About BUGS (*)

The BUGS (Bayesian inference Using Gibbs Sampling) software [27] is an implementation of Gibbs sampling (and sometimes also other Metropolis-within-Gibbs updates). The user supplies only the model (using a specialised 'programming language') and the data, and the BUGS software outputs MCMC simulation of a given length.

The model is given in BUGS by specifying the joint distribution $\hat{p}$

$$\hat{p}(x^{(1:d)}) = p_1(x^{(1)}) \prod_{i=2}^{d} p_i(x^{(i)} \mid x^{(1:i-1)}),$$

where '$x^{(1:i)}$' is a shorthand for '$x^{(1)}, \ldots, x^{(i)}$. This specifies $\hat{p}$ fully, and on the other hand, any d-dimensional distribution $\hat{p}$ can be factored like this.

Usually, the model is sparse, that is, $p_i(x^{(i)} \mid x^{(1:i-1)})$ do not depend on all $x^{(1:i-1)}$, but on a subset of 'parent' variables. This reflects conditional independencies, which define a directed acyclic graph. For instance, a Markov chain with initial distribution λ and transition probability P could be given as above, where $p_1 = \lambda$ and $p_i(x^{(i)} \mid x^{(1:i-1)}) = P(x^{(i-1)}, x^{(i)})$ for $i \geq 2$.

The distribution of interest p is a conditional distribution of $\hat{p}$, given some 'data'. For instance, if the first two variables $X^{(1)} = x_*^{(1)}$ and $X^{(2)} = x_*^{(2)}$ were observed, and the others not, then the MCMC targets the posterior distribution of $X^{(3:d)} \mid X_{(1:2)} = x_*^{(1:2)}$ which satisfies

$$p(x^{(3:d)}) \propto p_u(x^{(3:d)}) = \hat{p}(x_*^{(1)}, x_*^{(2)}, x^{(3:d)}).$$

This can be simulated with (Metropolis-within-)Gibbs that updates only the unobserved $x^{(3:d)}$, one at a time.

6.7 Langevin-type proposals (*)

One way to construct proposal distributions $q(x, y)$ in the Metropolis-Hastings algorithm is to use random-walk like proposals, but also use $\nabla \log p(x)$ to 'inform' the direction of proposals, based on the shape of p around x. The simplest such proposal is of the 'Langevin' type, where

$$Y_k = X_{k-1} + \frac{\tau}{2} \nabla \log p(X_{k-1}) + \sqrt{\tau} Z, \qquad Z \sim N(0, \Sigma), \tag{16}$$

for some parameters $\tau \in (0, \infty)$ and covariance $\Sigma \in \mathbb{R}^{d \times d}$.[13] This algorithm is known as the *Metropolis adjusted Langevin* algorithm (MALA).

MALA is just Metropolis-Hastings algorithm with proposal $q(x, y) = N(y; x + \frac{\tau}{2} \nabla \log p(x), \tau\Sigma)$ corresponding to (16). Note that in this case, $q(x, y) \neq q(y, x)$ and so the ratio $q(y, x)/q(x, y)$ does not vanish from the acceptance probability!

---

13. The proposal (16) stems from an Euler discretisation of the (overdamped) Langevin diffusion of the form $\mathrm{d}X_t = \frac{1}{2} \nabla \log p(X_t) + \mathrm{d}B_t$, which is a continuous-time Markov process that admits p as its stationary distribution...

6.8 Hamiltonian Monte Carlo (*)

In recent years, a so-called *Hamiltonian Monte Carlo* (HMC) MCMC algorithm has gained attention [cf. 17]. Its proposal is based on a physics-motivated continuous-time process (*Hamiltonian dynamics*) involving an auxlliary *momentum* random vector.

