Statistics Norway
Statistisk sentralbyrå

Statistics Norway
Department of Coordination and Development
and Research Department

Li-Chun Zhang and Joseph Sexton

Documents

ABC of Markov chain Monte Carlo

Foreword

Markov chain Monte Carlo (MCMC) has been one of the most active research and application areas for statistical methods of data analysis in the recent decade. Strongly rooted and influenced by the Bayesian statistical inference approach, the MCMC provides means of dealing with complicated models which are difficult, if not impossible, to handle otherwise. However, sampling/simulation based inference need not be restricted to Bayesian posterior calculations alone (which involve Monte Carlo methods). In some ways, Markov chain sampling can be compared to resampling under Bootstrap. Whereas Bootstrap resampling is either directed at the non-parametric empirical distribution of the sample, or the estimated parametric model, Markov chain sampling can in principle be targeted at any functions, including the likelihood and the posterior distribution. In both cases, what we do with the re-samples makes up the statistical inference, whereas how we get the specified/required samples is more of a numerical/technical issue. We therefore consider Markov chain sampling and Monte Carlo approximation two separate matters. In particular, Markov chain sampling provides us with inferential possibilities, Bayesian or not, which liberate the statisticians from numerical poverty, allowing applications to ever more complex situations.

This note has been designed as an introduction to the subject of Markov chain Monte Carlo. The three chapters deal with, respectively, Monte Carlo, Markov chain theory (relevant for Markov chain sampling), and Markov chain Monte Carlo. But we hope that it will be more than just an introduction. The materials have been organized so as to allow quick look-up of the various details. A number of examples, based on real-life data sets including the Norwegian Labour Force survey, have been worked out, and the relevant Splus codes included. In this way, even the skilled user may find it helpful as desk-reference.

Contents

Chapter 1

Monte Carlo

1.1 Introduction

Monte Carlo methods originate from the need to evaluate integrals of the following form:

$$I = E_\pi(f) = \int_\Omega f(x)\pi(x)dx, \tag{1.1}$$

where $\pi(x)$ is the probability density function (p.d.f.) of some random X and $f(x)$ some real-valued function such that $E_\pi(f)$ exists.

In contrast to deterministic numerical methods such as the composite Simpson's rule or Gaussian Quadrature or Laplace's approximation, Monte Carlo methods stochastically evaluate I (1.1). The various methods described in this chapter are all based on independent random samples, for which reason they sometimes are referred to as *independence Monte Carlo.*

Generally speaking, the precision of the Monte Carlo methods is controlled by the size of the sample, and has nothing to do with Ω. Errors can routinely be assessed based on the same sample generated, wherever the corresponding central limiting theorem (CLT) applies, which is a distinctive feature of these methods compared to many deterministic methods.

The Monte Carlo methods described in this chapter include the simple Monte Carlo, the acceptance sampling, and the importance sampling. These will be illustrated through examples as well as Splus transcripts. Variance reduction techniques will be briefly discussed in the end.

1.2 Simple Monte Carlo

Given random sample of independent, identically distributed replicates from $\pi(x)$, denoted by

$$X_1, \ldots, X_m \overset{i.i.d.}{\sim} \pi(x),$$

the *simple Monte Carlo* is defined as

$$I_s = \frac{1}{m}\sum_{i=1}^{m} f(x_i). \tag{1.2}$$

Due to the strong law of large numbers, we have

$$I_s \overset{a.s.}{\to} I \qquad \Leftrightarrow \qquad P[\lim_{m\to\infty} I_s = I] = 1.$$

Moreover, existence (i.e. finiteness) of

$$\sigma_\pi^2(f) = \int (f - I)^2 \pi(x) dx, \tag{1.3}$$

implies the following CLT

$$\sqrt{m}(I_s - I) \overset{D}{\to} N(0, \sigma_\pi^2).$$

Example The following transcript contains an Splus simple Monte Carlo routine, in case that $f(x) = \cos(x)$ and $\pi \simeq N(0,1)$, with sample size m (10000 by default) as the calling parameter:

```
smc.cos.norm <- function(m = 10000)
{
        x <- rnorm(m, 0, 1)      # random sample from N(0,1) in x
        i.s <- mean(cos(x))      # simple Monte Carlo in i.s
        sig2.pi <- var(cos(x))   # var_pi based on the same sample
        s.s <- sqrt(sig2.pi/m)   # standard error of i.s
        list(m = m, I.s = i.s, Sigma2.pi = sig2.pi, SD.s = s.s)
}
```

1.3 Acceptance sampling

Denote by $\psi(x)$ a *source distribution*, such that

$$\forall x \in \Omega,\ \pi(x) > 0 \Rightarrow \psi(x) > 0 \qquad \text{and} \qquad \sup_x\ \pi(x)/\psi(x) = a < \infty.$$

Acceptance sampling can be described as the following: let $\alpha(x) = \pi(x)/[a\psi(x)]$,

- generate $U \sim Unif(0,1)$ independent of $X \sim \psi(x)$,
- accept x if $u \leq \alpha(x)$; otherwise, repeat sampling of (U, X) till acceptance.

We have, based on acceptance of $X_1, ..., X_m$,

$$X_1,\ ...,\ X_m \quad | \quad \text{Acceptance} \qquad \overset{i.i.d.}{\sim} \pi(x),$$

to which the simple Monte Carlo (1.2) applies. In particular, the *acceptance rate* is given by

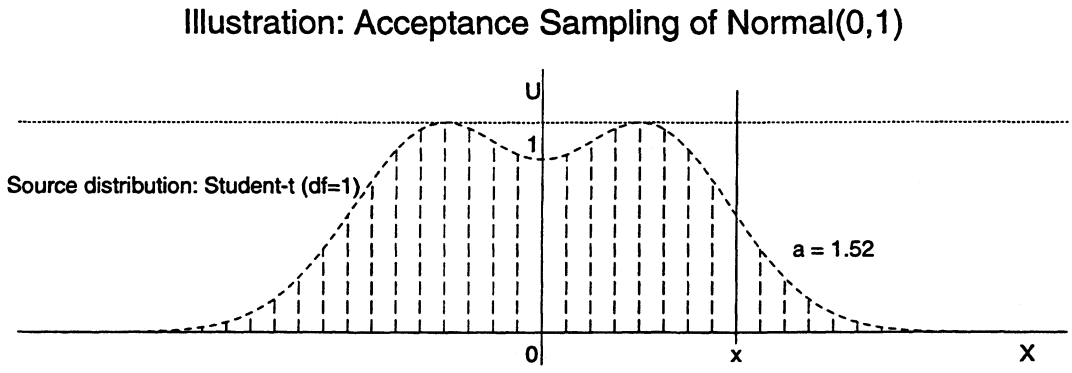
$$P[\text{Acceptance}] = 1/a. \tag{1.4}$$

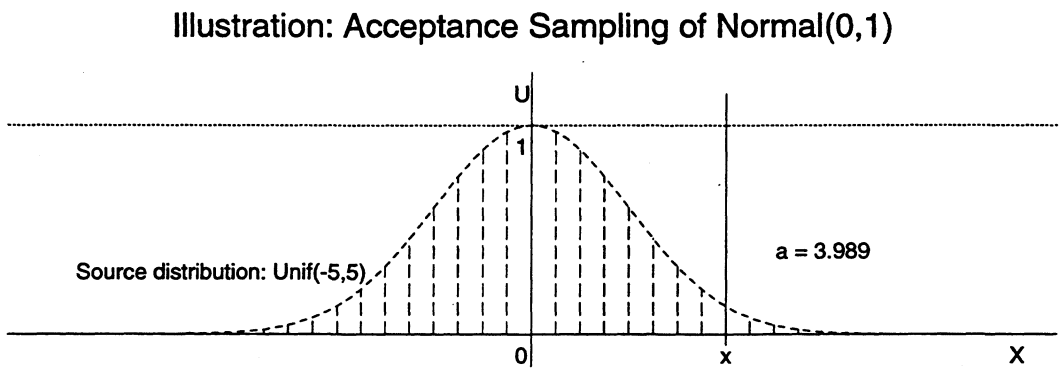
Generic Splus code for acceptance sampling

```
pi.x <- function(x)
{       calculation of the target p.d.f. for the sample }

psi.x <- function(x)
{       calculation of the source p.d.f. for the sample }

amc.sample <- function(n = 10000)
{
        generate independent source sample x of size n
        w.x <- pi.x(x)/psi.x(x) # p.d.f. ratio
        a <- max(w.x)   # estimation of a
        u <- unif(n, 0, 1)      # independent Unif sample
        accept <- u <= (w.x/a)  # acceptance?
        if (sum(accept) > 0) {
                x.a <- x[accept]        # the accepted x
        }
        else {
                cat(" No acceptance at all!\n")
                break   # abnormal termination
        }
        list(x.accept = x.a)
}
```

1.3.1 Understanding acceptance sampling

Illustration Let $\pi \simeq N(0,1)$. Marked below is the acceptance region in case of (a) $\psi \simeq T_1$ (student-t with one d.f.), and (b) $\psi \simeq Unif(-5,5)$, i.e. an improper source function.





The joint sampling p.d.f. is $p(u,x) = p(u)p(x) = 1 \cdot \psi(x)$. We have, for any $x_0 \in (-\infty, \infty)$,

$$\begin{aligned} P[X \le x_0 | \text{Acceptance}] &= \frac{P[\text{Acceptance} \cap X \le x_0]}{P[\text{Acceptance}]} = \frac{\int_{-\infty}^{x_0} \{\int_0^{a(x)} p(u,x)du\}dx}{\int_{-\infty}^{\infty} \{\int_0^{a(x)} p(u,x)du\}dx} \\ &= \frac{\int_{-\infty}^{x_0} \{\pi(x)/[a\psi(x)]\}\psi(x)dx}{\int_{-\infty}^{\infty} \{\pi(x)/[a\psi(x)]\}\psi(x)dx} = \frac{P_\pi[X \le x_0]/a}{1/a} = P_\pi[X \le x_0]. \end{aligned}$$

Moreover, the unconditional probability of acceptance is given by $1/a$, and is often used to measure the efficiency of the source distribution. Generally speaking, a good source function ψ should (i) have heavier tails than the target function π, while (ii) mimic the shape of π.

Remark Indeed, constant a in the acceptance sampling can be substituted with any b such that $b\psi(x) > \pi(x)$ for almost all x. The resulting acceptance rate is $1/b$, which is less efficient than the choice of $a = \sup_x \pi(x)/\psi(x)$ now that $a < b$.

1.3.2 Example: Truncated Normal distribution

Since $X \sim N(0,1) \Rightarrow \mu + \sigma X \sim N(\mu, \sigma^2)$, we only need to consider the standard case.

Consider $N(0,1)$ truncated to the tail area $[\theta, \infty)$. Obviously, one could sample from the untruncated $N(0,1)$ and retain those that happen to fall within the specified region, which in fact amounts to acceptance sampling with $N(0,1)$ as the source distribution. The acceptance rate is $1 - \Phi(\theta)$, where $\Phi()$ denotes the cumulative distribution function (C.D.F.) of $N(0,1)$, which can be very inefficient. For instance, at $\theta = 5$, we have $P[X \geq 5; X \sim N(0,1)] = 2.87 \times 10^{-7}$.

A relocated Exponential distribution (sometimes called a two-parameter Exponential distribution) with parameter set at the censorship point, i.e. $X \sim \theta + Exp(\theta)$, is highly efficient for severely truncated normal distributions (Geweke, 1995)[1]. For instance, at $\theta = 5$, the acceptance rate is about 96.4%. The relevant Splus code has been listed below.

Splus transcript

```
right.tail.norm <- function(theta = 5, n = 10000)
{
        x <- theta + rexp(n, theta)      # Exponential source function
        p.e <- dexp(x - theta, theta)    # psi(x) of the source function
        p.n <- dnorm(x)/(1 - pnorm(theta)) # pi(x) of left-censored N(0,1)
        w.x <- p.n/p.e
        a <- max(p.n/p.e)
        u <- runif(n, 0, 1)
        accept <- u <= (w.x/a)
        m <- sum(accept)          # size of the acceptance sample
        list(a = a, Prob.accept = 1/a, Obs.Rate.accept = m/n)
}
```

Truncation of $N(0,1)$ to the tail area $[\theta_1, \theta_2]$ can be handled by the same relocated Exponential distribution, with the corresponding (to θ_2) additional truncation.

In case that truncation of $N(0,1)$ is made to a more central region, the Uniform distribution over that restricted region sometimes provides a good source function. For instance, it gives an acceptance rate of 96% for $N(0,1)$ truncated to $(0, 0.5)$.

[1] Geweke, J. (1995). *Monte Carlo Simulation and Numerical Integration.* Staff Report 192, Federal Reserve Bank of Minneapolis

1.3.3 Invariance property subjected to proportionality

In many applications of the Monte Carlo methods, it would be the case that the target distribution π is only known up to a constant of proportionality. Suppose, then,

$$\pi(x) = \frac{p(x)}{\int p(x)dx} = c_p^{-1} p(x),$$

where $p(x)$ is known but not c_p $(< \infty)$. We have,

$$\begin{aligned} & a_\pi = \sup_x \ \pi(x)/\psi(x) = [\sup_x \ p(x)/\psi(x)]/c_p = a_p/c_p \\ \Rightarrow \qquad & \pi/(a_\pi \psi) = \frac{p/c_p}{(a_p/c_p)\cdot\psi} = p/(a_p\psi). \end{aligned}$$

In other words, the acceptance sampling remains invariant when applied to (p, ψ).

Suppose, in addition, $\psi = q/c_q$, we have

$$\begin{aligned} & a_\pi = \sup_x \ \pi/\psi = (c_q/c_p)\cdot \sup_x \ p/q \\ \Rightarrow \qquad & \pi/(a_\pi\psi) = \frac{(c_q/c_p)\cdot(p/q)}{(c_q/c_p)\cdot \sup_x \ p/q} = \frac{p}{(\sup_x \ p/q)\cdot q}. \end{aligned}$$

In other words, it remains invariant when acceptance is calculated based on (p, q).

Notice that

$$E_\psi(\pi/\psi) = \int_\Omega (\frac{\pi}{\psi}) \ \psi \ dx = 1.$$

We have, for $w(x) = p(x)/q(x) \propto \pi(x)/\psi(x)$,

$$a_\pi = \sup_x \ \pi/\psi = \frac{\sup_x \pi/\psi}{E_\psi(\pi/\psi)} = \frac{\sup_x p/q}{E_\psi(p/q)},$$

so that a consistent estimator of a_π is given by

$$\hat{a}_\pi = \frac{\max_i \ p(x_i)/q(x_i)}{m^{-1}\sum_i p(x_i)/q(x_i)}. \tag{1.5}$$

Remark The observation above is useful in practice. To run acceptance sampling, neither do we need to know the standardizing constant c_p for π nor the Jacobian of transformation for X. Comparison between the estimated a_π and the actual acceptance percentage on a particular run provides us with an opportunity of checking the program.

1.3.4 Example: Genetic linkage model

Suppose multinomial data $Y = (Y_1, Y_2, Y_3, Y_4)$ with probability and likelihood, respectively,

$$p = (\frac{2+\theta}{4}, \frac{1-\theta}{4}, \frac{1-\theta}{4}, \frac{\theta}{4}) \quad \text{and} \quad L(\theta; y) \propto (2+\theta)^{y_1}(1-\theta)^{y_2+y_3}\theta^{y_4}$$

(Rao, 1973)[2], with finite integral for $\theta \in (0,1)$. Given y, we may rescale the likelihood as

$$L^*(\theta; y) = L(\theta; y)/L(\hat{\theta}; y),$$

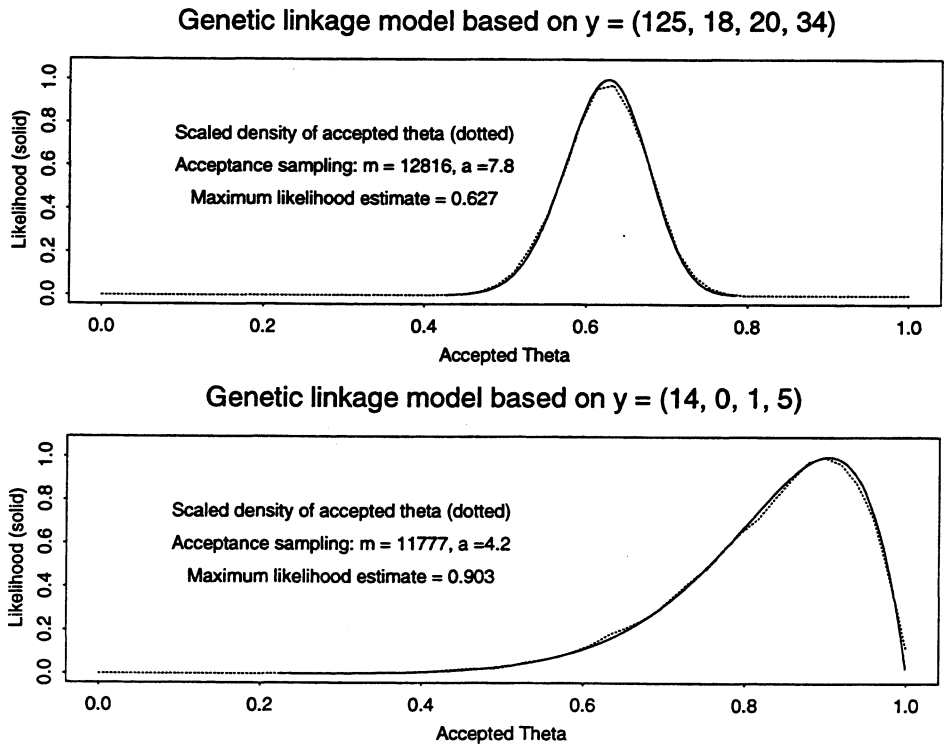
where $\hat{\theta}$ is the maximum likelihood estimate (m.l.e.). This gives us $L^* \in [0,1]$, and

$$\pi(\theta) = L^*(\theta) / \int L^*(\theta) d\theta.$$

Let $\psi \simeq Unif(0,1)$, i.e. $\psi = 1$. Together with our choice of L^* this implies that

$$a_\pi = \sup_\theta \pi/\psi = 1/(\int L^* d\theta) \quad \text{and} \quad P[\text{Acceptance}] = \int L^* d\theta = E_\psi(L^*).$$

For graphical inspection of the results, we estimate the density π based on the accepted draws, using the kernel density estimator provided in S-Plus (at window width 0.05). Since $\int \hat{\pi} d\theta = 1$ while $\int L^* d\theta = 1/a$, we scale $\hat{\pi}$ accordingly and plot $\hat{\pi}/\hat{a}$ against L^*. Simulation results and program, based on $y = (125, 18, 20, 34)$ and $(14, 0, 1, 5)$, are given below.



[2]Rao, C.R. (1973). *Linear Statistical Inference and Applications.* New York: Wiley.

Splus transcript

```
gene.s <- function(dat = 1, n = 50000, win = 0.05)
{
        y <- cbind(c(125, 18, 20, 34), c(14, 0, 1, 5))[, dat]
                        # which data set?
        theta <- c(0:1000)/1000 # precision of the numerical m.l.e.
        L <- ((2 + theta)^y[1]) * ((1 - theta)^(y[2] + y[3])) * (theta^y[4])
        L.max <- max(L)         # mode of the likelihood
        theta.hat <- theta[L == L.max]  # numerical m.l.e.
        theta <- runif(n, 0, 1) # UNIF(0,1) as source function
        L <- ((2 + theta)^y[1]) * ((1 - theta)^(y[2] + y[3])) * (theta^y[4])
        L <- L/L.max    # standardizing
        a <- max(L)/mean(L)     # the acceptance constant
        accept <- runif(n, 0, 1) <= L   # acceptance?
        theta <- theta[accept]  # acceptance sample
        L <- L[accept]
        m <- sum(accept)

        o.t <- order(theta)
        theta <- sort(theta)
        L <- L[o.t]     # arranging the sample in increasing order
        plot(theta, L, xlab = "Accepted Theta",
                ylab = "Likelihood (solid)", xlim = c(0, 1),
                ylim = c(0, 1.05), type = "l", lty = 1)
        d.t <- density(theta, width = win, from = 0, to = 1)
        lines(d.t$x, d.t$y/a, lty = 2)  # scaling so that integral = a
        text(0.3, 0.75, "Scaled density of accepted theta (dotted)")
        text(0.3, 0.6, paste("Acceptance sampling: m = ", m,
                ", a =", trunc(10 * a + 0.5)/10, sep = ""))
        text(0.3, 0.45, paste("Maximum likelihood estimate =", theta.hat))
        txt <- c("y = (125, 18, 20, 34)", "y = (14, 0, 1, 5)")[dat]
        title(paste("Genetic linkage model based on", txt))
}
```

1.3.5 Multivariate acceptance sampling

The univariate theory generalizes directly to the multivariate case. Usual candidate continuous source functions include multivariate Uniform, multinormal, and multivariate student-t.

Multivariate normal distribution. Denote by $N(0, \Sigma)$ the multinormal distribution with zero mean covariance matrix Σ. Let *upper-triangular matrix* C be the *Cholesky decomposition* of Σ, such that

$$C^T C = \Sigma.$$

Let $N(0, I)$ be with identity covariance matrix, we have

$$\mu + C^T X \sim N(\mu, \Sigma) \quad \text{where} \quad X \sim N(0, I).$$

In particular, acceptance sampling can be based on

$$q(x) = \prod_{j=1}^{K} \phi(x) \quad \propto \quad \psi(\mu + C^T x),$$

where $\phi(x)$ is the p.d.f. of $N(0, 1)$.

Splus transcript

```
r.m.norm <- function(m = 1, mu, sigma)
{
        d <- dim(sigma)[1]      # dimension of the distribution
        C <- chol((sigma + t(sigma))/2) # Cholesky decomposition
        x <- array(rnorm(m * d, 0, 1), c(d, m)) # m standard multinormal
        x.t <- mu + t(C) %*% x          # transformed sample
        q <- dnorm(x[1, ], 0, 1)        # proportional source density
        for(i in 2:d) {
                q <- q * dnorm(x[i, ], 0, 1)
        }
        list(x = x.t, pdf = q)
}
```

Multivariate student-t distribution Denote by χ^2_ν the χ^2-distribution with ν d.f.. Suppose

$$(X_1, \ldots, X_k) \overset{i.i.d.}{\sim} N(0,1) \quad \text{independent of} \quad (Y_1, \ldots, Y_k) \overset{i.i.d.}{\sim} \chi^2_\nu,$$

then $Z_i = X_i/\sqrt{Y_i/\nu}$, for $i = 1, \ldots, k$, form i.i.d. student T_ν-observations. We say that

$$\mu + C^T Z$$

has *location* μ, *scale* Σ and *degree of freedom* ν, where $\Sigma = C^T C$. In particular,

$$\psi(\mu + C^T z) \quad \propto \quad q(z) = \prod_{i=1}^{k} \gamma_\nu(z_i),$$

where $\gamma_\nu()$ is the p.d.f. of student-t distribution with ν degree of freedom.

Remark Standard parameterization of multivariate student-t distribution defines $Z = X/\sqrt{Y/\nu}$, where $X \sim N(\mu, \Sigma)$ and Y are independent χ^2_ν-variables, gives (μ, Σ, ν) a different interpretation.

