Hierarchical-block conditioning approximations for high-dimensional multivariate normal probabilities

Hyunseok Yang, Kyeongwon Lee, and Songhyun Kim

December 8, 2019

Department of Statistics, SNU

Table of contents

Introduction

Introduction

- The computation of the multivariate normal (MVN) probability

$$\Phi_n(\mathbf{a}, \mathbf{b}; 0, \mathbf{\Sigma}) = \int_a^b \frac{1}{\sqrt{(2\pi)^n |\mathbf{\Sigma}|}} \exp\left(-\frac{1}{2}\mathbf{x}^T \mathbf{\Sigma}^{-1} \mathbf{x}\right) d\mathbf{x}, \qquad (1)$$

  where **a** and **b** are integration limits, the mean vector μ is assumed to be 0, $\mathbf{\Sigma}$ is a positive-definite covariance matrix, is required for a variety of applications.
- Various methods to compute MVN probability are suggested such as Richtmyer Quasi-Monte Carlo(QMC) (Genz and Bretz, 2009)
- However, In high-dimensional settings (large n), it is hard to compute (1) directly.
- We review new approaches proposed by Cao et al. (2019) to approximate high-dimensional multivariate normal probability (1) using the hierarchical matrix $\mathcal{H}$ (Hackbusch, 2015) for the covariance matrix $\mathbf{\Sigma}$.

Motivation

The methods are based on

1. the bivariate conditioning method (Trinh and Genz, 2015) and
2. the hierarchical QMC method (Genton et al., 2018).

Multidimensional Conditioning Approximations

Quasi Monte Carlo Method

- MonteCarlo Error bound : $O(N^{-1/2})$ for monte carlo(MC) method
- Genz and Bretz (2009) claimed independent sample points is the reason of slow convergence.
- Via employing low discrepancy sets for sequence, QMC is asymptotically efficient than MC.
- With $\mathbf{\Delta} \sim U[0,1]^n$,

$$L_N = \{\mathbf{z} + \mathbf{\Delta} \text{ mod } 1 : \mathbf{z} \in K_N\}$$

$$K_N = \{i\mathbf{q} \text{ mod } 1, i = 1, \cdots, N\}$$

  where $\mathbf{q} = \sqrt{\mathbf{p}}$ and $\mathbf{p}$ is set of prime numbers.
- Since square root of prime numbers is irrational and linear independent over the rational numbers,

Quasi Monte Carlo Method

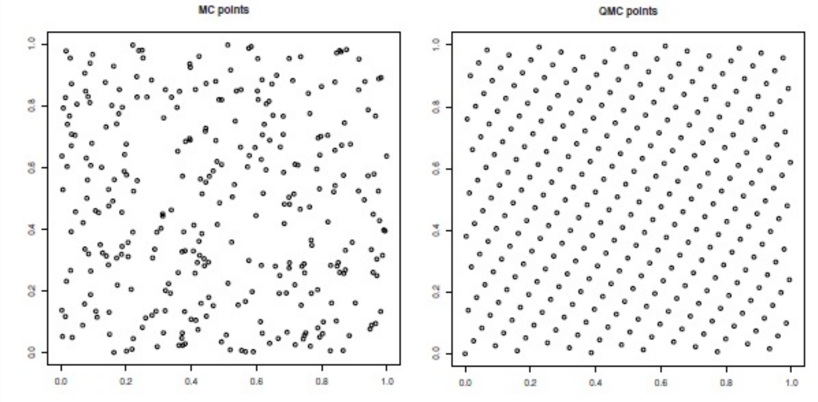


Figure 1: Comparison of MC and QMC sample points(Genz and Bretz, 2009)

