Stochastic Simulation

Autumn Semester 2024

Prof. Fabio Nobile *Assistant: Matteo Raviola*

Lab 11 – 28 November 2024

Markov Chain Monte Carlo

Exercise 1

In many applications of interest, it is not uncommon to encounter the need for sampling from a multi-modal distribution f. The theory developed so far can be directly applicable to these types of distributions. However, in practice, sampling from these distributions using MCMC can be computationally challenging, as we will investigate in this problem. Throughout this exercise, we will consider the bi-modal distribution

$$f(x;\gamma,x_0)=\frac{e^{-\gamma(x^2-x_0)^2}}{Z},\quad \gamma>0, \tag{1}$$

where Z is some normalizing constant. Depending on the values of γ and x_0, designing a sampling strategy to properly sample from (1) can become challenging. Intuitively, if both peaks are too far apart, using a random walk Metropolis (RWM) might not work, as it is possible for the sampler to get stuck on one of the peaks if the *step-size* is too small. Conversely, a RWM with very large *steps* might tend to reject quite often, thus rendering the whole sampling procedure inefficient. We begin by verifying this. Implement the RWM algorithm using as proposal distribution $q(x,y)=\mathcal{N}(x,\sigma^2)$ and target distribution $f(x;\gamma,x_0)$ for $\gamma=1$, $x_0=1,4,9,25$ and different choices of σ. Discuss the quality of your samples by analyzing the trace-plots (one realization of the chain), autocorrelation functions and histograms of the chains obtained.

Solution

An exemplary implementation of this problem is given at the end of the exercise. We implement the sampler for different values of x_0 and σ and show the results in Figure 1. As we can see from figure 1, one both x_0 and σ are of a similar magnitude (top row), then the sampler is able to correctly explore the distribution. This is contrary to what happens in the bottom row, where x_0 is much larger than σ and as such, the sampler tends to get stuck at one of the peaks of the distribution. This in turn can be fixed by increasing the size of σ, as shown in figure 2. Notice here that for σ around 10, the chain has better mixing and a more rapidly decaying auto-correlation plot, albeit the acceptance rate is 0.017, which is quite small. In general, it is difficult to choose an appropriate σ to sample from this type of multi-modal distributions when using random walk Metropolis. Some recent advances to overcome this difficulty are the so-called Hamiltonian Monte Carlo, the delayed-rejection Metropolis-Hastings, and the parallel-tempering algorithm. We refer the interested reader to [2]

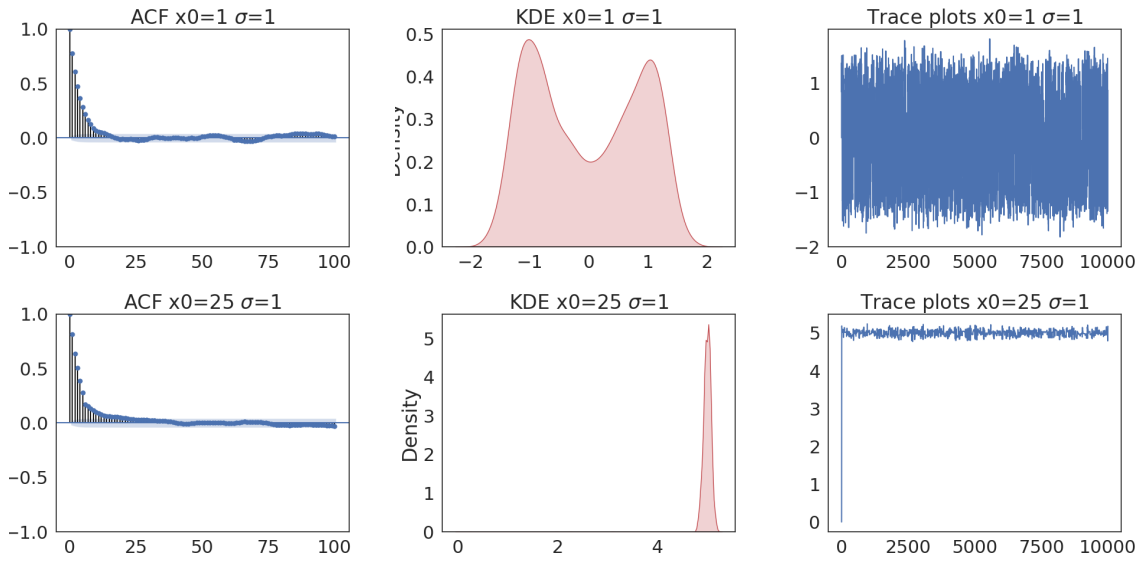

Figure 1: Results for $\sigma = 1$ and $x_0 = 1$ (top) and $x_0 = 25$ (bottom). The blue-shaded part on the auto-correlation plot joins the boundaries of an approximate 95% interval for the individual correlations.

