Maximizing conditional entropy of Hamiltonian Monte Carlo sampler

Tengchao Yu\textsuperscript{*}, Hongqiao Wang\textsuperscript{†}, Jinglai Li\textsuperscript{‡}

Abstract

The performance of Hamiltonian Monte Carlo (HMC) sampler depends critically on some algorithm parameters such as the integration time. One approach to tune these parameters is to optimize them with respect to certain prescribed design criterion or performance measure. We propose a conditional entropy based design criterion to optimize the integration time, which can avoid some potential issues in the often used jumping-distance based measures. For near-Gaussian distributions, we are able to derive the optimal integration time with respect to the conditional entropy criterion analytically. Based on the results, we propose an adaptive HMC algorithm, and we then demonstrate the performance of the proposed algorithm with numerical examples.

1 Introduction

Generating samples from a target distribution is an important practice in Bayesian inference. The Hamiltonian Monte Carlo (HMC) method \cite{Duane1987, Carlin1996}, initially developed in the Physics community \cite{Duane1982}, has become a very popular tool for sampling distributions with continuously differentiable density functions. The HMC method receives considerable attention from both the Statistics and the Machine Learning community, and many highly interesting improvements of the method have been developed in the last decade, for example, \cite{Duane1987, Carlin1996, Neal2011, Betancourt2015, Girolami2011, Betancourt2017}, just to name a few. The HMC method proposes new samples by simulating particle movement of a Hamiltonian system constructed from the target density for a given amount of time, and it can often achieve better performance than standard random-walk based Markov chain Monte Carlo (MCMC) algorithms as it takes advantage of the gradient information of the target distribution \cite{Duane1987, Carlin1996}. However, it is well-known that the performance of the HMC algorithm depends critically on the algorithm parameters, and in particular on the numerical integration time step and the number of steps. As such, tuning these parameters becomes an important task in the implementation of HMC. To this end, a number of methods have been proposed to tune these parameters, e.g., the optimal tuning derived in the infinite dimensional limit \cite{Duane1987}, the No U-turn sampler (NUTS) \cite{Betancourt2017}, and the Adaptive HMC sampler (AHMC) \cite{Betancourt2017}. When designing a tuning scheme of HMC, one usually relies on a specific design criterion or performance measure, and for example, in both \cite{Duane1987, Betancourt2017}, the tuning schemes make use of the expected squared jumping distance (ESJD), which was proposed in \cite{Gelman1996} for random walk MCMC algorithms.

The ESJD criterion, which seeks the largest jumping distance, may cause some potential issues in the HMC sampler. An intuitive example for this is that the particles jumps from one side of the space to the other in each iteration, and in this case even though they may travel a very long distance, the Markov chain may not even be ergodic \cite{Duane1987}. In Section 2.3 we shall demonstrate that this is exactly the case when the HMC proposal is tuned by maximizing the ESJD for the simple Gaussian target distributions. To address the issue we propose a new design criterion based on the conditional entropy (CE) of the HMC proposal and we show that this criterion can lead to the very optimal proposal in the Gaussian case (as will be shown later, when the target distribution is exactly Gaussian, the proposal becomes the target distribution itself). Second, for

\textsuperscript{*}Department of Mathematical Sciences, Shanghai Jiao Tong University, 800 Dongchuan Rd, Shanghai, China.
\textsuperscript{†}School of Mathematics and Statistics, Central South University, Changsha, Hunan, China.
\textsuperscript{‡}Department of Mathematical Sciences, University of Liverpool, Liverpool L61 6ZL, UK. Email: jinglai.li@liverpool.ac.uk
near-Gaussian distributions, we can analytically derive the optimal integration time with respect to the CE criterion, and using the results we develop an adaptive HMC algorithm. As a byproduct, we are able to derive the optimal covariance matrix for the artificial momentum states. With numerical examples, we show that the proposed algorithm has a rather competitive performance against existing methods for a certain class of problems (namely those with near-Gaussian distributions). To summarize, the main contribution of the present work is two fold: first we provide a CE based design criterion to tune the integration time of HMC proposal; second, for near-Gaussian distributions, we derive an analytical result of the optimal integration time under the CE criterion, using which we design an adaptive HMC algorithm.