Quasi Monte Carlo Method

$$
\begin{aligned}
\mathbf{\Phi}_n(\mathbf{a} \le \mathbf{x} \le \mathbf{b}; \mathbf{\Sigma}) &= \mathbf{\Phi}_n(a \le \mathbf{L}\mathbf{y} \le \mathbf{b}; I_n) \\
&= \int_{a_1 \le l_{11} y_1 \le b_1} \phi(y_1) \cdots \int_{a_n \le \mathbf{l}_n^t \mathbf{y} \le b_n} \phi(y_n) d\mathbf{y} \\
&= \int_{\tilde{a}_1}^{\tilde{b}_1} \phi(y_1) \int_{\tilde{a}_2(y_1)}^{\tilde{b}_2(y_1)} \phi(y_2) \cdots \int_{\tilde{a}_n(y_1,\cdots,y_{n-1})}^{\tilde{b}_n(y_1,\cdots,y_{n-1})} \phi(y_n) d\mathbf{y} \\
\text{with } \tilde{a}_i(y_1, \cdots, y_{i-1}) &= \frac{a_i - \sum_{j=1}^{i-1} l_{ij} y_j}{l_{ii}} \\
\text{and } (\tilde{b}_i(y_1, \cdots, y_{i-1})) &= \frac{b_i - \sum_{j=1}^{i-1} l_{ij} y_j}{l_{ii}} \\
&= \int_{\Phi(\tilde{a}_1)}^{\Phi(\tilde{b}_1)} \int_{\Phi(\tilde{a}_2(\Phi^{-1}(z_1)))}^{\Phi(\tilde{b}_2(\Phi^{-1}(z_1)))} \cdots \int_{\Phi(\tilde{a}_n(\Phi^{-1}(z_1),\cdots,\Phi^{-1}(z_{n-1})))}^{\Phi(\tilde{b}_n(\Phi^{-1}(z_1),\cdots,\Phi^{-1}(z_{n-1})))} d\mathbf{z}(y_i = \Phi^{-1}(z_i)) \\
&= (e_1 - d_1) \int_0^1 (e_2(w_1) - d_2(w_1)) \cdots \\
&\quad \int_0^1 (e_n(w_1, \cdots, w_{n-1}) - d_n(w_1, \cdots, w_{n-1}))) \int_0^1 d\mathbf{w} \\
\text{with } z_i &= d_i + (e_i - d_i) w_i \qquad (2)
\end{aligned}
$$

Quasi Monte Carlo Method

```
procedure MVN(μ, Σ, a, b, ns, N)
    L = cholesky(Σ)
    a = a − μ; b = b − μ
    T = 0, N = 0, V = 0
    p = vector of primes less than (5n log n+1)/4 ;q = √p
    P = 1_{ns}
    ans = 0
    for i = 1, · · · , ns do
        I_i = 0, Δ ∼ U(0, 1)^n
        for j = 1, · · · , N do
            X[1 : n, j] = (j + 1)q + Δ
            X[1 : n, j] = 2|X[1 : n, j] − floor(X[1 : n, j])| − 1
        end for
        sample = O_{n,N}
        s, c, d, dc, P = 0_N
        for j = 1, · · · , n do
            if j > 1 then
                c = min(1, c + X[j − 1, :] ⊙ dc)
                sample[i − 1, 1 : N] = Φ^{−1}(c)
                s = sample[1 : i − 1, 1 : N]^T L[1 : i − 1, i]
            end if
            P∗ = Φ((b−s)/L[i,i]) − Φ((a−s)/L[i,i])
        end for
        ans+ = mean(P)
    end for
    return ans/ns
end procedure
```

lined 1: Multivariate Normal Probability with Quasi Monte Carlo Method

Conditioning Approximation

Mendell and Elston (1974), Kamakura (1989), and Trinh and Genz (2015) exploit Cholesky factors from LDL decomposition rather than dealing with original covariance matrix. Biviarate example is follow.

$$\boldsymbol{\Sigma} = \begin{pmatrix} \boldsymbol{\Sigma}_{1,1} & \mathbf{R}^T \\ \mathbf{R} & \hat{\boldsymbol{\Sigma}} \end{pmatrix}, \text{ with } \mathbf{L} = \begin{pmatrix} \mathbf{I}_2 & \mathbf{O} \\ 1:\mathbf{M} & \mathbf{L} \end{pmatrix} \text{ and } \mathbf{D} = \begin{pmatrix} \mathbf{D}_1 & \mathbf{O} \\ \mathbf{O} & \widehat{\mathbf{D}} \end{pmatrix}$$