Splus transcript

```
r.m.stud <- function(m = 1, mu, sigma, d.f = 3)
{
        d <- dim(sigma)[1]      # dimension of the distribution
        C <- chol((sigma + t(sigma))/2) # Cholesky decomposition
        x <- array(rt(m * d, d.f), c(d, m))     # m*d student-t with d.f
        x.t <- mu + t(C) %*% x  # transformed sample
        q <- dt(x[1,  ], d.f)   # proportional source density
        for(i in 2:d) {
                q <- q * dt(x[i,  ], d.f)
        }
        list(x = x.t, pdf = q)
}
```

1.3.6 Example: Logistic regression

Consider the following data set (Tanner, 1993, p 14)[3]:

Days of Radiotherapy (X)	21	24	25	26	28	31	33	34	35	37	43	49
Response (Y)	1	1	1	1	1	1	1	1	1	1	1	1
Days of Radiotherapy (X)	51	55	25	29	43	44	46	46	51	55	56	58
Response (Y)	1	1	0	0	0	0	0	0	0	0	0	0

and the logistic regression model, i.e.

$$\log(p_i) - \log(1 - p_i) = \alpha + \beta x_i,$$

where $p_i = P[Y_i = 1|x_i]$ for $i = 1, ..., 24$. Let $\theta = (\alpha, \beta)$, the likelihood is

$$L(\theta; y) \propto \prod_{i=1}^{24} p_i^{y_i}(1 - p_i)^{1-y_i} \qquad \text{where} \quad \hat{\theta} = (\hat{\alpha}, \hat{\beta}) = (3.819, -0.087).$$

To ensure finite $\int L d\theta$, we restrict ourselves to the parameter region

$$\Theta = \Theta_\alpha \times \Theta_\beta \quad \text{where} \quad \Theta_\alpha = (-1, 9) \quad \text{and} \quad \Theta_\beta = (-0.25, 0.05).$$

Three source functions are made optional:

- $Unif(-1, 9) \times Unif(-0.25, 0.05)$,
- bivariate normal $N(\hat{\theta}, k\hat{\Sigma})$ where k is a tuning parameter and $\hat{\Sigma}$ the inverse of the observed information, also known as the *observed formation*,
- bivariate student-t with location μ, scale $k\Sigma$ and d.f. ν.

Notice that truncation of the sample to Θ is necessary except with the Uniform distribution. However, this changes only the density of the source function proportionally.

To visualize the results, we compare the profile likelihood with the marginal likelihood based on the accepted sample. The profile likelihood of e.g. α is defined as

$$L_P(\alpha; y) = \max_\beta L(\alpha, \beta; y).$$

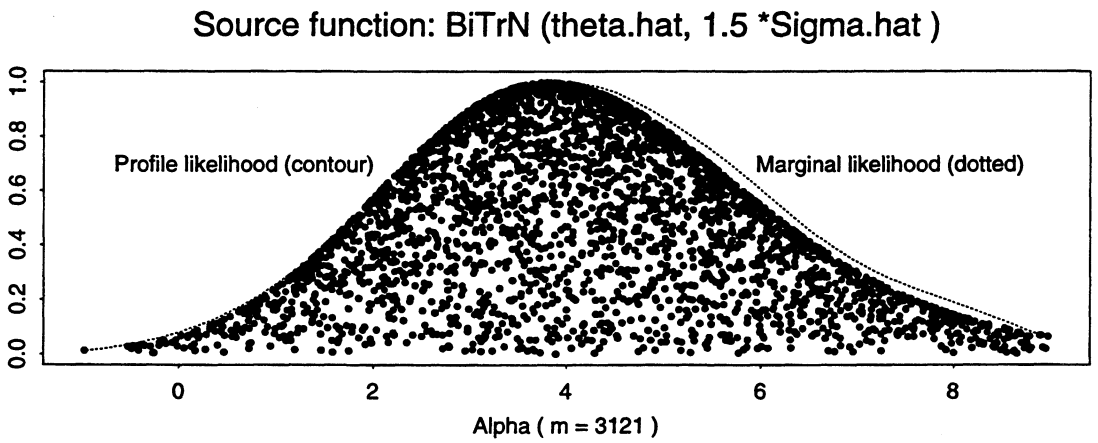
Whereas the α marginal of the likelihood is defined as

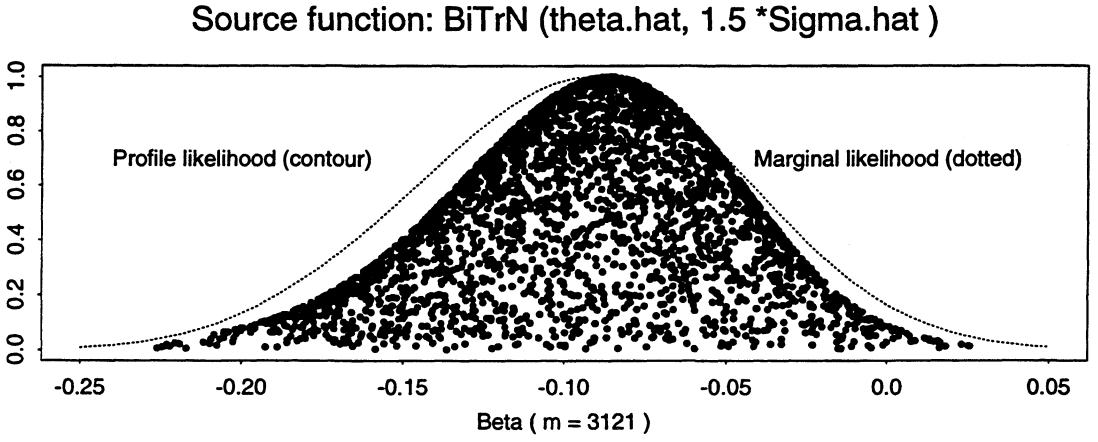
$$L_M(\alpha; y) = \int L(\alpha, \beta; y) d\beta.$$

[3]Tanner, M.A. (1993). *Tools for Statistical Inference. (2nd Edition).* Springler-Verlag.

In the first case, we may plot, say, the accepted α against their corresponding $L(\alpha, \beta; y)$ regardless of β. The contour of the plotted area converges to the profile likelihood of α, provided the sample covers everywhere in Θ as it tends to infinity. To get the α marginal of the likelihood, we simply estimate the marginal density of α based on the accepted pairs of parameters regardless of β. Finally, we plot the scaled $L_M/\max(L_M)$ against $L_P/\max(L_P)$, i.e. the contour of $L(\theta; y)/L(\hat{\theta}; y)$. The result may be influenced by the choice of the window width for L_M. Nevertheless, they confirmed to the basic impression that the marginals of the likelihood are somewhat flatter than the profiles of the likelihood.

Remark Starting with $m = 100$ and estimating a_π as explained before, we found acceptance rate at about 5% for the Uniform source function. The acceptance rate was best when the tuning parameter was at $k = 1.5$ for the Binormal distribution, among $k = 0.5,\ 1,\ 1.5,\ 2,\ 2.5,\ 3$. Slightly higher acceptance rate can be found for bivariate student-t at certain combinations of the parameters. Figures below have been obtained using truncated bivariate $N(\hat{\theta}, 1.5\hat{\Sigma})$ as $\psi(\theta)$ with initial sample size $n = 5000$.





Splus transcript

```
logit.s <- function(n = 1000, sampler = 2, k = 1.5, d.f = 3,
                a.lim = c(-1, 9), b.lim = c( -0.25, 0.05),
                theta = c(3.819, -0.087), win = c(2, 0.1))
{
        x <- c(21, 24:26, 28, 31, 33:35, 37, 43, 49, 51, 55, 25,
                29, 43, 44, 46, 46, 51, 55, 56, 58)
        y <- c(rep(1, 14), rep(0, 10))  # the data
        eta <- theta[1] + theta[2] * x
        p <- exp(eta)/(1 + exp(eta))
        L.hat <- exp(sum(y * eta + log(1 - p)))
        w <- p * (1 - p)
        j <- c(sum(w), sum(x * w), sum(x * w), sum(x^2 * w))
        j <- solve(array(j, c(2, 2)))   # the observed formation
        j <- (j + t(j))/2               # securing symmetry

# Unif*Unif
        if(sampler == 1) {
                z <- runif(n, a.lim[1], a.lim[2])
                z <- rbind(z, runif(n, b.lim[1], b.lim[2]))
                d.z <- rep(1/(diff(a.lim) * diff(b.lim)), n)
        }
# bivariate N(theta,k*Sigma)
        if(sampler == 2) {
                A <- chol(k * j)
                z.0 <- array(rnorm(2 * n), c(2, n))
                z <- theta + t(A) %*% z.0
                d.z <- dnorm(z.0[1,  ]) * dnorm(z.0[2,  ])
        }
# bivariate student-t
        if(sampler == 3) {
                A <- chol(k * j)
                z.0 <- array(rt(2 * n, d.f), c(2, n))
                z <- theta + t(A) %*% z.0
                d.z <- dt(z.0[1,  ], d.f) * dt(z.0[2,  ], d.f)
        }

        idx <- z[1,  ] <= a.lim[2] & z[1,  ] >= a.lim[1]
        idx <- idx & z[2,  ] <= b.lim[2] & z[2,  ] >= b.lim[1]
        z <- z[, idx]
        d.z <- d.z[idx]
        n <- sum(idx)   # sample after possible truncation
```

```
        L <- 1  # sample likelihood
        for(i in 1:length(y)) {
                eta <- z[1,  ] + z[2,  ] * x[i]
                p <- exp(eta)/(1 + exp(eta))
                L <- L * (p^y[i]) * ((1 - p)^(1 - y[i]))
        }
        L <- L/L.hat    # standardizing

        w <- L/d.z
        a <- max(w)/mean(w)
        accept <- runif(n, 0, 1) <= w/max(w)
        z <- z[, accept]
        L <- L[accept]
        m <- sum(accept)        # acceptance sample

        close.screen(all = T)
        split.screen(figs = c(2, 1))    # graphical display
        txt <- c("Unif * Unif", paste("BiTrN (theta.hat,", k, "*Sigma.hat )"),
                paste("Student-t ( location", "theta.hat, scale", k,
                "* Sigma.hat and d.f.", d.f, ")"))
        x.b <- paste(c("Alpha", "Beta"), "( m =", m, ")")
        x.lim <- rbind(a.lim, b.lim)
        for(i in 1:2) {
                screen(i)
                plot(z[i,  ], L, xlab = x.b[i], ylab = "", xlim = x.lim[i,  ])
                title(paste("Source function:", txt[sampler]))
                text(sum(x.lim[i,  ] * c(5, 1))/6, 0.7,
                        "Profile likelihood (contour)")
                text(sum(x.lim[i,  ] * c(1, 5))/6, 0.7,
                        "Marginal likelihood (dotted)")
                d.t <- density(z[i,  ], width = win[i], from = x.lim[i, 1],
                        to = x.lim[i, 2])
                points(d.t$x, d.t$y/max(d.t$y), type = "l", lty = 2)
        }

        list(a = a, Rate.accept = m/n)
}
```

1.4 Importance sampling

Let $\psi(x)$ be an *importance sampling density*, define the *importance weights* to be

$$w(x_i) = \pi(x_i)/\psi(x_i) \qquad \text{where} \quad X_1, ..., X_m \overset{i.i.d.}{\sim} \psi(x).$$

The corresponding *(weighted) importance Monte Carlo* is given by

$$I_m = \{\sum_{i=1}^{m} f(x_i)w(x_i)\}/\{\sum_{i=1}^{m} w(x_i)\}; \tag{1.6}$$

whereas the *simple importance Monte Carlo* is

$$I_0 = \frac{1}{m}\{\sum_{i=1}^{m} f(x_i)w(x_i)\}. \tag{1.7}$$

By the strong law of large numbers, we have

$$I_m \overset{a.s.}{\to} I \qquad \text{and} \qquad I_0 \overset{a.s.}{\to} I.$$

Moreover, if

$$E_\pi(w) = \int \pi^2(x)/\psi(x)dx < \infty \quad \text{and} \quad E_\pi(f^2w) = \int f^2(x)\pi^2(x)/\psi(x)dx < \infty,$$

then

$$\sqrt{m}(I_m - I) \overset{D}{\to} N(0, \sigma^2) \qquad \text{where} \quad \sigma^2 = E_\pi\{(f-I)^2w\}, \tag{1.8}$$

and

$$\sqrt{m}(I_0 - I) \overset{D}{\to} N(0, \sigma_0^2) \qquad \text{where} \quad \sigma_0^2 = Var_\psi(fw) = E_\pi(f^2w) - I^2. \tag{1.9}$$

Based on the same sample, the corresponding Monte Carlo estimates can be given as

$$\hat{\sigma}^2 = \{\sum_{i=1}^{m}[f(x_i) - I_m]^2w^2(x_i)\}/\{\sum_{i=1}^{m} w(x_i)\}.$$

and

$$\hat{\sigma}_0^2 = \frac{1}{m}\sum_{i=1}^{m}[f(x_i)w(x_i) - I_0]^2.$$

Generic Splus code for importance sampling

```
pi.x <- function(x)
{
        calculation of the target p.d.f. for the sample
}



psi.x <- function(x)
{
        calculation of the source p.d.f. for the sample
}



f.x <- function(x)
{
        evaluation of f(x)
}



import.mc <- function(n = 10000)
{
        generate independent importance sample of size n
        w <- pi.x(x)/psi.x(x)    # importance weights
        f <- f.x(x)              # f(x)

        i.m <- sum(w * f)/sum(w)         # weighted importance MC
        sigma <- sqrt(sum((f - i.m)^2 * w^2)/sum(w))

        i.0 <- mean(w * f)       # simple importance MC
        sigma.0 <- sqrt(var(f * w))

        list(I.m = i.m, sigma = sigma, I.0 = i.0, sigma.0 = sigma.0)
}
```

1.4.1 Understanding importance sampling and invariance property

Instead of using an acceptance/rejection mechanism to obtain an i.i.d. sample from $\pi(x)$, the importance sampling weights *all* the draws from the source distribution $\psi(x)$ to obtain a convergent approximation.

Remark If we think of $w(x_i)$ as the inverse of the inclusion probability, as in the case of sampling survey, then I_0 corresponds to $\hat{f}/N$, and I_m to $\hat{f}/\hat{N}$ even when N (i.e. the size of the population) is known. We therefore refer to I_0 as the simple importance Monte Carlo, and I_m the weighted one.

The simple importance Monte Carlo can be motivated by the following identity,

$$E_\pi(f) = \int f\pi dx = \int f(\pi/\psi)\psi dx = E_\psi(fw),$$

since I_0 is identical with I_s (1.2) applied to $E_\psi(fw)$. Whereas in the weighted importance Monte Carlo, also the dominate one in practice, a further identity is introduced, i.e.

$$E_\pi(f) = E_\psi(fw)/1 = E_\psi(fw)/E_\psi(w).$$

The simple Monte Carlo I_s (1.2) is now applied to both terms on the right-hand side, and the resulting ratio gives us the weighted Monte Carlo (1.6).

Both methods use all the sample generated. However, while the simple importance Monte Carlo is unbiased, the same is not true of the weighted importance Monte Carlo. Neither does strict inequality hold between their respective variances. The weighted importance Monte Carlo is invariant w.r.t. proportional transformation of π or/and ψ, since

$$w = p/q \quad \propto \quad \pi/\psi \qquad \Rightarrow \qquad I_m = \{\sum_i (f_i p_i/q_i)\}/\{\sum_i p_i/q_i\}.$$

it is also clear that the same may not be said of the simple importance Monte Carlo I_0.

Moreover, an invariant Monte Carlo estimator of the variance of the weighted importance Monte Carlo can be given as

$$\hat{\sigma}^2 = \frac{m \cdot \sum_i [f(x_i) - I_m]^2 w^2(x_i)}{[\sum_i w(x_i)]^2}. \tag{1.10}$$

To derive this from the estimator in case of $w = \pi/\psi$, we notice that $\sum_i w_i/m \xrightarrow{a.s.} 1$. However, the numerator and the denominator are now both proportional to w^2.

1.4.2 The central limiting theorem

The convergence of the CLT is essential in application. The δ-method gives, asymptotically,

$$\begin{aligned} Var_\psi(\frac{\sum_{i=1}^m f_i w_i}{\sum_{i=1}^m w_i}) &= \frac{Var_\psi(\sum_i f_i w_i)}{E_\psi^2(\sum_i w_i)} + \frac{E_\psi^2(f_i w_i)}{E_\psi^4(\sum_i w_i)} Var_\psi(\sum_i w_i) \\ &\qquad - 2\frac{E_\psi(\sum_i f_i w_i)}{E_\psi^3(\sum_i w_i)} \cdot \sum_i Cov_\psi(f_i w_i, w_i) \\ &= \frac{m \cdot Var_\psi(fw)}{(m \cdot 1)^2} + \frac{(m \cdot I)^2}{(m \cdot 1)^4} \cdot m \cdot Var_\psi(w) \\ &\qquad - 2\frac{m \cdot I}{(m \cdot 1)^3} \cdot m \cdot Cov_\psi(fw, w) \\ &= \frac{1}{m}\{[E_\pi(f^2 w) - I^2] + I^2 \cdot [E_\pi(w) - 1] \\ &\qquad - 2I \cdot [E_\pi(fw) - I]\} \\ &= \frac{1}{m} E_\pi(f^2 \cdot w - 2If \cdot w + I^2 \cdot w) \\ &= \frac{1}{m} E_\pi\{(f - I)^2 w\} \\ &= \frac{1}{m}\sigma^2, \end{aligned}$$

since

$$\begin{aligned} Var_\psi(fw) &= \int (f\pi/\psi)^2 \psi dx - [\int f(\pi/\psi)\psi dx]^2 \\ &= \int (f^2 w)\pi dx - I^2 \\ &= E_\pi(f^2 w) - I^2, \end{aligned}$$

and

$$\begin{aligned} Var_\psi(w) &= \int (\pi/\psi)^2 \psi dx - [\int (\pi/\psi)\psi dx]^2 \\ &= E_\pi(w) - 1, \end{aligned}$$

and, finally,

$$\begin{aligned} Cov_\psi(fw, w) &= E_\psi(fw^2) - E_\psi(fw)E_\psi(w) \\ &= \int f(\pi/\psi)^2 \psi - I \cdot 1 \\ &= E_\pi(fw) - I. \end{aligned}$$

1.4.3 Relative numerical efficiency (RNE)

Recall

$$\sigma_\pi^2 = Var_\pi(f) = \int (f-I)^2 \pi dx = \int (f-I)^2 w \psi dx = E_\psi\{(f-I)^2 w\}.$$

Averaging of $(f-I)^2 w$ is carried out w.r.t. $\psi(x)$, compared to $\pi(x)$ in the case of σ^2. We obtain, thus, Monte Carlo

$$\hat{\sigma}_\pi^2 = \sum_{i=1}^{m} [f(x_i) - I_m]^2 w(x_i)/m$$

and/or its invariant version, i.e.

$$\hat{\sigma}_\pi^2 = \{\sum_i [f(x_i) - I_m]^2 w(x_i)\}/\{\sum_i w(x_i)\}. \tag{1.11}$$

as a by-product of the importance sampling.

In particular,

$$w = 1 \quad \Leftrightarrow \quad \psi = \pi \qquad \Rightarrow \qquad \sigma_\pi^2 = \sigma^2$$

i.e. the difference between σ_π^2 and σ^2 is caused by the fact that the sample is not directly generated from $\pi(x)$.

Define the *relative numerical efficiency* of the weighted importance sampling density ψ (against the direct sampling density π) to be

$$RNE = \sigma_\pi^2/\sigma^2.$$

RNE less than 0.1, certainly less than 0.01, indicates poor efficiency, and possible failure of the underlying convergence conditions.

Two properties of ψ are specially helpful for improvement of RNE, (a) it has thicker tails than those of π, and (b) its shape closely mimic that of π, though it is difficult to formulate exact measures of, or precise balance between, them.

In addition, we may define the RNE of the weighted importance Monte Carlo against the simple importance Monte Carlo as

$$RNE = \sigma_0^2/\sigma^2.$$

1.4.4 Example: Relative numerical efficiency for target $N(0,1)$

Let $\pi(x) \simeq N(0,1)$ and $\psi(x)$ be student-T_ν where ν is the degree of freedom. Random $X \sim T_\nu$ has zero mean with variance $\nu/(\nu-2)$ for $\nu > 2$ — the variance does not exist for $\nu = 1$, and its tails are heavier than those of the normal distribution: while T_1 has the thickest tails, T_∞ reduces to $N(0,1)$, giving the closest mimicry of the shape of $N(0,1)$.

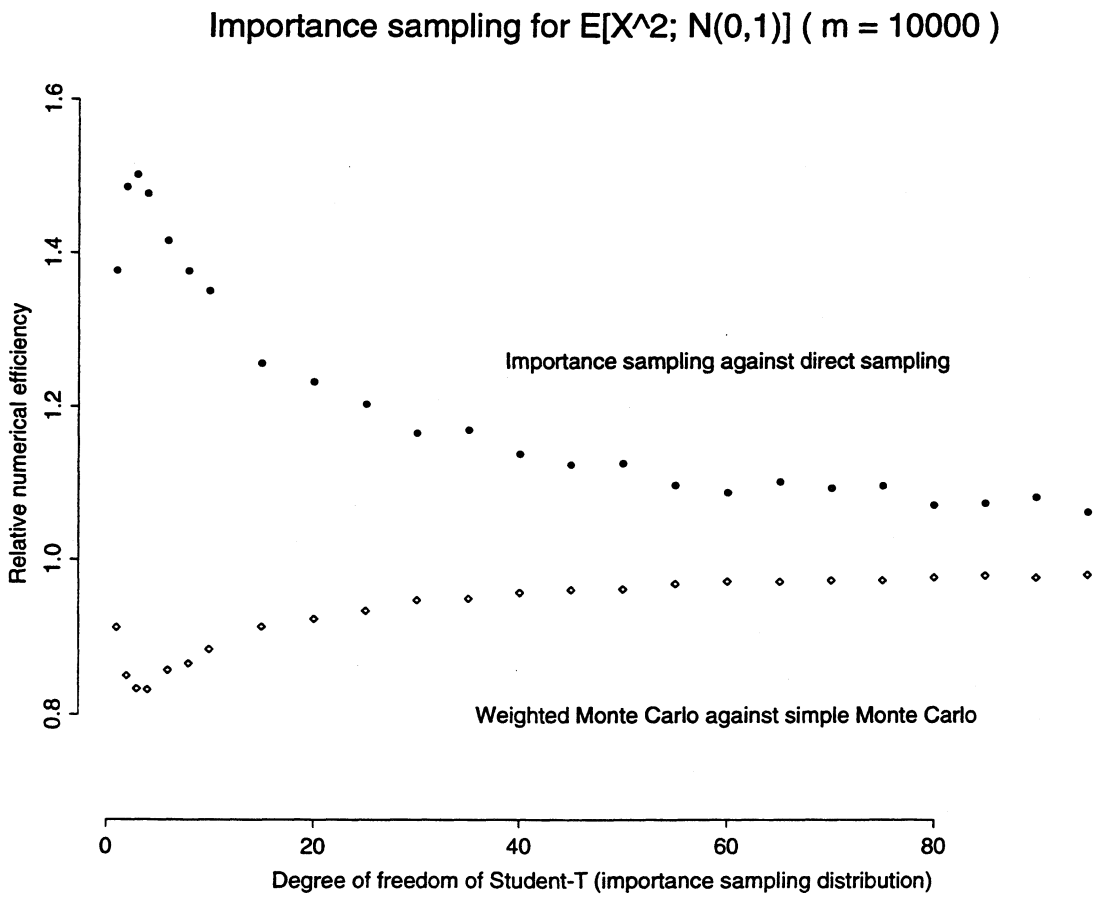
Let $f(x) = x^2$. Direct sampling (from π) is possible so that we may estimate the variance of (a) the simple Monte Carlo (1.2) based on direct sampling, i.e. σ_π^2, (b) the simple importance (1.7), and (c) the weighted importance (1.6).

