High-Dimensional Gaussian Sampling

Yuejia Zhang

April 7, 2023

Problem Definition

Sampling from a d-dimensional Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, where d may be large.

$$\pi(\boldsymbol{\theta}) = \frac{1}{(2\pi)^{d/2}\det(\boldsymbol{\Sigma})^{1/2}} \exp\left(-\frac{1}{2}(\boldsymbol{\theta}-\boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\boldsymbol{\theta}-\boldsymbol{\mu})\right).$$

Covariance matrix $\boldsymbol{\Sigma}$ positive definite. Precision matrix $\mathbf{Q} = \boldsymbol{\Sigma}^{-1}$ exists and also positive definite.

Special Cases

- $d = 1$

Algorithm 1 Box–Muller sampler

1: Draw $u_1, u_2 \overset{\text{i.i.d.}}{\sim} \mathcal{U}((0,1])$.
2: Set $\tilde{u}_1 = \sqrt{-2\log(u_1)}$.
3: Set $\tilde{u}_2 = 2\pi u_2$.
4: **return** $(\theta_1, \theta_2) = \left(\mu + \frac{\tilde{u}_1}{\sqrt{q}}\sin(\tilde{u}_2), \mu + \frac{\tilde{u}_1}{\sqrt{q}}\cos(\tilde{u}_2)\right)$.

Special Cases

Algorithm 2 Sampler when $\mathbf{Q}$ is a diagonal matrix

1: **for** $i \in [d]$ **do** ▷ In some programming languages, this loop can be vectorized.
2: Draw $\theta_i \sim \mathcal{N}(\mu_i, 1/q_i)$.
3: **end for**
4: **return** $\boldsymbol{\theta} = (\theta_1, \cdots, \theta_d)^\top$.

General Cases

Algorithm 3 Cholesky sampler

1: Set $\mathbf{C} = \mathrm{chol}(\mathbf{Q})$. $\triangleright$ $\mathbf{Q} = \mathbf{C}\mathbf{C}^\top$
2: Draw $\mathbf{z} \sim \mathcal{N}(\mathbf{0}_d, \mathbf{I}_d)$.
3: Solve $\mathbf{C}^\top \mathbf{w} = \mathbf{z}$ w.r.t. $\mathbf{w}$.
4: **return** $\boldsymbol{\theta} = \boldsymbol{\mu} + \mathbf{w}$.

Problem:

- Computational cost $\mathcal{O}(d^3 + d^2 T)$ (T is the number of samples), only when $\mathbf{Q}$ is unchanged.
- Storage requirement $\Theta(d^2)$.