2 The maximum conditional entropy HMC sampler

2.1 The HMC algorithm

Let $x$ be a $n$-dimensional random variable with distribution

$$\pi(x) = \exp(-U(x)).$$  \hspace{1cm} (1)

By using the Hamiltonian dynamics, one can design a very efficient scheme to draw samples from the distribution $\pi(x)$. Specifically one constructs an artificial Hamiltonian system that has $x$ as its position variable and the function $U(x)$ as the potential energy of the system, and then introduces an auxiliary variable $p$ to be the momentum of the system with kinetic energy $K(p)$. In practice, the kinetic energy is usually taken to be of a quadratic form:

$$K(p) = \frac{1}{2}p^T M^{-1} p,$$  \hspace{1cm} (2)

where $M$ is a positive definite symmetric matrix. The dynamics of the constructed system is governed by

$$\frac{dx}{dt} = \frac{\partial H}{\partial p}, \hspace{0.5cm} \frac{dp}{dt} = -\frac{\partial H}{\partial x}.$$  \hspace{1cm} (3)

Suppose the current position is $x_0$ and the HMC performs the following steps in each iteration,

- Sample an initial momentum state $p_0$ from the distribution $N(0, M)$;
- Solve the Eq. (3) with initial condition $(x_0, p_0)$, for a given amount of time $T$, obtaining the new states $(x_T, p_T)$;
- Accept the new position $x_T$ with probability

$$\min[1, \exp(H(x_0, p_0) - H(x_T, p_T))].$$  \hspace{1cm} (4)

Since the Hamiltonian system preserves its total energy,

$$H(x(t), p(t)) = U(x(t)) + K(p(t)),$$

it should be clear that if we can solve the Eq. (3) exactly, the acceptance probability is simply one. In practice, however, Eq. (3) must be solved numerically and as a result the acceptance probability is lower than one. The leapfrog algorithm [13] is commonly used to solve the system for its ability to preserve the time-reversibility, and moreover, when the integration time $T$ is large, one often use multiple leapfrog steps to integrate the ODE system (3) from 0 to $T$. 
2.2 Maximizing the conditional entropy

Now suppose that we can solve the Hamiltonian system (3) exactly, achieving the 100% acceptance probability, and an important question here is how to choose an optimal integration time $T$ for the best efficiency of the HMC algorithm? To answer this question, we first need to establish an optimality criterion or a performance measure of the HMC sampler.

A very natural choice for such a performance measure is the ESJD:

$$\mathbb{E}[\|x_T - x_0\|^2],$$

which was first proposed by Pasarica and Gelman [15] and was later applied to HMC in [3, 17]. As is mentioned earlier the main idea behind the ESJD criterion is to seek a proposal that moves the largest distance from the present location, as will be shown later, maximizing the ESJD may not lead to a good HMC proposal in certain circumstances and in particular, in the simple Gaussian case, the resulting chain becomes periodic and loses its ergodicity. This issue then motivates us to consider alternative design criterion.

Our intuition here is to choose that parameters that can provide the most “random” proposal, and thus we propose to determine the parameters by maximizing the conditional entropy (CE):

$$H(x_T|x_0) = \int \log \pi(x_T|x_0) \pi(x_T, x_0) d x_T d x_0$$

which quantities the uncertainty or randomness in $x_T$ provided $x_0$ is known. When $\pi(x_T|x_0)$ is Gaussian, the CE in Eq. (6) is (up to a constant) equal to

$$\mathbb{E}_{x_0} \left[ \log \det \text{Cov}[x_T|x_0] \right].$$

Thus when $\pi(x_T|x_0)$ is Gaussian or near-Gaussian, we can choose to optimize Eq. (7) for that it is usually easier to evaluate than Eq. (6).

2.3 Near-Gaussian distributions

In this section we consider the situation where the target distribution is near-Gaussian. The rationale here is that we can first derive the optimal algorithm parameters including $T$ by assuming the target distribution is exactly Gaussian, and it should be sensible to use derived parameter values for the actual target distribution provided it does not deviate too far from Gaussian. Such near-Gaussian distributions arise frequently in Bayesian inference problems, especially for those with a large amount of data, thanks to the asymptotic normality property [8]. Following this idea we shall assume the target distribution can be well approximated by a multivariate Gaussian $\pi(x) \approx \mathcal{N}(\mu, \Sigma)$, where $\mu$ and $\Sigma$ are the mean and the covariance of $\pi(x)$ respectively. Without loss of generality we shall assume $\mu = 0$ to keep the calculations simple.

The main result. In this case, we aim to determine the following algorithm parameters: the momentum covariance matrix $M$, and the integration time $T$. Here for the convenience of theoretical analysis, we need to impose a constraint on $M$: it commutes with $\Sigma$, and that is, we shall choose $M$ from $\mathcal{M}_F^+$, which is the class of all positive definite matrices that commutes with $\Sigma$. As is discussed in Section 2.2 these parameters will be determined by solving

$$\max_{\{M \in \mathcal{M}_F^+, T \in R^+\}} \mathbb{E}_{x_0} \left[ \log \det \text{Cov}[x_T|x_0] \right],$$

where $R^+$ denotes all positive real numbers. The following theorem states the main result of the work:

**Theorem 1.** Suppose that $\pi(x) = \mathcal{N}(0, \Sigma)$, $x_0 \sim \pi(x)$, $p_0 \sim \mathcal{N}(0, M)$, and $x(t)$ is the solution of the Hamiltonian system (3) with $K(p)$ given by Eq. (2). Let $x_T = x(T)$, and a solution of the optimization problem (8) is

$$M = \Sigma^{-1}, \quad \text{and} \quad T = (2m + 1)\pi/2,$$

for an arbitrary non-negative integer $m$. 

