# Fast Simulation of Hyperplane-Truncated Multivariate Normal Distributions

Yulai Cong*, Bo Chen*, and Mingyuan Zhou#

July 16, 2016

**Abstract**

We introduce a fast and easy-to-implement simulation algorithm for a multivariate normal distribution truncated on the intersection of a set of hyperplanes, and further generalize it to efficiently simulate random variables from a multivariate normal distribution whose covariance (precision) matrix can be decomposed as a positive-definite matrix minus (plus) a low-rank symmetric matrix. Example results illustrate the correctness and efficiency of the proposed simulation algorithms.

*Keywords*: Cholesky Decomposition; Conditional Distribution; Equality Constraints; High-Dimensional Regression; Structured Covariance/Precision Matrix.

## 1 Introduction

The need for simulation of multivariate normal (MVN) random variables subject to certain constraints (Gelfand et al., 1992) arises in a wide variety of settings, such as multinomial probit and logit models (Albert and Chib, 1993; Holmes and Held, 2006; Johndrow et al., 2013), Bayesian isotonic regression (Neelon and Dunson, 2004), Bayesian bridge regression (Polson et al., 2014), blind source separation (Schmidt,

* National Laboratory of Radar Signal Processing and Collaborative Innovation Center of Information Sensing & Understanding, Xidian University, Xi'an, Shaanxi 710071, China.
# McCombs School of Business, The University of Texas at Austin, Austin, TX 78712, USA.

2009), and unmixing of hyperspectral data (Altmann et al., 2014; Dobigeon et al., 2009). A typical problem is to sample a random vector $\boldsymbol{x} \in \mathbb{R}^k$ from a MVN distribution subject to $k$ inequality constraints as

$$\boldsymbol{x} \sim \mathcal{N}_{\mathcal{S}}(\boldsymbol{\mu}, \boldsymbol{\Sigma}), \quad \mathcal{S} = \{\boldsymbol{x} : \boldsymbol{l} \leq \mathbf{D}\boldsymbol{x} \leq \boldsymbol{u}\}, \tag{1}$$

where $\mathcal{N}_{\mathcal{S}}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ represents a MVN distribution truncated on the sample space $\mathcal{S}$, $\boldsymbol{\mu} \in \mathbb{R}^k$ is the mean, $\boldsymbol{\Sigma} \in \mathbb{R}^{k \times k}$ is the covariance matrix, $\mathbf{D} \in \mathbb{R}^{k \times k}$ is a full-rank matrix, $\boldsymbol{l} \in \mathbb{R}^k$, $\boldsymbol{u} \in \mathbb{R}^k$, and $\boldsymbol{l} < \boldsymbol{u}$. If the elements of $\boldsymbol{l}$ and $\boldsymbol{u}$ are permitted to be $-\infty$ and $+\infty$, respectively, then both single sided and fewer than $k$ inequality constraints are allowed. Equivalently, as in (Geweke, 1991, 1996), one may let $\boldsymbol{x} = \boldsymbol{\mu} + \mathbf{D}^{-1}\boldsymbol{z}$ and use Gibbs sampling (Geman and Geman, 1984; Gelfand and Smith, 1990) to sample the $k$ elements of $\boldsymbol{z}$ one at a time conditioning on all the others from a univariate truncated normal distribution, for which efficient algorithms exist (Robert, 1995; Damien and Walker, 2001; Chopin, 2011). To deal with the case that the number of linear constraints imposed on $\boldsymbol{x}$ exceed its dimension $k$ and to obtain better mixing, one may consider the Gibbs sampling algorithm for truncated MVN distributions proposed in (Rodriguez-Yam et al., 2004). In addition to Gibbs sampling, to sample truncated MVN random variables, one may also consider Hamiltonian Monte Carlo (Pakman and Paninski, 2014; Lan et al., 2014) and a minimax tilting method proposed in (Botev, 2016).

In this paper, we consider a related problem that we need to sample from a MVN distribution truncated on the intersection of a set of hyperplanes, which will be shown to be inherently related to the simulation of a conditional distribution of a MVN distribution. A naive approach, which linearly transforms a random variable drawn from the conditional distribution of a related MVN distribution, requires a large number of intermediate variables that are often computationally expensive to instantiate. To address this issue, we propose a fast and exact simulation algorithm that directly projects a MVN random variable onto the intersection of a set of hyperplanes. We further show that sampling from a MVN distribution, whose covariance (precision) matrix can be decomposed as the sum (difference) of a positive-definite matrix, whose inversion is known or easy to compute, and a low-rank symmetric matrix, may also be made significantly fast by exploiting this newly proposed stimulation algorithm for hyperplane-truncated MVN distributions.

## 2 Hyperplane-truncated and conditional MVNs

We express a $k$-dimensional MVN distribution truncated on the intersection of $k_2 < k$ hyperplanes as

$$\boldsymbol{x} \sim \mathcal{N}_{\mathcal{S}}(\boldsymbol{\mu}, \boldsymbol{\Sigma}), \quad \mathcal{S} = \{\boldsymbol{x} : \mathbf{G}\boldsymbol{x} = \boldsymbol{r}\}, \tag{2}$$

where

$$\mathbf{G} \in \mathbb{R}^{k_2 \times k}, \quad \boldsymbol{r} \in \mathbb{R}^{k_2},$$

and $\text{Rank}(\mathbf{G}) = k_2$. The probability density function can be expressed as

$$p(\boldsymbol{x} \,|\, \boldsymbol{\mu}, \boldsymbol{\Sigma}, \mathbf{G}, \boldsymbol{r}) = \frac{1}{Z} \exp\left[-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right] \delta(\mathbf{G}\boldsymbol{x} = \boldsymbol{r}), \tag{3}$$

where $Z$ is a constant ensuring $\int p(\boldsymbol{x} \,|\, \boldsymbol{\mu}, \boldsymbol{\Sigma}, \mathbf{G}, \boldsymbol{r}) d\boldsymbol{x} = 1$, and $\delta(x) = 1$ if the condition $x$ is satisfied and $\delta(x) = 0$ otherwise. Let us partition $\mathbf{G}$, $\boldsymbol{x}$, $\boldsymbol{\mu}$, and $\boldsymbol{\Sigma}$ as

$$\mathbf{G} = (\mathbf{G}_1, \mathbf{G}_2), \quad \boldsymbol{x} = \begin{bmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \end{bmatrix}, \quad \boldsymbol{\mu} = \begin{bmatrix} \boldsymbol{\mu}_1 \\ \boldsymbol{\mu}_2 \end{bmatrix}, \quad \text{and} \quad \boldsymbol{\Sigma} = \begin{bmatrix} \boldsymbol{\Sigma}_{11} & \boldsymbol{\Sigma}_{12} \\ \boldsymbol{\Sigma}_{21} & \boldsymbol{\Sigma}_{22} \end{bmatrix},$$

whose sizes are

$$(k_2 \times k_1, k_2 \times k_2), \quad \begin{bmatrix} k_1 \times 1 \\ k_2 \times 1 \end{bmatrix}, \quad \begin{bmatrix} k_1 \times 1 \\ k_2 \times 1 \end{bmatrix}, \quad \text{and} \quad \begin{bmatrix} k_1 \times k_1 & k_1 \times k_2 \\ k_2 \times k_1 & k_2 \times k_2 \end{bmatrix},$$

respectively, where $k = k_1 + k_2$ and $\boldsymbol{\Sigma}_{21} = \boldsymbol{\Sigma}_{12}^T$.

A special case that frequently arises in real applications is when $\mathbf{G}_1 = \mathbf{0}_{k_2 \times k_1}$ and $\mathbf{G}_2 = \mathbf{I}_{k_2}$, which means $(\mathbf{0}_{k_2 \times k_1}, \mathbf{I}_{k_2})\boldsymbol{x} = \boldsymbol{x}_2 = \boldsymbol{r}$ and the need is to simulate $\boldsymbol{x}_1$ given $\boldsymbol{x}_2 = \boldsymbol{r}$. For a MVN random variable $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, it is well known, *e.g.*, in Tong (2012), that the conditional distribution of $\boldsymbol{x}_1$ given $\boldsymbol{x}_2 = \boldsymbol{r}$, *i.e.*, the distribution of $\boldsymbol{x}$ restricted to $\mathcal{S} = \{\boldsymbol{x} : (\mathbf{0}_{k_2 \times k_1}, \mathbf{I}_{k_2})\boldsymbol{x} = \boldsymbol{r}\}$, can be expressed as

$$\boldsymbol{x}_1 \,|\, \boldsymbol{x}_2 = \boldsymbol{r} \sim \mathcal{N}\left[\boldsymbol{\mu}_1 + \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}(\boldsymbol{r} - \boldsymbol{\mu}_2), \quad \boldsymbol{\Sigma}_{11} - \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}\boldsymbol{\Sigma}_{21}\right]. \tag{4}$$

In a general setting where $\mathbf{G} \neq (\mathbf{0}_{k_2 \times k_1}, \mathbf{I}_{k_2})$, let us define a full rank linear transformation matrix $\mathbf{H} \in \mathbb{R}^{k \times k}$, with $(\mathbf{H}_1, \mathbf{H}_2)$ as the $(k \times k_1, k \times k_2)$ partition of $\mathbf{H}$, where the columns of $\mathbf{H}_1 \in \mathbb{R}^{k \times k_1}$ span the null space of the $k_2$ rows of $\mathbf{G}$, making $\mathbf{GH} = (\mathbf{GH}_1, \mathbf{GH}_2) = (\mathbf{0}_{k_2 \times k_1}, \mathbf{GH}_2)$, where $\mathbf{GH}_2$ is a $k_2 \times k_2$ full rank matrix. For example, a linear transformation matrix $\mathbf{H}$ that makes $\mathbf{GH} = (\mathbf{0}_{k_2 \times k_1}, \mathbf{I}_{k_2})$ can be constructed using the command $\mathbf{H} = \text{inv}([\text{null}(\mathbf{G})'; \mathbf{G}])$ in Matlab and

