1999). Methods for generating random orthogonal matrices. In Niederreiter, H. and Spanier, J., editors, *Monte Carlo and quasi-Monte Carlo Methods 1998*, pages 199–213, Berlin. Springer-Verlag.

Ghosh, S. and Henderson, S. G. (2002). Chessboard distributions and random vectors with specified marginals and covariance matrix. *Operations Research*, 50(5):820–834.

Ghosh, S. and Henderson, S. G. (2003). Behavior of the NORTA method for correlated random vector generation as the dimension increases. *ACM Transactions on Modeling and Computer Simulation*, 13(3):276–294.

Gupta, A. J. and Nagar, D. K. (2000). *Matrix variate distributions.* CRC Press, Boca Raton, FL.

Heiberger, R. M. (1978). Generation of random orthogonal matrices. *Journal of the Royal Statistical Society, Series C*, 27(2):199–206.

Hlawka, E. and Mück, R. (1972). Über eine Transformation von gleichverteilten Folgen. II. *Computing*, 9:127–138.

Hoff, P. D. (2009). Simulation of the matrix Bingham-von Mises-Fisher distribution, with applications to multivariate and relational data. *Journal of computational and graphical statistics*, 18(2):438–456.

Hoffman, Y. and Ribak, E. (1991). Constrained realizations of Gaussian fields – a simple algorithm. *The Astrophysical Journal*, 380:L5–L8.

Johnson, N. L., Kotz, S., and Balakrishnan, N. (1997). *Discrete Multivariate Distributions*. Wiley, New York.

Jones, M. C. (1985). Generating inverse Wishart matrices. *Communications in Statistics: Simulation and Computation*, 14(2):511–514.

Khatri, C. G. and Mardia, K. V. (1977). The von Mises-Fisher matrix distribution in orientation statistics. *Journal of the Royal Statistical Society, Series B*, 39(1):95–106.

Kolaczyk, E. D. (2009). *Statistical analysis of network data*. Springer, New York.

Kotz, S., Balakrishnan, N., and Johnson, N. L. (2000). *Continuous Multivariate Distributions, volume 1, Models and Applications*. Wiley, New York, 2nd edition.

Ku, Y.-C. and Bloomfield, P. (2010). Generating random Wishart matrices with fractional degrees of freedom in OX. Technical report, North Carolina State University.

Langsrud, O. (2005). Rotation tests. *Statistics and computing*, 15(1):53–60.

Leskovec, J., Chakrabarti, D., Kleinberg, J., and Faloutsos, C. (2005). Realistic, mathematically tractable graph generation and evolution, using Kronecker multiplication. In *Knowledge Discovery in Databases: PKDD 2005*, pages 133–145. Springer.

Li, D. X. (2000). On default correlation: A copula function approach. *Journal of fixed income*, 9(4):43–54.

Li, K.-H. (1994). Reservoir-sampling algorithms of time complexity $o(n(1 + \log(n/n)))$. *ACM Transactions on Mathematical Software*, 20(4):481–493.

Li, S. T. and Hammond, J. L. (1975). Generation of pseudorandom numbers with specified univariate distributions and correlation coefficients. *IEEE Transactions on Systems, Man, Cybernetics*, 5(5):557–561.

Marshall, A. W. and Olkin, I. (1967). A multivariate exponential distribution. *Journal of the American Statistical Association*, 62(317):30–44.

McNeil, A. J., Frey, R., and Embrechts, P. (2005). *Quantitative risk management: concepts, techniques and tools*. Princeton University Press, Princeton, NJ.

Molloy, M. and Reed, B. (1995). A critical point for random graphs with a given degree sequence. *Random structures & algorithms*, 6(2–3):161–180.

Muirhead, R. J. (1982). *Aspects of Multivariate Statistical Theory*. Wiley, New York.

Muller, M. E. (1959). A note on a method for generating points uniformly on $n$-dimensional spheres. *Communications of the ACM*, 2(4):19–20.

Nataf, A. (1962). Détermination des distributions de probabilities dont les marges sont données. *Comptes Rendus de l'Academie des Sciences, Paris*, 225:42–43.

Nelsen, R. B. (2006). *An introduction to copulas*. Springer, New York.

Newman, M. (2009). *Networks: an introduction*. Oxford University Press, Oxford.

Price, D. (1976). A general theory of bibliometric and other cumulative advantage processes. *Journal of the American Society for Information Science*, 27(5):292–306.

Rosenblatt, M. (1952). Remarks on a multivariate transformation. *Annals of Mathematical Statistics*, 23(3):470–472.

Salmon, F. (2009). Recipe for disaster: The formula that killed Wall Street. *Wired Magazine*, 17(3).

Sklar, A. (1959). Fonctions de répartition á $n$ dimension et leurs marges. *Publications de l'institut d Statistique de l'Université de Paris 8*, pages 229–231.

Smith, W. B. and Hocking, R. R. (1972). Algorithm AS 53: Wishart variate generator. *Journal of the Royal Statistical Society, Series C (Applied Statistics)*, 21(3):341–345.

Snijders, T. A. B., Pattison, P. E., Robins, G. L., and Handcock, M. S. (2006). New specifications for exponential random graph models. *Sociological Methodology*, 36(1):99–153.

Stewart, G. W. (1980). The efficient generation of random orthogonal matrices with an application to condition numbers. *SIAM Journal of Numerical Analysis*, 17(3):403–409.

Tanner, M. A. and Thisted, R. A. (1982). Remark AS R42: a remark on AS 127. Generation of random orthogonal matrices. *Journal of the Royal Statistical Society, Series C (Applied Statistics)*, 31(2):190–192.

Trefethen, L. N. and Bau, III, D. (1997). *Numerical linear algebra.* SIAM, Philadelphia, PA.

Ulrich, G. (1984). Computer generation of distributions on the $m$-sphere. *Applied Statistics*, 33(2):158–163.

Wasserman, S. and Faust, K. (1994). *Social network analysis: Methods and applications*, volume 8. Cambridge University Press, Cambridge.

Watts, D. J. and Strogatz, S. H. (1998). Collective dynamics of small-worldnetworks. *Nature*, 393(6684):440–442.

Wedderburn, R. W. M. (1975). Random rotations and multivariate normal simulation. Technical report, Rothamsted Experimental Station.

Wood, A. T. A. (1987). The simulation of spherical distributions in the Fisher-Bingham family. *Communications in Statistics: Simulation and Computation*, 16(3):885–898.

Yule, G. U. (1925). A mathematical theory of evolution, based on the conclusions of Dr. J. C. Willis, FRS. *Philosophical Transactions of the Royal Society of London. Series B. Biological Sciences*, 213:21–87.