

統計モデリング入門 2017 (f)

階層ベイズモデル Hierarchical Bayesian Model

久保拓弥 kubo@ees.hokudai.ac.jp

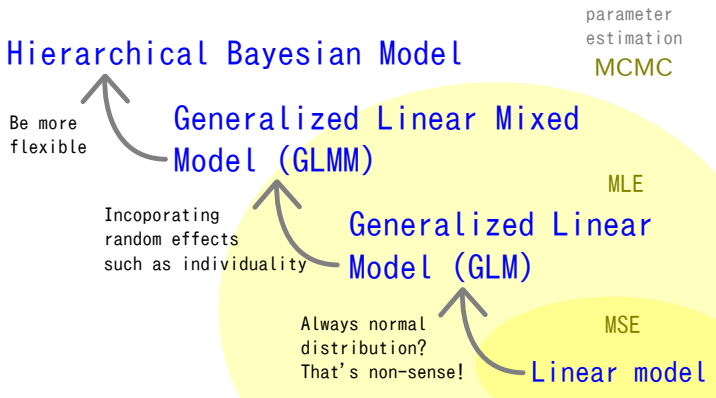
北大環境科学院の講義 <http://goo.gl/76c4i>

2019-08-05

ファイル更新時刻: 2019-07-18 17:35

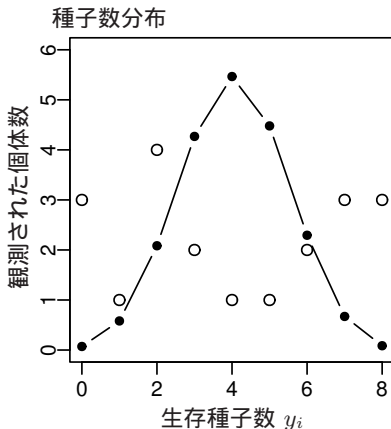
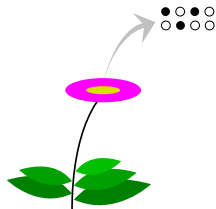
今日の統計モデル: 階層ベイズモデル

The development of linear models



そして **Markov Chain Monte Carlo (MCMC)**
を使った **Bayesian Estimation (ベイズ推定)**

Why? GLM is not enough ...



N 個のうち y 個
...という形式のデータ
なのに
二項分布ではまったく
説明できない?

階層ベイズモデルが必要!

Apply Hierarchical Bayesian Model (HBM)!

今日のハナシ

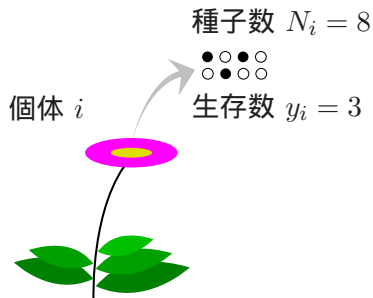
1. MCMC sampling のための例題

logistic regression: binomial distribution

and logit link function

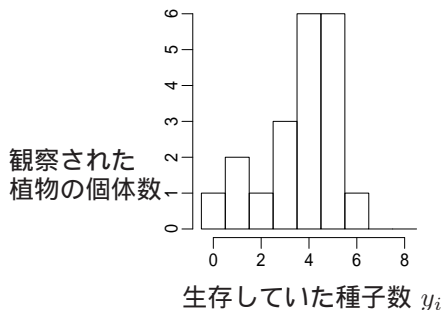
例題: 植物の種子の生存確率

- 架空植物の種子の生存を調べた
- 種子: 生きていれば発芽する
 - どの個体でも **8 個** の種子を調べた
- 生存確率: ある種子が生きている確率
- データ: 植物 **20** 個体, 合計 **160** 種子の生存の有無を調べた
- 73 種子が生きていた — このデータを統計モデル化したい



二項分布を説明するための例題

個体ごとの生存数	0	1	2	3	4	5	6	7	8
観察された個体数	1	2	1	3	6	6	1	0	0



これは個体差なしの均質な集団 (no individual difference!)