3
Figure 1: The scatter plots of the samples in the \((x, p)\) space. Top: the results of the HMC with three different values of \(T\) near the optimal value with respect to CE; Bottom: the results with three different values of \(T\) near the optimal value with respect to ESJD.

**Proof:** First we define \(A = M^{-1} \Sigma^{-1}\), and since \(M^{-1}\) and \(\Sigma^{-1}\) commutes, \(A\) is also a positive definite matrix. Now suppose that we conduct an eigenvalue decomposition of \(A\), yielding \(A = V \Lambda V^{-1}\), where \(V\) is a square matrix whose the \(i\)-th column is the eigenvector \(v_i\) of \(A\) and \(\Lambda\) is the diagonal matrix whose diagonal elements are the corresponding eigenvalues, \(\Lambda_{ii} = \lambda_i\).

Under the assumption that the target distribution \(\pi(x) = \mathcal{N}(0, \Sigma)\), the Hamiltonian system (3) can be solved analytically,

\[
x(t) = \sum_{i=1}^{n} (a_i \cos \sqrt{\lambda_i} t + b_i \sin \sqrt{\lambda_i} t) v_i,
\]

\[
p(t) = M \sum_{i=1}^{n} \sqrt{\lambda_i} (-a_i \sin \sqrt{\lambda_i} t + b_i \cos \sqrt{\lambda_i} t) v_i,
\]

where the coefficients \(\{a_i, b_i\}_{i=1}^{n}\) are determined via the initial conditions:

\[
x_0 = \sum_{i=1}^{n} a_i v_i, \quad p_0 = M \sum_{i=1}^{n} \sqrt{\lambda_i} b_i v_i.
\]

By some elementary calculations, we obtain from Eqs. \((10)\) that,

\[
C = \text{Cov}(x_T | x_0) = \sum_{i=1}^{n} \sum_{j=1}^{n} \mathbb{E}[b_i b_j] \sin(\sqrt{\lambda_i} T) \sin(\sqrt{\lambda_j} T) v_i v_j^T.
\]

Now using Eq. \((10a)\) and the facts that \(x_0 \sim \mathcal{N}(0, \Sigma)\) and \(p_0 \sim \mathcal{N}(0, M)\), we derive that \((b_1, ..., b_n)\) follows a multivariate Gaussian distribution and

\[
\mathbb{E}[b_i, b_j] = \Gamma_{i,j}, \quad \text{where} \quad \Gamma = \Lambda^{-\frac{1}{2}} V^{-1} M^{-1} V^{-T} \Lambda^{-\frac{1}{2}},
\]

for any \(0 \leq i, j \leq n\).

Substituting Eq. \((12)\) into Eq. \((11)\) yields,

\[
C_{i,j} = \sum_{i'=1}^{n} \sum_{j'=1}^{n} \Gamma_{i',j'} \sin(\sqrt{\lambda_{i'}} T) V_{i,i'} \sin(\sqrt{\lambda_{j'}} T) V_{j,j'}.
\]
Alternatively, we can rewrite Eq. (13) in a matrix form:

\[ C = V \Lambda \sin \Sigma \Lambda \sin V^T, \]

with

\[ \Lambda \sin = \text{diag}[\sin \sqrt{\lambda_1} T, ..., \sin \sqrt{\lambda_n} T]. \]  

(15)

It follows that

\[ \det C = \det (V \Lambda \sin \Sigma \Lambda \sin V^T) = \prod_{i=1}^{n} \lambda_{\Sigma,i} \frac{1 - \cos 2 \sqrt{\lambda_i} T}{2}, \]

where \( \lambda_{\Sigma,1}, ..., \lambda_{\Sigma,n} \) are the eigenvalues of \( \Sigma \). Now recall that we want to find a solution of

\[ \max_{\{M \in M^+_\Sigma, T \in R^+\}} E_{x_0}[\log \det \text{Cov}[x_T|x_0]] \]

\[ = \log \sum_{i=1}^{n} \frac{1 - \cos 2 \sqrt{\lambda_i} T}{2} + \log \det \Sigma. \]

(17)

It is obvious that the minimum of the problem is attained at

\[ \sqrt{\lambda_i} T = \frac{(2m + 1)}{2} \pi, \quad \text{for} \quad i = 1...n, \]

where \( m \) is an arbitrary integer. A special choice is that

\[ T = \frac{(2m + 1)}{2} \pi, \]

and \( \lambda_i = 1 \) for \( i = 1...n \). It follows immediately that matrix \( A \) is identity, which implies that \( M = \Sigma^{-1} \).