Remark Although $\sigma_\pi^2 = 2$ is known here, RNE becomes more stable when it is calculated based on its estimates.

The results suggest uniformly that

$$\sigma_0^2 < \sigma^2 < \sigma_\pi^2.$$

Maximum efficiency was reached by I_0 with T_3, where $\sigma_0^2 \approx \sigma_\pi^2/2$. It is also interesting to notice that thick tails alone raised the RNE of T_1 above that of T_d with large ν, although the latters more and more approach the shape of $N(0,1)$.



Splus transcript

```
rne.t <- function(m = 10000, d = c(1:4, 4 + c(1:3) * 2, 10 + c(1:17) * 5))
{
        B <- length(d)                  # number of d.f. of simulation
        btp <- array(0, c(4, B))        # tabulating the results
        dimnames(btp) <- list(c("a", "s2_pi", "s2_0", "s2_m"), d)

        for(i in 1:B) {
                d.f <- d[i]     # degree of freedom
                x <- rt(m, d.f) # importance sample
                f <- x^2
                w <- dnorm(x)/dt(x, d.f)        # importance weights
                btp["a", i] <- max(w)   # acceptance rates
                i.m <- sum(f * w)/sum(w)
                btp["s2_m", i] <- sum((f - i.m)^2 * w^2)/sum(w)
                btp["s2_pi", i] <- mean((f - i.m)^2 * w)
                btp["s2_0", i] <- var(f * w)
        }

        RNE <- btp["s2_pi", ]/btp["s2_m", ]   # I_m against I_s
        plot(d, RNE, bty = "n", xlab = paste("Degree of freedom of",
                "Student-T (importance sampling distribution)"), ylab =
                "Relative numerical efficiency", ylim = c(0.7, 1.6))
        text(60, 1.26, "Importance sampling against direct sampling")
        RNE <- btp["s2_0", ]/btp["s2_m", ]    # I_m against I_0
        points(d, RNE, pch = 5)
        text(60, 0.8, "Weighted Monte Carlo against simple Monte Carlo")
        title(paste("Importance sampling for E[X^2; N(0,1)] ( m =", m, ")"))

        list(tab.result = trunc(1000 * btp + 0.5)/1000)
}
```

1.4.5 Acceptance sampling or importance sampling?

Acceptance sampling and importance sampling are clearly related. The variance of the acceptance sampling is $\sigma_\pi^2 = \int (f - I)^2 \pi dx$ if we count only the accepted draws, and $a\sigma_\pi^2$ if we count all the draws, where $1/a$ is the probability of acceptance. The variance of the importance sampling is $\sigma^2 = \int (f - I)^2 w \pi dx$ where $w \leq a$, so that $\sigma^2 \leq a\sigma_\pi^2$, in which sense the importance sampling is more efficient unless it is very costly to evaluate $f(x)$. However, the exact sample generated by means of acceptance sampling opens up other inferential possibilities which are not available with importance sampling. In general, essential to both methods is to find good/working $\psi(x)$, than the choice between them.

Remark While $E_\psi(I_s) = I$ for I_s based on acceptance samples, so is I_0 (1.7) unbiased. The weighted importance Monte Carlo (1.6) is however biased in general.

Remark Given $X = x \sim \psi$, the corresponding $f(x)$ would be down-weighted by importance sampling if $w = \pi/\psi$ is small, which also implies that x would have a small probability of acceptance since $\alpha = \pi/(a\psi)$ is small as well. Similar compatibility holds in case of large $w(x)$.

Example The following Splus transcript compares acceptance and (weighted) importance sampling, where $N(0,1)$ left-censored at $\theta = 5$ is the target distribution, and $\psi \simeq \theta + Exp(\theta)$, and $f(x) = x$. The results indicate that the weighted importance sampling is slightly more efficient in this case.

```
rne.accept.import <- function(theta = 5, n = 10000, prop = F)
{
        x <- theta + rexp(n, theta)
        f <- x
        pi.x <- dnorm(x)        # with or without standardizing constant
        if(!prop) {
                pi.x <- pi.x/(1 - pnorm(theta))
        }
        psi.x <- dexp(x - theta, theta)
        w <- pi.x/psi.x
        a <- max(w)/mean(w)
        accept <- runif(n, 0, 1) <= w/max(w)
        f.a <- x[accept]        # f(x) based on the acceptance sample
        i.s <- mean(f.a)        # acceptance Monte Carlo
        i.m <- sum(f * w)/sum(w)        # weighted importance Monte Carlo
        sigma <- sqrt(n * sum((f - i.m)^2 * w^2))/sum(w)
        s.pi <- sqrt(sum((f - i.m)^2 * w)/sum(w))       # importance MC
        s.s <- sqrt(var(f.a))   # simple MC | acceptance
        list(MC.import = i.m, MC.accept = i.s, SD.import = sigma, SD.accept
                = c(s.s, s.pi), accept.rate = c(1/a, sum(accept)/n))
}
```

1.4.6 Combined importance sampling

It is possible to combine acceptance sampling with importance sampling — the resulting method can be referred to as the *combined importance sampling*.

Suppose target π, and importance sampling density ψ, and some constant $c > 0$. Define the *combined importance weight* as

$$w(x) = \begin{cases} \pi(x)/[c \cdot \psi(x)] & \text{if } \pi/\psi \geq c \\ \begin{cases} 1 \text{ with probability } \pi/(c\psi) \\ 0 \text{ otherwise} \end{cases} & \text{if } \pi/\psi < c \end{cases}$$

The resulting *combined importance Monte Carlo* takes the weighted form, i.e.

$$I_c = \{\sum_i f(x_i)w(x_i)\}/\{\sum_i w(x_i)\} \quad \xrightarrow{a.s.} \quad I = E_\pi(f).$$

Let $B = \{x; \pi(x)/\psi(x) \geq c\}$, we have

$$\begin{aligned} E_\psi(w) &= \int_B \frac{\pi}{c\psi}\psi dx + \int_{B^c} 1 \cdot \frac{\pi}{c\psi}\psi da + \int_{B^c} 0 \cdot (1 - \frac{\pi}{c\psi})\psi dx \\ &= P_\pi(B)/c + P_\pi(B^c)/c \\ &= 1/c, \end{aligned}$$

and

$$\begin{aligned} E_\psi(fw) &= \int_B f\frac{\pi}{c\psi}\psi dx + \int_{B^c} f \cdot \frac{\pi}{c\psi}\psi da + 0 \\ &= c^{-1}\int_{B \cup B^c} f\pi dx \\ &= I/c. \end{aligned}$$

For any given problem, there is a value of c which minimizes the variance of the combined Monte Carlo. See Müller (1991, Chapter 2)[4] for more details.

[4]Müller, P. (1991). *Numerical Integration in Bayesian Analysis*. PH.D. thesis, Purdue University.

1.5 Variance reduction

In any independence Monte Carlo methods, a single draw from ψ can be replaced by the mean of, say, k identically, but not independently distributed draws. They are numerous ways in which these can be set up, so that a reduction in the variance of the resulting Monte Carlo can be achieved. The most common variance reduction methods include antithetic variables (Geweke, 1988)[5], systematic sampling (McGrath, 1970)[6], control variables (Hammersley and Handscomb, 1964)[7], *etc.*.

Example (Antithetic importance Monte Carlo)
Suppose target π with mean μ_π, and $X \sim \psi$, define the *antithesis* of x as

$$x' = \mu_\pi - (x - \mu_\pi) = 2\mu_\pi - x.$$

The antithetic (weighted) importance Monte Carlo of $I = E_\pi(f)$ is given as, for $f_i = f(x_i)$ and $f_i' = f(x_i')$ and so on,

$$\begin{aligned} I_2 &= \frac{\sum_i f_i w_i + f_i' w_i'}{\sum_i w_i + w_i'} \qquad \text{where } w_i = \pi_i/\psi_i \\ &= \frac{\sum_i \frac{f_i w_i + f_i' w_i'}{w_i + w_i'}(w_i + w_i')/2}{\sum_i (w_i + w_i')/2} \\ &= \frac{\sum_i g_i v_i}{\sum_i v_i} \qquad \xrightarrow{a.s.} \quad I, \end{aligned}$$

where $g_i = (f_i w_i + f_i' w_i')/(w_i + w_i')$ and $v_i = (w_i + w_i')/2$. It can be shown that the variance of the I_2 is given by

$$Var(I_2) = \frac{1}{m} E_\pi\{(g - I)^2 w\},$$

and so on. Typically, this reduces the variance of the standard importance Monte Carlo provided π is more or less symmetric.

[5]Geweke, J. (1988). Antithetic acceleration of Monte Carlo integration in Bayesian inference. *Journal of Econometrics,* **38***: 73-89.*

[6]McGrath, E.I. (1970). *Fundamentals of Operations Research.* San Fransisco: West Coast University Press.

[7]Hammersley, J.M. and Handscomb, D.C. (1964). *Monte Carlo Methods.* London: Methuen and Company.

Chapter 2

Markov chain

2.1 Introduction

In this chapter we review some of the theory of Markov chains. Our aim is to provide the necessary background for understanding the methodology of Markov chain Monte Carlo in the next chapter.

Two results are above all important. The first of them is formulated in terms of the Ergodic Theorem, which ensures the desired convergence of Monte Carlo based on a Markov chain sample. The other one, expressed as the central limit theorem, measures the precision and the efficiency of basing Monte Carlo on such a Markov chain sample.

The Markov chain sample being dependent, the techniques by which these results can be derived are different from those of the previous chapter, which dealt with independent samples. However, efforts have been made to keep the presentation at a minimum technical level. Details which are not absolutely necessary have been excluded.

Having explained the definition of a Markov chain, we introduce increasingly stronger properties: irreducibility, recurrence, invariant distribution and positive recurrence, and finally reversibility. It is important to understand the difference between these properties, and which of them is needed for which type of convergence.

2.2 Markov chain

2.2.1 Definition

Denote by

$$(X_n)_{n\geq 0} \qquad \text{where} \quad X_n \in \Omega \quad \text{and} \quad n = 0, 1, 2, ...,$$

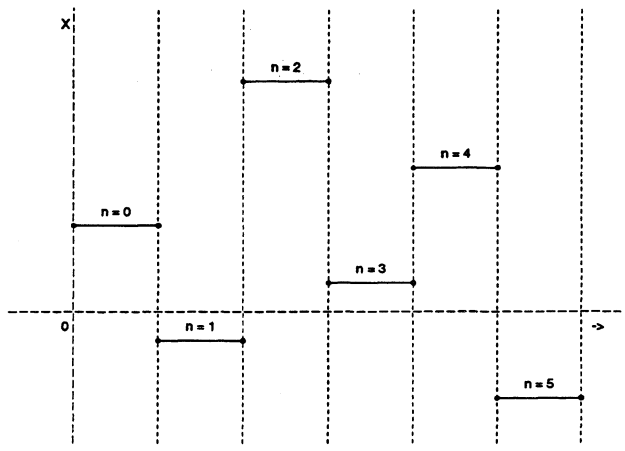
a *discrete-time* random process taking values from *state-space* Ω. Denote by ξ the distribution of X_0, i.e. the *initial distribution*. Denote by $P = p(x_i, x_{i+1})$ the probability function governing *transition* from x_i to x_{i+1} for $0 \leq i < \infty$, i.e. depending only on x_i, but not x_j for $j < i$. In this way the random process generated by ξ and P is a *Markov chain*, denoted by

$$(X_n)_{n\geq 0} \quad \sim \quad M(\xi, P).$$

More compactly, we may write

$$\begin{aligned}(X_n)_{n\geq 0} \sim M(\xi, P) \qquad &\Leftrightarrow \qquad \forall 0 < N < \infty, \\ P[(X_0, X_1, ..., X_N)] &= P[X_0]P[X_1|x_0]P[X_2|x_1] \cdots P[X_N|x_{N-1}] \\ &= \xi(x_0)p(x_0, x_1)p(x_1, x_2) \cdots p(x_{N-1}, x_N).\end{aligned}$$

Illustration The beginning of a discrete-time Markov chain:



Remark The transition P is constant over time, for which reason the Markov chain is said to be *time homogeneous*. Notice also the constant interval/period of time before the current state of the chain is changed: with continuous-time Markov chain, the time the random process spends at each state is independent, and exponentially distributed.

2.2.2 Weak and strong Markov Property

A Markov chain with fixed initial state x_0 has degenerate initial distribution, i.e.

$$\xi(x_0) = \delta_{x_0} = \begin{cases} 1 & \text{if } X_0 = x_0 \\ 0 & \text{otherwise} \end{cases}$$

in which case it is denoted by

$$M(\delta_{x_0}, P).$$

(Weak) Markov property (WMP)

$$(X_n)_{n\geq 0} \sim M(\xi, P) \qquad \Rightarrow \qquad \forall m < \infty,\ (X_{m+n})_{n\geq 0} \sim M(\delta_{x_m}, P).$$

Remark Given $X_m = x_m$ and $N < \infty$, $P[(X_{m+1}, ..., X_{m+N})|x_m]$ admits the required factorization by definition for *any* sequence of $(x_0, ..., x_{m-1}, x_m)$. According to the weak Markov property, therefore, what happens to a Markov chain from any *fixed* time in 'future' on, is entirely determined by the state at that point, and has nothing to do with whatever has happened before that.

Stopping time A *stopping time* τ is a random variable, for $\tau \in \{0, 1, 2, ..., \infty\}$, such that $\tau = m$ depends on $(X_0, ..., X_m)$ alone.

Example The first passage time of $x \in \Omega$, i.e. $\inf\{m \geq 1; X_m = x\}$, is a stopping time. So is the first recurrence time of $x \in \Omega$, i.e. $\inf\{m \geq 1; \exists n < m, X_n = X_m = x\}$. The last exit time of x, i.e. $\sup\{m \geq 0; X_m = x\}$, is in general not a stopping time.

Strong Markov property (SMP)

$$(X_n)_{n\geq 0} \sim M(\xi, P) \qquad \Rightarrow \qquad \forall \tau < \infty,\ (X_{\tau+n})_{n\geq 0} \sim M(\delta_{x_\tau}, P).$$

Remark Notice that exactly when τ occurs can not be known in advance. In contrast, it happens precisely at step m with WMP.

Remark This lack of memory of Markov chains, expressed here in terms of the WMP and the SMP, is a characteristic of particular importance. In fact, the Ergodic theorem later on ensures that, under mild regularity conditions, a Markov chain will eventually 'forget' all about its initial state, and enter into a state of equilibrium.

2.3 Discrete state-space theory

2.3.1 Some elementary calculations

Transition matrix The transition P of an $M(\xi, P)$ with (countable) discrete state-space Ω is a *stochastic matrix*, whose (i,j)-th element is given by

$$p_{ij} = P[X_{n+1} = j | x_n = i] = p(i,j)$$

for $i, j \in \Omega$. In particular, $\sum_j p_{ij} = 1$, which is the defining property of a stochastic matrix.

Transition after transition For any $n > 0$, we write

$$p_{ij}(n) = P[X_n = j | X_0 = i] = P_i[X_n = j],$$

which is given by the (i,j)-th element of P^n. In particular, P is a square matrix given finite Ω. Denote by λ_1, λ_2, ... its eigenvalues, such that

$$P = UD(\lambda)U^{-1}$$

i.e. diagonalizable, where $D(\lambda)$ is the diagonal matrix defined by λ. Since

$$P^n = UD(\lambda)U^{-1}UD(\lambda)U^{-1}\cdots UD(\lambda)U^{-1} = UD(\lambda^n)U^{-1},$$

λ^n are eigenvalues of P^n. Thus, if λ are all real and distinct, then, $\exists$ constants a such that

$$p_{ij}(n) = (P^n)_{i,j} = \sum_k a_k \lambda_k^n.$$

Hitting-time Random variable *hitting-time* of $j \in \Omega$ is defined as

$$H(j) = \inf\{m \geq 0; X_m = j\}.$$

The *hitting probabilities*, denoted by $h_i^j = P_i[H(j) < \infty]$ and $h_j^j = 1$, are given by the minimal non-negative solution to

$$h_i^j = \sum_{k \in \Omega} p_{ik} h_k^j$$

Whereas the *expected hitting-time*, denoted by $e_i^j = E_i[H(j)]$ and $e_j^j = 0$, are given by the minimal non-negative solution to

$$e_i^j = \sum_{k \in \Omega} p_{ik}(1 + e_k^j) = 1 + \sum_{k \in \Omega} p_{ik} e_k^j.$$

2.3.2 Irreducibility

For all $i \neq j \in \Omega$, i *leads to* j if $p_{ij}(n) > 0$ for some n, denoted by

$$i \to j.$$

Moreover, i *communicates with* j if $i \to j$ and $j \to i$, denoted by

$$i \leftrightarrow j.$$

Equivalent relation $\leftrightarrow$ divides Ω into disjoint *communicating classes.* A Markov chain, indeed P, with discrete state-space is *irreducible* if Ω consists of one single communicating class:

$$\forall\, i,\ j \in \Omega \quad \Rightarrow \quad i \leftrightarrow j.$$

Remark Irreducibility implies that the initial distribution ξ has no bearing on the convergent states of the Markov chain. Every state in Ω can be reached by the chain no matter where it starts.

On the other hand, a class C is *closed* if

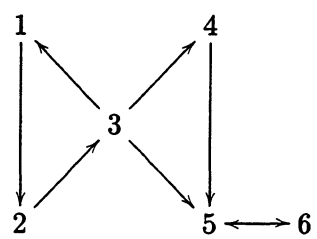
$$(i \in C) \cap (i \to j) \Rightarrow j \in C.$$

A state i is *absorbing* if $\{i\}$ is closed.

Example Diagram makes irreducibility easy to check. Consider P defined by the following stochastic matrix (Norris, 1998, p 11)[1], i.e.

$$P = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ \frac{1}{3} & 0 & 0 & \frac{1}{3} & \frac{1}{3} & 0 \\ 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}.$$

The corresponding diagram makes the solution obvious



The communicating classes are $\{1,2,3\}$, $\{4\}$ and $\{5,6\}$, where $\{5,6\}$ is closed in addition.

[1]Norris, J.R. (1997). *Markov Chains.* Cambridge University Press.

2.3.3 Recurrence

Denote by $\tau_i = \inf\{n \geq 1; X_n = i\}$ a stopping time called the *first passage time*. Denote by r_i the probability of return in finite time, i.e.

$$r_i = P[\tau_i < \infty | X_0 = i] = P_i[\tau_i < \infty].$$

A state i is *recurrent* if $r_i = 1$; it is *transient* otherwise.

Theorem *Given $(X_n)_{n\geq 0} \sim M(\xi, P)$, each state of Ω is either recurrent or transient, and*

$$r_i = 1 \Leftrightarrow \sum_{n=0}^{\infty} p_{ii}(n) = \infty \qquad \textit{and} \qquad r_i < 1 \Leftrightarrow \sum_{n=0}^{\infty} p_{ii}(n) < \infty.$$

Define $V_i = \sum_{n=0}^{\infty} \delta_{X_n=i}$ to be the number of *visits* to state i, such that

$$E[V_i | X_0 = i] = \begin{cases} \sum_{n=0}^{\infty} E[\delta_{X_n=i} | X_0 = i] = \sum_{n=0}^{\infty} p_{ii}(n) \\ \sum_{v=1}^{\infty} v P_i[V_i = v] = \sum_{v=1}^{\infty} \sum_{m=0}^{v-1} P_i[V_i = v] \\ \qquad\qquad = \sum_{m=0}^{\infty} \sum_{v=m+1}^{\infty} P_i[V_i = v] = \sum_{m=0}^{\infty} P_i[V_i > m]. \end{cases}$$

Finally, due to the SMP, τ_i is independent and identically distributed between each return, so that $P[V_i > m | X_0 = i] = r_i^m$ for $m \geq 0$, i.e. the probability of making m returns in a finite time.

Theorem *Any communicating class either consists of all recurrent states or all transient ones.* For any $i \leftrightarrow j$, there exists some $n, m \geq 0$, such that $p_{ij}(n) > 0$ and $p_{ji}(m) > 0$, and for all $t > 0$,

$$p_{ij}(n) p_{jj}(t) p_{ji}(m) \leq p_{ii}(n + t + m) \qquad \Rightarrow \qquad \sum_{t=0}^{\infty} p_{jj}(t) \leq \frac{\sum_{t=0}^{\infty} p_{ii}(n + t + m)}{p_{ij}(n) p_{ji}(m)}.$$

Thus, (i) transient i implies $\sum_{n=0}^{\infty} p_{jj}(n) < \infty$, i.e. transient j, and (ii) recurrent j implies recurrent i. Symmetry implies that the results also hold the other way around.

Remark Like irreducibility, recurrence (or transience) is a class property. Thus, that P is recurrent implies that it is irreducible. Indeed, recurrence of P ensures that each member state of Ω will be visited infinitely often, which makes the convergence of the Markov chain interesting.

Theorem *Every recurrent class is closed. Every finite closed class is recurrent.* Should recurrent i and $i \to j$ not imply $j \to i$, we would have $P_i[\tau_i = \infty] > 0 \Rightarrow r_i < 1$. Whereas the states of a finite closed class can not all be transient, since to nowhere can the chain escape.

Remark An irreducible Markov chain may be transient only if it has an infinite state-space.

2.3.4 Invariant distribution and positive recurrence

A probability distribution, denoted by $\pi = (\pi_i)_{i\in\Omega}$, is *invariant* for $M(\xi, P)$, indeed P, if

$$\pi = \pi P \qquad \Leftrightarrow \qquad \forall\, j \in \Omega,\ \pi_j = \sum_{i\in\Omega} \pi_i p_{ij}. \tag{2.1}$$

Invariant distribution π is also said to be *stationary* or *equilibrium*.

Remark (2.1) shows that $M(\pi, P)$ generates a dependent, but identically distributed sample.

Invariance and recurrence are closely related. Denote by η_i the *expected time of return*, i.e.

$$\eta_i = E[\tau_i | X_0 = i] = E_i[\tau_i]$$

of a recurrent state i. It is *positive recurrent* if $\eta_i < \infty$; otherwise it is *null recurrent*.

Theorem *Irreducible* $M(\xi, P) \Rightarrow (\forall\, i \in \Omega,\ \eta_i < \infty \quad \Leftrightarrow \quad \pi = \pi P$ *where* $\pi_i = 1/\eta_i)$.

Remark Positive recurrence strengthens the convergence of the Markov chain. In a way, it is about the rate of convergence, i.e. each member state of Ω will be reached in a finite period of time.

