Elliptical Slice Sampling

Francesca Panero and Xuewen Yu

October 20, 2017

1. Introduction

2. Methodology

2.1 Slice sampling

Slice sampling was introduced by R. M. Neal in 2003 in the homonym paper published in the Annals of Statistics. Neal's idea was to present a new approach to sample from a probability density function, that tried to overcome some drawbacks of the two major Monte Carlo Markov Chain methods of sampling: the Gibbs sampler and the Metropolis Hastings method.

The aim of these three methods is to sample from a probability density function from which we are not able to sample directly (very often, from posterior distributions in Bayesian models). Their basic idea is to construct a Markov chain that have as stationary distribution the target one we care about, and samples from this chain will eventually come from the desired distribution. Gibbs sampler exploits the possibility of sampling from the full conditional density functions of each parameter given all the others (that are usually simpler to deal with than the original joint distribution), since the complete vector of parameters will eventually be sampled from the joint distribution of these. No tuning of parameters of the chain is required, but we must be able to sample from these conditional distributions in order to implement the method, and sometimes this could not be the case. Nevertheless, some algorithms to overcome this problem have been proposed, in particular ARS (adaptive rejection sampling) and MARS (adaptive rejection Metropolis sampling). Metropolis Hasting, instead, initializes the chain with a value sampled from a proposal distribution from which we are able to sample from and that attains some nice links with the target distribution. This sample can be accepted or rejected according to a probability that depends on the likelihood of the new candidate point and the previous one. This method presents some scale parameters for which there does not exist a certain rule of decision; moreover, the proposal distribution does not always come from a straightforward reasoning.

Slice sampling tries to overcome these flaws in the previous methods. The basic idea lies in the fact that if we want to sample x from a distribution $p(x)$ that is proportional to a certain function $f(x)$, it would be sufficient just to sample uniformly from the area below $f(x)$. By defining an auxiliary random variable y, we exploit Gibbs sampler to sample from the two full conditional distributions. In particular, $y|x$ is distributed as an $Uniform(0, f(x))$ and $x|y$ is distributed as an uniform on the so called slice $S = \{x : y < f(x)\}$. The joint density of x and y is, then

$$p(x,y) = \begin{cases} 1/Z & \text{for } 0 < y < f(x) \\ 0 & \text{otherwise} \end{cases}$$

where $Z = \int f(x)dx$. From this it can easily be seen that

$$p(x) = \int_0^{f(x)} p(x,y)dy = \frac{1}{Z}f(x)$$

as desired. Sampling on the slice can be difficult, and it is sometimes substituted with some update for x which leaves invariant the uniform distribution.

2.2 Elliptical Slice Sampling

Elliptical Slice Sampling is a particular case of Monte Carlo markov chain method that avoids the tuning of parameters, simpler and often faster than other methods to sample from the posterior distribution of models with multivariate Gaussian prior.

Let $\mathbf{f}$ be the vector of latent variables and a Gaussian distribution with zero vector mean and covariance matrix Σ:

$$\mathbf{f} \sim \mathcal{N}(\mathbf{f}; \mathbf{0}, \Sigma) = |2\pi\Sigma|^{-1/2} \exp\left(-\frac{1}{2}\mathbf{f}^T\Sigma^{-1}\mathbf{f}\right)$$

Let

$$L(\mathbf{f}) = p(\text{data}|\mathbf{f})$$

be the likelihood function. Our target distribution is the posterior of this model:

$$p^*(\mathbf{f}) \propto \mathcal{N}(\mathbf{f}; \mathbf{0}, \Sigma)L(\mathbf{f})$$

3. Experiment

In this section, we will validate the elliptical slice sampling algorithm on 2 models: Gaussian regression and Log Gaussian Cox process.

