最終更新日：2017/05/20

福中公輔

JAGS 入門

統計モデルの母数を MCMC（マルコフ連鎖モンテカルロ法）で推定したい場合のソフトウェアとして，OpenBUGS というものがありました。しかし，このソフトはいつの間にか開発が止まってしまいました。『どうしようか』と困っていたところに，JAGS というものがあることを知りました。そこで，この JAGS の使い方を勉強する過程で，簡単な解説書をまとめてみようかと思い立ち，本テキストを執筆する次第です。

基本は朝倉書店から刊行されている『マルコフ連鎖モンテカルロ法』の適用例を JAGS で書き換えていくことがメインです。メモ書き程度の文章ですが，どなたかのお役に立てば幸いです。

1　JAGS のインストール

JAGS とは Just Another Gibbs Sampler の略で，MCMC を用いてベイズモデルの解析を行うことができます。JAGS はオープンソースの統計解析ソフト R との親和性が非常に高く，『rjags』パッケージを導入することで，R 上から JAGS を操作することが可能になります。coda パッケージも特に問題なく使用可能です。

JAGS は 2017 年 5 月現在，Version 4.2.0 が以下のサイトからダウンロードできます。

`http://sourceforge.net/projects/mcmc-jags/`

2　R による JAGS

R 上から JAGS を実行したい場合，rjags パッケージをインストールする必要があります。rjags は coda パッケージと lattice パッケージを依存関係に含みますので，以下のようなコマンドでインストールするとよいでしょう。

```
> install.packages("rjags", depend=TRUE)
```

3　JAGS による回帰分析の実行

R を使って JAGS を実行する場合，以下のステップを経る必要があります。

1. データの準備
2. モデルの定義
3. 初期値の設定
4. 分析の実行
5. 収束判定

RJAGS の場合，BRUGS とは異なり，外部ファイルとして準備する必要があるのはモデルの定義のみです。データや初期値は，R 上のオブジェクトとしてそのまま書くことができます。もちろん，データが大きい場合は，外部ファイルとして準備し，read.table() で読み込むことも可能です。また，収束判定には coda パッケージを利用します。詳しくは『マルコフ連鎖モンテカルロ法（2008，朝倉書店)』を参照して下さい。

3.1　データの準備

ここでは回帰分析用に表 1 のようなデータを準備します。x が説明変数，y が基準変数になります。散布図を描くと図 1 のようになります。

表 1　サンプルデータ

x	y
0	72
10	66
20	60
30	58
40	56
50	55
60	52
70	48

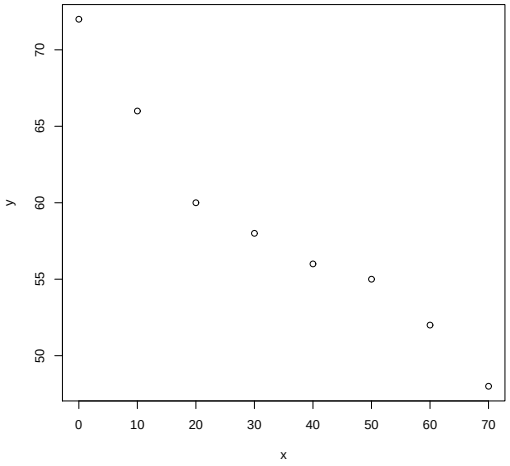

図 1　散布図

このデータを『regdata.csv』として外部ファイルに保存し，以下のコマンドでR のオブジェクトとして読み込みます。（データに欠損値がある場合は，該当部分に NA と入力しておきます。）

```
> reg_dat <- read.table("regdata.csv", sep=",", header=T)
```

次に，この『reg_dat』オブジェクトを，RJAGS で使えるようにリスト化します。リストには変数名以外に被験者数 N と回帰母数の数 K を指定しておきます。単回帰モデルの場合，回帰母数は切片と傾きの 2 つなので $K = 2$ となります。

```
> dat <- list(N=8, K=2, x=reg_dat$x, y=reg_dat$y)
```

3.2 モデルの定義

単回帰分析の場合，以下のようなモデルを定義し，外部ファイルとして保存します。ここではファイル名を『reg_model.txt』としました。

```
model{
    for(i in 1:N){
        y[i] ~ dnorm(mu[i], tau)
        mu[i] <- alpha[1] + alpha[2]*x[i]
    }

    for(i in 1:K){
        alpha[i] ~ dnorm(0, 1.0E-6)
    }

    tau ~ dgamma(1.0E-6, 1.0E-6)
    sigma2 <- 1/tau
}
```