,where $\boldsymbol{\Sigma}_{1,1}, \mathbf{D}_1$ is a 2×2 matrix. From $\mathbf{D}_1 = \boldsymbol{\Sigma}_{\mathbf{1},\mathbf{1}}$, $\mathbf{M} = \mathbf{R}\mathbf{D}_1^{-1}$, $\widehat{\mathbf{D}} = \hat{\boldsymbol{\Sigma}} - \mathbf{M}\mathbf{D}_1\mathbf{M}^T$

$$\boldsymbol{\Phi}_n(\mathbf{a}, \mathbf{b}; \mathbf{0}, \boldsymbol{\Sigma}) = \frac{1}{\sqrt{|\mathbf{D}|(2\pi)^n}} \int_{\alpha_1}^{\beta_1} \int_{\alpha_2}^{\beta_2} e^{-\frac{1}{2}\mathbf{x_2}^T \mathbf{D}_1^{-1} \mathbf{x}_2} \cdots \int_{\alpha_{2k-1}}^{\beta_{2k-1}} \int_{\alpha_{2k}}^{\beta_{2k}} e^{-\frac{1}{2}\mathbf{x_{2k}}^T \mathbf{D}_1^{-1} \mathbf{x}_{2k}} \tag{3}$$

Conditioning Approximation

Cao et al. (2019) generalizes bivariate method of Trinh and Genz (2015) to d-dimensional. Algorithms and details are following.

procedure LDL($\mathbf{\Sigma}$)
  $\mathbf{L} \leftarrow \mathbf{I}_m$, $\mathbf{D} \leftarrow \mathbf{O}_m$
  for $i = 1 : d : m - d + 1$ **do**
    $\mathbf{D}[i : i + d - 1, i : i + d - 1] \leftarrow \mathbf{\Sigma}[i : i + d - 1, i : i + d - 1]$
    $\mathbf{L}[i + d : m, i : i + d - 1] \leftarrow \mathbf{\Sigma}[i + d : m, i : i + d - 1]\mathbf{D}^{-1}[i : i + d - 1, i : i + d - 1]$
    $\mathbf{\Sigma}[i + d : m, i + d : m] \leftarrow \mathbf{\Sigma}[i + d : m, i + d : m] - \mathbf{L}[i + d : m, i : i + d - 1]\mathbf{D}^{-1}[i : i + d - 1, i : i + d - 1]\mathbf{L}[i : i + d - 1, i + d : m]$
    if $i + d < m$ **then**
      $\mathbf{D}[i + d : m, i + d : m] \leftarrow \mathbf{\Sigma}[i + d : m, i + d : m]$
    end if
  end for
  return $\mathbf{L}$ and $\mathbf{D}$
end procedure

lined 2: LDL decomposition

Conditioning Approximation

When $s = \frac{m}{d}$ is integer, results of Algorithm 2, $\mathbf{L}, \mathbf{D}$ can be written as

$$\mathbf{L} = \begin{pmatrix} \mathbf{I}_d & \mathbf{O}_d & \cdots & \mathbf{O}_d \\ \mathbf{L}_{2,1} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \mathbf{I}_d & \mathbf{O}_d \\ \mathbf{L}_{s,1} & \cdots & \mathbf{L}_{s,s-1} & \mathbf{I}_d \end{pmatrix}, \mathbf{D} = \begin{pmatrix} \mathbf{D}_1 & \mathbf{O}_d & \cdots & \mathbf{O}_d \\ \mathbf{O}_d & \ddots & \ddots & \vdots \\ \vdots & \ddots & \mathbf{D}_{s-1} & \mathbf{O}_d \\ \mathbf{O}_d & \cdots & \mathbf{O}_d & \mathbf{D}_s \end{pmatrix}$$

with d-dimensional identitiy matrix $\mathbf{I}_d$ and d-dimensional zero matrix $\mathbf{O}_d$ and d-dimensional positive-definite matrix $\mathbf{D}_1, \cdots, \mathbf{D}_s$. As in (3), tranformation, $Y = LX$ provides m-dimensional multivariate normal prabability as the product of s d-dimensional multivariate normal probabilities as below.

$$\mathbf{\Phi}_{\boldsymbol{m}}(\mathbf{a}, \mathbf{b}; \mathbf{0}, \mathbf{\Sigma}) = \int_{\alpha_1}^{\beta_1} \phi_d(\mathbf{y}_1; \mathbf{D}_1) \int_{\alpha_2}^{\beta_2} \phi_d(\mathbf{y}_2; \mathbf{D}_2) \cdots \int_{\alpha_s}^{\beta_s} \phi_d(\mathbf{y}_s; \mathbf{D}_s) d\mathbf{y}_s \cdots d\mathbf{y}_2 d\mathbf{y}_1 \quad (4)$$