In practice, since the Hamiltonian system needs to be solved numerically, it is certainly desirable to use smaller integration time \( T \), and thus in the HMC algorithm we set \( m = 1 \) and \( T = \pi/2 \). It is also worth noting that, while the trick to improve the efficiency of HMC by choosing \( M = \Sigma^{-1} \) has long been known mainly from an intuitive perspective [13], we are able to provide a justification for it based on the MCE principle.

**Comparison with ESJD: an univariate example.** Here we use a simple univariate example to demonstrate the difference between the CE criterion and the ESJD. Namely we assume that the target distribution is \( N(0, k) \), \( K(p) = p^2/(2m) \) and \( p_0 \sim N(0, m) \). In this case, by some elementary calculations we can derive that an integration time that maximizes ESJD [5] is \( T = \sqrt{km} \pi \), and the associated proposal is: \( x_T = -x_0 \), which means that the samples will only jump between two locations \((x_0 \text{ and } -x_0)\) and the resulting chain is certainly not ergodic. On the other hand, the optimal integration time with respect to CE is \( T = \sqrt{km} \pi/2 \), and the associated proposal is \( \pi(x_T|x_0) = N(0, k) \), i.e., to sample directly from the target distribution, which is the very optimal distribution in this case. We refer to the the SM for a detailed derivation of the optimal integration time with respect to CE and ESJD. It is also easy to derive that, the efficiency of the proposal behaves periodically with respect to \( T \) in this case. In example 1, we use numerical experiments to validate the theoretical analysis conducted here.

**Determining the number of leapfrog steps.** Now we have derived the optimal integration time \( T \), and as is discussed earlier, we need to integrate the Hamiltonian system [4] from 0 to \( T \) using the leapfrog algorithm. In particular, we usually need to perform leapfrog integration multiple times to achieve the necessary numerical precision, and to this end, the number of leapfrog steps, conventionally denoted by \( L \), is another key algorithm parameter to be specified. If we increase \( L \), the numerical integration becomes more accurate and the acceptance probability approaches to 1, and the price to pay is that more leapfrog steps means higher computational cost. We note here that in most existing works both \( L \) and \( \epsilon = T/L \) need to be determined simultaneously, and in that case higher acceptance probability does not necessarily imply a
Algorithm 1 Maximum Conditional Entropy HMC Sampler

Require: $U(x)$, $\text{Acc}_\text{min}$, $N_0$, $L_0$, $L_{\text{max}}$, $N_M$, $N_L$, $\rho$, $I_{\text{max}}$

1: Initialization: draw $N_0$ samples $\{x_1, x_2, \ldots, x_{N_0}\}$ using standard HMC sampler.
2: Estimate the sample covariance matrix $\hat{\Sigma}$ of $\{x_1, x_2, \ldots, x_{N_0}\}$;
3: Let $M = \hat{\Sigma}^{-1}$;
4: Let $T = \pi/2$;
5: Let $\text{Acc}_\text{old} = 0$, $L_\text{old} = L_0$ and $L = L_0$, and $I_L = 1$;
6: for $t = N_0$ to $N_{\text{max}}$ do
7: $\epsilon = T/L$;
8: Draw $p_t \sim N(0, M)$;
9: $(x^*, p^*) = \text{leapfrog}(x_t, p_t, M, U(x), \epsilon, L)$;
10: Draw $u \sim U(0, 1)$;
11: if $u < \min\{1, \exp(H(x_t, p_t) - H(x^*, p^*))\}$ then
12: $x_{t+1} = x^*$;
13: else
14: $x_{t+1} = x_t$;
15: if $t < N_M$ then
16: Update the sample covariance matrix $\hat{\Sigma}$ by including the new sample $x_{t+1}$;
17: Let $M = \hat{\Sigma}^{-1}$;
18: if $t \mod N_L = 0$ and $I_L = 1$ then
19: Let $\text{Acc}$ be the average acceptance probability of the last $N_L$ samples;
20: if $L = L_{\text{max}}$ then
21: $I_L = 0$;
22: if $\text{Acc}/L < \text{Acc}_\text{old}/L_\text{old}$ then
23: $L = L_\text{old}$;
24: else
25: if $\text{Acc} > \text{Acc}_\text{min}$ then
26: if $\text{Acc}/L < \text{Acc}_\text{old}/L_\text{old}$ then
27: $I_{\text{count}} = I_{\text{count}} + 1$;
28: if $I_{\text{count}} \geq I_{\text{max}}$ then
29: $I_L = 0$, $L = L_\text{old}$;
else
30: $\text{Acc}_\text{old} = \text{Acc}$, $L_\text{old} = L$;
31: $I_{\text{count}} = 0$;
32: $L = \min\{[\rho L_\text{old}], L_{\text{max}}\}$
else
33: $\text{Acc}_\text{old} = \text{Acc}$, $L_\text{old} = L$
34: $I_{\text{count}} = 0$;
35: $L = \min\{[\rho L_\text{old}], L_{\text{max}}\}$.

