# Bayesian modelling

## Monte Carlo methods, Markov chains and the Metropolis–Hastings algorithm

Léo Belzile

Last compiled Monday Jan 27, 2025

# Bayesian inference beyond conjugate models

How to circumvent the problem of intractable posteriors?

- simulation-based methods: accept-reject, Markov chain Monte Carlo, particle filters, etc.
- deterministic methods: (integrated nested) Laplace approximations, variational Bayes, expectation propagation, etc.

We focus on Monte Carlo methods.

# Simulation algorithms: inversion method

If $F$ is an absolutely continuous distribution function, then

$$F(X) \sim \mathsf{U}(0,1).$$

The inversion method consists in applying the quantile function $F^{-1}$ to $U \sim \mathsf{U}(0,1)$, viz.

$$F^{-1}(U) \sim X.$$

# Inversion method for truncated distributions

Consider a random variable $Y$ with distribution function $F$.

If $X$ follows the same distribution as $Y$, but restricted over the interval $[a, b]$, then

$$\Pr(X \leq x) = \frac{F(x) - F(a)}{F(b) - F(a)}, \qquad a \leq x \leq b,$$

Therefore,

$$F^{-1}[F(a) + \{F(b) - F(a)\}U] \sim X$$

# Fundamental theorem of simulation

Consider a $d$-variate random vector $\boldsymbol{X}$, independent of $U \sim \textbf{unif}(0,1)$ and $c > 0$. If $(\boldsymbol{X}, U)$ is uniformly distributed on the set

$$\mathcal{A}_f = \{(\boldsymbol{x}, u) : 0 \leq u \leq cf(\boldsymbol{x})\},$$

then $\boldsymbol{X}$ has density $f(\boldsymbol{x})$. Conversely, if $\boldsymbol{X}$ has density $f(\boldsymbol{x})$ and $U \sim \textbf{unif}(0,1)$ independently, then $[\boldsymbol{X}, cUf(\boldsymbol{X})]$ is uniformly distributed on $\mathcal{A}_f$

- $f$ is the marginal density of $\boldsymbol{X}$ since $f(\boldsymbol{x}) = \int_0^{f(\boldsymbol{x})} \mathrm{d}u$.
- If we can simulate uniformly from $\mathcal{A}_f$, then, we can discard the auxiliary variable $u$. See Devroye (1986), Theorem 3.1.

# Fundamental theorem of simulation in picture

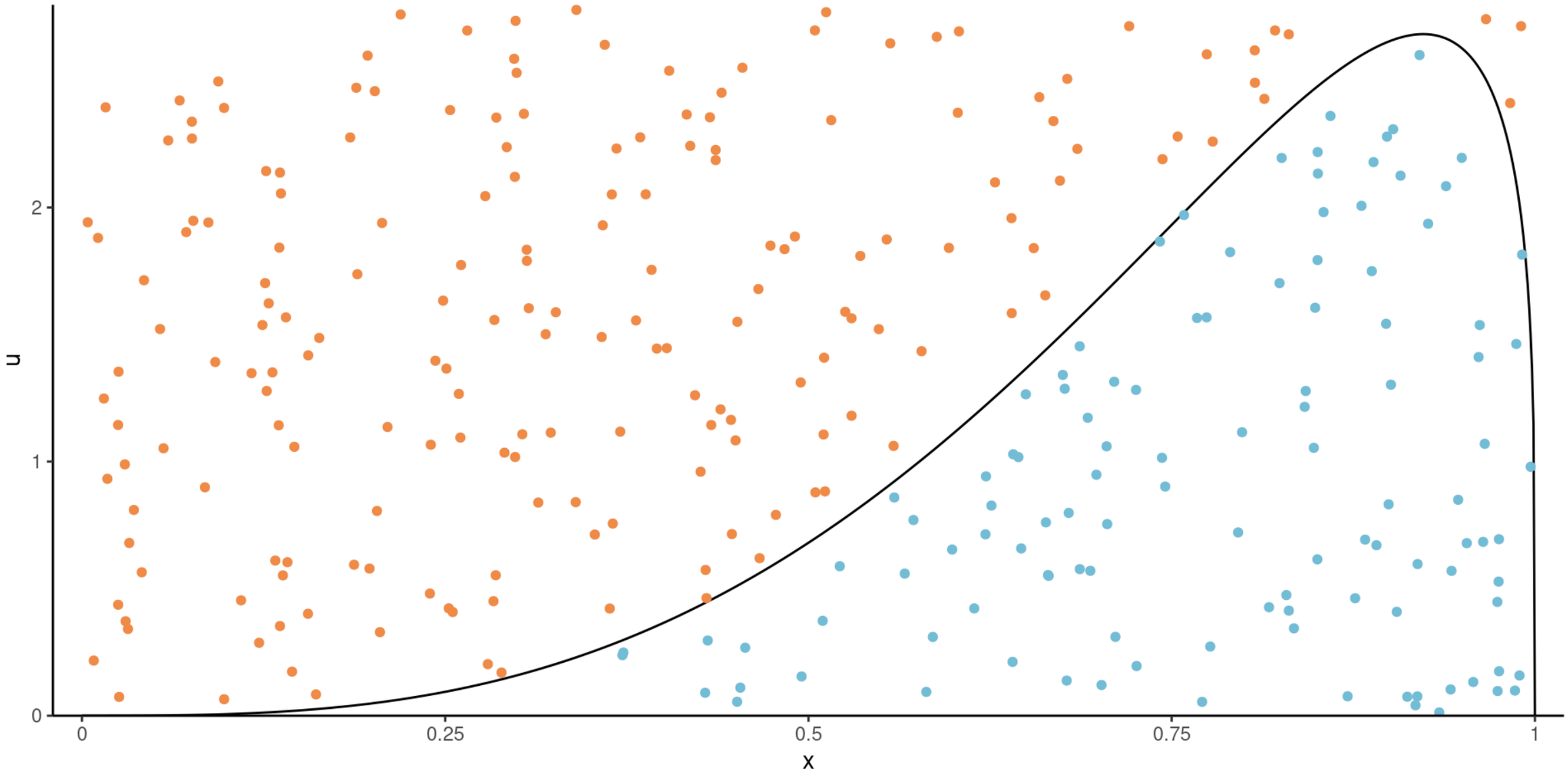


Figure 1: Illustration of the fundamental theorem of simulation. All points in blue below the density curve belong to $\mathcal{A}_f$.

