The Econometrics of DSGE Models

Giuseppe Ragusa
Luiss University

EIEF
Lecture 4: Bayesian computations

April 18, 2016

Outline

- Importance sampling
- Monte Carlo Markov Chain
  - Metropolis-Hastings
  - Gibbs sampler
- Readings

Bayesian computations

- To conduct inference using a posterior we need to be able to study its distribution
- Typically we run into these four typical situations:
  - We can simulate from the joint distribution $\theta \sim p(\theta|y)$
  - We cannot simulate from $p(\theta|y)$, but we can simulate from conditionals

$$p(\theta_j|\theta_1, \ldots, \theta_{j-1}, \theta_{j+1}, \ldots, \theta_J, y), \quad j = 1, \ldots, J$$

  - We cannot simulate from $p(\theta|y)$ or $p(\theta_j|\theta_1, \ldots, \theta_{j-1}, \theta_{j+1}, \ldots, \theta_J, y)$.

* *Simulate* here means that we can obtain draws from the distribution from well established algorithms

Bayesian computations

- We can *simulate* from the joint distribution $\theta \sim p(\theta|y)$
  - Very rare, practically never in real applications — at the best we are able to *evaluate* the joint density
- We cannot simulate from $p(\theta|y)$, but we can simulate from conditionals

$$p(\theta_j|\theta_1, \dots, \theta_{j-1}, \theta_{j+1}, \dots, \theta_J, y), \quad j = 1, \dots, J$$

  - This is a more common situation: we have already see this happening for the linear model with conjugate priors

$$\beta|\sigma^2, y \sim N(\tilde{\beta}, s^2 A_T^{-1})$$
$$\sigma^2|y\sigma \sim \Gamma^{-1}(\nu_T/2, \nu_T \sigma_T^2/2)$$

- We cannot simulate from $p(\theta|y)$ or $p(\theta_j|\theta_1, \dots, \theta_{j-1}, \theta_{j+1}, \dots, \theta_J, y)$
  - In this case we can only evaluate the joint density up to the proportionality constant, e.g., the likelihood from the linearized DSGE model

$$p(\theta|y) \propto \prod_{t=1}^{T} \phi(y_t; \mu(\theta), \Omega(\theta)) p(\theta)$$

Simulation techniques

- We will briefly see the following “simulation methods”
  - Monte Carlo integration
  - Importance Sampling
  - Monte-Carlo Markov Chain
    - Metropolis-Hastings
    - Gibbs sampler

Monte Carlo integration

- This techniques requires to be able to easily simulate from the *joint* distribution $p(\theta|y)$
- It can calculate *moments* of posteriors, that is,

$$\bar{h} = \int h(\theta)p(\theta|y)d\theta,$$

where $h(\theta)$ is a function which is often called a "*test function*"

  - $h(\theta) = \theta$, $\bar{h}$ is the posterior mean
  - $h(\theta) = \theta^2$, $\bar{h}$ is the uncentered second moment of the posterior
  - $h(\theta) = 1(\theta < u)$, $\bar{h}$ is $\int_{-\infty}^{u} p(\theta|y)d\theta$
  - $h(\theta) = 1(\theta \in A)$, $\bar{h}$ is $\int_A p(\theta|y)d\theta$

Strong Law of Large Numbers (SLLN)

Let $X_1, X_2, \ldots$ be a sequence of independent and identically distributed random variables, each having a finite mean $\mu = \int x_i f(x_i) dx_i$, where $f(x_i)$ is the common density of X_i.

Then with probability 1

$$\frac{X_1 + X_2 + \ldots + X_S}{S} \to \int x_i f(x_i) dx_i$$

This also works with variances and other quantities of interest, since a function of i.i.d. random variables are also i.i.d. random variables.

Monte Carlo integration, ctd

Main idea

- Sample $\theta_s \sim p(\theta|y)$, then,
  - if $\int |h(\theta)|p(\theta|y)d\theta < \infty$, we have, by the SLLN, that w.p.1.

$$\frac{1}{S}\sum_{s=1}^{S} h(\theta_s) \to \int h(\theta)p(\theta|y)d\theta$$

  - if $\int h(\theta)^2 p(\theta|y)d\theta < \infty$, we have by the Central Limit Theorem

$$\sqrt{S}\left[\frac{1}{S}\sum_{s=1}^{S} h(\theta_s) - \int h(\theta)p(\theta|y)d\theta\right] \xrightarrow{d} N(0, V_h),$$

$$V_h = E\left[\left(h(\theta) - \int h(\theta)p(\theta|y)d\theta\right)\left(h(\theta) - \int h(\theta)p(\theta|y)d\theta\right)'\right]$$

Monte Carlo integration, ctd

- Very easy to implement, once $p(\theta|y)$ is available
- e.g.,

$$\theta_1, \theta_2 | y \sim N(0, I_2)$$

Pseudocode

```
for $s \leq S$ do
    Draw $\theta_s$ from $p(\theta|y)$
end for
$\bar{h} \leftarrow \frac{1}{S}\sum_{s=1}^{S} h(\theta_s)$
```

Julia Code

```julia
S = 5000;
theta_s = randn(S,2);
mean_post = mean(theta_s, 1)
cdf = mean(theta_s.<1.96, 1)
set = mean(broadcast(&, ↩
    theta_s.<1.96, theta_s↩
    .>-1.96), 1)
```

- In practice, we never know how to simulate from directly from $p(\theta|y)$.....