,where $\boldsymbol{\alpha}_i = \mathbf{a}_i - \sum_{j=1}^{i-1} \mathbf{L}_{ij}\mathbf{y}_j, \boldsymbol{\beta}_i = \mathbf{b}_i - \sum_{j=1}^{i-1} \mathbf{L}_{ij}\mathbf{y}_j$

Conditioning Approximation

procedure $\texttt{CMVN}(\mathbf{\Sigma}, \mathbf{a}, \mathbf{b}, d)$
    $\mathbf{y} \leftarrow \mathbf{0}, P \leftarrow 1$
    for $i = 1 : s$ **do**
        $j \leftarrow (i-1)d$
        $\mathbf{g} \leftarrow \mathbf{L}[j+1 : j+d, 1 : j]\mathbf{y}[1 : j]$
        $\boldsymbol{\alpha} \leftarrow \mathbf{a}[j+1 : j+d] - \mathbf{g}$
        $\boldsymbol{\beta} \leftarrow \mathbf{b}[j+1 : j+d] - \mathbf{g}$
        $\mathbf{D}' \leftarrow \mathbf{D}[j+1 : j+d, j+1 : j+d]$
        $P \leftarrow P \cdot \mathbf{\Phi}_d(\boldsymbol{\alpha}, \boldsymbol{\beta}; \mathbf{0}, \mathbf{D}')$
        $\mathbf{y}[j+1 : j+d] \leftarrow E[\mathbf{Y}']$
    end for
    return P and $\mathbf{y}$
end procedure

lined 3: d-dimensional conditioning algorithm

Multidimensional Truncated Expectations

The truncated expectation is expressed as

$$E(X^{e_j}) = \frac{1}{\mathbf{\Phi}(\mathbf{a}, \mathbf{b}; \boldsymbol{\mu}, \mathbf{\Sigma})} \int_{\mathbf{a}}^{\mathbf{b}} x_j \phi_d(\mathbf{x}; \boldsymbol{\mu}, \mathbf{\Sigma}) d\mathbf{x} = \frac{1}{\mathbf{\Phi}(\mathbf{a}, \mathbf{b}; \boldsymbol{\mu}, \mathbf{\Sigma})} F_j^d(\mathbf{a}, \mathbf{b}; \boldsymbol{\mu}, \mathbf{\Sigma})$$

Theorem
(Kan and Robotti, 2017)

$$F_j^d(\mathbf{a}, \mathbf{b}; \boldsymbol{\mu}, \mathbf{\Sigma}) = \mu_j \mathbf{\Phi}_d(\mathbf{a}, \mathbf{b}; \boldsymbol{\mu}, \mathbf{\Sigma}) + \mathbf{e}_j^T \mathbf{\Sigma} \mathbf{c}$$

,where c is a vector with lth component defined as

$$c_l = \phi_1(a_l; \mu_l, \sigma_l^2) \Phi_{d-1}(\mathbf{a}_{-l}, \mathbf{b}_{-l}; \hat{\boldsymbol{\mu}}^1, \hat{\mathbf{\Sigma}}_l) - \phi_1(b_l; \mu_l, \sigma_l^2) \Phi_{d-1}(\mathbf{a}_{-l}, \mathbf{b}_{-l}; \hat{\boldsymbol{\mu}}^2, \hat{\mathbf{\Sigma}}_l)$$

$$\hat{\boldsymbol{\mu}}_l^1 = \mu_{-l} + \mathbf{\Sigma}_{-l,l} \frac{a_l - \mu_l}{\sigma_l^2}, \hat{\boldsymbol{\mu}}_l^2 = \mu_{-l} + \mathbf{\Sigma}_{-l,l} \frac{b_l - \mu_l}{\sigma_l^2},$$

$$\hat{\mathbf{\Sigma}}_l = \mathbf{\Sigma}_{-l,-l} - \frac{1}{\sigma_l^2} \mathbf{\Sigma}_{-l,l} \mathbf{\Sigma}_{l,-l}$$

Theorem 1 has same form with bivariate version of Trinh and Genz (2015) with $d = 2$ and it allows us to calculate $E[Y']$ in Algorithm 3 with $\mathbf{\Phi}$ which can be obtained with quasi monte calro method proposed by Genz (1992)

Multidimensional Conditioning Approximation with Univariate Reordering

Appropriate integration order on conditioning algorithm possibly improves estiation accuracy

- Schervish (1984) : integral with shortest integration interval widths be the outermost integration variables
- Gibson et al. (1994) : variables which have smallest expected values be the outermost integration variables.
  Since innermost integrals which have smaller variation have the most influence with this order, overall variance reduces.
- Trinh and Genz (2015) also employs this ordering, and Cao et al. (2019) generalized it to d-dimensional problem.