# Simulation algorithms: accept-reject

- **Target**: sample from density $p(x)$ (hard to sample from)
- **Proposal**: find a density $q(x)$ with nested support, $\text{supp}(p) \subseteq \text{supp}(q)$, such that

$$\frac{p(x)}{q(x)} \leq C, \quad C \geq 1.$$

# Rejection sampling algorithm

1. Generate $X$ from proposal with density $q(x)$.
2. Compute the ratio $R \leftarrow p(X)/q(X)$.
3. If $CU \leq R$ for $U \sim \mathsf{U}(0,1)$, return $X$, else go back to step 1.

# Remarks on rejection sampling

- Acceptance rate is $1/C$
  - we need on average $C$ draws from $q$ to get one from $p$
- $q$ must be more heavy-tailed than $p$
  - e.g., $q(x)$ Student-$t$ for $p(x)$ Gaussian
- $q$ should be cheap and easy to sample from!

# Designing a good proposal density

Good choices must satisfy the following constraints:

- pick a family $q(x)$ so that

$$C = \sup_x \frac{p(x)}{q(x)}$$

  is as close to 1 as possible.

- you can use numerical optimization with $f(x) = \log p(x) - \log q(x)$ to find the mode $x^\star$ and the upper bound $C = \exp f(x^\star)$.

# Accept-reject illustration

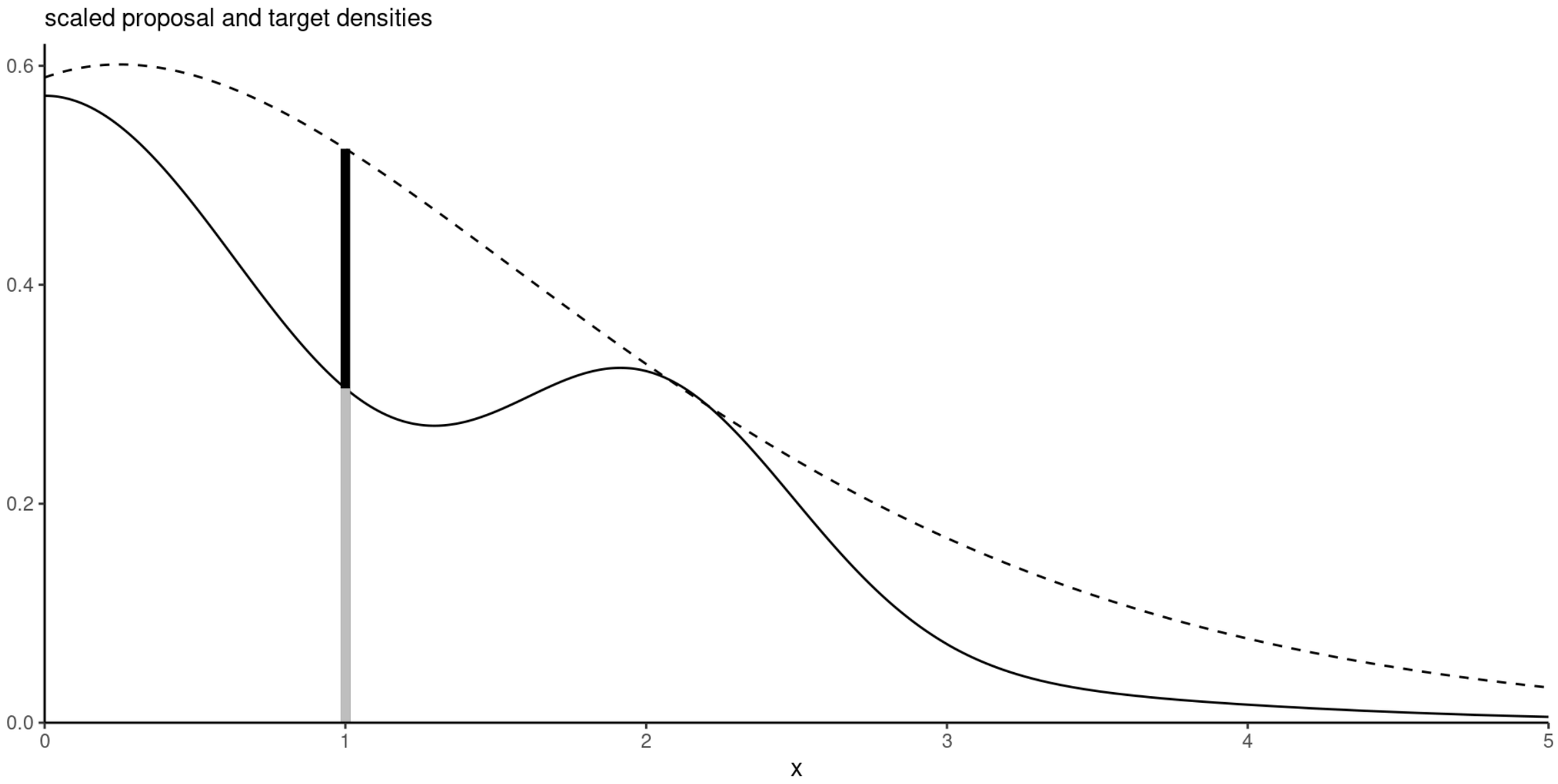


Figure 2: Target density (full) and scaled proposal density (dashed): the vertical segment at $x = 1$ shows the percentage of acceptance for a uniform slice under the scaled proposal, giving an acceptance ratio of 0.58.

# Truncated Gaussian via accept-reject

Consider sampling $Y \sim \mathsf{No}(\mu, \sigma^2)$, but truncated in the interval $(a, b)$. The target density is

$$p(x; \mu, \sigma, a, b) = \frac{1}{\sigma} \frac{\phi\left(\frac{x-\mu}{\sigma}\right)}{\Phi(\beta) - \Phi(\alpha)}.$$

for $\alpha = (a - \mu)/\sigma$ and $\beta = (b - \mu)/\sigma$. where $\phi(\cdot)$, $\Phi(\cdot)$ are respectively the density and distribution function of the standard Gaussian distribution.