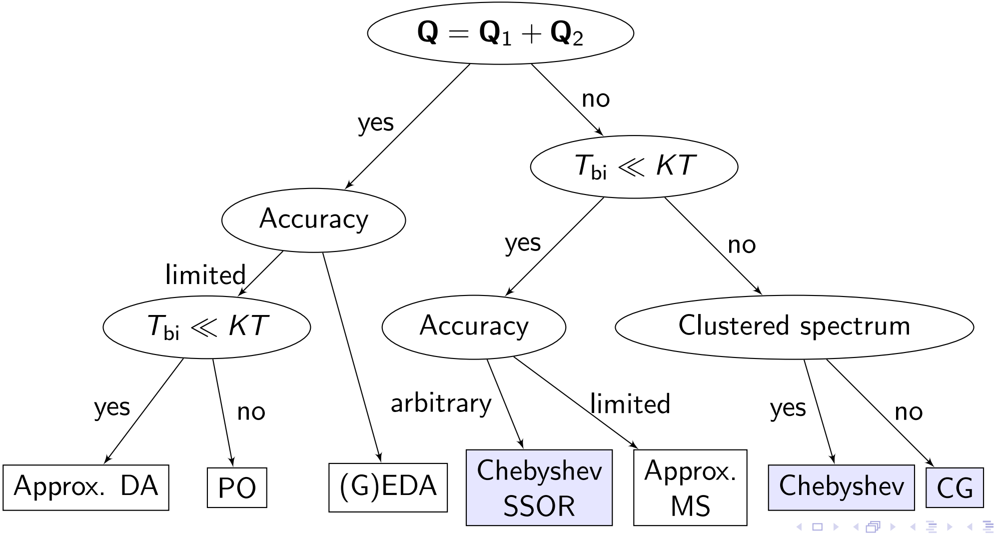
More Efficient Solutions

- Square Root approximation: Approximate $\mathbf{Q}^{1/2}$.
- Conjugate Gradient: Solve a linear system w.r.t. $\mathbf{Q}$.
- Matrix Splitting: A generalization of Gibbs Sampler.
- Data Augmentation: Introduce auxiliary variable.

Improvement:

- Computational cost $\mathcal{O}(Kd^2T)$ (K is the number of iterations), or $\mathcal{O}(d^2(T+T_{bi}))$ (T_{bi} is the number of burn-in samples).
- Storage requirement $\Theta(d)$.

How to Choose the Sampler



Bayesian Ridge Regression

Conditional prior for $\boldsymbol{\theta}$: Gaussian i.i.d.,

$$p(\boldsymbol{\theta} \mid \tau) \propto \exp\left(-\frac{1}{2\tau}||\boldsymbol{\theta}||^2\right),$$

$$p(\tau) \propto \frac{1}{\tau}\mathbf{1}_{\mathbb{R}_+\setminus\{0\}}(\tau).$$

Posterior:

$$p(\boldsymbol{\theta}, \tau \mid \mathbf{y}) \propto \frac{1}{\tau}\mathbf{1}_{\mathbb{R}_+\setminus\{0\}}(\tau)\ \exp\left(-\frac{1}{2\tau}||\boldsymbol{\theta}||^2 - \frac{1}{2\sigma^2}||\mathbf{y} - \mathbf{X}\boldsymbol{\theta}||^2\right).$$

Bayesian Ridge Regression (Cont.d)

Conditional posterior distribution associated to $\boldsymbol{\theta}$: Gaussian with precision matrix and mean vector

$$\mathbf{Q} = \frac{1}{\sigma^2}\mathbf{X}^\top\mathbf{X} + \tau^{-1}\mathbf{I}_d,$$

$$\boldsymbol{\mu} = \frac{1}{\sigma^2}\mathbf{Q}^{-1}\mathbf{X}^\top\mathbf{y}.$$

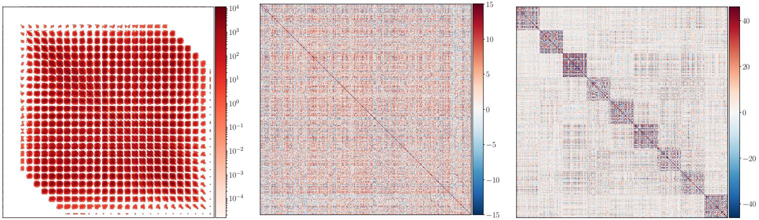


Figure: Examples of precision matrices $\mathbf{X}^\top\mathbf{X}$ for the MNIST, leukemia abd CoEPrA datasets.

Square Root Factorization

Extension of Cholesky sampler:

1. $\mathbf{Q} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^\top$.
2. $\mathbf{Q} = \mathbf{B}^2$ with $\mathbf{B} = \mathbf{U}\boldsymbol{\Lambda}^{1/2}\mathbf{U}^\top$.
3. $\mathbf{z} \sim \mathcal{N}(\mathbf{0}_d, \mathbf{I}_d)$.
4. Solve $\mathbf{B}\mathbf{w} = \mathbf{z}$ w.r.t. $\mathbf{w}$ and compute $\boldsymbol{\theta} = \boldsymbol{\mu} + \mathbf{w}$.