Example (One-dimensional random walk) Let $\Omega = \{0, \pm 1, \pm 2, ...\}$ with transition probability $p(i, i+1) = p$ and $p(i, i-1) = 1 - p = q$. Clearly, it is irreducible for $p \in (0, 1)$. To check if it is recurrent, we only need to consider one point, say, the origin, where

$$p_{00}(n) = \begin{pmatrix} n \\ \frac{n}{2} \end{pmatrix} p^{\frac{n}{2}} q^{\frac{n}{2}} \quad \text{if } \frac{n}{2} \in \Omega, \quad \text{and 0 otherwise.}$$

Stirling's formula $n! \sim \sqrt{2\pi n}(n/e)^n$ gives us, in case of $p = q$,

$$p_{00}(2n) \propto (4pq)^n/\sqrt{n} = 1/\sqrt{n} \qquad \Rightarrow \qquad \exists\, N,\ \sum_{n=N}^{\infty} p_{00}(2n) \propto \sum_{n=N}^{\infty} 1/\sqrt{n} = \infty.$$

Thus, the symmetric one-dimensional random walk is recurrent. Otherwise,

$$4pq = b < 1 \qquad \Rightarrow \qquad \exists\, N,\ \sum_{n\geq N} p_{00}(2n) \leq \sum_{n\geq N} b^n < \infty,$$

so that the asymmetric one-dimensional random walk is transient. Returning to the case of $p = q$,

$$\gamma_i = 1 \qquad \text{and} \qquad \gamma_i = \frac{1}{2}\gamma_{i-1} + \frac{1}{2}\gamma_{i+1} \qquad \Rightarrow \gamma = \gamma P.$$

Yet, $\sum_i \gamma_i = \infty$, i.e. not a probability function, so that the symmetric random walk is null recurrent.

2.3.5 Reversibility

Random process $Y_n = X_{N-n}$ is said to be the *time-reversal* of $(X_n)_{0 \le n \le N}$.

Theorem *Let $(X_n)_{0 \le n \le N} \sim M(\pi, P)$, with irreducible P and its invariant π. The time-reversal is then $(Y_n)_{0 \le n \le N} \sim M(\pi, P')$ with the same invariant π, where P' is given by*

$$\pi_j p'_{ji} = \pi_i p_{ij}.$$

First, P' is a stochastic matrix, since

$$\sum_i p'_{ji} = \pi_j^{-1} \sum_i \pi_i p_{ij} = \pi_j^{-1} \pi_j = 1.$$

Next, π is invariant for P', since

$$\sum_j \pi_j p'_{ji} = \sum_j \pi_i p_{ij} = \pi_i (\sum_j p_{ij}) = \pi_i \cdot 1 = \pi_i.$$

Finally, $M(\pi, P')$ defines a Markov chain, since

$$\begin{aligned} P[Y_0 = i_0 \cap Y_1 = i_1 \cap \cdots \cap Y_N = i_N] &= P[X_0 = i_N \cap X_1 = i_{N-1} \cap \cdots \cap X_N = i_0] \\ &= \pi_{i_N} p(i_N, i_{N-1}) p(i_{N-1}, i_{N-2}) \cdots p(i_1, i_0) \\ &= p'(i_{N-1}, i_N) \pi_{i_{N-1}} p(i_{N-1}, i_{N-2}) \cdots p(i_1, i_0) \\ &\vdots \\ &= p'(i_{N-1}, i_N) p'(i_{N-2}, i_{N-1}) \cdots p'(i_0, i_1) \pi_{i_0}. \end{aligned}$$

A probability measure, denoted by $\pi = (\pi_i)_{i \in \Omega}$, is in *detailed balance* with P if, $\forall i, j \in \Omega$,

$$\pi_i p_{ij} = \pi_j p_{ji}. \tag{2.2}$$

In particular, the time-reversal of $M(\pi, P)$ is the same $M(\pi, P)$, and the chain is *reversible.*

Remark Detailed balance implies invariance, since $(\pi P)_j = \sum_i \pi_i p_{ij} = \sum_i \pi_j p_{ji} = \pi_j$. The implication from detailed balance to positive recurrence (and invariant distribution), to recurrence, and finally, to irreducibility, ensures that all the important properties are satisfied once the Markov chain attains detailed balance. Moreover, (2.2) is computationally easier to handle than (2.1). In the context of Markov chain Monte Carlo, where the existence of a target function, denoted by π, is granted *a priori*. The detailed balance equation then provides a powerful means of constructing transition P with desired convergence properties.

2.3.6 Ergodic theorem

Observed frequency As before, let τ_i be the first passage time of i, and η_i the expected return time. In addition, let $V_i(n)$ be the number of visits to state i *before* time n, and $\bar{V}_i(n)$ the *observed frequency* of i based on the same sample, i.e.

$$V_i(n) = \sum_{m=0}^{n} \delta_{X_m=i} \qquad \text{and} \qquad \bar{V}_i(n) = n^{-1} V_i(n).$$

Ergodic theorem (I) *Irreducible* $(X_n)_{n\geq 0} \sim M(\xi, P) \Rightarrow \bar{V}_i(n) \overset{a.s.}{\to} \eta_i^{-1}$.
Let $T_i(k)$ be the time of the k-th return to i, and $\tau_i(k)$ the corresponding k-th wandering period,

$$\tau_i(k) = T_i(k) - T_i(k-1) \quad \Rightarrow \quad T_i[V_i(n)-1] = \sum_{k=1}^{V_i(n)-1} \tau_i(k) \leq n \leq \sum_{k=1}^{V_i(n)} \tau_i(k) = T_i[V_i(n)].$$

The SMP implies that $\tau_i(k)$, for $k = 1, 2, ...$, are independent and identically distributed. The strong law of large numbers implies that, in case of $\eta_i < \infty$ (positive recurrence),

$$\frac{\sum_{k=1}^{V_i(n)} \tau_i(k)}{V_i(n)} \overset{a.s.}{\to} \eta_i \qquad \text{and} \qquad \frac{\sum_{k=1}^{V_i(n)-1} \tau_i(k)}{V_i(n)} \overset{a.s.}{\to} \eta_i \lim_{n\to\infty} \frac{V_i(n)-1}{V_i(n)} = \eta_i,$$

forcing almost sure convergence of $1/\bar{V}_i(n)$ towards η_i. Otherwise, $\eta_i = \infty$, and $\forall t < \infty$,

$$P[\lim_{k\to\infty} T_i(k)/k \leq t] = P[\lim_{k\to\infty} T_i(k-1)/k \leq t] = 0 \quad \Rightarrow \quad \bar{V}_i(n) \overset{a.s.}{\to} 0 = 1/\eta_i.$$

Sample and invariant average Let $f(x)$ be any real-valued function. Let $\bar{f}_n$, and $\bar{f}_\pi$ be, respectively, its *sample*, and *invariant average* (in case of positive recurrence), i.e.

$$\bar{f}_n = \frac{1}{n} \sum_{k=0}^{n-1} f(X_k) \qquad \text{and} \qquad \bar{f}_\eta = E_\pi(f) \quad \text{where} \quad \pi_i = \frac{1}{\eta_i}.$$

Ergodic theorem (II) *Positive recurrent* $(X_n)_{n\geq 0} \sim M(\xi, P) \Rightarrow \bar{f}_n \overset{a.s.}{\to} E_\pi(f)$.

Remark A simple proof can be given for bounded f, i.e. $|f| \leq M < \infty$. A countable state-space Ω can be labeled as $\{1, 2, ...\}$. We have, $\forall N < \infty$,

$$\frac{1}{M}|\bar{f}_n - \bar{f}_\pi| \leq \sum_{i=1}^{\infty} |\frac{V_i(n)}{n} - \pi_i| \leq \sum_{i\leq N} |\frac{V_i(n)}{n} - \pi_i| + \sum_{i>N} [\frac{V_i(n)}{n} + \pi_i] \quad \overset{a.s.}{\to} \quad 2\sum_{i>N} \pi_i.$$

Since $\sum_{i=1}^{\infty} \pi_i = 1$ (i.e. finite), we have $\forall \epsilon > 0$, $\exists N(\epsilon) < \infty$, such that $2\sum_{i>N(\epsilon)} \pi_i < \epsilon$, and

$$P[\lim_n |\bar{f}_n - E_\pi(f)| < \epsilon] = 1.$$

2.4 General state-space theory

2.4.1 Some definitions

General state-space Ω is a *general state-space* w.r.t. distribution P, if the collection of subsets of Ω, denoted by E, on which P is defined, is a countably generated σ-algebra. We may also say that Ω is *E-/P-measurable.*

Remark This is the case in most applications. However, apart from certain mathematical details, the general state-space theory is almost the exact parallel of the discrete one. See e.g. Tierney (1994)[2] for a review.

Transition kernel The *transition kernel* P of $M(\xi, P)$ with measure $\mu()$ defined on a general state-space Ω is such that, $\forall 0 \leq n < \infty$ and $A \subset \Omega$,

$$P(x, A) = P[X_{n+1} \in A | X_n = x] = \int_A p(x, y)\mu(dy).$$

Transition after transition The two-step transition kernel derived from P is defined as

$$\begin{aligned} P^2(x, A) &= P[X_2 \in A | X_0 = x] \\ &= \int_\Omega P(y, A)P(x, dy) \\ &= \int_\Omega \{\int_A p(y, z)\mu(dz)\}\ p(x, y)\mu(dy). \end{aligned}$$

The n-step (for $n \geq 2$) transition kernel is recursively defined by

$$P^n = PP^{n-1}.$$

[2]Tierney, L. (1994). Markov chains for exploring posterior distributions. (With discussion). *The Annals of Statistics*, **22**, *1701 - 1762.*

2.4.2 Irreducibility

Let stopping time τ_A be the first passage time of $A \subset \Omega$, i.e.

$$\tau_A = \inf\{n \geq 1; X_n \in A\}.$$

$M(\xi, P)$ is *ψ-irreducible* for some probability distribution ψ if

$$\forall\, A \subset \Omega,\ \psi(A) > 0 \quad \Rightarrow \quad P[\tau_A < \infty] > 0.$$

$M(\xi, P)$ is *irreducible* if it is ψ-irreducible for some ψ, and ψ is an *irreducibility distribution.*

Remark Like the case with discrete state-space, the initial distribution ξ becomes aymptotically irrelevant provided irreducibility. The chain can reach all the interesting sets of Ω w.r.t. to the irreducibility distribution concerned. This is more general than in the discrete case.

Example (Random walk on the non-negative half line with an absorbing origin)

$$0 \longleftarrow 1 \longleftrightarrow 2 \longleftrightarrow \cdots \longleftrightarrow i \longleftrightarrow i+1 \longleftrightarrow \cdots$$

This is not an irreducible chain in the discrete case, since e.g. $1 \to 0$ but $0 \not\to 1$. However, the chain becomes irreducible under the present definition, since it is ψ-irreducible w.r.t. e.g. $\psi_x = 1$ if $x = 0$ and $\psi_x = 0$ everywhere else.

2.4.3 Invariant distribution and detailed balance

$M(\xi, P)$ has an invariant distribution, denoted by π, if

$$\pi = \pi P \quad \Leftrightarrow \quad \forall\, A \subset \Omega, \quad \pi(A) = \int_{\Omega} P(x, A)\pi(dx). \tag{2.3}$$

Theorem *Irreducible $M(\xi, P)$ with invariant distribution π implies that*

1. *it is π-irreducible,*
2. *π is the unique invariant distribution,*
3. *the chain is positive recurrent, so that $\forall A \subset \Omega$ where $\pi(A) > 0$, we have*
   - 3.1. *$P[X_n \in A$ infinitely often$|X_0 = x] > 0$ for all x, and*
   - 3.2. *$P[X_n \in A$ infinitely often$|X_0 = x] = 1$ for π-almost all x.*

Remark Unlike the discrete case, positive recurrence is by definition given by the existence of invariant π. It is also defined to be a class property. Since π is granted *a priori* in applications of MCMC, we have omitted the definition of recurrence, as it builds on several concepts not presented here. In particular, recurrence without invariant distribution is said to be null recurrent.

Invariant π and transition kernel P are in *detailed balance* if, for $P(x, dy) = p(x, y)\mu(dy)$,

$$\pi(x)p(x, y) = \pi(y)p(y, x). \tag{2.4}$$

2.4.4 Ergodic theorem

Define the *(n-step) average transition kernel* to be

$$\bar{P}^n(x, A) = \sum_{k=0}^{n} P^k(x, A)/(n+1).$$

General Ergodic theorem (I) *Irreducible* $(X_n)_{n\geq 0} \sim M(\xi, P)$ *with invariant distribution* π $\Rightarrow \sup_{A\subset\Omega} |\bar{P}^n(x, A) - \pi(A)| \overset{a.s.}{\to} 0$ *for* π*-almost all* x.

Remark This is the generalized version of the pointwise convergence of the observed frequency in the discrete case. It can be further strengthened, i.e. $\forall A \in \Omega$, $P^n(x, A) \overset{a.s.}{\to} \pi(A)$ for π-almost all x, provided $M(\xi, P)$ is aperiodic in addition.

Let f be a real-valued function with finite absolute *invariant average*, i.e.

$$E_\pi(|f|) = \int |f(x)|\pi(dx) < \infty.$$

General Ergodic theorem (II) Irreducible $(X_n)_{n\geq 0} \sim M(\xi, P)$ with invariant distribution π $\Rightarrow \bar{f}_n \overset{a.s.}{\to} E_\pi(f)$ for π-almost all x.

Remark This corresponds to the convergence of sample average in the discrete case.

Remark Unlike Ergodic theorem in the discrete case, the initial distribution ξ is not entirely irrelevant here: it should be absolutely continuous with π, i.e. granting null probability to π-null sets. More specifically, irreducible $M(\xi, P)$ with invariant π admits decomposition

$$\Omega = H \cup D,$$

where H is absorbing and recurrent, and D is π-null and *dissipative*, i.e. a countable union of transient sets. The set H is called a *Harris set* for the chain, where

$$P[X_n \in A \text{ infinitely often} | X_0 = x] = 1 \qquad \text{for all } x \in H.$$

A Markov chain is *Harris recurrent* if and only if Ω is a Harris set, where initial X_0 is entirely irrelevant.

Example (Random walk on the non-negative half line) W.r.t. degenerate irreducibility distribution $\psi_0 = 1$, the chain is Harris recurrent if $p \leq 1/2$, in which case with unity probability it ends up with the absorbing state 0; otherwise it is recurrent but not Harris recurrent, where $H = \{0\}$ and $D = \{1, 2, ...\}$.

2.5 The central limit theorem

2.5.1 The central limit theorem

The Markov chain sample average converges almost surely to the invariant average provided positive recurrence. Its efficiency is governed by the CLT for Markov chains[3].

Central limit theorem (CLT) *Irreducible, reversible* $(X_n)_{n\geq 0} \sim M(\xi, P)$ *with invariant* π,

$$\sqrt{n}[\bar{f}_n - E_\pi(f)] \xrightarrow{D} N(0, \sigma_f^2)$$

where

$$\begin{aligned} nVar(\bar{f}_n) \xrightarrow{a.s.} \sigma_f^2 &= Var_\pi(f) + 2\sum_{h=1}^{\infty} Cov(f_0, f_h) \\ &= \gamma_0 + 2\sum_{h=1}^{\infty} \gamma_h \quad < \infty. \end{aligned} \tag{2.5}$$

With independent sample the variance would have simply been $Var_\pi(f)$. In any case, we have

$$\begin{aligned} nVar(\bar{f}_n) &= \frac{1}{n}\sum_{i=1}^{n} Var_\pi(f_i) + \frac{2}{n}\sum_{i<j} Cov(f_i, f_j) \\ &= Var_\pi(f) + \frac{2}{n}\sum_{i=1}^{n-1}\sum_{h=1}^{n-i} Cov(f_i, f_{i+h}) \\ &= Var_\pi(f) + 2\sum_{h=1}^{n-1}(1 - \frac{h}{n})Cov(f_0, f_h) \\ &= \gamma_0 + 2\sum_{h=1}^{n-1}(1 - \frac{h}{n})\gamma_h, \end{aligned}$$

since, by reversibility and invariance,

$$Cov(f_i, f_{i+h}) = Cov(f_0, f_h) = Cov(f_0, f_{-h}).$$

This gives us a monotone increasing sequence which converges almost surely to $\sum_{h=-\infty}^{\infty} Cov(f_0, f_h)$, provided the sum exists.

Remark Notice that the CLT here requires reversibility, which is stronger than positive recurrence in the case of Ergodic theorem.

[3]The version presented here was ascribed to Kipnis and Varadhan, and described in
Geyer, C.J. (1992). Practical Markov chain Monte Carlo. (With discussion). *Statistical Science*, **7**, *473 - 511.*

2.5.2 Variance estimation

No sample of finite size can to used to estimate γ_h as $h \to \infty$, so that no consistent estimator of σ_f^2 can be formed based on the corresponding sample covariances. Of course, this does not imply that there exist no consistent estimators.

Batch-mean estimator A simple *batch-mean* estimator can be constructed by *sequentially* dividing the sample into q sub-samples, denoted by $s_1, ..., s_q$, each of size m, i.e. $n = m \cdot q$. The batch-means, i.e.

$$\bar{f}_m^{(k)} = \sum_{i \in s_k} f_i/m \qquad \text{for} \quad k = 1, ..., q,$$

converge in distribution to i.i.d. Normal sample, i.e.

$$\bar{f}_m^{(1)}, \ ..., \ \bar{f}_m^{(q)} \quad \overset{i.i.d.}{\sim} \quad N[E_\pi(f), \sigma_f^2/m].$$

The sample variance of these batch-means, multiplied by m, provides an estimator of σ_f^2, i.e.

$$\hat{\sigma}_{bat}^2 = \frac{m}{q-1} \sum_{k=1}^{q} (\bar{f}_m^{(k)} - \bar{f}_n)^2.$$

Remark The convergence of the batch-means is valid under the same conditions as those of the CLT. The batch-mean estimator is simple but inefficient: for practical situations, the Markov needs to be long enough so that each batch is much longer than the characteristic mixing time of the chain.

Generic Splus transcript

```
sigma.batch <- function(x, q)
{
        n <- length(x)                  # sample size
        m <- trunc(n/q)                 # batch-length
        y <- array(x[(n - m * q + 1):n], c(m, q))
                        # each column of y forms a batch
        f.m <- c(t(rep(1/m, m)) %*% y)  # q batch means
        m * var(f.m)    # output batch-mean estimate of sigma2_f
}
```

Window estimator The sample covariance of lag-h is given by

$$\hat{\gamma}_h = \frac{1}{n}\sum_{j=1}^{n-h}(f_j - \bar{f}_n)(f_{j+h} - \bar{f}_n).$$

A *window* estimator is defined as

$$\hat{\sigma}^2_{win} = w_0\hat{\gamma}_0 + 2\sum_{h=1}^{\infty} w_h\hat{\gamma}_h,$$

where $0 \leq w_h \leq 1$ may dependent on the sample size n. Truncation window estimators are typical in practice, i.e.

$$w_h = 1 \quad \text{for} \quad 0 \leq i \leq K_n \qquad \text{and} \qquad w_h = 0 \quad \text{for} \quad i > K_n,$$

where K_n is some constant depending on n.

Remark Under strong regularity conditions w_h can give consistent window estimator. However, it is unclear whether this is possible under the mild conditions under which the CLT holds. Notice also that the standardizing constant for $\hat{\gamma}_h$ is n instead of $n - h$.

Generic Splus transcript

```
sigma.window <- function(x, w)
{
        max.lag <- length(w)    # no need for more lags
        a.v <- c(acf(x, lag.max = max.lag, type = "covariance", plot = F)$acf)
                # (a) Splus routine "acf" returns sample autocovariance
                #       upto lag-"max.lag" (default value = log(length(x)))
                # (b) type = "correlation" returns autocorrelation instead
                # (c) "plot = F" turns off default graphical display
        a.v <- a.v[ - length(a.v)]      # lag-0 corresponds to w[1]
        2 * sum(w * a.v) - w[1] * a.v[1]        # window sigma2_f
}
```

Initial sequence estimator Geyer (1992)[4] noticed that, for a reversible Markov chain, it can be shown that

$$\Gamma_h = \gamma_{2h} + \gamma_{2h+1}$$

is non-negative, non-increasing, convex function of h. The *initial m-sequence* estimator, denoted by

$$\hat{\sigma}^2_{ini} = \hat{\gamma}_0 + 2\sum_{h=1}^{2m+1} \hat{\gamma}_h,$$

can be defined w.r.t. any of these three characteristics:

1. the initial *positive* sequence estimator is such that

$$\hat{\Gamma}_h > 0 \quad \text{for} \quad h \leq m \qquad \text{and} \qquad \hat{\Gamma}_{m+1} \leq 0,$$

2. the initial *monotone* sequence estimator is such that

$$\hat{\Gamma}_1 \geq \cdots \geq \hat{\Gamma}_m > 0 \qquad \text{and} \qquad \hat{\Gamma}_m < \hat{\Gamma}_{m+1},$$

3. the initial *convex* sequence estimator is such that, in addition to being monotone,

$$\hat{\Gamma}_{h-1} + \hat{\Gamma}_{h+1} \geq 2\hat{\Gamma}_h \quad \text{for} \quad h < m \qquad \text{and} \qquad \hat{\Gamma}_{m-1} + \hat{\Gamma}_{m+1} < 2\hat{\Gamma}_m.$$

Remark The initial sequence estimators are special cases of the truncation window estimator. It is not clear that any of them are consistent given reversibility. But they are obviously over-estimates compared to any other window estimators. Indeed, "Theorem 3.2" of Geyer (1992) states that they are consistent over-estimates in the sense that, for any of the three initial sequence estimators,

$$P[\lim_{m\to\infty} \hat{\sigma}^2_{ini} \geq \sigma^2_f] = 1.$$

[4]Geyer, C.J. (1992). Practical Markov chain Monte Carlo. *Statistical Science*, **7**, *473 - 511*.

Generic Splus transcript for the initial sequence estimators

```
sigma.ini <- function(x, max.lag = 100)
{
        a.v <- c(acf(x, lag.max = max.lag, type = "covariance", plot = F)$acf)
        k <- trunc(length(a.v)/2)       # how many Gamma can we form?
        gma <- array(a.v[1:(2 * k)], c(2, k))
        gma <- gma[1,  ] + gma[2,  ]    # the sample Gamma

        npos <- gma <= 0        # idnetifying the non-positive Gamma
        npos[k] <- T    # securing against the case of all positive
        pos <- min(c(0:k)[npos])        # the first one of them
        s.pos <- 2 * sum(gma[1:pos]) - a.v[1]   # initial positive sigma2_f

        goon <- T
        mono <- 0
        while(goon) {
                mono <- mono + 1        # update monotone sequence
                goon <- (gma[mono + 1] <= gma[mono]) & (mono < k - 1)
        }
        mono <- min(mono, pos)  # positivity guarantee
        s.mono <- 2 * sum(gma[1:mono]) - a.v[1] # initial monotone sigma2_f

        goon <- T
        cnv <- 1
        while(goon) {
                cnv <- cnv + 1  # update convex sequence
                goon <- (gma[cnv - 1] + gma[cnv + 1] >= 2 * gma[cnv]) & (cnv <
                        k - 1)
        }
        cnv <- min(cnv, mono)   # monotonicity guarantee
        s.cnv <- 2 * sum(gma[1:(cnv - 1)]) - a.v[1]     # convex sigma2_f
        c(s.pos, s.mono, s.cnv) # output the three ini_seq_est
}
```

2.5.3 Example: Autoregression model AR(1)

Consider an autoregression model AR(1), i.e. let ρ be the lag-one autocorrelation, and

$$X_t = \rho X_{t-1} + \epsilon_t \qquad \text{where} \quad \epsilon_1, \ ..., \ \epsilon_n \overset{i.i.d.}{\sim} N(0, \tau^2).$$

Clearly, $X_t \perp (X_0, ..., X_{t-2}) | x_{t-1}$, so that $(X_t)_{t=0}^n$ is a Markov chain. Let $f(x) = x$. We may apply the various variance estimators and compare the results against the theoretical σ_f^2, i.e.