生存確率 q と二項分布の関係

- 生存確率 q を推定するために**二項分布** という確率分布を使う
- 個体 i の N_i 種子中 y_i 個が生存する確率

$$p(y_i | q) = \binom{N_i}{y_i} q^{y_i} (1 - q)^{N_i - y_i},$$

- In this example
 - **個体差はない** no individual difference
 - つまり **すべての個体で同じ生存確率 q**

ゆうど

尤度: 20 個体ぶんのデータが観察される確率

- 観察データ $\{y_i\}$ が確定しているときに
- パラメータ q は値が自由にとりうると考える
- 尤度は 20 個体ぶんのデータが得られる確率の積, パラメータ q の関数として定義される

$$L(q|\{y_i\}) = \prod_{i=1}^{20} p(y_i | q)$$

個体ごとの生存数	0	1	2	3	4	5	6	7	8
観察された個体数	1	2	1	3	6	6	1	0	0

対数尤度 方程式と 最尤推定

- この尤度 $L(q \mid \text{データ})$ を最大化するパラメータの推定量 \hat{q} を計算したい
- 尤度を対数尤度になおすと

$$\begin{aligned}\log L(q \mid \text{データ}) &= \sum_{i=1}^{20} \log \binom{N_i}{y_i} \\ &+ \sum_{i=1}^{20} \{y_i \log(q) + (N_i - y_i) \log(1 - q)\}\end{aligned}$$

- この対数尤度を最大化するように未知パラメーター q の値を決めてやるのが**最尤推定**

最尤推定 (MLE): 二項分布の場合

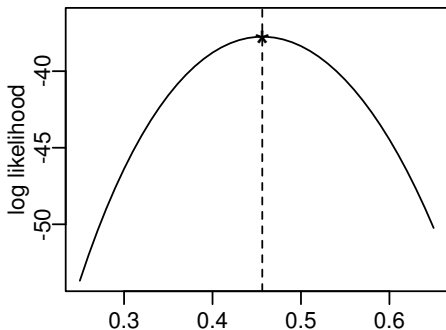
- 対数尤度 $L(q \mid \text{データ})$ が最大になるパラメーター q の値をさがすこと

- 対数尤度 $\log L(q \mid \text{データ})$ を q で偏微分して 0 となる \hat{q} が対数尤度最大

$$\partial \log L(q \mid \text{データ}) / \partial q = 0$$

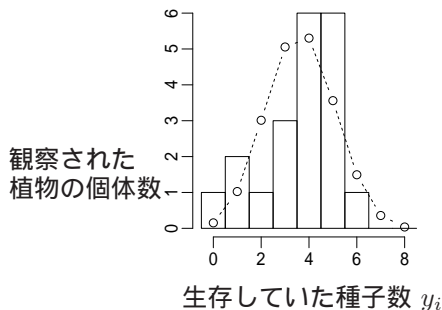
- 生存確率 q が全個体共通の場合の最尤推定量・最尤推定値は

$$\hat{q} = \frac{\text{生存種子数}}{\text{調査種子数}} = \frac{73}{160} = 0.456 \text{ ぐらい}$$



二項分布で説明できる 8 種子中 y_i 個の生存

$$\hat{q} = 0.46 \text{ なので } \binom{8}{y} 0.46^y 0.54^{8-y}$$



2. The same data, but applying Markov Chain Monte-Carlo (MCMC)

最尤推定と Markov chain Monte Carlo (MCMC) はちがう!

そして “なんとなく” ベイズ統計モデルと関連づけ

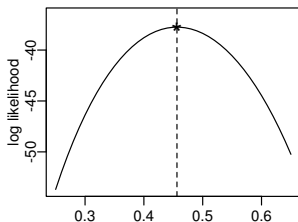
ここでやること: 尤度と MCMC の関係を考える

- さきほどの簡単な例題 (生存確率) のデータ解析を
- 最尤推定ではなく
- Markov chain Monte Carlo (MCMC) 法のひとつである **Metropolis Method** (Metropolis method) であつかう
- 得られる結果: 「パラメーターの値の分布」.....??

