Computational Statistics II

Unit B.2: MALA algorithm & Hamiltonian Monte Carlo

Tommaso Rigon

University of Milano-Bicocca

Ph.D. in Economics, Statistics and Data Science

Unit B.2

Main concepts

- The MALA algorithm
- Optimal scaling and pre-conditioning
- Hamiltonian Monte Carlo

- Associated **R** code is available on the website of the course

Main references

- Girolami, M. and Calderhead, B. (2011). Riemann manifold Langevin and Hamiltonian Monte Carlo methods. *JRSS-B*, **73**(2), 123–214.
- Neal, R. M. (2011). MCMC using Hamiltonian dynamics. CRC press.
- Roberts, G. O. and Rosenthal, J. S. (2001). Optimal scaling for various Metropolis-Hastings algorithms. *Statistical Science*, **16**(4), 351–367.

Limitations of the MH algorithm

- The random walk Metropolis algorithm is very popular and appealing among practitioners because it is **general** and **easy to implement**.
- In addition, the RWM is quite **robust** to the choice of the tuning (scaling) parameters.
- Unfortunately, this seductive simplicity leads to performance that scales poorly with **increasing dimension** and **increasing complexity** of the target density.
- Even when the proposal is optimally chosen, the RWM relies on **local moves** that leads to slow mixing, especially in high dimensions.
- The proposal distribution of a RWM is indeed **randomly exploring** the interesting parts of the posterior density, without taking into account its structure.

Gradient-based methods

- Intuitively, we are seeking for better proposal distributions that incorporate the **structure** of the target density and therefore will lead to **faster mixing**.
- Let $\pi(\boldsymbol{\theta} \mid \boldsymbol{X})$ be a **continuous** and **differentiable** posterior density in $\mathbb{R}^p$.
- The key quantity we will exploit is the **gradient** of the logarithm of the target density, written

$$\nabla_{\boldsymbol{\theta}} \log \pi(\boldsymbol{\theta} \mid \boldsymbol{X}) = \nabla_{\boldsymbol{\theta}} \log \pi(\boldsymbol{X} \mid \boldsymbol{\theta}) + \nabla_{\boldsymbol{\theta}} \log \pi(\boldsymbol{\theta}).$$

- The gradient of the target density is often available in **closed form** and it does not require the knowledge of the normalizing constant.
- The gradient informs about the **direction** and the **rate of increase** of a given function.
- For instance, for a given value $\boldsymbol{\theta}^{(r)}$ and $\epsilon > 0$, the update

$$\boldsymbol{\theta}^{(r+1)} \longleftarrow \boldsymbol{\theta}^{(r)} + \epsilon \, \nabla_{\boldsymbol{\theta}} \log \pi(\boldsymbol{\theta}^{(r)} \mid \boldsymbol{X}),$$

  leads to an increase of $\pi(\boldsymbol{\theta} \mid \boldsymbol{X})$, for ϵ small enough. This corresponds to the well-known **gradient ascent** method.

Langevin diffusions

- Incorporating the gradient in the MCMC procedure is an intuitive and appealing idea: this will push the Markov chain towards values having **higher density**.
- Besides this intuition, there is a strong theoretical justification for gradient adjusted MH proposals, based on **Langevin diffusions**.
- Let $\boldsymbol{B}^{(t)}$ be a p-dimensional standard Brownian motion.
- We consider a continuous-time stochastic process $\boldsymbol{\theta}^{(t)}$ satisfying the following stochastic differential equation

$$\mathrm{d}\boldsymbol{\theta}^{(t)} = \frac{1}{2}\nabla_{\boldsymbol{\theta}} \log \pi(\boldsymbol{\theta}^{(t)} \mid \boldsymbol{X})\mathrm{d}t + \mathrm{d}\boldsymbol{B}^{(t)}.$$

- **Key result**. The stationary distribution of the above Langevin diffusion is the posterior density $\pi(\boldsymbol{\theta} \mid \boldsymbol{X})$.

The MALA algorithm

- In practice, we need to consider **discrete approximations** of the Langevin diffusion, for example using the so-called **Euler method**.
- This leads to the following discrete-time stochastic process

$$\boldsymbol{\theta}^{(r+1)} = \boldsymbol{\theta}^{(r)} + \frac{\epsilon^2}{2} \nabla_{\boldsymbol{\theta}} \log \pi(\boldsymbol{\theta}^{(r)} \mid \boldsymbol{X}) + \epsilon \, \boldsymbol{z}^{(r)},$$

  for any chosen discretization step $\epsilon > 0$, and with iid $\boldsymbol{z}^{(r)} \sim \mathrm{N}_p(0, \boldsymbol{I}_p)$.
- This discrete approximation is no longer guaranteed to converge to $\pi(\boldsymbol{\theta} \mid \boldsymbol{X})$.
- There is a delicate trade-off between the accuracy of this approximation ($\epsilon \to 0$) and the sampling efficiency, increasing as ϵ grows.
- This issue is solved by treating the above distribution as **proposal density** of a Metropolis-Hastings algorithm.

The MALA algorithm

- The Metropolis adjusted Langevin algorithm (MALA) therefore can be seen as a specific MH algorithm with proposal distribution

$$(\boldsymbol{\theta}^* \mid \boldsymbol{\theta}) \sim \mathsf{N}_p \left(\boldsymbol{\theta} + \frac{s_p^2}{2} \nabla_{\boldsymbol{\theta}} \log \pi(\boldsymbol{\theta} \mid \boldsymbol{X}),\ s_p^2 I_p \right),$$

  where $s_p^2 > 0$ is some **tuning parameter** that must be carefully chosen.

- **Remark**. This proposal distribution is not symmetric as in the RWM case, therefore the acceptance probability takes into account also the proposal densities, namely

$$\alpha = \min \left\{ 1, \frac{\pi(\boldsymbol{\theta}^* \mid \boldsymbol{X})}{\pi(\boldsymbol{\theta} \mid \boldsymbol{X})} \frac{q(\boldsymbol{\theta} \mid \boldsymbol{\theta}^*)}{q(\boldsymbol{\theta}^* \mid \boldsymbol{\theta})} \right\}.$$

- There is strong theoretical and empirical evidence showing that the price paid for computing the gradient is compensated by a much faster mixing in several problems.

Asymptotics and optimal scaling

- As for the RWM, some insights about the **optimal choice** of s_p^2 can be gained by looking at the **asymptotic behaviour** of the MALA, for large values of p.