We have $f(\mathbf{Q}) = \mathbf{U}f(\boldsymbol{\Lambda})\mathbf{U}^\top$ for real continuous f.
Approximate $f(\lambda_i) \approx 1/\sqrt{\lambda_i}, \quad \forall i \in [d]$ with Chebyshev polynomials.

Chebyshev Sampler

The change of interval:

$$g_j = \left[\cos\left(\pi\frac{2j+1}{2K_{\text{cheby}}}\right)\frac{(\lambda_u-\lambda_l)}{2}+\frac{\lambda_u+\lambda_l}{2}\right]^{-1/2}, \quad j\in[0,K_{\text{cheby}}].$$

The Chebyshev coefficients:

$$c_k = \frac{2}{K_{\text{cheby}}}\sum_{j=0}^{K_{\text{cheby}}} g_j\cos\left(\pi k\frac{2j+1}{2K_{\text{cheby}}}\right), \quad k\in[0,K_{\text{cheby}}].$$

Chebyshev Sampler

Algorithm 4 Approx. square root sampler using Chebyshev polynomials

Draw $\mathbf{z} \sim \mathcal{N}(\mathbf{0}_d, \mathbf{I}_d)$.
Set $\alpha = \dfrac{2}{\lambda_u - \lambda_l}$ and $\beta = \dfrac{\lambda_u + \lambda_l}{\lambda_u - \lambda_l}$.
Initialize $\mathbf{u}_1 = \alpha \mathbf{Q}\mathbf{z} - \beta \mathbf{z}$ and $\mathbf{u}_0 = \mathbf{z}$.
Set $\mathbf{u} = \frac{1}{2} c_0 \mathbf{u}_0 + c_1 \mathbf{u}_1$ and $k = 2$.
while $k \leq K_{\text{cheby}}$ **do** ▷ Compute the K_{cheby}-truncated Chebyshev series.
    Compute $\mathbf{u}' = 2(\alpha \mathbf{Q}\mathbf{u}_1 - \beta \mathbf{u}_1) - \mathbf{u}_0$.
    Set $\mathbf{u} = \mathbf{u} + c_k \mathbf{u}'$.
    Set $\mathbf{u}_0 = \mathbf{u}_1$ and $\mathbf{u}_1 = \mathbf{u}'$.
    $k = k + 1$.
end while

Perturbation before Optimization

Rewrite in *information form*:

$$\pi(\boldsymbol{\theta}) \propto \exp\left(-\frac{1}{2}\boldsymbol{\theta}^\top \mathbf{Q}\boldsymbol{\theta} + \mathbf{b}^\top \boldsymbol{\theta}\right),$$

where $\mathbf{b} = \mathbf{Q}\boldsymbol{\mu}$.

1. Draw a Gaussian vector $\mathbf{z}' \sim \mathcal{N}(\mathbf{0}_d, \mathbf{Q})$.
2. Solve a linear system $\mathbf{Q}\boldsymbol{\theta} = \mathbf{Q}\boldsymbol{\mu} + \mathbf{z}'$ using conjugate gradient methods. (If $\mathbf{u} \sim \mathcal{N}(\mathbf{Q}\boldsymbol{\mu}, \mathbf{Q})$, then $\mathbf{Q}^{-1}\mathbf{u} \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{Q}^{-1})$.)

Optimization with Perturbation

The linear system we solved

$$\mathbf{Q}\boldsymbol{\theta} = \mathbf{b} + \mathbf{z}'$$

can also be seen as a perturbed version of the linear system

$$\mathbf{Q}\boldsymbol{\theta} = \mathbf{b},$$

where $\mathbf{b} = \mathbf{Q}\boldsymbol{\mu}$.

Add a perturbation step (a univariate Gaussian sampling step) to turn the classical CG solver into a CG sampler.

Sequentially builds a Gaussian vector with a covariance matrix being the best k-rank approximation of $\mathbf{Q}^{-1}$ in the Krylov subspace $\mathcal{K}_k(\mathbf{Q}, \mathbf{r}^{(0)})$.