Importance Sampling

The objective is the same as before

$$\bar{h} = \int h(\theta)p(\theta|y)d\theta.$$

Suppose that

- we can evaluate the density up to a proportionality constant; that is, given

  $$p(\theta|y) = \gamma(\theta)/Z$$

  we can evaluate $\gamma(\theta)$ but not Z
- e.g., robust linear regression model (Zellner)

$$p(\beta,\sigma^2|y) \propto \underbrace{L(y,\beta,\sigma)}_{\text{student's } t \text{likelihood}} \times \underbrace{N(\beta_0,\sigma^2 A_0^{-1})}_{\text{Normal prior}}\underbrace{\Gamma^{-1}(\nu_0/2,\sigma_0^2/2)}_{\text{Inverted }\Gamma\text{prior}}$$

Importance Sampling

Mean idea:

- We have a distribution $g(\theta)$ from which we can easily simulate
- Since

$$\begin{aligned}\bar{h} &= \int h(\theta)(\gamma(\theta)/Z)d\theta \\ &= \frac{\int h(\theta)\gamma(\theta)d\theta}{Z} \\ &= \frac{1}{Z}\int h(\theta)\frac{\gamma(\theta)}{g(\theta)}g(\theta)d\theta \\ &= \frac{1}{Z}\int w(\theta)\,h(\theta)\,g(\theta)d\theta, \quad w(\theta) = \frac{\gamma(\theta)}{g(\theta)}.\end{aligned}$$

- We approximate

$$\bar{h}_S = \frac{\frac{1}{S}\sum_{s=1}^{S} w(\theta_s)h(\theta_s)}{\frac{1}{S}\sum_{s=1}^{S} w(\theta_s)},$$

where $\{\theta_1,\ldots\theta_S\}$ is a sample from $g(\theta)$

Importance sampling, ctd.

If $\int |h(\theta)| p(\theta|y) d\theta < \infty$, then

$$\frac{\frac{1}{S}\sum_{s=1}^{S} w(\theta_s) h(\theta_s)}{\frac{1}{S}\sum_{s=1}^{S} w(\theta_s)} \xrightarrow{p} \int h(\theta) p(\theta|y) d\theta$$

- The choice of the importance density $g(\theta)$ is key for the performance of the sampler
  - the closer is $\gamma(\theta)/Z$ to $g(\theta)$ the better will the algorithm will work
  - if $\gamma(\theta)/g(\theta)$ is unbounded then the variance of $\bar{h}_S$ is infinite: $g(\theta)$ must have thicker tails than $\gamma(\theta)/Z$
- A starting point for $g(\theta)$ is to use the asymptotic approximation of the posterior; thus,

$$\theta_s \sim N(\hat{\theta}^{MLE}, -(TH)^{-1})$$

- Requiring fat tails we can use

$$\theta_s \sim MT(\hat{\theta}, -(TH)^{-1}, \nu)$$

Importance sampling, cts

Algorithm 1 Importance sampling algorithm

```
**for** $s \leq S$ **do**
    Draw $\theta_s$ from $g$
    $w_s \leftarrow \gamma(\theta_s)/g(\theta_s)$
    $w_s \leftarrow w_s/\sum_{s=1}^{S} w_s$
**end for**
$\bar{h}_S \leftarrow \frac{1}{S}\sum_{s=1}^{S} h(\theta_s)w_s$
```

Importance sampling

Consider simulating from $\theta \sim N(0,1)$ with importance distribution $g(\theta) = \mathscr{T}_3(\theta)$.

Julia code

```julia
S = 5000;      ## Number of simulations
using Distributions
t3 = TDist(3) ## Importance
sn = Normal()
theta_s = rand(t3, S, 1); # Draw from importance
w = pdf(sn, theta_s)./pdf(t3, theta_s);
w = w./sum(w);
mean_target = sum(w.*theta_s);
pr_target = sum(w.*broadcast(&, theta_s.<1.96, theta_s.>-1.96));
println("Mean of target density: ", mean_target)
println("Pr(-1.96<theta<1.96) target density: ",pr_target)
```