- Let us assume again that the posterior distribution has the following form

$$\pi(\boldsymbol{\theta} \mid \boldsymbol{X}) = \prod_{j=1}^{p} f(\theta_j), \qquad \text{var}(\boldsymbol{\theta} \mid \boldsymbol{X}) = \sigma^2 I_p$$

  namely the components of $\boldsymbol{\theta}$ are iid from some density f, satisfying the same **smoothness conditions** mentioned in **Unit B.1** and described in Roberts et al. (1997).

- It turns out that in order to get sensible asymptotic results, in this case we need to set

$$s_p^2 = \ell^2 / p^{1/3},$$

  as opposed to the p^{-1} term we have in the RWM case.

Diffusion processes

- As for the RWM case, let us define a **speeded-up** continuous-time stochastic process,

$$Z^{(t)} = \theta_1^{([tp^{1/3}])},$$

parametrized to make jumps every $p^{-1/3}$ units.

Theorem (Roberts and Rosenthal, 1997)

The continuous time stochastic process Z weakly converges to

$$Z \xrightarrow{d} W, \qquad p \to \infty,$$

where W is a diffusion process satisfying the stochastic differential equation

$$\mathrm{d}W^{(t)} = h(\ell)^{1/2}\mathrm{d}B^{(t)} + \frac{h(\ell)\nabla \log f(W^{(t)})}{2}\mathrm{d}t,$$

where the **speed of the diffusion** is

$$h(\ell) = \ell^2\, 2\Phi\left(-\mathcal{J}\ell^3\right),$$

for some constant $\mathcal{J}$ that only depends on f.

Speed of diffusion $h(\ell)$

- As in the RMW, the speed of diffusion $h(\ell)$ is strictly related to the asymptotic variance. Hence, we will look for the **optimal** ℓ that maximizes the diffusion $h(\ell)$.

- Let $\tau_g(\ell)$ be the integrated autocorrelation of the MALA. Then for large p it holds that

$$\tau_g(\ell)^{-1} \approx h(\ell)\, e_g\, p^{-1/3},$$

  where $e_g > 0$ is some constant depending only on $g(\cdot)$.

- **Remark**. These findings imply that the MALA algorithm has complexity $\mathcal{O}(p^{1/3})$, which is considerably more efficient than the $\mathcal{O}(p)$ complexity of the RWM.

- In practice, these theoretical results suggest that the MALA algorithm should have better performance especially in high-dimensional problems (large p).

Speed of diffusion and acceptance rate

- Let us recall that the acceptance rate of a MH algorithm is informally defined as

$$A_p(\ell) = \lim_{R\to\infty} \frac{\text{“\# of accepted moves”}}{R}.$$

- Then, it can be shown that in the MALA case we have

$$\lim_{p\to\infty} A_p(\ell) = A(\ell) = 2\Phi\left(-\mathcal{J}\ell^3\right),$$

  implying, as in the RWM case, that **speed of diffusion** relates to the **acceptance rate**.

- In practice, the optimal value ℓ_{opt} maximizing $h(\ell)$ does not require to calculate $\mathcal{J}$.

- Indeed, the asymptotic acceptance rate evaluated at the optimum is such that

$$A(\ell_{\text{opt}}) \approx 0.574,$$

  so ℓ can be chosen by trial and error or by means of adaptive methods.

- **Remark**. This means that the optimally scaled MALA has faster mixing than the RWM.

Getting practical

- Let us consider again the **logistic regression** problem under iid Gaussian priors and variance 100, using the Pima indian dataset.
- In this example, we **do not standardize the predictors**, to make the problem more challenging.
- Recall that the **gradient** of the log-posterior in this case is easily obtained as follows

$$\nabla_{\boldsymbol{\theta}} \log \pi(\boldsymbol{\theta} \mid \boldsymbol{X}) = \nabla_{\boldsymbol{\theta}} \log \pi(\boldsymbol{X} \mid \boldsymbol{\theta}) + \nabla_{\boldsymbol{\theta}} \log \pi(\boldsymbol{\theta}) = \boldsymbol{X}^{\intercal}(\boldsymbol{y} - \boldsymbol{\pi}) - \beta/100,$$

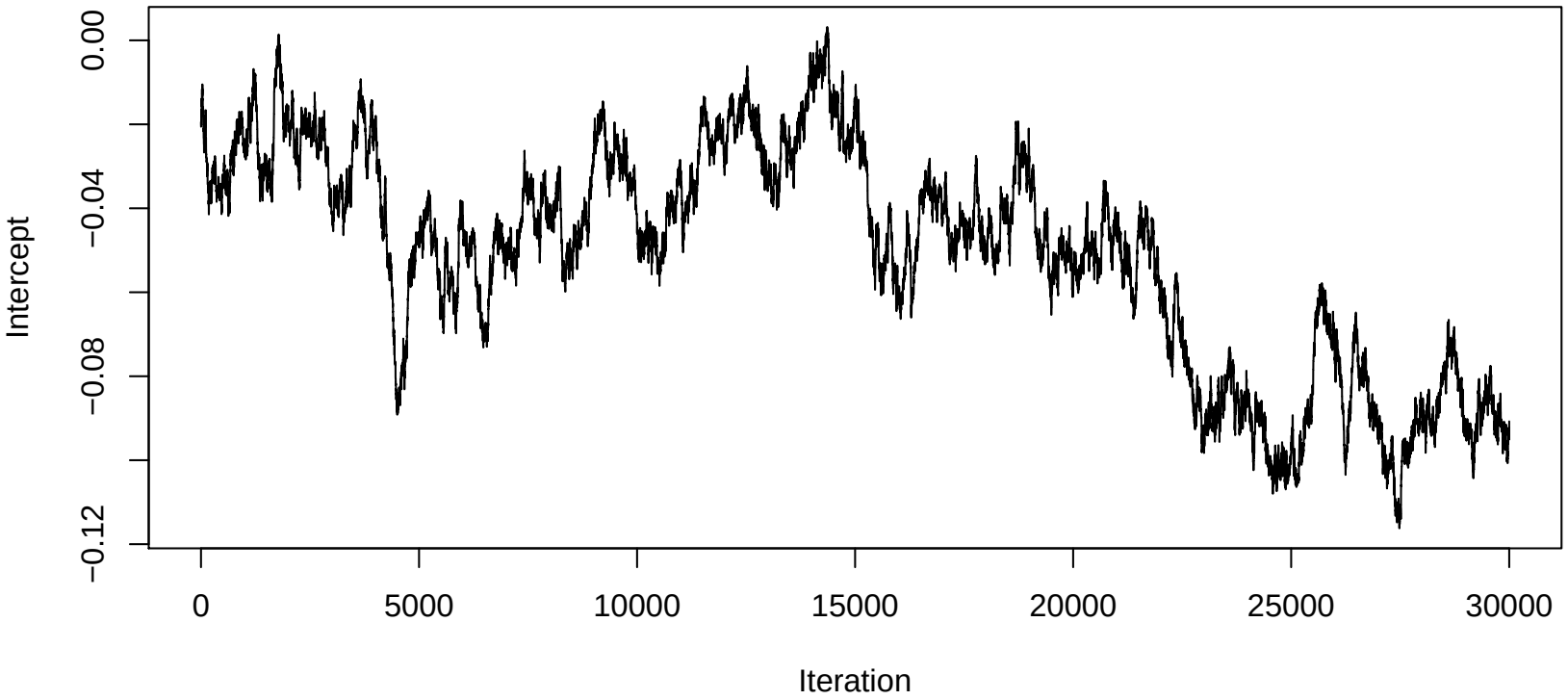
  where each entry of $\boldsymbol{\pi}$ is $[1 + \exp\{-(x_{i1}\beta_1 + \cdots + x_{ip}\beta_p)\}]^{-1}$, for $i = 1, \ldots, n$.
- The mathematical simplicity of the gradient follows from the fact that the logistic regression belong to an **exponential family**.