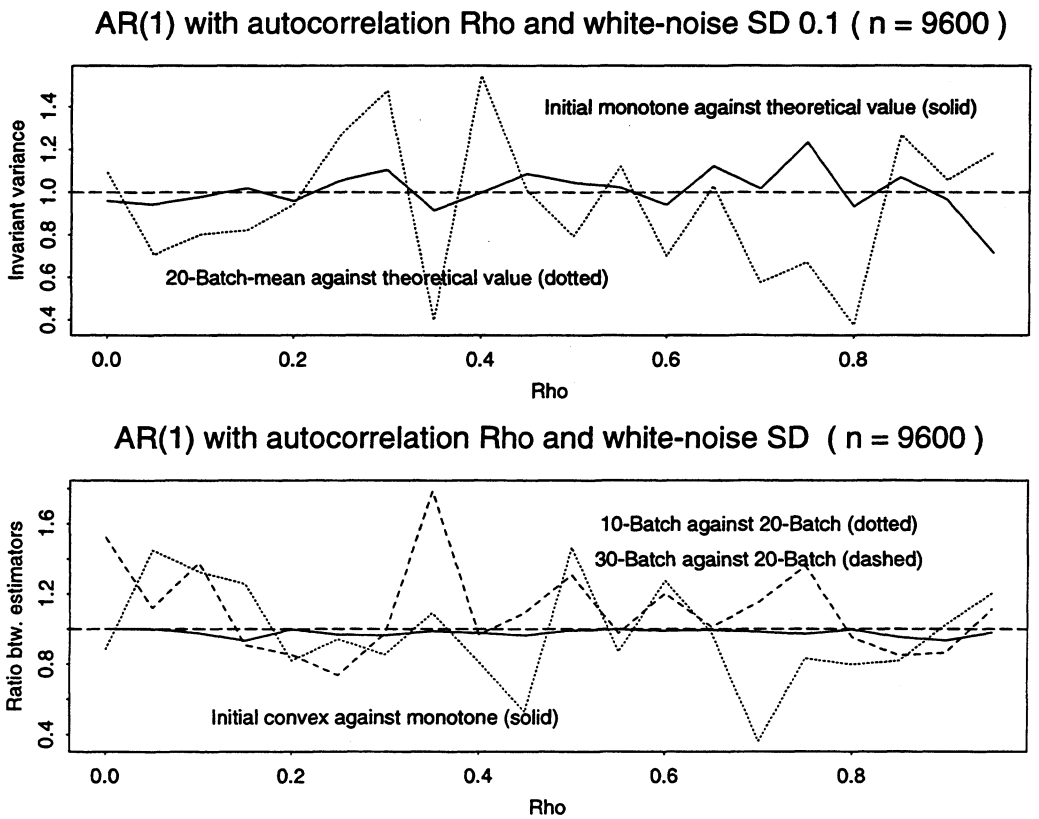
$$\sigma_f^2 = \gamma_0(1+\rho)/(1-\rho) = \tau^2/(1-\rho)^2.$$

We set up the following simple simulation framework. Let

$$\tau = 0.1 \qquad \text{and} \qquad \rho = (i-1)/20 \quad \text{for} \quad i = 1, \ ..., \ 20.$$

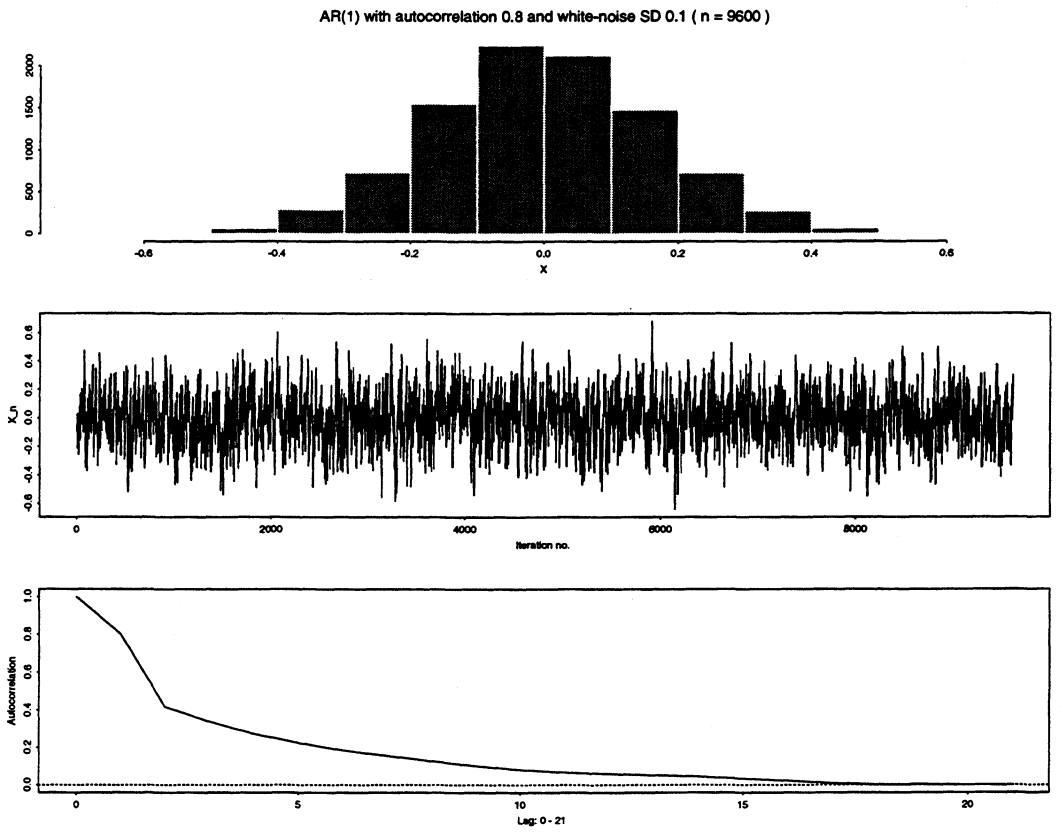
Starting at $x_0 = 0$, we generate a sample of 10000 observations at each ρ. To ensure that the Markov chain accepted has reached equilibrium, we throw away the first 4% points so that the final sample size is $n = 9600$. Based on these, we calculate, for each ρ, the initial positive and monotone sequence estimators, as well as the 10-, 20- and 30-Batch-mean estimators.

Remark The part of simulated sample which is thrown away is referred to as the "burn-in" period. We notice that sample size of 9600 is probably too small for the larger values of ρ.



It can be seen that the Batch-mean estimators are rather sensitive towards the number of batches formed. This complicates its application, since no general results seem to be available for optimal choice in this respect. The burn-in influenced the results to an extent more than one might have expected, so that it remains an issue which should not be dismissed without some inspection in a given situation. Moreover, repeating the simulation at the same value of ρ shows that the variance estimators are unstable for large ρ, such that they should be treated with caution.

The importance of various diagnostics in simulations like these can not be over-stated. For illustration we have included the following plots based on a single run: (a) the histogram indicates the invariant distribution, (b) the step-by-step Markov chain shows how well the chain mixes, and (c) the sample autocorrelations form the basis of inference.



Splus transcript

```
sigma.ar1 <- function(m = 10000, rho = c(0:19)/20, tau = 0.1,
        q = c(10, 20, 30), ini = 0, burnin = 0.04, max.lag = 200)
{
        est <- c("sigma_f", "pos", "mono", "cnvx", paste(q, "-bat", sep = ""))
        tbl <- array(0, c(length(est), length(rho)))    # tabulation of results
        dimnames(tbl) <- list(est, rho)

        for(i in 1:length(rho)) {
                r <- rho[i]     # fixed parameter value
                cat(" rho =", r, "\t")  # display the parameter
                tbl["sigma_f", i] <- (tau/(1 - r))^2    # theoretical sigma2_f
                x <- rep(ini, m)        # starting at x_0 = ini
                for(j in 2:m) { # generating AR(1) process
                        x[j] <- rnorm(1, r * x[j - 1], tau)
                }
                x <- x[round(m * burnin + 1):m] # burn-in
                n <- length(x)  # final sample size
                for(j in 1:length(q)) { # batch sigma2_f
                        tbl[4 + j, i] <- sigma.batch(x, q[j])
                }
                tbl[c("pos", "mono", "cnvx"), i] <- sigma.ini(x, max.lag)
        }

        sink("sigma.btp")
        cat(tbl)
        sink()  # save the results
        list(simulation.result = tbl)
}
```

Chapter 3

Markov chain Monte Carlo

3.1 Introduction

The first part of the Ergodic theorem ensures that the sample frequency of a Markov chain converges pointwise to the invariant distribution. The sample is dependent if it is formed from a single chain, whereas it is independent if we take only one state from a number of chains with independent starting values. Statistical inference can be based either on such a single-chain dependent sample or a multiple-chain independent sample.

The second part of the Ergodic Theorem ensures that the sample average almost surely converges to the invariant average under mild regularity conditions. Various Monte Carlo methods can thus be performed through *Markov chain (MC) sampling*, which is the so called *Markov chain Monte Carlo (MCMC)*.

The MC sampling opens up other inferential possibilities as well. For instance, profile likelihood inference is feasible just like in the case of acceptance sampling. Whereas integrated likelihood methods[1] should naturally find their applications here. Indeed, the promising field of non-(standard) Bayesian inference through MC sampling is only starting to be explored.

In this chapter we explain the Metropolis-Hastings (MH) algorithm as a general MC sampling technique. The Gibbs sampler is shown to be a special case of the MH algorithm. It is also possible to build an Acceptance-Rejection (AR) step into the MH-algorithm, which gives us the MH-AR algorithm. We shall describe a number of practical convergence diagnostics. Case studies will be included for illustration.

[1]Berger, J.O. and Liseo, B. and Wolpert, R.L. (1999). Integrated likelihood methods for eliminating nuisance parameters. *Statistical Science*, **14**, 1-28.

3.2 Metropolis-Hastings (MH) algorithm

Denote by $\pi(x)$ the target invariant distribution. Let $\psi(x, y)$ be some arbitrary p.d.f. of y conditional on x, called the *sampler*. Generate a Markov chain, denoted by $(X_i)_{i\geq 0}$, iteratively as follows: suppose $X_i = x$ at the i-th step,

- generate $u \sim Unif(0, 1)$ independent of $y \sim \psi(x, y)$, where $\psi(x, y)$ is the sampler,
- update $X_{i+1} = y$ if $u \leq \alpha(x, y)$ and $X_{i+1} = x$ otherwise, where

$$\alpha(x, y) = \min\{1, \frac{\pi(y)\psi(y, x)}{\pi(x)\psi(x, y)}\}.$$

The resulting Markov chain converges to the invariant distribution $\pi(x)$ in the sense of the Ergodic Theorem, provided it is irreducible and aperiodic.

Generic Splus code for univariate MH algorithm

```
diagnos <- function(x,i)
{       convergence diagnostic based on Markov chain x and counter i }

mcs.mh <- function(x.0,n)
{
        x <- rep(0,n)           # n = maximum length of the chain
        x[1] <- x.0             # x.0 = starting value
        goon <- T               # initialization
        i <- 1                  # i = the iteration counter
        while (goon) {
                y <- sampler(x[i])      # conditional sampling of y given x
                alpha <- alpha.xy(x[i], y)      # acceptance threshold
                accept <- runif(1, 0, 1) <= alpha
                if (accept) {
                        x[i+1] <- y
                }
                else {
                        x[i+1] <- x[i]
                }
                i <- i + 1      # update the iteration counter
                goon <- diagnos(x, i)   # convergence diagnostic
        }
        list(x = x)
}
```

3.2.1 Understanding the MH algorithm

In MC sampling the target $\pi(x)$ is known *a priori*, possibly upto a proportionality constant. Recall that a sufficient, as well as practical, condition for a Markov chain (irreducible and aperiodic) with transition kernel $P(x,\cdot)$, to converge to $\pi(x)$ in the sense of the Ergodic Theorem, is the detailed balance equation, i.e.

$$\pi(x)P(x,dy) = \pi(y)P(y,dx).$$

The transition kernel defined in the MH algorithm provides a general solution.

The transition kernel of the MH-algorithm can be written as, for any $A \subset \Omega$,

$$\begin{aligned} P(x,A) &= \int_A \psi(x,y)\alpha(x,y)\mu(dy) + I_{x\in A}\{1 - \int_\Omega \psi(x,y)\alpha(x,y)\mu(dy)\} \\ &= \int_A p(x,y)\mu(dy) + r(x)I_{x\in A}, \end{aligned}$$

where $r(x)$ is the probability of remaining at x. By definition, $I_{x\in dy} = 0$ and

$$\pi(x)\psi(x,y) > \pi(y)\psi(y,x) \Rightarrow \alpha(y,x) = 1,$$

and *vice versa*. We obtain the detailed balance as

$$\pi(x)P(x,dy) = \pi(x)p(x,y)\mu(dy) = \pi(y)p(y,x)\mu(dx) = \pi(y)P(y,dx).$$

In addition,

$$\begin{aligned} \int P(x,A)\pi(dx) &= \int [\int_A p(x,y)\mu(dy)]\pi(x)\mu(dx) + \int r(x)I_{x\in A}\pi(x)\mu(dx) \\ &= \int_A [\int p(x,y)\pi(x)\mu(dx)]\mu(dy) + \int_A r(x)\pi(x)\mu(dx) \\ &= \int_A [\int p(y,x)\pi(y)\mu(dx)]\mu(dy) + \int_A r(x)\pi(x)\mu(dx) \\ &= \int_A [1 - r(y)]\pi(y)\mu(dy) + \int_A r(x)\pi(x)\mu(dx) \\ &= \int_A \pi(y)\mu(dy) \\ &= \pi(A), \end{aligned}$$

so that π is the invariant distribution of the Markov chain generated by the MH algorithm.

Finally, it is also clear that MC sampling by the MH algorithm is invariant towards possible proportionality constants in π and ψ.

3.2.2 Random walk, independence and autoregressive chains

Random walk chain The MH-algorithm generates a *random walk* chain if

$$\psi(x,y) = \psi(|y-x|), \qquad \text{i.e. } \alpha(x,y) = \min\{1, \frac{\pi(y)}{\pi(x)}\}.$$

The chain moves 'upwards' whenever possible and 'downwards' with a probability of $\pi(y)/\pi(x)$. This was in fact the Metropolis algorithm suggested by Metropolis *et al.* in 1953[2].

Random walk chains are typically generated using samplers which are symmetric about the current state x, such as the (multivariate) normal and student-t distributions. It remains to be decided the *spread* of the sampler $\psi(z)$ where $z = y - x$, i.e. 'how big the steps of the random walk are'. Both the acceptance rate of each updating and the mixing rate of the chain will be affected. Suggestions[3] have been made that the acceptance rate should be tuned at about 50% in one-dimensional case, and about 25% for large to infinite dimensional problems.

Independence chain The MH algorithm generates a so-called *independence* chain if

$$\psi(x,y) = \psi(y),$$

since the candidate y is drawn independently of the current state x. The sample is nevertheless dependent due to the acceptance-rejection mechanism, i.e.

$$\alpha(x,y) = \min\{1, \frac{w(y)}{w(x)}\} \qquad \text{where} \quad w(x) = \frac{\pi(x)}{\psi(x)}.$$

The independence chain was proposed by Hastings in 1970[4].

The independence sampler ψ should mimic the shape of the target π, as in the case of independent acceptance sampling. In fact,

$$\psi \overset{a.s.}{=} \pi \quad \Rightarrow \quad \alpha(x,y) \overset{a.s.}{=} 1$$

in which case we recover the independent sample. In particular, the independence-chain Monte Carlo resembles the importance Monte Carlo: the former builds up probability mass over the points with large weights $w(x)$, by staying at such points for longer periods of time; whereas the latter does so by assigning them larger pieces of share in the sample average.

---

[2]Metropolis, N. and Rosenbluth, A.W. and Teller, M.N. and Teller, E. (1953). Equations of state calculations by fast computing machines. *J. Chem. Phys.*, **21**, 1081-91.

[3]Roberts, G.O. and Gelman, A. and Gilks, W.R. (1994). Weak convergence and optimal scaling of random walk Metropolis algorithms. *Technical Report*, University of Cambridge.

[4]Hastings, W.K. (1970). Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, **57**, 97-109.

Autoregressive chain In an *autoregressive chain*[5] the candidate value is given as

$$y = a + b(x - a) + z \qquad \text{where} \quad z \sim q(z),$$

for some independent sampler q. This represents an intermediate transition kernel: we obtain

- the random walk chain by setting

$$b = 1;$$

- the independence chain if

$$a = b = 0;$$

- shrinkage towards a if

$$0 < b < 1;$$

- a method of antithetic variates if

$$b = -1,$$

  which provides a simple way of introducing negative correlations between the successive states. This is most effect when π is approximately symmetric, and often helps to reduce the variance of MCMC estimators of linear functions.

[5] Tierney, L. (1994). Markov chains for exploring posterior distributions (with discussions). *Ann. Statist.*, **22**, 1701-62.

3.2.3 Approximate profile likelihood

First-order profile likelihood inference Suppose bipartition of parameter-vector into

$$\theta = (\xi,\ \gamma),$$

where ξ denotes the interest parameter and γ the nuisance part. First-order likelihood analysis of ξ can be based on the profile log-likelihood of ξ, whose asymptotic distribution is given by

$$2\{l_P(\hat{\xi}) - l_P(\xi)\} \sim \chi^2(d_\xi)$$

where d_ξ is the dimension of ξ (Barndorff-Nielsen and Cox, 1994)[6].

Remark To calculate the exact profile likelihood by definition, we need to find, for each value of ξ, the m.l.e. of γ, which can be time-consuming.

Sample profile likelihood Suppose a sample of $\theta_1, ..., \theta_m$ has been obtained, *no matter* from what the distribution, denoted by ψ. If we plot ξ_i against $L(\theta_i)$, for $i = 1, ..., m$, the contour would converge to $L_P(\xi)$, provided the sample covers the entire parameter space, denoted by Θ, as $m \to \infty$, i.e.

$$\forall A \in \Theta \ \cap \ \int_A L(\theta)d\theta > 0 \quad \Rightarrow \quad \psi(A) > 0.$$

The condition is obviously satisfied by

$$\theta_1, ..., \theta_m \quad \sim \quad \psi(\theta) \quad \propto \quad L(\theta),$$

and we may either use independent samples coming through e.g. the acceptance sampling, or dependent samples by means of MC sampling. In any case, the naive *sample profile likelihood* based on such a sample is given as

$$\tilde{L}_P(\xi) = \max_{i:\xi_i=\xi} L(\xi, \gamma_i).$$

Remark The sampling techniques mentioned here are in general more efficient than, say, random uniform samples over Θ, but probably not the most efficient ones. Indeed, the higher probability a sampler would visit the subsets of Θ containing $\hat{\gamma}|\xi$, the more efficient it is for simulation of $L_P(\xi)$.

Remark The sample profile likelihood provides quick means of graphical inspection of $L_P(\xi)$, but is in general not suitable for numerical calculations, such as estimation of confidence intervals of ξ.

[6]Barndorff-Nielsen, O.E. and Cox, D.R. (1994). *Inference and Asymptotics.* London: Chapman and Hall.

Approximate profile likelihood To improve on the naive sample profile likelihood, we define the *approximate profile likelihood* w.r.t. $\theta_1, ..., \theta_m$, as

$$\hat{L}_P(\xi) = \max_{j=1,...,m} L(\xi, \gamma_j) \quad \geq \quad \max_{i:\xi_i=\xi} L(\xi, \gamma_i), \tag{3.1}$$

so that $\hat{L}_P$ always improves on $\tilde{L}_P$, i.e. the contour. This is a Rao-Blackwellization-like[7] procedure. The improvement comes at an extra cost of evaluating, for each ξ at which one wishes to calculate $\hat{L}_P(\xi)$, the likelihood of $L(\xi, \gamma_j)$ for all γ_j in the sample.

Remark In a computing environment like Splus, which handles vector calculations, the extra effort would be small compared to repeated iterative maximization at each ξ.

Generic Splus code for scalar approximate profile likelihood

```
L.obs <- function(xi, gamma)
{
        theta <- rbind(rep(xi, dim(gamma)[2]), gamma)
        calculate the likelihood of theta
}

pro.rao <- function(theta)              # theta = p*m matrix of sample
{
        L.p <- rep(0,dim(theta)[2])     # only at the sample values
        for(i in 1:length(L.p)) {
                L.p[i] <- max(L.obs(xi = theta[1, i], gamma = theta[-1,  ])
        }
        list(L.pro = L.p)
}
```

[7] **Rao-Blackwellization** Suppose a sample of bivariate $(x, y) \sim \pi(x, y)$ is available. Suppose that we are interested in calculating the marginal p.d.f. $\pi(x)$, and that we know the form of $\pi(x|y)$. Since $y_1, ..., y_m$ form a sample from the marginal $\pi(y)$, we may approximate $\pi(x)$ by the simple Monte Carlo, i.e.

$$\pi(x) = \int \pi(x|y)\pi(y)dy = \frac{1}{m}\sum_{i=1}^{m} \pi(x|y_i).$$

This technique is often referred to as the *Rao-Blackwellization*. It generally improves on the standard smoothing techniques based on $x_1, ..., x_m$ alone, now that the evaluation of $\pi(x_i)$ is able to make use of *all* the sample points. The main draw-back is that $\pi(x|y)$ may not be available in closed form.

3.2.4 Example: Approximate profile likelihood analysis of a simple nonresponse model for the Norwegian Labour Force Survey

Consider the Norwegian Labour Force Survey (LFS) data of the 2nd quarter in 1995:

	LFS-Employment	Not LFS-Employment	Nonresponse
Register-Employment	12881	1158	518
Not Register-Employment	1829	6726	796

Let $X = 1$ stand for Register-Employment, and $X = 0$ otherwise. Let $Y = 1$ stand for LFS-Employment, and $Y = 0$ otherwise. Let $R = 1$ stand for Response, and $R = 0$ otherwise. A simple non-ignorable nonresponse model is such that

$$P[R = 1|(x, y)] = P[R = 1|y],$$

i.e. nonresponse is independent of the Register conditional to the LFS. The model is said to be non-ignorable since the LFS-Employment Rate among the respondents differs from that among the nonrespondents. An analysis of post-stratification under such non-ignorable nonresponse can be found in Zhang (1999)[8].