Multidimensional Conditioning Approximation with Univariate Reordering

procedure RCMVN($\boldsymbol{\Sigma}$, $\mathbf{a}$, $\mathbf{b}$, d)

  $\mathbf{y} \leftarrow \mathbf{0}$, $\mathbf{C} \leftarrow \boldsymbol{\Sigma}$

  for $i = 1 : m$ **do**

    if $i > 1$ **then**

      $\mathbf{y}[i-1] \leftarrow \frac{\phi(a')-\phi(b')}{\Phi(b')-\Phi(a')}$

    end if

    $j \leftarrow \text{argmin}_{i \leq j \leq m}\{\Phi(\frac{\mathbf{b}[j]-\mathbf{C}[j,1:i-1]\mathbf{y}[1:i-1]}{\sqrt{\boldsymbol{\Sigma}[j,j]-\mathbf{C}[j,1:i-1]\mathbf{C}^T[j,1:i-1]}}) - \Phi(\frac{\mathbf{a}[j]-\mathbf{C}[j,1:i-1]\mathbf{y}[1:i-1]}{\sqrt{\boldsymbol{\Sigma}[j,j]-\mathbf{C}[j,1:i-1]\mathbf{C}^T[j,1:i-1]}})\}$

    $\boldsymbol{\Sigma}[:,(i,j)] \leftarrow \boldsymbol{\Sigma}[:,(j,i)];\boldsymbol{\Sigma}[(i,j),:] \leftarrow \boldsymbol{\Sigma}[(j,i),:]$

    $\mathbf{C}[:,(i,j)] \leftarrow \mathbf{C}[:,(j,i)];\mathbf{C}[(i,j),:] \leftarrow \mathbf{C}[(j,i),:]$

    $\mathbf{a}[(i,j)] = \mathbf{a}[(j,i)]$

    $\mathbf{b}[(i,j)] = \mathbf{b}[(j,i)]$

    $\mathbf{C}[i,i] \leftarrow \sqrt{\boldsymbol{\Sigma}[i,i] - \mathbf{C}[i,1:i-1]\mathbf{C}^T[i,1:i-1]}$

    $\mathbf{C}[j,i] \leftarrow \frac{\boldsymbol{\Sigma}[j,i]-\mathbf{C}[i,1:i-1]\mathbf{C}^T[j,1:i-1]}{\mathbf{C}[i,i]}$, for $j = i+1, \cdots, m$

    $a' = \frac{\mathbf{a}[i]-\mathbf{C}[i,1:i-1]y[1:i-1]}{\mathbf{C}[\mathbf{i},\mathbf{i}]}$

    $b' = \frac{\mathbf{b}[i]-\mathbf{C}[i,1:i-1]y[1:i-1]}{\mathbf{C}[\mathbf{i},\mathbf{i}]}$

  end for

  return CMVN($\boldsymbol{\Sigma}$, $\mathbf{a}$, $\mathbf{b}$, d) as in Algorithm 3

end procedure

lined 4: d-dimensional conditioning algorithm with univariate reordering

Hierarchical-Block Approximation

Hierarchical Cholesky Decomposition

Hackbusch (2015) proposed hiarchical matrix and its cholesky decomposition method. $A = LU$ have the structure

$$\begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} = \begin{pmatrix} L_{11} & O \\ L_{21} & L_{22} \end{pmatrix} \begin{pmatrix} L_{11}^T & L_{12}^T \\ O & L_{22}^T \end{pmatrix}$$

with lower triangular matrix L_{11}, L_{22}. It leads to four tasks:

(a) compute L_{11} via Cholesky decomposition of A_{11}

(b) compute L_{12} from $L_{21}L_{11}^T = A_{21}$

(c) low rank approximation of $L_{12} = UV^T$

(d) compute L_{22} via Cholesky decomposition of $A_{22} - L_{21}L_{21}^T$

We have applied low rank approximation with svd to (c) each block of its decomposition to make implementation efficiently and save storage while accuracy is preserved. : i.e. $A = UDV^T = \sum_{i=1}^{n} d_i u_i v_i^T \approx \sum_{i=1}^{k} d_i u_i v_i^T$.