CG Sampler

Set $k=1$, $\mathbf{r}^{(0)} = \mathbf{c} - \mathbf{Q}\boldsymbol{\omega}^{(0)}$, $\mathbf{h}^{(0)} = \mathbf{r}^{(0)}$, $d^{(0)} = \mathbf{h}^{(0)\top}\mathbf{Q}\mathbf{h}^{(0)}$ and $\mathbf{y}^{(0)} = \boldsymbol{\omega}^{(0)}$.
while $\left\|\mathbf{r}^{(k)}\right\| \geq \epsilon$ **do**
  Set $\gamma^{(k-1)} = \dfrac{\mathbf{r}^{(k-1)\top}\mathbf{r}^{(k-1)}}{d^{(k-1)}}$.
  Set $\mathbf{r}^{(k)} = \mathbf{r}^{(k-1)} - \gamma^{(k-1)}\mathbf{Q}\mathbf{h}^{(k-1)}$.
  Set $\eta^{(k)} = -\dfrac{\mathbf{r}^{(k)\top}\mathbf{r}^{(k)}}{\mathbf{r}^{(k-1)T}\mathbf{r}^{(k-1)}}$.
  Set $\mathbf{h}^{(k)} = \mathbf{r}^{(k)} - \eta^{(k)}\mathbf{h}^{(k-1)}$.
  Set $d^{(k)} = \mathbf{h}^{(k)\top}\mathbf{Q}\mathbf{h}^{(k)}$.
  Set $\mathbf{y}^{(k)} = \mathbf{y}^{(k-1)} + \dfrac{z}{\sqrt{d^{(k-1)}}}\mathbf{h}^{(k-1)}$ where $z \sim \mathcal{N}(0,1)$. ▷ Perturbation
  $k = k+1$.
end while
Set $\boldsymbol{\theta} = \boldsymbol{\mu} + \mathbf{y}^{(K_{\mathrm{CG}})}$ where K_{CG} is the number of CG iterations.
return $\boldsymbol{\theta}$.

Conditional Gaussian Distribution

If $\boldsymbol{\theta} \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{Q}^{-1})$, then

$$\mathrm{E}(\boldsymbol{\theta}_i|\boldsymbol{\theta}_{-i}) = \boldsymbol{\mu} - \frac{1}{\mathbf{Q}_{ii}} \sum_{j \neq i} \mathbf{Q}_{ij}(\boldsymbol{\theta}_j - \boldsymbol{\mu}_j),$$

$$\mathrm{Prec}(\boldsymbol{\theta}_i|\boldsymbol{\theta}_{-i}) = \mathbf{Q}_{ii},$$

$$\mathrm{Corr}(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j|\boldsymbol{\theta}_{-ij}) = -\frac{\mathbf{Q}_{ij}}{\sqrt{\mathbf{Q}_{ii}\mathbf{Q}_{jj}}}.$$

Compare the above results with

$$\mathrm{Var}(\boldsymbol{\theta}_i) = \boldsymbol{\Sigma}_{ii},$$

$$\mathrm{Corr}(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j) = \frac{\boldsymbol{\Sigma}_{ij}}{\sqrt{\boldsymbol{\Sigma}_{ii}\boldsymbol{\Sigma}_{jj}}}.$$

Gibbs Sampler

Algorithm 5 Component-wise Gibbs sampler

Input: Number T of iterations and initialization $\boldsymbol{\theta}^{(0)}$.

Set $t = 1$.
while $t \leq T$ **do**
  for $i \in [d]$ **do**
    Draw $z \sim \mathcal{N}(0, 1)$.
    Set $$\theta_i^{(t)} = \frac{[\mathbf{Q}\boldsymbol{\mu}]_i}{Q_{ii}} + \frac{z}{\sqrt{Q_{ii}}} - \frac{1}{Q_{ii}} \left(\sum_{j>i} Q_{ij}\theta_j^{(t-1)} + \sum_{j<i} Q_{ij}\theta_j^{(t)} \right).$$
  end for
  Set $t = t + 1$.
end while
return $\boldsymbol{\theta}^{(T)}$.