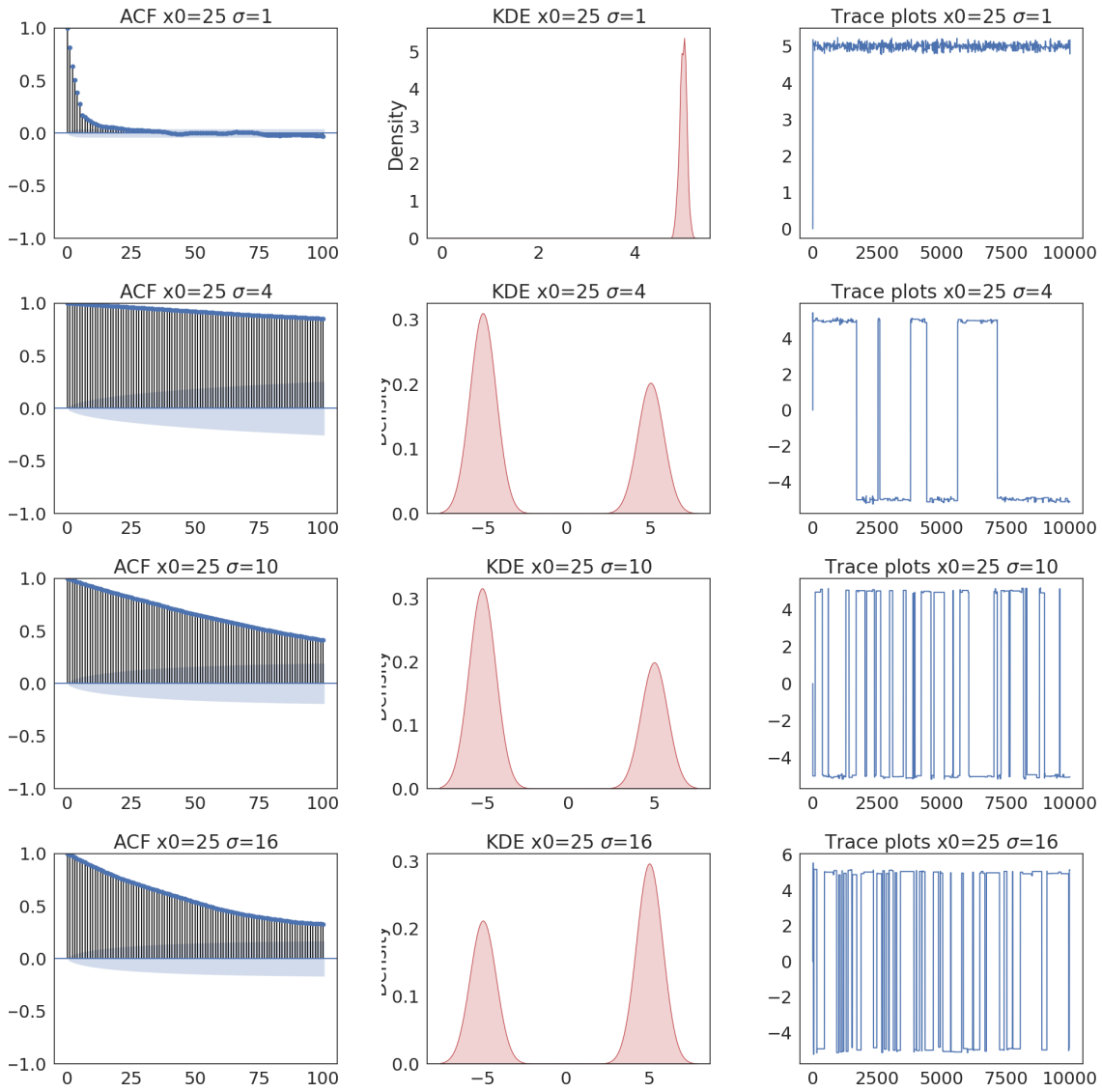

Figure 2: Results for $x_0 = 25$ and different values of σ, from top to bottom $\sigma = 1, 4, 10, 16$. The blue-shaded part on the autocorrelation plot joins the boundaries of an approximate 95% interval for the individual correlations.