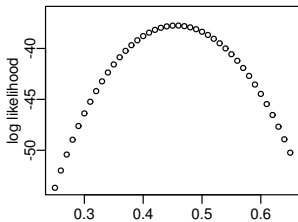
MCMC をもちださなくてもいい簡単すぎる問題
説明のためあえて Metropolis Method を適用してみる

MCMC 法を説明するための例題

連続的な対数尤度関数

 $\log L(q)$ 

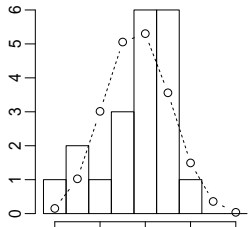
離散化: q がとびとびの値をとる



説明を簡単にするため

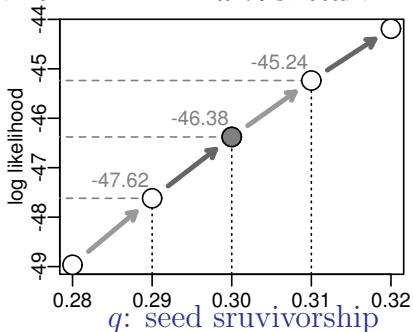
生存確率 q の軸を離散化する

(実際には離散化する必要などない)



試行錯誤による q の最尤推定値の探索

ちょっと効率の悪い「試行錯誤の最尤推定」



- ① q の値の「行き先」を「両隣」どちらかにランダムに決める
- ② 「行き先」が現在の尤度より高ければ、 q の値をそちらに変更
- ③ 尤度が変化しなくなるまで (1), (2) をくりかえす

この例題の Metropolis Method のルール

① パラメーター q の初期値を選ぶ

(ここでは q の初期値が 0.3)

② q を増やすか減らすかをランダムに決める

(新しく選んだ q の値を q_{new} としましょう)

③ q_{new} における尤度 $L(q_{\text{new}})$ ともとの尤度 $L(q)$ を比較

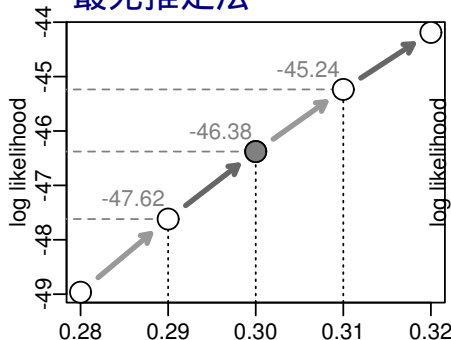
- $L(q_{\text{new}}) \geq L(q)$ (あてはまり改善): $q \leftarrow q_{\text{new}}$
- $L(q_{\text{new}}) < L(q)$ (あてはまり改悪):
 - 確率 $r = L(q_{\text{new}})/L(q)$ で $q \leftarrow q_{\text{new}}$
 - 確率 $1 - r$ で q を変更しない

④ 手順 2. にもどる

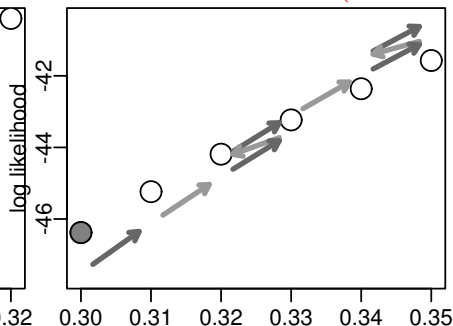
($q = 0.01$ や $q = 0.99$ でどうなるんだ, といった問題は省略)

Metropolis Method Rule: how to move q

最尤推定法



メトロポリス法 (MCMC)

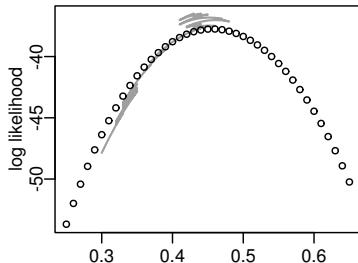


Metropolis Method だと

「単調な山のぼり」にはならない

対数尤度関数の「山」でうろろうろする q の値

Metropolis Method (そして一般の MCMC) は
最適化ではない (not seeking the optimal point!)

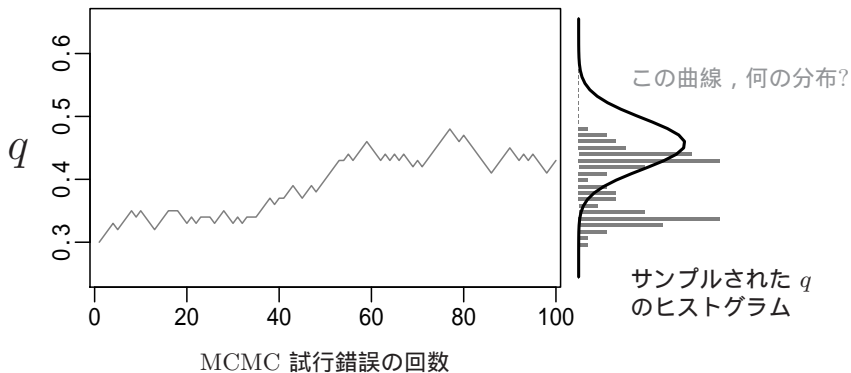


ときどきはでに落っこちる

何のためにこんなことをやるのか? What for MCMC?