MALA algorithm in practice

- After some trial and error, we set $s_p = 0.0017$, to get the optimal acceptance rate.
- However, the results are a **complete disaster**. The chain does not reach convergence and the samples are garbage.

```
# Scaling parameter (after few trials)
s <- 0.0017

# Running the MCMC (R = 30000, burn_in = 5000)
fit_MCMC <- as.mcmc(MALA(R = R, burn_in = burn_in, y, X, s, S = diag(ncol(X))))

summary(effectiveSize(fit_MCMC)) # Effective sample size (beta)
#    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
#   2.900   9.358  27.201  44.321  46.238 166.223

summary(R / effectiveSize(fit_MCMC)) # Integrated autocorrelation time (beta)
#    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
#   180.5   728.7  1208.5  2671.3  3283.2 10343.4

summary(1 - rejectionRate(fit_MCMC)) # Acceptance rate (beta)
#    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
# 0.5638  0.5638  0.5638  0.5638  0.5638  0.5638
```

MALA algorithm in practice



- The true posterior density places mass in values in the interval $(-12, -7)$. The sampled values are not even close to that region $\Longrightarrow$ the results are garbage.

What went wrong?

- These performance issues are not specific of MALA. Indeed, a vanilla RWM with isotropic covariance matrix $\boldsymbol{S} = s^2 I_p$ would also perform quite badly.
- In both cases, the theory assumes a posterior distribution with iid components. Broadly speaking, we need the posterior variances of each components $\theta_1, \ldots, \theta_p$ to be similar.
- If we **standardize the predictors**, the results improve remarkably. However, this is a workaround that can not be applied in general models.
- In the RWM case we solved this issue by considering a covariance matrix $\boldsymbol{S}$ depening on the posterior covariance matrix $\boldsymbol{\Sigma}$.
- The very same strategy can be applied to the MALA algorithm, as well as in more elaborate contexts such as Hamiltonian Monte Carlo.

Rotating the diffusion

- Let $\boldsymbol{\Sigma}$ be the posterior covariance and let $\boldsymbol{\Sigma} = \boldsymbol{A}\boldsymbol{A}^\intercal$ be its **Cholesky decomposition**.
- Let us consider a rotation of the parameters $\tilde{\boldsymbol{\theta}} = \boldsymbol{A}^{-1}\boldsymbol{\theta}$ (reparametrization), implying that the new set of parameters are such that

$$\text{var}(\tilde{\boldsymbol{\theta}} \mid \boldsymbol{X}) = \boldsymbol{A}^{-1}(\boldsymbol{A}\boldsymbol{A}^\intercal)(\boldsymbol{A}^{-1})^\intercal = \boldsymbol{I}_p,$$

  i.e. they are **orthogonal**. If the posterior of $\boldsymbol{\theta}$ is Gaussian, they are also independent.
- Therefore, the MALA based on the rotated Langevin diffusion is

$$\mathrm{d}\tilde{\boldsymbol{\theta}}^{(t)} = \frac{1}{2}\nabla_{\tilde{\boldsymbol{\theta}}} \log \pi(\tilde{\boldsymbol{\theta}}^{(t)} \mid \boldsymbol{X})\mathrm{d}t + \mathrm{d}\boldsymbol{B}^{(t)}.$$

- This leads to a MALA proposal distribution targeting the posterior law of $\tilde{\boldsymbol{\theta}}$, namely

$$(\tilde{\boldsymbol{\theta}}^* \mid \tilde{\boldsymbol{\theta}}) \sim \mathrm{N}_p\left(\tilde{\boldsymbol{\theta}} + \frac{s_p^2}{2}\nabla_{\tilde{\boldsymbol{\theta}}} \log \pi(\tilde{\boldsymbol{\theta}} \mid \boldsymbol{X}),\ s_p^2 \boldsymbol{I}_p\right),$$

  which is expected to perform well due to the orthogonality of $\tilde{\boldsymbol{\theta}}$.

Rotating the diffusion

- By pre-multiplying by $\boldsymbol{A}$ the diffusion for $\tilde{\boldsymbol{\theta}}^{(t)}$, one obtain that

$$\mathrm{d}\boldsymbol{\theta}^{(t)} = \frac{1}{2}\boldsymbol{\Sigma}\,\nabla_{\boldsymbol{\theta}} \log \pi(\boldsymbol{\theta}^{(t)} \mid \boldsymbol{X})\mathrm{d}t + \boldsymbol{A}\,\mathrm{d}\boldsymbol{B}^{(t)},$$

  which by construction targets the posterior law of $\boldsymbol{\theta}$.

