Warwick Summer School on Complexity and Inference - Lecture II: Inference

Dalia Chakrabarty[1]

[1]University of Warwick,
Department of Statistics

May 15, 2012

Outline

- Optimisation - shortcomings of gradient methods, constrained optimisation, search.
- Monte Carlo - motivation, numerical integration, simulation, simulated annealing.
- MCMC - motivation, Metropolis-Hastings, Independent sampler, Langevin, Gibbs sampling.

Optimisation - introduction

In our pursuit of understanding the behaviour of a system - even when such understanding is not "complete" - we often find it useful to achieve a model of the system parameters such that some behaviour is "optimal", according to a pre-set criterion.

- What is the optimal distance between an approaching car and I, for me to cross safely?
- Perhaps that is how the brain works - sampling (locally) for information and then optimising.

Statement of the problem

Minimise (or maximise) a function $f : \mathcal{X} \longrightarrow \mathcal{H}$, where $\mathcal{H} \subset \mathbb{R}$ (for real objective functions) and $\mathcal{X} \subset \mathbb{R}^N$ (for finite dimensional systems)., i.e. the attempt is to minimise or maximise the objective function $f(\mathbf{x})$, where $\mathbf{x} = (x_1, x_2, \ldots, x_N)^T$ is the model parameter vector (Adby & Dempster).
If the optimal inputs are sought,

$$\mathbf{x}_{min} = \arg\min_{\mathbf{x}} f(\mathbf{x}) \tag{1}$$

If optimal inputs are sought when there are constraint(s) on some of the N parameters,

$$\mathbf{x}_{min} = \arg\min_{\mathbf{x}} f(\mathbf{x}) \quad \text{subject to} \quad g_k(\mathbf{x}) = 0, h_k(\mathbf{x}) \leq 0. \tag{2}$$

Hessian

$$f(\mathbf{x}+\delta\mathbf{x})-f(\mathbf{x})=\mathbf{J}(\mathbf{x})\delta\mathbf{x}+\frac{1}{2}\delta\mathbf{x}^T\mathbf{H}\delta\mathbf{x}, \tag{3}$$

where $f:\mathbb{R}^N \Longrightarrow \mathbb{R}$, $\mathbf{J}$ is the Jacobian and the Jacobian of the gradient of f is the $\mathbf{H}^{(N\times N)}$ Hessian matrix, i.e. the matrix with $\frac{\partial^2 f}{\partial x_i^2}$ along the diagonal and $\frac{\partial^2 f}{\partial x_i \partial x_j}$, $i \neq j$ off-diagonal.

Second order partial derivative test

How to distinguish between convergence to local minima and global minima?

If root of $f'(\mathbf{x})$ is identified at $\mathbf{x}_0$

$\mathbf{H}$ positive definite at $\mathbf{x}_0 \Longrightarrow f$ is minimum at $\mathbf{x}_0$.

$\mathbf{H}$ negative definite at $\mathbf{x}_0 \Longrightarrow f$ is maximum at $\mathbf{x}_0$.

$\mathbf{H}$ has positive and negative eigenvalues $\Longrightarrow \mathbf{x}_0$ is a saddle point.

If none of the above, second order partial derivative test is inconclusive.

Unconstrained optimisation

- Gradient descent - highest space rate of change in $f(\mathbf{x})$ is along $\nabla f(\mathbf{x})$. So minimisation suggests $\mathbf{x}_{i+1} = \mathbf{x}_i - \gamma_i \nabla f(\mathbf{x}_i)$, then $f(\mathbf{x}_{i+1}) < f(\mathbf{x}_i)$, i.e. $\mathbf{x}_0, \mathbf{x}_1, \ldots$ will converge to minima of f.
- Newton method - find roots of ∇f to identify stationary points of f. 2nd order Taylor expansion of f around x_i gives $f(x_i + \delta x) = f(x_i) + f^{'}(x_i)\delta x + (1/2) f^{''}(x_i)(\delta x)^2$. If a maxima or minima occurs at x_i, $f^{'} + f^{''}(x_i)\delta x = 0$. Sequence $\{x_i\}$ generated by $\delta x = x_{i+1} - x_i = -\dfrac{f^{'}(x_i)}{f^{''}(x_i)}$ will coverge to root of $f^{'}(x)$. In high-dim $\mathbf{x}_{i+1} = \mathbf{x}_i - \gamma(\mathbf{H} f(\mathbf{x}_i))^{-1} \nabla f(\mathbf{x}_i)$.

