The Gibbs Sampling Algorithm

Gibbs sampling is named after J. W. Gibbs but was not developed by him[1].

The idea that motivates Gibbs sampling is that it is usually easier to sample from 1-dimensional distributions that from d-dimensional ones. And for some target distributions it is easier to sample from their 1-dimensional *conditional* distributions than from either the full distribution or its 1-dimensional *marginalised* distributions.

The PDFs of the conditional distributions can be found from PDF of the joint distribution. For the target distribution $P(x^0, x^1 \ldots, x^{d-1})$ the 1-dimensional conditional distribution $P(x^k|x^{-k})$ (where x^{-k} denotes all the components of the vector x except x^k; i.e. $x^{-k} = \{x^0, \ldots, x^{k-1}, x^{k+1}, \ldots, x^{d-1}\}$) has a PDF proportional to the PDF of the joint distribution;

$$P(x^k|x^{-k}) = \frac{P(x^0, x^1 \ldots, x^{d-1})}{\int \mathrm{d}x^k \; P(x^0, x^1 \ldots, x^{d-1})} \propto P(x^0, x^1 \ldots, x^{d-1}). \tag{1}$$

In this context, proportional means that the denominator is not a function of x^k.

A basic version of the Gibbs sampling algorithm proceeds as follows. An arbitrary starting value is chosen (e.g. by drawing x_0 from some arbitrary initialisation distribution α on $\mathcal{X}$) and at each iteration one parameter component is chosen to be updated from the corresponding conditional target distribution.

[1] Geman & Geman (1984) "Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images", IEEE Transactions on Pattern Analysis and Machine Intelligence, **6** 6, 721–741 doi:10.1109/TPAMI.1984.4767596

Algorithm 0.1 Gibbs

$x_0 \sim \alpha$ ▷ Initialise
$i \leftarrow 0$
while $i \geq 0$ **do** ▷ Iterate $i = 0, 1, 2, \ldots$
  $k \sim \text{Cat}(w_0, w_1, \ldots, w_{d-1})$ ▷ Choose component
  $y \sim P(x^k | x_i^{-k})$
  $x_{i+1} \leftarrow (x_i^0, \ldots, x_i^{k-1}, y, x_i^{k+1}, \ldots, x_i^{d-1})$ ▷ Markov transition
  $i \leftarrow i + 1$
end while

The output of the Gibbs algorithm (Alg. 0.1) is the Markov chain $x_0, x_1, x_2, \ldots$

The weights $w_0, \ldots, w_{d-1}$ satisfy $\sum_{k=0}^{d-1} w_k = 1$. These are the parameters of a *categorical distribution* that controls how often each component gets updated. (The categorical distribution is a discrete distribution describing a random variable that can take on one of d possible values $\{0, 1, \ldots, d-1\}$, with the specified probabilities $\{w_0, w_1, \ldots, w_{d-1}\}$.) The weights can be chosen arbitrary by the user, but are often set equally; i.e. $w_k = 1/d$, for all k.

The Markov chain is formally infite. The Gibbs algorithm (Alg. 0.1) never terminates. Any practical implementation of Alg. 0.1 (or any of the other MCMC algorithms that will be described below) must periodically test to see if the finite Markov chain obtained so far looks to have converged and terminate the algorithm. Some possible convergence diagnostics will be discussed later in the course.

The weights $w_0, \ldots, w_{d-1}$ satisfy $\sum_{k=0}^{d-1} w_k = 1$. These are the parameters of a *categorical distribution* that controls how often each component gets updated. The weights can be arbitrary, but are often set equally; i.e. $w_k = 1/d$, for all k.

If the weights set equally, a nice feature of the Gibbs algorithm is that it has no free parameters and requires no user inputs, such as a choice for a proposal distribution.

Box 0.1: Gibbs sampling demo

We will use the Gibbs method to sample the following 2D target distribution,

$$P(x, y) = y \exp(-[xy + y]) \quad \text{with } 0 < x, y < \infty. \tag{i}$$

The 1D distribution of x conditioned on a value of y (i.e. treating y as constant) is

$$P(x|y) \propto \exp(-yx). \tag{ii}$$

We recognise this as being the exponential distribution; i.e. $x|y \sim \text{Exp}(y)$. Similarly, the 1D of y conditioned on a given value of x is

$$P(y|x) \propto y \exp(-[x + 1]y), \tag{iii}$$

which we recognise as the gamma distribution; i.e. $y|x \sim \text{Gamma}\left(2, \frac{1}{x+1}\right)$.