Python code:

```
import numpy as np
import matplotlib.pyplot as plt
import statsmodels.graphics.tsaplots as sm
import seaborn as sns; sns.set(color_codes=True)# Defines the pdf
sns.set(font_scale=2) #fontsize in plots
sns.set_style("white")

def f(x,x0,gamma=1):
    return np.exp(-gamma*(x**2-x0)**2 )
```

```
#defines the metropolis-hastings routine
def MH(sigma,x0):
    N=10000
    X=np.zeros(N)
    X[0]=0;
    px=f(X[0],x0)

    for i in range(N-1):
        y=X[i]+sigma*np.random.standard_normal(1)
        py=f(y,x0)
        px=f(X[i],x0)
        if py/px > np.random.random(1):
            X[i+1]=y
        else:
            X[i+1]=X[i]

    plt.plot(X)
    plt.gca().set_rasterized(True)
    plt.title('Trace plots x0='+str(x0)+' $\sigma$='+str(sigma))
    plt.savefig('../figures/Trace_plots_x0='+str(x0)+'sigma'+str(sigma)+'.png')
    plt.show()

    sm.plot_acf(X,lags=100)
    plt.gca().set_rasterized(True)
    plt.title('ACF x0='+str(x0)+' $\sigma$='+str(sigma))
    plt.savefig('../figures/ACF_x0='+str(x0)+'sigma'+str(sigma)+'.png')
    plt.show()

    sns.kdeplot(X, shade=True, color="r")
    plt.gca().set_rasterized(True)
    plt.title('KDE x0='+str(x0)+' $\sigma$='+str(sigma))
    plt.savefig('../figures/KDE_x0='+str(x0)+'sigma'+str(sigma)+'.png')
    plt.show()

    return X

## Runs experiments
xx=np.array([1,25])
ss=np.array([1,4,10,16] )

for i in range(len(xx)):
    for j in range(len(ss)):
        MH(ss[j],xx[i])
```

Exercise 2

Ideally, we would like to obtain (approximately) i.i.d samples from a target distribution f using Markov Chain Monte Carlo (MCMC) algorithms. One practical way of doing so is via *sub-sampling* (also called *batch sampling*), which is implemented to reduce or eliminate correlation between the successive values in the Markov chain. That is, instead of considering the entire chain $\{X_n \colon n \geq 0\}$, say, this technique sub-samples the chain with a batch size $k > 1$, so that only the values $\{X_{kn} \colon n \geq 0\}$ are considered. If the covariance $\mathrm{Cov}_f(X_0, X_n)$ vanishes as $n \to \infty$, then the idea of sub-sampling is quite natural since X_{kn} and $X_{k(n+1)}$ can be considered to be approximately independent for k sufficiently big; estimating such a k may be difficult in practice though. While sub-sampling provides a way of generating (approx.) i.i.d. samples from f and may thus be useful assessing the convergence of a MCMC method, it necessarily leads to an efficiency loss. Let $\{X_n \in \mathbb{R}^d \colon n \geq 0\}$ be a Markov chain with a unique stationary distribution f, and $X_0 \sim f$ (i.e., the chain is at equilibrium). Take $\phi \colon \mathbb{R}^d \to \mathbb{R}$ such that $\mathbb{E}_f\big(|\phi|^2\big) < \infty$ and consider two estimators for $\mu = \mathbb{E}_f(\phi)$, namely one that uses the entire Markov chain ($\hat{\mu}$) and one based on sub-sampling ($\hat{\mu}_k$) using only every k-th value:

$$\hat{\mu} = \frac{1}{Nk} \sum_{n=1}^{Nk} \phi(X_n) \;, \quad \text{and} \quad \hat{\mu}_k = \frac{1}{N} \sum_{n=1}^{N} \phi(X_{nk}) \;.$$

Show that the variance of $\hat{\mu}$ satisfies $\mathrm{Var}_f(\hat{\mu}) \leq \mathrm{Var}_f(\hat{\mu}_k)$ for every $k > 1$.

Solution

Let $k > 1$. Then define $\hat{\mu}_k^{(0)}, \hat{\mu}_k^{(1)}, \ldots, \hat{\mu}_k^{(k-1)}$ as the shifted versions of $\hat{\mu}_k$, in the sense that:

$$\hat{\mu}_k^{(j)} = \frac{1}{N} \sum_{n=1}^{N} \phi(X_{nk-j}) \;, \quad j = 0, 1, \ldots, k-1 \;.$$