Importance sampling

Bad behavior

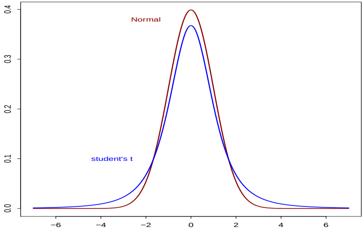
We consider now simulating from

$$\theta \sim \mathscr{T}_3(\theta)$$

with importance distribution

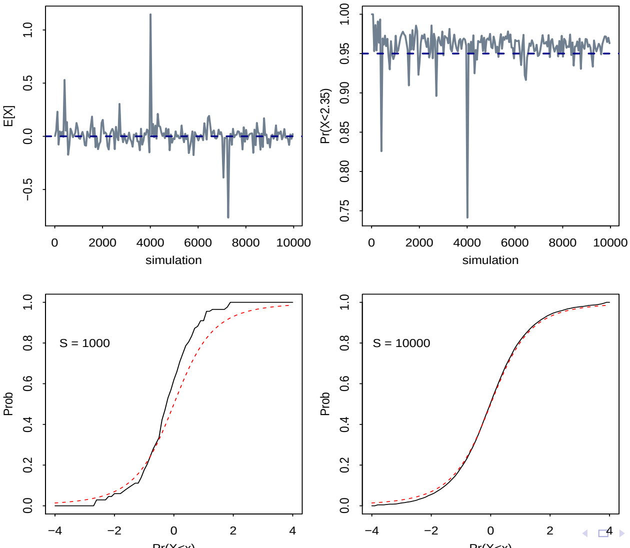
$$g(\theta) = N(0,1)$$

Notice that in this case the *target* has thinner tails than the *importance:*



Importance sampling

Bad behavior



Importance sampling

Application

Consider the robust linear model of Zellner

$$
\begin{aligned}
p(\beta,\sigma^2|y) &\propto \underbrace{\prod_{t=1}^{T}\left\{\Gamma\left(\frac{1}{2}\right)\Gamma\left(\frac{\nu}{2}\right)\sigma\right\}^{-1}\left[1+\frac{(y-x_t\beta)^2}{\nu\sigma^2}\right]^{-(\nu+1)/2}}_{\text{student's } t\text{likelihood}} \times \underbrace{\sigma^{-1}}_{\text{improper prior}} \\
&\propto \sigma^{-(T+1)}\prod_{t=1}^{T}\left[1+\frac{(y_t-x_t\beta)^2}{\nu\sigma^2}\right]^{-(\nu+1)/2} \\
&= \exp\left\{-(T+1)\log\sigma^2-\frac{(\nu+1)}{2}\sum_{t=1}^{T}\log\left[1+\frac{(y_t-x_t\beta)^2}{\nu\sigma^2}\right]\right\}
\end{aligned}
$$

We apply the Importance Sampling to this model.

Acceptance/Rejection Algorithm

- Let g be a density that can be simulated by some known method, and suppose there exists a constant c such that $\gamma(\theta) \leqslant c\,g(\theta)$ for all θ. To obtain a sample from $p(\theta|y) = \gamma(\theta)/Z$

Algorithm 2 Acceptance/Rejection sampling

```
for $s \leq S$ do
    repeat
        Generate $X$ from $g$
        Generate $U$ from $U[0,1]$
    until $U \leqslant \gamma(X)/c\,g(X)$
    $\theta_s \leftarrow X$
end for
```

Monte Carlo Markov Chain

- Method of sampling a target probability distribution by constructing Markov Chain such that the target is the stationary distribution of the chain
- Our task is to find a chain that *converges* to the target distribution
- Large statistical/probabilist literature has traditionally focused on the opposite problem: given a chain, find the stationary distribution
- Markov chain theory is very difficult to master, but the algorithms that results from it are relatively easy to apply

What is a Markov Chain?

Definition: a stochastic process in which future states are independent of past states given the present state

Stochastic process: a consecutive set of random (not deterministic) quantities defined on some known state space Θ

- think of Θ as the parameter space.
- consecutive implies a time component, indexed by t.

Consider a draw of $\theta^{(t)}$ to be a state at iteration t. The next draw $\theta^{(t+1)}$ is dependent only on the current draw θ_t, and not on any past draws.
This satisfies the Markov property:

$$p(\theta^{(t+1)}|\theta^{(1)}, \theta^{(2)}, \theta^{(t)}) = p(\theta^{(t+1)}|\theta^{(t)})$$

Transition kernel

For discrete state space (k possible states): a k × k matrix of transition probabilities.

Example: Suppose k = 3. The 3 × 3 transition matrix **P** would be

$$\mathbf{P} = \begin{pmatrix} p(\theta_A^{(t+1)}|\theta_A^{(t)}) & p(\theta_B^{(t+1)}|\theta_A^{(t)}) & p(\theta_C^{(t+1)}|\theta_A^{(t)}) \\ p(\theta_A^{(t+1)}|\theta_B^{(t)}) & p(\theta_B^{(t+1)}|\theta_B^{(t)}) & p(\theta_C^{(t+1)}|\theta_B^{(t)}) \\ p(\theta_A^{(t+1)}|\theta_C^{(t)}) & p(\theta_B^{(t+1)}|\theta_C^{(t)}) & p(\theta_C^{(t+1)}|\theta_C^{(t)}) \end{pmatrix}$$

where the subscripts index the 3 possible values (A, B, C) that θ can take.

The rows sum to one and define a conditional probability mass function, conditional on the *current* state. The columns are the marginal probabilities of being in a certain state in the next period.