- This leads to a **pre-conditioned** MALA, targeting the posterior law of $\boldsymbol{\theta}$, namely

$$(\boldsymbol{\theta}^* \mid \boldsymbol{\theta}) \sim \mathrm{N}_p\left(\boldsymbol{\theta} + \frac{s_p^2}{2}\boldsymbol{\Sigma}\nabla_{\boldsymbol{\theta}} \log \pi(\boldsymbol{\theta} \mid \boldsymbol{X}),\ s_p^2\boldsymbol{\Sigma}\right),$$

  with $s_p^2 = \ell^2/p^{1/3}$.

- In other words, this simple modification of the original MALA proposal is **equivalent** to running the MALA algorithm on the orthogonal parametrization.

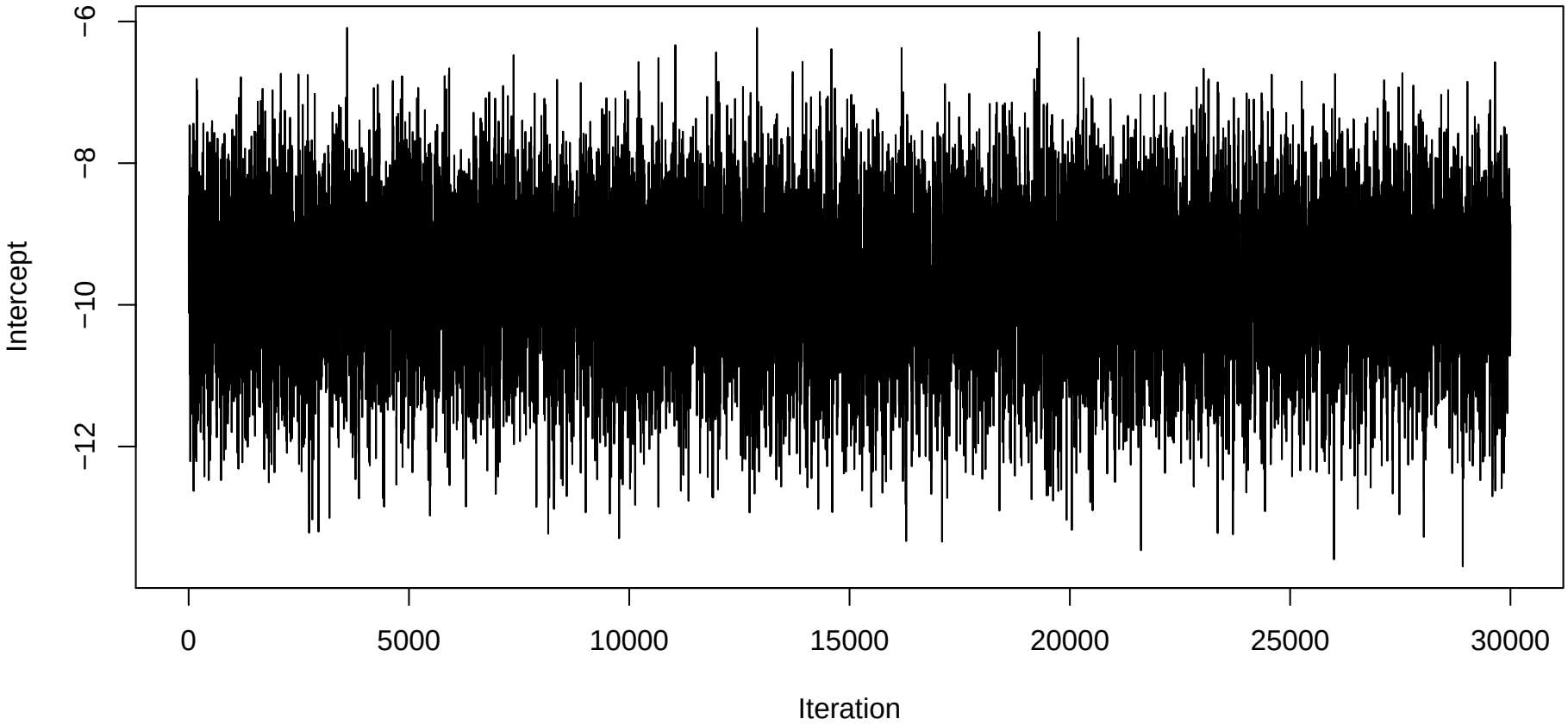
- Obviously $\boldsymbol{\Sigma}$ is unknown, but fast approximations and/or adaptive strategies can be used, following the guidelines outlined in **unit B.1**.

Pre-conditioned MALA algorithm

- After some trial and error, we set $s_p = 1.68$, to get the optimal acceptance rate.
- The pre-conditioned MALA, based on the Laplace approximation, leads to a very high effective sample size.

```
# Covariance matrix is selected via Laplace approximation
fit_logit <- glm(y ~ X - 1, family = binomial(link = "logit"))
S <- vcov(fit_logit)
s_p <- 1.68 # After some trial ad error

# Running the MCMC (R = 30000, burn_in = 5000)
fit_MCMC <- as.mcmc(MALA(R = R, burn_in = burn_in, y, X, s_p, S))

summary(effectiveSize(fit_MCMC)) # Effective sample size (beta)
#    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
#    8583    8762    9196    9063    9312    9409

summary(R / effectiveSize(fit_MCMC)) # Integrated autocorrelation time (beta)
#    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
#   3.189   3.222   3.263   3.314   3.424   3.495

summary(1 - rejectionRate(fit_MCMC)) # Acceptance rate (beta)
#    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
# 0.5686  0.5686  0.5686  0.5686  0.5686  0.5686
```