Possible shortcomings

- May not be differentiable all over the search space- minima on the path of discontinuity. Need to identify discontinuites!
- Objective function may not be convex. If not strictly convex, may not have just one minima in the broader search space - then need to identify region over which $f(\mathbf{x})$ has only one minima.
- Narrow valleys imply large change in $f(\mathbf{x})$ with small change in $\mathbf{x} \rightsquigarrow$ slow convergence. Need to identify narrow valleys!
- In high dimensions - hard.

Lagrange multipliers

The equality constraints are incorporated into the objective function as

$$f(\mathbf{x}) + \sum_{k=1}^{M} \lambda_k g_k(\mathbf{x}) \tag{4}$$

where λ_k, $k = 1, \ldots, M$ are the M lagrange multipliers. The inequality constraints are incorporated using the slack variables z_k such that $h_k(\mathbf{x}) \leq 0$ transforms to $h_k(\mathbf{x}) + z_k^2 = 0$. Then the Lagrangian is $f(\mathbf{x}) + \sum_{k=1}^{M} \lambda_k g_k(\mathbf{x}) + \sum_{k=1}^{L} \mu_k [h_k(\mathbf{x}) + z_k^2]$

Gradient of the objective function is orthogonal to the surface of the active constraints.

Kuhn-Tucker conditions - 1^{st} order

Then for $\tilde{\mathbf{x}}$ to be a minimum,

$$
\begin{aligned}
\nabla f(\tilde{\mathbf{x}}) + \sum_{k=1}^{M} \lambda_k \nabla g_k(\tilde{\mathbf{x}}) + \sum_{k=1}^{L} \mu_k \nabla h_k(\tilde{\mathbf{x}}) &= 0 \qquad (5) \\
\text{or at the minima} \quad \nabla(\text{Lagrangian}) &= 0 \\
\text{and} \quad \mu_i h_i(\tilde{\mathbf{x}}) &= 0 \\
\tilde{\mu}_i &\geq 0
\end{aligned}
$$

The first equation expresses the condition of stationarity (Kuhn & Tucker 1951).

Kuhn-Tucker conditions - 2^{nd} order

If the relevant functions are twice differentiable, then at a minima,

$$\mathbf{x}^T \left[\nabla^2 f(\tilde{\mathbf{x}}) + \sum_{k=1}^{M} \tilde{\lambda}_k \cdot \nabla^2 g_k(\tilde{\mathbf{x}}) + \sum_{k=1}^{L} \tilde{\mu}_k \cdot \nabla^2 h_k(\tilde{\mathbf{x}}) \right] \mathbf{x} > 0 \quad (6)$$

$$\begin{aligned} \text{at the minima} \quad \nabla g_k(\tilde{\mathbf{x}}) \cdot \mathbf{x} = 0 \quad \textit{when} \quad \mu_k &> 0 \\ \nabla g_k(\tilde{\mathbf{x}}) \cdot \mathbf{x} \geq 0 \quad \textit{when} \quad \mu_k &= 0 \\ \nabla h_k(\tilde{\mathbf{x}}) \cdot \mathbf{x} &= 0 \end{aligned}$$

Convergence criterion of constrained optimisation techniques must converge to a point $\tilde{\mathbf{x}}$ satisfying the 1st or 2nd order KTT conditions. All functions convex $\Longrightarrow$ 1st order conditions guarantee global minima.

Constrained optimisation methods

- Methods that account for the constraints explicitly - barrier methods:
  1. Direct search methods - when close to constraint, modify direction of search. Thus, repulsion away from constraints. These methods may not rely on differentiability of the functions and are not affected by lack of robustness in numrical differentiation.
  2. Gradient methods - when violation of constraint is impending, change direction given by the negative gradient, into the feasible region.
- Methods that account for constraints implicity - penalise constraint violation. As a result, these are more widely applicable. These include sequential penalty transforms, exact penalty transforms.

Francesca, van Beek & Walsh, 2006.

Penalty function based methods - sequential penalty transforms

The basic idea is to rephrase the constrained optimisation problem as a sequence of unconstrained optimisation problems, where the sequence is configured to reduce the set of unconstrained optimisations, as equivalent to the original problem (Fiacco & McCorick 1968, Lootsma 1972).