# Accept-reject (crude version)

1. Simulate $X \sim \text{No}(\mu, \sigma^2)$
2. reject any draw if $X < a$ or $X > b$.

The acceptance rate is $C^{-1} = \{\Phi(\beta) - \Phi(\alpha)\}$

```r
# Standard Gaussian truncated on [0,1]
candidate <- rnorm(1e5)
trunc_samp <- candidate[candidate >= 0 & candidate <= 1]
# Acceptance rate
length(trunc_samp)/1e5
```

```
[1] 0.34289
```

```r
# Theoretical acceptance rate
pnorm(1)-pnorm(0)
```

```
[1] 0.3413447
```

# Accept-reject for truncated Gaussian

Since the Gaussian is a location scale family, the inversion method gives

$$X \sim \mu + \sigma \Phi^{-1}\left[\Phi(\alpha) + \{\Phi(\beta) - \Phi(\alpha)\}U\right]$$

We however need to evaluate $\Phi$ numerically (no closed-form expression).

The method fails for *rare event* simulation because the computer returns

- $\Phi(x) = 0$ for $x \leq -39$
- $\Phi(x) = 1$ for $x \geq 8.3$,

implying that $a \leq 8.3$ for this approach to work (Botev & L'Écuyer, 2017).

# Simulating tails of Gaussian variables

We consider simulation from a standard Gaussian truncated above $a > 0$

Write the density of the truncated Gaussian as (Devroye, 1986, p. 381)

$$f(x) = \frac{\exp(-x^2/2)}{\int_a^\infty \exp(-z^2/2)\mathrm{d}z} = \frac{\exp(-x^2/2)}{c_1}.$$

Note that, for $x \geq a$,

$$c_1 f(x) \leq \frac{x}{a}\exp\left(-\frac{x^2}{2}\right) = a^{-1}\exp\left(-\frac{a^2}{2}\right)g(x);$$

where $g(x)$ is the density of a Rayleigh variable shifted by $a$.

The constant $C = \exp(-a^2/2)(c_1 a)^{-1}$ approaches 1 quickly as $a \to \infty$ (asymptotically optimality).

# Accept-reject: truncated Gaussian with Rayleigh

The shifted Rayleigh has distribution function

$$G(x) = 1 - \exp\{(a^2 - x^2)/2\}, x \geq a.$$

**Marsaglia algorithm**

1. Generate a shifted Rayleigh above $a$, $X \leftarrow \{a^2 - 2\log(U)\}^{1/2}$ for $U \sim \mathsf{U}(0,1)$
2. Accept $X$ if $XV \leq a$, where $V \sim \mathsf{U}(0,1)$.

For sampling on $[a, b]$, propose from a Rayleigh truncated above at $b$ (Botev & L'Écuyer, 2017).

```
a <- 8.3
niter <- 1000L
X <- sqrt(a^2 + 2*rexp(niter))
samp <- X[runif(niter)*X <= a]
```

# Markov chains

Plain ordinary Monte Carlo is great, but few algorithms are generic enough to be useful in complex high-dimensional problems.

We will instead typically build Markov chains that target an invariant stationary distribution.

# Caveats?

Markov chain Monte Carlo methods generate **correlated** draws.

**Questions**:

1. can we use them as ordinary independent samples?
2. what is the price to pay?

We need to do a little theoretical detour to answer these questions.

# Stationarity and Markov property

A stochastic process is

- **(strongly) stationary** if the distribution of $\{X_1, \ldots, X_t\}$ is the same as that of $\{X_{n+1}, \ldots X_{t+n}\}$ for any value of $n$ and given $t$.
- **weakly stationary** if $\mathsf{E}(X_t) = \mu$ for all $t$, and $\mathsf{Cov}(X_t, X_{t+h}) = \gamma_h$ does not depend on $t$.
- **Markov** if it satisfies the Markov property: given the current state of the chain, the future only depends on the current state and not on the past.

Strong stationarity implies weak stationarity.

# Autoregressive process of order 1

Consider a first-order autoregressive process, or $\mathsf{AR}(1)$,

$$Y_t = \mu + \phi(Y_{t-1} - \mu) + \varepsilon_t,$$

where

- $\phi$ is the lag-one correlation,
- $\mu$ the global mean
- $\varepsilon_t$ is an iid innovation with mean zero and variance $\sigma^2$

If $|\phi| < 1$, the process is stationary, otherwise the variance increases with $t$.

# Unconditional moments via recursion

The Gaussian $\mathsf{AR}(1)$ stationarity process follows $Y_t \sim \mathsf{Gauss}(\mu, \sigma^2/(1-\phi^2)$ marginally.

$$\begin{aligned} Y_t &= \mu(1-\phi) + \varepsilon_t + \phi\{\mu + \phi(Y_{t-2} - \mu) + \varepsilon_{t-1}\} \\ &= \mu + \sum_{j=0}^{\infty} \phi_j \varepsilon_{t-j} \end{aligned}$$

whence $\mathsf{E}(Y_t) = \mu$ and

$$\mathsf{Va}(Y_t) = \mathsf{Va}\left(\sum_{j=0}^{\infty} \phi_j \varepsilon_{t-j}\right) = \sum_{j=0}^{\infty} \phi_j^2 \mathsf{Va}(\varepsilon_{t-j}) = \frac{\sigma^2}{(1-\phi^2)}$$

where the geometric series converges if $\phi < 1$ and diverges otherwise.