$\mathbf{H}$ <−solve(rbind(t(Null(t($\mathbf{G}$))), $\mathbf{G}$)) in R. With $\mathbf{H}$ and $\mathbf{H}^{-1}$, one may re-express the constraints as $\mathcal{S} = \{\boldsymbol{x} : (\mathbf{0}_{k_2 \times k_1}, \mathbf{GH}_2)(\mathbf{H}^{-1}\boldsymbol{x}) = \boldsymbol{r}\}$. Denote $\boldsymbol{z} = \mathbf{H}^{-1}\boldsymbol{x}$, then we can generate $\boldsymbol{x}$ by letting $\boldsymbol{x} = \mathbf{H}\boldsymbol{z}$, where

$$\boldsymbol{z} \sim \mathcal{N}_{\mathcal{D}}[\mathbf{H}^{-1}\boldsymbol{\mu}, \mathbf{H}^{-1}\boldsymbol{\Sigma}(\mathbf{H}^{-1})^T], \;\; \mathcal{D} = \{\boldsymbol{z} : \mathbf{GH}_2\boldsymbol{z}_2 = \boldsymbol{r}\} = \{\boldsymbol{z} : \boldsymbol{z}_2 = (\mathbf{GH}_2)^{-1}\boldsymbol{r}\}. \quad (5)$$

More specifically, denoting $\boldsymbol{\Lambda} = [\mathbf{H}^{-1}\boldsymbol{\Sigma}(\mathbf{H}^{-1})^T]^{-1} = \mathbf{H}^T\boldsymbol{\Sigma}^{-1}\mathbf{H}$ as the precision matrix for $\boldsymbol{z}$, we have

$$\begin{bmatrix} \boldsymbol{\Lambda}_{11} & \boldsymbol{\Lambda}_{12} \\ \boldsymbol{\Lambda}_{21} & \boldsymbol{\Lambda}_{22} \end{bmatrix} = \mathbf{H}^T\boldsymbol{\Sigma}^{-1}\mathbf{H} = \begin{bmatrix} \mathbf{H}_1^T\boldsymbol{\Sigma}^{-1}\mathbf{H}_1 & \mathbf{H}_1^T\boldsymbol{\Sigma}^{-1}\mathbf{H}_2 \\ \mathbf{H}_2^T\boldsymbol{\Sigma}^{-1}\mathbf{H}_1 & \mathbf{H}_2^T\boldsymbol{\Sigma}^{-1}\mathbf{H}_2 \end{bmatrix}, \quad (6)$$

and hence $\boldsymbol{x}$ truncated on $\mathcal{S}$ can be naively generated using the following algorithm.

---

**Algorithm 1** Simulation of the hyperplane truncated MVN distribution $\boldsymbol{x} \sim \mathcal{N}_{\mathcal{S}}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, where $\mathcal{S} = \{\boldsymbol{x} : \mathbf{G}\boldsymbol{x} = \boldsymbol{r}\}$, by transforming a random variable drawn from the conditional distribution of another MVN distribution.

- Find $\mathbf{H} = (\mathbf{H}_1, \mathbf{H}_2)$ that satisfies $\mathbf{GH} = (\mathbf{GH}_1, \mathbf{GH}_2) = (\mathbf{0}_{k_2 \times k_1}, \mathbf{GH}_2)$, where $\mathbf{GH}_2$ is a full rank matrix;
- Let $\boldsymbol{z}_2 = (\mathbf{GH}_2)^{-1}\boldsymbol{r}$, $\boldsymbol{\Lambda}_{11} = \mathbf{H}_1^T\boldsymbol{\Sigma}^{-1}\mathbf{H}_1$, and $\boldsymbol{\Lambda}_{12} = \mathbf{H}_1^T\boldsymbol{\Sigma}^{-1}\mathbf{H}_2$;
- Sample $\boldsymbol{z}_1 \,|\, \boldsymbol{z}_2 = (\mathbf{GH}_2)^{-1}\boldsymbol{r} \sim \mathcal{N}(\boldsymbol{\mu}_{z_1}, \boldsymbol{\Lambda}_{11}^{-1})$, where

$$\boldsymbol{\mu}_{z_1} = (\mathbf{I}_{k_1}, \mathbf{0}_{k_1 \times k_2})\mathbf{H}^{-1}\boldsymbol{\mu} - \boldsymbol{\Lambda}_{11}^{-1}\boldsymbol{\Lambda}_{12}\left[(\mathbf{GH}_2)^{-1}\boldsymbol{r} - (\mathbf{0}_{k_2 \times k_1}, \mathbf{I}_{k_2})\mathbf{H}^{-1}\boldsymbol{\mu}\right];$$

- Return $\boldsymbol{x} = \mathbf{H}\boldsymbol{z} = \mathbf{H}_1\boldsymbol{z}_1 + \mathbf{H}_2(\mathbf{GH}_2)^{-1}\boldsymbol{r}$.

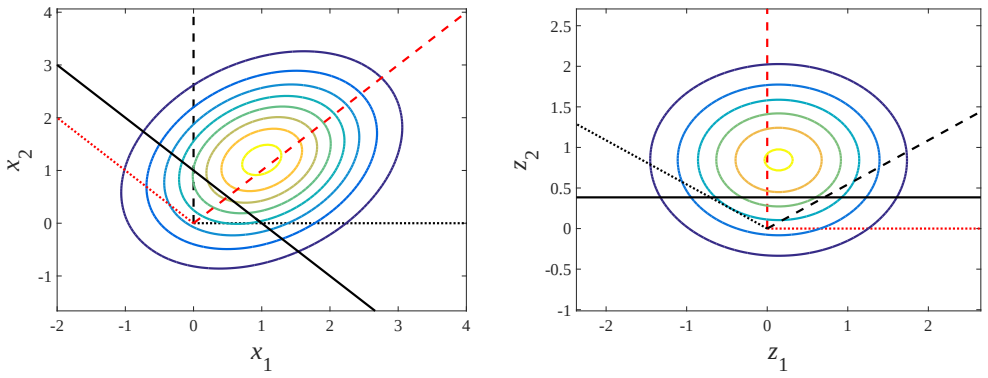
---

For illustration, we consider a simple 2-dimensional example with $\boldsymbol{\mu} = (1, 1.2)^T$, $\boldsymbol{\Sigma} = [(1, 0.3)^T, (0.3, 1)^T]$, $\mathbf{G} = (1, 1)$, and $\boldsymbol{r} = 1$. If we choose $\mathbf{H}_1 = (-0.7071, 0.7071)^T$ and $\mathbf{H}_2 = (1.3, 1.3)^T$, then we have $z_2 = (\mathbf{GH}_2)^{-1}\boldsymbol{r} = (2.6)^{-1} = 0.3846$, $\boldsymbol{\Lambda}_{11} = 1.4285$, and $\boldsymbol{\Lambda}_{12} = 0$; as shown in Figure 1, we may generate $\boldsymbol{x}$ using

$$\boldsymbol{x} = (-0.7071, 0.7071)^T z_1 + (1.3, 1.3)^T z_2$$

where $z_1 \sim \mathcal{N}(0.1414, 0.7)$ and $z_2 = 0.3846$.

For high dimensional problems, however, Algorithm 1 in general requires a large number of intermediate variables that could be computationally expensive to compute. In the following discussion, we will show how to completely avoid instantiating these intermediate variables.




(a) (b)

Figure 1: Illustration of (a) $p(\boldsymbol{x})$ in (3), where $\boldsymbol{\mu} = (1, 1.2)^T$, $\boldsymbol{\Sigma} = [(1, 0.3)^T, (0.3, 1)^T]$, $\mathbf{G} = (1, 1)$, and $\boldsymbol{r} = 1$, and (b) $p(\boldsymbol{z})$ in (5), where $\mathbf{H}_1 = (-0.7071, 0.7071)^T$, $\mathbf{H}_2 = (1.3, 1.3)^T$, and $\mathbf{H}^{-1} = [(-0.7071, 0.3846)^T, (0.7071, 0.3846)^T]$. The coordinate systems of $\boldsymbol{x}$ and $\boldsymbol{z}$ are shown in black and red, respectively, and the first and second axes of a coordinate system are shown as dotted and dashed lines, respectively.

# 3 Fast and exact simulation of MVN distributions

Instead of using Algorithm 1, we first provide a theorem to show how to efficiently and exactly simulate from a hyperplane-truncated MVN distribution. In the Appendix, we provide two different proofs. The first proof employs an existing algorithm of Hoffman and Ribak (1991) and Doucet (2010), which describes how to simulate from the conditional distribution of a MVN distribution shown in (4) without computing $\boldsymbol{\Sigma}_{11} - \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}\boldsymbol{\Sigma}_{21}$ and its Cholesky decomposition, to facilitate the derivations.

---

**Algorithm 2** Simulation of the hyperplane truncated MVN distribution $\boldsymbol{x} \sim \mathcal{N}_{\mathcal{S}}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, where $\mathcal{S} = \{\boldsymbol{x} : \mathbf{G}\boldsymbol{x} = \boldsymbol{r}\}$, by transforming a random variable drawn from $\boldsymbol{y} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$.

---

- Sample $\boldsymbol{y} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$;
- Return $\boldsymbol{x} = \boldsymbol{y} + \boldsymbol{\Sigma}\mathbf{G}^T(\mathbf{G}\boldsymbol{\Sigma}\mathbf{G}^T)^{-1}(\boldsymbol{r} - \mathbf{G}\boldsymbol{y})$, which can be realized using
  - Solve $\boldsymbol{\alpha}$ such that $(\mathbf{G}\boldsymbol{\Sigma}\mathbf{G}^T)\boldsymbol{\alpha} = \boldsymbol{r} - \mathbf{G}\boldsymbol{y}$;
  - Return $\boldsymbol{x} = \boldsymbol{y} + \boldsymbol{\Sigma}\mathbf{G}^T\boldsymbol{\alpha}$.