better proposal even without considering the computational cost for computing the proposal. In our method, however, since the total integration time $T$ is fixed, it is reasonable to assume that increasing $L$, which in turn increases the acceptance probability, improves the performance of the algorithm. Based on this idea, we shall seek the value of $L$ that provides the highest acceptance rate per computational cost (which is usually measure by $L$). Namely, we use an adaptive scheme to gradually increase $L$ until the average acceptance rate per $L$ does not improve.

Adaptively estimating the covariance matrix. Another important issue in the method is that it requests the knowledge of the target covariance matrix. Here we follow the idea of the adaptive MCMC algorithms to estimate the covariance from the sample history and specifically the target covariance is updated using
the method given in [10]. We emphasize here that, the adaptive algorithm does not require an accurate estimation of the target covariance in advance (i.e. from a pilot runs); rather it adaptively improves the estimate of the covariance as the sample size increases [11]. We present the complete algorithm in Alg. 1 and provide some remarks on the algorithm in the following:

- In step 9 the function leapfrog($x_t, p_t, M, U(x), \epsilon, L$) represents to solve the Hamiltonian system specified by $M$ and $U(x)$ from the initial condition $x_t$ and $p_t$, using the leapfrog algorithm with stepsize $\epsilon$ for $L$ steps.

- In step 13 we fix the maximum number of iterations in which we update the parameters to be $N_M$.

- Steps 16 and 17 update matrix $M$ from samples, and the formulas for these computations are Eqs. (2) and (3) [10] respectively;

- In steps 33 and 37 we update $L$ by multiplying the current value of it by a factor $\rho$.

- In steps 21 and 29 we set the condition for stopping the update of $L$: the $Acc/L$ quantity does not improve in $I_{\text{max}}$ consecutive steps and the minimal acceptance probability $Acc_{\text{min}}$ has been reached.

3 Numerical examples

In this section, we provide three examples to demonstrate the performance of the proposed maximum CE (MCE) method. The first two are mathematical examples and the last one is a real-world application.

Example 1: Univariate Gaussian. We first use a Gaussian example to compare the CE criterion with ESJD. Specifically we take target distribution to be the univariate standard Gaussian distribution $N(0, 1)$, the kinetic energy to be $K(p) = p^2/2$, and $p_0$ to be standard Gaussian as well. As has been discussed, the optimal integration time $T$ computed by the proposed MCE method is $\pi/2$, while that computed by ESJD is $\pi$. We thus take six different values of $T$: $T = 0.45\pi, 0.49\pi, 0.5\pi$, which are close to the optimal value by MCE, and $T = 0.95\pi, 0.99\pi, \pi$, which are close to the optimal value predicted by ESJD. In all the tests we fix the starting position to be $x = 1$, and since for this toy problem, we have the analytical solution (see the SM) and so we do not need to use leapfrog. To demonstrate the behavior of the proposals, we plot the first 100 samples produced by each proposal in Fig. 1. First it can be seen here that if we use $T = \pi$ which is exactly the optimal value predicted by ESJD, the samples are fixed at $x = -1$ and $x = 1$ and in this case the sampler fails completely. The samples deviate away from the two points as $T$ is slightly perturbed from $\pi/2$, but most samples are still concentrated near the two locations, indicating poor performance of the proposals. On the other hand, the samples drawn by the proposals determined with the MCE method distribute well according to the target distribution, which demonstrates that the integration time predicted by the MCE method does lead to good proposals in this Gaussian example.

| parameters | $\theta_1$ | $\theta_2$ | $\theta_3$ | $\theta_4$ | $\theta_5$ | $\theta_6$ | $\theta_7$ | $\theta_8$ | $\mu$ | $\tau$
|------------|------------|------------|------------|------------|------------|------------|------------|------------|-------|-------
| MCES       | 10.3       | 7.5        | 6.0        | 7.3        | 5.0        | 6.0        | 10.0       | 7.8        | 7.3   | 5.7   |
|            | (7.1)      | (5.8)      | (6.9)      | (6.1)      | (5.9)      | (6.2)      | (6.1)      | (7.0)      | (4.2) | (3.6) |
| NUTS       | 10.1       | 7.4        | 6.0        | 7.2        | 5.1        | 6.0        | 9.8        | 7.7        | 7.2   | 5.5   |
|            | (7.0)      | (5.8)      | (6.8)      | (6.0)      | (5.8)      | (6.1)      | (6.1)      | (6.8)      | (4.2) | (3.7) |

Table 1: The posterior mean and the standard deviation (in parenthesis) computed by MCES and NUTS for Example 2.