3.1 Model Description

3.1.1 Gaussian Regression

Observations y_n are drawn from Normal distribution with mean f_n and variance σ_n^2, for $n = 1, ..., N$. Let N denote the sample size, D denote the number of dimensions. $\mathbf{f} = (f_1, ..., f_N) \sim N(0, \Sigma)$. To simulate $\mathbf{f}$, we define the covariance matrix as

$$\Sigma_{i,j} = \sigma_f^2 exp(-\frac{1}{2}\sum_{d=1}^{D}(x_{d,i} - x_{d,j})^2/l^2) \tag{1}$$

Covariance matrix Σ is computed by inputing $\mathbf{X}$, which is a $D \times N$ matrix. $\mathbf{X} = [\mathbf{x_1}, ..., \mathbf{x_N}]^\mathbf{T}$, where the column vector $\mathbf{x_n}$, $n = 1, ..., N$, is the 'feature' vector with D dimensions and f_n is a function of x_n. We will draw $\mathbf{x_n}$ from a D-dimensional unit hypercube for all n. We can simulate observations y_n after generating f_n and fixing σ_n^2 and σ_f^2. $y_1, ..., y_N$ are $i.i.d$ normal variables and $\mathbf{y}|\mathbf{f} \sim \mathbf{N}(\mathbf{f}, \sigma_\mathbf{n}^\mathbf{2}\mathbf{I})$, so the likelihood function as a function of $\mathbf{f}$ is:

$$L_r(\mathbf{f}) = \prod_{\mathbf{n=1}}^{\mathbf{N}} N(y_n; f_n, \sigma_n^2) \tag{2}$$

3.1.2 Log Gaussian Cox process

Cox process is a "doubly stochastic" Poisson process with a stochastic intensity measure [**?**]. Log Gaussian Cox process is introduced by Moller [**?**] as the Cox process where the logarithm of the internsity function is a Gaussian process. Mathematically, let y_n denote the observations. Then $y_n \sim Poisson(\lambda_n)$ with mean λ_n. The intensity function can be estimated given the log Gaussian Cox process observation within a bounded subset. This means we can partition the space finitely into N bins and y_n is the number of events in bin n for all $n = 1, ..., N$. We assume that every bin has a constant intensity function λ_n. Let m be the offset to the log mean λ_n, and define it as the sum of the mean log-intensity of the Poisson process and the log of the bin size [**?**].

$$y_n|f_n \sim Poisson(exp(f_n + m)) \tag{3}$$

$$\mathbf{f} \sim \mathbf{N}(\mathbf{0}, \mathbf{\Sigma}) \tag{4}$$

Σ is defined in Equation (1). Then the likelihood of $\mathbf{y}$ is:

$$L_p(\mathbf{f}) = \prod_{\mathbf{n=1}}^{\mathbf{N}} \frac{\lambda_{\mathbf{n}}^{\mathbf{y_n}} \mathbf{exp}(-\lambda_{\mathbf{n}})}{\mathbf{y_n}!}, \lambda_{\mathbf{n}} = \mathbf{e}^{\mathbf{f_n}+\mathbf{m}} \tag{5}$$

3.2 Implementation and Results

3.2.1 Gaussian Regression

Let $l = 1$, $\sigma_f^2 = 1$, $\sigma_n^2 = 0.3^2$. Firstly, we validate the Elliptical Slice Sampling algorithm on Gaussian regression model when $\mathbf{f}$ is bivariate Normal variable. In this case, let $N = 2$ and $D = 1$ such that $\{\mathbf{x_n}\}_{\mathbf{n=1}}^{\mathbf{2}}$ is one dimensional. Since both prior distribution and likelihood function are Gaussian, the posterior distribution of $\mathbf{f}$ should be Gaussian. We perform Henze-Zirkler's test to assess whether the outputs follow Bivariate Normal distribution. This is based on the measure of distance, which is nonnegative, between the characteristic function of the multivariate normality and the empirical characteristic function. The distribution of the test statistic is approximately log normal. We achieved a p-value much larger than 0.05, which indicates that there is no strong evidence to reject the null hypothesis that the outputs of the algorithm follows multivariate normal distribtuion. We can visualise the distribution of outputs in Figure 1. As a necessary condition for mutivariate normality, each variable should have normal distribution. This can be verified by QQ-plot as shown in Figure 2.