Rewrite into Gauss–Seidel Linear Systems

Each iteration solves the linear system

$$(\mathbf{L}+\mathbf{D})\theta^{(t)} = \mathbf{Q}\mu + \mathbf{D}^{1/2}\mathbf{z} - \mathbf{L}^\top\theta^{(t-1)},$$

where $\mathbf{z} \sim \mathcal{N}(\mathbf{0}_d, \mathbf{I}_d)$.
By setting $\mathbf{M} = \mathbf{L} + \mathbf{D}$ and $\mathbf{N} = -\mathbf{L}^\top$ so that $\mathbf{Q} = \mathbf{M} - \mathbf{N}$,

$$\mathbf{M}\theta^{(t)} = \mathbf{Q}\mu + \tilde{\mathbf{z}} + \mathbf{N}\theta^{(t-1)},$$

where $\mathbf{N} = -\mathbf{L}^\top$ is strictly upper triangular and $\tilde{\mathbf{z}} \sim \mathcal{N}(\mathbf{0}_d, \mathbf{D})$ is easy to sample.

Matrix Splitting Sampler

Algorithm 6 MCMC sampler based on exact matrix splitting

Input: Number T of iterations, initialization $\theta^{(0)}$ and splitting $\mathbf{Q} = \mathbf{M} - \mathbf{N}$.

1: Set $t = 1$.
2: **while** $t \leq T$ **do**
3: Draw $\tilde{\mathbf{z}} \sim \mathcal{N}(\mathbf{0}_d, \mathbf{M}^\top + \mathbf{N})$.
4: Solve $\mathbf{M}\theta^{(t)} = \mathbf{Q}\mu + \tilde{\mathbf{z}} + \mathbf{N}\theta^{(t-1)}$ w.r.t. $\theta^{(t)}$.
5: Set $t = t + 1$.
6: **end while**
7: **return** $\theta^{(T)}$.

Other Matrix Splitting Schemes

Table: The matrices $\mathbf{D}$ and $\mathbf{L}$ denote the diagonal and strictly lower triangular parts of $\mathbf{Q}$, respectively, and ω is a positive scalar.

Sampler	$\mathbf{M}$	$\mathbf{N}$	$\operatorname{cov}(\tilde{\mathbf{z}}) = \mathbf{M}^\top + \mathbf{N}$	convergence
Richardson	$\mathbf{I}_d/\omega$	$\mathbf{I}_d/\omega - \mathbf{Q}$	$2\mathbf{I}_d/\omega - \mathbf{Q}$	$0 < \omega < 2/\left\|\mathbf{Q}\right\|$
Jacobi	$\mathbf{D}$	$\mathbf{D} - \mathbf{Q}$	$2\mathbf{D} - \mathbf{Q}$	$\left\|Q_{ii}\right\| > \sum_{j\neq i}\left\|Q_{ij}\right\| \ \forall i \in [d]$
Gauss–Seidel	$\mathbf{D} + \mathbf{L}$	$-\mathbf{L}^\top$	$\mathbf{D}$	always
SOR	$\mathbf{D}/\omega + \mathbf{L}$	$\frac{1-\omega}{\omega}\mathbf{D} - \mathbf{L}^\top$	$\frac{2-\omega}{\omega}\mathbf{D}$	$0 < \omega < 2$

Error of t-th Order Polynomial

Given a linear system $\mathbf{Q}\boldsymbol{\theta} = \mathbf{v}$ and linear solvers based on the matrix splitting $\mathbf{Q} = \mathbf{M} - \mathbf{N}$, consider the recursion,

$$\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} + \mathbf{M}^{-1}(\mathbf{v} - \mathbf{Q}\boldsymbol{\theta}^{(t)}).$$