Hierarchical Cholesky Decomposition

Hierachical cholesky decomposition of $n \times n$ matrix into $m \times m$ blocks is implemented like below.

procedure HCHOL(A, n,m,rank)
    for $i = 1 : log_2(\frac{n}{m})$ **do**
        $nb = n/2^i$
        x = 0, y = nb
        for $j = 1 : 2^{i-1}$ **do**
            $\mathbf{U}, \mathbf{D}, \mathbf{V} = lowrankSVD(A[xbegin + 1 : xbegin + nb, ybegin + 1 : ybegin + nb], rank)$
            $\mathbf{A}[x + 1 : x + nb, y + 1 : y + rank] = \mathbf{UD}$
            $\mathbf{A}[x + 1 : x + nb, y + rank + 1 : y + nb] = \mathbf{O}$
            $\mathbf{A}[y + 1 : y + nb, x + 1 : x + rank] = \mathbf{VD}$
            $\mathbf{A}[y + 1 : y + nb, x + rank + 1 : x + nb] = \mathbf{O}$
            $x+ = 2nb$, $y+ = 2nb$
        end for
    end for
end procedure

lined 5: Hierachical cholesky decomposition

The Hierarchical-Block Conditioning Method

Let $\phi_m(\mathbf{x}; \mathbf{\Sigma})$ be a pdf of the m-dimensional normal distribution $N(\mathbf{0}, \mathbf{\Sigma})$ and $(\mathbf{B}, \mathbf{U}\mathbf{V}^T)$ be the hierarchical Cholesky decompostion of the covariance matrix $\mathbf{\Sigma}$. Then,

$$\Phi_n(\mathbf{a}, \mathbf{b}; \mathbf{0}, \mathbf{\Sigma}) = \int_{\mathbf{a}_1'}^{\mathbf{b}_1'} \phi_m(\mathbf{x}_1; \mathbf{B}_1\mathbf{B}_1^T) \cdots \int_{\mathbf{a}_r'}^{\mathbf{b}_r'} \phi_r(\mathbf{x}_r; \mathbf{B}_r\mathbf{B}_r^T) d\mathbf{x}_r \cdots d\mathbf{x}_1. \tag{5}$$

,where $\mathbf{a}'$, $\mathbf{b}'$, $i = 1, \cdots, r$, are the corresponding segments of the updated $\mathbf{a}$ and $\mathbf{b}$.

Note the probabilities $\Phi_m(\mathbf{a}_i, \mathbf{b}_i; \mathbf{0}, \mathbf{B}_i\mathbf{B}_i^T)$ can be computed using

1. Quasi-Monte Carlo method (`HMVN`, Method 1 in Cao et al. (2019))
2. d-dimensional conditioning algorithm (`HCMVN`, Method 2 in Cao et al. (2019))
3. d-dimensional conditioning algorithm with univariate reordering (`HRCMVN`, Method 3 in Cao et al. (2019)).

These methods are more effective and easily parallelizable than the classical methods.

The Hierarchical-Block Conditioning Method

procedure HMVN(a, b, $\mathbf{\Sigma}$, d)
  $\mathbf{x} \leftarrow \mathbf{0}$ and $P \leftarrow 1$
  $[\mathbf{B}, \mathbf{UV}] \leftarrow$ `choldecomp_hmatrix`$(\mathbf{\Sigma})$
  for $i = 1 : r$ **do**
    $j \leftarrow (i-1)m$
    if $i > 1$ **then**
      $o_r \leftarrow$ row offset of $\mathbf{U}_{i-1}\mathbf{V}_{i-1}^T$
      $o_c \leftarrow$ column offset of $\mathbf{U}_{i-1}\mathbf{V}_{i-1}^T$
      $l \leftarrow \dim(\mathbf{U}_{i-1}\mathbf{V}_{i-1}^T)$
      $\mathbf{g} \leftarrow \mathbf{U}_{i-1}\mathbf{V}_{i-1}^T\mathbf{x}[o_c + 1 : o_c + l]$
      $\mathbf{a}[o_r + 1 : o_r + l] = \mathbf{a}[o_r + 1 : o_r + l] - \mathbf{g}$
      $\mathbf{b}[o_r + 1 : o_r + l] = \mathbf{a}[o_r + 1 : o_r + l] - \mathbf{g}$
    end if
    $\mathbf{a}_i \leftarrow \mathbf{a}[j + 1 : j + m]$
    $\mathbf{b}_i \leftarrow \mathbf{b}[j + 1 : j + m]$
    $P = P * \Phi_m(\mathbf{a}_i, \mathbf{b}_i; \mathbf{0}, \mathbf{B}_i\mathbf{B}_i^T)$
    $\mathbf{x}[j + 1 : j + m] \leftarrow \mathbf{B}_i^{-1}E(\mathbf{X}_i)$
  end for
end procedure