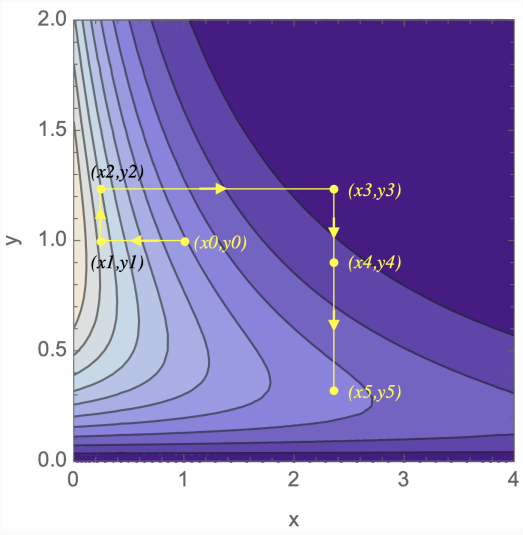
We know how to sample from both the exponential and gamma distributions so we can implement the Gibbs method.

We initialise the chain at position $(x_0, y_0) = (1, 1)$ (this is arbitrary).

First iteration. Randomly choose $k \in \{0, 1\}$. If we choose $k = 0$, we update the x-coordinate by drawing $x_1 \sim \text{Exp}(1)$ and move horizontally to $(x_1, y_1 = y_0)$.

Second iteration. If we choose $k = 1$, we update the y-coordinate by drawing $y_2 \sim \text{Gamma}(2, 1/(x_1 + 1))$; move horizontally to $(x_2 = x_1, y_2)$.

Five iterations are shown superposed on the target distribution below.



The Gibbs algorithm clearly defines a time-homogeneous Markov chain. This Markov chain will typically be irreducible (although, see counter example below). The algorithm is designed such that the target distribution is the stationary distribution of the chain.

Theorem 0.0.1. *The Gibbs algorithm (Alg. 0.1) produces a Markov chain $x_0, x_1, \ldots$ that satisfies the detailed balance condition with $\pi = P$.*

Proof. The Gibbs algorithm defines a Markov chain with transition probabilities are

$$\rho(y, x) = w_k \delta^{(d-1)}(y^{-k} - x^{-k}) P(y^k | x^{-k}), \tag{2}$$

Substituting this into the left-hand side of the detailed balance condition in Eq. **??** gives

$$\begin{aligned} \text{LHS} &= P(x)\rho(y, x) && (3)\\ &= P(x^k|x^{-k})P(x^{-k})w_k\delta^{(d-1)}(y^{-k} - x^{-k})P(y^k|x^{-k}), && (4) \end{aligned}$$

where on the second line we have used $P(x) = P(x^k|x^{-1})P(x^{-k})$. Substituting for $\rho(x, y)$ into the right-hand side of the detailed balance condition gives

$$\begin{aligned} \text{RHS} &= P(y)\rho(x, y) && (5)\\ &= P(y^k|y^{-k})P(y^{-k})w_k\delta^{(d-1)}(x^{-k} - y^{-k})P(x^k|y^{-k}) && (6)\\ &= P(y^k|x^{-k})P(x^{-k})w_k\delta^{(d-1)}(x^{-k} - y^{-k})P(x^k|x^{-k}), && (7) \end{aligned}$$

where on the last line we have used the properties of the Dirac delta function. Comparing Eqs. 4 and 7 shows that LHS = RHS which proves the result. □

Gibbs sampling is useful when sampling from the full target distribution is difficult, but sampling from the 1-dimensional conditional distributions of each variable is easier.

Note, although the Gibbs algorithm generally produces an irreducible Markov chain this is not strictly guaranteed; see, for example, the counter example in Fig. 1.

Although easy to implement, Gibbs sampling produces a Markov chain that can exhibit undesirable *random walk* behaviour and take a large number of iterations to *diffuse* across the target space. This is true in particular for highly correlated target distributions.