Thus, $\min f(\mathbf{x})$ subject to $g_k(\mathbf{x}) \leq 0$, $k = 1, \ldots, M$

is equivalent to $\min \xi_i(\mathbf{x}) = \min[f(\mathbf{x}) + s_k \sum_{k=1}^{M} h(z_k(\mathbf{x}))]$,

where $z_k(\mathbf{x})$ is the distance of the solution $\mathbf{x}$ from the feasibility region - and is therefore dependent on the constraint g_k - and $h(\cdot)$ is a monotonically non-decreasing penalty function such that $h(0) = 0$.

So, the idea is to solve the 2nd minimisation problem and use the result as input for the next iteration, with a bigger penalty parameter, $s_2 > s_1 \ldots \Longrightarrow$ the 1st minimisation is solved.

Barrier methods

Aim: minimize $f(\mathbf{x})$ subject to $g_i(\mathbf{x}) \geq 0$.
Define the barrier function

$$B(\mathbf{x}, \mu) = f(\mathbf{x}) - \mu \sum_{i=1}^{M} \log(g_i(\mathbf{x})) \tag{7}$$

So $\mu \longrightarrow 0 \Longrightarrow \min(B(\mathbf{x}, \mu)) \longrightarrow$ sought solution. So we minimise $B(\cdot, \cdot)$. Thus,

$$B^{'}(\mathbf{x}, \mu) = \nabla f(\mathbf{x}) - \mu \sum_{i=1}^{M} \frac{\nabla(g_i(\mathbf{x}))}{g_i(\mathbf{x})}. \tag{8}$$

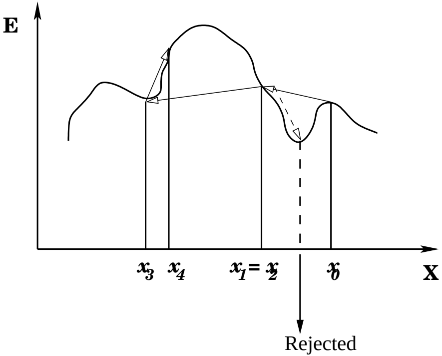
Brute force search methods

A blind search that can - in principle - proceed without any knowledge of the application. Of limited use in real-life complex problems.

- Initiate the algorithm with a candidate solution $\mathbf{x}_0$ for the optimisation problem.
- Generate a next candidate solution $\tilde{\mathbf{x}}$.
- Check if $\tilde{\mathbf{x}}$ is a solution for the given optimisation problem.
- If so, accept $\tilde{\mathbf{x}}$ as a solution. If not, generate a new candidate solution and proceed as before.

- How to generate new candidate solution in high-dimensional space?
- When to stop?

Simulated annealing - approximation to the global minima



Simulated annealing - approximation to the global minima

Kirkpatrick, Gelatt & Vecchi (1983).

- Initiate with a seed solution $\mathbf{x}_0$.
- Propose next solution $\tilde{\mathbf{x}}_1$.
- Check if $\tilde{\mathbf{x}}_1$ is accepted at pre-set probability.
- If so, accept $\tilde{\mathbf{x}}_1 = \mathbf{x}_1$ as a solution. If not, reject $\tilde{\mathbf{x}}_1$ and generate a new candidate solution.
- Save $\mathbf{x}_i$ as "best" solution if $\mathcal{L}(\mathbf{x}_i) > \mathcal{L}(\mathbf{x}_j), j = 0, \ldots, x - i$.
- Probability of transition from state $\mathbf{x}_i$ to candidate state $\tilde{\mathbf{x}}_{i+1}$ depends on the current temperature parameter T.
- Probability of accepting proposed candidate state $P(\mathcal{L}_i, \mathcal{L}_{i+1}, T)$. This is in some algorithms placed as 1 if $\mathcal{L}_{i+1} > \mathcal{L}_i$ but $\exp((\mathcal{L}_{i+1} - \mathcal{L}_i)/T)$.
- Cooling schedule: $T_i = f(T_0, i)$.

Optimisation using sampling

Optimisation problems in which decision is made or learning of system parameter happens in consequence of a process that results in noisy inputs.

When there are multiple secondary maxima, in addition to a global maxima - which represents the most likely solution - the most likely solution is not the best solution. In a high-dimensional situation, a direct global search becomes difficult. Randomly generate sample of models, distributed as the objetive fuction - approximates search space.

