とWinBUGS

# Rで統計解析入門

(20) ベイズ統計「超」入門

# WinBUGS と R2WinBUGS のセットアップ

1. 本資料で使用するデータを以下からダウンロードする
   http://www.cwk.zaq.ne.jp/fkhud708/files/R-intro/R-stat-intro_data.zip
2. WinBUGS のホームページから下記ファイルをダウンロードし WinBUGS14.exe をインストールする
   - WinBUGS14.exe
     http://www.mrc-bsu.cam.ac.uk/bugs/winbugs/WinBUGS14.exe
   - キー「WinBUGS14_immortality_key.txt」
     http://www.mrc-bsu.cam.ac.uk/bugs/winbugs/WinBUGS14_immortality_key.txt
   - パッチ（version 1.4.3）「WinBUGS14_cumulative_patch_No3_06_08_07_RELEASE.txt」
     http://www.mrc-bsu.cam.ac.uk/bugs/winbugs/WinBUGS14_cumulative_patch_No3_06_08_07_RELEASE.txt
3. パッチ（version 1.4.3）を下記フォルダに保存する
   <C:¥Program Files¥WinBUGS14>

# WinBUGS と R2WinBUGS のセットアップ

3. WinBUGS を起動する※

   <C:¥Program Files¥WinBUGS14¥WinBUGS14.exe>

4. [File] → [Open] からパッチ（version 1.4.3）を開き，

   [Tools] → [Decode] を選択し，[Decode ALL] を選択する

5. キーについても 4. と同様の手順を行う

6. 下記フォルダに「Key.ocf」が入っているか確認し，インストール完了

   <C:¥Program Files¥WinBUGS14¥Bugs¥Code>

7. R を起動し以下を実行する※ → その後，作業ディレクトリに移動する

```
> install.packages("R2WinBUGS", dep = T)
> library(R2WinBUGS)
> setwd("C:/temp")
```



※ Vista/7 の場合は，右クリックから「管理者権限として実行」

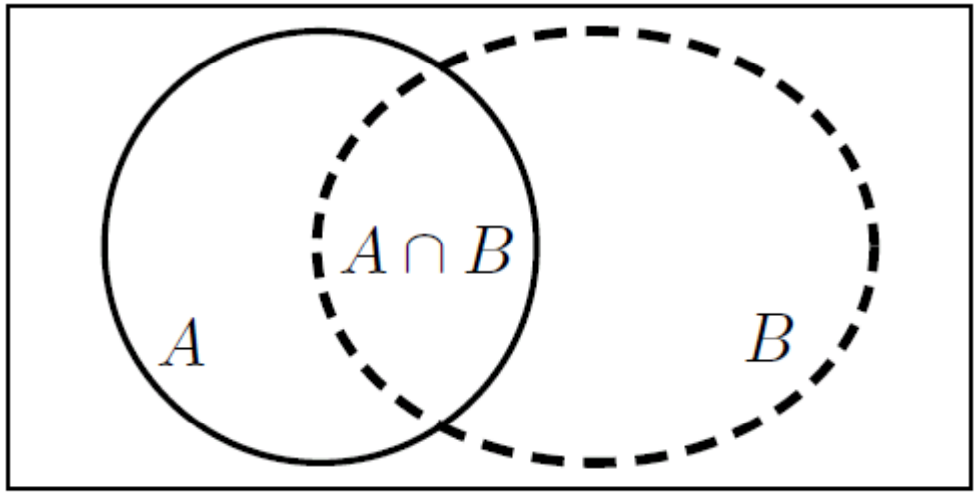
# 本日のメニュー

1. 条件付き確率とベイズの定理
2. ベイズの定理の適用例
3. マルコフ連鎖モンテカルロ法
4. ベイズ統計の適用例
   - 正規分布（分散既知）の問題
   - ロジスティック回帰分析
   - 単回帰分析

【参考】WinBUGS 上でベイズ推定を行う手順

# 条件付き確率

- 2 つの事象 A と B について，p(A) と p(B) をそれぞれ
  「 A が起きる確率」「B が起きる確率」とする

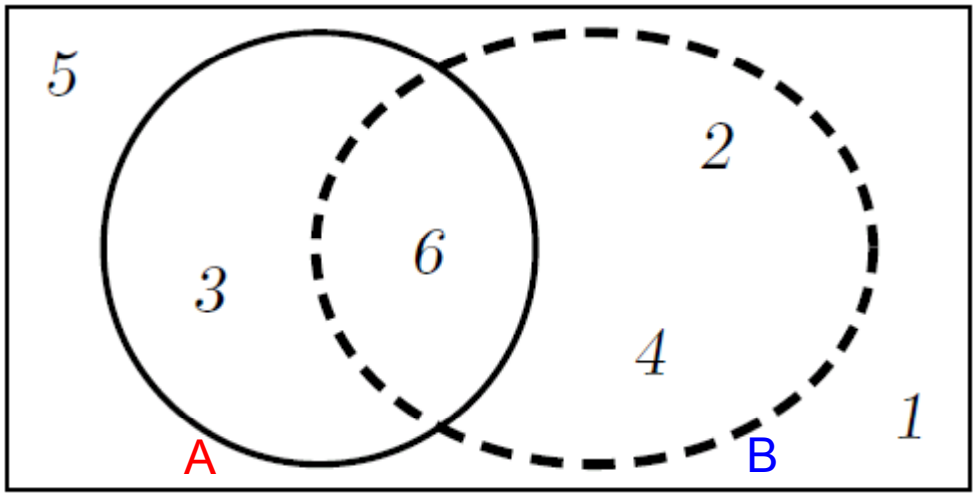



- このとき「 A が与えられたときの B の条件付き確率」は以下となる

$$p(B \mid A) = \frac{P(B \cap A)}{P(A)}$$

# 条件付き確率の例

- A と B をそれぞれ「3 の倍数」「2 の倍数」とする
- p(A) は「1,2,3,4,5,6」のうち「3,6」が起きる確率なので，1/3




- 「 A が与えられたときの B が起こる条件付き確率」である p(B|A) は「3,6」のうち「6」が起きる確率なので，1/2 となる

$$p(B \mid A) = \frac{P(B \cap A)}{P(A)} = \frac{1/6}{2/6} = \frac{1}{2}$$

# ベイズの定理

- 先ほどの「 A が与えられたときの B の条件付き確率」の式より

$$P(B \cap A) = p(B \mid A) \times P(A)$$

- 上式を，「 B が与えられたときの A の条件付き確率」

$$p(A \mid B) = \frac{P(A \cap B)}{P(B)}$$

の p(A∩B) に代入することで以下を得る（ p(A∩B) = p(B∩A) に注意）

$$p(B \mid A) = \frac{P(A \mid B)}{P(A)} \times P(B)$$

- 上式の A を「興味のあるパラメータ θ 」，Bを「データ y 」に置き換え以下を得る → これがベイズの定理

$$p(\theta \mid y) = \frac{P(y \mid \theta)}{P(y)} \times P(\theta)$$

# ベイズの定理

$$p(\theta \mid y) = \frac{P(y \mid \theta)}{P(y)} \times P(\theta)$$

- p(θ)　：パラメータ θ の事前分布
- p(y|θ)：尤度
- p(θ|y)：パラメータ θ の事後分布
- p(y)　：p(θ|y) の全確率が 1 になるための基準化定数
- ちなみに，「ベイズの定理」の表現として，p(y) を省略した形で

  事後分布 ∝ 尤度×事前分布

  と表記することが多い（∝：比例するという意味）

# 本日のメニュー

1. 条件付き確率とベイズの定理
2. ベイズの定理の適用例
3. マルコフ連鎖モンテカルロ法
4. ベイズ統計の適用例
   - 正規分布（分散既知）の問題
   - ロジスティック回帰分析
   - 単回帰分析

【参考】WinBUGS 上でベイズ推定を行う手順

# ベイズの定理の適用例

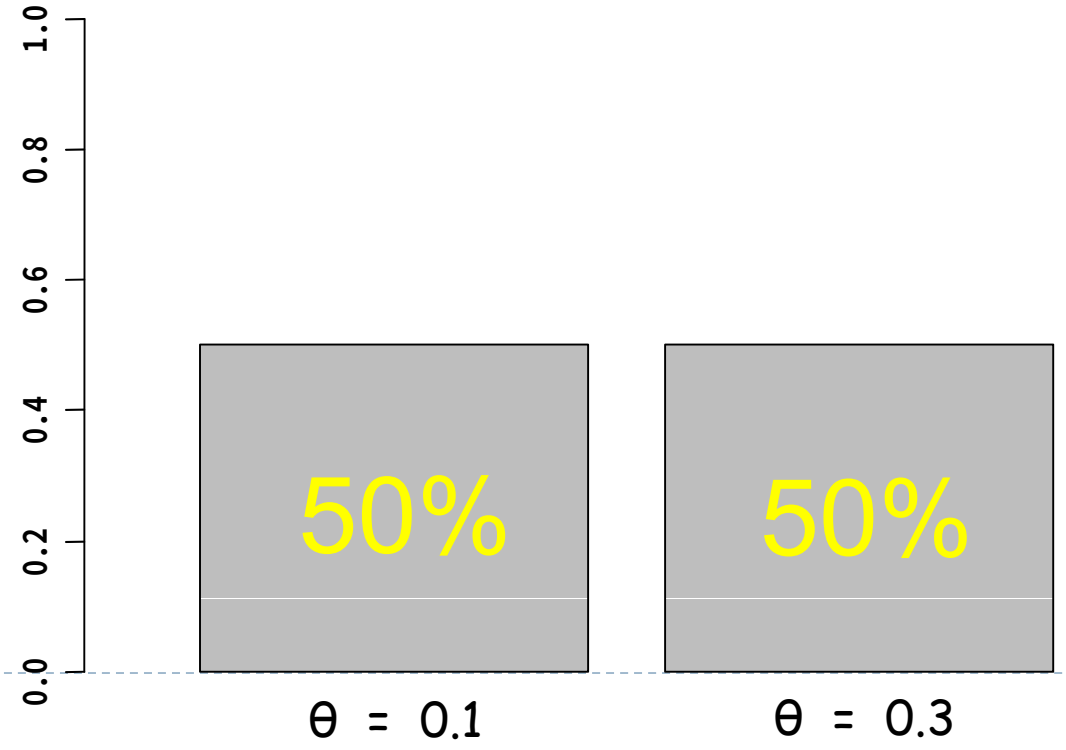
- うつ病を患っている患者さんに対して薬剤による治療を行う
- 事前情報では，この薬剤の改善割合 θ は 0.1（10%）か 0.3（30%）のどちらかである
- θ は 0.1 か 0.3 かは分からない（どちらも等確率で起こり得る感じ）
- 実際に 5 人の患者さんに薬剤を投与したところ 2 人の患者さんが「改善あり」となった
- このとき，改善割合 θ が 0.1 と 0.3 のどちらであるかをベイズの定理により推測してみる

# ベイズの定理の適用例

- 5 人の患者さんに薬剤を投与したところ 2 人の患者さんが「改善あり」
- 改善割合 θ が 0.1 と 0.3 のどちらであるかをベイズの定理により推測
- 場面設定は以下の通り
  - θ：改善割合（ 0.1 か 0.3 のいずれか）
  - p(θ)：改善割合 θ の事前分布（ 0.1 となる確率も 0.3 となる確率も 0.5 ）
  - y：データ（ n = 5人中，「改善あり」となった患者さんの人数）
  - p(y|θ)：改善割合 θ に関する尤度は二項分布 ${}_5C_2 \times \theta^2 \times (1-\theta)^3$ に従う

# ベイズの定理の適用例

- 事前分布は下図のような分布
- ベイズの定理を用いてパラメータ θ の事後分布 p(θ|y) を求め，このグラフ（分布）を更新してみる



# ベイズの定理の適用例

- θ = 0.1 のときの事前分布と尤度は以下となる
  - p(θ)　= 0.5
  - $p(y|\theta) = {}_5C_2 \times 0.1^2 \times (1-0.1)^3 = 0.0729$
- θ = 0.3 のときの事前分布と尤度は以下となる
  - p(θ)　= 0.5
  - $p(y|\theta) = {}_5C_2 \times 0.3^2 \times (1-0.3)^3 = 0.3087$

| θ | 事前分布<br>p(θ) | 尤度<br>p(y\|θ) | 尤度×事前分布<br>p(θ)×p(y\|θ) |
|---|---|---|---|
| 0.1 | 0.5 | 0.0729 | 0.0365 |
| 0.3 | 0.5 | 0.3087 | 0.1544 |
| 計 | 1.000 | | 0.1908 |

# ベイズの定理の適用例

- θ = 0.1 のときの「尤度×事前分布」は 0.0365
- θ = 0.3のときの「尤度×事前分布」は 0.1544

  ⇒ この 2 つの和は 0.1908 となり 1 にならないので，このままでは確率分布にはなりえない
- そこで，2 つの「尤度×事前分布」の和が 1 になるように，それぞれの「尤度×事前分布」の値を 0.1908 で割ってみる※