Pre-conditioned MALA algorithm in practice



General comments

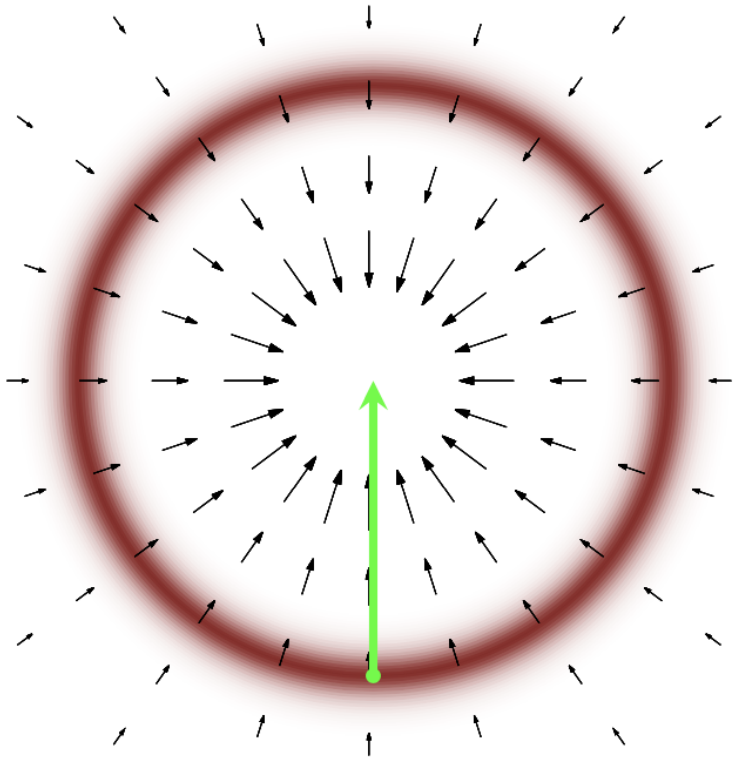
- The take-home message is that pre-conditioning is crucial for both the RWM and the MALA in order to get practically useful samples.
- In high-dimensional problems, when obtaining an estimate for $\boldsymbol{\Sigma}$ could be problematic, it is recommended to use at least a diagonal matrix specification.
- Gradient-based methods such as MALA are more fragile than RWM, meaning that misscalibrations for the proposal of density leads to drastic performance drop.
- These issues motivated recent works on **robust** gradient-based proposals aimed at correcting this behaviour.

Main reference

Livingstone, S. and G. Zanella (2020). The Barker proposal: combining robustness and efficiency in gradient-based MCMC. arXiv:1908.11812.

Hamiltonian Monte Carlo

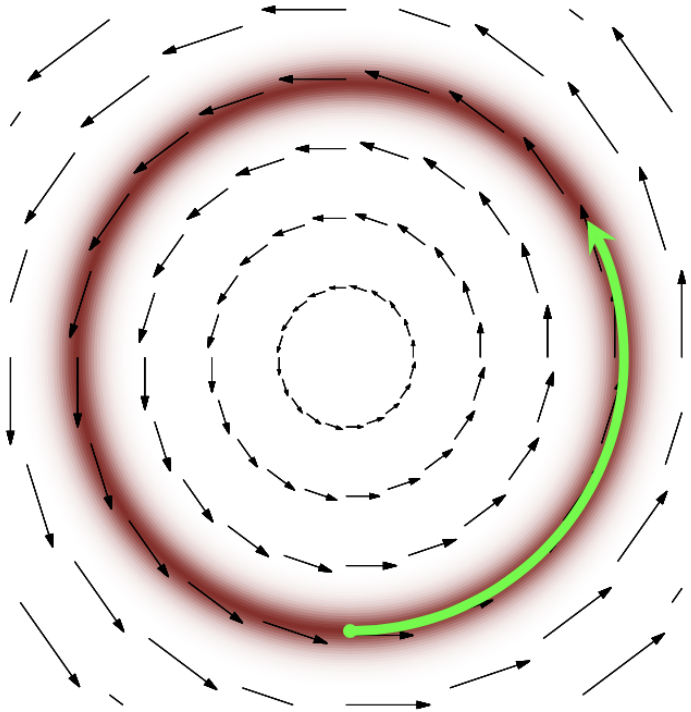
- In complicated inferential problems, pushing the Markov chain **towards the mode**, as in the MALA, may lead to an inefficient exploration of the parameter space.



MALA dynamics. Picture ~~stolen~~ taken from Betancourt (2017).

Hamiltonian Monte Carlo

- Ideally, we would like the Markov Chain to explore the values of a given **level set**, following the complex dynamics that are implied by the **gradient information**.



Hamiltonian dynamics. Picture ~~stolen~~ taken from Betancourt (2017).

Hamiltonian Monte Carlo

- Hamiltonian Monte Carlo (HMC) is essentially a Metropolis-Hastings algorithm with a "smart" proposal distribution based on the **gradient**.
- HMC performs several steps in the parameter space before accepting / rejecting the move, therefore favoring bigger jumps and better mixing.
- The proposed value is far from the previous one, but it remains on a similar level set. This is in contrast with RWM, in which big jumps often leads to values with low density.
- HMC, also known as Hybrid Monte Carlo, has been known for some time in physics, but it seems to have be considered only recently in statistics.

Main reference

Neal, R. M. (2011). MCMC using Hamiltonian dynamics. CRC press.

HMC: basic quantities

- Recall that $\boldsymbol{\theta} \subseteq \mathbb{R}^p$ is the parameter of interests. We aim at sampling from the posterior distribution $\pi(\boldsymbol{\theta} \mid \boldsymbol{X})$.
- In HMC we rely on an **auxiliary** set of parameters $\boldsymbol{\psi} \subseteq \mathbb{R}^p$ independent on $\boldsymbol{\theta}$, so that the joint density of $(\boldsymbol{\theta}, \boldsymbol{\psi})$ is given by

$$\pi(\boldsymbol{\theta}, \boldsymbol{\psi} \mid \boldsymbol{X}) = \pi(\boldsymbol{\theta} \mid \boldsymbol{X})\, \pi(\boldsymbol{\psi}).$$

- We let $\boldsymbol{\psi} \sim \mathrm{N}_p(0, \boldsymbol{M})$ be multivariate Gaussian with zero mean and covariance $\boldsymbol{M}$.
- The term $\mathcal{H}(\boldsymbol{\theta}, \boldsymbol{\psi}) = -\log \pi(\boldsymbol{\theta}, \boldsymbol{\psi} \mid \boldsymbol{X})$ is called the **Hamiltonian**, which equals

$$\mathcal{H}(\boldsymbol{\theta}, \boldsymbol{\psi}) = -\log \pi(\boldsymbol{\theta} \mid \boldsymbol{X}) + \frac{1}{2} \boldsymbol{\psi}^\intercal \boldsymbol{M}^{-1} \boldsymbol{\psi},$$

  up to an irrelevant additive constant not depending on $(\boldsymbol{\theta}, \boldsymbol{\psi})$.
- In HMC we will sample values from the joint distribution $\pi(\boldsymbol{\theta}, \boldsymbol{\psi} \mid \boldsymbol{X})$ and then we will discard the samples for $\boldsymbol{\psi}$.