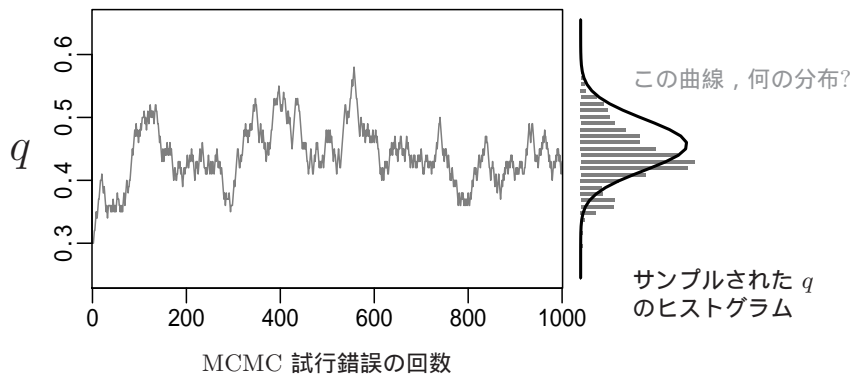
q の変化していく様子を記録してみよう

Sampling q values based on MCMC rules



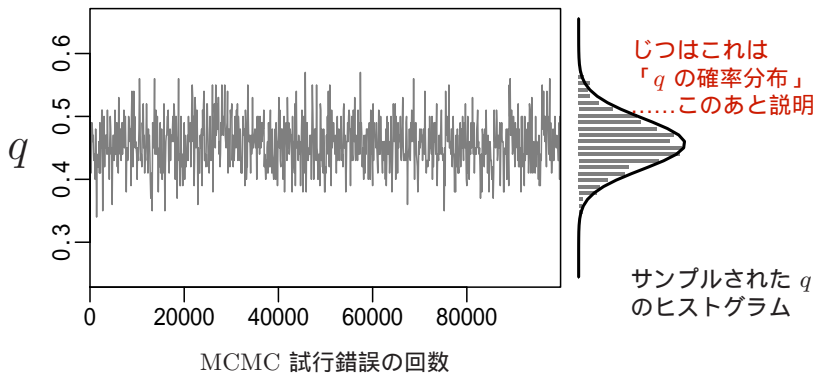
もっと試行錯誤してみたほうがいいのか?

もっと長くサンプリングしてみる longer sampling



まだまだ.....?

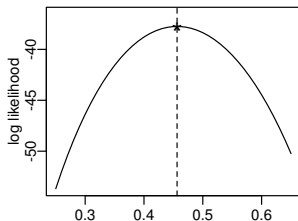
もっともっと長くサンプリングしてみる more ...



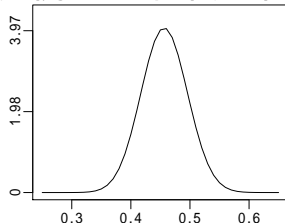
なんだか、ある「山」のかたちにとまとまったぞ?

What for MCMC sampling?

対数尤度 $\log L(q)$



尤度 $L(q)$ に
比例する確率分布

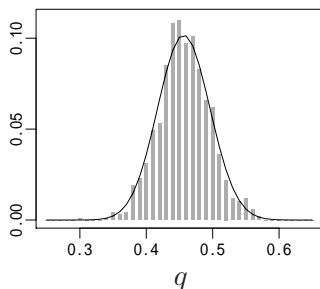


尤度に比例する確率分布からのランダムサンプル

最尤推定はパラメーターの値の点推定

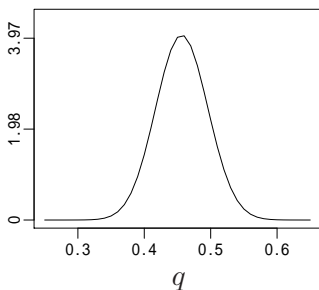
MCMC は “パラメーターの事後分布” (推定したいこと)