---

**Theorem 1.** *Suppose $\boldsymbol{x}$ is simulated with Algorithm 2, then it is distributed as $\boldsymbol{x} \sim \mathcal{N}_{\mathcal{S}}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, $\mathcal{S} = \{\boldsymbol{x} : \mathbf{G}\boldsymbol{x} = \boldsymbol{r}\}$, where $\mathbf{G} \in \mathbb{R}^{k_2 \times k}$, $\boldsymbol{r} \in \mathbb{R}^{k_2}$, and Rank$(\mathbf{G}) = k_2 < k$.*

The above algorithm and theorem show that one may draw $\boldsymbol{y}$ from the unconstrained MVN as $\boldsymbol{y} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ and directly map it to a vector $\boldsymbol{x}$ on the intersection of hyperplanes using $\boldsymbol{x} = \boldsymbol{\Sigma}\mathbf{G}^T(\mathbf{G}\boldsymbol{\Sigma}\mathbf{G}^T)^{-1}\boldsymbol{r} + \left[\mathbf{I} - \boldsymbol{\Sigma}\mathbf{G}^T(\mathbf{G}\boldsymbol{\Sigma}\mathbf{G}^T)^{-1}\mathbf{G}\right]\boldsymbol{y}$. For illustration, with the same $\boldsymbol{\mu}$, $\boldsymbol{\Sigma}$, $\mathbf{G}$, and $\boldsymbol{r}$ as those in Figure 1, we show in Figure 2 a simple two dimensional example, where the unrestricted Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ is represented with a set of ellipses, and the constrained sample space $\mathcal{S}$ is represented as a straight line in the two-dimensional setting. With $\boldsymbol{\Sigma}\mathbf{G}^T(\mathbf{G}\boldsymbol{\Sigma}\mathbf{G}^T)^{-1}\boldsymbol{r} = (0.5, 0.5)^T$, $\left[\mathbf{I} - \boldsymbol{\Sigma}\mathbf{G}^T(\mathbf{G}\boldsymbol{\Sigma}\mathbf{G}^T)^{-1}\mathbf{G}\right] = [(0.5, -0.5)^T, (-0.5, 0.5)^T]$, one may directly maps a sample $\boldsymbol{y} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ to a vector on the constrained space. For example, if $\boldsymbol{y} = (1, 2)^T$, then it would be mapped to $\boldsymbol{x} = (0, 1)^T$ on the straight-line.

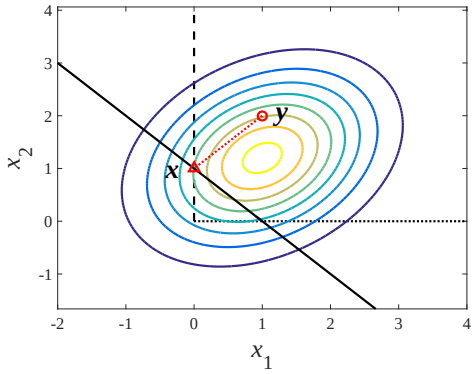



Figure 2: A two dimensional demonstration of Algorithm 2 that maps a random sample from $\boldsymbol{y} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ to a sample in the constrained space using $\boldsymbol{x} = \boldsymbol{\Sigma}\mathbf{G}^T(\mathbf{G}\boldsymbol{\Sigma}\mathbf{G}^T)^{-1}\boldsymbol{r} + \left[\mathbf{I} - \boldsymbol{\Sigma}\mathbf{G}^T(\mathbf{G}\boldsymbol{\Sigma}\mathbf{G}^T)^{-1}\mathbf{G}\right]\boldsymbol{y}$. For example, if $\boldsymbol{\mu} = (1, 1.2)^T$, $\boldsymbol{\Sigma} = [(1, 0.3)^T, (0.3, 1)^T]$, $\mathbf{G} = (1, 1)$, and $\boldsymbol{r} = 1$, then $\boldsymbol{y} = (1, 2)^T$ would be mapped to $\boldsymbol{x} = (0, 1)^T$ on a straight line using Algorithm 2.

## 3.1 Fast simulation of MVN distributions with structured covariance or precision matrices

For fast simulation of MVN distributions with structured covariance or precision matrices, our idea is to relate them to higher-dimensional hyperplane-truncated MVN distributions, with block-diagonal covariance matrices, that can be efficiently simulated with Algorithm 2. We first introduce an efficient algorithm for the simulation of a MVN distribution, whose covariance matrix is a positive-definite matrix subtracted by a low-rank symmetric matrix. Such kind of covariance matrices commonly arise in the conditional distributions of MVN distributions, as shown in (4). We then further extend this algorithm to the simulation of a MVN distribution whose precision

(inverse covariance) matrix is the sum of a positive-definite matrix and a low-rank symmetric matrix. Such kind of precision matrices commonly arise in the conditional posterior distributions of the regression coefficients in both linear regression and generalized linear models.

**Theorem 2.** *The probability density function (PDF) of the MVN distribution*

$$\boldsymbol{x}_1 \sim \mathcal{N}(\boldsymbol{\mu}_1,\ \boldsymbol{\Sigma}_{11} - \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}\boldsymbol{\Sigma}_{21}), \tag{7}$$

*is the same as the PDF of the marginal distribution of* $\boldsymbol{x}_1 = (x_1, \ldots, x_{k_1})^T$ *in* $\boldsymbol{x} = (\boldsymbol{x}_1^T, x_{k_1+1}, \ldots, x_k)^T$*, whose PDF is expressed as*

$$\begin{aligned} p(\boldsymbol{x} \,|\, \mu, \tilde{\boldsymbol{\Sigma}}, \mathbf{G}, \boldsymbol{r}) &= \mathcal{N}_{\{\boldsymbol{x}:\mathbf{G}^T\boldsymbol{x}=\boldsymbol{r}\}}\left(\boldsymbol{\mu}, \tilde{\boldsymbol{\Sigma}}\right) \\ &= \frac{1}{Z}\exp\left[-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T\tilde{\boldsymbol{\Sigma}}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right]\delta(\mathbf{G}\boldsymbol{x}=\boldsymbol{r}), \end{aligned} \tag{8}$$

*where* $Z$ *is a normalization constant,* $\mathbf{G}_1 = \boldsymbol{\Sigma}_{21}\boldsymbol{\Sigma}_{11}^{-1}$ *is a matrix of size* $k_2 \times k_1$*,* $\mathbf{G}_2$ *is a user-specified full rank invertible matrix of size* $k_2 \times k_2$*,* $\boldsymbol{r} \in \mathbb{R}^{k_2}$ *is a user-specified vector, and*

$$\mathbf{G} = (\mathbf{G}_1, \mathbf{G}_2) \in \mathbb{R}^{k_2 \times k}, \qquad \boldsymbol{\mu} = \begin{bmatrix} \boldsymbol{\mu}_1 \\ \boldsymbol{\mu}_2 \end{bmatrix} \in \mathbb{R}^k, \qquad \tilde{\boldsymbol{\Sigma}} = \begin{bmatrix} \boldsymbol{\Sigma}_{11} & \mathbf{0} \\ \mathbf{0} & \tilde{\boldsymbol{\Sigma}}_{22} \end{bmatrix} \in \mathbb{R}^{k \times k}, \tag{9}$$

*where*

$$\boldsymbol{\mu}_2 = \mathbf{G}_2^{-1}(\boldsymbol{r} - \boldsymbol{\Sigma}_{21}\boldsymbol{\Sigma}_{11}^{-1}\boldsymbol{\mu}_1), \tag{10}$$

$$\tilde{\boldsymbol{\Sigma}}_{22} = \mathbf{G}_2^{-1}\left(\boldsymbol{\Sigma}_{22} - \boldsymbol{\Sigma}_{21}\boldsymbol{\Sigma}_{11}^{-1}\boldsymbol{\Sigma}_{12}\right)(\mathbf{G}_2^{-1})^T. \tag{11}$$

The above theorem shows how the simulation of a MVN distribution, whose covariance matrix is a positive-definite matrix minus a symmetric matrix, can be realized by the simulation of a higher-dimensional hyperplane-truncated MVN distribution. By construction, it makes the covariance matrix $\tilde{\boldsymbol{\Sigma}}$ of the truncated-MVN be block diagonal, but still provides the flexibility to customize the full-rank matrix $\mathbf{G}_2$ and the vector $\boldsymbol{r}$. Further removing that flexibility by specifying the parameters of the hyperplane-tunicated MVN distribution as $\mathbf{G} = (\mathbf{G}_1, \mathbf{G}_2) = (\boldsymbol{\Sigma}_{21}\boldsymbol{\Sigma}_{11}^{-1}, \mathbf{I}_{k_2})$ and $\boldsymbol{r} = \boldsymbol{\Sigma}_{21}\boldsymbol{\Sigma}_{11}^{-1}\boldsymbol{\mu}_1$, we have the following Corollary as a special case of Theorem 2.

**Corollary 3.** *The PDF of the MVN distribution*

$$\boldsymbol{x}_1 \sim \mathcal{N}(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_{11} - \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}\boldsymbol{\Sigma}_{21}) \tag{12}$$

*is the same as the PDF of the marginal distribution of* $\boldsymbol{x}_1$ *in* $\boldsymbol{x} = (\boldsymbol{x}_1^T, x_{k_1+1}, \ldots, x_k)^T$*, whose PDF is expressed as*

$$\begin{aligned} p(\boldsymbol{x}) &= \mathcal{N}_{\{\boldsymbol{x}:\, \boldsymbol{\Sigma}_{21}\boldsymbol{\Sigma}_{11}^{-1}\boldsymbol{x}_1 + \boldsymbol{x}_2 = \boldsymbol{\Sigma}_{21}\boldsymbol{\Sigma}_{11}^{-1}\boldsymbol{\mu}_1\}}\left(\boldsymbol{\mu}, \tilde{\boldsymbol{\Sigma}}\right) \\ &= \frac{1}{Z}\exp\left[-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T\tilde{\boldsymbol{\Sigma}}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right]\delta(\boldsymbol{\Sigma}_{21}\boldsymbol{\Sigma}_{11}^{-1}\boldsymbol{x}_1 + \boldsymbol{x}_2 = \boldsymbol{\Sigma}_{21}\boldsymbol{\Sigma}_{11}^{-1}\boldsymbol{\mu}_1), \end{aligned} \tag{13}$$

*where* $Z$ *is a normalization constant and*

$$\boldsymbol{\mu} = \begin{bmatrix} \boldsymbol{\mu}_1 \\ \mathbf{0} \end{bmatrix} \in \mathbb{R}^k, \qquad \tilde{\boldsymbol{\Sigma}} = \begin{bmatrix} \boldsymbol{\Sigma}_{11} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Sigma}_{22} - \boldsymbol{\Sigma}_{21}\boldsymbol{\Sigma}_{11}^{-1}\boldsymbol{\Sigma}_{12} \end{bmatrix} \in \mathbb{R}^{k\times k}. \tag{14}$$

Further applying Theorem 1 to Corollary 3, as described in detail in the Appendix, a MVN random variable $\boldsymbol{x}$ with a structured covariance matrix can be generated as follows, where there is no need to compute $\boldsymbol{\Sigma}_{11} - \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}\boldsymbol{\Sigma}_{21}$ and its Cholesky decomposition, which could lead to a significant saving in computation if $k_2 \ll k_1$.