ここで y[i] は基準変数，x[i] は説明変数を表しています。また alpha[1] は回帰式における切片，alpha[2] は傾きを，tau は誤差分散の逆数を表しています。したがって，sigma2 が誤差分散になります。

また，各回帰母数の事前分布には平均 0，分散 10^6 の正規分布を設定しています。このような分散の大きな分布にしているのは，無情報事前分布を仮定したいからです。一方，誤差分散の逆数である tau にはガンマ分布を仮定しています。

3.3 初期値の設定

推定すべき母数（alpha[1]，alpha[2]，tau）の初期値の設定をします。これもデータオブジェクトと同様にリスト化して作成します。ここでは全て 1 に設定しました。もっと母数が多くなった場合は，より適切に指定する必要があります。例えば最尤解の結果を初期値にするなどが考えられます。

```
> inits <- list(alpha=c(1,1), tau=1)
```

3.4 分析の実行

まず，モデルの定義ファイルを R に読み込ませます。そのために rjags のライブラリを実行します。

```
> library(rjags)
```

その後，関数 jags.model() を用いて，以下のように R 内でモデルの定義をします。指定する引数は『モデルを定義したファイル名』『データオブジェクト』『初期値オブジェクト』『マルコフ連鎖の連鎖数』です。なお，ここでは連鎖数を 1 にしました。

```
> m <- jags.model(
+ file = "reg_model.txt",
+ data = dat,
+ inits = inits,
+ n.chain = 1
+ )
```

次にマルコフ連鎖を更新します。

```
> update(m, 1000)
```

計算が終了したら，以下に示すように MCMC 標本の経験的事後分布の平均値を見ます。そのためには関数 jags.samples() を用いて次のように入力します。

```
> jags.samples(m, c("alpha", "sigma2"), 1000)
```

すると以下のような結果が返されました。

```
$alpha
mcarray:
[1] 69.2044968 -0.3073111

Marginalizing over: iteration(1000),chain(1)

$sigma2
mcarray:
[1] 6.324364

Marginalizing over: iteration(1000),chain(1)
```

上記を見ると，切片が 69.20，傾きが -0.31，誤差分散が 6.32 になっていることがわかります。ただしこの結果は，ほぼ全ての標本を利用しているので，アルゴリズムがバーンイン期間において抽出した標本も含んでしまっています。そこでこの点を考慮し，バーンイン期間を除いて事後分布からサンプリングを行います。

```
> codadata <- coda.samples(
+ m,
+ c("alpha", "sigma2"),
+ n.iter = 5000
+ )
```

ここでは繰り返し数が 5000 のサンプリングを行っています。summary() 関数を使って結果を表示すると以下のようになりました。

```
> summary(codadata)

Iterations = 2001:7000
Thinning interval = 1
Number of chains = 1
Sample size per chain = 5000

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

            Mean      SD  Naive SE Time-series SE
alpha[1] 69.0636 1.58459 0.0224095       0.052578
alpha[2] -0.3051 0.03763 0.0005322       0.001249
sigma2    6.1839 5.78354 0.0817916       0.115930

2. Quantiles for each variable:

            2.5%     25%     50%     75%  97.5%
alpha[1] 66.0105 68.1091 69.0122 69.9853 72.247
alpha[2] -0.3849 -0.3269 -0.3037 -0.2817 -0.231
sigma2    1.7255  3.1635  4.5942  7.1185 20.159
```