Let $q = P[X = 1]$ which is known from the Register. Define the parameters of the model as

$$\begin{aligned} r_1 &= P[R = 1|y = 1] & r_0 &= P[R = 1|y = 0] \\ p_1 &= P[Y = 1|x = 1] & p_0 &= P[Y = 1|x = 0], \end{aligned}$$

denoted by $\theta = (p_1, p_0, r_1, r_0)^T$. The interest parameter is the overall LFS-employment Rate

$$p = qp_1 + (1 - q)p_0.$$

Index the joint data (table above) as a 2×3-matrix, denoted by (n_{ij}) for $i = 1, 2$ and $j = 1, 2, 3$, with the corresponding cell-probabilities (ξ_{ij}), i.e.

$$\xi = \begin{bmatrix} qp_1r_1 & q(1-p_1)r_0 & qp_1(1-r_1) + q(1-p_1)(1-r_0) \\ (1-q)p_0r_1 & (1-q)(1-p_0)r_0 & (1-q)p_0(1-r_1) + (1-q)(1-p_0)(1-r_0) \end{bmatrix}.$$

The likelihood and its logarithm are given as

$$L(\theta) = L(\xi) \propto \prod_{i=1,2;j=1,2,3} \xi_{ij}^{n_{ij}} \quad \text{and} \quad l(\theta) = l(\xi) = \sum_{i=1,2;j=1,2,3} n_{ij} \log \xi_{ij}.$$

The maximum likelihood estimator (m.l.e.) can be obtained through the EM-algorithm, giving us $\hat{\theta} = (0.912, 0.202, 0.971, 0.901)$, and $\hat{p} = 0.637$. In comparison, the simple sample mean is

[8]Zhang, L.-C. (1999). A note on post-stratification when analyzing binary survey data subject to nonresponse. *J. Off. Statist.*, **15**, 329-34.

$\hat{p}_{srs} = 0.651$, and the post-stratified estimate $\hat{p}_{pst} = 0.645$, which corrects about 50% of the bias in $\hat{p}_{srs}$ under the nonresponse model.

First-order likelihood inference of the interest parameter p, with nuisance parameters θ subjected to the restriction of $p = qp_1 + (1-q)p_0$, can be based on the profile likelihood $L_P(p)$. Exact profile likelihood requires repeated EM-algorithm, and is time-consuming. Instead, we shall calculate the approximate profile likelihood (3.1) using three different sampling techniques. Notice that, all parameters taking value from $(0,1)$, both Θ and $L(\theta)$ have finite measures in the present case.

- The first method is the acceptance sampling, which gives us an independent sample. As the source function, we take the multivariate Normal distribution, located at $\hat{\theta}$. We take the observed formation $\hat{j}^{-1}$, multiplied by an inflation constant k, as the covariance matrix. Tentative pre-runs at $k = 0.5, 1, 1.5, 2, 2.5, 3$ with $m = 100$ suggests $k = 1.5$ as the best choice. The corresponding acceptance rate is about 40%, which is not bad for a four-dimensional problem.

  Remark It is tedious to derive the observed information, i.e.

  $$j(\theta) = -\partial^2 l(\theta)/\partial\theta^2,$$

  directly w.r.t. θ. However, we can easily write down the 6×4-Jacobian of transformation

  $$J = \partial\xi/\partial\theta \qquad \Rightarrow \qquad j(\theta) = J^T j(\xi) J,$$

  where $j(\xi)$ is a diagonal matrix with n_{ij}/ξ_{ij}^2 as its diagonal elements.

- The same multivariate Normal distribution can be used as an independence sampler, which gives an independence chain with $L(\theta)$ as its invariant distribution. The acceptance rate is about $60 - 70\%$ here.

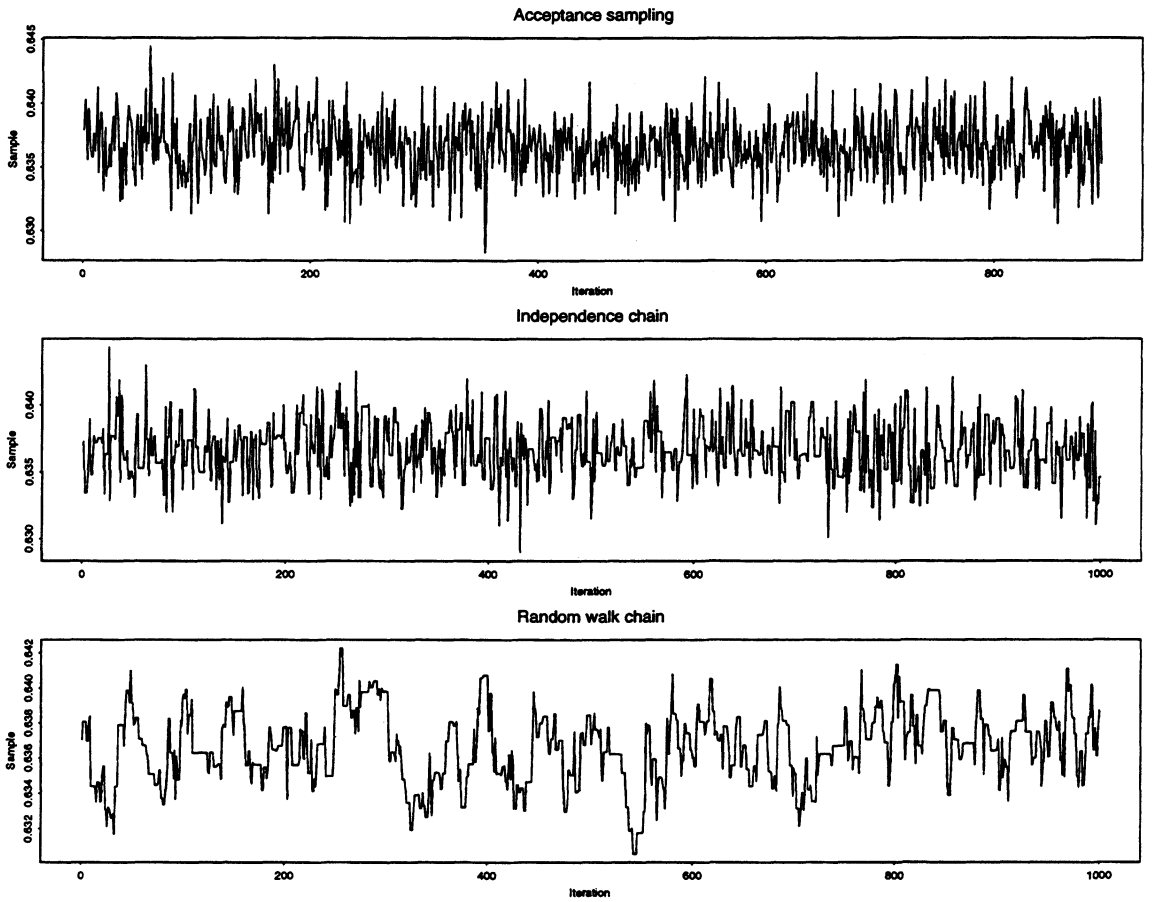
- Finally, we construct a random walk chain through a multivariate Normal sampler, centered at the current state with shrinked observed formation as the covariance matrix. The acceptance rate is about 40% at $k = 0.6$.

Setting $n = 2200$, we retained an independent sample of size $m = 894$ through the acceptance sampling. We then ran both Markov chains for 1000 iterations, and took the first 107 iteration (about 10%) as the burn-in period, so that all the three samples are of the same size.

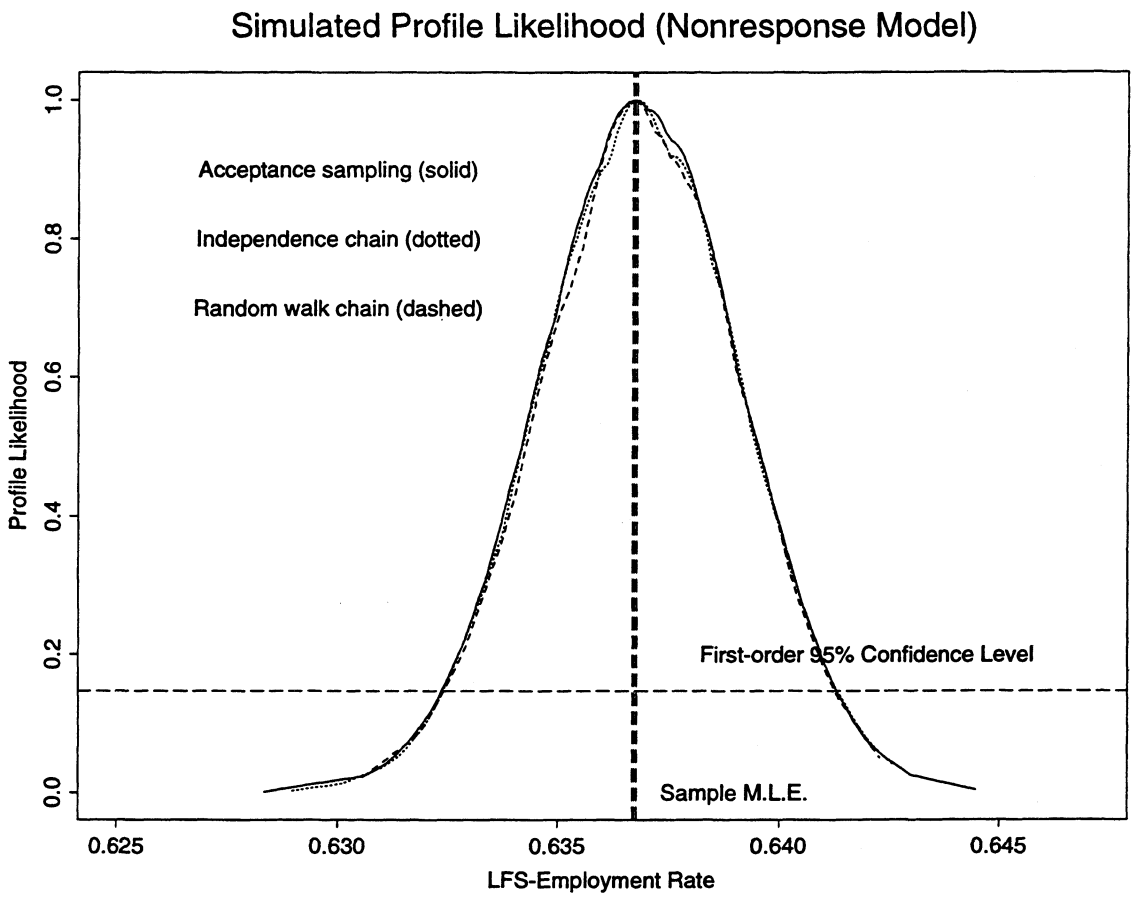
The sample pathes (of p) have been given in the figure, from which the random walk chain clearly is the slowest mixing of the three. Estimation of the standard deviations of the sample $\bar{p}$ based on the initial positive estimator gave us $\hat{\sigma}_\pi = 2.32 \times 10^{-3}$ for the acceptance sampling, and $\hat{\sigma} = 2.52 \times 10^{-3}$ for the independence chain, and $\hat{\sigma} = 7.42 \times 10^{-3}$ for the random

walk chain. The sample autocorrelation of the independence chain decayed rather quickly. Nevertheless, the acceptance sampling appeared to be the most efficient in this case.

A Markov chain has to be run one iteration after another, and therefore always takes much longer time to complete. In contrast, the acceptance sampling is able to take advantage of the parallel vector computing facility of the Splus. The main problem is to find a workable source function in high-dimensional problems. Explicit observed formations can be difficult to obtain, and the likelihood may be highly skewed. The independence chain faces the same problem. In short, the random walk chain is the easiest to construct, but generally results in the largest autocorrelations and, therefore, is often the least efficient.



In any case, we have calculated the approximate profile likelihood $\hat{L}_P(p)$ based on each sample. They have been plotted together for comparison. In particular, the 95% confidence intervals agree very well with each other (to the precision of 10^{-3}). In fact, much better than the estimated standard deviations of $\bar{p}$ have suggested. Notice that both $\hat{p}_{pst} = 0.645$ (post-stratification) and $\hat{p}_{srs} = 0.651$ (simple sample mean) fall outside of the 95% confidence region. Indeed, the bias caused by nonresponse clearly dominates the sampling variance in the overall error of the estimator of LFS-Employment.


Simulated Profile Likelihood (Nonresponse Model)
Acceptance sampling (solid)
Independence chain (dotted)
Random walk chain (dashed)
First-order 95% Confidence Level
Sample M.L.E.
Profile Likelihood
0.0
0.2
0.4
0.6
0.8
1.0
0.625
0.630
0.635
0.640
0.645
LFS-Employment Rate

Splus script

```
# calculating the observed formation
j.hat <- function(y, theta, q)
{
        xi <- array(0, c(2, 3))          # xi = canonical parameter
        xi[1, 1] <- q * theta[1] * theta[3]
        xi[1, 2] <- q * (1 - theta[1]) * theta[4]
        xi[1, 3] <- q - sum(xi[1, ])
        xi[2, 1] <- (1 - q) * theta[2] * theta[3]
        xi[2, 2] <- (1 - q) * (1 - theta[2]) * theta[4]
        xi[2, 3] <- 1 - q - sum(xi[2, ])  # the 2*3 cell-probabilities
        j.xi <- diag(c(y[1, ], y[2, ])/(c(xi[1, ], xi[2, ])^2))
        Jacobian <- array(0, c(6, 4))   # J = Jacobian of transformation
        Jacobian[1, ] <- q * c(theta[3], 0, theta[1], 0)
        Jacobian[2, ] <- q * c( - theta[4], 0, 0, 1 - theta[1])
        Jacobian[3, ] <-  - q * c(theta[3] - theta[4], 0, theta[1],
                                  1 - theta[1])
        Jacobian[4, ] <- (1 - q) * c(0, theta[3], theta[2], 0)
        Jacobian[5, ] <- (1 - q) * c(0,  - theta[4], 0, 1 - theta[2])
        Jacobian[6, ] <-  - (1 - q) * c(0, theta[3] - theta[4],
                                  theta[2], 1 - theta[2])
        t(Jacobian) %*% j.xi %*% Jacobian
}

# calculating the log-likelihood (theta in matrix form)
log.L <- function(y, theta, q)
{
        xi.11 <- q * theta[1, ] * theta[3, ]  # cell-prob. (1,1)...
        l <- y[1, 1] * log(xi.11)
        xi.12 <- q * (1 - theta[1, ]) * theta[4, ]
        l <- l + y[1, 2] * log(xi.12)
        xi.13 <- q - xi.11 - xi.12
        l <- l + y[1, 3] * log(xi.13)
        xi.21 <- (1 - q) * theta[2, ] * theta[3, ]
        l <- l + y[2, 1] * log(xi.21)
        xi.22 <- (1 - q) * (1 - theta[2, ]) * theta[4, ]
        l <- l + y[2, 2] * log(xi.22)
        xi.23 <- 1 - q - xi.21 - xi.22
        l + y[2, 3] * log(xi.23)
}
```

```
# acceptance sampling of the nonresponse model for the Norwegian LFS
nores.acpt <- function(n = 3000, met = 1, ifl = 1.5, d.f = 3, store = T)
{
        y <- array(c(12881, 1829, 1158, 6726, 518, 796), c(2, 3))
        dimnames(y) <- list(c("x=1", "x=0"), c("y=1", "y=0", "nores"))
        q <- 0.613      # register employment rate
        theta <- c(c(0.559, 0.078)/c(q, 1 - q), 1 - c(0.029, 0.099))
        l.0 <- log.L(y, cbind(theta, theta), q)[1]      # initial para
        l.max <- sum(y * log(y/sum(y))) # global maximum log-L
        j.obs <- j.hat(y, theta, q)     # observed information
        sigma <- .solve(j.obs)  # asymptotic covariance matrix
        A <- chol(ifl * sigma)  # inflated Cholesky decomposition

        if(met == 1) { # multinormal N(theta,k*Sigma)
                z.0 <- array(rnorm(4 * n), c(4, n))     # std. normal
                z <- theta + t(A) %*% z.0       # sample transformation
                d.z <- dnorm(z.0[1,  ]) * dnorm(z.0[2,  ])      # pdf/J
                d.z <- d.z * dnorm(z.0[3,  ]) * dnorm(z.0[4,  ])
        }
        if(met == 2) { # multivariate student-t with d.f
                z.0 <- array(rt(4 * n, d.f), c(4, n))   # iid sample
                z <- theta + t(A) %*% z.0       # sample transformation
                d.z <- dt(z.0[1,  ], d.f) * dt(z.0[2,  ], d.f)  # pdf/J
                d.z <- d.z * dt(z.0[3,  ], d.f) * dt(z.0[4,  ], d.f)
                d.z <- d.z * dt(z.0[3,  ], d.f) * dt(z.0[4,  ], d.f)
        }
        idx <- rep(T, n)        # truncation of sample if necessary
        for(i in 1:4) {
                idx <- idx & z[i,  ] < 1 & z[i,  ] > 0
        }
        z <- z[, idx]
        d.z <- d.z[idx]
        m <- sum(idx)
        cat(m, "sample retained...\n variance ratio:\n")
        cat(diag(var(t(z)))/diag(sigma), "\n")  # sample charact.

        l <- log.L(y, z, q)     # sample log-likelihood
        L <- exp(l - max(l))    # standardized likelihood
        cat("sample m.l.e. =", z[, L == max(L)], "\n")
        w <- L/d.z
```

```
        a <- max(w)/mean(w)
        cat("(max_w, a, P[Accept]) =", c(max(w), a, 1/a), "\n")
        x.a <- w/max(w)
        accept <- runif(m, 0, 1) <= x.a
        cat("accept =", c(sum(accept), sum(accept)/m), "\n")
        z <- z[, accept]
        L <- L[accept]
        n <- sum(accept)
        print(sigma)    # covariance matrix of the target function
        print(var(t(z)))        # cov_matrix of the accepted sample
        if(store) {     # store the sample for detailed analysis
                sink("nores.sim")
                cat(z)
                sink()
        }
}
```

```
# independence-chain MC sampling of the nonresponse model ---
# mix = option for mixture algorithm & m = number of multiple chains
nores.idp <- function(n = 1000, mix = F, m = 10, ifl = 1.5, store = T)
{
        y <- array(c(12881, 1829, 1158, 6726, 518, 796), c(2, 3))
        dimnames(y) <- list(c("x=1", "x=0"), c("y=1", "y=0", "nores"))
        q <- 0.613      # register employment rate
        theta <- c(c(0.559, 0.078)/c(q, 1 - q), 1 - c(0.029, 0.099))
        l.0 <- log.L(y, cbind(theta, theta), q)[1]      # initial para
        l.max <- sum(y * log(y/sum(y))) # global maximum log-L
        j.obs <- j.hat(y, theta, q)     # observed information
        sigma <- .solve(j.obs)  # asymptotic covariance matrix
        A <- chol(ifl * sigma)  # inflated Cholesky decomposition
        w.x <- exp(l.0 - l.max) # improper current weight
        move <- n - 1           # move = counter of number of moves
        z.s <- array(theta, c(4, n))    # Markov chain sample

        for(k in 2:n) { # independence-chain MC sampling
                if(trunc(k/100) == k/100) {
                        cat(k, " ")
                }
                z.0 <- array(rnorm(4 * m), c(4, m))     # std. normal
                z <- theta + t(A) %*% z.0       # sample transformation
                d.z <- dnorm(z.0[1,  ]) * dnorm(z.0[2,  ])      # pdf/J
                d.z <- d.z * dnorm(z.0[3,  ]) * dnorm(z.0[4,  ])
                idx <- rep(T, m)        # truncation if necessary
                for(i in 1:4) {
                        idx <- idx & z[i,  ] < 1 & z[i,  ] > 0
                }
                z <- z[, idx]
                d.z <- d.z[idx]
                m <- sum(idx)

                l <- log.L(y, z, q)     # sample log-likelihood
                L <- exp(l - l.max)     # standardized likelihood
                w.z <- L/d.z    # weights of the candidates
                if(!mix) {
                        s <- 1          # the first chain
                }
                else {
```

```
                s <- sample(1:m, 1)      # random selection
        }

        w.z <- w.z[s]
        z <- z[, s]
        alpha <- min(1, w.z/w.x)         # acceptance probability
        stay <- runif(1, 0, 1) > alpha
        if(stay) {
                z.s[, k] <- z.s[, k - 1]
                move <- move - 1         # update number of moves
        }
        else {
                z.s[, k] <- z
                w.x <- w.z
        }
}

cat("\n move =", c(move, move/n), "\n")
print(sigma)    # cov_matrix of the target function
print(var(t(z.s[, (n/10):n])))  # cov_matrix of the MC sample
if(store) {     # store the sample for detailed analysis
        sink("nores.sim")
        cat(z.s)
        sink()
}
close.screen(all = T)   # graphical display of the sample path
split.screen(figs = c(4, 1))
for(i in 1:4) {
        screen(i)
        plot(1:n, z.s[i,  ], type = "l")
}
}
```

```
# random-walk-chain MC sampling of the nonresponse model ---
# mix = option for mixture algorithm & m = number of multiple chains
nores.rdw <- function(n = 1000, mix = F, m = 10, shr = 0.6, store = T)
{
        y <- array(c(12881, 1829, 1158, 6726, 518, 796), c(2, 3))
        dimnames(y) <- list(c("x=1", "x=0"), c("y=1", "y=0", "nores"))
        q <- 0.613      # register employment rate
        theta <- c(0.912, 0.2015, 1 - c(0.029, 0.099))
        l.0 <- log.L(y, cbind(theta, theta), q)[1]      # initial para
        l.max <- sum(y * log(y/sum(y))) # global maximum log-L
        j.obs <- j.hat(y, theta, q)     # observed information
        sigma <- .solve(j.obs)  # asymptotic covariance matrix
        A <- chol(shr * sigma)  # inflated Cholesky decomposition
        w.x <- exp(l.0 - l.max) # current weight
        move <- n - 1           # move = counter of number of moves
        x <- theta              # starting value
        z.s <- array(theta, c(4, n))    # Markov chain sample

        for(k in 2:n) { # random-walk-chain MC sampling
                if(trunc(k/100) == k/100) {
                        cat(k, " ")
                }
                z.0 <- array(rnorm(4 * m), c(4, m))     # std. normal
                z <- x + t(A) %*% z.0   # multi_N(theta,k*Sigma)
                idx <- rep(T, m)        # truncation if necessary
                for(i in 1:4) {
                        idx <- idx & z[i,  ] < 1 & z[i,  ] > 0
                }
                z <- z[, idx]
                m <- sum(idx)
                w.z <- exp(log.L(y, z, q) - l.max)      # sample log-L
                if(!mix) {
                        s <- 1          # the first chain
                }
                else {
                        s <- sample(1:m, 1)     # random selection
                }
                w.z <- w.z[s]
                z <- z[, s]
                alpha <- min(1, w.z/w.x)        # acceptance probability
```

```
                stay <- runif(1, 0, 1) > alpha
                if(stay) {
                        z.s[, k] <- x
                        move <- move - 1        # update number of moves
                }
                else {
                        z.s[, k] <- x <- z
                        w.x <- w.z
                }
        }
        cat("\n move =", c(move, move/n), "\n")
        print(sigma)
        print(var(t(z.s[, (n/10):n])))
        if(store) {
                sink("nores.sim")
                cat(z.s)
                sink()
        }
        close.screen(all = T)
        split.screen(figs = c(4, 1))
        for(i in 1:4) {
                screen(i)
                plot(1:n, z.s[i,  ], type = "l")
        }
}
```

```
# display of more detailed results (on p) of the MC sampling ---
# mixing (disp = 1) & profile-L (disp = 2) based on acceptance sampling
# (met = 1) & independence chain (met = 2) & random walk chain (met = 3)
disp.nrs <- function(fil = "nores.sim", n = 894, burn = 0, met = 1,
        disp = 1, q = 0.613, alpha = 0.95, p.lim = c(0.625, 0.647))
{
        y <- array(c(12881, 1829, 1158, 6726, 518, 796), c(2, 3))
        dimnames(y) <- list(c("x=1", "x=0"), c("y=1", "y=0", "nores"))
        z <- array(scan(fil), c(4, n))[, (burn + 1):n]
        n <- dim(z)[2]
        p <- q * z[1,  ] + (1 - q) * z[2,  ]
        txt <- c("Acceptance sampling", "Independence chain",
                "Random walk chain")[met]

        if(disp == 1) {
                plot(1:n, p, type = "l", ylab = "Sample",
                                xlab = "Iteration")
                title(txt)
        }
        else {
                l <- log.L(y, z, q)
                l.hat <- max(l)
                L <- exp(l - l.hat)     # sample likelihood
                mle <- p[L == max(L)]
                if(length(mle) > 1) {
                        mle <- mle[1]
                }
                cat("sample mle =", mle, "\n")
                p.l <- sort(p)[trunc((n * (1 - alpha))/2 + 0.5)]
                p.h <- sort(p)[trunc((n * (1 + alpha))/2 + 0.5)]
                cat("sample", 100 * alpha, "pct CI_p =",
                        c(p.l, p.h), "\n")

                L.p <- rep(0, n)        # approximate profile likelihood
                for(i in 1:n) {
                        if(trunc(i/100) == (i/100)) {
                                cat(i, " ")
                        }
                        x <- z
                        x[1,  ] <- (p[i] - (1 - q) * x[2,  ])/q
```

```
                l <- max(log.L(y, x, q))
                L.p[i] <- exp(l - l.hat)
        }
        l.hat <- max(L.p)       # standardizing
        L.p <- L.p/l.hat
        L <- L/l.hat
        mle <- p[L.p == max(L.p)]
        if(length(mle) > 1) {
                mle <- mle[1]
        }
        cat("\n", "approximate mle =", mle, "\n")
        L.a <- exp( - qchisq(alpha, 1)/2)
        diff <- abs(L.p - L.a)
        low <- p < mle
        p.l <- p[low][diff[low] == min(diff[low])]
        p.h <- p[!low][diff[!low] == min(diff[!low])]
        cat("profile", 100 * alpha, "pct CI_p =",
                c(p.l, p.h), "\n")

        plot(p, L, ylab = "Likelihood", xlim = p.lim,
                ylim = c(0, 1))
        points(sort(p), L.p[order(p)], type = "l", lty = 2)
        abline(v = mle, lty = 3)
        abline(h = L.a, lty = 3)
    }
}
```

3.3 Product of kernels

Mixture and cycle MH algorithm Suppose $P_1, ..., P_m$ are all Markov kernels with invariant distribution π. Tierney (1994)[9] showed that, under weak conditions, they can be combined to generate Markov chains with the same invariant π.