The HMC is based on the target distribution $\tilde{p}(x, m) = p(x)q(m)$, where the auxiliary 'momentum' variable m has distribution q, a density of $N(0, \Sigma)$. The related 'Hamiltonian' can be written as

$$H(x, m) := -\log \tilde{p}(x, m) = U(x) + K(m),$$

where $U(x) := -\log p(x)$ and $K(m) := -\log q(m) = \frac{1}{2} m^T \Sigma^{-1} m$ (up to a constant). The proposal is *inspired by* the following system of differential equations:

$$\mathrm{d}m_t/\mathrm{d}t = -\nabla U(x_t) \qquad\qquad \mathrm{d}x_t/\mathrm{d}t = \Sigma^{-1} m_t. \tag{17}$$

These differential equations leave $\tilde{p}$ invariant (that is, if $(m_0, x_0) \sim \tilde{\pi}$, then also $(m_t, x_t) \sim \tilde{\pi}$ for any $t > 0$!), but of course we cannot solve them exactly. HMC uses a specific kind of numerical approximation of (17), (with $L \geq 1$ steps and with step size $\tau > 0$) in order to construct the proposals, and an acceptance ratio which ensures reversibility.

Algorithm 6.49 (Hamiltonian Monte Carlo)**.** Let $X_0 \equiv x_0$ s.t.$p(x_0) > 0$. For $k = 1, \ldots, n$:

(i) Draw $M_{k-1} \sim q$.
(ii) Calculate $(\hat{X}_k, \hat{M}_k) \leftarrow \mathrm{LF}(X_{k-1}, M_{k-1})$
(iii) Generate $U_k \sim \mathcal{U}(0, 1)$, and if $U_k \leq \alpha(X_{k-1}, M_{k-1}; \hat{X}_k, \hat{M}_k)$ *accept* and set $X_k = \hat{X}_k$, otherwise *reject* and set $X_k = X_{k-1}$, where the *acceptance probability* α is defined as follows:

$$\alpha(x, m; \hat{x}, \hat{m}) := \min\left\{1, \frac{\tilde{p}(\hat{x}, \hat{m})}{\tilde{p}(x, m)}\right\} = \min\left\{1, \exp\big(H(x, m) - H(\hat{x}, \hat{m})\big)\right\}.$$

where

LF(x_0, m_0):
For $t = 1, \ldots, L$:
(i) $\hat{m}_t \leftarrow m_{t-1} + \frac{\tau}{2} \nabla \log p(x_{t-1})$
(ii) $x_t \leftarrow x_{t-1} + \tau \Sigma^{-1} \hat{m}_t$
(iii) $m_t \leftarrow \hat{m}_t + \frac{\tau}{2} \nabla \log p(x_t)$
Return $(x_L, -m_L)$

(NB: The *momentum flip* in the end of LF($\cdot$) is unnecessary in practice, but included here for mathematical convenience...)

The HMC algorithm looks similar to Metropolis-Hastings (and indeed may be seen as an instance of a generalisation of Metropolis-Hastings).

The key observations required to check p-reversibility of the HMC are:

1. If $X_{k-1} \sim p$, then $(X_{k-1}, M_{k-1}) \sim \tilde{p}$.

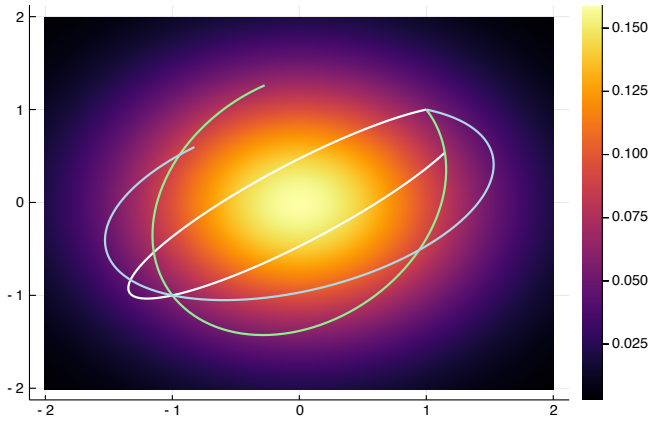

Figure 16: Three trajectories $(x_0, \ldots, x_L)$ of the leapfrog integrator starting from $x_0 = [1, 1]^T$ with three independent realisations of m_0 from $N(0, I_2)$. Here, $p = N(0, I_2)$ with density values shown as background color, $L = 100$ and $\tau = 0.05$.