また MCMC 反復の履歴や MCMC 標本の経験密度をプロットすることもできます。

```
> plot(codadata)
```

履歴と経験密度を図 2 に示します。

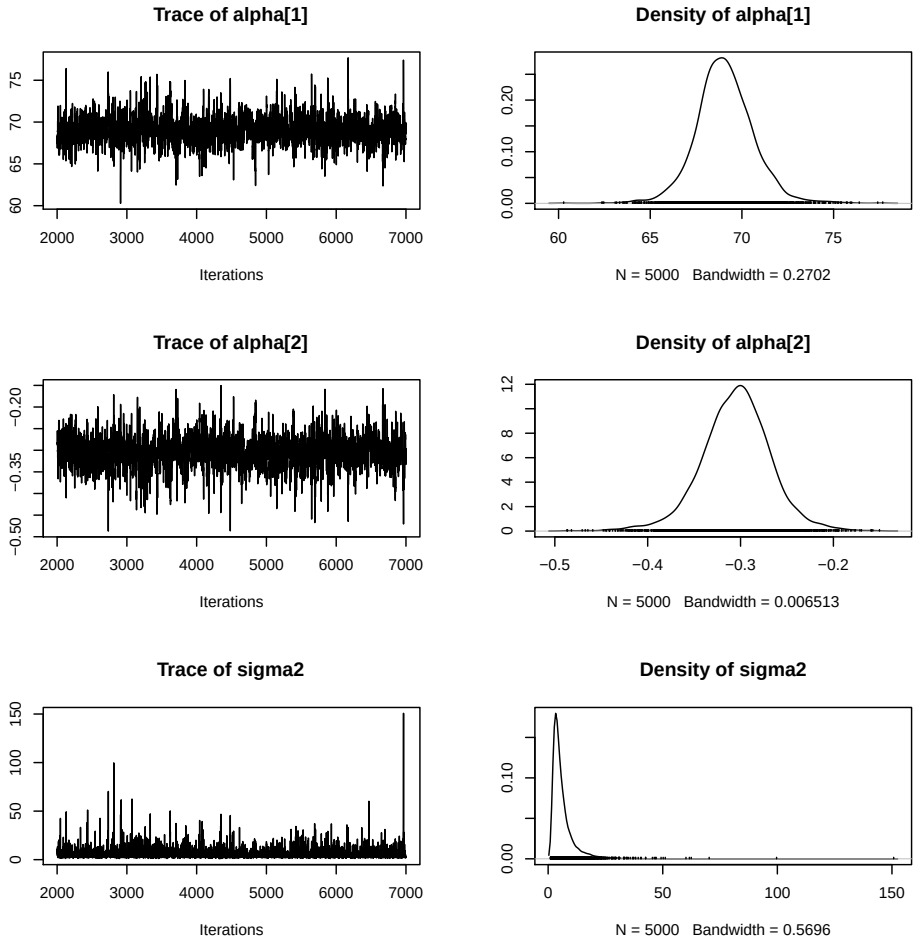

図 2　左：MCMC 反復の履歴，右：MCMC 標本の経験密度

3.5　収束判定

最後にパッケージ coda を利用して，連鎖の収束判定を行います。今回は説明のため，先に事後統計量の算出を行いましたが，本来は連鎖の収束が確認された後に事後統計量の検討を行います。この点には注意しておきましょう。

さて，上記で codadata というオブジェクトを作成しましたが，これを coda に渡します。そのために codamenu() 関数を実行します。ここでは『2　→　codadata　→　2　→　1　→　4　→　8　→　4　→　y』の順で選択していき，Geweke の指標を観察します。

```
> codamenu()
CODA startup menu

1: Read BUGS output files
2: Use an mcmc object
3: Quit

Selection: 2

Enter name of saved object (or type "exit" to quit)
1:codadata
Checking effective sample size ...OK
CODA Main Menu

1: Output Analysis
2: Diagnostics
3: List/Change Options
4: Quit

Selection: 2
CODA Diagnostics Menu

1: Geweke
2: Gelman and Rubin
3: Raftery and Lewis
4: Heidelberger and Welch
5: Autocorrelations
```

```
6: Cross-Correlations
7: List/Change Options
8: Return to Main Menu

Selection: 1

GEWEKE CONVERGENCE DIAGNOSTIC (Z-score)
=======================================

Iterations used = 2001:7000
Thinning interval = 1
Sample size per chain = 5000

$chain1

Fraction in 1st window = 0.1
Fraction in 2nd window = 0.5

alpha[1] alpha[2]   sigma2
 -1.3163   1.2849   0.3939

Geweke plots menu

1: Change window size
2: Plot Z-scores
3: Change number of bins for plot
4: Return to Diagnostics Menu

Selection: 4
CODA Diagnostics Menu
```

```
1: Geweke
2: Gelman and Rubin
3: Raftery and Lewis
4: Heidelberger and Welch
5: Autocorrelations
6: Cross-Correlations
7: List/Change Options
8: Return to Main Menu

Selection: 8
CODA Main Menu

1: Output Analysis
2: Diagnostics
3: List/Change Options
4: Quit

Selection: 4

Are you sure you want to quit (y/N)?
:y
```

該当部分だけ取り出すと，以下のようになっていることが分かります。

```
alpha[1] alpha[2]   sigma2
 -1.3163   1.2849   0.3939
```

この数値は Z スコアを表しています。したがって，絶対値が 1.96 以下に収まっている母数は収束していると判断します。本例では全てそうなっていますので，全パラメータに関して収束状況がよいことが分かります。

4 おまけ

試しに通常の回帰分析を行ってみます。

```
> lm(y ~ x, data=reg_dat)

Call:
lm(formula = y ~ x, data = reg_dat)

Coefficients:
(Intercept)            x
    69.0000      -0.3036
```

このように結果はほぼ一致していることが分かります。