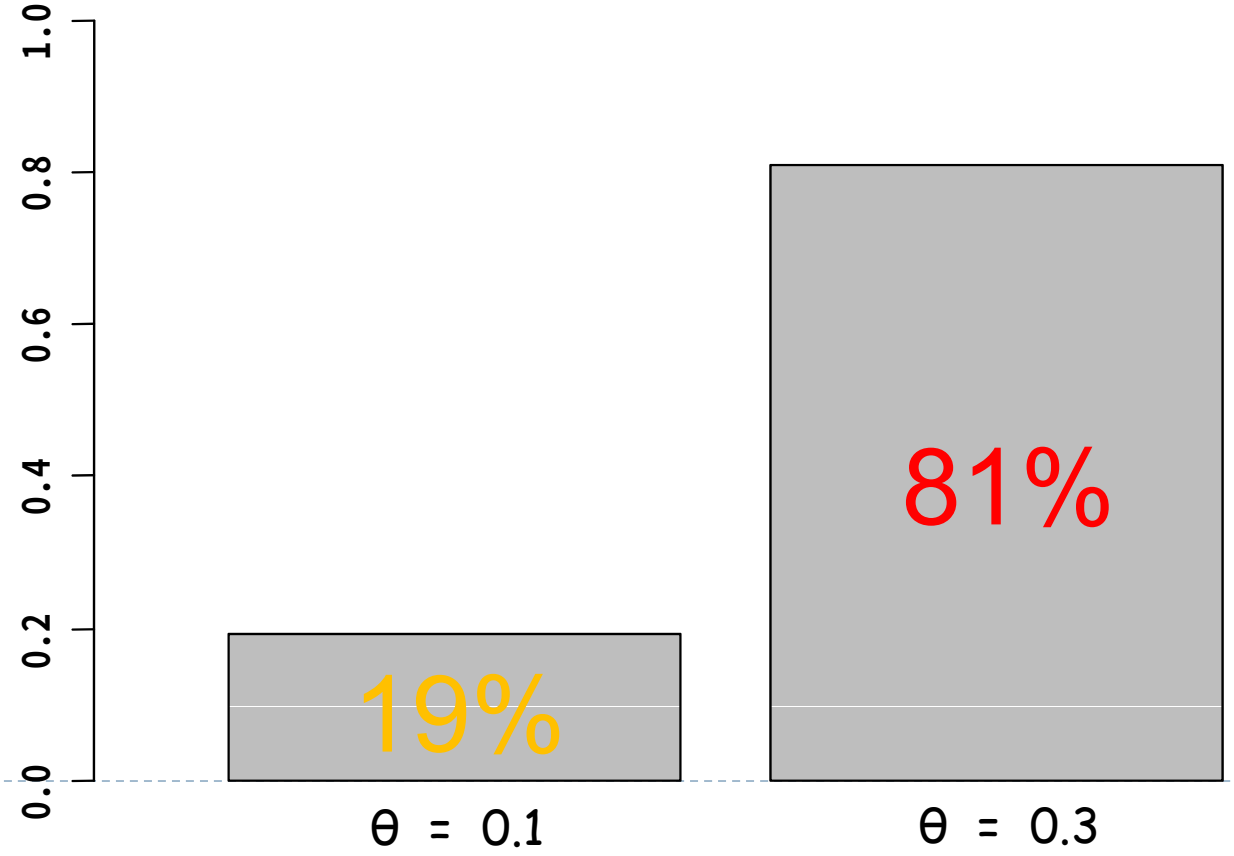
| θ | 事前分布<br>p(θ) | 尤度<br>p(y\|θ) | 尤度×事前分布<br>p(θ)×p(y\|θ) | 事後分布<br>**p(θ\|y)** |
|---|---|---|---|---|
| 0.1 | 0.5 | 0.0729 | 0.0365 | **0.1910** |
| 0.3 | 0.5 | 0.3087 | 0.1544 | **0.8090** |
| 計 | 1.000 | | 0.1908 | **1.0000** |

÷ 0.1908

※ 実はこの 0.1908 が p(y)

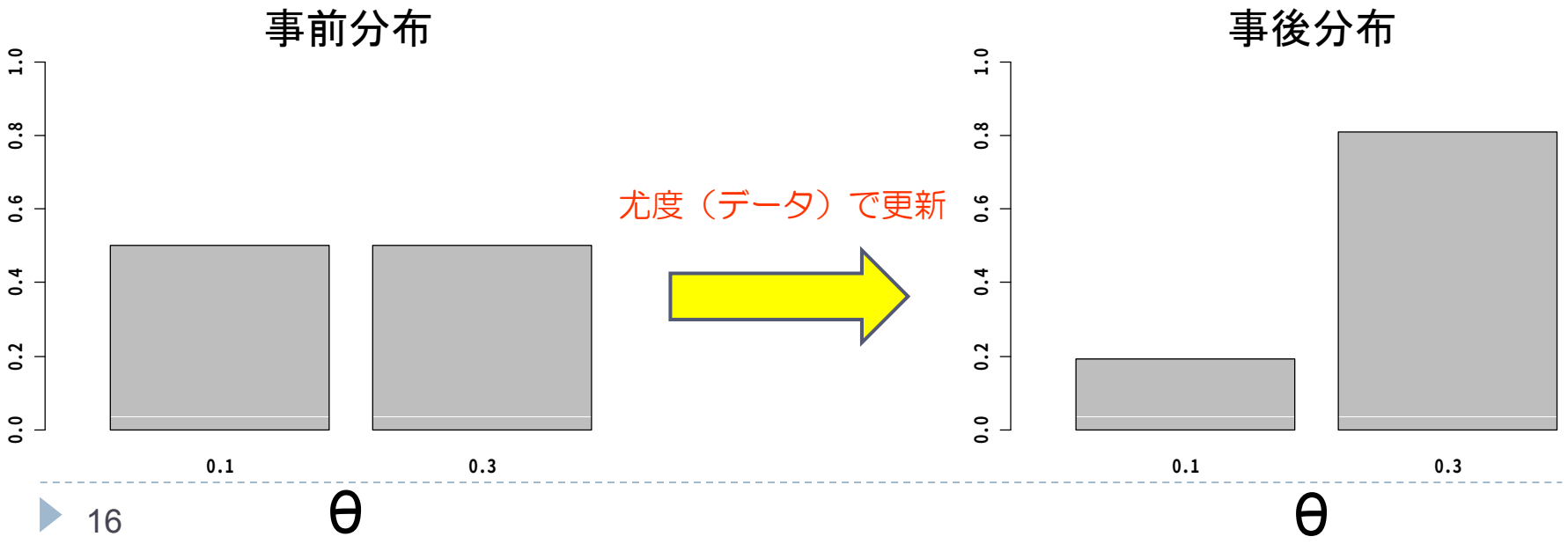
# ベイズの定理の適用例

- 事後分布が求まった → グラフにすると以下の通り
  - θ = 0.1 である確率は 19%
  - θ = 0.3 である確率は 81%



# ベイズの定理の適用例

- このように「改善割合 θ は 0.1（10%）か 0.3（30%）のどちらか（等確率）である」という事前分布を「 5 人中 2 人の患者さんが改善あり」という尤度（データ）で更新し，「改善割合 θ が 0.3（30%）になる確率が高いので，改善割合 θ は 0.3（30%）っぽい」という事後分布を求めることがベイズ解析の目的



# 前頁のグラフを作成するプログラム

```
> f <- function(x) 10*x^2*(1-x)^3
> f(0.1)
[1] 0.0729
> f(0.3)
[1] 0.3087
> f(0.1)/2
[1] 0.03645
> f(0.3)/2
[1] 0.15435
> f(0.1)/2+f(0.3)/2
[1] 0.1908
> f(0.3)/2/0.1908
[1] 0.8089623
> f(0.1)/2/0.1908
[1] 0.1910377
> barplot(c(0.5,0.5), names=c(0.1,0.3), xlim=c(0,2.5), ylim=c(0,1))
> barplot(c(1-0.8090,0.8090), names= (0.1,0.3), xlim=c(0,2.5), ylim=c(0,1))
```

# 本日のメニュー

1. 条件付き確率とベイズの定理
2. ベイズの定理の適用例
3. マルコフ連鎖モンテカルロ法
4. ベイズ統計の適用例
   - 正規分布（分散既知）の問題
   - ロジスティック回帰分析
   - 単回帰分析

【参考】WinBUGS 上でベイズ推定を行う手順

# 事後分布を求める方法

- 事前分布を設定した後，尤度（データ）で更新することで事後分布を求める方法は 2 つある

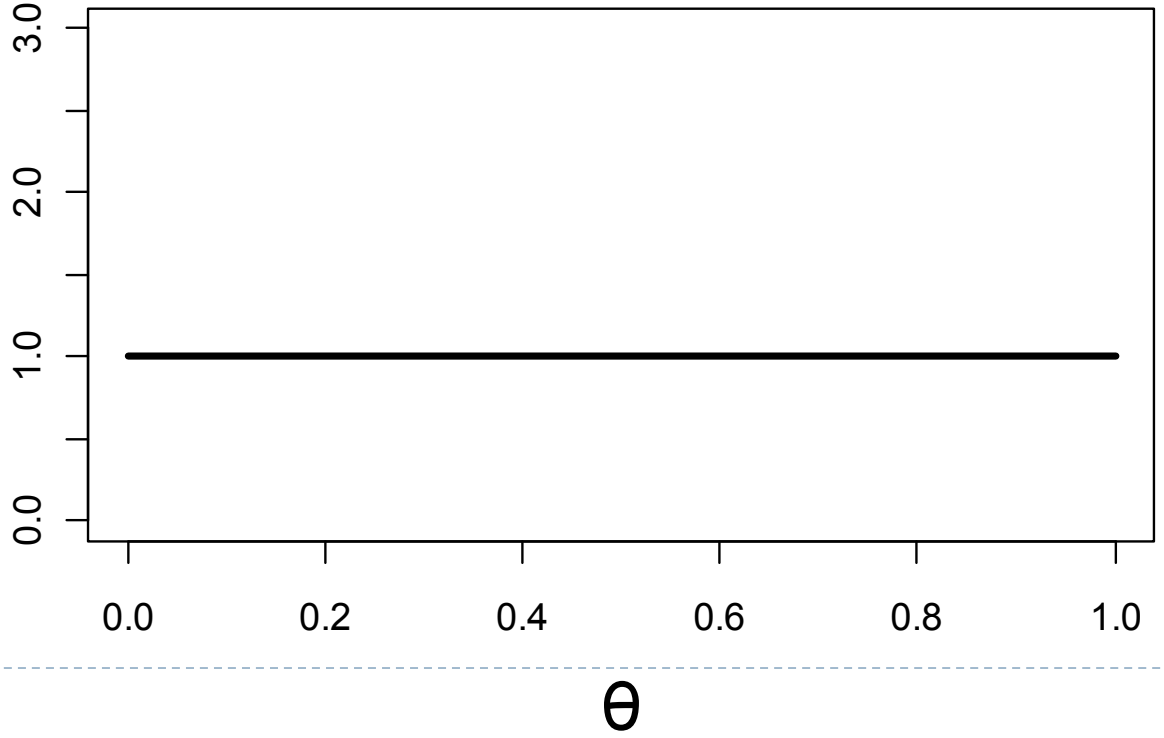
- 解析的に事後分布を求める方法
- マルコフ連鎖モンテカルロ法（MCMC；Markov Chain Monte Carlo）により事後分布の乱数を生成する方法

# 解析的に事後分布を求める方法

- うつ病を患っている患者さんに対して薬剤による治療を行う
- 事前情報では，この薬剤の改善割合 θ が 0（0%）～ 1（100%）のどの辺りかは予想できなかった
- 実際に 5 人の患者さんに薬剤を投与したところ 2 人の患者さんが「改善あり」となった
- 改善割合 θ がどのような分布であるかをベイズの定理により推測する
- 場面設定は以下の通り
  - θ：改善割合（ 0 ～ 1 ），パラメータ
  - p(θ)：θ の事前分布をベータ分布 beta(1,1)：p(θ) = 1 （0≦θ≦1）とする
    （このような事前分布を無情報事前分布とよぶ）
  - y：データ（ n = 5人中，「改善あり」となった患者さんの人数）
  - p(y|θ)：θ に関する尤度は二項分布 ${}_5C_2 \times \theta^2 \times (1-\theta)^3$ にに従うが，
    θ に無関係の部分を省き，$p(y|\theta) \propto \theta^2 \times (1-\theta)^3$ とおく

# 解析的に事後分布を求める方法

- θ の事前分布（ベータ分布 beta(1,1) ）は下図のような一様な分布（無情報事前分布とよぶ）
- ベイズの定理を用いてパラメータ θ の事後分布 p(θ|y) を求め，このグラフ（分布）を更新してみる



# 解析的に事後分布を求める方法

- ベイズの定理の式より

  $p(\theta|y) \propto p(y|\theta) \times p(\theta) = \theta^2 \times (1-\theta)^3$

- となるので，事後分布 $p(\theta|y)$ は $\theta^2 \times (1-\theta)^3$ に比例した式になるが，このままでは全確率が 1 にならないので確率分布にならない