The error at iteration t,

$$\mathbf{e}^{(t+1)} = \boldsymbol{\theta}^{(t+1)} - \mathbf{Q}^{-1}\mathbf{v},$$

is equal to

$$(\mathbf{I}_d - \mathbf{M}^{-1}\mathbf{Q})^t \mathbf{e}^{(0)}.$$

Can we find another t-th order polynomial P_t that achieves a lower error?

$$\rho(\mathrm{P}_t(\mathbf{M}^{-1}\mathbf{Q})) < \rho((\mathbf{I}_d - \mathbf{M}^{-1}\mathbf{Q})^t).$$

Polynomial Accelerated Solver

Consider the second-order iterative scheme, for any $t \in \mathbb{N}$,

$$\boldsymbol{\theta}^{(t+1)} = \alpha_t \boldsymbol{\theta}^{(t)} + (1 - \alpha_t)\boldsymbol{\theta}^{(t-1)} + \beta_t \mathbf{M}^{-1}(\mathbf{v} - \mathbf{Q}\boldsymbol{\theta}^{(t)}),$$

where $(\alpha_t, \beta_t)_{t\in\mathbb{N}}$ are a set of acceleration parameters.
This iterative method yields an error at step t given by

$$\mathbf{e}^{(t+1)} = \mathrm{P}_t(\mathbf{M}^{-1}\mathbf{Q})\mathbf{e}^{(0)},$$

where P_t stands for a scaled Chebyshev polynomial.
Optimal values for $(\alpha_t, \beta_t)_{t\in\mathbb{N}}$ are given by

$$\alpha_t = \tau_1 \beta_t \quad \text{and} \quad \beta_t = \left(\tau_1 - \tau_2^2 \beta_{t-1}\right)^{-1},$$

$\tau_1 = [\lambda_{\min}(\mathbf{M}^{-1}\mathbf{Q}) + \lambda_{\max}(\mathbf{M}^{-1}\mathbf{Q})]/2$ and $\tau_2 = [\lambda_{\max}(\mathbf{M}^{-1}\mathbf{Q}) - \lambda_{\min}(\mathbf{M}^{-1}\mathbf{Q})]/4$.

Symmetric Splitting Scheme

Denote by $\mathbf{M}_{\text{SOR}}$ and $\mathbf{N}_{\text{SOR}}$ the matrices involved in the SOR splitting such that $\mathbf{Q} = \mathbf{M}_{\text{SOR}} - \mathbf{N}_{\text{SOR}}$.
Then for any $0 < \omega < 2$, the SSOR (symmetric SOR) splitting is defined by $\mathbf{Q} = \mathbf{M}_{\text{SSOR}} - \mathbf{N}_{\text{SSOR}}$ with

$$\mathbf{M}_{\text{SSOR}} = \frac{\omega}{2-\omega}\mathbf{M}_{\text{SOR}}\mathbf{D}^{-1}\mathbf{M}_{\text{SOR}}^{\top} \quad \text{and} \quad \mathbf{N}_{\text{SSOR}} = \frac{\omega}{2-\omega}\mathbf{N}_{\text{SOR}}\mathbf{D}^{-1}\mathbf{N}_{\text{SOR}}^{\top} .$$

Approximate Matrix Splitting

Algorithm 7 MCMC sampler based on approximate matrix splitting

Input: Number T of iterations, initialization $\theta^{(0)}$ and splitting $\mathbf{Q} = \mathbf{M} - \mathbf{N}$.

1: Set $t = 1$.
2: **while** $t \leq T$ **do**
3: Draw $\tilde{\mathbf{z}}' \sim \mathcal{N}(\mathbf{0}_d, \tilde{\mathbf{M}})$. ▷ $\tilde{\mathbf{M}} = \mathbf{D}$ or $2\left(\mathbf{D} + 2\omega \mathbf{I}_d\right)$.
4: Solve $\mathbf{M}\theta^{(t)} = \mathbf{Q}\mu + \tilde{\mathbf{z}}' + \mathbf{N}\theta^{(t-1)}$.
5: Set $t = t + 1$.
6: **end while**
7: **return** $\theta^{(T)}$.

1

PPA

1