# Unconditional moments via tower law

If the process is weakly stationary, then $\mathsf{E}_{Y_t}(Y_t) = \mathsf{E}_{Y_{t-1}}(Y_{t-1})$

$$\begin{aligned}\mathsf{E}_{Y_t}(Y_t) &= \mathsf{E}_{Y_{t-1}}\left\{\mathsf{E}_{Y_t|Y_{t-1}}(Y_t)\right\}\\ &= \mu(1-\phi) + \phi\mathsf{E}_{Y_{t-1}}(Y_{t-1})\end{aligned}$$

and so the unconditional mean is $\mu$. For the variance, we have

$$\begin{aligned}\mathsf{E}_{Y_t}(Y_t) &= \mathsf{E}_{Y_{t-1}}\left\{\mathsf{Va}_{Y_t|Y_{t-1}}(Y_t)\right\} + \mathsf{Va}_{Y_{t-1}}\left\{\mathsf{E}_{Y_t|Y_{t-1}}(Y_t)\right\}\\ &= \sigma^2 + \mathsf{Va}_{Y_{t-1}}\left\{\mu + \phi(Y_{t-1}-\mu)\right\}\\ &= \sigma^2 + \phi^2\mathsf{Va}_{Y_{t-1}}(Y_{t-1}).\end{aligned}$$

# Variance of sample average

Intuitively, a sample of correlated observations carries less information than an independent sample of draws.

We want the variance of the sample average, which is

$$\mathsf{Va}\left(\overline{Y}_T\right) = \frac{1}{T^2}\sum_{t=1}^{T}\sum_{s=1}^{T}\mathsf{Co}(Y_t, Y_s)$$

$$= \frac{1}{T^2}\sum_{t=1}^{T}\mathsf{Va}(Y_t) + \frac{2}{T^2}\sum_{t=1}^{T-1}\sum_{s=t+1}^{T}\mathsf{Co}(Y_t, Y_s).$$

# Autocovariance and Markov property

The covariance at lag $k$, in terms of innovations, gives

$$\gamma_k = \mathsf{Co}(Y_t, Y_{t-k}) = \mathsf{Va}(\phi Y_{t-1}, Y_{t-k}) + \mathsf{Va}(\varepsilon_t, Y_{t-k}) = \phi\gamma_{k-1}$$

so by recursion $\gamma_k = \gamma^k \mathsf{Va}(Y_t)$.

The $\mathsf{AR}(1)$ process is first-order Markov since the conditional distribution $p(Y_t \mid Y_{t-1}, \ldots, Y_{t-p})$ equals $p(Y_t \mid Y_{t-1})$.

# Variance of sample average, redux

If a central limit theorem applies, the limiting variance of the sample mean simplifies to

$$\lim_{T\to\infty} T\mathsf{Va}\left(\overline{Y}_T\right) = \tau^2\left\{1+2\sum_{t=1}^{\infty}\gamma_t\right\}.$$

which is a function of

- the unconditional variance $\tau^2$
- the lag-$k$ autocorrelation $\mathsf{Cor}(Y_t, Y_{t+k}) = \gamma_k$.

# Correlogram

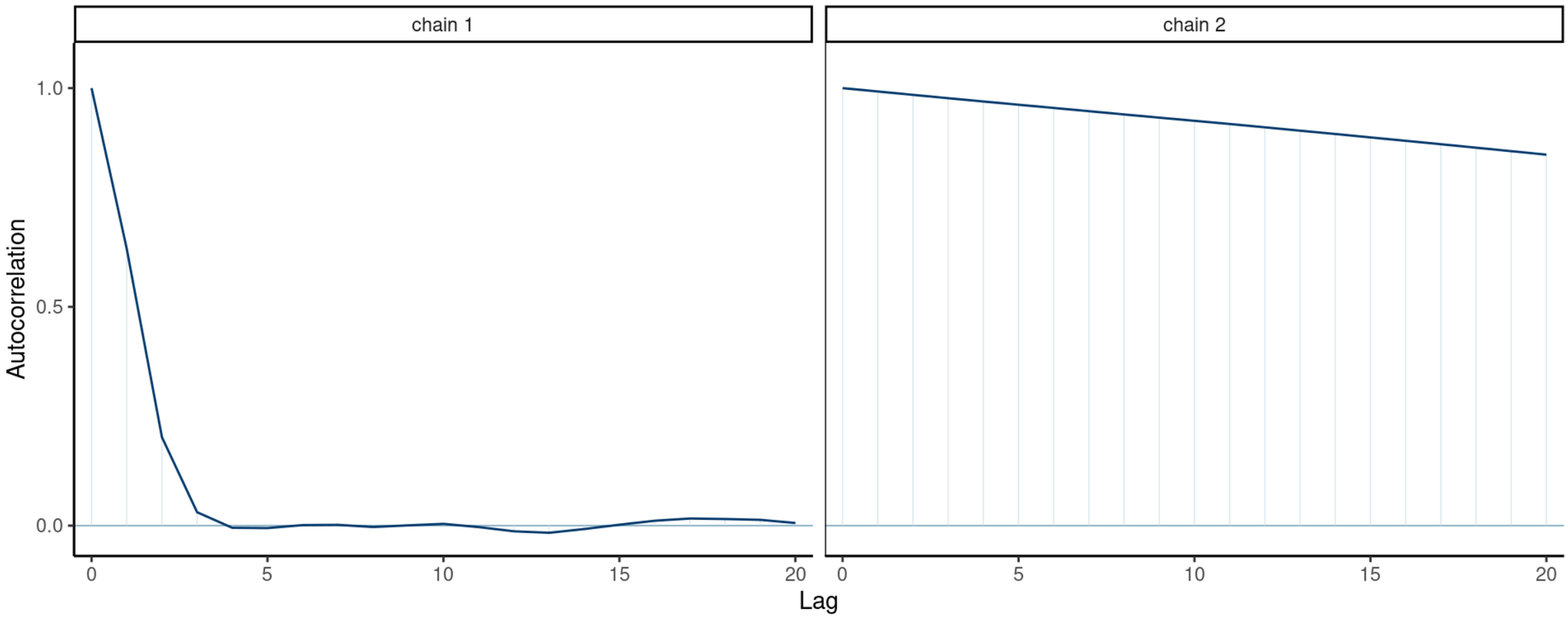


Figure 3: Correlogram of two Markov chains. These plots, often called acf or autocorrelation functions, show the lag-k sample autocorrelation against lag number.