- In the *mixture* algorithm, one of the kernels will be slected at each updating step, according to some fixed probability, say, $a_1, ..., a_m$. This generates an irreducible and aperiodic chain if any of these kernels is irreducible and aperiodic.
- In the *cycle* algorithm, each kernel will be used in a cyclic order. The irreducibility and aperiodicity of the combined kernels can not be proved under the same weak condition as above, but need to be verified from case to case.

Block-at-a-time updating In another generalization of the MH algorithm, the current state is updated block-at-a-time or, simply, component-wise. Suppose q-partition of X, i.e.

$$X = (X_1, X_2, ..., X_q).$$

Let $P_1, ..., P_q$ be, respectively, transition kernels with invariant distribution $\pi(x_1|x_2, ..., x_q)$, $\pi(x_2|x_1, x_3, ..., x_q)$, ..., $\pi(x_q|x_1, ..., x_{q-1})$, i.e. the distribution of each part conditional to the rest parts. Each updating of the current x can be broken into q-updates corresponding to

$$X_1 \sim \pi(x_1|x_2, ..., x_q), \quad X_2 \sim \pi(x_2|x_1, x_3, ..., x_q), \quad ..., \quad X_q \sim \pi(x_q|x_1, ..., x_{q-1}).$$

The combined kernel generates a Markov chain with the invariant π. This is sometimes referred to as the *product of kernels principle*.

A complete split into component-wise updating may result into slow mixing chains, which sometimes can be improved by blocking together the highly correlated components. On the other hand, one may combine the kernels either systematically or randomly. To ensure irreducibility and aperiodicity, it is often enough if each block would be updated infinitely often as the chain evolves. Extra concern is required to preserve reversibility.

[9] Tierney, L. (1994). Markov chains for exploring posterior distributions (with discussions). *Ann. Statist.*, 22, 1701-62.

3.3.1 Gibbs sampler

Suppose p-component state vector, i.e.

$$X = (X_1, ..., X_p)^T$$

The *Gibbs sampler* updates the current state x component-wise, i.e.

- generate

$$\begin{aligned} x_1^* &\sim \pi(x_1|x_2, ..., x_p) \\ x_2^* &\sim \pi(x_2|x_1^*, x_3, ..., x_p) \\ &\vdots \\ x_p^* &\sim \pi(x_p|x_1^*, ..., x_{p-1}^*). \end{aligned}$$

- update $x = x^*$, and iterate.

The Gibbs sampler is a direct application of product of kernels principle, which ensures us that $\pi(x)$ is the invariant distribution of the resulting Markov chain, provided it remains irreducible and aperiodic.

Remark Instead of component-wise, the Gibbs sampler can also be applied block-at-a-time, so long as exact sampling from the conditional distributions is feasible.

3.3.2 Understanding the Gibbs sampler

Let p-component x and y differ only at, say, the i-th component, i.e.

$$x = (x_1, ..., x_{i-1}, x_i, x_{i+1}, ..., x_p) \qquad \text{and} \qquad y = (x_1, ..., x_{i-1}, x_i^*, x_{i+1}, ..., x_p).$$

We have

$$\psi(x, y) = \pi(x_i^* | x_1, ..., x_{i-1}, x_{i+1}, ..., x_p) \quad \text{and} \quad \psi(y, x) = \pi(x_i | x_1, ..., x_{i-1}, x_{i+1}, ..., x_p).$$

It follows that

$$\alpha(x, y) = \min\{\frac{\pi(y)\psi(y, x)}{\pi(x)\psi(x, y)},\ 1\} \equiv 1.$$

In other words, the Gibbs sampler is a special case of the MH algorithm, with combined component-wise transition kernels and unity acceptance probability.

Illustration The Gibbs sampling can be thought of as an application of the fact that conditional distributions jointly determine the joint distribution, provided the latter exists. Take, for instance, bivariate $x = (x_1, x_2)^T$, i.e.

$$\begin{aligned}
\pi_{x_1}(x_1) &= \int \pi(x_1|x_2)\pi_{x_2}(x_2)dx_2 \\
&= \int \pi(x_1|x_2)[\int \pi(x_2|z)\pi_{x_1}(z)dz]dx_2 \\
&= \int [\int \pi(x_1|x_2)\pi(x_2|z)dx_2]\pi_{x_1}(z)dz \\
&= \int h(x_1, z)\pi_{x_1}(z)dz.
\end{aligned}$$

This is known as the *fixed point integral equation*. It shows that the combined conditional kernels lead to an invariant distribution with the desired marginal distributions, provided they exist. Since conditional sampling is guaranteed by construction, we thereby obtain the joint target distribution. As an example, consider bivariate Bernoulli random variables with joint and conditional distributions:

$$\pi(x) = \begin{pmatrix} p_{00} & p_{01} \\ p_{10} & p_{11} \end{pmatrix},\ \pi(x_1|x_2) = \begin{pmatrix} \frac{p_{00}}{p_{00}+p_{10}} & \frac{p_{10}}{p_{00}+p_{10}} \\ \frac{p_{01}}{p_{01}+p_{11}} & \frac{p_{11}}{p_{01}+p_{11}} \end{pmatrix},\ \pi(x_2|x_1) = \begin{pmatrix} \frac{p_{00}}{p_{00}+p_{01}} & \frac{p_{01}}{p_{00}+p_{01}} \\ \frac{p_{10}}{p_{10}+p_{11}} & \frac{p_{11}}{p_{10}+p_{11}} \end{pmatrix}.$$

It is a straightforward exercise to check, say,

$$\pi_{x_1}(x_1) = \pi_{x_1}(x_1)[\pi(x_2|x_1)\pi(x_1|x_2)] \qquad \text{where} \quad \pi_{x_1}(x_1) = (p_{00} + p_{01},\ p_{10} + p_{11}).$$

3.3.3 Metropolis-Hastings Acceptance-Rejection (MH-AR)

When direct sampling from the conditional distributions is not possible, we need to apply some acceptance/rejection mechanism block-at-a-time, within each MH iteration. The following *Metropolis-Hastings Acceptance-Rejection (MH-AR) algorithm* secures the reversibility of the resulting Markov chain.

Let independence sampler

$$\psi(x, y) = \psi(y)$$

and a known constant c be such that $c\psi(x)$ does not necessarily dominate $\pi(x)$. Define

$$C = \{x;\ \pi(x) > c\psi(x)\},$$

which is not a probability-null set. At the current $X_i = x$,

- generate $u \in Unif(0, 1)$ independent of $y \sim \psi(y)$,
- let $w(x) = \pi(x)/\psi(x)$, and

$$\alpha(x, y) = \begin{cases} 1 & \text{if } x \notin C \\ c\psi(x)/\pi(x) & \text{if } x \in C \text{ and } y \notin C \\ \min\{1, w(y)/w(x)\} & \text{if } x, y \in C \end{cases}$$

- update $X_{i+1} = y$ if $u \leq \alpha(x, y)$ and $X_{i+1} = x$ otherwise.

The MH-AR algorithm has been designed to secure reversibility. No where in the derivation did we require that $\int \pi(x)dx = 1$; and we only need to know π upto some proportionality constant. In other words, we may apply the MH-AR algorithm to p, where $\pi(x) = p(x)/\int p(x)dx$ regardless of the unknown $\int p(x)dx$.

3.3.4 Understanding the MH-AR algorithm

Let A be the event

$$U \leq \pi(Y)/c\psi(Y), \quad \text{with unconditional probability } d = P[A].$$

The p.d.f. of Y conditional to A is given by

$$q(y) = \frac{\psi(y)}{d} \cdot \min\{1, \frac{\pi(y)}{c\psi(y)}\} = \begin{cases} \frac{\pi(y)}{cd} & \text{if } y \notin C \\ \frac{\psi(y)}{d} & \text{if } y \in C. \end{cases}$$

To ensure reversibility, we adjust $q(x)$ and $q(y)$ with suitable $\alpha(x, y)$ and $\alpha(y, x)$ for each x or y coming through such a pseudo-AR procedure, so that

$$\pi(x)q(y)\alpha(x, y) = \pi(y)q(x)\alpha(y, x).$$

We have,

- in case of $x, y \notin C$,

  $$\pi(x)q(y) = \pi(x)\pi(y)/(cd) = \pi(y)\pi(x)/(cd) = \pi(y)q(x),$$

  i.e. $\alpha(x, y) = \alpha(y, x) = 1$;

- in case of $x \in C$ and $y \notin C$,

  $$\pi(y)q(x) = \pi(y)\psi(x)/d < \pi(x)\pi(y)/(cd) = \pi(x)q(y),$$

  since $\psi(x) < \pi(x)/c$, i.e.

  $$\alpha(x, y) = c\psi(x)/\pi(x) \quad \text{and} \quad \alpha(y, x) = 1;$$

- in case of $x \notin C$ and $y \in C$,

  $$\alpha(x, y) = 1 \quad \text{and} \quad \alpha(y, x) = c\psi(y)/\pi(x);$$

- in case of $x, y \in C$,

  $$\alpha(x, y) = \begin{cases} w(y)/w(x) & \text{if } \pi(x)\psi(y) > \pi(y)\psi(x) \\ 1 & \text{otherwise.} \end{cases}$$

Summarizing the four cases gives us $\alpha(x, y)$ as defined earlier.

3.3.5 Example: Rat Growth Data

Gelfand *et al.* (1990)[10] applied the Gibbs sampler to a hierarchical model of "Rat Growth Data". The data contain weights of 30 young rats measured at 5 different time points, i.e. 8, 15, 22, 29 and 36 days after birth. Let y_{ij} denote the weight of the i-th rat at the j-th measurement, and x_j the time of measurement, for $1 \leq i \leq 30$ and $1 \leq j \leq 5$. We assume

$$Y_{ij} = \alpha_i + \beta_i x_j + \epsilon_{ij} \quad \text{where} \quad \epsilon_{ij} \sim N(0, \sigma^2) \text{ and } \begin{pmatrix} \alpha_i \\ \beta_i \end{pmatrix} \sim N(\begin{pmatrix} \alpha_c \\ \beta_c \end{pmatrix}, \Sigma).$$

Remark The model was motivated from an exploratory analysis of the data: the rats were born weighing different amounts, and they grow linearly but at different rates, and the variation across the rats appears to come from a symmetric distribution resembling the normal one.

Let $\mu_c = (\alpha_c, \beta_c)^T$ and $\theta_i = (\alpha_i, \beta_i)^T$. Let $y_i = (y_{i1}, .., y_{i5})^T$ and $\epsilon_i = (\epsilon_{i1}, .., \epsilon_{i5})^T$. We have

$$(Y_i - Z\mu_c) = Z(\theta_i - \mu_c) + \epsilon_i \quad \text{where} \quad Z = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ x_1 & x_2 & x_3 & x_4 & x_5 \end{pmatrix}^T.$$

In particular, Y_i has normal distribution, whose first two moments are given as

$$E(Y_i) = Z\mu_c \qquad \text{and} \qquad \Sigma_Y = E(Y_i - Z\mu_c)(Y_i - Z\mu_c)^T = Z\Sigma Z^T + \sigma^2 I.$$

The likelihood function is, let $\lambda^T = (\sigma^2, \mu_c^T, \Sigma)$,

$$L(\lambda; y) \propto \pi_{OBS}(\lambda) = |\Sigma_Y|^{-15} \exp\{-\frac{1}{2}\sum_{i=1}^{30}(Y_i - Z\mu_c)^T \Sigma_Y^{-1}(Y_i - Z\mu_c)\}.$$

Whereas if the θ_i were observed, the *(latent) likelihood* would have a simpler form, i.e.

$$\begin{aligned} L_{LAT}(\lambda; y, \theta) \propto \pi_{COM}(\lambda, \theta) = \prod_{i=1}^{30} \sigma^{-5} \exp\{-\frac{1}{2}\sigma^{-2}(y_i - Z\theta_i)^T(y_i - Z\theta_i)\} \\ \times |\Sigma^{-1}| \exp\{-\frac{1}{2}(\theta_i - \mu_c)^T \Sigma^{-1}(\theta_i - \mu_c)\}. \end{aligned}$$

In particular, $\theta = (\theta_1, ..., \theta_{30})$ *augment* the data (y_{ij}) in the sense that

$$(\lambda, \theta) \sim \pi_{COM} \quad \Rightarrow \quad \lambda \sim \pi_{OBS}, \quad \text{since} \quad \pi_{OBS}(\lambda) = \int \pi_{COM}(\lambda, \theta) d\theta.$$

[10]Gelfand, A.E., Hills, S.E., Racine-Poon, A. and Smith, A.F.M. (1990). Illustration of Bayesian inference in normal data using Gibbs sampling. *J. Am. Statist.*, **85**, 972-985.

Conditional distribution upto a proportional constant To construct the Gibbs sampler, we need to find the corresponding conditional distributions. Generally speaking, let $\lambda = (\theta, \xi)$ where θ is a scalar. To derive $\pi(\theta|\xi)$ from the joint $\pi(\theta, \xi)$, we notice that

$$\pi(\theta|\xi_0) = \frac{\pi(\theta, \xi_0)}{\int \pi(\theta, \xi_0) d\theta} = \frac{\pi(\theta, \xi_0)}{\pi(\xi_0)} \propto \pi(\theta, \xi_0),$$

so that the distribution of θ conditional on $\xi = \xi_0$ is proportional to the joint $\pi(\theta, \xi_0)$, i.e. fixed at $\xi = \xi_0$. In particular, direct sampling is feasible provided $\pi(\theta, \xi_0)$ is proportional to some familiar and standard distribution functions.

The convoluted form of π_{OBS} leads to non-standard conditional distributions, and thus complicates the Gibbs sampling. Whereas inspection of $\pi = \pi_{COM}$ gives us

$$\begin{aligned}
\pi(\mu_c|\theta, \sigma^2, \Sigma) &\propto \exp\{-\frac{1}{2}\sum_{i=1}^{30}(\theta_i - \mu_c)^T\Sigma^{-1}(\theta_i - \mu_c)\} \simeq N(\bar{\theta}, \Sigma/30) \\
\pi(\theta_i|\mu_c, \sigma^2, \Sigma) &\simeq N(D(Z^T y_i \sigma^{-2} + \Sigma^{-1}\mu_c), D) \quad \text{where } D = (Z^T Z\sigma^{-2} + \Sigma^{-1})^{-1} \\
\pi(\sigma^2|\mu_c, \theta, \Sigma) &\propto (1/\sigma^2)^{75} \exp\{-\frac{1}{2\sigma^2}\sum_{i=1}^{30}(y_i - Z\theta_i)^T(y_i - Z\theta_i)\} \\
&\simeq IG(75 - 1,\ \frac{1}{2}\sum_{i=1}^{30}(y_i - Z\theta_i)^T(y_i - Z\theta_i)) \\
\pi(\Sigma^{-1}|\sigma^2, \mu_c, \theta) &\propto |\Sigma^{-1}|^{15} \exp\{\frac{1}{2}Tr(\sum_{i=1}^{30}(\theta_i - \mu_c)(\theta_i - \mu_c)^T\Sigma^{-1})\} \\
&\simeq W(\{\sum_{i=1}^{30}(\theta_i - \mu_c)(\theta_i - \mu_c)^T\}^{-1},\ 33),
\end{aligned}$$

where IG denotes the inverse-gamma distribution, and $W(A, m)$ the Wishart distribution.

Implementation The Gibbs sampling consists now of iterations among these 4 conditional distributions, except that the conditional distribution of Σ^{-1} has been modified into

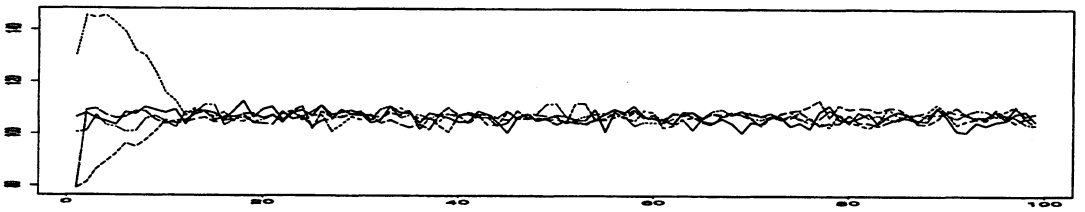
$$W(\{\sum_{i=1}^{30}(\theta_i - \mu_c)(\theta_i - \mu_c)^T + \rho R\}^{-1}, 33 + \rho) \quad \text{where } R = \begin{pmatrix} 100 & 0 \\ 0 & 0.1 \end{pmatrix} \text{ and } \rho = 2.$$

This is equivalent to introducing, for Σ^{-1}, the prior

$$\pi(\Sigma^{-1}) = W(R^{-1},\ \rho).$$

Without it the algorithm will have a tendency towards generating Σ^{-1} that are close to being singular, since its conditional distribution blows up around such matrices.

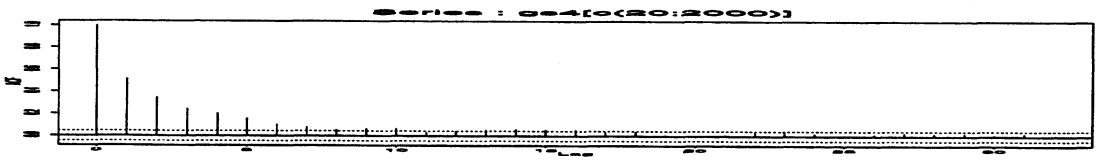
Diagnostic plots We have run the Gibbs sampling with five different starting points. Multiple starting points are helpful in assessing the convergence of the Markov chain. It also gives a more complete picture of the likelihood surface e.g. in case of multi-modality. With the same number of total iterations, multiple-chains are more likely to visit the different areas of the target distribution than a single chain.



The first 100 iterations for α_c from 5 different starting points.

The sample paths of the other parameters are similar. It is clear that even though the algorithm has been started at 5 very different points, the Markov chains arrived at the same area of the parameter space within about 20 iterations. Convergence towards the invariant distribution does not take a long time.

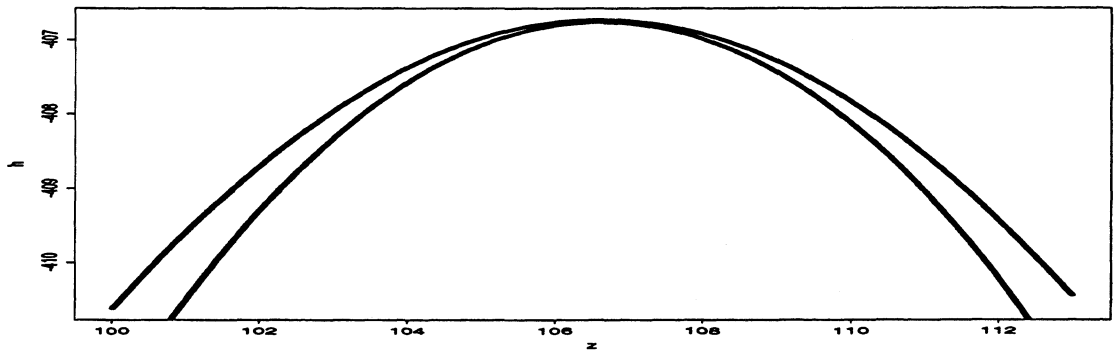
Likewise, the sample autocorrelations of the different parameters provide valuable information on the mixing of the chain, i.e. how fast it forgets its past. Quick mixing often implies quick convergence, as well as high efficiency. It was evident from figures like the one below (for α_c) that the autocorrelations decay quickly and the chain mixes well, which is consistent with the impression from the sample-path plot above.



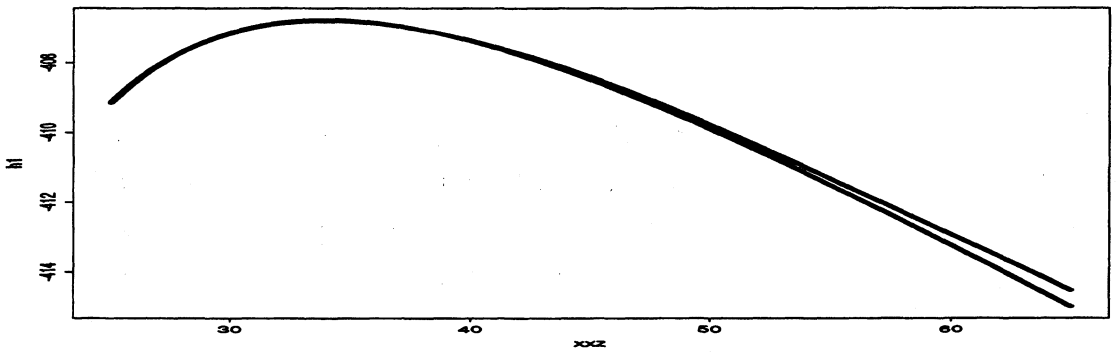
Sample autocorrelations of α_c with the Gibbs sampler

Histograms of the same parameter from different chains is also a useful device. Needless to say, convergence of the algorithm implies that these histograms should resemble each other. To save space we have not included the relevant plots. However, inspection of these showed similarity across chains.