---

**Algorithm 3** Simulation of the MVN distribution

$$\boldsymbol{x}_1 \sim \mathcal{N}(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_{11} - \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}\boldsymbol{\Sigma}_{21}).$$

---

- Sample $\boldsymbol{y}_1 \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_{11})$ and $\boldsymbol{y}_2 \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_{22} - \boldsymbol{\Sigma}_{21}\boldsymbol{\Sigma}_{11}^{-1}\boldsymbol{\Sigma}_{12})$ ;
- Return $\boldsymbol{x}_1 = \boldsymbol{\mu}_1 + \boldsymbol{y}_1 - \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}(\boldsymbol{\Sigma}_{21}\boldsymbol{\Sigma}_{11}^{-1}\boldsymbol{y}_1 + \boldsymbol{y}_2)$, which can be realized using
  - Solve $\boldsymbol{\alpha}$ such that $\boldsymbol{\Sigma}_{22}\boldsymbol{\alpha} = \boldsymbol{\Sigma}_{21}\boldsymbol{\Sigma}_{11}^{-1}\boldsymbol{y}_1 + \boldsymbol{y}_2$;
  - Return $\boldsymbol{x}_1 = \boldsymbol{\mu}_1 + \boldsymbol{y}_1 - \boldsymbol{\Sigma}_{12}\boldsymbol{\alpha}$.

---

**Corollary 4.** *A random variable simulated with Algorithm 3 is distributed as* $\boldsymbol{x}_1 \sim \mathcal{N}(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_{11} - \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}\boldsymbol{\Sigma}_{21})$.

The efficient simulation algorithm for a MVN distribution with a structured covariance matrix can also be further extended to a MVN distribution with a structured precision matrix, as described below.

**Algorithm 4** Simulation of the MVN distribution

$$\boldsymbol{\beta} \sim \mathcal{N}\left[\boldsymbol{\mu}_\beta, (\mathbf{A} + \boldsymbol{\Phi}^T\boldsymbol{\Omega}\boldsymbol{\Phi})^{-1}\right].$$

- Sample $\boldsymbol{y}_1 \sim \mathcal{N}(\mathbf{0}, \mathbf{A}^{-1})$ and $\boldsymbol{y}_2 \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Omega}^{-1})$ ;
- Return $\boldsymbol{\beta} = \boldsymbol{\mu}_\beta + \boldsymbol{y}_1 - \mathbf{A}^{-1}\boldsymbol{\Phi}^T(\boldsymbol{\Omega}^{-1} + \boldsymbol{\Phi}\mathbf{A}^{-1}\boldsymbol{\Phi}^T)^{-1}(\boldsymbol{\Phi}\boldsymbol{y}_1 + \boldsymbol{y}_2)$, which can be realized using
  - Solve $\boldsymbol{\alpha}$ such that $(\boldsymbol{\Omega}^{-1} + \boldsymbol{\Phi}\mathbf{A}^{-1}\boldsymbol{\Phi}^T)\boldsymbol{\alpha} = \boldsymbol{\Phi}\boldsymbol{y}_1 + \boldsymbol{y}_2$.
  - Return $\boldsymbol{\beta} = \boldsymbol{\mu}_\beta + \boldsymbol{y}_1 - \mathbf{A}^{-1}\boldsymbol{\Phi}^T\boldsymbol{\alpha}$.

**Corollary 5.** *The random variable obtained with Algorithm 4 is distributed as* $\boldsymbol{\beta} \sim \mathcal{N}(\boldsymbol{\mu}_\beta, \boldsymbol{\Sigma}_\beta)$*, where* $\boldsymbol{\Sigma}_\beta = (\mathbf{A} + \boldsymbol{\Phi}^T\boldsymbol{\Omega}\boldsymbol{\Phi})^{-1}$.

Further generalizing Corollary 5, assuming $\boldsymbol{\Phi} \in \mathbb{R}^{n\times p}$ and $\mathbf{A} \in \mathbb{R}^{p\times p}$ and denoting $p$ as the number of predictors and $n$ as the number of data samples, Corollary 6 described below shows, if $p \gg n$, how to efficiently sample from the conditional posterior of $\boldsymbol{\beta}$ in the linear regression model

$$\boldsymbol{t} \sim \mathcal{N}(\boldsymbol{\Phi}\boldsymbol{\beta}, \boldsymbol{\Omega}^{-1}),\ \boldsymbol{\beta} \sim \mathcal{N}(\mathbf{0}, \mathbf{A}^{-1}), \tag{15}$$

where different constructions on $\mathbf{A}$ lead to a wide variety of regression models (Caron and Doucet, 2008; Carvalho et al., 2010; Polson et al., 2014). Note that Corollary 6, useful for high-dimensional regression, is essentially the same as Proposition 2.1 of (Bhattacharya et al., 2015) if $\boldsymbol{\Omega} = \mathbf{I}_n$. Experimental results in Bhattacharya et al. (2015) show that this simulation algorithm could be significantly more efficient than that of Rue (2001) for high-dimensional regression if $p \gg n$.

**Algorithm 5** Simulation of the MVN distribution

$$\boldsymbol{\beta} \sim \mathcal{N}\left[(\mathbf{A} + \boldsymbol{\Phi}^T\boldsymbol{\Omega}\boldsymbol{\Phi})^{-1}\boldsymbol{\Phi}^T\boldsymbol{\Omega}\boldsymbol{t},\ \ (\mathbf{A} + \boldsymbol{\Phi}^T\boldsymbol{\Omega}\boldsymbol{\Phi})^{-1}\right].$$

- Sample $\boldsymbol{y}_1 \sim \mathcal{N}(\mathbf{0}, \mathbf{A}^{-1})$ and $\boldsymbol{y}_2 \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Omega}^{-1})$ ;
- Return $\boldsymbol{\beta} = \boldsymbol{y}_1 + \mathbf{A}^{-1}\boldsymbol{\Phi}^T(\boldsymbol{\Omega}^{-1} + \boldsymbol{\Phi}\mathbf{A}^{-1}\boldsymbol{\Phi}^T)^{-1}(\boldsymbol{t} - \boldsymbol{\Phi}\boldsymbol{y}_1 - \boldsymbol{y}_2)$, which can be realized using
  - Solve $\boldsymbol{\alpha}$ such that $(\boldsymbol{\Omega}^{-1} + \boldsymbol{\Phi}\mathbf{A}^{-1}\boldsymbol{\Phi}^T)\boldsymbol{\alpha} = \boldsymbol{t} - \boldsymbol{\Phi}\boldsymbol{y}_1 - \boldsymbol{y}_2$;
  - Return $\boldsymbol{\beta} = \boldsymbol{y}_1 + \mathbf{A}^{-1}\boldsymbol{\Phi}^T\boldsymbol{\alpha}$.

**Corollary 6.** *The random variable obtained with Algorithm 5 is distributed as* $\boldsymbol{\beta} \sim \mathcal{N}(\boldsymbol{\mu}_\beta, \boldsymbol{\Sigma}_\beta)$*, where* $\boldsymbol{\mu}_\beta = \boldsymbol{\Sigma}_\beta \boldsymbol{\Phi}^T \boldsymbol{\Omega} \boldsymbol{t}$ *and* $\boldsymbol{\Sigma}_\beta = (\mathbf{A} + \boldsymbol{\Phi}^T \boldsymbol{\Omega} \boldsymbol{\Phi})^{-1}$.

# 4 Illustrations

Below we provide two simple examples to illustrate Theorem 1, which shows how to efficiently simulate from a hyperplane-truncated MVN distribution, and Corollary 4, which shows how to efficiently simulate from a MVN distribution with a structured covariance matrix.

**Example 1:** *Simulation of a hyperplane-truncated MVN distribution as*

$$\boldsymbol{x} \sim \mathcal{N}_{\mathcal{S}}[\boldsymbol{\mu}, a\,\mathrm{diag}(\boldsymbol{\phi})], \quad \mathcal{S} = \left\{\boldsymbol{x} : \mathbf{1}^T \boldsymbol{x} = 1\right\},$$

*where* $\boldsymbol{x} \in \mathbb{R}^k$*,* $\boldsymbol{\mu} \in \mathbb{R}^k$*,* $\mathbf{1}^T \boldsymbol{x} = \sum_{i=1}^k x_i$*,* $\boldsymbol{\phi} \in \mathbb{R}^k$*,* $a > 0$*,* $\phi_i > 0$ *for* $i \in \{1, \cdots, k\}$*, and* $\mathbf{1}^T \boldsymbol{\phi} = \sum_{i=1}^k \phi_i = 1$*, can be realized as follows.*

- *Sample* $\boldsymbol{y}$ *from* $\mathcal{N}[\boldsymbol{\mu}, a\mathrm{diag}(\boldsymbol{\phi})]$*;*
- *Return* $\boldsymbol{x} = \boldsymbol{y} + (1 - \mathbf{1}^T \boldsymbol{y})\boldsymbol{\phi}$.