Monte Carlo

History

Monte Carlo methods

Solving a deterministic problem by using random numbers. Is most pertinent for a system for which the observables are uncertain and can work even for systems with large number of degrees of freedom.

- Generate samples from a chosen probability distribution.
- Pass the generated sample through a criterion or perform some computation with it.

Monte Carlo methods incorporate uncertainties in observables (inputs) to learn system behaviour. Prior to these methods, simulations of a (simple) known system behaviour were used to generate estimates of uncertainty in relevant model parameters.

Monte Carlo methods - example

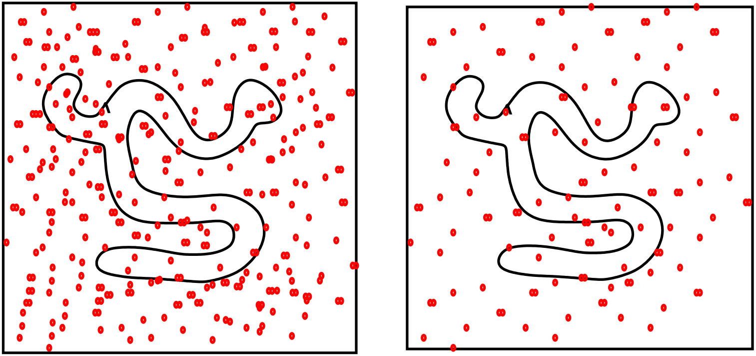


Figure: $\dfrac{\text{Area of figure}}{\text{Area of square}} = \dfrac{\text{Number of particles inside figure}}{\text{Total number of particles}}$

Numerical integration

Average over generated samples approximates the truth. Approximation improves with number of generated samples.

The maximum a posteriori solution for the model parameter vector $\boldsymbol{\theta}$, given data D is:

$$\boldsymbol{\theta}^\star = \arg\max_{\boldsymbol{\theta}} \pi(\boldsymbol{\theta}|D) \tag{9}$$

If problem involves learning of an “optimal” value of a function $f(\boldsymbol{\theta}, x)$, of the learnt $\boldsymbol{\theta}$ - could be the utility - then we seek

$$\begin{aligned} \hat{x} &= \arg\max_{\theta} \sum_{\boldsymbol{\theta}} \pi(\boldsymbol{\theta}) f(\boldsymbol{\theta}, x) \qquad (10) \\ &\approx \sum_i \frac{f(\boldsymbol{\theta}_i, x)}{N} \end{aligned}$$

where the sample $\{\boldsymbol{\theta}_1 \ldots, \boldsymbol{\theta}_n\}$ is drawn from $\pi(\boldsymbol{\theta}|D)$.

Simulation - using the Metropolis example

Generate sample - the sequence $\{\theta_1, \ldots, \theta_N\}$ such that $\lim_{N \longrightarrow \infty} N(\theta_i)/N = \Pr(\theta_i)$**.**

1. Start with $\theta = \theta_0$
2. At the i-th step, generate $\tilde{\theta}_{t+1}$ from a proposal density $Q(\tilde{\theta}_{t+1}|\theta_t)$.
3. Check if $\dfrac{\Pr(\tilde{\theta}_{t+1})}{\Pr(\theta_t)} > 1$. Then $\theta_{t+1} = \tilde{\theta}_{t+1}$. Else, accept $\theta_{t+1} = \tilde{\theta}_{t+1}$ if $\dfrac{\Pr(\tilde{\theta}_{t+1})}{\Pr(\theta_t)} > r$, $r \sim \mathcal{U}[0,1]$. This defines the acceptance probability $\Pr(\tilde{\theta}_{t+1}, \theta_t)$ in the Metropolis algorithm.
4. Repeat last step N times.

Evolution to the equilibrium distribution

No. of samples generated in state $\tilde{\theta}_{t+1}$ from another state θ_t = $N(\theta_t)\Pr(\theta_t, \tilde{\theta}_{t+1}) - N(\tilde{\theta}_{t+1})\Pr(\tilde{\theta}_{t+1}, \theta_t)$

$$\begin{aligned} &= N(\theta_t) - N(\tilde{\theta}_{t+1})\frac{\Pr(\theta_t)}{\Pr(\tilde{\theta}_{t+1})} \quad \text{if} \quad \Pr(\tilde{\theta}_{t+1}) > \Pr(\theta_t) \quad (11) \\ &= N(\theta_t)\frac{\Pr(\tilde{\theta}_{t+1})}{\Pr(\theta_t)} - N(\tilde{\theta}_{t+1}) \quad \text{if} \quad \Pr(\theta_t) > \Pr(\tilde{\theta}_{t+1}) \end{aligned}$$

So $N(\theta_i)/N = \Pr(\theta_i) \implies$ No. of samples generated in state $\tilde{\theta}_{t+1}$ from another state θ_t = 0.