# Variance of sample mean of AR(1)

The lag-$k$ correlation of the stationary autoregressive process of order 1 is $\phi^k$, so

$$T\mathsf{Va}\left(\overline{Y}_T\right) = \sigma^2(1+\phi)/(1-\phi).$$

For an independent sample, we have

$$T\mathsf{Va}\left(\overline{Y}_T\right) = \sigma^2/(1-\phi^2).$$

# Inefficiency curve for AR(1)

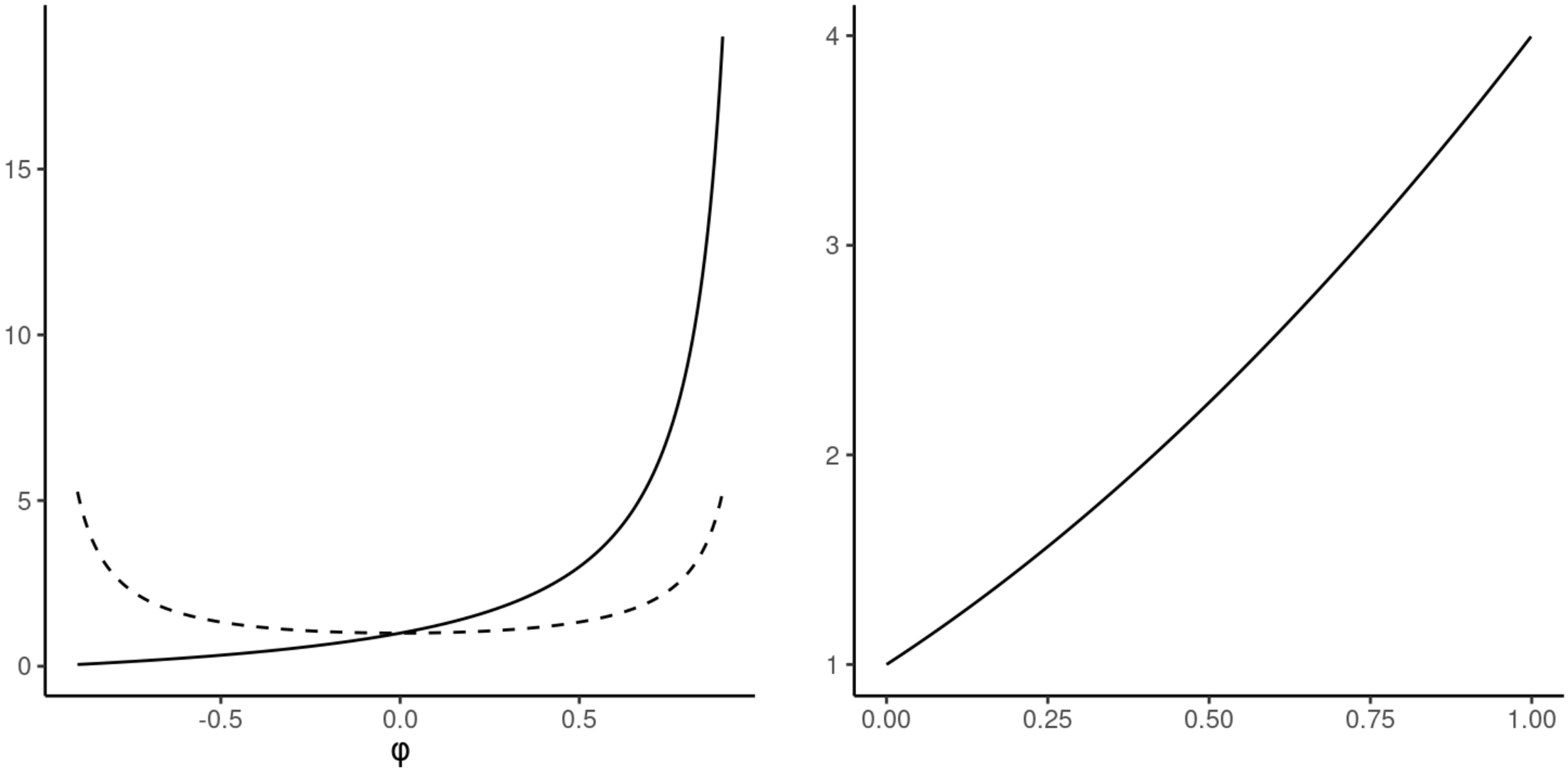


Figure 4: Left: scaled asymptotic variance of the sample mean for AR(1) (full line) and independent observations with the same marginal variance (dashed). Right: variance ratio for positive correlations.

To get the same precision for the mean of $\mathsf{AR}(1)$ process with $\phi \approx 0.75$ than with i.i.d. data, we would need 9 times as many observations.

# Morale of the story

The price to pay for having correlated samples is

**inefficiency**

The higher the autocorrelation, the larger the variability of our estimators.

# When can we use Markov chains?

If a Markov chain is irreducible and acyclic, it has a unique stationary distribution.

- **irreducibility**: means that the chain can move from anywhere to anywhere, so it doesn't get stuck in part of the space forever.
- **acyclic**: cyclical chains loop around and visit periodically a state

Ergodic theorem is our guarantee of convergence (does not require acyclic).

# Examples of cyclical or reducible chains

Consider discrete Markov chains over the integers $1, 2, 3$ with transition matrices

$$P_1 = \begin{pmatrix} 0.5 & 0.3 & 0.2 \\ 0 & 0.4 & 0.6 \\ 0 & 0.5 & 0.5 \end{pmatrix}, \quad P_2 = \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}.$$

Chain 1 is reducible to $\{2, 3\}$, chain 2 is cyclical.

# Law of large number (ergodic theorem)

Ergodicity means that two segments of a time series far enough apart act as independent.

Let $\{Y_t\}$ is a weakly stationary sequence with mean $\mathsf{E}(Y_t) = \mu$ and $\gamma_h = \mathsf{Cov}(Y_t, Y_{t+h})$. Then, if the autocovariance series is convergent, meaning

$$\sum_{t=0}^{\infty} |\gamma_h| < \infty,$$

then $\{Y_t\}$ is ergodic for the mean and $\overline{Y} \xrightarrow{\text{p}} \mu$.