The sampling steps in the above Example directly follow Algorithm 2 and Theorem 1 with the distribution parameters specified as $\boldsymbol{\Sigma} = a\mathrm{diag}(\boldsymbol{\phi})$, $\mathbf{G} = \mathbf{1}$, and $\boldsymbol{r} = 1$.

**Example 2:** Simulation of a MVN distribution as

$$\boldsymbol{x}_1 \sim \mathcal{N}[\boldsymbol{\mu}_1, a\,\mathrm{diag}(\boldsymbol{\phi}_1) - a\,\boldsymbol{\phi}_1 \boldsymbol{\phi}_1^T],$$

where $\boldsymbol{x}_1 \in \mathbb{R}^{k-1}$, $\boldsymbol{\mu}_1 \in \mathbb{R}^{k-1}$, $a > 0$, $\boldsymbol{\phi}_1 = (\phi_1, \ldots, \phi_{k-1})^T$, $\phi_i > 0$ for $i \in \{1, \ldots, k-1\}$, and $\sum_{i=1}^{k-1} \phi_i < 1$, can be realized as follows.

- Sample $\boldsymbol{y}_1 \sim \mathcal{N}[\mathbf{0}, a\,\mathrm{diag}(\boldsymbol{\phi}_1)]$ and $\boldsymbol{y}_2 \sim \mathcal{N}(0, a^{-1}\,\phi_k)$, where $\phi_k = 1 - \sum_{i=1}^{k-1} \phi_i$;
- Return $\boldsymbol{x}_1 = \boldsymbol{\mu}_1 + \boldsymbol{y}_1 - (\mathbf{1}^T \boldsymbol{y}_1 + a\boldsymbol{y}_2)\boldsymbol{\phi}_1$.

Denoting $\boldsymbol{x} = (\boldsymbol{x}_1^T, x_k)^T$, $\boldsymbol{\phi} = (\boldsymbol{\phi}_1^T, \phi_k)^T$, $\boldsymbol{\mu} = (\boldsymbol{\mu}_1^T, \mu_k)^T$, and $\mu_k = 1 - \mathbf{1}^T \boldsymbol{\mu}_1$, the above sampling steps can also be equivalently expressed as follows.

- Sample $\boldsymbol{y} \sim \mathcal{N}[\boldsymbol{\mu}, a\,\mathrm{diag}(\boldsymbol{\phi})]$;
- Return $\boldsymbol{x}_1 = \boldsymbol{y}_1 + (1 - \mathbf{1}^T \boldsymbol{y})\boldsymbol{\phi}_1$.

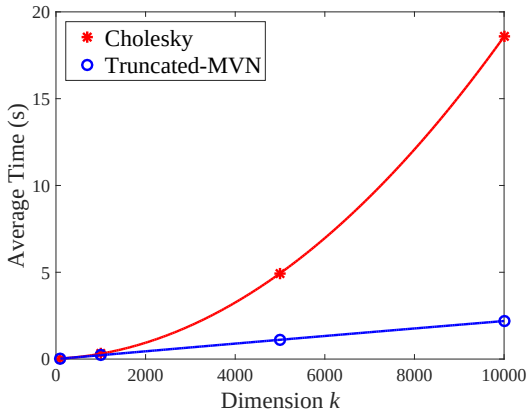



Figure 3: Comparison of two simulation algorithms' average times to simulate 10,000 $k$-dimensional random samples from $\boldsymbol{x}_1 \sim \mathcal{N}[\boldsymbol{\mu}_1, a\,\mathrm{diag}(\boldsymbol{\phi}_1) - a\,\boldsymbol{\phi}_1\boldsymbol{\phi}_1^T]$. The naive simulation algorithm is based on Cholesky decomposition, whereas the fast simulation algorithm employs a hyperplane-truncated MVN distribution.

Directly following Algorithm 3 and Corollary 4, the first sampling approach for the above example can be derived by specifying the distribution parameters as $\boldsymbol{\Sigma}_{11} = a\mathrm{diag}(\boldsymbol{\phi}_1)$, $\boldsymbol{\Sigma}_{12} = \boldsymbol{\phi}_1$, $\boldsymbol{\Sigma}_{21} = \boldsymbol{\phi}_1^T$, and $\boldsymbol{\Sigma}_{22} = a^{-1}$, while the second approach can be derived by specifying $\boldsymbol{\Sigma}_{11} = a\mathrm{diag}(\boldsymbol{\phi}_1)$, $\boldsymbol{\Sigma}_{12} = a\boldsymbol{\phi}_1$, $\boldsymbol{\Sigma}_{21} = a\boldsymbol{\phi}_1^T$, and $\boldsymbol{\Sigma}_{22} = a$.

To illustrate the efficiency of the proposed algorithms, we simulate from the MVN distribution $\boldsymbol{x}_1 \sim \mathcal{N}[\boldsymbol{\mu}_1, a\,\mathrm{diag}(\boldsymbol{\phi}_1) - a\,\boldsymbol{\phi}_1\boldsymbol{\phi}_1^T]$ using both a naive implementation via Cholesky decomposition of the covariance matrix and the fast simulation algorithm for a hyperplane-truncated MVN random variable described in Example 2. We set the dimension from $k = 10^2$ up to $k = 10^4$ and set $\boldsymbol{\mu} = (1/k, \ldots, 1/k)$ and $a = 0.5$. For each $k$ and each simulation algorithm, we perform 100 independent random trials, in each of which $\boldsymbol{\phi}$ is sampled from the Dirichlet distribution $\mathrm{Dir}(1, \ldots, 1)$ and 10,000 independent random samples are simulated using that same $\boldsymbol{\phi}$ on a 2.9 GHz computer.

As shown in Figure 3, for the proposed hyperplane-truncated MVN distribution based simulation algorithm, the average time of simulating 10,000 random samples increases linearly in the dimension $k$. By contrast, for the naive simulation algorithm based on Cholesky decomposition, whose computational complexity is $O(k^3)$ (Golub and Van Loan, 2012), the average simulation time increases at a significantly faster rate as the dimension $k$ increases.

For numerical verification, with the 10,000 simulated $k = 10^4$ dimensional random samples in a random trial, we randomly choose two dimensions and display their joint distribution using a contour plot. As in Figure 4, shown in the first row are the contour plots of five different random trials for the naive Cholesky decomposition

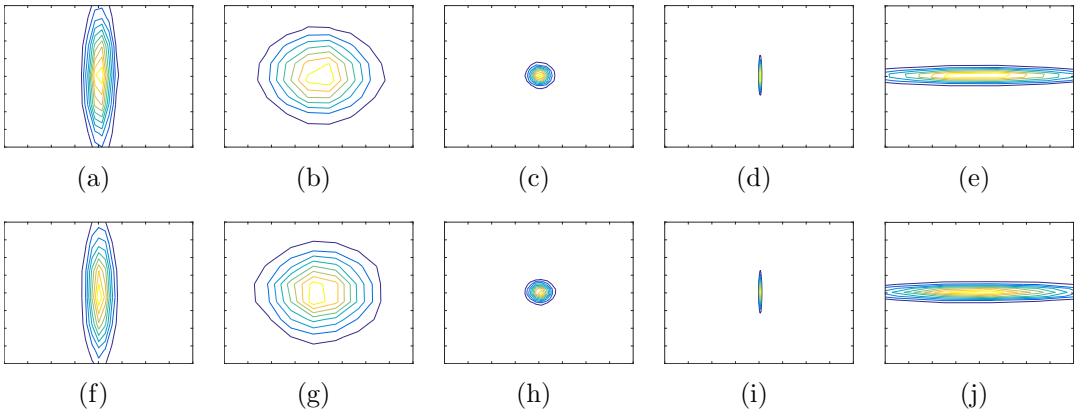



Figure 4: Comparison of the contour plots of two randomly selected dimensions of the 10,000 $k = 10^4$ dimensional random samples simulated with two different methods. Each of the five columns corresponds to a random trial. Shown in the first row are the results of the naive simulation algorithm based on Cholesky decomposition, and shown in the second row are the results of the fast simulation algorithm that employs a hyperplane-truncated MVN distribution.

based simulation algorithm, whereas shown in the second row are the corresponding ones for the proposed fast simulation algorithm. As expected, the contour lines of the two figures in the same column closely resemble each other.

# 5 Conclusions

A fast and exact simulation algorithm is developed for a multivariate normal (MVN) distribution whose sample space is constrained on the intersection of a set of hyperplanes, which is shown to be inherently related to the conditional distribution of a unconstrained MVN distribution. The proposed simulation algorithm is further generalized to efficiently simulate from a MVN distribution, whose covariance (precision) matrix can be decomposed as the sum (difference) of a positive-definite matrix and a low-rank symmetric matrix, using a higher dimensional hyperplane-truncated MVN distribution whose covariance matrix is block-diagonal. It is of interest to investigate how to generalize the proposed fast simulation algorithms for simulation of a MVN distribution subject to both equality and inequality constraints.

# 6 Acknowledgements

The authors thank Yingbo Li and Xiaojing Wang for helpful discussions.

# References


J. H. Albert and S. Chib. Bayesian analysis of binary and polychotomous response data. *J. Amer. Statist. Assoc.*, 88(422):669–679, 1993.

Y. Altmann, S. McLaughlin, and N. Dobigeon. Sampling from a multivariate Gaussian distribution truncated on a simplex: a review. In *Statistical Signal Processing (SSP), 2014 IEEE Workshop on*, pages 113–116. IEEE, 2014.

A. Bhattacharya, A. Chakraborty, and B. K. Mallick. Fast sampling with Gaussian scale-mixture priors in high-dimensional regression. *arXiv preprint arXiv:1506.04778*, 2015.

Z. Botev. The normal law under linear restrictions: simulation and estimation via minimax tilting. *J. Roy. Statist. Soc.: Series B*, 2016.