は こういう分布ですよ と推定している

MCMC の結果として得られた q の 経験分布

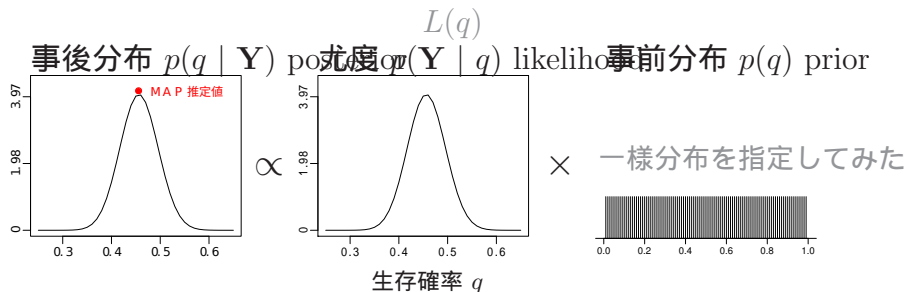
- データと統計モデル (二項分布) を決めて, MCMC サンプルングすると, $p(q)$ からのランダムサンプルが得られる
- このランダムサンプルをもとに, q の平均や 95% 区間などがわかる — 便利じゃないか!

(非ベイズな統計学)

しかし **ふつうの統計学** では q の分布 とかありえない

- パラメーター $q = 0.4500000 \dots$ といったスカラー値であり, 分布ではない!
- 信頼区間は “ q の推定値のばらつき” でもない!
- **ベイズ統計学** なら “パラメーターの分布” としても何の問題もありません

ベイズ統計モデル：尤度・事前分布・事後分布



3. Softwares for MCMC sampling

“Gibbs sampling” などが簡単にできるような.....

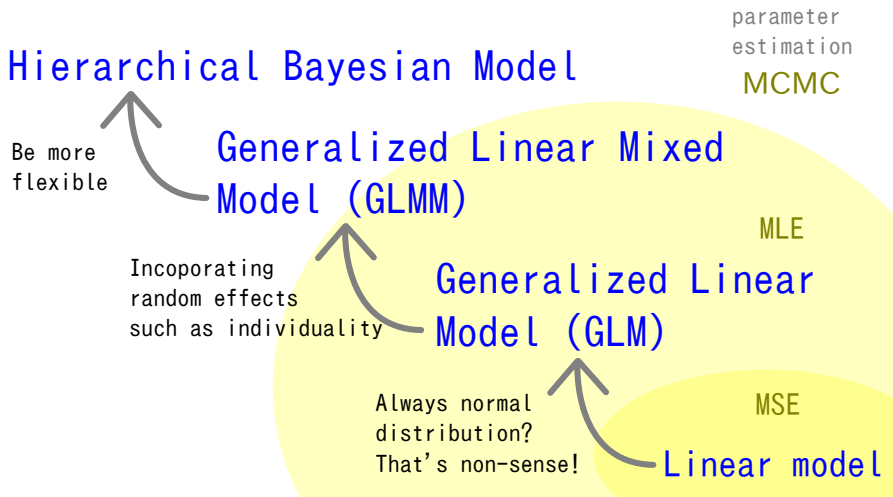
事後分布から効率よくサンプリングしたい

統計ソフトウェア R... is it enough?

`http://www.r-project.org/`



The development of linear models



簡単な GLMM なら R だけで推定可能

- R にはいろいろな GLMM の最尤推定関数が準備されている
- `library(glmmML)` の `glmmML()`
- `library(lme4)` の `lmer()`
- `library(nlme)` の `nlme()` (正規分布のみ)
- しかし もうちょっと複雑な GLMM , たとえば個体差 + 地域差をいれた統計モデルの最尤推定は かなり 難しい (ヘンな結果が得られたりする)
- 積分がたくさん入っている尤度関数の評価がしんどい