# Ergodicity and transformations

The ergodic theorem is a law of large numbers for stochastic processes that allows for serial dependence between observations, provided the latter is not too large.

Any transformation $g(\cdot)$ of a stationary and ergodic process $\{Y_t\}$ retains the properties, so $\overline{g} = T^{-1}\sum_{t=1}^{T} g(Y_t) \to \mathsf{E}\{g(Y_t)\}$ as $T \to \infty$.

# Convergence of Markov chains

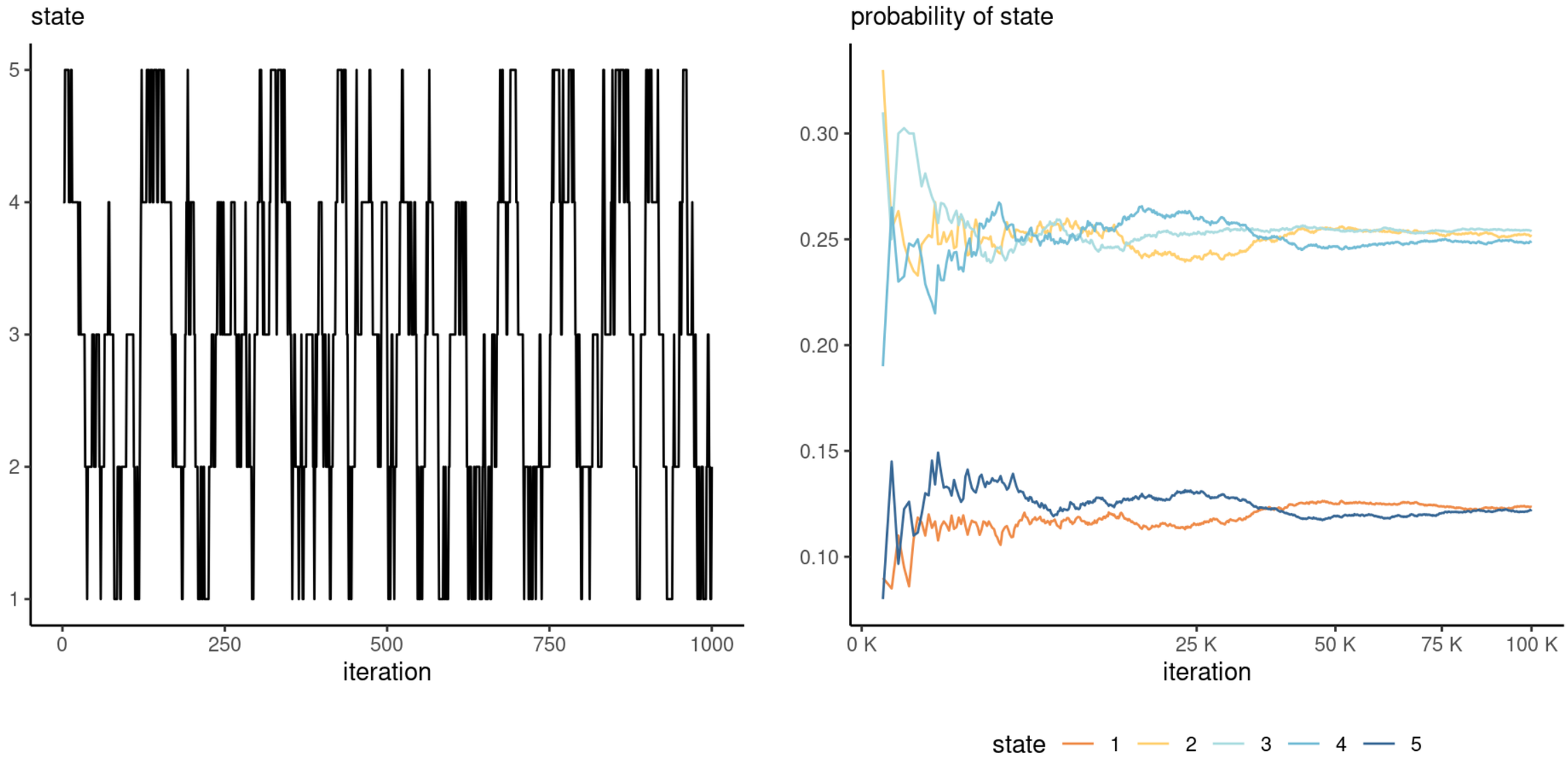


Figure 5: Discrete Markov chain on integers from 1 to 5, with traceplot of 1000 first iterations (left) and running mean plots of sample proportion of each state visited (right).

# Markov chain Monte Carlo

We consider simulating from a distribution with associated density function $\propto p(\boldsymbol{\theta})$.

- known up to a normalizing factor not depending on $\boldsymbol{\theta}$.

We use a conditional density $q(\boldsymbol{\theta} \mid \boldsymbol{\theta}^{\text{cur}})$ to generate proposals from the current value $\boldsymbol{\theta}^{\text{cur}}$.

# Transition kernel and detailed balanced

## Theoretical details provided for completeness

A transition kernel $K(\boldsymbol{\theta}^{\text{cur}}, \boldsymbol{\theta}^{\text{prop}})$ proposes a move from the current value $\boldsymbol{\theta}^{\text{cur}}$ to a proposal $\boldsymbol{\theta}^{\text{prop}}$.

If our target is $p(\boldsymbol{\theta})$, then the Markov chain satisfies the **detailed balance** condition with respect to $p(\cdot)$ if

$$K(\boldsymbol{\theta}^{\text{cur}}, \boldsymbol{\theta}^{\text{prop}})p(\boldsymbol{\theta}^{\text{cur}}) = K(\boldsymbol{\theta}^{\text{prop}}, \boldsymbol{\theta}^{\text{cur}})p(\boldsymbol{\theta}^{\text{prop}}).$$

If a Markov chain satisfies the detailed balance with respect to $p(\cdot)$, then the latter is necessarily the invariant density of the Markov chain and the latter is reversible.