lined 6: Hierarchical-block conditioning algorithm

Computational Complexity

$M(\cdot)$ denotes the complexity of the QMC simulation in the given dimension.
Table 1 shows that the time efficiency of the d-dimensional conditioning algorithm mainly comes from lowering the dimension in which the QMC simulation is performed.

	MVN prob	Trunc exp	Upd limits
`HMVN`	$\frac{n}{m}M(m)$	$2nM(m) + O(nm^2)$	$O(mn + knlog(n/m))$
`HCMVN`	$\frac{n}{d}M(d) + O(m^2n)$	$2nM(d) + O(nd^2)$	$O(mn + knlog(n/m))$
`HRCMVN`	$\frac{n}{d}M(d) + O(m^2n)$	$2nM(d) + O(nd^2)$	$O(mn + knlog(n/m))$

Table 1: Complexity decomposition of the `HMVN`, `HCMVN`, and `HRCMVN`

- The updating cost is independent of the method.
- The complexity of the univariate reordering is $O(m^2n)$, the same as the complexity of computing the MVN probabilities in `HCMVN`
- Since `HCMVN` and `HRCMVN` perform the QMC simulation in d-dimensions, these two methods are not greatly affected by the choice of m.

Block Reordering

Block Reordering

- The cdf value for n-dimensioned multivariate normal variable comprises of m multiplications of d-dimensional integrals.
- Recall the RCMVN algorithm(3) : as computing each d-dimensional integral values, integration variables were arranged in order of increasing order of CMVN probability values, from outer to inner
- Permutes the block of LDL-decomposed covariance matrix, in order of RCMVN probability values of each blocks

Block Reordering

Elements

Typography

```
The theme provides sensible defaults to
\emph{emphasize} text, \alert{accent} parts
or show \textbf{bold} results.
```

becomes

The theme provides sensible defaults to *emphasize* text, accent parts or show **bold** results.

Font feature test

- Regular
- *Italic*
- SmallCaps
- **Bold**
- ***Bold Italic***
- **Bold SmallCaps**
- `Monospace`
- *`Monospace Italic`*
- **`Monospace Bold`**
- ***`Monospace Bold Italic`***

Lists

Items

- Milk
- Eggs
- Potatos

Enumerations

1. First,
2. Second and
3. Last.

Descriptions

PowerPoint Meeh.

Beamer Yeeeha.

Animation

- This is important

Animation

- This is important
- Now this

Animation

- This is important
- Now this
- And now this

Animation

- This is really important
- Now this
- And now this

Figures

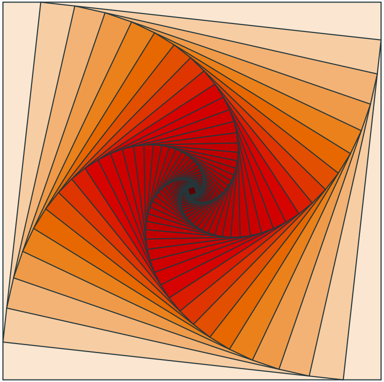


Figure 2: Rotated square from texample.net.

Tables

Table 2: Largest cities in the world (source: Wikipedia)

City	Population
Mexico City	20,116,842
Shanghai	19,210,000
Peking	15,796,450
Istanbul	14,160,467

Blocks

Three different block environments are pre-defined and may be styled with an optional background color.

Default
Block content.

Alert
Block content.

Example
Block content.

Default
Block content.