- ところで，ベータ関数 B(a,b)：

$$B(a,b) = \int_0^1 \theta^{a-1}(1-\theta)^{b-1} d\theta$$

- なるものを持ち出すと，先ほどの $\theta^2 \times (1-\theta)^3$ をベータ関数 B(2+1,3+1) で割り算したものは以下のベータ分布 beta(3,4) となる

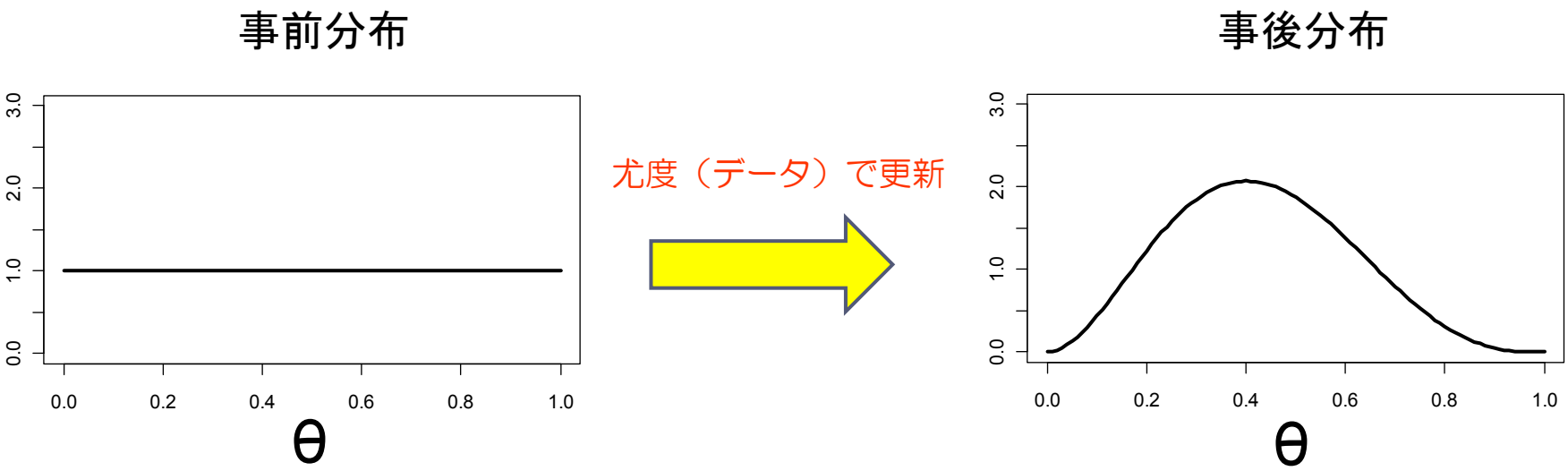
$$beta(3,4) = \frac{\theta^2(1-\theta)^3}{B(2+1,3+1)}$$

# 解析的に事後分布を求める方法

- ベータ分布 beta(3,4) の全確率は

$$\int_0^1 \frac{\theta^2(1-\theta)^3}{B(2+1,3+1)}d\theta = 1$$

となるので，最終的に事後分布 p(θ|y) はベータ分布 beta(3,4)となる



# 【参考】ベータ分布＆事後分布の平均と分散

- ベータ分布 beta(a,b) の平均と分散は以下である

$$\text{事後平均} = \frac{a}{a+b} \quad , \quad \text{事後分散} = \frac{ab}{(a+b)^2(a+b+1)}$$

- よって，事後分布 p(θ|y) の事後平均と事後分散はそれぞれ以下となる

$$\text{事後平均} = \frac{3}{3+4} = 0.4286, \quad \text{事後分散} = \frac{3\times 4}{(3+4)^2(3+4+1)} = 0.0306$$

# 解析的に事後分布を求める方法

- 解析的に解く場合，運よく式展開が出来ればよいが，このような非常に単純な状況設定においても，ベータ関数を持ち出す必要がある位で，事後分布を解析的に求めることは結構手間がかかる
  ⇒ 参考：共役分布
- 複雑な状況（複雑な事前分布や複数のパラメータを設定する場合）になると，事後分布を解析的に求めることはもっと難しくなり，実質計算不能になることがほとんど
- 事後分布を解析的に求めることは難しいことが多いので，事後分布を解析的に求めることをあきらめ，事後分布に従う乱数を生成することで事後分布を求めたことにしようという方法がある
  ⇒ これがマルコフ連鎖モンテカルロ法（**MCMC**）

# マルコフ連鎖モンテカルロ法

- マルコフ連鎖モンテカルロ法（MCMC）という方法により事後分布に従う乱数を生成し，事後分布に関する特徴をつかむことを考える
  ⇒ WinBUGS と R2WinBUGS の登場！
- 手順は以下の通り

1. 「モデル式」を記述した bugs ファイルを作成し，作業ディレクトリに保存
2. R 上で以下を実行する（ R2WinBUGS を呼び出し，作業ディレクトリへ）

```
> library(R2WinBUGS)
> setwd("C:/temp")
```

3. データ入力，パラメータの初期値設定をした後，関数 bugs() を実行し，パラメータの事後分布に従う乱数を生成する
4. 事後分布の情報（要約統計量，分布のグラフ，収束判定）を得る

# 「モデル式」を記述した bugs ファイルの作成

```
# model
model {
  theta ~ dbeta(1, 1)
  y     ~ dbin(theta, n)
}
```

1 行目：「#」はコメント文であることを表す

2 行目：モデル式の先頭は「model { 」とする

3 行目： theta（θ）がベータ分布 beta(1,1) に従っていることを表す

4 行目：データ y が二項分布 Binomial(theta, n) に従っていることを表す

5 行目：モデル式の末尾は「 } 」とする

⇒ というテキストファイル「 winbugs-0.txt 」を C:/temp に保存する

# モデル式の書式

- パラメータやデータが特定の確率分布に従うことを以下のように表す
  - パラメータやデータ ~ dxxxx
  - 「~」は「特定の確率分布に従う」ことを表す
  - 「dxxxx」の「d」は「確率分布（distribution）」であることを表す
    例：データ y が二項分布に従う場合は y ~ dbin(theta,n)
  - 「xxxx」に確率分布の名前を指定する
  - WinBUGS で用意されている確率分布の一覧は次頁
- この例のデータは「n = 5」「y = 2」と，1 つの変数に対してデータが 1 つしかないのでモデル式は単純であるが，1 つの変数に対してデータが複数ある場合は，for 文を用いてもう少し複雑な記述が必要（後述）

# 【参考】WinBUGS で使える関数一覧

確率分布に関する関数一覧

| 確率分布名 | WinBUGS の関数 |
|---|---|
| ベルヌーイ分布 | dbern(p) |
| 二項分布 | dbin(p, n) |
| 多項分布（テーブル分布） | dcat(p[]) |
| 負の二項分布 | dnegbin(p, r) |
| ポアソン分布 | dpois(lambda) |
| ベータ分布 | dbeta(a, b) |
| $\chi^2$分布 | dchisqr(k) |
| 二重指数分布 | ddexp(mu, tau) |
| 指数分布 | dexp(lambda) |
| ガンマ分布 | dgamma(a, b) |
| 対数正規分布 | dlnorm(mu, tau) |
| ロジスティック分布 | dlogis(a, b) |
| 正規分布 | dnorm(mu, 1/sigma$^2$) |
| t分布 | dt(mu, tau, k) |
| 一様分布 | dunif(a, b) |

数学関数一覧

| 機能 | WinBUGSの 関数 |
|---|---|
| 絶対値 | abs(x) |
| cos(x) | cos(x) |
| exp(x) | exp(x) |
| log(x) | log(x) |
| ロジット関数：ln(x/(1-x)) | logit(x) |
| 最大値 | max(x, y) |
| 平均値 | mean(...) |
| 最小値 | min(x, y) |
| 標準正規分布の累積分布関数 | phi(x) |
| 累乗：$x^y$ | pow(x, y) |
| sin(x) | sin(x) |
| 丸め関数 | round(x) |
| 標準偏差 | sd(x) |
| 定義関数：x 以上ならば 1,<br>それ以外ならば 0 | step(x) |
| 総和 | sum(x) |

※ 正規分布 dnorm() の第 2 引数は分散ではなく分散の逆数である点に注意

# 各種設定

2. R2WinBUGS を呼び出した後，作業ディレクトリへ移動する

```
> library(R2WinBUGS)
> setwd("C:/temp")
```

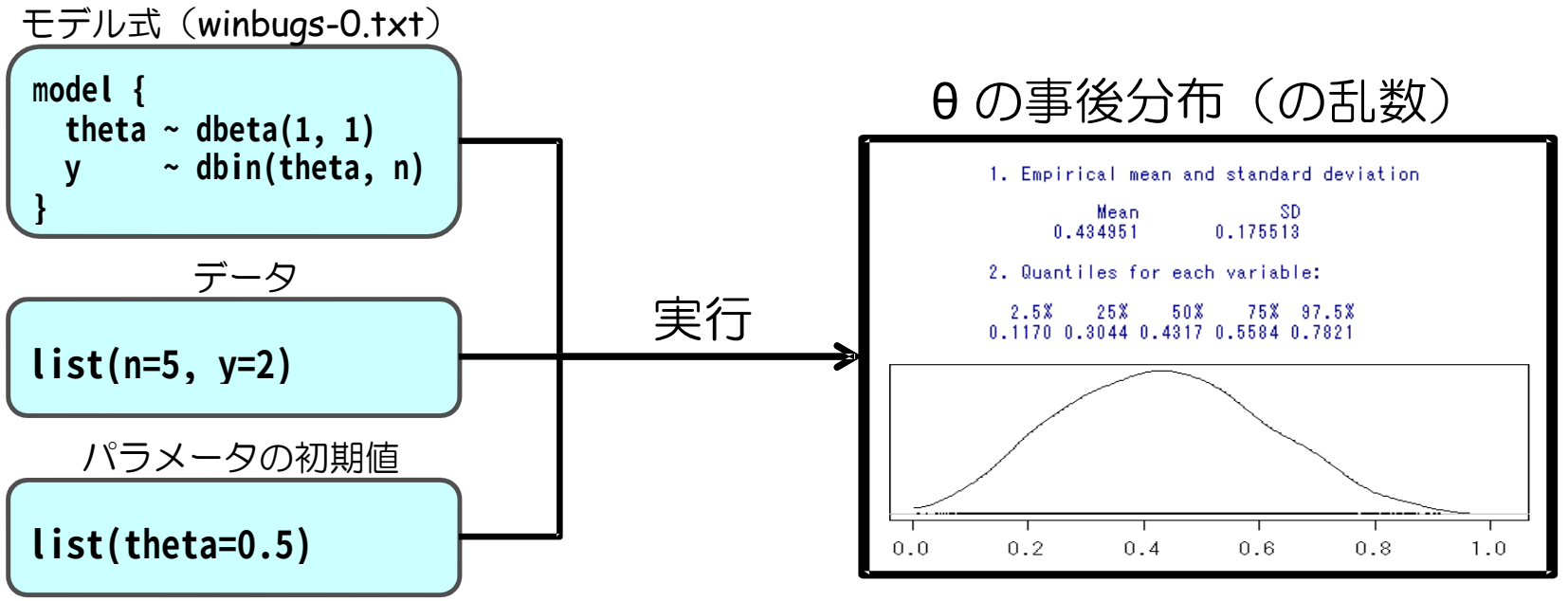
3. データ入力，パラメータの初期値設定を行う

```
> # データを入力＋データの名前を変数 data に格納
> n    <- 5
> y    <- 2
> data <- list("n", "y")

> # 乱数を生成するパラメータの初期値を入力
> init <- list( list(theta=0.5) )  # listの中にlistで指定する

> # 乱数を生成するパラメータの名前を指定
> parameters <- c("theta")
```

# 関数 bugs() を実行 → 事後分布に従う乱数の生成

3. 関数 bugs() を実行する

```
> tmp    <- bugs(data, init, parameters, model.file="winbugs-0.txt",
+                n.chains=1, n.thin=3, n.burnin=1000, n.iter=10000,
+                DIC=FALSE, debug=FALSE, codaPkg=TRUE,
+                bugs.dir="C:/Program Files/WinBUGS14/")
> result <- read.bugs(tmp)
```



# 関数 bugs() を実行 → 事後分布に従う乱数の生成

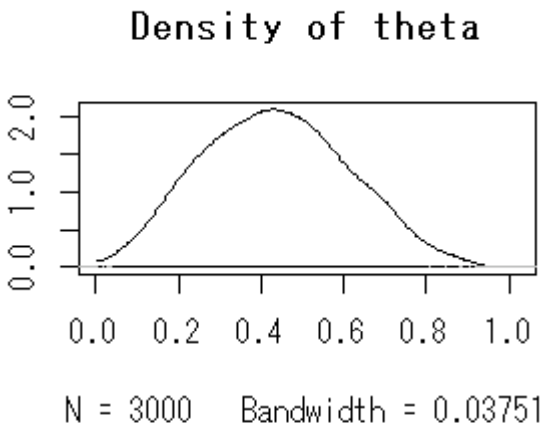
3. 関数 bugs() を実行する → 変数 result に θ の事後分布に従う乱数が

```
> tmp    <- bugs(data, init, parameters, model.file="winbugs-0.txt",
+                n.chains=1, n.thin=3, n.burnin=1000, n.iter=10000,
+                DIC=FALSE, debug=FALSE, codaPkg=TRUE,
+                bugs.dir="C:/Program Files/WinBUGS14/")
> result <- read.bugs(tmp)
```