For continuous state space (infinite possible states), the transition kernel is a bunch of conditional PDFs: $f(\theta_i^{(t+1)}|\theta_j^{(t)})$.

The working of a Markov chain

1. Define a starting distribution $\Pi^{(0)} = (p(\theta_A), p(\theta_B), p(\theta_C))$
2. At iteration 1, the distribution $\Pi^{(1)}$ from which $\theta^{(1)}$ is drown, is given

$$\Pi^{(1)} = \Pi^{(0)}\mathbf{P}$$

3. Similarly, at iteration 2, $\Pi^{(2)}$—from which $\theta^{(2)}$ is drown—is given

$$\Pi^{(2)} = \Pi^{(1)}\mathbf{P}$$

4. At iteration t, $\Pi^{(t)}$ from which $\theta^{(t)}$ is given

$$\Pi^{(t)} = \Pi^{(t-1)}\mathbf{P}$$

Stationary Distribution

- Define a stationary distribution Π to be some distribution Π such that $\Pi = \Pi\mathbf{P}$
- Typically, at least for all the algorithms used in Bayesian statistics, the Markov chain will typically converge to Π regardless of our starting points.

Idea

Devise a Markov chain whose stationary distribution Π is the desired posterior distribution $p(\theta|y)$. We can then run this chain to get draws that are approximately from $p(\theta|y)$ — once the chain has converged.

Both the **Metropolis Hastings algorithm** and the **Gibbs sampler** are Markov Chain designed in such a way to have a limiting Π that is equal to the posterior distribution $p(\theta|y)$.

MCMC: Metropolis-Hastings algorithm

The Metropolis Algorithm follows the following steps:

1. Choose a starting value $\theta^{(0)}$
2. At iteration t, draw a candidate θ^* from a proposal distribution $q(\theta^*|\theta^{(t-1)})$, e.g. $N(\theta^{(t-1)}, \tau\Sigma)$
3. Compute an acceptance ratio (probability):

$$r = \min\left\{\frac{\gamma(\theta^*)/q(\theta^*|\theta^{(t-1)})}{\gamma(\theta^{(t-1)})/q(\theta^{(t-1)}|\theta^*)}, 1\right\}$$

4. Accept θ^* as $\theta^{(t)}$ with probability r. If r is not accepted, then $\theta^{(t)} = \theta^{(t-1)}$
5. Repeat step 2-4 S times to get S draws from $p(\theta|y)$

MCMC: The Metropolis-Hastings

Algorithm 3 Metropolis algorithm

```
Choose an initial value $\theta^(0)$.
for $t = 1 : S$ do
    Draw $\theta^*$ from $q(\cdot|\theta^{(t-1)})$
    Calculate $r = \min\left\{\frac{\gamma(\theta^*)/q(\theta^*|\theta^{(t-1)})}{\gamma(\theta^{(t-1)})/q(\theta^{(t-1)}|\theta^*)}, 1\right\}$
    Draw $u \sim \text{Uniform}(0, 1)$
    if $u \leqslant r$ then
        $\theta^{(t)} = \theta^*$
    else
        $\theta^{(t)} = \theta^{(t-1)}$
    end if
end for
```

Step 1: Choose starting value $\theta^{(0)}$

This is equivalent to drawing from our initial stationary distribution

The important thing to remember is that $\theta^{(0)}$ must have positive probability

$$p(\theta^{(0)}|y) > 0$$

Otherwise, we are starting with a value that cannot be drawn.

A common practice is to start the chain at the maximum a posterior, that is, the value that solves

$$\max p(y|\theta)p(\theta).$$

Burn-in

- Since convergence usually occurs regardless of our starting point, we can usually pick any feasible (for example, picking starting draws that are in the parameter space) starting point.
- However, the time it takes for the chain to converge varies depending on the starting point. As a matter of practice, most people throw out a certain number of the first draws, known as the burn-in. This is to make our draws closer to the stationary distribution and less dependent on the starting point.
- However, it is unclear how much we should burn-in since our draws are all slightly dependent and we don't know exactly when convergence occurs.

Step 2: Draw θ^* from $q(\theta^*|\theta^{(t-1)})$

The proposal distribution $q(\theta^*|\theta^{(t-1)})$ determines where we move to in the next iteration of the Markov chain (analogous to the transition kernel). **The support of the jumping distribution must contain the support of the posterior.**

The original **Metropolis algorithm** required that $q(\theta^*|\theta^{(t-1)})$ be be a symmetric distribution (such as the normal distribution), that is

$$q(\theta^*|\theta^{(t-1)}) = q(\theta^{(t-1)}|\theta^*)$$

and thus the acceptance ratio would simplify as

$$r = \min\left\{\frac{\gamma(\theta^*)}{\gamma(\theta^{(t)})}, 1\right\}.$$

Step 2: Draw θ^* from $q(\theta^*|\theta^{(t-1)})$, ctd.

If the proposal distribution does not depend on $\theta^{(t-1)}$, $q(\theta^*|\theta^{(t-1)}) = q(\theta^*)$ then we have what is known as **independent Metropolis-Hastings sampling**.