MCMC - paradigm shift

We saw $i.i.d$ variables $\sim$ relevant density function π.
Now - correlated samples from a progressively evolving distributions that eventually approach the target distribution.
The correlated samples - present approximate structure of distribution they are sampled from (Robert & Casella, 2010).

- Accommodates cases when very little known about π.
- High-dim implementaion can be easily broken down to smaller. easier problems.

Markov chains

A finite Markov chain $\{\theta_i\}$ is defined by

$$\theta_{i+1}|\theta_0, \theta_1, \ldots, \theta_i \sim K(\theta_i, \theta_{i+1}) \quad (12)$$

where $K(\theta_i, \theta_{i+1})$ is called the Markov kernel which can be thought of as a generalisation of the transition matrix relevant to Markov processes in a finite state space. For example, if we consider the random walk, the Markov chain is defined by $\theta_{i+1} = \theta_i + \epsilon_i$ so that $\theta_{i+1} \sim \mathcal{N}(\theta_i, \sigma)$.

Markov chains

- Stationarity: $\theta_i \sim f \implies \theta_{i+j} \sim f, j > 0$.
- Stationarity $\implies$ Irreducibility, i.e. starting from state θ_i, $\forall\, i$, it is possible to get to any state θ_j.
- Irreducibility $\implies$ all states are, or no state is, periodic.

Stationarity implies that the stationary distribution f is the limiting distribution - ergodicity: $\lim_{t \longrightarrow 0} P^t(\beta_i, \beta_j) = \pi(\beta_j)$. So, the ergodic Markov chain that is sampled from f will converge to simulations of f. Then, expectation of a function $h(\beta)$ is given by arithetic average of $h_i := h(\beta_i)$.

- Proper posteriors for convergence.

Metropolis-Hastings

Choose a Markov kernel K so that the Markov chain generated by it converges to the target density f (Metropolis et. al (1953), Hasting (1973)).

M-H algorithm allows for construction of K so that the stationary distribution f is achieved, by choosing the conditional proposal density $q(\beta'|\beta)$, where

- $\dfrac{f(\beta')}{q(\beta'|\beta)}$ is known up to a constant independent of β.
- $q(\beta'|\beta)$ is flexible enough to explore the full support of f, for any β'.

Metropolis-Hastings

- **For a given β_i, simulate B'_{i+1} from $q(\beta'_{i+1}|\beta_i)$.**
- **Then**

$$\begin{aligned} B_{i+1} &= B'_{i+1} \quad \textbf{with probability} \quad \alpha(\beta_i, \beta'_{i+1}), \\ B_{i+1} &= B_i \quad \textbf{with probability} \quad 1 - \alpha(\beta_i, \beta'_{i+1}), \end{aligned} \tag{13}$$

where the acceptance probability

$$\alpha(\beta_i, \beta'_{i+1}) = \min\left(\frac{f(\beta'_{i+1})}{f(\beta_i)} \frac{q(\beta_i|\beta'_{i+1})}{q(\beta'_{i+1}|\beta_i)}, 1\right).$$

Metropolis-Hastings vs. Simulated Annealing

- From point of view of implementation - maximisation of the objective function, as opposed to exploring the support of f.
- From point of view of convergence - convergence to maxima of objective function as opposed to, convergence to f.
- From point of view of structure of samples - $.i.d.$ samples, as opposed to correlated samples.