Approximate profile log-likelihood Suppose we are interested in parameter ψ, and ξ contains the rest parameters. We compare the profile log-likelihood $l_P(\xi)$ with $l(\psi, \hat{\xi})$, where $\hat{\xi}$ denotes the m.l.e. of ξ. Notice that the curvature of $l(\psi, \hat{\xi})$ gives the variance of the normal approximation of ψ, and its mode the m.l.e. $(\hat{\psi}, \hat{\xi})$. Two plots are given below.



The approximate profile log-likelihood (the wider curve) of α_c and $l(\alpha_c; \hat{\xi})$ where $\hat{\xi}$ is the m.l.e. of the rest parameters.



The approximate profile log-likelihood (the wider curve) of σ^2 and $l(\sigma^2; \hat{\xi})$ where $\hat{\xi}$ is the m.l.e. of the rest parameters.

The profile log-likelihood of α_c is somewhat wider than $l(\alpha_c; \hat{\xi})$, where ξ denotes the rest parameters. Whereas the two curves are closer to each other in the case of σ^2. The approximate profile log-likelihoods appear smooth, which indicate their numerical accuracy. Now that l_P was based on 1000 sampled parameter values, the smoothing process required 1 million evaluations of the likelihood, which would be burdensome in Splus. However, the same amount of calculation took only about 7 minutes in C.

Generic Splus code of the Gibbs sampler

```
# Data is a 150*1 column vector consisting of the weight measurements
# & Z = design matrix & v = Sigma & s2 = sigma^2 & mu = mu_c.
# Furthermore: c2 is a 60*2 matrix consisting of 30 2*2-identity matrices
# stacked on top of each other & c5 of 30 5*5-identity matrices
# & id30 = 30*30-identity matrix & id5 = 5*5-identity matrix.

gibbs.rat <- function(m = 1000)          # m = number of iterations
{
        for(k in 2:m) {
# conditional sampling of theta
                h1 <- solve(t(Z) %*% Z/s2[k - 1] + v[k - 1,  ,  ])
                h2 <- kronecker(id30, h1 %*% t(Z)) %*% data/s2[k - 1]
                        + c2 %*% ( h1 %*% v[k - 1,  ,  ] %*% t(mu[k - 1,  ]))
                h3 <- t(rmultnorm(1, null60, kronecker(id30, h1)))
                theta[k, , ] <- matrix(h2 + h3, ncol = 2, nrow = 30, byrow = T)

# conditional sampling of mu
                h1 <- solve(v[k - 1,  ,  ])/30
                h2 <- cbind(mean(theta[k,  , 1]), mean(theta[k,  , 2]))
                mu[k,  ] <- rmultnorm(1, h2, h1)

# conditional sampling of sigma^2
                h1 <- (data - kronecker(id30, Z) %*% matrix(t(theta[k,  ,  ]),
                        ncol = 1, nrow = 60))
                s2[k] <- (0.5 * (t(h1) %*% h1))/rgamma(1, 74)

# conditional sampling of Sigma
                h2 <- cbind(0, 0)
                h3 <- cbind(theta[k,  , 1] - mu[k, 1], theta[k,  , 2]
                        - mu[k, 2 ])
                h1 <- solve(t(h3) %*% h3 + 2 * R)
                h4 <- rmultnorm(35, h2, h1)
                v[k,  ,  ] <- t(h4) %*% h4

# the log-likelihood
                h1 <- data - c5 %*% (Z %*% t(mu[k,  ]))
                h2 <- solve(Z %*% solve(v[k,  ,  ]) %*% t(Z) + id5 * s2[k])
                h3 <- t(h1) %*% kronecker(id30, h2) %*% h1
                like[k, 1] <- Re((15) * log(det(h2)) - 0.5 * h3)
        }
}
```

3.4 Convergence diagnostics

General, practical bounds on the convergence rate of the MH algorithm, or the Gibbs sampler, are rare in practice. A number of methods have been proposed to monitor the behavior of the Markov chain(s) using the sample outputs, sometimes together with other information available. When MC sampling is terminated based on such techniques, they are called the *convergence diagnostics*, which can be divided into two categories, depending on whether they are applied to single, or multiple chains.

Some of the simple single-chain techniques include e.g. plot of the sample path, sample autocorrelations of the same parameter, sample cross correlations between different parameters, *etc.*. Common to these methods is their largely visual character. As such they do not provide precise numeric assessments, but can be useful in detecting slow mixing of the chain. For instance, if the sample autocorrelations decrease slowly, or if there exist high cross correlations between different parameters.

In the remaining part of this section, we shall describe several more sophisticated single- as well as multiple-chain convergence diagnostics.

3.4.1 Geweke-Z

The following method is due to Geweke (1992)[11]. Suppose real-valued function $f(x)$:

- Divide the chain into two *windows*, the first of which contains the first $100\alpha\%$ of the sample, and the second the last $100\beta\%$.
- Calculate the sample average within each of the two windows, denoted by $\bar{f}_1$ and $\bar{f}_2$, and estimate their respective asymptotic variances, denoted by τ_1 and τ_2.
- We have, for sufficiently large $1-\alpha-\beta$, i.e. approximately independent $\bar{f}_1$ and $\bar{f}_2$,

$$Z = \frac{\bar{f}_1 - \bar{f}_2}{\sqrt{\tau_1 + \tau_2}} \quad \sim \quad N(0,1). \tag{3.2}$$

- Tail values of z suggests, at least, that the chain was not convergent in the first window.

Remark Sometimes $\alpha = 0.1$ and $\beta = 0.5$ are used as the default setting. If the test statistic $Z = z$ leads to rejection of convergence, one may throw away the first window and repeat the test for the remaining sample, and so on. This is also one of the reasons why α is chosen to be smaller than β.

Splus script

```
geweke.z <- function(f, alpha = 0.1, beta = 0.5, ini = 1, max.lag = 100)
{
        n <- length(f)   # the sample size
        w.1 <- f[1:round(n * alpha)]      # the first window
        tau.1 <- sigma.ini(w.1, max.lag)[ini]
                # ini = which initial sequence estimator?
        w.2 <- f[round(n * beta):n]       # the second window
        tau.2 <- sigma.ini(w.2, max.lag)[ini]
        z <- (mean(w.1) - mean(w.2))/sqrt(tau.1 + tau.2)
        list(Test.Z = z, F.z = pnorm(z))
}
```

[11] Geweke J. (1992). Evaluating the accuracy of sampling-based approaches to calculating posterior moments. In *Bayesian Statistics* **4**, (ed. J.M. Bernardo, J.O. Berger, A.P. Dawid, and A.F.M. Smith). Clarendon Press, Oxford, UK.

3.4.2 Raftery-Lewis-N

The *Raftery-Lewis-N*[12] method attempts to calculate the required number of iteration, denoted by N, such that, for a real function $Y(x)$,

$$P[F(y_\alpha) \in (\alpha - \text{Tol}, \alpha + \text{Tol})] = p \quad \text{where} \quad P[Y \leq y_\alpha] = \alpha. \tag{3.3}$$

In other words, the culmulative distribution of the α-quantile of Y is estimated to within an error bound of Tol with probability p. The triplet (α, Tol, p) determines the output of the Raftery-Lewis-N diagnostic, and has to be supplied by the user.

1. Thinning parameter Define process (Z_n) as

$$Z_n = \begin{cases} 1 & \text{if } Y_n \leq y_\alpha \\ 0 & \text{otherwise} \end{cases},$$

which is not necessarily a Markov chain though it has been derived from one. By taking every k-th state from (Z_n), denoted by

$$Z_n^{(k)} = Z_{1+(n-1)k},$$

we *thin out* the chain. The thinned chained becomes approximately Markov for sufficiently large k. The first step in Raftery-Lewis-N is, therefore, to determine the *thinning parameter* k.

Implementation Based on a test-run of the MC sampling, we may estimate y_α by the sample α-quantile. We then fit the first- and second-order autoregressive (AR) model for $(Z_n^{(k)})$ at $k = 1, 2, ...$, until we reach the smallest k, at which the first-order AR model is preferred to the second-order one. This gives us the estimated thinning parameter. Let $\hat{\sigma}^2$ be the estimated variance of the residuals under the AR model. Typically, non-Bayesian[13] model selection prefers the AR model which yields the least Akaike's information Criterion (AIC), i.e.

$$AIC(k) = \frac{1}{n}(-2\hat{L}_k + 2k) = \log \hat{\sigma}^2 + 2k/n.$$

[12]Raftery, A.E. and Lewis, S.M. (1995). Implementing MCMC. In *Markov Chain Monte Carlo in Practice*, eds. W.R. Gilks, S. Richardson and D.J. Spiegelhalter. Chapman and Hall.

[13]Raftery and Lewis (1995) suggested the Bayesian Information Criterion (BIC) as the method of model selection.

2. Burn-in length How many iterations does it take before the chain reaches the equilibrium? The answer gives us the *burn-in length*, denoted by M. The calculation is based on $(Z_n^{(k)})$. Let the transition matrix of the two-state Markov chain $(Z_n^{(k)})$ be

$$P = \begin{pmatrix} 1-a & a \\ b & 1-b \end{pmatrix} \quad \Rightarrow \quad \pi = (\pi_0,\ \pi_1) = (\frac{b}{a+b},\ \frac{a}{a+b}),$$

where π is the invariant distribution. The m-step transition matrix is given as

$$P^m = \begin{pmatrix} \pi_0 & \pi_1 \\ \pi_0 & \pi_1 \end{pmatrix} + \frac{\lambda^m}{a+b} \begin{pmatrix} a & -a \\ -b & b \end{pmatrix},$$

where $\lambda = 1-a-b$ is also the second, i.e. the smallest, eigenvalue of P. Suppose we require that $p_{ji}(m)$, for $i, j = 0, 1$, be within ϵ of π_i. We have,

$$\begin{cases} e_0 = (1,\ 0) \\ e_1 = (0,\ 1) \\ p_{ji}(m) = e_j P^m e_i^T \end{cases} \quad \Rightarrow \quad |\lambda|^m \leq \epsilon \, \frac{(a+b)}{\max(a,b)} = \gamma(a,b,\epsilon) \quad \Rightarrow \quad M = k \, \frac{\log \gamma(a,b,\epsilon)}{\log |\lambda|}.$$

Implementation Based on $(Z_n^{(k)})$, we estimate (a, b) by the corresponding sample frequencies. This gives us estimate of λ as $1-a-b$. The burn-in length M may then be estimated for any user-supplied ϵ.

3. MC sample size How many more iterations do we need to achieve the desired accuracy? The answer gives us the *sample size*, denoted by N. Let $(Z_n^{(k)})$ be derived from the sample α-quantile of Y, we have, given equilibrium of the chain,

$$\hat{P}[Y \leq \hat{y}_\alpha] = \bar{Z}_n^{(k)} = \frac{1}{n} \sum_{t=1}^{n} Z_n^{(k)} \quad \simeq \quad N(\alpha, \frac{(2-a-b)ab}{n(a+b)^3}).$$

Let $\Phi(\cdot)$ denote the standard normal C.D.F., requirement (3.3) is satisfied if

$$N = k \, \frac{(2-a-b)ab}{(a+b)^3} \{\text{Tol}^{-1} \Phi^{-1}(\frac{1}{2}(p+1))\}^2.$$

Implementation As a reasonable routine practice, Raftery and Lewis suggested applying the diagnostic to each parameter of interest twice, at $\alpha = 0.025$ and $\alpha = 0.975$. The tolerance of error can be set at $\text{Tol} = \frac{1}{2}\min(\alpha, 1-\alpha) = 0.0125$, and $p = 0.95$.

4. Convergence diagnostic in practice Having obtained a pilot sample, the Raftery-Lewis-N yields convergence diagnostics, which can be used to modify the settings of the MC sampling.

Pilot sample size The sample size N would be minimized if $(Z_n^{(1)})$ form an independent sample. This minimum sample size, denoted by N_{min}, can be used as the size of a pilot sample. More explicitly, independence implies that

$$a = 1 - b = \pi_1 = 1 - \alpha \quad \text{and} \quad M = 0 \quad \text{and} \quad k = 1,$$

so that, at the routine setting $\alpha = 0.025$, and Tol $= 0.0125$, and $p = 0.95$.

$$N_{min} = \frac{\alpha(1-\alpha)}{\text{Tol}^2}\{\Phi^{-1}(\frac{1}{2}(p+1))\}^2 = 600,$$

Relative efficiency of the MC sampling Let

$$I = \frac{M+N}{N_{min}},$$

which is 1 for independence sampling. The increment from the unity is due to the dependence in the sample. The value of I therefore measures the efficiency of the MC sampling relatively to the independence sampling. Low efficiency may be caused by high cross-correlations among the components of X, or slow mixing of the sampler, or sometimes bad starting values. In general, $I > 5$ indicates that the MC sampling is behaving poorly.

Splus script[14]

```
raf.lew.N <- function(y, thin = 5, alpha = 0.025, tol = 0.0125,
                p = 0.95, err = 0.01)
{
        phi <- qnorm((p + 1)/2)^2/tol^2
        N.min <- phi * alpha * (1 - alpha)
        n <- length(y)          # obtained sample size
        if(n < N.min) {
                cat(" Not big enough pilot sample to diagnostics!\n")
                break
        }
        y.a <- sort(y)[round(n * alpha)]        # sample alpha-quantile
        z <- 1 * (y <= y.a)             # deriving (0,1)-process Z
        k <- 0
        goon <- T
        while(goon & k < thin) {        # thin = upper limit of k
                k <- k + 1              # z.s = thinned (0,1)-process
                z.s <- array(z[1:trunc(n/k)], c(k, trunc(n/k)))[1, ]
                est <- ar.yw(z.s, aic = F, order = 2)
                goon <- est$aic[2] > est$aic[3]         # AIC-selection
        }
        if(goon) {
                cat(" Failed to obtain the first-order-Z!\n")
                break
        }
        s.1 <- z.s[ - length(z.s)] == 1
        s.2 <- z.s[-1] == 1     # the successive states
        a <- sum(s.1 & !s.2)/sum(s.1)   # a = p(1,0)
        b <- sum(!s.1 & s.2)/sum(!s.1)  # b = p(0,1)
        pi.0 <- b/(a + b)
        pi.1 <- a/(a + b)
        lambda <- 1 - a - b
        gamma <- (err * (a + b))/max(a, b)
        M <- trunc((k * log(gamma))/log(abs(lambda)) + 0.5)
        N <- trunc((k * phi * (2 - a - b) * a * b)/(a + b)^3 + 0.5)

        list(k = k, M = M, N = N, I = (M + N)/N.min)
}
```

[14]There exist several public programs for implementing the Raftery-Lewis-N convergence diagnostic. A Fortran program **gibbsit** can be obtained by sending e-mail message 'send gibbsit from general' to *statlib@stat.cmu.edu*. An S version may be obtained by sending the message 'send gibbsit from S' to the same address. An XLISP-STAT version is available at the URL http://ftp.stat.ucla.edu..

3.4.3 Gelman-Rubin-R

This multiple-chain method was first proposed by Gelman and Rubin (1992)[15]. The basic idea is to detect when the Markov chains have 'forgotten' their starting points, by comparing several independent chains with overly dispersed starting points.

Let $y = Y(x)$ be a summary statistic for which convergence is desired. Suppose m parallel, independent Markov chains each of the size n, giving us

$$y = \begin{pmatrix} y_{11} & y_{12} & \cdots & y_{1n} \\ y_{21} & y_{22} & \cdots & y_{2n} \\ \vdots & & & \\ y_{m1} & y_{m2} & \cdots & y_{mn} \end{pmatrix} \quad \text{and} \quad \bar{y}_i = \frac{1}{n}\sum_{j=1}^{n} y_{ij} \quad \text{and} \quad \bar{y} = \frac{1}{m}\sum_{i=1}^{m} \bar{y}_i,$$

and the between-sequence variance B and the within-sequence variance W as

$$B = \frac{n}{m-1}\sum_{i=1}^{m}(\bar{y}_i - \bar{y})^2 \quad \text{and} \quad W = \frac{1}{m}\sum_{i=1}^{m} s_i^2, \text{ where } s_i^2 = \frac{1}{n-1}\sum_{j=1}^{n}(y_{ij} - \bar{y}_i)^2.$$

Remark The number of chains is usually set at $2 \leq m \leq 10$. Some considerations are necessary in choosing overly dispersed starting points, i.e. $(y_{11}, ..., y_{m1})$, especially in high-dimensional problems. It is often helpful in this respect to locate as many local maximums as possible, if not all.

Unless the finite chains have covered all parts of the state space, W is likely an underestimate of $Var(Y)$, whereas the following estimator likely an overestimate, i.e.

$$V = \frac{n-1}{n}W + \frac{1}{n}B.$$

Define the *estimated potential scale reduction* as

$$\hat{R} = \frac{V}{W} \quad \xrightarrow{P} \quad 1, \text{ as } n \to \infty.$$

Remark In calculating $\hat{R}$, the first half of the obtained chains is routinely discarded as burn-in. However, it is certainly possible to apply the single-chain convergence diagnostics here in order to make choices which are more precise and, possibly, economic. Values of $\hat{R}$ larger than 1.2 are taken to indicate that more simulations are needed.

[15] Gelman, A. and Rubin, D.B. (1992). Inference from iterative simulation using multiple sequences (with discussions). *Statistical Science*, **7**, 457-511.

Splus script

```
gel.rub.R <- function(y, burn = 0.5)
{
        n <- dim(y)[2]                  # n = length of the chain
        y <- y[, round(n * burn + 1):n] # possible burn-in
        m <- dim(y)[1]                  # m = number of chains
        n <- dim(y)[2]                  # updating n
        y.i <- c(y %*% rep(1/n, n))     # within-sequence means
        s.2 <- diag(var(t(y)))          # within-sequence variances
        W <- mean(s.2)                  # underestimate of Var(Y)
        B <- n * var(y.i)               # between-sequence variance
        V <- (W * (n - 1))/n + B/n      # overestimate of Var(Y)

        list(R = V/W, V = V, W = W, B = B)
}
```

Recent publications in the series Documents

98/3: S. Holtskog: Residential Consumption of Bioenergy in China. A Literature Study

98/4 B.K. Wold: Supply Response in a Gender-Perspective, The Case of Structural Adjustments in Zambia. Tecnical Appendices

98/5 J. Epland: Towards a register-based income statistics. The construction of the Norwegian Income Register

98/6 R. Chodhury: The Selection Model of Saudi Arabia. Revised Version 1998

98/7 A.B. Dahle, J. Thomasen and H.K. Østereng (eds.): The Mirror Statistics Exercise between the Nordic Countries 1995

98/8 H. Berby: A Demonstration Data Base for Business Register Management. A data base covering Statistical Units according to the Regulation of the European Union and Units of Administrative Registers

98/9 R. Kjeldstad: Single Parents in the Norwegian Labour Market. A changing Scene?

98/10 H. Brüngger and S. Longva: International Principles Governing Official Statistics at the National Level: are they Relevant for the Statistical Work of International Organisations as well?

98/11 H.V. Sæbø and S. Longva: Guidelines for Statistical Metadata on the Internet

98/12 M. Rønsen: Fertility and Public Policies - Evidence from Norway and Finland

98/13 A. Bråten and T. L. Andersen: The Consumer Price Index of Mozambique. An analysis of current methodology – proposals for a new one. A short-term mission 16 April - 7 May 1998

98/14 S. Holtskog: Energy Use and Emmissions to Air in China: A Comparative Literature Study

98/15 J.K. Dagsvik: Probabilistic Models for Qualitative Choice Behavior: An introduction

98/16 H.M. Edvardsen: Norwegian Regional Accounts 1993: Results and methods

98/17 S. Glomsrød: Integrated Environmental-Economic Model of China: A paper for initial discussion

98/18 H.V. Sæbø and L. Rogstad: Dissemination of Statistics on Maps

98/19 N. Keilman and P.D. Quang: Predictive Intervals for Age-Specific Fertility

98/20 K.A. Brekke (Coauthor on appendix: Jon Gjerde): Hicksian Income from Stochastic Resource Rents

98/21 K.A.Brekke and Jon Gjerde: Optimal Environmental Preservation with Stochastic Environmental Benefits and Irreversible Extraction

99/1 E. Holmøy, B. Strøm and T. Åvitsland: Empirical characteristics of a static version of the MSG-6 model

99/2 K. Rypdal and B. Tornsjø: Testing the NOSE Manual for Industrial Discharges to Water in Norway

99/3 K. Rypdal: Nomenclature for Solvent Production and Use

99/4 K. Rypdal and B. Tornsjø: Construction of Environmental Pressure Information System (EPIS) for the Norwegian Offshore Oil and Gas Production

99/5 M. Søberg: Experimental Economics and the US Tradable SO_2 Permit Scheme: A Discussion of Parallelism

99/6 J. Epland: Longitudinal non-response: Evidence from the Norwegian Income Panel

99/7 W. Yixuan and W. Taoyuan: The Energy Account in China: A Technical Documentation

99/8 T.L. Andersen and R. Johannessen: The Consumer Price Index of Mozambique: A short term mission 29 November – 19 December 1998

99/9 L-C. Zhang: SMAREST: A Survey of SMall ARea ESTimation

99/10 L-C. Zhang: Some Norwegian Experience with Small Area Estimation

99/11 H. Snorrason, O. Ljones and B.K. Wold: Mid-Term Review: Twinning Arrangement 1997-2000, Palestinian Central Bureau of Statistics and Statistics Norway, April 1999

99/12 K.-G. Lindquist: The Importance of Disaggregation in Economic Modelling

99/13 Y. Li: An Analysis of the Demand for Selected Durables in China

99/14 T.I. Tysse and K. Vaage: Unemployment of Older Norwegian Workers: A Competing Risk Analysis

99/15 L. Solheim and D. Roll-Hansen: Photocopying in Higher Education

99/16 F. Brunvoll, E.H. Davila, V. Palm, S. Ribacke, K. Rypdal og L. Tangden: Inventory of Climate Change Indicators for the Nordic Countries. 93s.

99/17 P. Schøning, M.V. Dysterud og E. Engelien: Computerised delimitation of urban settlements: A method based on the use of administrative registers and digital maps. 17s.

Documents

PORTO BETALT
VED
INNLEVERINGA
P.P.
NORGE/NOREG

Tillatelse nr.
159 000/502

B *Returadresse:*
Statistisk sentralbyrå
Postboks 8131 Dep.
N-0033 Oslo

Statistics Norway
P.O.B. 8131 Dep.
N-0033 Oslo

Tel: +47-22 86 45 00
Fax: +47-22 86 49 73

ISSN 0805-9411

Statistisk sentralbyrå
Statistics Norway