Notice that the estimator $\hat{\mu}$ can then be written as:

$$\hat{\mu} = \frac{1}{k} \sum_{j=0}^{k-1} \hat{\mu}_k^{(j)} \;,$$

so that the variance of $\hat{\mu}$ satisfies:

$$\begin{aligned} \mathrm{Var}_f(\hat{\mu}) = \mathrm{Var}_f\left(\frac{1}{k} \sum_{j=0}^{k-1} \hat{\mu}_k^{(j)} \right) &= \frac{\mathrm{Var}_f\big(\hat{\mu}_k^{(0)}\big)}{k} + \sum_{i \neq j} \frac{\mathrm{Cov}_f\big(\hat{\mu}_k^{(i)}, \hat{\mu}_k^{(j)}\big)}{k^2} \\ &\leq \frac{\mathrm{Var}_f\big(\hat{\mu}_k^{(0)}\big)}{k} + \sum_{i \neq j} \frac{\sqrt{\mathrm{Var}_f\big(\hat{\mu}_k^{(i)}\big)\, \mathrm{Var}_f\big(\hat{\mu}_k^{(j)}\big)}}{k^2} \end{aligned}$$

in view of the Cauchy–Schwarz inequality and the stationarity. The claim then follows form the stationarity of the Markov chain again, indeed

$$\begin{aligned} \mathrm{Var}_f(\hat{\mu}) &\leq \frac{\mathrm{Var}_f\big(\hat{\mu}_k^{(0)}\big)}{k} + \sum_{i \neq j} \frac{\sqrt{\mathrm{Var}_f\big(\hat{\mu}_k^{(i)}\big)\, \mathrm{Var}_f\big(\hat{\mu}_k^{(j)}\big)}}{k^2} \\ &= \frac{\mathrm{Var}_f\big(\hat{\mu}_k^{(0)}\big)}{k} + \frac{k-1}{k}\, \mathrm{Var}_f\big(\hat{\mu}_k^{(0)}\big) = \mathrm{Var}_f\big(\hat{\mu}_k\big) \;. \end{aligned}$$

Exercise 3

Let $X \subset \mathbb{R}^d$ and $P_i : X \times \mathcal{B}(X) \to [0,1]$, $i = 1 \ldots, m$ be a Markov transition kernels on X with $\mathcal{B}(X)$ the associated $\sigma-$algebra.

(a) Given $a_1, \ldots, a_m \in \mathbb{R}^+$, such that $\sum_{i=1}^m a_i = 1$, show that $P(x, A) = \sum_{i=1}^m a_i P_i(x, A)$ is a Markov kernel.

(b) Suppose that a measure $\pi : \mathcal{B} \to [0,1]$ is invariant for each kernel P_i. Show that it is also invariant for $P = \sum_{i=1}^m a_i P_i$, where $a_1, \ldots, a_m \in \mathbb{R}^+$, such that $\sum_{i=1}^m a_i = 1$. If each P_i is reversible, is P reversible?

(c) Under the same assumptions for point (b), define the Markov operator $\mathcal{P}_i$ associated to P_i (i.e., $\pi\mathcal{P}_i = \int P(x, \cdot) d\pi(x)$). Then, show that π is also invariant for $\mathcal{P} = \mathcal{P}_{i_1} \circ \cdots \circ \mathcal{P}_{i_k}$, for any choice of $i_1, \ldots, i_k$. If each P_i is reversible, for which choice of $i_1, \ldots, i_k$ is $\mathcal{P}$ reversible?

Solution

(a) We need to verify that

1. $\forall A \in \mathcal{B}$, $P(\cdot, A)$ is measurable.
2. $\forall x \in X$, $P(x, \cdot)$ is a probability measure on $(X, \mathcal{B}(X))$.

For the first we have that $\forall A \in \mathcal{B}(X)$, $P(\cdot, A) = \sum a_i P_i(\cdot, A)$ which is measurable as a linear combination of measurable functions. Moreover, $\forall x \in X$, $P(x, \cdot) = \sum_i a_i P_i(x, \cdot)$ is a probability measure on $(X, \mathcal{B}(X))$ since it is a convex combination of probability measures. Notice, in particular, that $P(x, X) = \sum_i a_i P_i(x, X) = \sum_i a_i = 1$.

(b) Each kernel P_i preserves π, that is

$$\int_X P_i(x, A)\pi(dx) = \pi(A), \quad A \in \mathcal{B}(X), \quad \forall i = 1, \ldots, m. \tag{2}$$

We have $\forall A \in \mathcal{B}(X)$ $\int_X P(x, A)\pi(dx) = \sum_i a_i \int_X P_i(x, A)\pi(dx) = \sum_i a_i \pi(A) = \pi(A)$, hence P also preserves π.