Hamiltonian dinamics

- The gradient of $\mathcal{H}(\boldsymbol{\theta}, \boldsymbol{\psi})$ with respect to $(\boldsymbol{\theta}, \boldsymbol{\psi})$ admits a physical interpretation as the **time evolution**, with respect to a **fictitious time** t, of an Hamiltonian dynamic system.

- Let us assume that $(\boldsymbol{\theta}(t), \boldsymbol{\psi}(t))$ is a deterministic evolution flowing according to the following set of **differential equations**

$$\frac{\mathrm{d}\boldsymbol{\theta}(t)}{\mathrm{d}t} = \frac{\partial \mathcal{H}(\boldsymbol{\theta}, \boldsymbol{\psi})}{\partial \boldsymbol{\psi}} = \boldsymbol{M}^{-1}\boldsymbol{\psi}(t),$$
$$\frac{\mathrm{d}\boldsymbol{\psi}(t)}{\mathrm{d}t} = -\frac{\partial \mathcal{H}(\boldsymbol{\theta}, \boldsymbol{\psi})}{\partial \boldsymbol{\theta}} = \nabla_{\boldsymbol{\theta}} \log \pi(\boldsymbol{\theta}(t) \mid \boldsymbol{X}).$$

- By assuming that $\mathcal{H}(\boldsymbol{\theta}, \boldsymbol{\psi})$ does not depend on time t, for any $t, s \in \mathbb{R}$ we have that

$$(\boldsymbol{\theta}(t+s), \boldsymbol{\psi}(t+s)) = \mathcal{T}_s\{\boldsymbol{\theta}(t), \boldsymbol{\psi}(t)\},$$

  for some mapping $\mathcal{T}_s$ depending only on s.

Properties of Hamiltonian dynamics

- These differential equations preserve the value of the Hamiltonian, namely

$$\mathcal{H}\{\boldsymbol{\theta}(t), \boldsymbol{\psi}(t)\} = \mathcal{H}\{\boldsymbol{\theta}(t+s), \boldsymbol{\psi}(t+s)\},$$

- Hence, they also preserve the value of the **joint density**

$$\pi(\boldsymbol{\theta}(t) \mid \boldsymbol{X})\, \pi(\boldsymbol{\psi}(t)) = \pi(\boldsymbol{\theta}(t+s) \mid \boldsymbol{X})\, \pi(\boldsymbol{\psi}(t+s)).$$

- **Remark**. Any move according to Hamiltonian dynamics preserve the **level set**.

- The mapping $\mathcal{T}_s$ is also **time-reversible**, which is crucial for showing that MCMC updates that use the dynamics leave the desired distribution invariant.

- Moreover, the mapping **preserves the volume**, a property which significantly simplifies the computations of the MCMC algorithm.

A Gaussian example

- Let us assume that $(\boldsymbol{\theta}, \boldsymbol{\psi}) \sim \mathrm{N}_2(0, I_2)$, so that the Hamiltonian is equal to

$$\mathcal{H}\{\boldsymbol{\theta}(t), \boldsymbol{\psi}(t)\} = \frac{\boldsymbol{\theta}(t)^2}{2} + \frac{\boldsymbol{\psi}(t)^2}{2}.$$

- In this special case, the **differential equations** simplify as follows

$$\frac{\mathrm{d}\boldsymbol{\theta}(t)}{\mathrm{d}t} = \frac{\partial \mathcal{H}(\boldsymbol{\theta}, \boldsymbol{\psi})}{\partial \boldsymbol{\psi}} = \boldsymbol{\psi}(t),$$
$$\frac{\mathrm{d}\boldsymbol{\psi}(t)}{\mathrm{d}t} = -\frac{\partial \mathcal{H}(\boldsymbol{\theta}, \boldsymbol{\psi})}{\partial \boldsymbol{\theta}} = -\boldsymbol{\theta}(t).$$

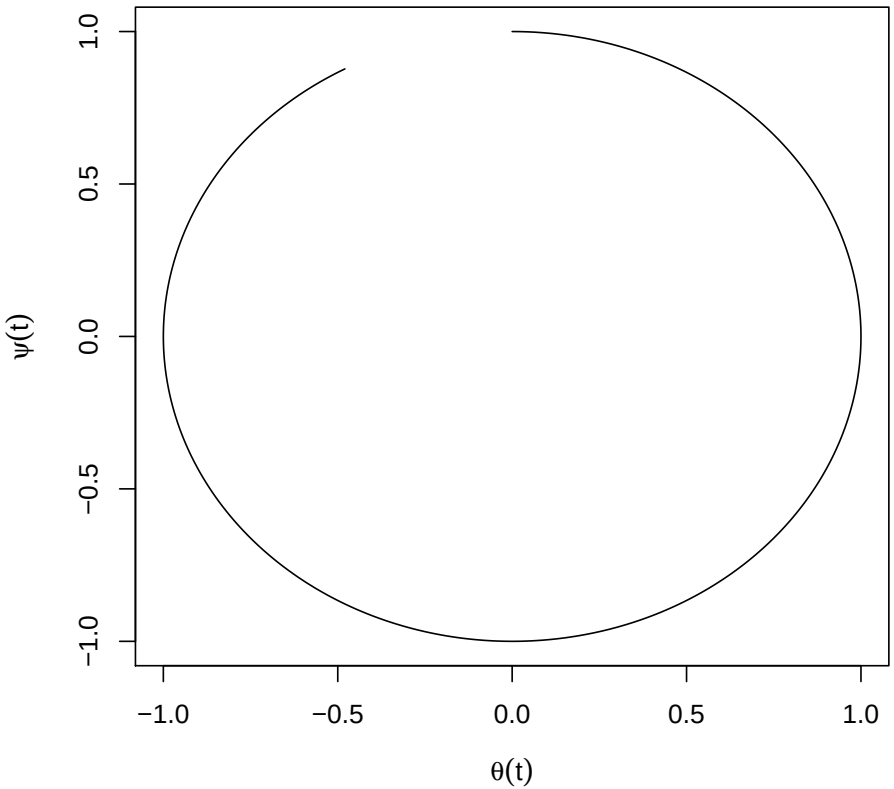
- It is easy to show that the solution are in the following form

$$\boldsymbol{\theta}(t) = \rho \cos(\alpha + t), \qquad \boldsymbol{\psi}(t) = -\rho \sin(\alpha + t),$$

  for some constants ρ and α.