Example 2: Eight school problem. Our second example is the Eight School problem in [8], which is a hierarchical Bayesian inference application. While referring interested readers to [8], we here omit all the application background and proceed directly to the mathematical setup of the problem. Specifically,
let \( \{\theta_1, \ldots, \theta_8\} \) be the parameters of interest, and \( \{(y_1, \sigma_1), \ldots, (y_8, \sigma_8)\} \) be the data. Let \( \mu \) and \( \tau \) be the hyperparameters specifying the prior of \( \theta_1, \ldots, \theta_8 \). The hierarchical model is:

\[
\begin{align*}
\mu & \sim \text{Uniform}[-15, 15], \quad \tau \sim \text{Uniform}[0, 15] \\
\theta_i & \sim N(\mu, \tau), \quad y_i \sim N(\theta_i, \sigma_i), \quad i = 1 \ldots 8.
\end{align*}
\]

We use the same data as those in [8] in the inference problem, and we sample the posterior distribution with the MCE-HMC method and the NUTS. First to validate the proposed method, we draw \( 10^5 \) samples from the posterior distribution using both the MCE sampler and NUTS, and the posterior means and variances for all the parameters obtained by both methods are reported in Table 1, from which we can see that the results computed by both methods agree well with each other, up to certain statistical errors. We then plot the obtained posterior histograms for \( \mu \) and \( \tau \) in Fig. 2, and we can see from the figure that both distributions are significantly apart from Gaussian. Next we compare the performance of the MCE sampler and NUTS. To do so, for each method we repeat the simulations 100 times, and in each simulation we draw \( 10000 \) samples with additional \( 1000 \) samples use in the burn-in period. The algorithm parameters of MCE sampler used in this example are summarized in Table 2. In this work NUTS is implemented using the Matlab package [14] by Nishimura. We compute the ESS/L for the results of each simulation, which is used as a performance measure of the samplers, and we show the box-plots of the ESS/L results in Fig. 3. The ESS is computed using the code provided in [14]:

\[
\text{ESS} = \frac{n}{1 + 2 \sum_{k=1}^{\infty} \rho(k)}
\]

where \( n \) is the number of samples and \( \rho(k) \) is the auto-correlation of lag \( k \). The plots show that the MCE sampler achieves evidently higher ESS per \( L \) than NUTS, even for the two dimensions that are evidently non-Gaussian. Additional test results (using different algorithm parameters) are provided in the SM.

**Example 3: Bayesian Logistic regression with the German credit data.** Our last example is the German credit data available at [6], a popular benchmark problem for Logistic regression. Simply put, this problem aims to classify people described by a set of attributes as good or bad credit risks. Here we use the modified version with all numerical attributes [6], which has 1000 instances each with a 24-dimensional numerical input and a binary output. For further details of the dataset, please refer to the description of the data set at [6]. Here the problem is formulated as a Logistic regression [12] and the regression coefficients \( \beta = (\beta_0, \beta_1, \ldots, \beta_n) \) with \( n = 24 \) are estimated with a Bayesian inference. The prior distribution is chosen to be standard Gaussian: \( \beta \sim N(0, I) \). Just like the previous example, for each method we repeat the simulations 100 times, and in each simulation we draw 10000 samples with additional 1000 samples used for burn-in. The algorithm parameters are the same as those used in Example 2. To validate the MCES method, we have verified that the posterior means and variances computed by both methods agree well with each other (the

| Parameter | \( L_0 \) | \( L_{\text{max}} \) | \( \rho \) | Acc_{\text{min}} | \( N_{\text{max}} \) |
|-----------|-----------|-----------------|-----|--------------|-----------|
| Value     | 1         | 60              | 1.2 | 60%          | 10000     |

| Parameter | \( N_L \) | \( N_M \) | \( N_0 \) | \( I_{\text{max}} \) |
|-----------|-----------|-----------|-----------|-----------------|
| Value     | 200       | 2000      | 1000      | 2               |

Table 2: Algorithm parameters of MCE-HMC.