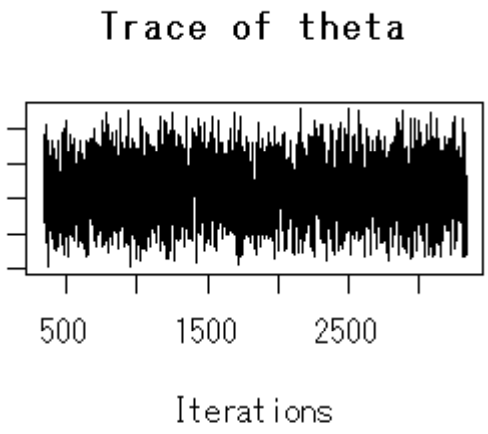
- data, init, parameters：データやパラメータの初期値等を指定
- model.file：bugs ファイル（winbugs-0.txt）の名前を指定
- 乱数の数：( n.iter − n.burnin ) ÷ n.thin = 3000 個
  ⇒ 連鎖の数(n.chains) = 1 ， 生成した乱数の最初の n.burnin = 1000 個を捨て，
  乱数の相関を減らすために n.thin = 3 個おきに事後分布の乱数を採用する
  （乱数の品質が上がる）
- debug=FALSE：エラーが出た時はここを TRUE にしてデバッグを行う
- 変数 result に θ の事後分布に従う乱数が格納される

# 結果の要約

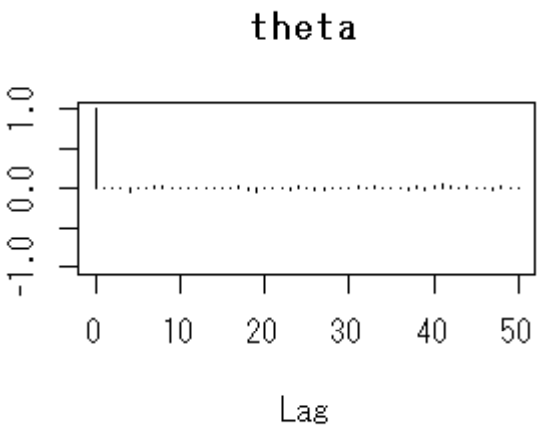
```
> summary(result)                          # θの事後分布の乱数の要約統計量を表示

1. Empirical mean and standard deviation for each variable, plus ...:
          Mean             SD       Naive SE Time-series SE
      0.434951       0.175513       0.003204       0.002777

2. Quantiles for each variable:
  2.5%    25%    50%    75%  97.5%
0.1170 0.3044 0.4317 0.5584 0.7821

> densplot(result)                         # 事後分布のグラフ
> traceplot(result)                        # トレースプロット
> autocorr.plot(result, lag.max=50)        # 自己相関のグラフ
```




【事後分布のグラフ】



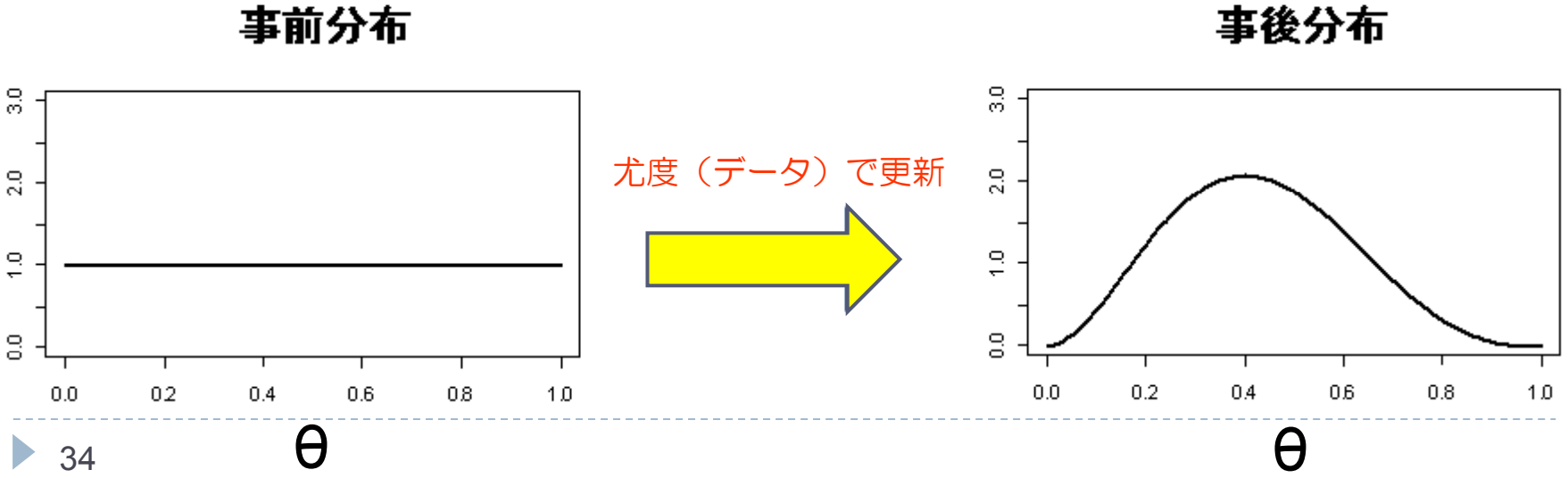

【トレースプロット】




【自己相関のグラフ】

# θ の事後分布の乱数の要約統計量

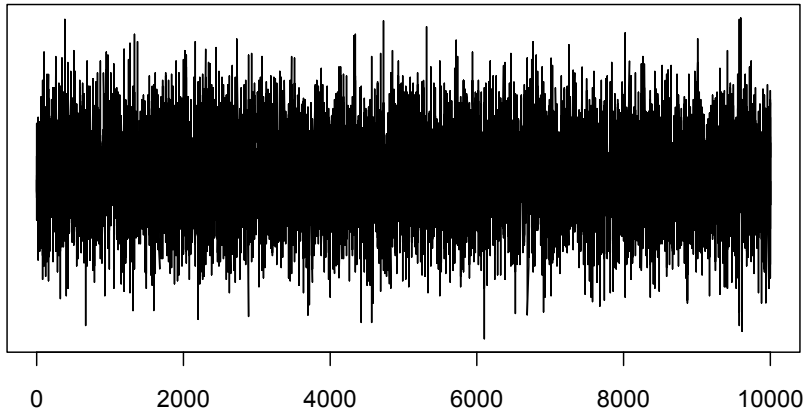
- 「θ の事後分布の乱数の要約統計量」や「乱数の密度推定」を
  「θ の事後分布の要約統計量」や「事後分布の密度」の代用とする
- 例えば，「θ の事後分布の乱数の平均が 0.43，標準偏差が 0.17」となったが，これより「θ の事後分布の平均が 0.43，標準偏差が 0.17」と解釈する
  ⇒ これがマルコフ連鎖モンテカルロ法



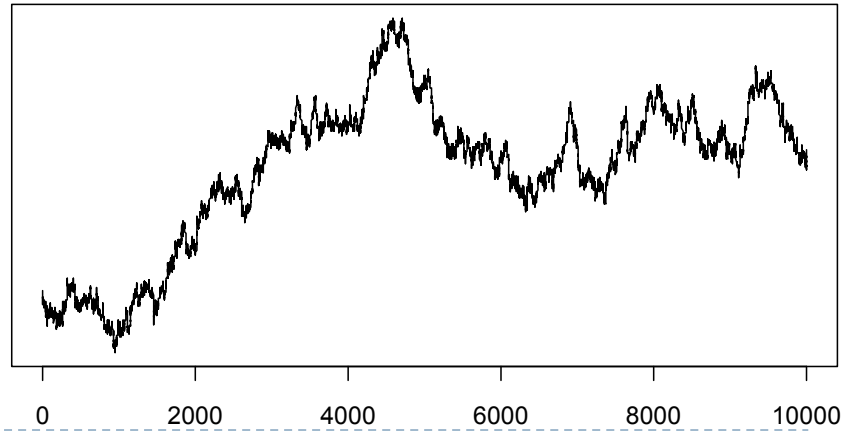
# トレースプロット

- 乱数を順番にプロットしたもの（横軸：乱数の順番，縦軸：乱数の値）
- マルコフ連鎖モンテカルロ法で生成した乱数は，生成した最初の方の乱数は品質が悪く（何らかの傾向がみられる），後の方の乱数は品質が良い（傾向がみられない）という特徴がある
  ⇒ 先のプログラムでは，最初の方の乱数（burn-in）は捨てている
- トレースプロットから，今生成した乱数の品質が良いかどうかを確認することが出来る（何らかの傾向がみられる場合は品質が悪い）

【品質が良い場合】



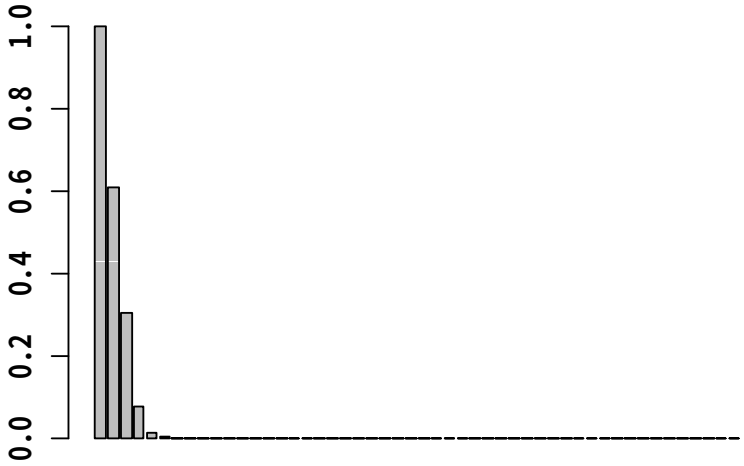

【品質が悪い場合】



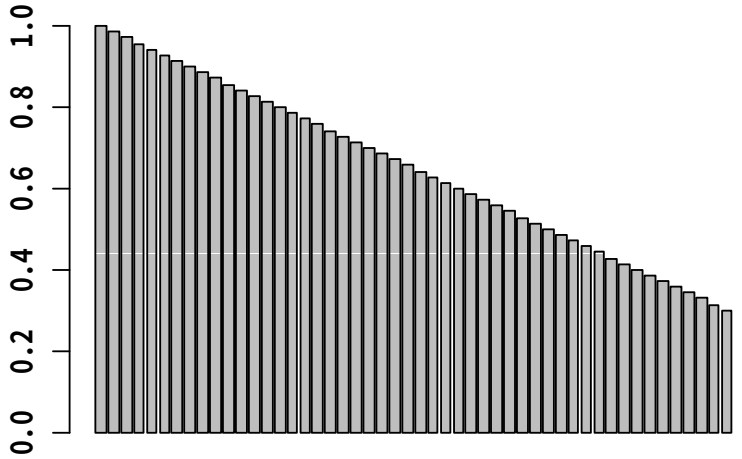
# 自己相関のグラフ

- 乱数の自己相関の結果（横軸：ラグ（何個前の乱数同士の相関を取るか），縦軸：相関の度合い）
- マルコフ連鎖モンテカルロ法で生成した乱数は，それぞれが独立標本（であるように見立てたもの）なので，ラグを大きくしても相関が高い場合は品質が悪く，ラグを大きくすると相関がすぐに低くなる場合は品質が良い

【品質が良い場合】




【品質が悪い場合】



# 【参考】マルコフ連鎖の収束に関する検定

- 帰無仮説：マルコフ連鎖が収束している（品質が良い）
  に関する検定手法もある
  - Geweke's convergence diagnostic：|z| < 1.96 以下ならば品質が良いと判断
  - Gelman and Rubin's convergence diagnostic：chain が 2 個以上必要

```
> geweke.diag(result)

[[1]]

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

  theta
-0.4598
```