さまざまな MCMC アルゴリズム

いろいろな MCMC

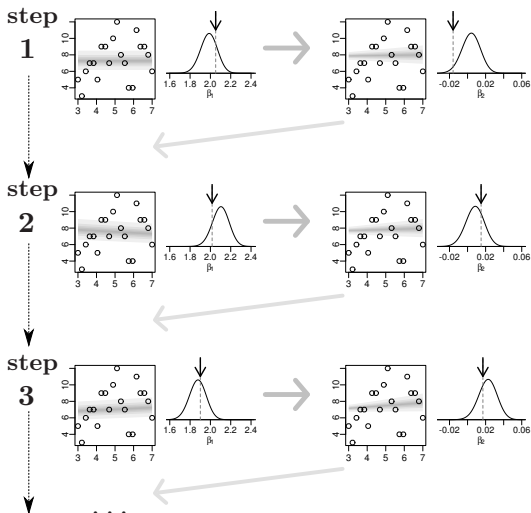
- **Metropolis Method**: 試行錯誤で値を変化させていく MCMC
 - Metropolis-Hastings: その改良版
- **Gibbs sampling**: 条件つき確率分布を使った MCMC
 - 複数の変数 (パラメーター・状態) を効率よくサンプリング
- **HMC sampling**: Stan で使われている
 - 久保はよくわかっていない

Gibbs sampling とは何か?

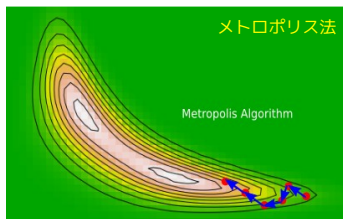
- MCMC アルゴリズムのひとつ
- 複数のパラメーターの MCMC サンプリングに使う
- 例: パラメーター β_1 と β_2 の Gibbs sampling
 - ① β_2 に何か適当な値を与える
 - ② β_2 の値はそのままにして, その条件のもとでの β_1 の MCMC sampling をする (条件つき事後分布)
 - ③ β_1 の値はそのままにして, その条件のもとでの β_2 の MCMC sampling をする (条件つき事後分布)
 - ④ 2. - 3. をくりかえす
- 教科書の第 9 章の例題で説明

図解: Gibbs sampling (統計モデリング入門の第9章)

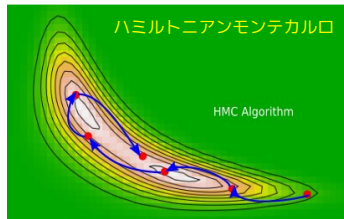
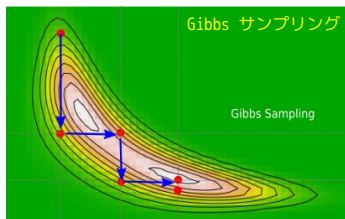
MCMC β_1 のサンプリング β_2 のサンプリング



MCMC sampling 法のあれこれ



いずれも
“いめーじ”
です...



松浦さんの本，ハミルトニアンモンテカルロ法
でベイズ統計モデルの事後分布を高速に推定

便利な “BUGS” 汎用 Gibbs sampler たち

- BUGS 言語 (+ っぽいもの) でベイズモデルを記述できるソフトウェア
 - **WinBUGS** — 歴史を変えて.....さようなら?
 - **JAGS** — お手軽で良い, どんな OS でも動く
 - **Stan** — いま一番の注目 (Hamiltonian MC)
— 今日は紹介しませんが

えーと.....BUGS 言語って何?

このベイズモデルを BUGS 言語で記述したい

データ $Y[i]$
種子数8個のうちの生存数

二項分布

$\text{dbin}(q, 8)$

生存確率 q

無情報事前分布

BUGS 言語コード

```
for (i in 1:N.sample) {
  Y[i] ~ dbin(q, 8)
}
q ~ dunif(0.0, 1.0)
```

矢印は手順ではなく、依存関係をあらわしている

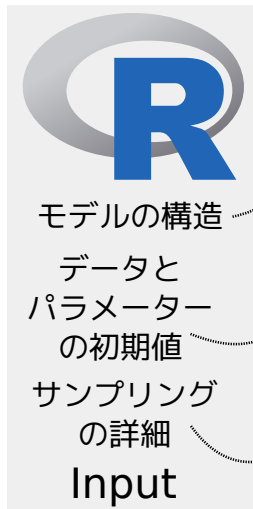
BUGS 言語: ベイズモデルを記述する言語

Spiegelhalter et al. 1995. BUGS: Bayesian Using Gibbs Sampling version 0.50.