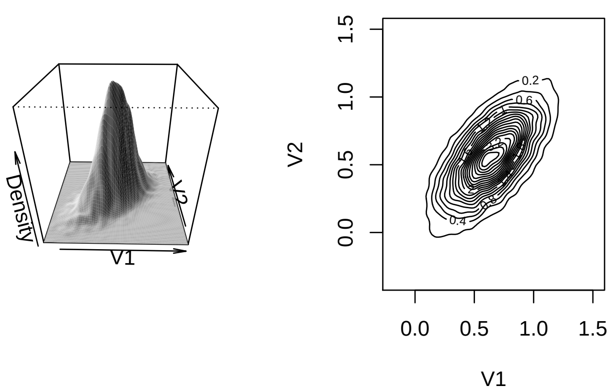


Figure 1: Perspective (left) and Contour (right) plots for bivariate outputs

Then we assess the performance of the algorithm on Gaussian regression when $N = 200$, i.e. $\mathbf{f}$ is 200-dimensional. As stated in the model description section, $\mathbf{f}$ can be generated by inputing X to covariance matrix. So we first simulated datasets X. In order to compare the performance of algorithm for different dimensions, i.e. D, of feature vectors, 3 synthetic datasets X_1,X_2,X_3 will be simulated with $D = 1, 5, 10$ respectively.

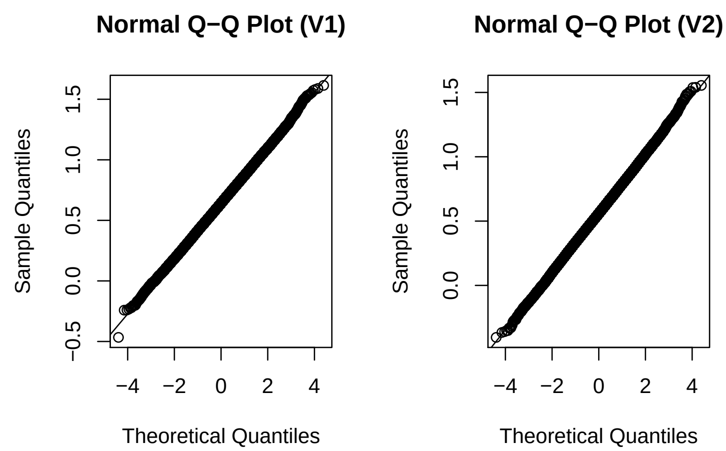
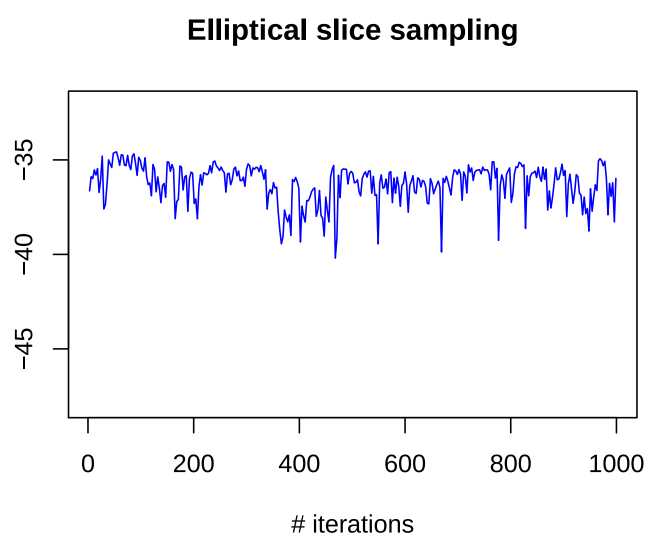


Figure 2: QQ plot of f_1 and f_2 from Elliptical Slice Sampling algorithm

Figure 3: Trace plot of log likelihood of 333 points in the first 1000 iterations by taking every 3 iterations



The traces of log-likelihoods and the first dimension of outputs for $D = 1$ are shown in Figure 3 and 4 respectively, which reveal frequent jumps in both chains. The effective sample size for it is 3177.