There are many variants of the of Gibbs sampling algorithm, some are described in the following (unnumbered) subsections. Many of these variants are designed to try and reduce or eliminate this random walk behaviour.

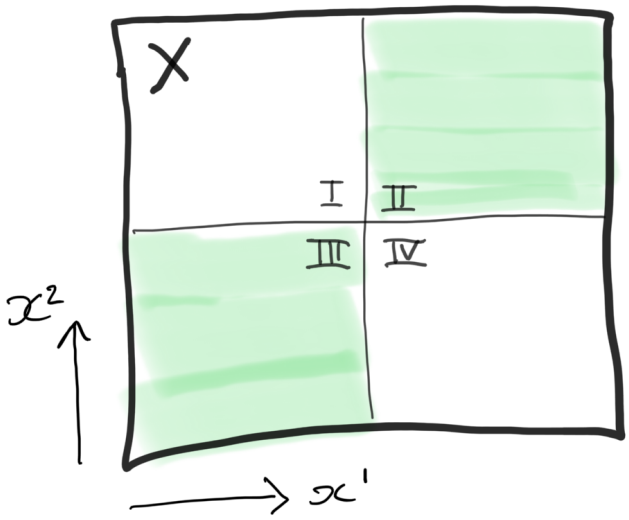


Figure 1: Consider a 2D target distribution with support only in the shaded regions II and III; i.e. $\int_I \mathrm{d}x\, P(x) = \int_{IV} \mathrm{d}x\, P(x) = 0$. A chain initialised in II will *never* move to III (or vice versa) under Gibbs evolution because the two steps required to do so (one horizontal and one vertical) necessarily enters one of the forbidden regions I or IV.

0.0.0.1 Gibbs "Sweep" Sampling

Instead of choosing just one component to update at each iteration, in this common variant of the Gibbs algorithm every component gets updated (in a fixed order) at every iteration in a process called a *sweep*.

This "sweep" version of the Gibbs sampling algorithm proceeds as follows. At each iteration every component is updated by drawing from the 1-dimensional conditional distribution conditioned on the values of all of the parameters updated so far.

Algorithm 0.2 Gibbs Sweep

$x_0 \sim \alpha$ ▷ Initialise
$i \leftarrow 0$
while $i \geq 0$ **do** ▷ Iterate $i = 0, 1, 2, \ldots$
  for $k = 0, 1, \ldots, d-1$ **do** ▷ The "sweep"
    $y^k \sim P(x^k | y^0, \ldots, y^{k-1}, x_i^{k+1}, \ldots, x_i^{d-1})$
  end for
  $x_{i+1} \leftarrow (y^0, \ldots, y^{d-1})$ ▷ Markov transition
  $i \leftarrow i + 1$
end while

This version of the Gibbs sampling algorithm does not satisfy the detailed balance condition, but does still converge to the target distribution.

Lemma 0.0.2. *The Gibbs sweep algorithm (Alg. 0.2) produces a Markov chain $x_0, x_1, \ldots$ that has P as its unique stationary distribution.*

Proof. We show this for the $(d = 2)$-dimensional case. Let $P(x) \equiv P(x^0, x^1)$ be the target distribution. Consider a move from (x^0, x^1) to (y^0, y^1); in the Gibbs sweep this proceeds in stages: $(x^0, x^1) \to (y^0, x^1) \to (y^0, y^1)$. The Markov transition probability is

$$\rho\big([y^0, y^1], [x^0, x^1]\big) = P(y^0|x^1)P(y^1|y^0), \tag{8}$$

$$= \frac{P(y^0, x^1)}{\int \mathrm{d}a\ P(a, x^1)} \frac{P(y^0, y^1)}{\int \mathrm{d}b\ P(y^0, b)}, \tag{9}$$

where the definition of the 1-dimensional conditional distributions in Eq. 1 has been used. Substitute this into the following integral for the *stationarity condition* in Eq. **??**,

$$\int \mathrm{d}x\ P(x)\rho(y, x) = \int \mathrm{d}x^0 \int \mathrm{d}x^1\ P(x^0, x^1)\rho\big([y^0, y^1], [x^0, x^1]\big) \tag{10}$$

$$= \frac{P(y^0, y^1)}{\int \mathrm{d}b\ P(y^0, b)} \int \mathrm{d}x^1\ \frac{P(y^0, x^1)}{\int \mathrm{d}a\ P(a, x^1)} \int \mathrm{d}x^0\ P(x^0, x^1) \tag{11}$$

$$= P(y_0, y_1) \tag{12}$$

$$= P(y). \tag{13}$$

This shows the *stationarity condition* is satisfied. The case of general d follows similarly. □