2. The leapfrog integrator $\mathrm{LF}(\,\cdot\,)$ is *reversible*, in the sense that if $(\hat{x}, \hat{m}) = \mathrm{LF}(x, m)$, then $(x, m) = \mathrm{LF}(\tilde{x}, \tilde{m})$. (Or, equivalently, it is an involution: $\mathrm{LF}(\mathrm{LF}(x, m)) = (x, m)$.)
3. The leapfrog integrator $LF(\,\cdot\,)$ is isometric, or volume-preserving.

See [8] for details, as well as result showing the p-irreducibility of the HMC (which turns out to be a non-trivial exercise!).

There are a number of user-friendly implementations of (variants of) HMC. Stan [5] is the most popular, and has an interface similar to BUGS, allowing to build model from blocks. Stan can provide good performance in some scenarios where BUGS struggles, but it does not always outperform BUGS. If you intend to use Stan, there are certain inherent restrictions that come with it, which are good to know:

- Discrete variables cannot be unknowns.
- Unknowns need to be (easily transformable) to $\mathbb{R}$ (or $\mathbb{R}^d$).[14]
- Tail behaviour and geometry of p may have a dramatic influence in performance.
- The variables need to be (roughly) unit-scaled.

Even though the HMC (and its implementation in Stan) have showed great promise in many practical situations, they may not always provide a reliable outcome, and this may not be easy to predict.

This is in contrast with Gibbs sampling and random-walk proposals, which are rather well understood by now (including their weaknesses!).

14. Stan transforms $x > 0$ and $x \in (a, b)$ automatically with exponential and logistic transformations.

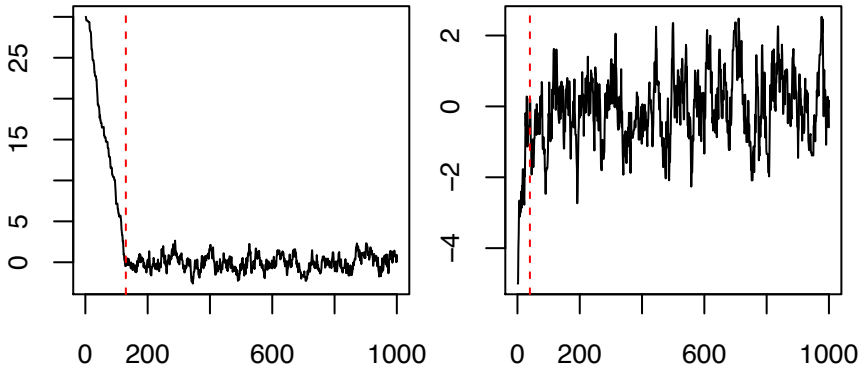

Figure 17: The first 1000 samples simulated from Example 6.27 with $a = 1$ and with with $x_0 = 30$ (left) and $x_0 = -5$ (right). The red vertical line indicates the 'burn-in time'.

7 MCMC convergence and mixing

With MCMC, there are two issues considering the reliability of the calculated averages: $I^{(n)}_{p,q,\mathrm{MH}}(f) = n^{-1}\sum_{k=1}^{n} f(X_k)$:

- The MCMC chain does not start from the invariant/stationary distribution, so $\mathbb{E}[f(X_k)] \neq \mathbb{E}_p[f(X)]$, and the difference may well be substantial for small k. This can induce significant *bias* to the estimator.
- It is not direct to assess the reliability of MCMC averages, because of the dependence of the random variables (X_k). The dependence usually adds *variance* to the estimator, when compared against simple Monte Carlo averages.