- Hence, the mapping $\mathcal{T}_s$ is a rotation by s radians clockwise around the origin.

A Gaussian example



- Trajectory of the dynamics with $\rho = 1$, $\alpha = -\pi/2$ and for values of $t \in [0, 2\pi - 1/2]$.

The ideal HMC algorithm

- If the solution of the Hamiltonian dynamics were available in **closed form**, the HMC would proceed as follows.
- Recall that we aim at sampling values from the joint distribution $\pi(\boldsymbol{\theta}, \boldsymbol{\psi} \mid \boldsymbol{X})$ using a algorithm. Let $\boldsymbol{\theta}^{(r)}$ be the current value of the chain.
- At each step of the chain, draw a new value $\boldsymbol{\psi}$ sampling from a multivariate Gaussian distribution $\mathrm{N}_p(0, \boldsymbol{M})$; this identifies a **new level set**.
- Obtain the proposed values $(\boldsymbol{\theta}^*, \boldsymbol{\psi}^*)$ remaining on the given level set, by applying the mapping

$$(\boldsymbol{\theta}^*, \boldsymbol{\psi}^*) = \mathcal{T}_s\{\boldsymbol{\theta}^{(r)}, \boldsymbol{\psi}\},$$

  for a certain value of time s, which is basically a tuning parameter.
- Thanks the properties of Hamiltonian dynamics, the acceptance probability is always 1, therefore the next value of the chain coincides with the proposal $\boldsymbol{\theta}^{(r+1)} \longleftarrow \boldsymbol{\theta}^*$.

Approximating Hamiltonian dynamics

- Unfortunately, most of the times the Hamiltonian differential equations do not admit a closed-form solution, thus **approximations** are required.
- To maintain the main properties of the ideal HMC, we look for discretized Hamiltonian dynamics **preserving the volume** and also being **time-reversible**.
- Among several methods that aims at solving this issue, we focus on the **leapfrog** method which unfortunately does not keep the Hamiltonian exactly constant over time.
- The leapfrog method is essentially a variation and more reliable version of the natural Euler's discretization method.
- **Remark**. The leapfrog method retains most of the properties of the ideal HMC, thus making it extremely appealing for sampling purposes.

The leapfrog method

- We aim at approximating the mapping $\mathcal{T}_\epsilon$ for some small $\epsilon > 0$.
- For any given time t and set of values $\boldsymbol{\theta}(t), \boldsymbol{\psi}(t)$, a small step ahead in time of size ϵ in the Hamiltonian dynamics can be obtained using the leapfrog method.
- We make a first update on the **auxiliary variables** of size $\epsilon/2$, namely we get

$$\boldsymbol{\psi}(t + \epsilon/2) = \boldsymbol{\psi}(t) + \frac{\epsilon}{2} \nabla_{\boldsymbol{\theta}} \log \pi(\boldsymbol{\theta}(t) \mid \boldsymbol{X}).$$

- Secondly, we make a full step for the **variables of interest**, namely

$$\boldsymbol{\theta}(t + \epsilon) = \boldsymbol{\theta}(t) + \epsilon \, \boldsymbol{M}^{-1} \boldsymbol{\psi}(t + \epsilon/2).$$

- Finally, we update again the **auxiliary variable** of size $\epsilon/2$, so that

$$\boldsymbol{\psi}(t + \epsilon) = \boldsymbol{\psi}(t + \epsilon/2) + \epsilon \, \nabla_{\boldsymbol{\theta}} \log \pi(\boldsymbol{\theta}(t + \epsilon) \mid \boldsymbol{X}).$$

A single HMC step

- A single step of HMC proceeds as follows. Let $\boldsymbol{\theta}^{(r)}$ be the current value of the chain.
- Draw a new value $\boldsymbol{\psi}$ sampling from a multivariate Gaussian distribution $\mathrm{N}_p(0, \boldsymbol{M})$.
- Obtain the proposed values $(\boldsymbol{\theta}^*, \boldsymbol{\psi}^*)$ by applying L times the **leapfrog method** with step-size ϵ and starting time $t = 0$, thus aiming at approximating the ideal dynamics

$$(\boldsymbol{\theta}^*, \boldsymbol{\psi}^*) \approx \mathcal{T}_s\{\boldsymbol{\theta}^{(r)}, \boldsymbol{\psi}\},$$

  for a certain time value $s = L\,\epsilon$.

- Due to the symmetricity of the proposal, **accept or reject** the proposed value with a probability depending only on the Hamiltonian, namely

$$\min[1, \exp\{-\mathcal{H}(\boldsymbol{\theta}^*, \boldsymbol{\psi}^*) + \mathcal{H}(\boldsymbol{\theta}^{(r)}, \boldsymbol{\psi}^{(r)})\}],$$

  which is usually very close to 1 as the Hamiltonian is kept **approximately constant** by the leapfrog method.

Connection with the MALA algorithm

- There is a **strong connection** between MALA and HMC, even though these methods rely on very different notions.
- Indeed, it can be shown HMC using a single $L = 1$ leapfrog step coincides with the pre-conditioned MALA algorithm.
- More precisely, at each step we propose from

$$(\boldsymbol{\theta}^* \mid \boldsymbol{\theta}) \sim \mathrm{N}_p \left(\boldsymbol{\theta} + \frac{\epsilon^2}{2} \boldsymbol{M}^{-1} \nabla_{\boldsymbol{\theta}} \log \pi(\boldsymbol{\theta} \mid \boldsymbol{X}),\ \epsilon^2 \boldsymbol{M}^{-1} \right),$$

  and the acceptance probability coincides with that of MALA.
- This connections highlights that the covariance matrix of the Gaussian auxiliary variables is of great **practical importance**.
- In practice, it is advised to set $\boldsymbol{M} = \boldsymbol{\Sigma}^{-1}$, where $\boldsymbol{\Sigma}$ represents an estimate for the posterior covariance; see Neal (2010) for further details and a **deeper perspective**.

How to choose ϵ and L?