With this proposal, the candidate draws θ^* are drawn from the same distribution, regardless of what the previous draw was

This can be extremely efficient or extremely inefficient, depending on how close the proposal distribution is to the posterior.

Generally speaking, chain will behave well only if the proposal distribution has heavier tails than the posterior.

Step 3: Compute acceptance ratio r

$$r = \min\left\{\frac{\gamma(\theta^*)/q(\theta^*|\theta^{(t-1)})}{\gamma(\theta^{(t-1)})/q(\theta^{(t-1)}|\theta^*)}, 1\right\}$$

In the case where the proposal distribution is symmetric,

$$r = \min\left\{\frac{\gamma(\theta^*)}{\gamma(\theta^{(t-1)})}, 1\right\}$$

If our candidate draw has **higher** probability than our current draw $(r = 1)$, then our candidate is better so we definitely **accept it**. Otherwise, our candidate is accepted according to the ratio of the probabilities of the candidate and current draws.

Note that since r is a ratio, we only need $p(\theta|y)$ up to a constant of proportionality since $p(y)$ cancels out in both the numerator and denominator.

In the case where the proposal distribution is not symmetric,

$$r = \min \left\{ \frac{\gamma(\theta^*)/q(\theta^*|\theta^{(t-1)})}{\gamma(\theta^{(t-1)})/q(\theta^{(t-1)}|\theta^*)}, 1 \right\}$$

We need to weight our evaluations of the draws at the posterior densities by how likely we are to draw each draw.

For example, if we are very likely to jump to some θ^* , then $q(\theta^*|\theta^{(t-1)})$ is likely to be high, so we should accept less of them than some other θ^* that we are less likely to jump to.

Step 4: Decide whether to accept θ^*

Accept θ^* as $\theta^{(t)}$ with probability r. If θ^* is not accepted, then $\theta^{(t)} = \theta^{(t-1)}$

1. For each θ^* , draw a value u from the Uniform$(0, 1)$ distribution.
2. If $u < r$, accept θ^*. Otherwise use $\theta^{(t-1)}$.

The acceptance probability is

$$\Pr(u < r) = \int_0^r dx = r$$

Candidate draws with higher density than the current draw are always accepted.

Acceptance Rates

- It is important to monitor the acceptance rate (the fraction of candidate draws that are accepted) of your Metropolis-Hastings algorithm.
- If your acceptance rate is too high, the chain is probably not *mixing* well (not moving around the parameter space quickly enough).
- If your acceptance rate is too low, your algorithm is too inefficient (rejecting too many candidate draws).
- What is too high and too low depends on your specific algorithm, but generally
  - random walk: somewhere between 0.25 and 0.50 is *recommended*
  - independent: something close to 1 is preferred

Monitoring convergence

- A key issue in the successful implementation of Metropolis-Hastings or any other MCMC sampler (such as the Gibbs sampler) is the number of runs (steps) until the chain approaches stationarity
- A poor choice of starting values and/or proposal distribution can greatly increase the required burn-in time, and an area of much current research is whether an optimal starting point and proposal distribution can be found.
- While sometime convergence can be assessed visually by looking at the **traceplot** of the sampler, here are statistical tools that can be used to asses whether the chain (after discarding the burn-in sample) has converged or not.
- A chain is said to be **poorly mixing** if it says in small regions of the parameter space for long periods of time, as opposed to a **well mixing** chain that seems to happily explore the space.

The proposal distribution

- Given ε drawn from a given distribution Q, the proposal can be obtained

$$\underbrace{\theta^* = \theta^{(t-1)} + \tau\Sigma^{1/2}\varepsilon,}_{\text{random walk}} \qquad \underbrace{\theta^* = \bar{\theta} + \tau\Sigma^{1/2}\varepsilon}_{\text{independent}}$$

  where τ is a scalar tuning parameter and $\Sigma = \Sigma^{1/2}\Sigma^{1/2}$ p.d. matrix.
- Common choices for Q are the normal and t-student
- In theory, the parameter $(\bar{\theta}, \tau, \Sigma)$ do not matter....... in practice they do
  - $\bar{\theta}$ (for the independent MH) is the maximizer of the log-posterior
  - Σ is usually equal to the negative inverse of the hessian of the log-posterior evaluated at the maximum
  - τ is set in such a way to have a desired acceptance level
- Adaptive versions of the Metropolis algorithm exists

MCMC: The Metropolis algorithm

Application

We want to sample from:

$$p(\beta,\sigma^2|y) \propto \underbrace{\prod_{t=1}^{T}\left\{\Gamma\left(\frac{1}{2}\right)\Gamma\left(\frac{\nu}{2}\right)\sigma\right\}^{-1}\left[1+\frac{(y-x_t\beta)^2}{\nu\sigma^2}\right]^{-(\nu+1)/2}}_{\text{student's } t\text{likelihood}} \times \underbrace{\sigma^{-1}}_{\text{improper prior}}$$

$$\propto \sigma^{-(T+1)}\prod_{t=1}^{T}\left[1+\frac{(y_t-x_t\beta)^2}{\sigma^2\nu}\right]^{-(\nu+1)/2}$$

with proposal density:

$$q(\theta^*|\theta^{(t-1)}) \propto \exp\left\{-\frac{1}{2}(\theta^*-\theta^{(t-1)})'(\tau\Sigma)^{-1}(\theta-\theta^{(t-1)})\right\}$$