(c) Note that, since $\pi\mathcal{P}_i = \pi$ by assumption, we have $\pi\mathcal{P} = \pi\mathcal{P}_{i_1} \circ \cdots \circ \mathcal{P}_{i_k} = \pi$. Furthermore, if each $\mathcal{P}_i$ is reversible, then $\mathcal{P} = \mathcal{P}_{i_1} \circ \cdots \circ \mathcal{P}_{i_{n-1}} \circ \mathcal{P}_{i_n} \circ \mathcal{P}_{i_{n-1}} \circ \cdots \circ \mathcal{P}_{i_1}$ is reversible.

Exercise 4

At every iteration of the general Metropolis–Hastings algorithm, a new candidate state $\boldsymbol{Y}_{n+1}$ is proposed by sampling $\boldsymbol{Y}_{n+1} \sim q(\boldsymbol{X}_n, \cdot)$, given the current state $\boldsymbol{X}_n$. Here, $q(\boldsymbol{x}, \boldsymbol{y})$ is the so-called proposal density. Consider now the case where the proposal does not depend on the current state, that is $q(\boldsymbol{x}, \boldsymbol{y}) \equiv q(\boldsymbol{y})$, so that the proposed candidate is $\boldsymbol{Y}_{n+1} \sim q$. This particular Markov Chain Monte Carlo (MCMC) variant is sometimes called *independent Metropolis–Hastings algorithm* with fixed proposal (or simply *independence sampler*). Let's denote the target density by f. As such, this MCMC variant appears very similar to the Accept–Reject method for sampling from f (cf. Lab 02).

1. Suppose there exists a positive constant C such that $f(\boldsymbol{x}) \leq Cq(\boldsymbol{x})$ for any $\boldsymbol{x} \in \operatorname{supp}(f) = \{\boldsymbol{x} \in \mathbb{R}^d \colon f(\boldsymbol{x}) > 0\}$. Show that the expected acceptance probability of the independent Metropolis–Hastings algorithm is *at least* $\frac{1}{C}$ whenever the chain is stationary. How does this compare to the expected acceptance probability of an Accept–Reject method?

2. Let us compare the independent Metropolis–Hastings algorithm and the Accept–Reject method in some more detail by an example. Specifically, the goal is to sample from a Gamma distribution with shape parameter α and scale parameter β, denoted by $\text{Gamma}(\alpha, \beta)$, so that the target PDF reads $f(x) \equiv f(x; \alpha, \beta) = \beta^\alpha x^{\alpha-1} e^{-\beta x}/\Gamma(\alpha)\mathbb{I}_{\{x \geq 0\}}$, where $\Gamma(\cdot)$ denotes the Gamma function.

   (a) Implement the Accept–Reject method to sample from $\text{Gamma}(\alpha, 1)$ for $\alpha > 1$, using the PDF of the $\text{Gamma}(a, b)$ distribution with $a = [\alpha]$ as auxiliary density (here $[\alpha]$ denotes the integer part of α).[1] Show that $b = [\alpha]/\alpha$ is the optimal choice for b.

   (b) Use your Accept–Reject method to generate m random numbers $X_1, \ldots, X_m$ with each $X_i \sim \text{Gamma}(\alpha, 1)$, when using $n = 5000$ random variables $Y_1, \ldots, Y_n$ from the auxiliary $\text{Gamma}([\alpha], [\alpha]/\alpha)$ distribution. Notice that m is a random variable, which is smaller than n due to rejections. Perform the simulations for $\alpha = 4.85$.

   (c) Implement the independent Metropolis–Hastings algorithm using as proposal q the PDF of the $\text{Gamma}([\alpha], [\alpha]/\alpha)$ distribution.

   (d) Use the same sample $Y_1, \ldots, Y_n$ used within the Accept–Reject method, now in the corresponding Metropolis–Hastings algorithm to generate $n = 5000$ realizations of the target distribution $\text{Gamma}(\alpha, 1)$ with $\alpha = 4.85$.

   (e) Compare both methods with respect to:

      i. their acceptance rates,

      ii. their estimates for the mean of the $\text{Gamma}(4.85, 1)$ distribution, which is 4.85,

      iii. the correctness of the target distribution,

   Discuss your results.

[1] *Hint:* *Recall that* $\sum_{k=1}^K \xi_k \sim \text{Gamma}(K, \beta)$ *for* $K \in \mathbb{N}$, *if* $\xi_k \overset{\text{i.i.d.}}{\sim} \text{Gamma}(1, \beta) \equiv \text{Exp}(\beta)$.