Figure 2: The histograms of the posterior samples for \( \mu \) (left) and \( \tau \) (right).
actual results are provided in the SM). We estimate the ESS/L of each dimension for each simulation, and we then compute the mean and the standard deviation of the 100 trials. We show the results in Table 3, and the results indicate that the MCES has a higher average ESS/L on 16 of the total 25 dimensions, and more importantly the lowest ESS/L across all the dimensions for NUTS is 0.053, while that for MCES is 0.17, which is three times higher than that of NUTS. These results suggest that the MCES method has a good performance in this Logistic problem arising from real world applications.

Table 3: The mean and standard deviations of ESS/L for MCES and NUTS in Example 3. For all the dimensions, the higher mean values are marked in bold.
4 Conclusions

In this work we propose a new CE based design criterion for tuning the algorithm parameters in HMC. We show that the CE criterion can address some limitation of the distance based design criteria such as ESJD. For near-Gaussian distribution we are able to derive the analytical solution to the resulting optimization problem. We then develop an adaptive HMC algorithm based on the results. Numerical examples demonstrate that the proposed method has rather good performance even when the target distributions are considerably different from Gaussian. Several issues and limitations of the method need to be addressed in the future. First, Algorithm 1 terminates the adaptation after a fixed number of iterations, which may potentially affect the efficiency of the algorithm, if the adaptation is terminated prematurely. To this end, an interesting question is that whether the chain can converge without such a mandatory termination. Second, in this work we are only able to conduct a rather limited comparison of the proposed method with NUTS. We acknowledge that the state-of-the-art implementations of NUTS (the package used here only provides a basic implementation of NUTS) are available on the STAN platform [4], and we plan to conduct a thorough comparison of the proposed method with NUTS using STAN for a large range of problems, to better understand the advantages and the limitations of the MCE method. Finally, the most serious restriction of the method is, of course, the near-Gaussian assumption, which makes the method unsuitable for strongly non-Gaussian distributions, e.g., those with multiple modes. It is thus of significant interest to apply the CE criterion to strongly non-Gaussian distributions and develop suitable HMC algorithms for them. We plan to investigate these problems in the future.

Appendix: Derivation of the MCE and the maximum ESJD solutions

This section provides details of the derivation of the optimal integration time with respect to CE and with respect to ESJD when the target distribution is $N(0, k)$. We also take $K(p) = p^2/(2m)$ and $p_0 \sim N(0, m)$. In this case it is easy to derive that the solution of the Hamiltonian system is

$$
x(t) = A \cos(\sqrt{\frac{1}{km}} t + \phi_0), \quad p(t) = -A \sqrt{\frac{m}{k}} \sin(\sqrt{\frac{1}{km}} t + \phi_0)
$$

and the initial conditions are

$$
x_0 = A \cos(\phi_0), \quad p_0 = -A \sqrt{\frac{m}{km}} \sin(\phi_0).
$$

From Eq. (18), we obtain

$$
x_T = A \cos(\sqrt{\frac{1}{km}} T + \phi_0) = [x_0 \cos(\sqrt{\frac{1}{km}} T) + p_0 \sqrt{\frac{k}{m}} \sin(\sqrt{\frac{1}{km}} T)].
$$

As $p_0 \sim N(0, m)$, it follows immediately that

$$
\pi(x_T|x_0) = N \left( x_0 \cos(\sqrt{\frac{1}{km}} T), k \sin^2(\sqrt{\frac{1}{km}} T) \right).
$$

Next we shall consider the two criteria separately.

First we consider the CE criterion, which seeks to maximize $\mathbb{E} \log \text{Var}[x_T|x_0]$, and since $\pi(x_T|x_0)$ is univariate Gaussian, it is equivalent to

$$
\max_{T>0} \mathbb{E}_{x_0} [\log \text{Var}[x_T|x_0]] := \log[k \sin^2(\sqrt{\frac{1}{km}} T)]
$$

$$
= \log[1 - \cos(2\sqrt{\frac{1}{km}} T)] + \log(\frac{k}{2}).
$$

(20)
It is easy to see that the solution is,

\[ T = \frac{1}{2}\sqrt{km}(\pi + 2J\pi), \]

where \( J \) is an arbitrary non-negative integer. Certainly we should take \( J = 0 \) and so we obtain the smallest \( T \) as larger \( T \) implies higher computational cost of the numerical integration. Thus the optimal solution with respect to the CE criterion is

\[ T = \frac{\pi}{2}\sqrt{km}. \]

Next we shall derive the optimal value of \( T \) with respect to ESJD. That is we want to solve,

\[ \max_{T > 0} \mathbb{E}_{x_0, p_0}[||x_T - x_0||^2]. \]