MCMC: The Metropolis algorithm

Application: Robust linear model

```
loglik = function(theta,y, X, nu)
    ## Calculate the likelihood of the robust linear model
    T, k  = size(X);
    beta  = theta[1:k];
    # Notice parameter transformation
    sigma = exp(theta[k+1]);
    r = y-X*beta;
    s = nu*sigma^2;
    tmp = log(1+(r.^2)/s);
    loglik = -(T+1)*log(sigma) -((nu+1)/2)*sum(tmp);
    return loglik
end
```

```
metrop = function(sim, y, X, nu)
    # Metropolis algorithm for the linear model with T-student error
    # -- parameters
    #  sim: number of simulations
    #  y,X: data
    #   nu: degrees of freedom of the t-student distributiosn
    # -- output
    # A matrix of dimension k x sim, where k is equal to the number of
    # column plu one. THe first k rows contain MCMC draws for beta,
    # while the last has draws for log(sigma)
    T, k = size(X);
    beta_ols = X\y; # equivalent to (X'X)^-1 X'y
    s_ols    = ((y-X*beta_ols)'*(y-X*beta_ols)/(T-k)).^(1/2);
    theta0    = [beta_ols; log(s_ols)]; # log(sigma) is sampled
    ## Find maximum a posterior and hession of log-posterior at the minimum.
    f(beta)  = -loglik(beta, y, X, nu)
    res      = optimize(f, theta0, method=:bfgs)
    Sigma = inv(hessian(f, res.minimum))
    ## Setup N(0, Sigma) distribution
    NormDist = MvNormal(zeros(k+1), Sigma)
    ## Initialization
    theta      = zeros(k+1, sim+1);
    gamma_s    = loglik(res.minimum, y, X, 3);
    theta[:,1] = res.minimum
    for s=1:sim
```

```
% Number of draws
S = 50000;

% Generate "fake" linear model data
% set sedd to replicate
rng(1245);
T = 200;
k = 2;
nu = 3;
T3 = TDist(3)
y = rand(T3, T,1);
X = [ones(T,1), randn(T, (k-1))];

% Call metropolis algorithm
chain = metropol(S, y, X, nu);
```

Robust linear model

OLS estimates

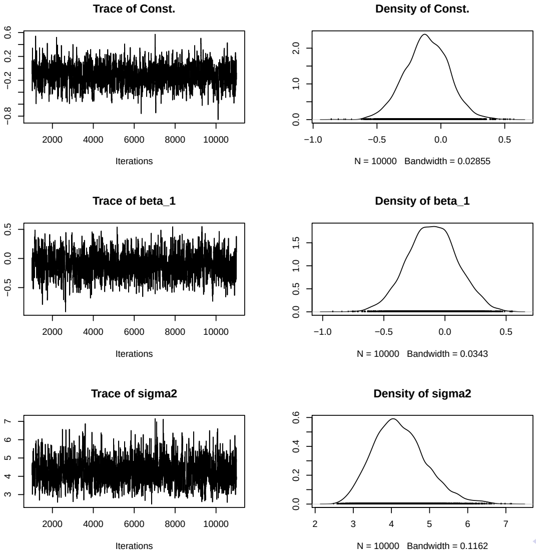
```
## Error in eval(expr, envir, enclos):  object ’chain’ not found
## Error in colnames(chain) <- names(theta):  object ’chain’ not found

##
## Call:
## lm(formula = y ~ X - 1)
##
## Residuals:
##      Min        1Q   Median        3Q       Max
## -144.609   -0.465    1.001    2.506   128.693
##
## Coefficients:
##         Estimate Std. Error t value Pr(>|t|)
## Const.   -1.303      1.292  -1.008    0.315
## beta_1    1.175      1.472   0.798    0.426
##
## Residual standard error: 18.14 on 198 degrees of freedom
## Multiple R-squared:  0.007409,Adjusted R-squared:  -0.002617
## F-statistic: 0.739 on 2 and 198 DF,  p-value: 0.4789
```

```
##
##
## @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
## The Metropolis acceptance rate was 0.44682
## @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
##
## Iterations = 1001:11000
## Thinning interval = 1
## Number of chains = 1
## Sample size per chain = 10000
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##           Mean     SD Naive SE Time-series SE
## Const. -0.1100 0.1717 0.001717       0.005848
## beta_1 -0.1011 0.2042 0.002042       0.007027
## sigma2  4.2224 0.6956 0.006956       0.023831
##
## 2. Quantiles for each variable:
##
##           2.5%     25%      50%      75%  97.5%
## Const. -0.4492 -0.2219 -0.11043 0.005888 0.2227
## beta_1 -0.4985 -0.2401 -0.09963 0.035951 0.3033
## sigma2  3.0303  3.7258  4.15981 4.653041 5.7415
```