F. Caron and A. Doucet. Sparse Bayesian nonparametric regression. In *ICML*, pages 88–95. ACM, 2008.

C. M. Carvalho, N. G. Polson, and J. G. Scott. The horseshoe estimator for sparse signals. *Biometrika*, page asq017, 2010.

N. Chopin. Fast simulation of truncated gaussian distributions. *Statistics and Computing*, 21(2):275–288, 2011.

P. Damien and S. G. Walker. Sampling truncated normal, beta, and gamma densities. *Journal of Computational and Graphical Statistics*, 10(2):206–215, 2001.

N. Dobigeon, S. Moussaoui, M. Coulon, J.-Y. Tourneret, and A. O. Hero. Joint Bayesian endmember extraction and linear unmixing for hyperspectral imagery. *IEEE Transactions on Signal Processing*, 57(11):4355–4368, 2009.

A. Doucet. A note on efficient conditional simulation of Gaussian distributions. *Departments of Computer Science and Statistics, University of British Columbia*, 2010.

A. E. Gelfand and A. F. M. Smith. Sampling-based approaches to calculating marginal densities. *J. Amer. Statist. Assoc.*, 85(410):398–409, 1990.

A. E. Gelfand, A. F. Smith, and T.-M. Lee. Bayesian analysis of constrained parameter and truncated data problems using Gibbs sampling. *J. Amer. Statist. Assoc.*, 87(418): 523–532, 1992.

S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Trans. Pattern Anal. Mach. Intell.*, pages 721–741, 1984.

J. Geweke. Efficient simulation from the multivariate normal and student-t distributions subject to linear constraints and the evaluation of constraint probabilities. In *Computing science and statistics: Proceedings of the 23rd symposium on the interface*, pages 571–578, 1991.

J. F. Geweke. Bayesian inference for linear models subject to linear inequality constraints. In *Modelling and Prediction Honoring Seymour Geisser*, pages 248–263. Springer, 1996.

G. H. Golub and C. F. Van Loan. *Matrix computations*, volume 3. JHU Press, 2012.

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

C. C. Holmes and L. Held. Bayesian auxiliary variable models for binary and multinomial regression. *Bayesian Analysis*, 1(1):145–168, 2006.

J. Johndrow, D. Dunson, and K. Lum. Diagonal orthant multinomial probit models. In *AISTATS*, pages 29–38, 2013.

S. Lan, B. Zhou, and B. Shahbaba. Spherical Hamiltonian Monte Carlo for constrained target distributions. In *ICML*, pages 629–637, 2014.

B. Neelon and D. B. Dunson. Bayesian isotonic regression and trend analysis. *Biometrics*, 60(2):398–406, 2004.

A. Pakman and L. Paninski. Exact Hamiltonian Monte Carlo for truncated multivariate Gaussians. *Journal of Computational and Graphical Statistics*, 23(2):518–542, 2014.

N. G. Polson, J. G. Scott, and J. Windle. The Bayesian bridge. *J. Roy. Statist. Soc.: Series B*, 76(4):713–733, 2014.

C. P. Robert. Simulation of truncated normal variables. *Statistics and computing*, 5(2): 121–125, 1995.

G. Rodriguez-Yam, R. A. Davis, and L. L. Scharf. Efficient Gibbs sampling of truncated multivariate normal with application to constrained linear regression. *Technical report*, 2004.

H. Rue. Fast sampling of Gaussian Markov random fields. *J. Roy. Statist. Soc.: Series B*, 63(2):325–338, 2001.

M. Schmidt. Linearly constrained Bayesian matrix factorization for blind source separation. In *NIPS*, pages 1624–1632, 2009.

Y. L. Tong. *The multivariate normal distribution*. Springer Science & Business Media, 2012.

# Appendix

**Algorithm 6** (Hoffman and Ribak, 1991; Doucet, 2010) Simulation of the conditional distribution of $\boldsymbol{x}_1$ given $\boldsymbol{x}_2 = \boldsymbol{r}$ as $\boldsymbol{x}_1 \,|\, \boldsymbol{x}_2 = \boldsymbol{r} \sim \mathcal{N}\left[\boldsymbol{\mu}_1 + \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}(\boldsymbol{r} - \boldsymbol{\mu}_2),\ \boldsymbol{\Sigma}_{11} - \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}\boldsymbol{\Sigma}_{21}\right]$, where the joint distribution of $\boldsymbol{x} = (\boldsymbol{x}_1^T, \boldsymbol{x}_2^T)$ follows $\boldsymbol{x} \sim (\boldsymbol{\mu}, \boldsymbol{\Sigma})$.

- Sample $\boldsymbol{y} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ and denote $\boldsymbol{y}_1 = (y_1, \ldots, y_{k_1})^T$ and $\boldsymbol{y}_2 = (y_{k_1+1}, \ldots, y_k)^T$;
- Return $\boldsymbol{x}_1 = \boldsymbol{y}_1 + \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}(\boldsymbol{r} - \boldsymbol{y}_2)$.

*Proof of Theorem 1.* Let us denote $\boldsymbol{\Lambda} = \mathbf{H}^T\boldsymbol{\Sigma}^{-1}\mathbf{H}$ as a precision matrix that can be partitioned as in (6). For Algorithm 1, instead of directly simulating $\boldsymbol{z}_1$ given $\boldsymbol{z}_2$ using the conditional distribution of the MVN, we apply Algorithm 6 (Hoffman and Ribak, 1991; Doucet, 2010) to modify its sampling steps as follows.

- Sample $\tilde{\boldsymbol{z}} \sim \mathcal{N}[\mathbf{H}^{-1}\boldsymbol{\mu}, \mathbf{H}^{-1}\boldsymbol{\Sigma}(\mathbf{H}^{-1})^T]$, and denote $\tilde{\boldsymbol{z}}_1 = (z_1, \ldots, z_{k_1})^T$ and $\tilde{\boldsymbol{z}}_2 = (z_{k_1+1}, \ldots, z_k)$;
- Let $\boldsymbol{z} = (\boldsymbol{z}_1^T, \boldsymbol{z}_2^T)^T$, where $\boldsymbol{z}_2 = (\mathbf{G}\mathbf{H}_2)^{-1}\boldsymbol{r}$ and $\boldsymbol{z}_1 = \tilde{\boldsymbol{z}}_1 - \boldsymbol{\Lambda}_{11}^{-1}\boldsymbol{\Lambda}_{12}(\boldsymbol{z}_2 - \tilde{\boldsymbol{z}}_2)$, and return

$$\begin{aligned}\boldsymbol{x} = \mathbf{H}\boldsymbol{z} &= \mathbf{H}_1\boldsymbol{z}_1 + \mathbf{H}_2(\mathbf{G}\mathbf{H}_2)^{-1}\boldsymbol{r} \\ &= \mathbf{H}_1\tilde{\boldsymbol{z}}_1 + \mathbf{H}_1\boldsymbol{\Lambda}_{11}^{-1}\boldsymbol{\Lambda}_{12}\tilde{\boldsymbol{z}}_2 + (\mathbf{H}_2 - \mathbf{H}_1\boldsymbol{\Lambda}_{11}^{-1}\boldsymbol{\Lambda}_{12})(\mathbf{G}\mathbf{H}_2)^{-1}\boldsymbol{r} \\ &= (\mathbf{H}_1, \mathbf{H}_1\boldsymbol{\Lambda}_{11}^{-1}\boldsymbol{\Lambda}_{12})\tilde{\boldsymbol{z}} + (\mathbf{H}_2 - \mathbf{H}_1\boldsymbol{\Lambda}_{11}^{-1}\boldsymbol{\Lambda}_{12})(\mathbf{G}\mathbf{H}_2)^{-1}\boldsymbol{r}. \qquad (16)\end{aligned}$$

Therefore, we can equivalently generate $\boldsymbol{x}$ as follows.

- Sample $\boldsymbol{y} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$.
- Return $\boldsymbol{x} = (\mathbf{H}_1, \mathbf{H}_1\boldsymbol{\Lambda}_{11}^{-1}\boldsymbol{\Lambda}_{12})\mathbf{H}^{-1}\boldsymbol{y} + (\mathbf{H}_2 - \mathbf{H}_1\boldsymbol{\Lambda}_{11}^{-1}\boldsymbol{\Lambda}_{12})(\mathbf{G}\mathbf{H}_2)^{-1}\boldsymbol{r}$.

The computation can be significantly simplified if $\boldsymbol{\Lambda}_{12} = \mathbf{0}$, which means

$$\boldsymbol{\Lambda}_{12} = \mathbf{H}_1^T\boldsymbol{\Sigma}^{-1}\mathbf{H}_2 = \mathbf{0}.$$

Since $\mathbf{H}_1^T\mathbf{G}^T = 0$ by definition, to make $\boldsymbol{\Lambda}_{12} = \mathbf{0}$, if and only if we have $\mathbf{H}_2$ as

$$\mathbf{H}_2 = \boldsymbol{\Sigma}\mathbf{G}^T\mathbf{M},$$

where $\mathbf{M} \in \mathbb{R}^{k_2 \times k_2}$ is an arbitrary full rank matrix, under which we have

- Sample $\boldsymbol{y} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$.
- Return $\boldsymbol{x} = (\mathbf{H}_1, \mathbf{0}_{k \times k_2})\mathbf{H}^{-1}\boldsymbol{y} + \boldsymbol{\Sigma}\mathbf{G}^T(\mathbf{G}\boldsymbol{\Sigma}\mathbf{G}^T)^{-1}\boldsymbol{r}$, or return

$$\boldsymbol{x} = \boldsymbol{y} - (\mathbf{0}_{k \times k_1}, \boldsymbol{\Sigma}\mathbf{G}^T\mathbf{M})\mathbf{H}^{-1}\boldsymbol{y} + \boldsymbol{\Sigma}\mathbf{G}^T(\mathbf{G}\boldsymbol{\Sigma}\mathbf{G}^T)^{-1}\boldsymbol{r}. \tag{17}$$