7.1 Burn-in bias

MCMC Markov chain X_n converges in distribution to p as $n \to \infty$ (under an aperiodicity condition, cf. Theorem 6.7). The common practice with MCMC is to discard b first values of the Markov chain $X_0, \ldots, X_b$, to minimise bias. It is assumed that X_{b+1} will have approximately the distribution p, and then use the estimator

$$\frac{1}{n-b}\sum_{k=b+1}^{n} f(X_k).$$

The initial period $X_0, \ldots, X_b$ is called *burn-in* of the MCMC.

Remark 7.1. Several statistics may be calculated in order to 'detect' a bias in MCMC. However, they usually rely on certain rather strong assumptions, such as the asymptotic normality, or at least unimodality of the target.

7.2 Asymptotic variance of MCMC

With classical Monte Carlo and importance sampling, the confidence intervals can be constructed with help of the CLT, and the associated variance is relatively straightforward to calculate.

Also Markov chains satisfy CLT in many cases. For example, we may record the following statement without proof.

Theorem 7.2. *If the Metropolis-Hastings Markov chain (X_k) on finite $\mathbb{X}$ is irreducible and aperiodic, then*

$$\sqrt{n}\big[I^{(n)}_{p,q,\mathrm{MH}}(f) - \mathbb{E}_p[f(X)]\big] \xrightarrow{n\to\infty} N(0, \sigma^2_{\mathrm{MH}}), \tag{18}$$

with $\sigma^2_{\mathrm{MH}} = \lim_{n\to\infty} n\mathrm{Var}\big(I^{(n)}_{p,\mathrm{MH}}(f)\big) < \infty$.

Remark 7.3. The CLT (18) holds quite generally, *under certain technical regularity conditions*. Because there are no general and easily verifiable conditions available, we shall not detail a more general form of the CLT, but assume it to hold.

We shall look next at an expression of the CLT variance (when finite), which gives a method to estimate the CLT variance.

Theorem 7.4. *Let $X_0, X_1, \ldots$ be a stationary Markov chain, that is, $X_0 \sim p$, where p is the invariant distribution. Suppose $f : \mathbb{S} \to \mathbb{R}$ such that $\mathbb{E}_p[f^2(X)] < \infty$ and denote $Y_k = f(X_k)$.*

Assuming $\sum_{k=1}^{\infty} \rho_k < \infty$ where $\rho_k := \mathrm{Corr}\big(Y_0, Y_k\big)$, we have

$$\lim_{n\to\infty} n\mathrm{Var}\bigg(\frac{1}{n}\sum_{k=1}^{n} f(X_k)\bigg) = \mathrm{Var}_p(f(X))\bigg(1 + 2\sum_{k=1}^{\infty}\rho_k\bigg).$$

Remark 7.5. With MCMC, X_0 is of course never exactly a sample of p, but as discussed earlier, X_b can be regarded to have approximately the distribution p whenever b is large. Therefore, if we apply Theorem 7.4 to $\tilde{X}_n := X_{b+n}$ for $n \geq 0$, the result is still relevant. (Rigorous extension to arbitrary initial measure is possible, but we shall not consider it here.)

Remark 7.6. Theorem 7.4 holds more generally, for any (weak-sense) stationary process $(Y_k)_{k\geq 1}$.

Proof of Theorem 7.4. Let us define $Y_k = f(X_k)$, and $\bar{Y}_k = Y_k - \mathbb{E}[Y_k]$, then

$$\begin{aligned}
\mathrm{Var}\bigg(\frac{1}{n}\sum_{k=1}^{n} f(X_k)\bigg) &= \frac{1}{n^2}\mathbb{E}\bigg[\bigg(\sum_{k=1}^{n}\bar{Y}_k\bigg)^2\bigg] \\
&= \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}\mathbb{E}[\bar{Y}_i\bar{Y}_j] \\
&= \frac{\mathrm{Var}_p\big(f(X)\big)}{n} + \frac{2}{n^2}\sum_{i=1}^{n}\sum_{j=i+1}^{n}\mathrm{Cov}(Y_i, Y_j) \\
&= \frac{\mathrm{Var}_p\big(f(X)\big)}{n}\bigg(1 + \frac{2}{n}\sum_{h=1}^{n-1}(n-h)\rho_h\bigg).
\end{aligned}$$