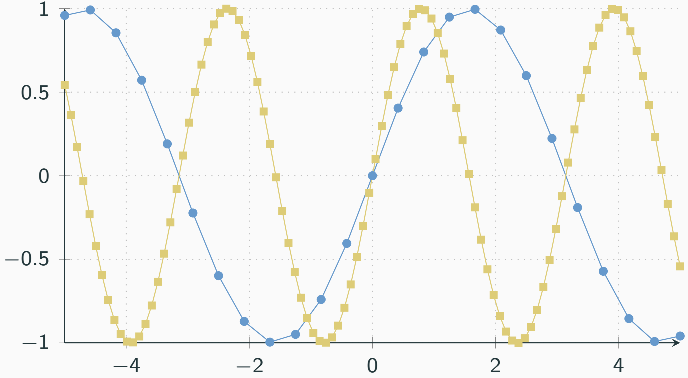
Alert
Block content.

Example
Block content.

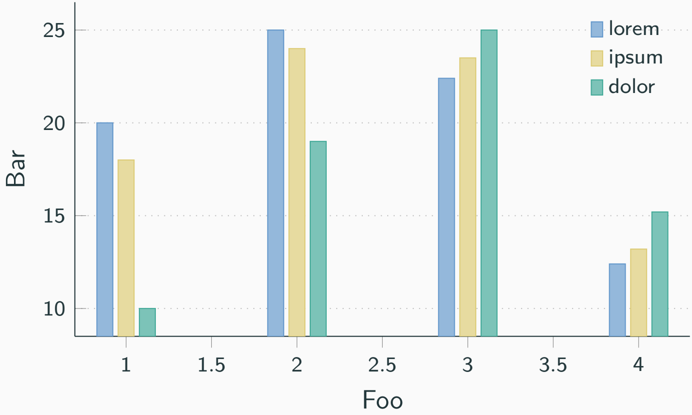
Math

$$e = \lim_{n \to \infty} \left(1 + \frac{1}{n}\right)^n$$

Line plots



Bar charts



Quotes

Veni, Vidi, Vici

Frame footer

metropolis defines a custom beamer template to add a text to the footer. It can be set via

```
\setbeamertemplate{frame footer}{My custom footer}
```

References

Some references to showcase [allowframebreaks] **?????**

Conclusion

Summary

Get the source of this theme and the demo presentation from

`github.com/matze/mtheme`

Questions?

Backup slides

Sometimes, it is useful to add slides at the end of your presentation to refer to during audience questions.

The best way to do this is to include the `appendixnumberbeamer` package in your preamble and call `\appendix` before your backup slides.

metropolis will automatically turn off slide numbering and progress bars for slides in the appendix.

References

Cao, J., Genton, M. G., Keyes, D. E., and Turkiyyah, G. M. (2019). Hierarchical-block conditioning approximations for high-dimensional multivariate normal probabilities. *Statistics and Computing*, 29(3):585–598.

Genton, M. G., Keyes, D. E., and Turkiyyah, G. (2018). Hierarchical decompositions for the computation of high-dimensional multivariate normal probabilities. *Journal of Computational and Graphical Statistics*, 27(2):268–277.

Genz, A. (1992). Numerical computation of multivariate normal probabilities. *Journal of computational and graphical statistics*, 1(2):141–149.

References ii

Genz, A. and Bretz, F. (2009). *Computation of multivariate normal and t probabilities*, volume 195. Springer Science & Business Media.

Gibson, G., Glasbey, C., and Elston, D. (1994). Monte carlo evaluation of multivariate normal integrals and sensitivity to variate ordering. *Advances in Numerical Methods and Applications*, pages 120–126.

Hackbusch, W. (2015). *Hierarchical matrices: algorithms and analysis*, volume 49. Springer.

Kamakura, W. A. (1989). The estimation of multinomial probit models: A new calibration algorithm. *Transportation Science*, 23(4):253–265.

Kan, R. and Robotti, C. (2017). On moments of folded and truncated multivariate normal distributions. *Journal of Computational and Graphical Statistics*, 26(4):930–934.

Mendell, N. R. and Elston, R. (1974). Multifactorial qualitative traits: genetic analysis and prediction of recurrence risks. *Biometrics*, pages 41–57.

Schervish, M. J. (1984). Algorithm as 195: Multivariate normal probabilities with error bound. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 33(1):81–94.

Trinh, G. and Genz, A. (2015). Bivariate conditioning approximations for multivariate normal probabilities. *Statistics and Computing*, 25(5):989–996.