# Transition kernel of Metropolis–Hastings algorithm

## Theoretical details provided for completeness

The Metropolis–Hastings algorithm has transition kernel for a move from $\boldsymbol{x}$ to a proposal $\boldsymbol{y}$

$$K(\boldsymbol{x}, \boldsymbol{y}) = \alpha(\boldsymbol{x}, \boldsymbol{y}) q(\boldsymbol{y} \mid \boldsymbol{x}) + \{1 - r(\boldsymbol{x})\} \mathsf{I}(\boldsymbol{y} = \boldsymbol{x})$$

where $r(\boldsymbol{x}) = \int \alpha(\boldsymbol{x}, \boldsymbol{y}) q(\boldsymbol{y} \mid \boldsymbol{x}) \mathrm{d}\boldsymbol{y}$ is the average probability of acceptance of a move from $\boldsymbol{x}$, $\mathsf{I}(\cdot = \boldsymbol{x})$ is a point mass at $\boldsymbol{x}$, and $\alpha(\cdot)$ is defined on the next slide.

The Metropolis–Hastings algorithm satisfies detailed balanced.

# Metropolis–Hastings algorithm

Starting from an initial value $\boldsymbol{\theta}_0$:

1. draw a proposal value $\boldsymbol{\theta}_t^\star \sim q(\boldsymbol{\theta} \mid \boldsymbol{\theta}_{t-1})$.

2. Compute the acceptance ratio

$$R = \frac{p(\boldsymbol{\theta}_t^\star)}{p(\boldsymbol{\theta}_{t-1})} \frac{q(\boldsymbol{\theta}_{t-1} \mid \boldsymbol{\theta}_t^\star)}{q(\boldsymbol{\theta}_t^\star \mid \boldsymbol{\theta}_{t-1})}$$

3. With probability $\alpha = \min\{R, 1\}$, accept the proposal and set $\boldsymbol{\theta}_t \leftarrow \boldsymbol{\theta}_t^\star$, otherwise set the value to the previous state, $\boldsymbol{\theta}_t \leftarrow \boldsymbol{\theta}_{t-1}$.

# Interpretation

- If $R > 1$, the proposal has higher density and we always accept the move.
- If we reject the move, the Markov chain stays at the current value, which induces autocorrelation.
- Since the acceptance probability depends only on the density through ratios, normalizing factors of $p$ and $q$ cancel out.

# Symmetric proposals and random walk

If the proposal is symmetric, the ratio of proposal densities is

$$q(\boldsymbol{\theta}_{t-1} \mid \boldsymbol{\theta}_t^\star)/q(\boldsymbol{\theta}_t^\star \mid \boldsymbol{\theta}_{t-1}) = 1.$$

Common examples include random walk proposals

$$\boldsymbol{\theta}_t^\star \leftarrow \boldsymbol{\theta}_{t-1} + \tau Z, \qquad Z$$

where $Z$ is a mean zero, variance one random variable.

# Independent proposals

- If we pick instead a global proposal, we must ensure that $q$ samples in far regions (recall rejection sampling), otherwise …
- Good proposals include heavy tailed distribution such as Student-$t$ with small degrees of freedom, centered at the maximum a posteriori $\widehat{\boldsymbol{\theta}}$ and with scale matrix $-\mathbf{H}^{-1}(\boldsymbol{\theta}_t^\star)$, where $\mathbf{H}(\cdot)$ is the Hessian of the log posterior.

# Upworthy data example

We model the Poisson rates for headlines with questions or not. Our model is

$$
\begin{aligned}
Y_i &\sim \mathsf{Po}(n_i\lambda_i), \qquad (i = 1, 2)\\
\lambda_1 &= \exp(\beta + \kappa)\\
\lambda_2 &= \exp(\beta)\\
\beta &\sim \mathsf{No}(\log 0.01, 1.5)\\
\kappa &\sim \mathsf{No}(0, 1)
\end{aligned}
$$

# Implementation details: data and containers

## In regression models, scale inputs if possible.

```
data(upworthy_question, package = "hecbayes")
# Compute sufficient statistics
data <- upworthy_question |>
  dplyr::group_by(question) |>
  dplyr::summarize(ntot = sum(impressions),
                   y = sum(clicks))
# Create containers for MCMC
niter <- 1e4L
chain <- matrix(0, nrow = niter, ncol = 2L)
colnames(chain) <- c("beta","kappa")
```

# Implementation details: log posterior function

## Perform all calculations on the log scale to avoid numerical overflow!

```r
# Code log posterior as sum of log likelihood and log prior
loglik <- function(par, counts = data$y, offset = data$ntot, ...){
  lambda <- exp(c(par[1] + log(offset[1]), par[1] + par[2] + log(offset[2])))
 sum(dpois(x = counts, lambda = lambda, log = TRUE))
}
# Note common signature of function
logprior <- function(par, ...){
  dnorm(x = par[1], mean = log(0.01), sd = 1.5, log = TRUE) +
    dnorm(x = par[2], log = TRUE)
}
logpost <- function(par, ...){
  loglik(par, ...) + logprior(par, ...)
}
```

# Implementation details: proposals

Use good starting values for your Markov chains, such as maximum a posteriori.

```
# Compute maximum a posteriori (MAP)
map <- optim(
  par = c(-4, 0.07),
  fn = logpost,
  control = list(fnscale = -1),
  offset = data$ntot,
  counts = data$y,
  hessian = TRUE)
# Use MAP as starting value
cur <- map$par
# Compute logpost_cur - we can keep track of this to reduce calculations
logpost_cur <- logpost(cur)
# Proposal covariance
cov_map <- -2*solve(map$hessian)
chol <- chol(cov_map)
```

# Implementation details: Metropolis–Hastings algorithm

Use seed for reproducibility, do not compute posterior twice, compute log of acceptance ratio.

```
set.seed(80601)
naccept <- 0L
for(i in seq_len(niter)){
  # Multivariate normal proposal - symmetric random walk
  prop <- c(rnorm(n = 2) %*% chol + cur)
  logpost_prop <- logpost(prop)
  logR <- logpost_prop - logpost_cur
  if(logR > -rexp(1)){
    cur <- prop
    logpost_cur <- logpost_prop
    naccept <- naccept + 1L
  }
  chain[i,] <- cur
}
```