Convergence to f is dictated by the choice of q. The parametrisation of efficiency of the algorithm is via the acceptance rate:

$$\bar{\alpha} = \lim_{I \longrightarrow 0} \sum_{i=1}^{I} \alpha(B_i, B'_{i+1}) = \int \alpha(\beta_i, \beta'_{i+1}) f(\beta_i) q(\beta'_{i+1}|\beta_i) d\beta_i \, d\beta'_{i+1}$$

Salient features

- In the symmetric case, when $q(\beta'_{i+1}|\beta_i) = q(\beta_i|\beta'_{i+1})$, $\alpha(\beta_i, \beta'_{i+1})$ depends on $f(\beta'_{i+1})/f(\beta_i)$.
- If domain of q is small compare to range of f, the chain has difficulty converging. Not grammatically wrong to propose β'_{i+1} from outside the range of f, i.e. $f(\beta'_{i+1}$=0, but then the proposed state is going to be rejected $\longrightarrow$ chain stuck over most steps.
- Even when $f(\beta'_{i+1})/q(\beta_i|\beta'_{i+1})$ is less than $f(\beta_i)/q(\beta'_{i+1}|\beta_i)$, the proposed state may be accepted, depending on how these numbers compare to each other. But if the ratio of thes numbers suggests too many rejections, performance of M-H will be depreciated.
- The algorithm employs ratios and thereby does away with the need for determining the normalisation constants.

Independent sampler

q is independent of the present state, i.e. $q(\beta'_{i+1}|\beta_i) = q(\beta'_{i+1})$. Then acceptance probability depends on $\min\left(\frac{f(\beta'_{i+1})q(\beta_i)}{f(\beta_i)q(\beta'_{i+1})}, 1\right)$

- Generalisation of accept-reject.

Random walk

$B'_{i+1} = B_i + \epsilon_i$, **i.e.** $B'_{i+1} \sim \mathcal{N}(B_i, \sigma^2)$.
Then $q(\beta'_{i+1}|\beta_i) = g(\beta'_{i+1} - \beta_i)$.

- To reduce this random walk algorithm to the Metropolis algorithm, consider the function g to be symmetric, centred at 0.
- For random walk algorithms, the acceptance probability does not depend on g.
- But, choice of g will affect range of values of B'_{i+1} and acceptance rate.
- $B'_{i+1} \sim \mathcal{N}(B_i, \sigma^2)$, $B'_{i+1} \sim \mathcal{U}[B_i - \delta, B_i + \delta]$. This scale δ affects correlation amongst samples abd convergence. Bigger δ implies the chain hovers around the same value over long periods of time, while small δ implies chains moves slowly away from current state.

Other than random walk

Some disadvantages of the random walk algorithm:

- wastage of a large number of steps between modes.
- since proposal is symmetric, nearly half the iterations involve revisiting states it has visited before.

Hence alternatives $\longrightarrow$ introduce a gradient of f in the definition of the proposal density of the Langevin algorithm:

$$\begin{aligned} B'_{i+1} &= B_i + \frac{\sigma^2}{2}\nabla \log f(B_i) + \sigma\epsilon_i, \quad \epsilon_t \sim g(\epsilon) \qquad (14) \\ \alpha(\beta_i, \beta'_{i+1}) &= \min\left(\frac{f(\beta'_{i+1})}{f(\beta_i)} \frac{g[(\beta_i - \beta'_{i+1})/\sigma - \sigma\nabla \log f(\beta'_{i+1})/2]}{g[(\beta_i - \beta'_{i+1})/\sigma - \sigma\nabla \log f(\beta_i)/2]}, 1\right). \end{aligned}$$

Here, σ is a scale. σ is fixed. But Langevin causes differential strengthening of the local modes.

Joint distribution and conditional distributions

In the case f is a multivariate probability distribution, transition is from one joint update to another.

f is a joint distr over β, sampling from the joint distribution of $\beta^{(1)}, \beta^{(2)}, \ldots, \beta^{(n)}$ can be difficult or impossible. In contrast, the conditional distribution of $\beta^{(i)}|\beta^{(1)}, \beta^{(2)}, \ldots, \beta^{(i-1)}, \beta^{(i+1)}, \ldots, \beta^{(n)}$ might be easy.

One way is Gibbs sampling (Geman & Geman 1984).

Gibbs sampling

Sought: j samples from $f(\beta^{(1)}, \beta^{(2)}, \ldots, \beta^{(n)})$.

- Start with $\boldsymbol{\beta}_0$.
- Let the current value of the vector be $\boldsymbol{\beta}_i$.
- Then sample $\beta_{i+1}^{(k)} \sim f(\beta_{i+1}^{(k)} | \beta_{i+1}^{(1)}, \beta_{i+1}^{(2)}, \ldots, \beta_i^{(k-1)}, \beta_i^{(k+1)}, \ldots, \beta_i^{(n)})$, $\forall\, k = 1, \ldots, n$.