There are many possible variants of the Gibbs algorithm, but you have to be careful; it is easy to make a small, seemingly insignificant change that breaks everything. The following

not-Gibbs algorithm (Alg. 0.3) looks very similar to the Gibbs sweep algorithm (Alg. 0.2); however, this variation this doesn't work. The not-Gibbs algorithm produces a Markov that doesn't converge to the target distribution.

Algorithm 0.3 Not Gibbs (WARNING: this doesn't work!)

$x_0 \sim \alpha$ ▷ Initialise
$i \leftarrow 0$
while $i \geq 0$ **do** ▷ Iterate $i = 0, 1, 2, \ldots$
  for $k = 0, 1, \ldots, d-1$ **do** ▷ The "sweep"
    $y^k \sim P(x^k | x_i^0, \ldots, x_i^{k-1}, x_i^{k+1}, \ldots, x_i^{d-1})$
  end for
  $x_{i+1} \leftarrow (y^0, \ldots, y^{d-1})$ ▷ Markov transition
  $i \leftarrow i + 1$
end while

0.0.0.2 Blocked Gibbs Sampling

The basic Gibbs algorithm only updates one parameter component at each iteration; it does this by sampling from the 1-dimensional conditional distribution for that parameter. However, it is not necessary to sample only from 1-dimensional conditional distributions. Sometimes, it may be possible to sample from higher-dimensional conditional distributions thereby updating several parameters at each iteration.

The *blocked Gibbs* sampling algorithm achieves this by grouping parameters together into a number $g < d$ of groups, or *blocks*, of one or more components; e.g.

$$(x^0, x^1, \ldots, x^{d-1}) = (\theta^0, \theta^1, \ldots, \theta^{g-1}), \tag{14}$$

where each block $\theta^\mu \in \mathbb{R}^{d_\mu}$ consists of the parameters $\theta^\mu = (x^{\sum_{\mu'<\mu} d_{\mu'}}, \ldots, x^{\sum_{\mu'\leq\mu} d_{\mu'}})$, the index $\mu = 0, 1, \ldots, g-1$, and $\sum_{\mu=0}^{g-1} d_\mu = d$.

The blocked Gibbs algorithm requires the blocks θ^μ to be chosen such that it is possible to sample from the distributions conditioned on the parameters in the other blocks; $P(\theta^\mu | \theta^{-\mu})$.

A basic blocked Gibbs sampling algorithm proceeds as follows.

Algorithm 0.4 Blocked Gibbs

$x_0 \sim \alpha$ ▷ Initialise
$i \leftarrow 0$
while $i \geq 0$ **do** ▷ Iterate $i = 0, 1, 2, \ldots$
  $\mu \sim \text{Cat}(w_0, w_1, \ldots, w_{d-1})$ ▷ Choose block
  $\Theta^\mu \sim P(\theta^\mu | \theta^{-\mu})$
  $x_{i+1} \leftarrow (\theta^0, \ldots, \theta^{\mu-1}, \Theta^\mu, \theta^{\mu+1}, \ldots, \theta^{g-1})$ ▷ Markov transition
  $i \leftarrow i + 1$
end while

Lemma 0.0.3. *The blocked Gibbs algorithm produces a Markov chain $x_0, x_1, \ldots$ that satisfies the detailed balance condition with $\pi = P$.*

Proof. The proof is essentially identical to that of theorem 0.0.1. □

The version of the blocked Gibbs algorithm presented in Alg. 0.4 chooses which parameter block to update randomly at each iteration. It is of course also possible to perform a *blocked Gibbs sweep* algorithm (a combination of Algs. 0.2 and 0.4) where the parameter blocks are all updated sequentially at each iteration.