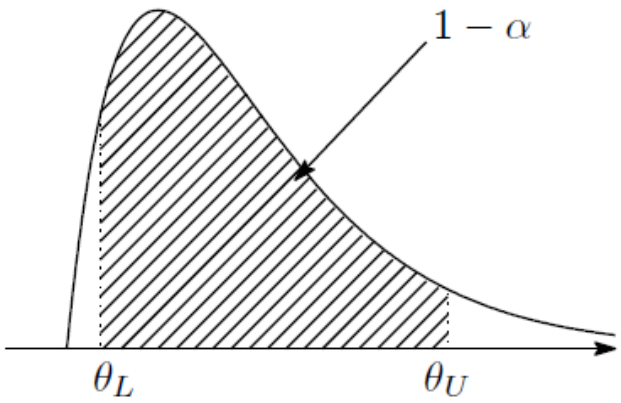
# 95% 確信区間

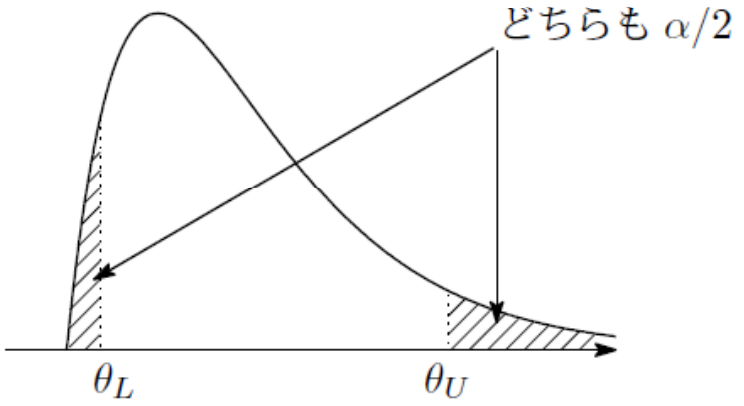
```
> summary(result)

2. Quantiles for each variable:
  2.5%    25%    50%    75%  97.5%
0.1170 0.3044 0.4317 0.5584 0.7821
```

- 上記の赤線部は，事後分布（に従う乱数）の両側 95% 確信区間（credible interval）で，パラメータ θ が 95% の確率で含まれる区間を表す
  ⇒ [0.1170, 0.7821] が両側 95% 確信区間（ Equal-Tail Interval ）
- 頻度論における信頼区間（confidence interval）は，ベイズ解析では確信区間と呼び，解釈も頻度論の区間とは異なるので注意
- 頻度論の信頼区間を「θ が 95% の確率で含まれる」とするのはダメで，「データの収集と解析を 100 回繰り返して 100 個の信頼区間を得たときに，95 個の信頼区間がパラメータ θ を含んでいる」という回りくどい解釈となってしまうが，ベイズの確信区間はパラメータ θ の分布から得られるものなので，「パラメータ θ が区間に含まれる確率が 95% である」という解釈ができる
- 確信区間は 2 種類あるが，まずは Equal-Tail Interval の解説から

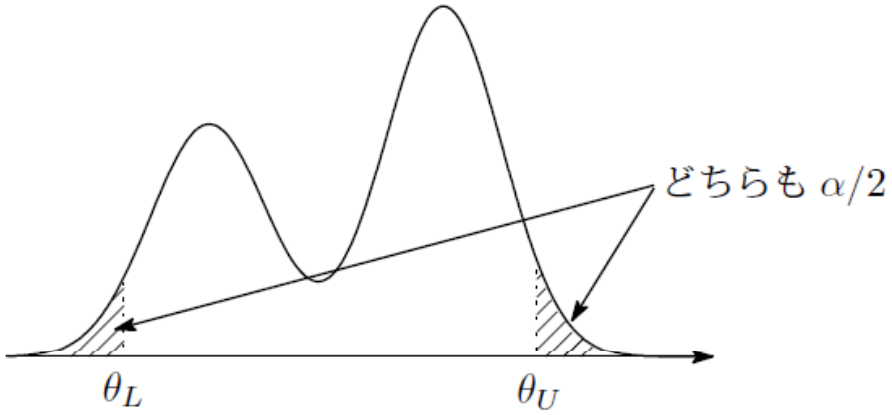
# α % Equal-Tail Interval

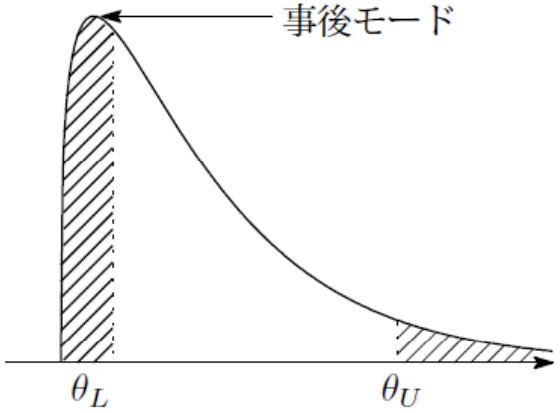
- 事後分布の右端から α/2 の面積と左端 α/2% の面積を除いた部分




- 分布がどんな形であっても「右端 α/2 と左端 α/2 を除いた部分」を確信区間とするため，確信度が高い部分が確信区間から除かれる可能性がある

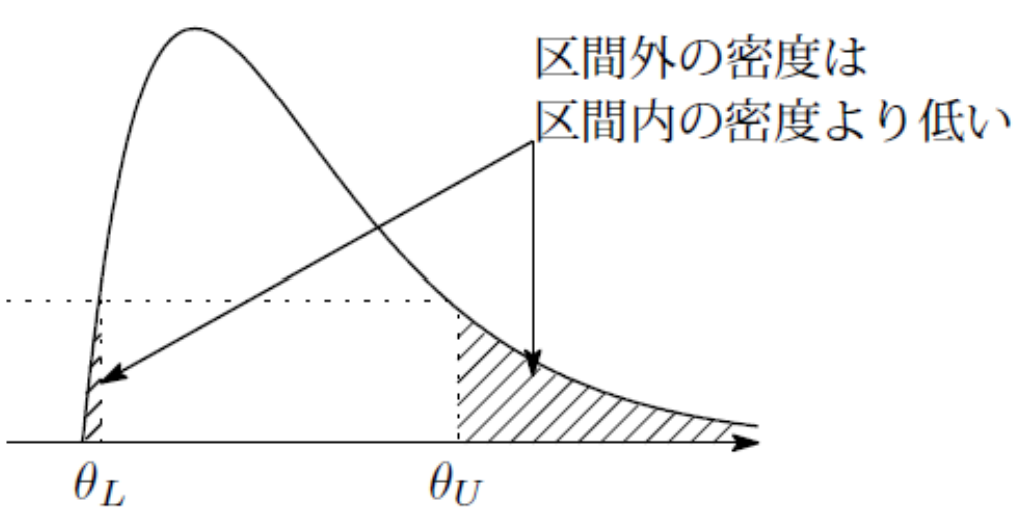


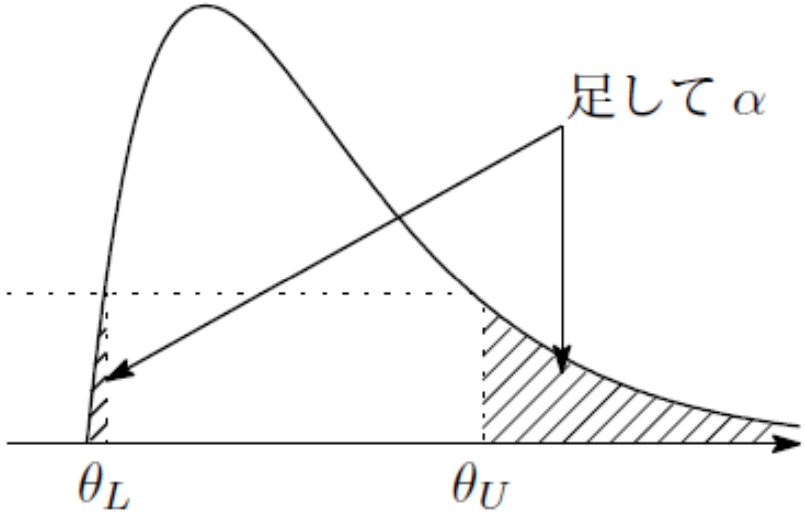




※ 事後モード：事後分布のモード（最頻値）

# α % HPD Interval（Highest Posterior Density Interval）

- 「確信区間の面積は 1-α 」「確信区間内の密度は区間外の密度よりも必ず高い」の 2 条件を満たす



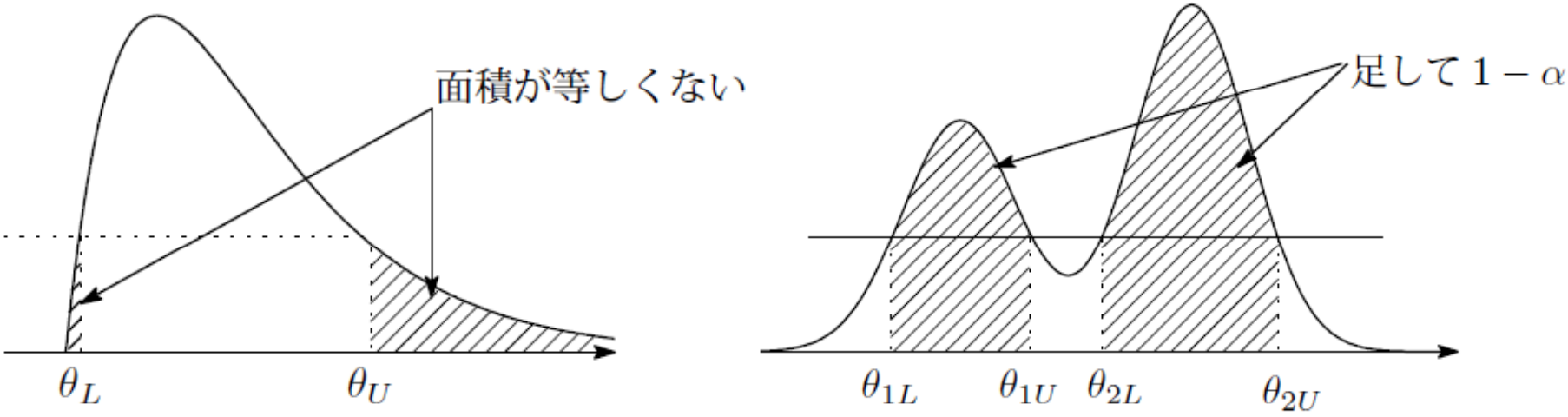






※ HPD Interval：最高事後密度区間

# 95 % HPD Interval（Highest Posterior Density Interval）

- 「確信区間外の右裾と左裾の面積が異なる」
  「分布の形によっては確信区間が分割される」という特徴がある




```
> HPDinterval(result, prob=0.95)   # これは HPD

[[1]]
       lower  upper
theta 0.1091 0.7684
attr(,"Probability")
[1] 0.95
```

# 「超」基本なのでここでは扱わないが大事な事項

- 共役分布について：
  データが 2 値：事前分布も事後分布もベータ分布
  データが連続（分散既知）：事前分布も事後分布も正規分布，等
- 事前分布の選び方：
  無情報事前分布，共役事前分布，悲観的事前分布，等
- マルコフ連鎖モンテカルロ法について：
  仕組み：マルコフ連鎖，定常分布，等
  連鎖の数：複数の chain が望ましい
  burn-in（最初に捨てる乱数）：マルコフ連鎖が収束するまでは捨てる
  テクニック：中心化しておいたほうが推定がうまく行きやすい

# 本日のメニュー

1. 条件付き確率とベイズの定理
2. ベイズの定理の適用例
3. マルコフ連鎖モンテカルロ法
4. ベイズ統計の適用例
   - 正規分布（分散既知）の問題
   - ロジスティック回帰分析
   - 単回帰分析

【参考】WinBUGS 上でベイズ推定を行う手順

# 例 1：正規分布（分散既知）の問題

- うつ病患者における QOL について調査することを考える
- 事前情報ではうつ病患者の QOL の平均は 5，分散が 9 となっていた
- ここで，あるうつ病患者集団 5 人の QOL を測定したところ，
  平均値は 6 （データ：8，4，6，7，5）であった
- 我々は QOL の分散が 10 であると知っている（　分散既知の仮定）
- QOL （パラメータを μ とする）を幾らと推定するか？
- 場面設定は以下
  - μ：QOL の平均
  - p(μ)：μ の事前分布は「平均 5，分散 9 の正規分布」に従うと仮定
  - y：データ，「平均 μ，分散 10 の正規分布」に従うと仮定
  - $p(y|\mu) \propto p(y|\mu) \times p(\mu)$ に従う乱数を R2WinBUGS により生成

# 例 1：正規分布（分散既知）の問題

1. 作業ディレクトリに以下が記述された winbugs-1.txt を作成する

```
# model
model {
  mu ~ dnorm(5, 0.012345)
  for (i in 1:n) {
    y[i] ~ dnorm(mu, 0.01)
  }
}
```

2. データ入力，パラメータの初期値設定を行う

```
> y    <- c(8, 4, 6, 7, 5)
> n    <- length(y)
> data <- list("n", "y")
> init <- list( list(mu=6) )    # listの中にlistで指定する
> parameters <- c("mu")
```



※ 正規分布 dnorm() の第 2 引数は分散ではなく「分散の逆数」であるので，
μ（mu） dnorm() の第 2 引数は $1/9^2 = 0.012345$，y の第 2 引数は $1/10^2 = 0.01$

# 例 1：正規分布（分散既知）の問題

1. 作業ディレクトリに以下が記述された winbugs-1.txt を作成する

```
# model
model {
  mu ~ dnorm(5, 0.012345)
  for (i in 1:n) {
    y[i] ~ dnorm(mu, 0.01)
  }
}
```