Robust linear model

MCMC plot



The Metropolis-Hastings algorithm

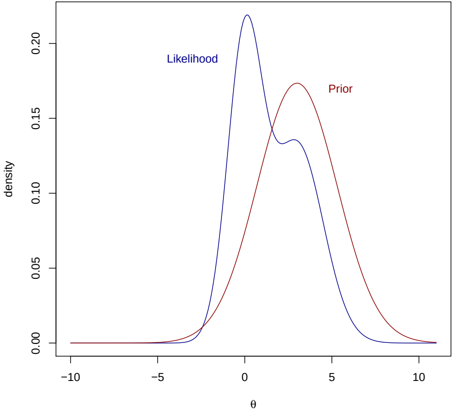
Algorithm 4 Metropolis-Hastings algorithm

```
Choose an initial value $\theta_1$.
for $s = 1 : S$ do
    Draw $\theta^*$ from $q(\cdot|\theta_s)$
    Calculate $r = \min\left\{\frac{\gamma(\theta^*)q(\theta_s|\theta^*)}{\gamma(\theta_s)q(\theta^*|\theta_s)}, 1\right\}$
    Draw $U \sim U[0,1]$
    if $U \leqslant r$ then
        $\theta_{s+1} = \theta^*$
    else
        $\theta_{s+1} = \theta_s$
    end if
end for
```

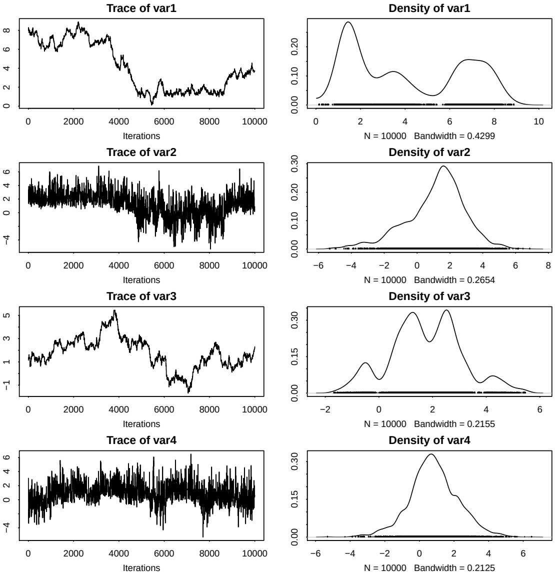
Generalizes the Metropolis algorithm for asymmetric $q(\cdot|\cdot)$.

MCMC: Convergence

A poorly mixing chain can arise because the target distribution is multimodal and our choice of starting values traps us near one of the modes (such multimodal posteriors can arise if we have a strong prior in conflict with the observed data).



MCMC: (Non) Convergence

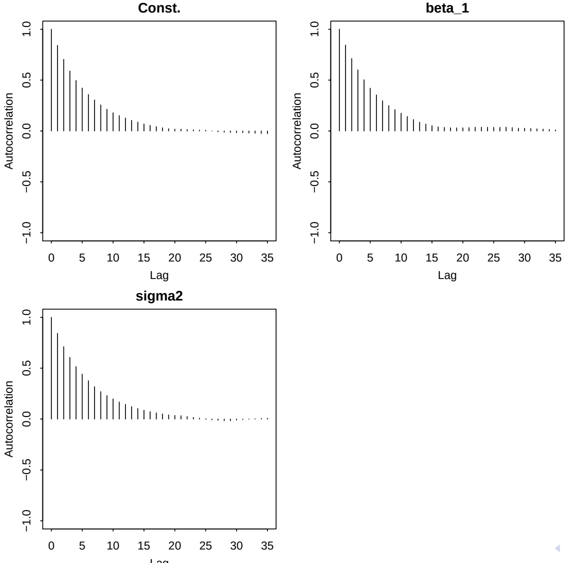


MCMC: Monitoring convergence

- Convergence is about convergence of distribution
- How to monitor convergence to an invariant distribution?
  - Run multiple chain starting from disperse initial conditions and see whether the chains are “similar” (Gelman and Rubin diagnostic)
  - Trace plots
  - Autocorrelation of draws
  - Geweke’s test for the equality of means
  - [Many other ways]