Solution

1. Let $C > 0$ such that $f(\boldsymbol{x}) \leq Cq(\boldsymbol{x})$ for any $\boldsymbol{x} \in \text{supp}(f)$. Suppose that the chain is stationary. Then the expected acceptance probability is

$$\begin{aligned}
\mathbb{E}\left(\min\left\{\frac{f(\boldsymbol{Y}_{n+1})q(\boldsymbol{X}_n)}{f(\boldsymbol{X}_n)q(\boldsymbol{Y}_{n+1})}, 1\right\}\right) &= \int \mathbb{I}_{\left\{\frac{f(\boldsymbol{y})q(\boldsymbol{x})}{q(\boldsymbol{y})f(\boldsymbol{x})} \geq 1\right\}} f(\boldsymbol{x})q(\boldsymbol{y})\, d\boldsymbol{x}\, d\boldsymbol{y} \\
&\quad + \int \frac{f(\boldsymbol{y})q(\boldsymbol{x})}{q(\boldsymbol{y})f(\boldsymbol{x})} \mathbb{I}_{\left\{\frac{f(\boldsymbol{y})q(\boldsymbol{x})}{q(\boldsymbol{y})f(\boldsymbol{x})} < 1\right\}} f(\boldsymbol{x})q(\boldsymbol{y})\, d\boldsymbol{x}\, d\boldsymbol{y} \\
&= 2\int \mathbb{I}_{\left\{\frac{f(\boldsymbol{y})q(\boldsymbol{x})}{q(\boldsymbol{y})f(\boldsymbol{x})} \geq 1\right\}} f(\boldsymbol{x})q(\boldsymbol{y})\, d\boldsymbol{x}\, d\boldsymbol{y} \\
&\geq 2\int \mathbb{I}_{\left\{\frac{f(\boldsymbol{y})}{q(\boldsymbol{y})} \geq \frac{f(\boldsymbol{x})}{q(\boldsymbol{x})}\right\}} f(\boldsymbol{x})\frac{f(\boldsymbol{y})}{C}\, d\boldsymbol{x}\, d\boldsymbol{y} \\
&= \frac{2}{C}\mathbb{P}\left(\frac{f(\boldsymbol{X}_1)}{q(\boldsymbol{X}_1)} \geq \frac{f(\boldsymbol{X}_2)}{q(\boldsymbol{X}_2)}\right) = \frac{1}{C}\,,
\end{aligned}$$

   where the last equality follows form the fact that $\boldsymbol{X}_1$ and $\boldsymbol{X}_2$ are independent and both distributed according to f by assumption. In contrast, the average acceptance probability for an Accept–Reject method is always equal to $1/C$.

2. Proving that $b = [\alpha]/\alpha$ is the optimal choice for b follows from straightforward calculations. In fact, the ratio $f(x;\alpha,1)/f(x;a,b)$ is

$$\frac{f(x;\alpha,1)}{f(x;a,b)} = b^{-a}x^{\alpha-a}e^{-(1-b)x}\frac{\Gamma(a)}{\Gamma(\alpha)}\,,$$

   for any $x \geq 0$, which yields the bound $C = b^{-a}\left(\frac{\alpha-a}{(1-b)e}\right)^{\alpha-a}$ for $b < 1$. The optimal choice for b then follows by optimization. A possible `Python` implementation for this exercise is shown at the end of this Section. The code produced, for example the following output regarding the acceptance rate:

```
----------------------------------------
AR acceptance rate 0.9044862518089725
MH acceptance rate 0.9352
----------------------------------------
AR mean 4.814883929306847
MH mean 4.830617617460223
----------------------------------------
```

   confirming the theoretical result proved above. Furthermore, the code also produced the plots shown in Figures 3 and 4.

   These plots show that both methods produce accurate approximations to the mean and the PDF. Moreover, we would expect to see that the sample generated by the MH algorithm contains more correlations, this is because the the AR method should, in fact, produce independent samples, while the MH is a Markov chain. This, however, is difficult to appreciate in the autocorrelation plots, as the samples obtained using MH de-correlate quite rapidly.

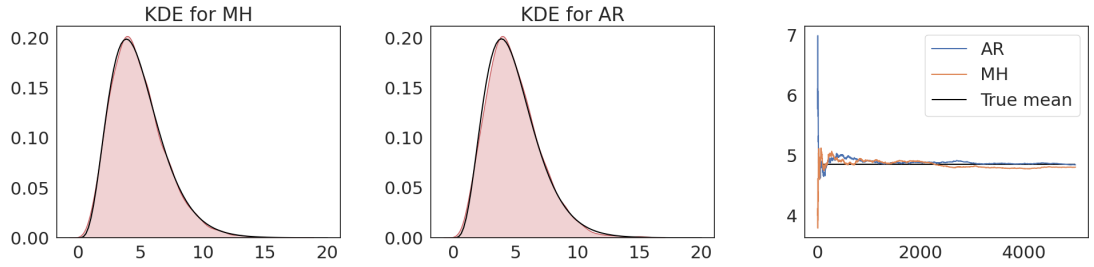

Figure 3: (Left) Kernel density estimate (KDE) Metropolis Hastings. (Middle) KDE accept-reject method. Both shown on top of true density. (Right) Ergodic estimator of the mean for both methods.