Let us denote $(\mathbf{0}_{k \times k_1}, \boldsymbol{\Sigma}\mathbf{G}^T\mathbf{M})\mathbf{H}^{-1} = \mathbf{C}$. We have $(\mathbf{0}_{k \times k_1}, \boldsymbol{\Sigma}\mathbf{G}^T\mathbf{M}) = \mathbf{C}\mathbf{H} = (\mathbf{C}\mathbf{H}_1, \mathbf{C}\boldsymbol{\Sigma}\mathbf{G}^T\mathbf{M})$ and hence $\mathbf{C}\mathbf{H}_1 = \mathbf{0}$ and $\mathbf{C}\boldsymbol{\Sigma}\mathbf{G}^T\mathbf{M} = \boldsymbol{\Sigma}\mathbf{G}^T\mathbf{M}$. Since $\mathbf{G}\mathbf{H}_1 = 0$, we have $\mathbf{C} = \boldsymbol{\Sigma}\mathbf{G}^T(\mathbf{G}\boldsymbol{\Sigma}\mathbf{G}^T)^{-1}\mathbf{G}$. The proof is completed by substituting $(\mathbf{0}_{k \times k_1}, \boldsymbol{\Sigma}\mathbf{G}^T\mathbf{M})\mathbf{H}^{-1}$ in (17) with $\boldsymbol{\Sigma}\mathbf{G}^T(\mathbf{G}\boldsymbol{\Sigma}\mathbf{G}^T)^{-1}\mathbf{G}$. □

*Alternative Proof of Theorem 1.* To solve the problem in (3), one may solve an equivalent problem in (5) by defining an invertible transformation matrix $\mathbf{H}$ that satisfies $\mathbf{G}\mathbf{H}_1 = \mathbf{0}_{k_2 \times k_1}$. Let us denote $\boldsymbol{\Lambda} = \mathbf{H}^T\boldsymbol{\Sigma}^{-1}\mathbf{H}$ as a precision matrix that can be partitioned as in (6). To simply the problem in (5), we choose the transformation matrix $\mathbf{H}$ to make $\boldsymbol{z}_1$ and $\boldsymbol{z}_2$ be independent to each other. Since $\boldsymbol{z}$ follows a MVN distribution, $\boldsymbol{z}_1$ and $\boldsymbol{z}_2$ are independent to each other if and only if

$$\boldsymbol{\Lambda}_{12} = \mathbf{H}_1^T\boldsymbol{\Sigma}^{-1}\mathbf{H}_2 = \mathbf{0}.$$

Since $\mathbf{H}_1^T\mathbf{G}^T = \mathbf{0}$ by definition, to make $\boldsymbol{\Lambda}_{12} = \mathbf{0}$, if and only if we have $\mathbf{H}_2$ as

$$\mathbf{H}_2 = \boldsymbol{\Sigma}\mathbf{G}^T\mathbf{M},$$

where $\mathbf{M} \in \mathbb{R}^{k_2 \times k_2}$ is an arbitrary full rank matrix. Accordingly, we have

$$\mathbf{H}^{-1}\boldsymbol{\Sigma}(\mathbf{H}^{-1})^T = \begin{bmatrix} (\mathbf{H}_1^T\boldsymbol{\Sigma}^{-1}\mathbf{H}_1)^{-1} & \mathbf{0} \\ \mathbf{0} & (\mathbf{H}_2^T\boldsymbol{\Sigma}^{-1}\mathbf{H}_2)^{-1} \end{bmatrix}.$$

Thus with $\mathbf{H}$ satisfying $\mathbf{G}\mathbf{H}_1 = \mathbf{0}_{k_2 \times k_1}$ and $\mathbf{H}_2 = \boldsymbol{\Sigma}\mathbf{G}^T\mathbf{M}$, one can transform the original problem in (3) to that in (5), where $\boldsymbol{z}_1$ and $\boldsymbol{z}_2$ are independent and the restrictions $\mathbf{G}\boldsymbol{x} = \boldsymbol{r}$ and $\boldsymbol{z}_2 = (\mathbf{G}\mathbf{H}_2)^{-1}\boldsymbol{r}$ imply each other. Following the naive approach shown in Algorithm 1, one can generate $\boldsymbol{x}$ from (3) as follows

- Find $\mathbf{H} = (\mathbf{H}_1, \mathbf{H}_2)$ with $\mathbf{H}_2 = \boldsymbol{\Sigma}\mathbf{G}^T\mathbf{M}$ and with $\mathbf{H}_1$ satisfying $\mathbf{G}\mathbf{H}_1 = \mathbf{0}_{k_2 \times k_1}$;
- Sample $\boldsymbol{z}_1 \sim \mathcal{N}[(\mathbf{I}_{k_1}, \mathbf{0}_{k_1 \times k_2})\mathbf{H}^{-1}\boldsymbol{\mu}, (\mathbf{H}_1^T\boldsymbol{\Sigma}^{-1}\mathbf{H}_1)^{-1}]$;

- Return $\boldsymbol{x} = \mathbf{H}\begin{bmatrix} \boldsymbol{z}_1 \\ \mathbf{M}^{-1}(\mathbf{G}\boldsymbol{\Sigma}\mathbf{G}^T)^{-1}\boldsymbol{r} \end{bmatrix}$.

However, this naive approach contains intermediate variables that could be computationally expensive to compute. Below we present a method to bypass these intermediate variables. Since the last step could be reexpressed as

$$\begin{aligned}\boldsymbol{x} &= \mathbf{H}_1\boldsymbol{z}_1 + \mathbf{H}_2\mathbf{M}^{-1}(\mathbf{G}\boldsymbol{\Sigma}\mathbf{G}^T)^{-1}\boldsymbol{r} \\ &= \mathbf{H}_1\boldsymbol{z}_1 + \mathbf{H}_2\boldsymbol{z}_2 + \mathbf{H}_2[\mathbf{M}^{-1}(\mathbf{G}\boldsymbol{\Sigma}\mathbf{G}^T)^{-1}\boldsymbol{r} - \boldsymbol{z}_2] \\ &= \mathbf{H}\boldsymbol{z} + \boldsymbol{\Sigma}\mathbf{G}^T(\mathbf{G}\boldsymbol{\Sigma}\mathbf{G}^T)^{-1}\boldsymbol{r} - \boldsymbol{\Sigma}\mathbf{G}^T(\mathbf{M}\boldsymbol{z}_2),\end{aligned}$$

where $\boldsymbol{z} = (\boldsymbol{z}_1^T, \boldsymbol{z}_2^T)^T$ and $\boldsymbol{z}_2 \in \mathbb{R}^{k_2}$ is a vector whose values can be chosen arbitrarily. In addition, since $\mathbf{M}$ is an arbitrary full-rank matrix, we can let

$$\boldsymbol{z}_2 \sim \mathcal{N}[(\mathbf{0}_{k_2\times k_1}, \mathbf{I}_{k_2})\mathbf{H}^{-1}\boldsymbol{\mu}, (\mathbf{H}_2^T\boldsymbol{\Sigma}^{-1}\mathbf{H}_2)^{-1}],$$

which means $\boldsymbol{z} \sim \mathcal{N}[\mathbf{H}^{-1}\boldsymbol{\mu}, \mathbf{H}^{-1}\boldsymbol{\Sigma}(\mathbf{H}^{-1})^T]$, and choose $\mathbf{M}$ to make

$$\mathbf{G}\boldsymbol{x} = \mathbf{G}\mathbf{H}\boldsymbol{z} + \mathbf{G}\boldsymbol{\Sigma}\mathbf{G}^T(\mathbf{G}\boldsymbol{\Sigma}\mathbf{G}^T)^{-1}\boldsymbol{r} - \mathbf{G}\boldsymbol{\Sigma}\mathbf{G}^T(\mathbf{M}\boldsymbol{z}_2) = \boldsymbol{r},$$

which means $\mathbf{G}\boldsymbol{\Sigma}\mathbf{G}^T(\mathbf{M}\boldsymbol{z}_2) = \mathbf{G}\mathbf{H}\boldsymbol{z}$. Thus we have

$$\boldsymbol{x} = \mathbf{H}\boldsymbol{z} + \boldsymbol{\Sigma}\mathbf{G}^T(\mathbf{G}\boldsymbol{\Sigma}\mathbf{G}^T)^{-1}(\boldsymbol{r} - \mathbf{G}\mathbf{H}\boldsymbol{z}).$$

In addition, since if $\boldsymbol{z} \sim \mathcal{N}[\mathbf{H}^{-1}\boldsymbol{\mu}, \mathbf{H}^{-1}\boldsymbol{\Sigma}(\mathbf{H}^{-1})^T]$, then $\boldsymbol{y} = \mathbf{H}\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. Therefore, without the need to compute any intermediate variables, one may use Algorithm 2 to generate $\boldsymbol{x}$ from the hyperplane truncated MVN distribution.