Once again from Eq. (18) we obtain,

\[ x_T - x_0 = A\cos(\sqrt{\frac{1}{km}}T + \phi_0) - A\cos(\phi_0) = x_0[\cos(\sqrt{\frac{1}{km}}T) - 1] + p_0\sqrt{\frac{k}{m}}\sin(\sqrt{\frac{1}{km}}T). \]

Then we have,

\[ \mathbb{E}_{x_0, p_0}[||x_T - x_0||^2] = 2k[1 - \cos(\sqrt{\frac{1}{km}}T)]. \]

Thus, maximizing the ESJD becomes,

\[ \max_{T > 0} 2k[1 - \cos(\sqrt{\frac{1}{km}}T)], \]

and the solution is

\[ T = \sqrt{km}(\pi + 2J\pi), \]

and for the same reason as above we take \( J = 0 \), which yields the optimal integration time with respect to the ESJD,

\[ T = \pi\sqrt{km}. \]

References

[1] Christophe Andrieu and Johannes Thoms. A tutorial on adaptive mcmc. *Statistics and computing*, 18(4):343–373, 2008.

[2] Alessandro Barp, François-Xavier Briol, Anthony D Kennedy, and Mark Girolami. Geometry and dynamics for markov chain monte carlo. *Annual Review of Statistics and Its Application*, 5:451–471, 2018.

[3] Alexandros Beskos, Natesh Pillai, Gareth Roberts, Jesus-Maria Sanz-Serna, Andrew Stuart, et al. Optimal tuning of the hybrid monte carlo algorithm. *Bernoulli*, 19(5A):1501–1534, 2013.

[4] Bob Carpenter, Andrew Gelman, Matthew D Hoffman, Daniel Lee, Ben Goodrich, Michael Betancourt, Marcus Brubaker, Jiqiang Guo, Peter Li, and Allen Riddell. Stan: A probabilistic programming language. *Journal of statistical software*, 76(1), 2017.

[5] Tianqi Chen, Emily Fox, and Carlos Guestrin. Stochastic gradient hamiltonian monte carlo. In *International conference on machine learning*, pages 1683–1691, 2014.

[6] Dheeru Dua and Casey Graff. UCI machine learning repository, 2017.

[7] Simon Duane, Anthony D Kennedy, Brian J Pendleton, and Duncan Roweth. Hybrid monte carlo. *Physics letters B*, 195(2):216–222, 1987.
[8] Andrew Gelman, Hal S Stern, John B Carlin, David B Dunson, Aki Vehtari, and Donald B Rubin. *Bayesian data analysis*. Chapman and Hall/CRC, 2013.

[9] Mark Girolami and Ben Calderhead. Riemann manifold langevin and hamiltonian monte carlo methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 73(2):123–214, 2011.

[10] Heikki Haario, Eero Saksman, Johanna Tamminen, et al. An adaptive metropolis algorithm. *Bernoulli*, 7(2):223–242, 2001.

[11] Matthew D Hoffman and Andrew Gelman. The no-u-turn sampler: adaptively setting path lengths in hamiltonian monte carlo. *Journal of Machine Learning Research*, 15(1):1593–1623, 2014.

[12] David W Hosmer Jr, Stanley Lemeshow, and Rodney X Sturdivant. *Applied logistic regression*, volume 398. John Wiley & Sons, 2013.

[13] Radford M Neal. Mcmc using hamiltonian dynamics. In Steve Brooks, Andrew Gelman, Galin Jones, and Xiao-Li Meng, editors, *Handbook of markov chain monte carlo*, pages 131–162. Chapman & Hall/CRC, 2011.

[14] Aki Nishimura. (recycled) no-u-turn-sampler: Matlab implementation. [https://github.com/aki-nishimura/NUTS-matlab](https://github.com/aki-nishimura/NUTS-matlab), 2017.

[15] Cristian Pasarica and Andrew Gelman. Adaptively scaling the metropolis algorithm using expected squared jumped distance. *Statistica Sinica*, 20(1):343–364, 2007.

[16] Heiko Strathmann, Dino Sejdinovic, Samuel Livingstone, Zoltan Szabo, and Arthur Gretton. Gradient-free hamiltonian monte carlo with efficient kernel exponential families. In *Advances in Neural Information Processing Systems*, pages 955–963, 2015.

[17] Ziyu Wang, Shakir Mohamed, and Nando Freitas. Adaptive hamiltonian and riemann manifold monte carlo. In *International Conference on Machine Learning*, pages 1462–1470, 2013.

[18] Yichuan Zhang and Charles A Sutton. Quasi-newton methods for markov chain monte carlo. In *Advances in Neural Information Processing Systems*, pages 2393–2401, 2011.