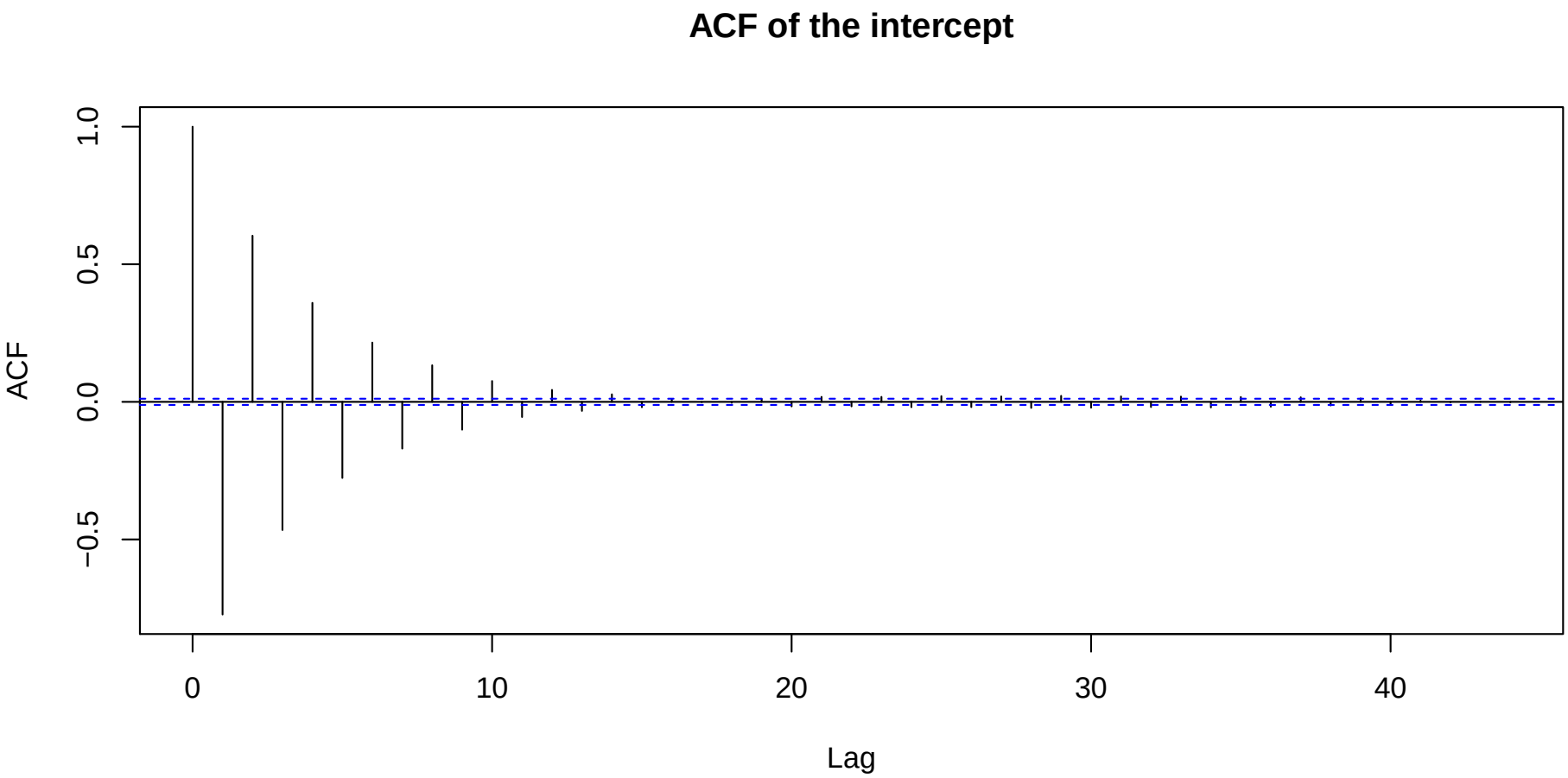
- Beside the covariance matrix $\boldsymbol{M}$, there are other 2 tuning parameters that must be chosen: the stepsize ϵ and number of leapfrog steps L.
- The **trajectory length** ϵL is often set equal to some constant, say $\epsilon L = 1$ or selected by trial and error.
- Small values for the **step-size** ϵ increase the goodness of the leapfrog approximation but require larger values for L and leads to a higher computational costs.
- Large values for the **step-size** ϵ could lead to catastrophic results, as the approximated trajectory could diverge from the ideal dynamics.
- The NO-U-TURN algorithm implemented in **Stan** automatically selects L so that the trajectory completes a "loop".
- However, the NO-U-TURN requires a somewhat complex procedure that preserve the reversibility of the chain.

HMC algorithm in practice

- After some trial and error, we set $\epsilon = 0.1$ and $L = 10$.
- We again relied on the Laplace approximation for $\hat{\boldsymbol{\Sigma}} = \boldsymbol{M}^{-1}$, which leads to an extremely high effective sample size.

```
epsilon <- 0.25 # Stepsize - After some trial ad error
L <- 10 # Number of leapfrog steps

# Covariance matrix is selected via laplace approximation
fit_logit <- glm(y ~ X - 1, family = binomial(link = "logit"))
S <- vcov(fit_logit)
# Running the MCMC
fit_MCMC <- as.mcmc(HMC(R = R, burn_in = burn_in, y, X, epsilon, S, L))

# Running the MCMC (R = 30000, burn_in = 5000)
summary(effectiveSize(fit_MCMC)) # Effective sample size
#   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
# 215765  222946  226610  225565  228360  233334
summary(R / effectiveSize(fit_MCMC)) # Integrated autocorrelation time
#   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
# 0.1286  0.1314  0.1324  0.1331  0.1346  0.1390
summary(1 - rejectionRate(fit_MCMC)) # Acceptance rate
#   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
# 0.9892  0.9892  0.9892  0.9892  0.9892  0.9892
```

HMC algorithm in practice



- The ACF has an alternate sign behaviour, implying that the integrated autocorrelation $\tau_g < 1$ is smaller than 1, leading to an efficiency which is higher than iid sampling.

Summary of the results

- The following table compare the **average** results. Here ESS represents the estimated and average **effective sample size**.
- A suitably tuned HMC can be extremely effective and it is the clear winner in this case.
- The implementation could be even improved by writing it in **Rcpp**!
- Refer to the link: `https://tommasorigon.github.io/CompStat/exe/un_B2.html`

	Seconds	ESS	ESS / Sec.	Acceptance rate
MH Laplace in **Rcpp**	0.61	1165.76	1897.52	0.27
MALA	3.43	44.32	12.91	0.56
Pre-conditioned MALA	3.85	9063.32	2351.34	0.57
HMC	15.32	225565.17	14724.08	0.99
Stan	78.85	29864.59	378.77	1