# Implementation details: analysis of output

```r
# Posterior summaries
summary(coda::as.mcmc(chain))
# Computing standard errors using batch means
sqrt(diag(mcmc::olbm(chain, batch.length = niter/40)))
```

```
1. Empirical mean and standard deviation for each variable,
   plus standard error of the mean:

          Mean       SD  Naive SE Time-series SE
beta  -4.51268 0.001697 1.697e-05      6.176e-05
kappa  0.07075 0.002033 2.033e-05      9.741e-05

2. Quantiles for each variable:

          2.5%      25%      50%      75%    97.5%
beta  -4.51591 -4.51385 -4.51273 -4.51154 -4.50929
kappa  0.06673  0.06933  0.07077  0.07212  0.07463
```

# Standard errors for posterior means – batch means

Geyer (2011) recommends to segment the time series into batches

1. Break the chain of length $B$ (after burn in) in $K$ blocks of size $\approx K/B$.
2. Compute the sample mean of each segment.
3. Compute the standard deviation of the segments mean.
4. Rescale by $K^{-1/2}$ to get standard error of the global mean.

# Illustration of batch means

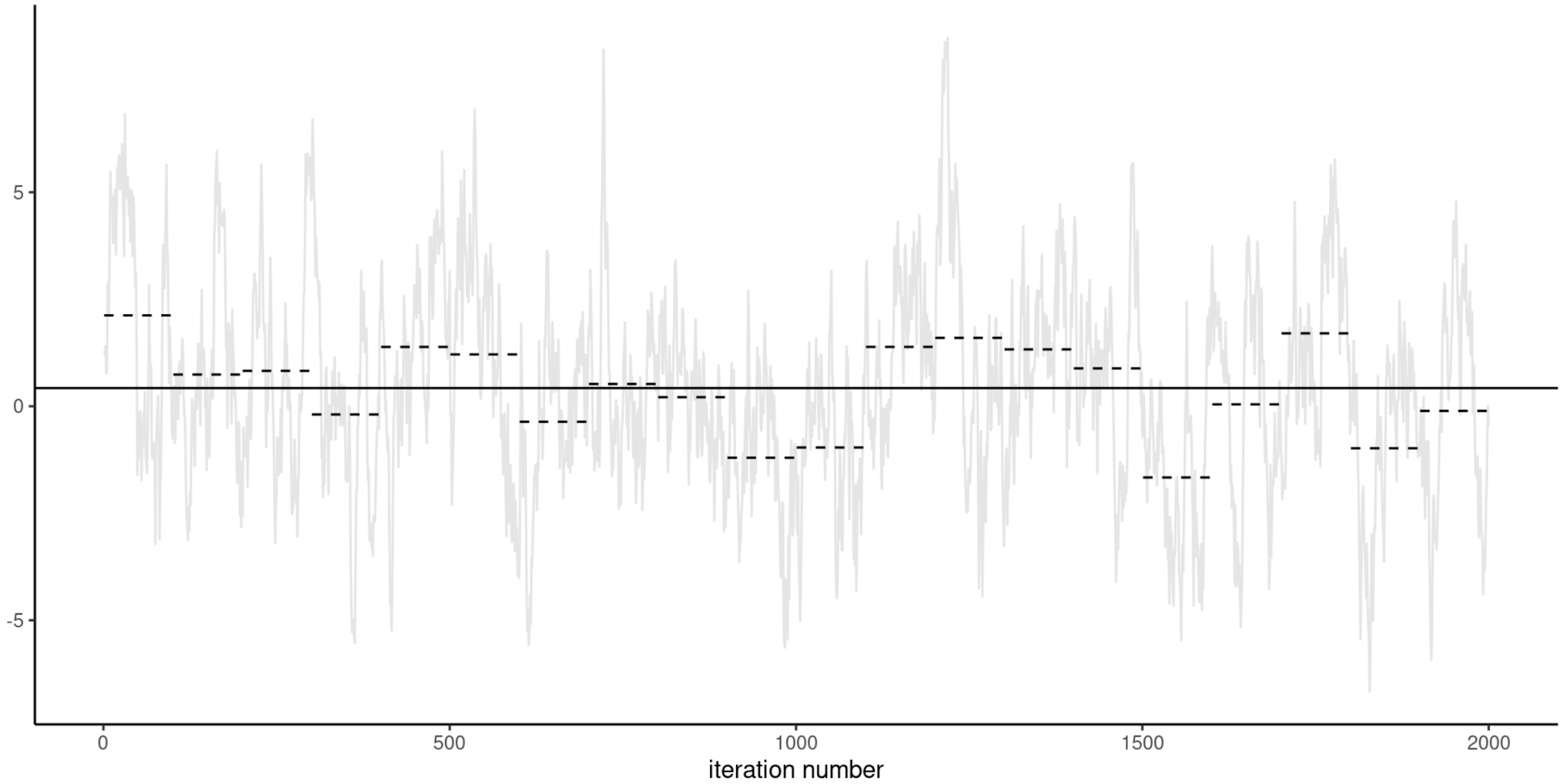


Figure 6: Calculation of the standard error of the posterior mean using the batch method.

# Standard errors for posterior means — autoregressive

We can also fit an high-order autoregressive process $\mathsf{AR}(p)$ and approximate the unconditional variance by that, and divide by $\sqrt{T}$.

- Standard methods (Yule–Walker equations, maximum likelihood, spectral estimation) apply.

# References

Botev, Z., & L'Écuyer, P. (2017). Simulation from the normal distribution truncated to an interval in the tail. *Proceedings of the 10th EAI International Conference on Performance Evaluation Methodologies and Tools on 10th EAI International Conference on Performance Evaluation Methodologies and Tools*, 23–29. https://doi.org/10.4108/eai.25-10-2016.2266879

Devroye, L. (1986). *Non-Uniform Random Variate Generation*. Springer. http://www.nrbook.com/devroye/

Geyer, C. J. (2011). Introduction to Markov chain Monte Carlo. In S. Brooks, A. Gelman, G. Jones, & X. L. Meng (Eds.), *Handbook of Markov chain Monte Carlo* (pp. 3–48). CRC Press. https://doi.org/10.1201/b10905