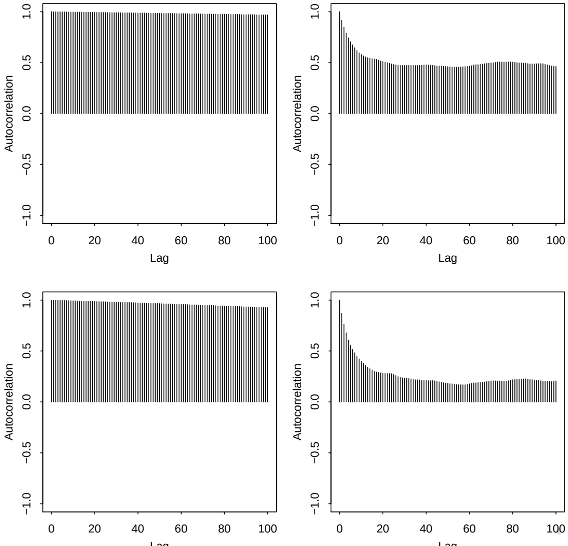
Monitoring convergence

Autocorrelation plot: Robust linear model



Monitoring convergence

Autocorrelation plot: Of the multimodal posterior



Monitoring convergence

Geweke's diagnostic

```
##
## Fraction in 1st window = 0.1
## Fraction in 2nd window = 0.5
##
## Const. beta_1 sigma2
##  1.856  1.632  1.245
```

- Geweke's diagnostic for Markov chains is based on a test for equality of the means of the first and last part of a Markov chain (by default the first 10% and the last 50%).
- If the samples are drawn from the stationary distribution of the chain, the two means are equal and Geweke's statistic has an asymptotically standard normal distribution.
- The test statistic is the difference between the two sample means divided by its estimated standard error.
- The standard error is estimated from the spectral density at zero and so takes into account any autocorrelation.

Gibbs sampler

The problem is to sample from

$$p(\theta_1, \theta_2 | y)$$

by conditional distribution

$$p(\theta_1, \theta_2 | y) = p(\theta_1 | \theta_2, y) p(\theta_2 | y)$$

While it is impossible to simulate from $p(\theta_1, \theta_2 | y)$, it is feasible to simulate from the conditionals.

- The Gibbs sampler (introduced in the context of image processing by Geman and Geman 1984), is a special case of Metropolis-Hastings sampling wherein the random value is always accepted

Gibbs sampler

Algorithm 5 Gibbs sampler

```
Choose $(\theta_2^{(1)}, \ldots, \theta_k^{(1)})$
for $s = 1 : S$ do
    Draw $\theta_1^{(s+1)} \sim p(\theta_1 | \theta_2^{(s)}, \theta_3^{(s)}, \ldots, \theta_k^{(s)}, y)$
    Draw $\theta_2^{(s+1)} \sim p(\theta_2 | \theta_1^{(s+1)}, \theta_3^{(s)}, \ldots, \theta_k^{(s)}, y)$
    $\vdots$
    Draw $\theta_k^{(s+1)} \sim p(\theta_k | \theta_1^{(s+1)}, \theta_2^{(s+1)}, \ldots, \theta_{k-1}^{(s+1)}, y)$
end for
```

Gibbs sampler

Linear model

The conjugate posteriors for the parameters of the linear model are

$$
\begin{aligned}
\beta | y, \sigma^2 &\sim N\left(\tilde{\beta}, \sigma^2 A_T^{-1}\right) \\
\sigma^2 | y &\sim \Gamma^{-1}\left(\nu_T/2, \nu_T \sigma_T^2/2\right)
\end{aligned}
$$

Let

$$
\begin{aligned}
\hat{\beta} &= (X'X)^{-1}X'Y, \\
s^2 &= (y - X'\hat{\beta})'(y - X'\hat{\beta})/(T-k) \\
A_T &= (A_0 + X'X)^{-1} \\
\tilde{\beta} &= A_T(A_0\beta_0 + X'Y) \\
\nu_T &= \nu_0 + T \\
\nu_T\sigma_T^2 &= \nu_0\sigma_0^2 + (T-k)s^2 + (\beta_0 - \tilde{\beta})A_0(\beta_0 - \tilde{\beta}) + (\hat{\beta} - \tilde{\beta})X'X(\hat{\beta} - \tilde{\beta})
\end{aligned}
$$

Gibbs sampler

Linear model

Description of the Gibbs algorithm in this particular case:

1. Draw $\sigma^2_{(1:S)} \sim \Gamma^{-1}(\nu_T/2, \nu_T \sigma^2_T/2)$
2. For $j = 1:S$: Draw $\beta_j \sim N\left(\tilde{\beta}, \sigma^2_{(j-1)} A_T^{-1}\right)$

Literature

- Robert, Christian P., and George Casella. Monte Carlo statistical methods. Vol. 319. New York: Springer, 2004.
- Geweke, John, and Michael Keane. “Computationally intensive methods for integration in econometrics.” Handbook of econometrics 5 (2001): 3463-3568.
- Geweke, John. ”Using simulation methods for Bayesian econometric models: inference, development, and communication.” Econometric Reviews 18.1 (1999): 1-73.
- Chib, Siddhartha, and Edward Greenberg. ”Markov chain Monte Carlo simulation methods in econometrics.” Econometric Theory 12 (1996): 409-431.
- Chib, Siddhartha, and Edward Greenberg. ”Understanding the metropolis-hastings algorithm.” The American Statistician 49.4 (1995): 327-335.
- Geweke, John. ”Bayesian inference in econometric models using Monte Carlo integration.” Econometrica: Journal of the Econometric Society (1989): 1317-1339.