Multiply with n and take limits, and apply Lemma 7.7 to show that $n^{-1}\sum_{h=1}^{n-1} h\rho_h \xrightarrow{n\to\infty} 0$. □

Lemma 7.7 (Kronecker)**.** *Suppose* $(x_k)_{k\geq 1}$ *is a sequence of real numbers with* $\sum_{k=1}^{\infty} x_k = s \in \mathbb{R}$*. Then,* $n^{-1}\sum_{k=1}^{n} kx_k \xrightarrow{n\to\infty} 0$*.*

Definition 7.8. The *integrated autocorrelation time* of (Y_i) and the *effective sample size* of $(Y_1, \ldots, Y_n)$ are defined, respectively, as

$$\mathrm{IACT} := 1 + 2\sum_{i=1}^{\infty} \rho_i \qquad \text{and} \qquad n_{\mathrm{eff}} := \frac{n}{\mathrm{IACT}}.$$

The definitions of 'effective sample size' makes sense when we use Theorem 7.4 to deduce that for n large enough

$$\mathrm{Var}(I^{(n)}_{p,q,\mathrm{MH}}) \approx \frac{\mathrm{IACT}}{n}\mathrm{Var}_p\big(f(X)\big) = \frac{1}{n_{\mathrm{eff}}}\mathrm{Var}_p\big(f(X)\big).$$

Suppose then $(Z_1, \ldots, Z_{\lfloor n_{\mathrm{eff}}\rfloor})$ are independent from p, the classical Monte Carlo satisfies

$$\mathrm{Var}\Big(\frac{1}{\lfloor n_{\mathrm{eff}}\rfloor}\sum_{k=1}^{\lfloor n_{\mathrm{eff}}\rfloor} f(Z_k)\Big) = \frac{1}{\lfloor n_{\mathrm{eff}}\rfloor}\mathrm{Var}_p\big(f(X)\big).$$

So, the mean estimator based on the sample $X_1, \ldots, X_n$ from MCMC is (asymptotically) as efficient as the one based on $Z_1, \ldots, Z_{\lfloor n_{\mathrm{eff}}\rfloor} \overset{\text{i.i.d.}}{\sim} p$.

Remark 7.9 (*). Simple (and traditional) way to estimate IACT (and equivalently the asymptotic variance or n_{eff}) is to sum sample autocorrelations up to a truncation point, which is chosen based on an inspection of the sample autocorrelations. However, there are also reasonably straightforward and provably consistent estimators of the asymptotic variance [9].

Remark 7.10. Note that a MCMC sample $(X_k)_{k=1,\ldots,n}$ does *not* have a single effective sample size n_{eff}, but n_{eff} depends on the function. So if you are interested in different functions $f_1, \ldots, f_m : \mathbb{X} \to \mathbb{R}$, you need to calculate $n^{(1)}_{\mathrm{eff}}, \ldots, n^{(m)}_{\mathrm{eff}}$! This is particularly important if $\mathbb{X} = \mathbb{R}^d$, and $f_i(x) = x_i$, in which case the effective sample size of different coordinates may differ significantly.

7.3 Practical summary

When using MCMC, always do the following checks:

(i) Plot MCMC traces of the variables and key functions of the variables. They should look stationary after burn-in.

(ii) Make multiple MCMC runs from different initial state x_0 and check that the marginal distributions (or the estimators) look similar.
This test reveals if your chain is 'almost reducible'.

(iii) Plot sample autocorrelations of the variables and functions (e.g. `autocor` of `StatsBase`).

(iv) Calculate n_{eff} and check that it is reasonably large. Use it to construct a CI:

$$\Big[I_{p,q,\mathrm{MH}}(f) \pm \beta\frac{\hat{\sigma}_n}{\sqrt{n_{\mathrm{eff}}}}\Big],$$