- データ y が複数レコードあるので，データ数 n 回分だけ for 文を回す
- y[1] , ・ ・ ・ , y[5] はそれぞれ「平均 μ ，分散 10 の正規分布」に従うので，
  for (i in 1:n) { } の中で y[i] ~ dnorm(mu, 0.01) とする
- パラメータ μ（mu）は，くり返す必要が無いので for 文の外側



※ 正規分布 dnorm() の第 2 引数は分散ではなく「分散の逆数」であるので，
μ（mu） dnorm() の第 2 引数は $1/9^2 = 0.012345$，y の第 2 引数は $1/10^2 = 0.01$

# 例 1：正規分布（分散既知）の問題

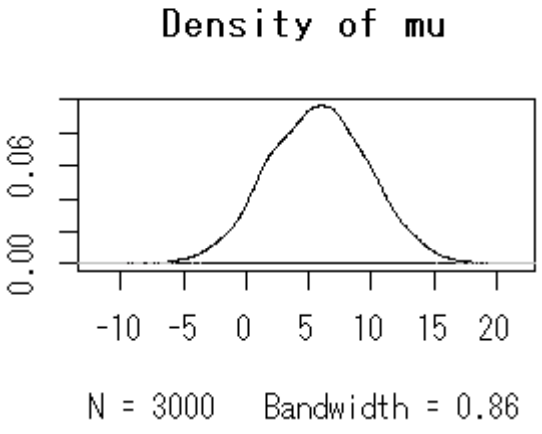
3. 関数 bugs() を実行する

```
> tmp    <- bugs(data, init, parameters, model.file="winbugs-1.txt",
+                n.chains=1, n.thin=3, n.burnin=1000, n.iter=10000,
+                DIC=FALSE, debug=FALSE, codaPkg=TRUE,
+                bugs.dir="C:/Program Files/WinBUGS14/")
> result <- read.bugs(tmp)
> summary(result)   # 事後分布の要約

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

          Mean             SD       Naive SE Time-series SE
       5.85673        4.05233        0.07398        0.08649

2. Quantiles for each variable:

  2.5%    25%    50%    75%  97.5%
-2.092  3.146  5.904  8.582 13.900
```

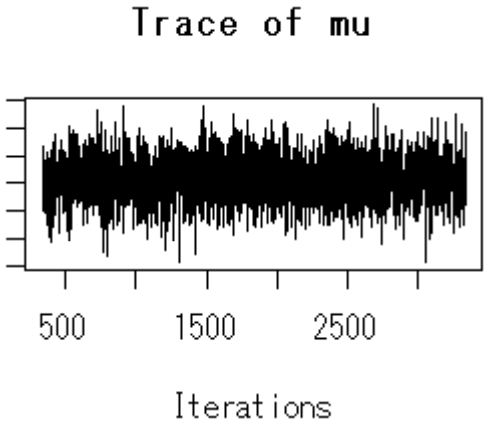
# 例 1：正規分布（分散既知）の問題

3. 結果のグラフ化

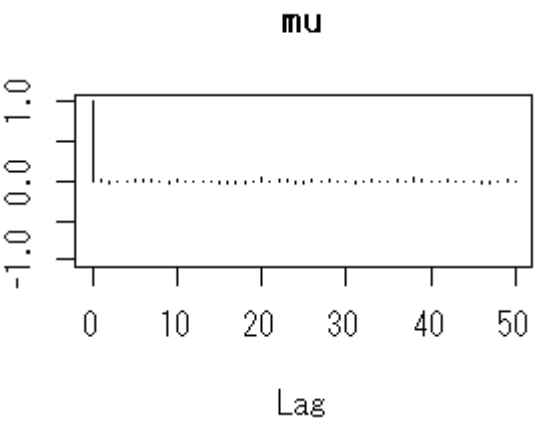
```
> densplot(result)
> traceplot(result)
> autocorr.plot(result, lag.max=50)
```




【事後分布のグラフ】




【トレースプロット】




【自己相関のグラフ】

# 例 1：正規分布（分散既知）の問題

- うつ病患者における QOL について調査することを考える
- 事前情報ではうつ病患者の QOL の平均は 5，分散が 9 となっていた
- ここで，あるうつ病患者集団 5 人の QOL を測定したところ，
  平均値は 6（データ：8，4，6，7，5）であった
- 我々は QOL の分散が 10 であると知っている（　分散既知の仮定）
- パラメータ μ の事後平均は 5.85，事後標準偏差は 4.05
  （分散は $4.05^2 = 16.40$ ）となり，両側95%確信区間
  （Equal-Tail Interval）は [-2.092 13.900] となった
  ⇒ パラメータ μ の事後分布の平均は 5.85 となったので，
  QOL は 5.85 であると推定した

# 例 2 の準備：データ「AB」の読み込み

1. データ「winbugs-AB.csv」を以下からダウンロードする
   http://www.cwk.zaq.ne.jp/fkhud708/files/R-intro/R-stat-intro_data.zip
2. 「winbugs-AB.csv」を「C:/temp」に格納する
3. R を起動し，2. の場所に移動し，データを読み込む

```
> setwd("c:/temp")                    # csv がある場所に移動
> AB <- read.csv("winbugs-AB.csv") # csv を読み込む
> head(AB)
  GROUP y DURATION
1     1 1        1
2     1 1        3
3     1 1        2
4     1 1        4
:     : :        :
```

# 例 2 の準備：架空のデータ「AB」の変数

- GROUP：薬剤の種類（A：1，B：0）
- y：改善の有無（1：改善あり，0：改善なし）
- DURATION：罹病期間（数値，単位は年）

# 例 2 の準備：架空のデータ「AB」

| GROUP | y | DURATION |
|---|---|---|
| 1 | 1 | 1 |
| 1 | 1 | 3 |
| 1 | 1 | 2 |
| 1 | 1 | 4 |
| 1 | 1 | 2 |
| 1 | 1 | 2 |
| 1 | 1 | 4 |
| 1 | 1 | 2 |
| 1 | 1 | 5 |
| 1 | 1 | 7 |
| 1 | 0 | 4 |
| 1 | 0 | 6 |
| 1 | 0 | 3 |
| 1 | 0 | 7 |
| 1 | 0 | 8 |
| 1 | 1 | 6 |
| 1 | 1 | 5 |
| 1 | 0 | 7 |
| 1 | 0 | 12 |
| 1 | 0 | 10 |

| GROUP | y | DURATION |
|---|---|---|
| 0 | 1 | 9 |
| 0 | 1 | 5 |
| 0 | 1 | 2 |
| 0 | 0 | 7 |
| 0 | 0 | 2 |
| 0 | 1 | 11 |
| 0 | 1 | 3 |
| 0 | 0 | 6 |
| 0 | 0 | 7 |
| 0 | 0 | 13 |
| 0 | 0 | 15 |
| 0 | 0 | 9 |
| 0 | 0 | 8 |
| 0 | 0 | 7 |
| 0 | 0 | 9 |
| 0 | 0 | 8 |
| 0 | 0 | 2 |
| 0 | 0 | 10 |
| 0 | 0 | 8 |
| 0 | 0 | 4 |

# 例 2：ロジスティック回帰分析

- うつ病を患っている患者さん n=40 人に薬剤を投与し，「改善あり」となる割合を評価する
- GROUP を薬剤の種類（ A=1 又は B=0 ）とする
- DURATION を罹病期間（単位は年）とする
- y を改善の有無（ 1：改善あり，0：改善なし）を表す確率変数で，ベルヌーイ分布に従うとする
- パラメータ $\alpha$，$\beta_1$，$\beta_2$ の事前分布をいずれも正規分布：N(0, 10000) とし，以下のロジスティック回帰モデルを考え，パラメータ $\alpha$ ，$\beta_1$，$\beta_2$ の事後分布を求める

  改善の有無の対数オッズ = $\alpha + \beta_1 \times$GROUP + $\beta_2 \times$DURATION

# 例 2：ロジスティック回帰分析

1. 作業ディレクトリに以下が記述された winbugs-2.txt を作成する

```
model {
  alpha ~ dnorm(0, 1.0E-5)
  beta1 ~ dnorm(0, 1.0E-5)
  beta2 ~ dnorm(0, 1.0E-5)
  for (i in 1:n) {
    logit(p[i]) <- alpha + beta1*GROUP[i] + beta2*DURATION[i]
    Y[i] ~ dbern(p[i])
  }
}
```

- p[i] を確率，logit(p[i]) を対数オッズ，for 文の中で logit(p[i]) に代入
- y[i] は確率 p[i] のベルヌーイ分布に従うので，y[i] ~ dbern(p[i]) と記述
- パラメータ α，$\beta_1$，$\beta_2$ は正規分布：N(0, 10000) に従うので，
  パラメータ名y[i] ~ dnorm(0,1.0E-5) と記述（くり返さないので for 文の外側）



※ 正規分布 dnorm() の第 2 引数は分散ではなく「分散の逆数」であるので，
dnorm() の第 2 引数は 1/10000 = 0.0001 = 1.0E-5（$1.0 \times 10^5$ の意味）

# 例 2：ロジスティック回帰分析

2. データ入力，パラメータの初期値設定を行う

```
# データ
Y         <- AB$Y
GROUP     <- AB$GROUP
DURATION <- AB$DURATION
n         <- length(Y)
data      <- list("n", "Y", "GROUP", "DURATION")

# パラメータ
init1 <- list(alpha=0, beta1=0, beta2=0)
init2 <- list(alpha=1, beta1=1, beta2=1)
init3 <- list(alpha=2, beta1=2, beta2=2)
inits <- list(init1, init2, init3)
parameters <- c("alpha", "beta1", "beta2")
```

- chain（事後分布に従う乱数の列）を複数発生させる場合は上記のようにする
  ⇒ 初期値を複数設定し，リストとして 1 つの変数 inits に格納

# 例 2：ロジスティック回帰分析

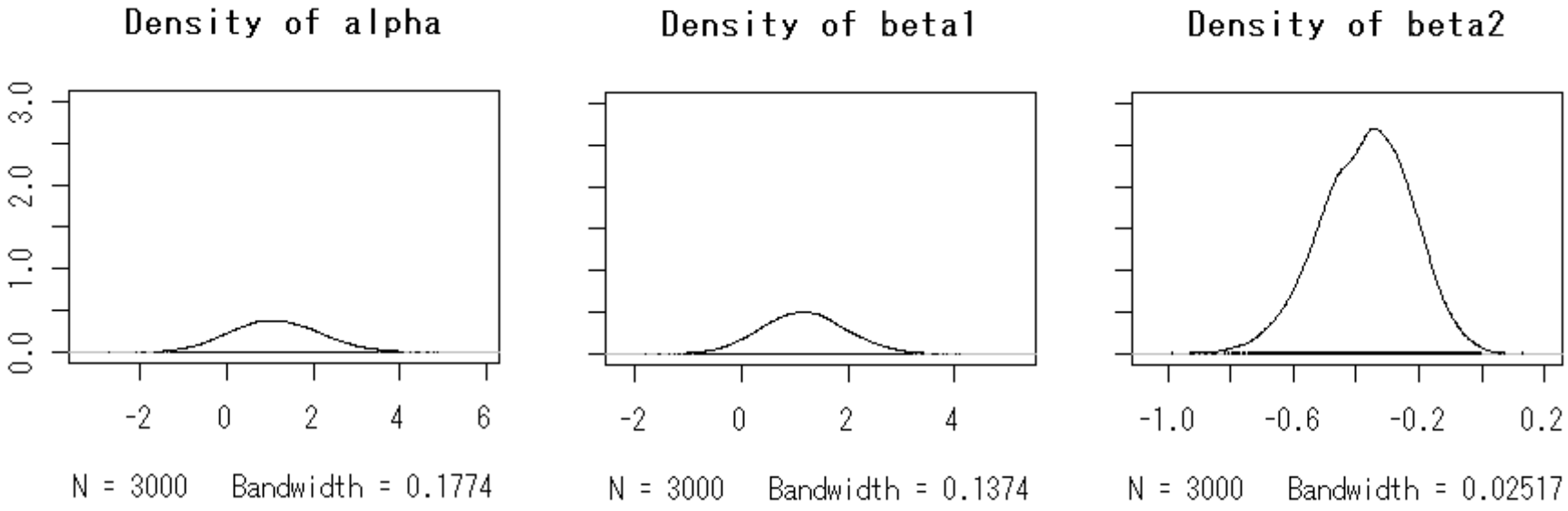
3. 関数 bugs() を実行する

```
> tmp    <- bugs(data, inits, parameters, model.file="winbugs-2.txt",
+                n.chains=3, n.thin=3, n.burnin=1000, n.iter=10000,
+                DIC=FALSE, debug=FALSE, codaPkg=TRUE,
+                bugs.dir="C:/Program Files/WinBUGS14/")
> result <- read.bugs(tmp)
> summary(result)   # 事後分布の要約

          Mean     SD Naive SE Time-series SE
alpha  1.1251 1.0337 0.010896       0.019867
beta1  1.1796 0.8008 0.008441       0.010579
beta2 -0.3685 0.1467 0.001546       0.002700

2. Quantiles for each variable:

          2.5%     25%     50%     75%   97.5%
alpha -0.8340  0.4198  1.1085  1.8220  3.2191
beta1 -0.3647  0.6346  1.1670  1.7080  2.7740
beta2 -0.6737 -0.4657 -0.3599 -0.2657 -0.1015
```