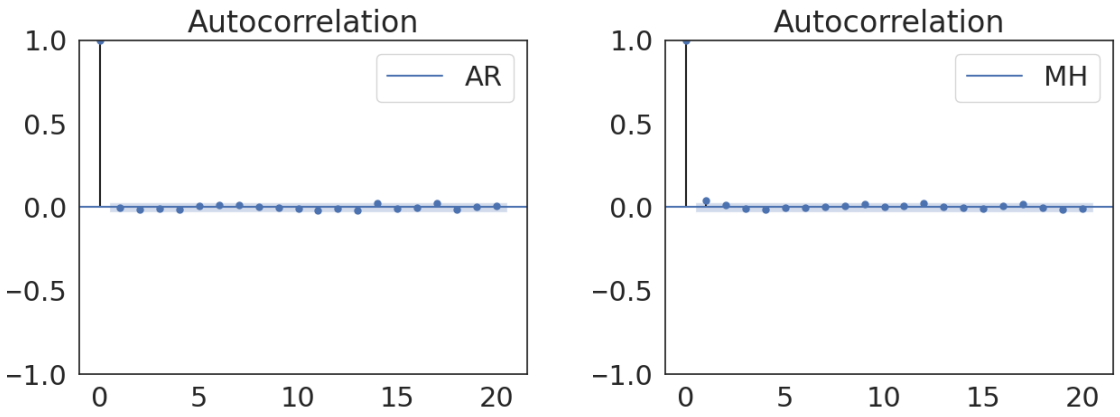

Figure 4: Autocorrelation plot of AR (left) and MH (right). The blue band represents a 95% confidence interval.

Python code:

```
import numpy as np
import scipy.special as sps
import matplotlib.pyplot as plt
from pandas import Series
import statsmodels.graphics.tsaplots as sm
import seaborn as sns
sns.set(color_codes=True)# Defines the pdf
sns.set(font_scale=2) #fontsize in plots
sns.set_style("white")


def f(x,a,b):
    return b**a*x**(a-1)*np.exp(-b*x)/sps.gamma(a)*1*(x>=0)

def Cfunc(alpha, a, b):
    return ((alpha - a)/(1-b))**(alpha-a) * np.exp(a-alpha) * sps.gamma(a)/sps.gamma(alpha)/b

#### Some parameters
alpha=4.85
a = np.floor(alpha)
b = np.floor(alpha)/alpha
C = Cfunc(alpha,a,b)

#### Does the accept reject part
n = 5000
Aar = 0
Xar = np.zeros(n)
i = 0
p = 0
while i<n:
    xi = np.random.gamma(shape = a, scale = 1/b, size=1)
    ratio = f(xi,alpha,1)/(f(xi,a,b)*C)
    p+=1
    if np.random.random(1) < ratio:
        Xar[i]=xi
        i+=1
Aar=n/p # acceptance probability

### Does the Metropolis Hastings part
Xmh = np.zeros(n)
Xmh[0] = 4
Amh=0
for i in range(n-1):
    #samples proposal
    y = np.random.gamma(shape = a, scale = 1/b, size=1)
```

```
    #computes MH ratio
    py=f(y,alpha,1)
    px=f(Xmh[i],alpha,1)
    qyx=f(Xmh[i],a,b)
    qxy=f(y,a,b)

    ratio=(py*qyx )/(px*qxy)
    if np.random.random(1) < ratio:
        Xmh[i+1]=y
        Amh+=1
    else:
        Xmh[i+1]=Xmh[i]
Amh=Amh/n

print('----------------------------------------')
print('AR acceptance rate '+str(Aar))
print('MH acceptance rate '+str(Amh))
print('----------------------------------------')
print('AR mean '+str(np.mean(Xar)))
print('MH mean '+str(np.mean(Xmh)))
print('----------------------------------------')


xx=np.linspace(0,20,1000)
plt.plot(xx,C*f(xx,a,b))
plt.plot(xx,f(xx,alpha,1))
plt.legend(["Cg(x)","f(x)"] )
plt.savefig("../figures/densities.png")
plt.show()

nn=np.arange(1,n+1)
plt.plot(nn,np.cumsum(Xar)/nn)
plt.plot(nn,np.cumsum(Xmh)/nn)
plt.hlines(alpha,0,n,color='black')
plt.legend(["AR","MH","True mean"] )
plt.savefig("../figures/erg_mean.png")
plt.show()

sm.plot_acf(Xar,lags=20)
plt.gca().set_rasterized(True)
plt.legend(["AR"] )
plt.savefig("../figures/acfar.png")
plt.show()

sm.plot_acf(Xmh,lags=20)
plt.gca().set_rasterized(True)
plt.legend(["MH"] )
```

```
plt.savefig("../figures/acfMH.png")
plt.show()

sns.kdeplot(Xar, shade=True, color="r")
plt.gca().set_rasterized(True)
x = np.linspace(0,20,100)
plt.plot(x, f(x,alpha,1),color='black')
plt.title('KDE for AR')
plt.savefig('../figures/densAR.png')
plt.show()

sns.kdeplot(Xmh, shade=True, color="r")
plt.gca().set_rasterized(True)
plt.plot(x, f(x,alpha,1),color='black')
plt.title('KDE for MH')
plt.savefig('../figures/densMH.png')
plt.show()
```