3.2.2 Log Gaussian Cox process

Data of mining disasters is provided by Jarret [?]. There are 191 events happening during 40550 days, which are partitioned into 811 bins such that each bin contains 50 days. Given the dates when the events happened, the number of events in each bin can be computed, which is y_n. Let $sigma_f^2 = 1$, $l = 13516$, $N = 811$, $D = 1$ and $m = log(191/811)$.

The figure sizes have been customised so that you can easily put two images side-by-side.

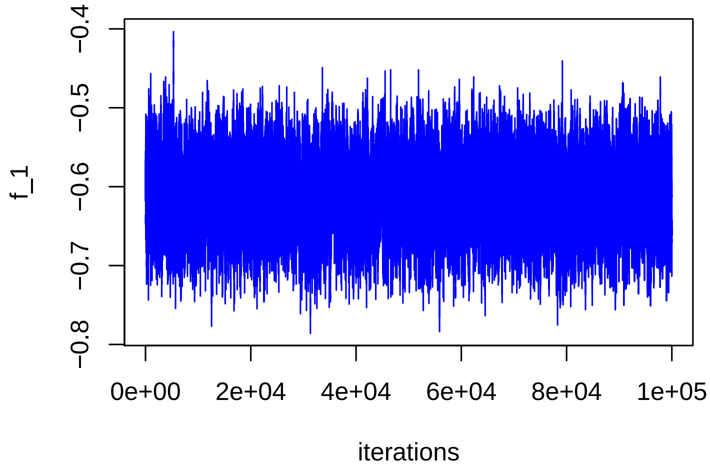
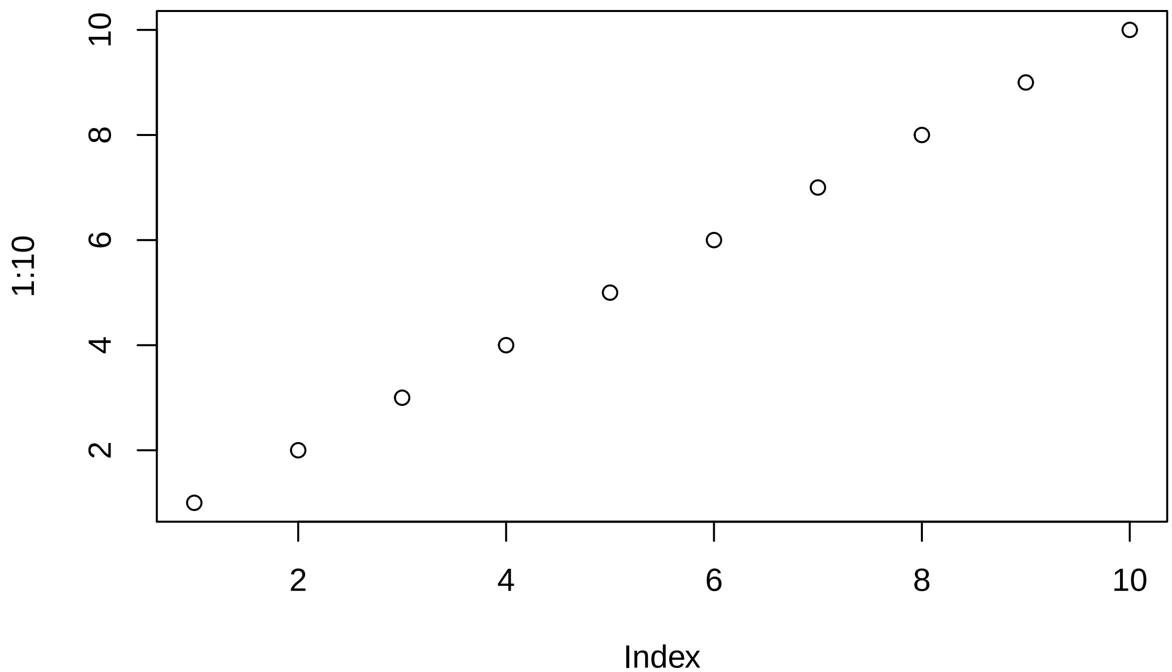
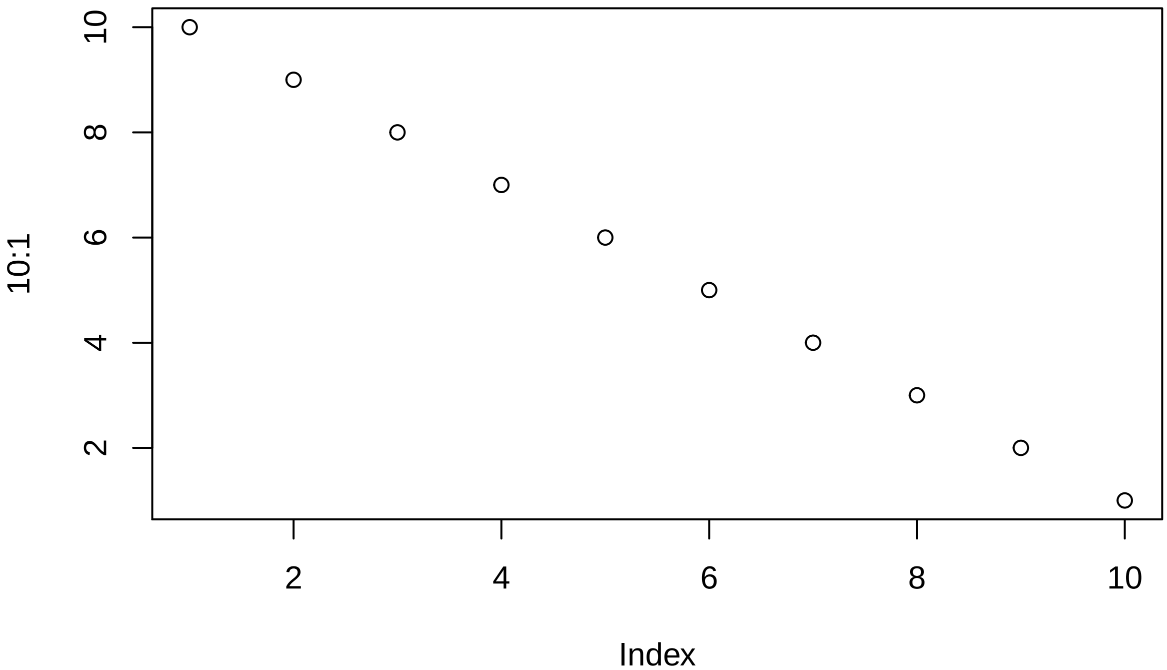


Figure 4: Trace plot of the first dimension dimension of output f of 100000 iterations

```
plot(1:10)
plot(10:1)
```





You can enable figure captions by `fig_caption: yes` in YAML:

```
output:
  rmarkdown::html_vignette:
    fig_caption: yes
```

Then you can use the chunk option `fig.cap = "Your figure caption."` in **knitr**.

More Examples

You can write math expressions, e.g. $Y = X\beta + \epsilon$, footnotes[1], and tables, e.g. using `knitr::kable()`.

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 Wag	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4
Datsun 710	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1
Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1
Hornet Sportabout	18.7	8	360.0	175	3.15	3.440	17.02	0	0	3	2
Valiant	18.1	6	225.0	105	2.76	3.460	20.22	1	0	3	1
Duster 360	14.3	8	360.0	245	3.21	3.570	15.84	0	0	3	4
Merc 240D	24.4	4	146.7	62	3.69	3.190	20.00	1	0	4	2
Merc 230	22.8	4	140.8	95	3.92	3.150	22.90	1	0	4	2
Merc 280	19.2	6	167.6	123	3.92	3.440	18.30	1	0	4	4

Also a quote using `>`:

> "He who gives up [code] safety for [code] speed deserves neither." (via)

[1] A footnote here.