# 例２：ロジスティック回帰分析

3. 結果のグラフ化

```
> densplot(result)
> traceplot(result)
> autocorr.plot(result, lag.max=50)
```




【事後分布のグラフ】

# 例 2：ロジスティック回帰分析

- うつ病を患っている患者さん n=40 人に薬剤を投与し，「改善あり」となる割合を評価する
- GROUP を薬剤の種類（ A=1 又は B=0 ）とする
- DURATION を罹病期間（単位は年）とする
- y を改善の有無（ 1：改善あり，0：改善なし）を表す確率変数で，ベルヌーイ分布に従うとする
- パラメータ α，$\beta_1$，$\beta_2$ の事前分布をいずれも正規分布：N(0, 10000) とし，以下のロジスティック回帰モデルを考え，パラメータ α ，$\beta_1$，$\beta_2$ の事後分布（結果は前々頁）より，以下のモデルと推定された

  改善の有無の対数オッズ = 1.1251 + 1.1796×GROUP
  − 0.3685×DURATION

# 【参考】例 2'：データを行列で渡す場合

1. 作業ディレクトリに以下が記述された winbugs-2_2.txt を作成する

```
model {
  alpha ~ dnorm(0, 1.0E-5)
  beta1 ~ dnorm(0, 1.0E-5)
  beta2 ~ dnorm(0, 1.0E-5)
  for (i in 1:n) {
    logit(p[i]) <- alpha + beta1*X[i,1] + beta2*X[i,2]
    Y[i] ~ dbern(p[i])
  }
}
```

- GROUP と DURATION のデータを行列 X として格納し，WinBUGS に渡すことを考える

# 【参考】例 2'：データを行列で渡す場合

2. データ入力，パラメータの初期値設定を行い，関数 bugs を実行

```
> # データ
> Y      <- AB$Y
> X      <- matrix( c(AB$GROUP,AB$DURATION), ncol=2)
> n      <- length(Y)
> data   <- list("n", "Y", "X")

> # パラメータ
> init1 <- list(alpha=0, beta1=0, beta2=0)
> init2 <- list(alpha=1, beta1=1, beta2=1)
> init3 <- list(alpha=2, beta1=2, beta2=2)
> inits <- list(init1, init2, init3)
> parameters <- c("alpha", "beta1", "beta2")

> # 実行
> tmp    <- bugs(data, inits, parameters, model.file="winbugs-2_2.txt",
+                n.chains=3, n.thin=3, n.burnin=1000, n.iter=10000,
+                DIC=FALSE, debug=TRUE, codaPkg=TRUE,
+                bugs.dir="C:/Program Files/WinBUGS14/")
> result <- read.bugs(tmp)
```

# 例 3：単回帰分析

- $\mathbf{x}$ = (1, 2, 3, 4, 5)，$\mathbf{y}$=(1, 2, 3, 4, 5.1) について以下の回帰式を考える

  $y_i = \beta_1 + \beta_2 \times x_i + \varepsilon_i \qquad \varepsilon_i \sim N(0,\ 1/\tau_1) \qquad (i=1,\ldots,5)$

- 上記モデルから以下の関係式を得る

  $y_i \sim N(\mu_i,\ 1/\tau_1)$

  $\mu_i = \beta_1 + \beta_2 \times x_i \qquad (i=1,\ldots,5)$

- また，パラメータ $\tau_1$ と $\beta_j$ (j=1, 2)，及び超パラメータ $\tau_2$ について，以下の事前分布を仮定する

  $\beta_j \sim N(0,\ \tau_2) \qquad (j=1,2)$

  $\tau_j \sim Gamma(0.001,\ 0.001) \qquad (j=1,2)$

  $\sigma_j = 1/(\tau_j)^{1/2} \qquad (j=1,2)$

- 各パラメータの事後分布を求めるために，以下のベイズの定理を用いる

  $p(\beta_1,\ \beta_2,\ \tau_1,\ \tau_2|\mathbf{x},\ \mathbf{y}) \propto p(\mathbf{y}|\beta_1,\ \beta_2,\ \tau_1,\ \tau_2,\ \mathbf{x})$

  $\times\ p(\tau_1) \times p(\beta_1|\tau_2) \times p(\beta_2|\tau_2) \times p(\tau_2)$



※ 超パラメータ：hyperparaeter

# 例 3：単回帰分析

1. 作業ディレクトリに以下が記述された winbugs-3.txt を作成する

```
model {
  for (i in 1:n) {
    y[i]  ~  dnorm(mu[i], tau[1])
    mu[i] <- beta[1] + beta[2]*x[i]
  }
  for (i in 1:2) {
    beta[i]  ~  dnorm(0, tau[2])
    tau[i]   ~  dgamma(0.001, 0.001)
    sigma[i] <- sqrt(1/tau[i])
  }
}
```

# 例 3：単回帰分析

2. データ入力，パラメータの初期値設定を行う

```
> x      <- c(1, 2, 3, 4, 5)
> y      <- c(1, 2, 3, 4, 5.1)
> n      <- 5
> data   <- list("x", "y", "n")
> inits <- list( list(beta=c(0,0), tau=c(1,1)) ) # list中にlistで指定
> parameters <- c("beta","tau","sigma")
```

3. 関数 bugs() を実行する

```
> tmp    <- bugs(data, inits, parameters, model.file="winbugs-3.txt",
+                n.chains=1, n.thin=3, n.burnin=1000, n.iter=10000,
+                DIC=FALSE, debug=TRUE, codaPkg=TRUE,
+                bugs.dir="C:/Program Files/WinBUGS14/")
> result <- read.bugs(tmp)
```

# 例 3：単回帰分析

```
> summary(result)

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

             Mean        SD  Naive SE Time-series SE
beta[1]   -0.03515   0.07561 0.0013804      0.0027502
beta[2]    1.01850   0.02283 0.0004168      0.0007662
sigma[1]   0.06150   0.04444 0.0008113      0.0011984
sigma[2]   1.30112   3.04551 0.0556031      0.0573345
tau[1]   505.76762 419.05705 7.6509000      8.0280394
tau[2]     1.94275   1.94666 0.0355409      0.0427187

2. Quantiles for each variable:
             2.5%       25%       50%        75%     97.5%
beta[1]  -0.18044  -0.07094  -0.03705  -0.001757    0.1252
beta[2]   0.97309   1.00800   1.01900   1.029000    1.0630
sigma[1]  0.02493   0.03801   0.04996   0.070193    0.1695
sigma[2]  0.37547   0.60933   0.85915   1.337000    4.6811
tau[1]   34.80975 202.97500 400.50000 692.150000 1609.0500
tau[2]    0.04563   0.55962   1.35450   2.693500    7.0933
```

# 本日のメニュー

1. 条件付き確率とベイズの定理
2. ベイズの定理の適用例
3. マルコフ連鎖モンテカルロ法
4. ベイズ統計の適用例
   - 正規分布（分散既知）の問題
   - ロジスティック回帰分析
   - 単回帰分析

【参考】WinBUGS 上でベイズ推定を行う手順

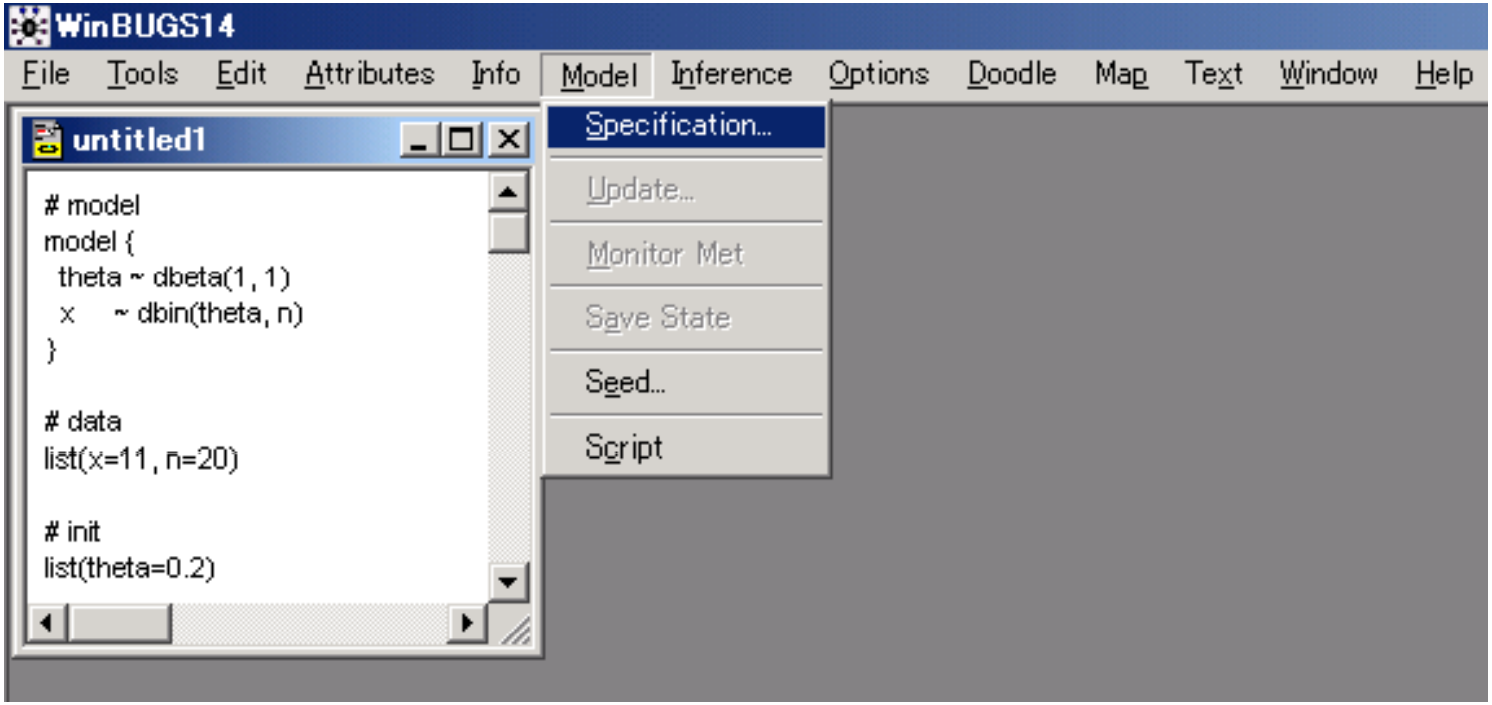
# 【参考】 WinBUGS 上でベイズ推定を行う手順

1. WinBUGS を起動し，[File] → [New] を選択するとエディタが開くので，そこにモデル式，データ，初期値を記述する

```
# model
model {
  theta ~ dbeta(1, 1)
  y     ~ dbin(theta, n)
}

# data
list(y = 2, n = 5)

# init
list(theta = 0.5)
```

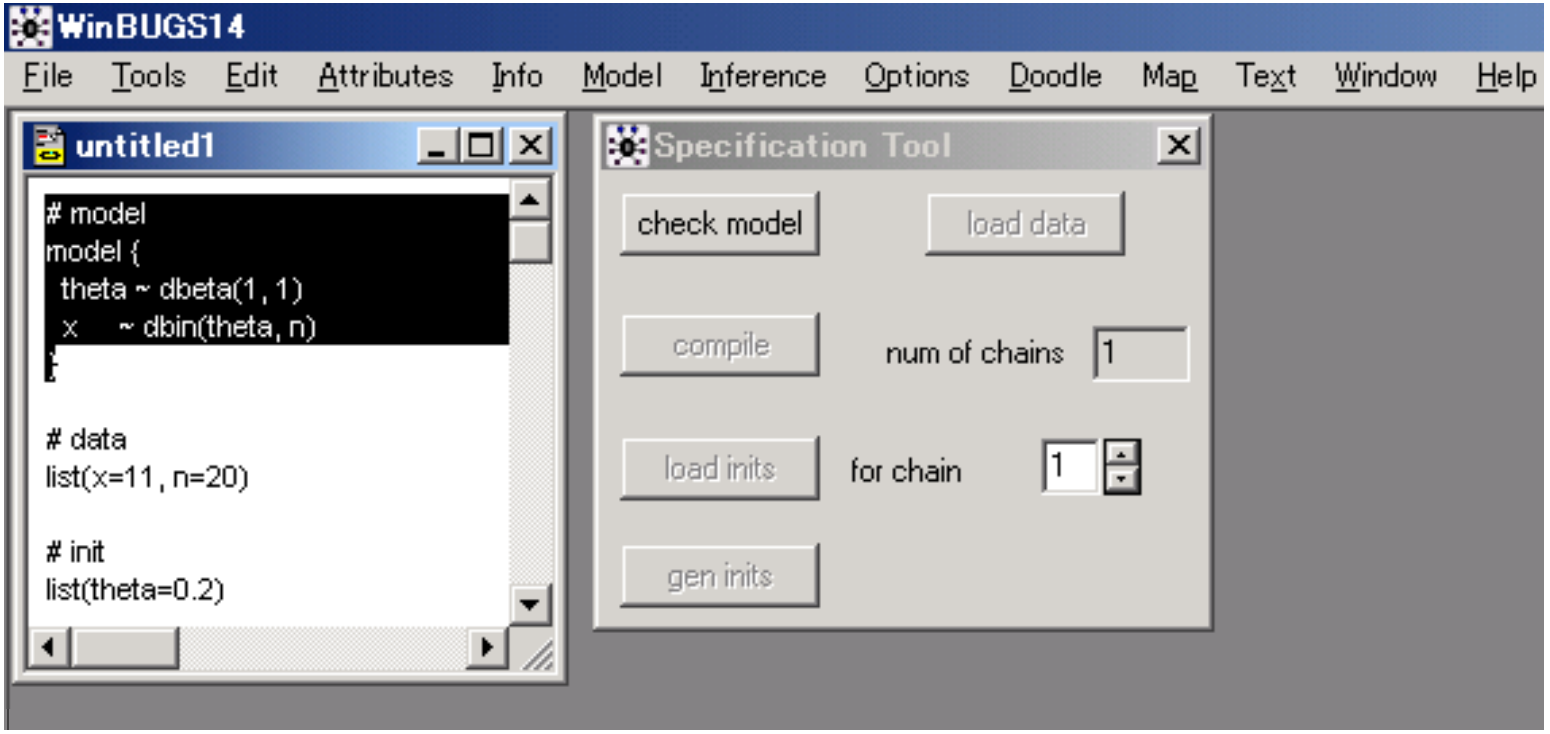
# 【参考】 WinBUGS 上でベイズ推定を行う手順

2. [Model] → [Specification...] を選択すると「Specification Tool」のウインドウが表示される
   もし，エラーメッセージを確認したい場合は [Info] → [Open Log] を選択してログウインドウを表示する