□

*Proof of Theorem 2.* Using the matrix inversion lemma on (12), we have

$$\begin{aligned}&p(\boldsymbol{x}_1) \propto \exp\left[-\frac{1}{2}(\boldsymbol{x}_1 - \boldsymbol{\mu}_1)^T\left(\boldsymbol{\Sigma}_{11} - \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}\boldsymbol{\Sigma}_{21}\right)^{-1}(\boldsymbol{x}_1 - \boldsymbol{\mu}_1)\right], \\ &\propto \exp\left\{-\frac{1}{2}(\boldsymbol{x}_1 - \boldsymbol{\mu}_1)^T\left[\boldsymbol{\Sigma}_{11}^{-1} + \boldsymbol{\Sigma}_{11}^{-1}\boldsymbol{\Sigma}_{12}\left(\boldsymbol{\Sigma}_{22} - \boldsymbol{\Sigma}_{21}\boldsymbol{\Sigma}_{11}^{-1}\boldsymbol{\Sigma}_{12}\right)^{-1}\boldsymbol{\Sigma}_{21}\boldsymbol{\Sigma}_{11}^{-1}\right](\boldsymbol{x}_1 - \boldsymbol{\mu}_1)\right\}.\end{aligned} \tag{18}$$

Using (10), we have $\mathbf{G}\boldsymbol{\mu} = \mathbf{G}_1\boldsymbol{\mu}_1 + \mathbf{G}_2\boldsymbol{\mu}_2 = \boldsymbol{r}$. Since $\mathbf{G}\boldsymbol{x} = \boldsymbol{r}$, we further have $\mathbf{G}(\boldsymbol{x} - \boldsymbol{\mu}) = 0$ and hence $\mathbf{G}_1(\boldsymbol{x}_1 - \boldsymbol{\mu}_1) = -\mathbf{G}_2(\boldsymbol{x}_2 - \boldsymbol{\mu}_2)$. Therefore, given the construction of $\boldsymbol{\mu}_2$ as in (10), we can replace the equality constraint $\mathbf{G}\boldsymbol{x} = \boldsymbol{r}$ on $\boldsymbol{x}$ by

requiring $(\boldsymbol{x}_2 - \boldsymbol{\mu}_2) = -\mathbf{G}_2^{-1}\mathbf{G}_1(\boldsymbol{x}_1 - \boldsymbol{\mu}_1)$. Using this equivelent constraint together with (3), we have

$$
\begin{aligned}
& p(\boldsymbol{x} \,|\, \boldsymbol{\mu}, \tilde{\boldsymbol{\Sigma}}, \mathbf{G}, \boldsymbol{r}) \\
\propto & \exp\left[-\frac{1}{2}(\boldsymbol{x}_1 - \boldsymbol{\mu}_1)^T \boldsymbol{\Sigma}_{11}^{-1}(\boldsymbol{x}_1 - \boldsymbol{\mu}_1) - \frac{1}{2}(\boldsymbol{x}_2 - \boldsymbol{\mu}_2)^T \tilde{\boldsymbol{\Sigma}}_{22}^{-1}(\boldsymbol{x}_2 - \tilde{\boldsymbol{\mu}}_2)\right] \delta(\mathbf{G}\boldsymbol{x} = \boldsymbol{r}) \\
\propto & \exp\left\{-\frac{1}{2}(\boldsymbol{x}_1 - \boldsymbol{\mu}_1)^T \left[\boldsymbol{\Sigma}_{11}^{-1} + \mathbf{G}_1^T(\mathbf{G}_2^{-1})^T \tilde{\boldsymbol{\Sigma}}_{22}^{-1}\mathbf{G}_2^{-1}\mathbf{G}_1\right](\boldsymbol{x}_1 - \boldsymbol{\mu}_1)\right\} \\
& \times \delta\left[\boldsymbol{x}_2 = \boldsymbol{\mu}_2 - \mathbf{G}_2^{-1}\mathbf{G}_1(\boldsymbol{x}_1 - \boldsymbol{\mu}_1)\right] \\
= & \mathcal{N}\left\{\boldsymbol{x}_1; \boldsymbol{\mu}_1, \left[\boldsymbol{\Sigma}_{11}^{-1} + \mathbf{G}_1^T(\mathbf{G}_2^{-1})^T \tilde{\boldsymbol{\Sigma}}_{22}^{-1}\mathbf{G}_2^{-1}\mathbf{G}_1\right]^{-1}\right\} \\
& \times \delta\left[\boldsymbol{x}_2 = \boldsymbol{\mu}_2 - \mathbf{G}_2^{-1}\mathbf{G}_1(\boldsymbol{x}_1 - \boldsymbol{\mu}_1)\right] \qquad (19)
\end{aligned}
$$

It is clear that the marginal distribution of $\boldsymbol{x}_1$ in (19) matches the conditional distribution of $\boldsymbol{x}_1$ in (18) if we further construct $\tilde{\boldsymbol{\Sigma}}_{22}$ using (11). □

*Proof of Corollary 4.* Applying Theorem 1 to Corollary 3, we can generate $\boldsymbol{x}$ with

- Sample $\boldsymbol{y} \sim \mathcal{N}\left(\boldsymbol{\mu}, \tilde{\boldsymbol{\Sigma}}\right)$;
- Return $\boldsymbol{x} = \boldsymbol{y} + \tilde{\boldsymbol{\Sigma}}\mathbf{G}^T[\mathbf{G}\tilde{\boldsymbol{\Sigma}}\mathbf{G}^T]^{-1}(\boldsymbol{r} - \mathbf{G}\boldsymbol{y})$.

Since $\boldsymbol{\mu} = \begin{bmatrix}\boldsymbol{\mu}_1 \\ \mathbf{0}\end{bmatrix}$, $\tilde{\boldsymbol{\Sigma}} = \begin{bmatrix}\boldsymbol{\Sigma}_{11} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Sigma}_{22} - \boldsymbol{\Sigma}_{21}\boldsymbol{\Sigma}_{11}^{-1}\boldsymbol{\Sigma}_{12}\end{bmatrix}$, $\mathbf{G} = (\mathbf{G}_1, \mathbf{G}_2) = (\boldsymbol{\Sigma}_{21}\boldsymbol{\Sigma}_{11}^{-1}, \mathbf{I}_{k_2})$, and $\boldsymbol{r} = \boldsymbol{\Sigma}_{21}\boldsymbol{\Sigma}_{11}^{-1}\boldsymbol{\mu}_1$, we have $\tilde{\boldsymbol{\Sigma}}\mathbf{G}^T = \begin{bmatrix}\boldsymbol{\Sigma}_{12} \\ \boldsymbol{\Sigma}_{22} - \boldsymbol{\Sigma}_{21}\boldsymbol{\Sigma}_{11}^{-1}\boldsymbol{\Sigma}_{12}\end{bmatrix}$ and $[\mathbf{G}\tilde{\boldsymbol{\Sigma}}\mathbf{G}^T]^{-1} = \boldsymbol{\Sigma}_{22}^{-1}$. Since $\tilde{\boldsymbol{\Sigma}}$ is block diagonal, we can independently sample $\boldsymbol{y}_1$ and $\boldsymbol{y}_2$ as $\boldsymbol{y}_1 \sim \mathcal{N}(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_{11})$ and $\boldsymbol{y}_2 \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_{22} - \boldsymbol{\Sigma}_{21}\boldsymbol{\Sigma}_{11}^{-1}\boldsymbol{\Sigma}_{12})$, respectively, with which we can further sample $\boldsymbol{x}$ as

$$
\begin{bmatrix}\boldsymbol{x}_1 \\ \boldsymbol{x}_2\end{bmatrix} = \begin{bmatrix}\boldsymbol{y}_1 \\ \boldsymbol{y}_2\end{bmatrix} + \begin{bmatrix}\boldsymbol{\Sigma}_{12} \\ \boldsymbol{\Sigma}_{22} - \boldsymbol{\Sigma}_{21}\boldsymbol{\Sigma}_{11}^{-1}\boldsymbol{\Sigma}_{12}\end{bmatrix} \boldsymbol{\Sigma}_{22}^{-1}(\boldsymbol{\Sigma}_{21}\boldsymbol{\Sigma}_{11}^{-1}\boldsymbol{\mu}_1 - \boldsymbol{\Sigma}_{21}\boldsymbol{\Sigma}_{11}^{-1}\boldsymbol{y}_1 - \boldsymbol{y}_2).
$$

Thus we can let $\boldsymbol{x}_1 = \boldsymbol{\mu}_1 + \boldsymbol{y}_1' - \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}\left(\boldsymbol{\Sigma}_{21}\boldsymbol{\Sigma}_{11}^{-1}\boldsymbol{y}_1' + \boldsymbol{y}_2\right)$, where $\boldsymbol{y}_1' = \boldsymbol{y}_1 - \boldsymbol{\mu}_1 \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_{11})$. □

*Proof of Corollary 5.* Using the matrix inversion lemma, we have

$$
\boldsymbol{\Sigma}_\beta = \mathbf{A}^{-1} - \mathbf{A}^{-1}\boldsymbol{\Phi}^T(\boldsymbol{\Omega}^{-1} + \boldsymbol{\Phi}\mathbf{A}^{-1}\boldsymbol{\Phi}^T)^{-1}\boldsymbol{\Phi}\mathbf{A}^{-1}. \qquad (20)
$$

The proof is completed by using Corollary 4 with $\boldsymbol{\mu}_1 = \boldsymbol{\mu}_\beta$, $\boldsymbol{\Sigma}_{11} = \mathbf{A}^{-1}$, $\boldsymbol{\Sigma}_{12} = \mathbf{A}^{-1}\boldsymbol{\Phi}^T$, and $\boldsymbol{\Sigma}_{22} = \boldsymbol{\Omega}^{-1} + \boldsymbol{\Phi}\mathbf{A}^{-1}\boldsymbol{\Phi}^T$. □

*Proof of Corollary 6.* The proof is completed by applying Corollary 5 with

$$\begin{aligned}
\boldsymbol{\mu}_\beta &= [\mathbf{A}^{-1} - \mathbf{A}^{-1}\boldsymbol{\Phi}^T(\boldsymbol{\Omega}^{-1} + \boldsymbol{\Phi}\mathbf{A}^{-1}\boldsymbol{\Phi}^T)^{-1}\boldsymbol{\Phi}\mathbf{A}^{-1}]\boldsymbol{\Phi}^T\boldsymbol{\Omega}\boldsymbol{t} \\
&= \mathbf{A}^{-1}\boldsymbol{\Phi}^T(\boldsymbol{\Omega}^{-1} + \boldsymbol{\Phi}\mathbf{A}^{-1}\boldsymbol{\Phi}^T)^{-1}[(\boldsymbol{\Omega}^{-1} + \boldsymbol{\Phi}\mathbf{A}^{-1}\boldsymbol{\Phi}^T)\boldsymbol{\Omega}\boldsymbol{t} - \boldsymbol{\Phi}\mathbf{A}^{-1}\boldsymbol{\Phi}^T\boldsymbol{\Omega}\boldsymbol{t}] \\
&= \mathbf{A}^{-1}\boldsymbol{\Phi}^T(\boldsymbol{\Omega}^{-1} + \boldsymbol{\Phi}\mathbf{A}^{-1}\boldsymbol{\Phi}^T)^{-1}\boldsymbol{t}.
\end{aligned}$$

□