Exercise 5 (Optional)

Consider a Markov chain $\{X_n\} \sim \text{Markov}(\pi, P)$ on a discrete state space $\mathcal{X}$ at equilibrium, with P irreducible, and π the unique invariant probability measure of P. Let l^2_π be the Hilbert space $l^2_\pi = \{f : \mathcal{X} \to \mathbb{R} : \sum_{i\in\mathcal{X}} f(i)^2\pi_i < \infty\}$ with inner product $(f,g)_{l^2_\pi} = \sum_{i\in\mathcal{X}} f(i)g(i)\pi_i$, and $l^2_{\pi,0} = \{f \in l^2_\pi : \mathbb{E}_\pi[f] = 0\}$.

1. Show that if (P, π) are in detailed balance, then $(Pf, g)_{l^2_\pi} = (f, Pg)_{l^2_\pi}$ for any $f, g \in l^2_\pi$
2. Show that $\mathbb{E}[f(X_n)f(X_m)] = (P^{m-n}f, f)_{l^2_\pi}$ for any $f \in l^2_\pi$ and $m > n$.
3. Consider now the estimator
$$\hat{\mu}_N = \frac{1}{N}\sum_{n=1}^{N} f(X_n)$$
of $\mu = \mathbb{E}_\pi[f]$ under the assumption that $f \in l^2_\pi$. Show that $\mathbb{E}_\pi[\hat{\mu}_N] = \mu$, and
$$\mathbb{V}ar[\hat{\mu}_N] = \frac{1}{N}\sum_{l=0}^{N} c_l (P^l \tilde{f}, \tilde{f})_{l^2_\pi},$$
with $\tilde{f} = f - \mathbb{E}_\pi[f] \in l^2_{\pi,0}$ and
$$c_{l,N} = \begin{cases} 1, & l = 0 \\ 2(1 - \frac{l}{N}), & l > 0 \end{cases} \tag{3}$$
4. Conclude that the asymptotic variance $\mathbb{V}(f,p) \coloneqq \lim_{N\to\infty} N\mathbb{V}ar_\pi(\hat{\mu}_N)$ satisfies $\mathbb{V}(f,p) = ((2(I-P)^{-1} - I)\tilde{f}, \tilde{f})_{l^2_\pi}$ if
$$\sup_{g\in l^2_{\pi,0}} \frac{(Pg, g)_{l^2_\pi}}{\|g\|_{l^2_\pi}} = \beta < 1. \tag{4}$$

5. Consider now the two irreducible transition matrices P_1 and P_2, both in detailed balance with π and satisfying (4) for some β_1, β_2. Show that if $(P_1)_{ij} \geq (P_2)_{ij} \forall i \neq j$, then

$$\mathbb{V}(f, P_1) \leq \mathbb{V}(f, P_2), \tag{5}$$

for any $f \in l^2_\pi$.

Hint: Take $P(\lambda) = (1-\lambda)P_1 + \lambda P_2, \lambda \in [0,1]$ and show that $\frac{d}{d\lambda}\mathbb{V}(f, P(\lambda)) \geq 0$.

References

[1] A. Gelaman, W.R Gilks and G. Roberts, *Weak convergence and optimal scaling of random walk Metropolis algorithms*, Ann. Appl. Probab. Volume 7, Number 1 (1997), 110-120.
[2] S. Brooks, A. Gelman. *Handbook of Markov Chain Monte Carlo*, 2011, CRC press.