# 【参考】 WinBUGS 上でベイズ推定を行う手順

3. モデル式全体，又は文字列「model」のみをマウスで選択して [Check model] をクリックし，WinBUGSにモデル式をチェックしてもらう



# 【参考】 WinBUGS 上でベイズ推定を行う手順

4. チェックした結果，問題がなければ，(3)と同様の方法で，
「#data」の部分をマウスで選択して [load data] をクリックする

5. [compile] をクリックして命令をコンパイルする

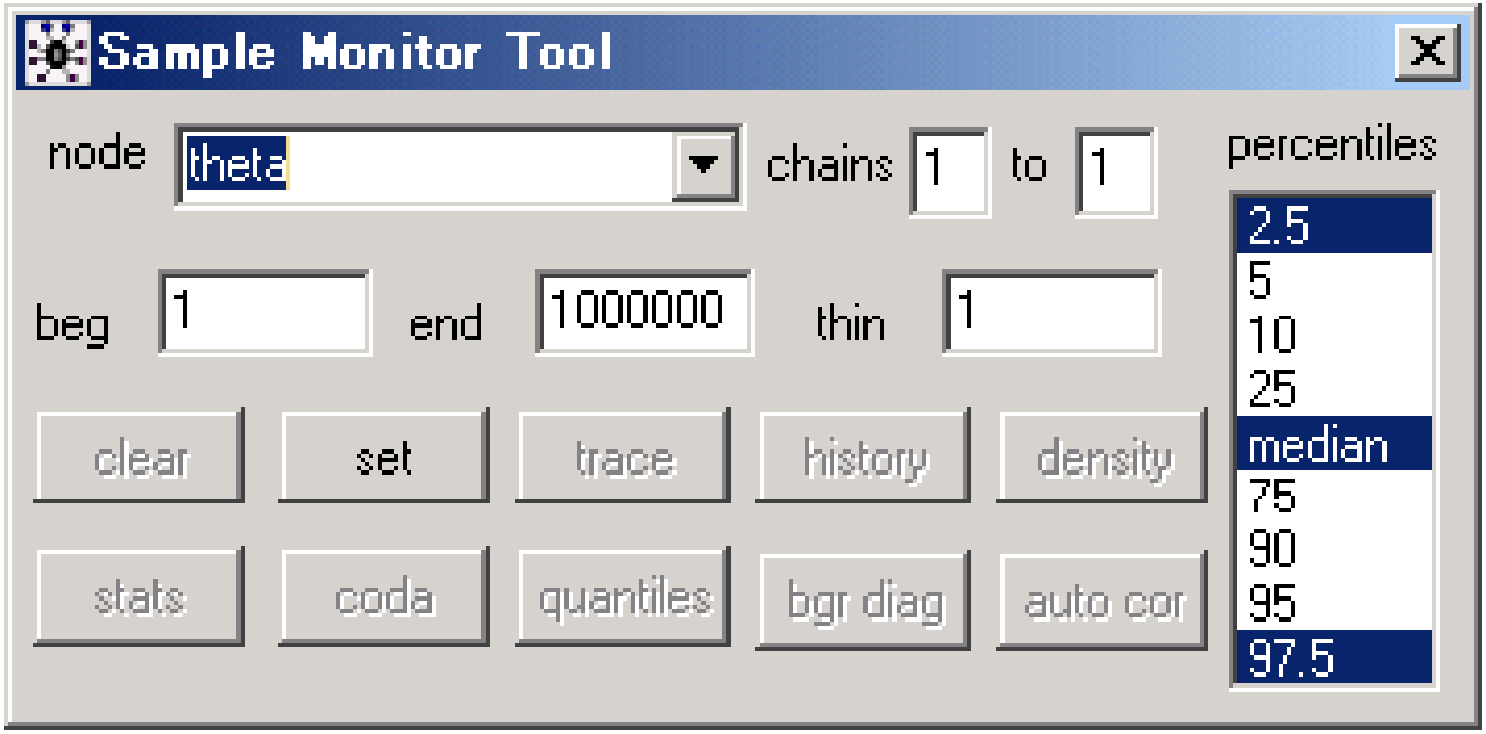
6. コンパイルした結果，問題がなければ，
   - 初期値を設定する場合は，3. と同様の方法で，「#init」の部分をマウスで選択して [load init] をクリックする
   - 初期値設定をWinBUGSに任せる場合は，[gen inits] をクリックする
（事前分布からの乱数が初期値に使われる）

7. [Inference] → [Samples...] をクリックして「Sample Monitor Tool」のウインドウを表示する

# 【参考】 WinBUGS 上でベイズ推定を行う手順

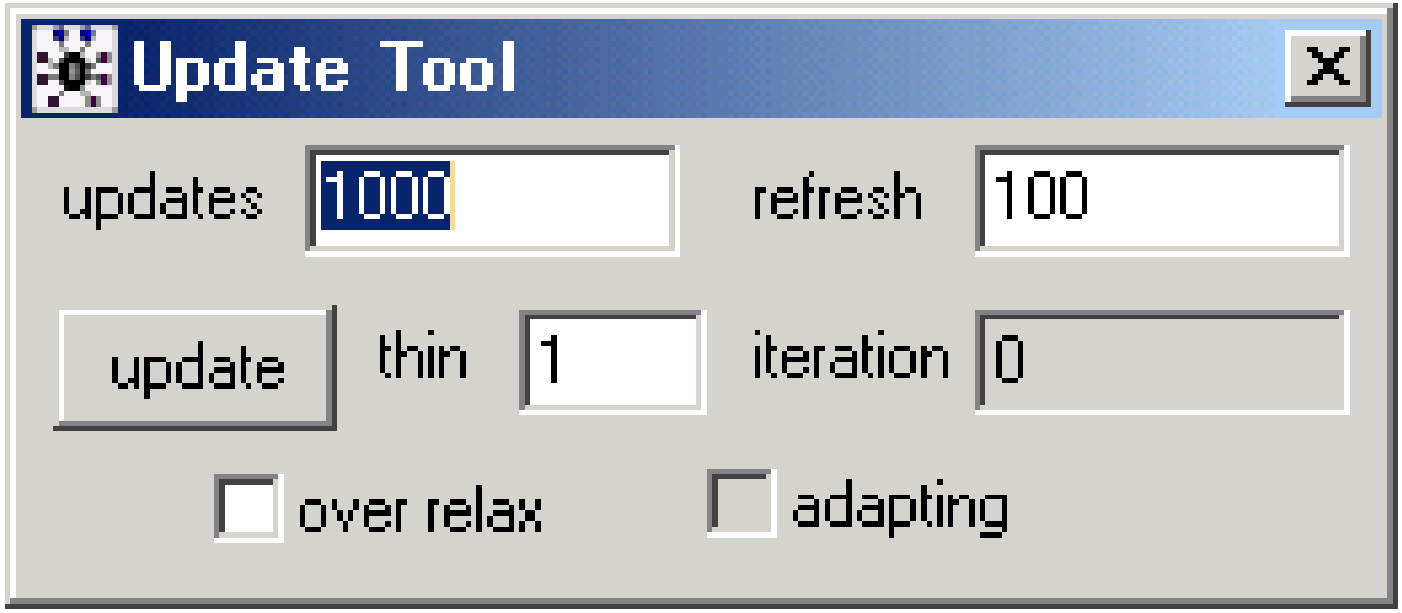
8. 「Sample Monitor Tool」の [node] にパラメータを指定して [set] をクリックする

   パラメータが複数ある場合は，パラメータ数だけ手順を繰り返す



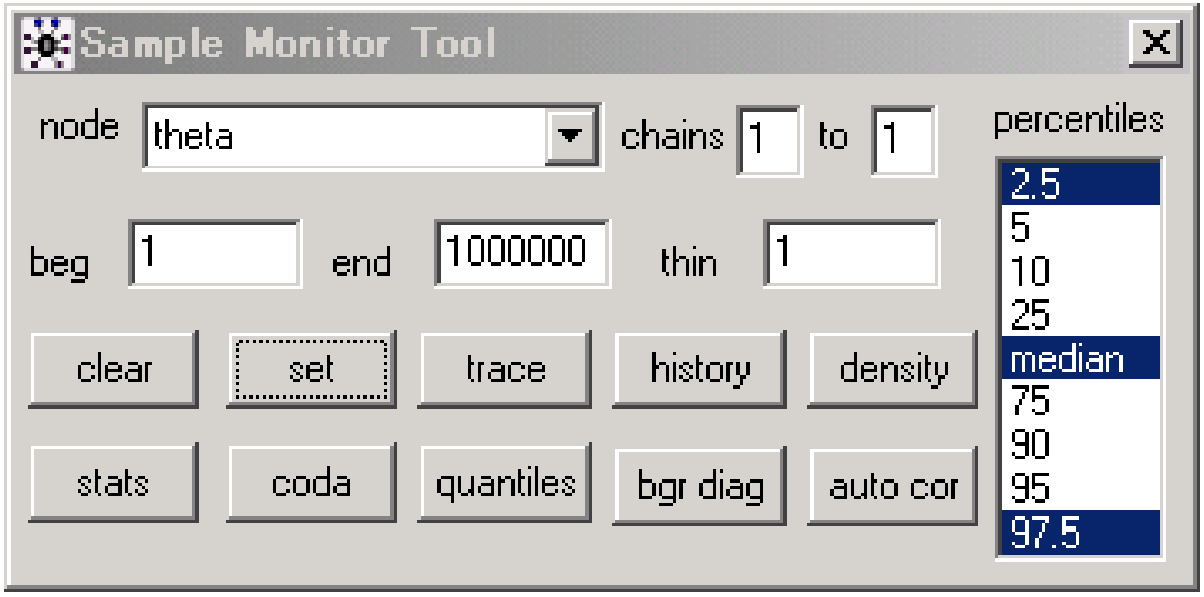
# 【参考】 WinBUGS 上でベイズ推定を行う手順

9. パラメータの指定が完了したら，[Model] → [Update...] を選択する「Update Tool」のウインドウが表示されるので，各種設定を行った後，事後分布からのサンプリングを行う



# 【参考】 WinBUGS 上でベイズ推定を行う手順

10. 結果を確認する場合は，「node」から確認したいパラメータを選択した後，「Sample Monitor Tool」のウインドウから
「density（事後分布の密度関数）」「stats（事後分布の統計量）」などを表示する
「node」に「*」を入力すれば，全パラメータの結果が表示される



# 本日のメニュー

1. 条件付き確率とベイズの定理
2. ベイズの定理の適用例
3. マルコフ連鎖モンテカルロ法
4. ベイズ統計の適用例
   - 正規分布（分散既知）の問題
   - ロジスティック回帰分析
   - 単回帰分析

【参考】WinBUGS 上でベイズ推定を行う手順

# 参考文献


- 統計学（白旗 慎吾 著，ミネルヴァ書房）
- 道具としてのベイズ統計（涌井 良幸 著，日本実業出版社）
- ベイズ統計学入門（渡部 洋 著，福村出版）
- Bayesian Approaches to Clinical Trials and Health-Care Evaluation（ David J. Spiegelhalter et. al. 著，Wiley）
- Understanding Computational Bayesian Statistics（William M. Bolstad 著，Wiley）
- The R Tips 第 2 版（オーム社）
- R 流！イメージで理解する統計処理入門（カットシステム）

とWinBUGS

# Rで統計解析入門

終