いろいろな OS で使える JAGS 4.3.0

- R core team のひとり Martyn Plummer さんが開発
 - Just Another Gibbs Sampler
- C++ で実装されている
 - R がインストールされていることが必要
- Linux, Windows, Mac OS X バイナリ版もある
- 開発進行中
- R からの使う: `library(rjags)`

JAGS を R の “したうけ” として使う



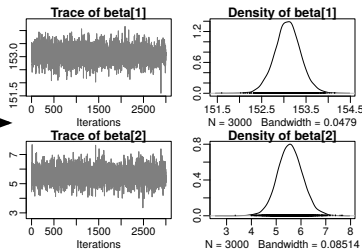
BUGS言語



JAGS

MCMC sampling from
posterior distributions

事後分布からの
ランダムサンプル



Output

R から JAGS にこんなかんじで仕事を命じる (1 / 3)

```
library(rjags)
library(R2WinBUGS) # to use write.model()

model.bugs <- function()
{
  for (i in 1:N.data) {
    Y[i] ~ dbin(q, 8) # 二項分布にしたがう
  }
  q ~ dunif(0.0, 1.0) # q の事前分布は一様分布
}

file.model <- "model.bug.txt"
write.model(model.bugs, file.model) # ファイル出力

# 次につづく.....
```


R から JAGS にこんなかんじで仕事を命じる (2 / 3)

```
load("mcmc.RData") # (data.RData ではなく mcmc.RData!!)
list.data <- list(Y = data, N.data = length(data))
inits <- list(q = 0.5)
n.burnin <- 1000
n.chain <- 3
n.thin <- 1
n.iter <- n.thin * 1000

model <- jags.model(
  file = file.model, data = list.data,
  inits = inits, n.chain = n.chain
)
```

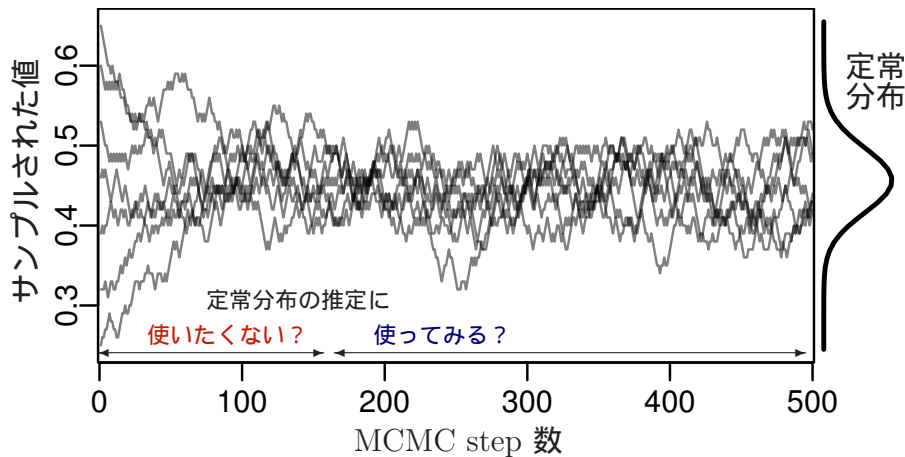
```
# まだ次につづく.....
```

R から JAGS にこんなかんじで仕事を命じる (3 / 3)

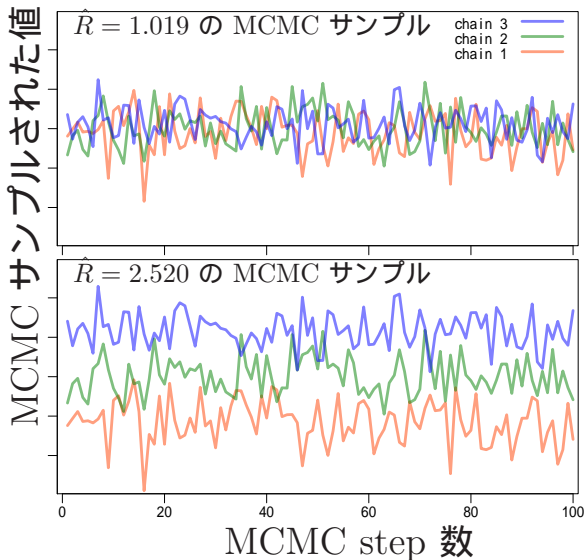
```
# burn-in
update(model, n.burnin) # burn in

# サンプリング結果を post.mcmc.list に格納
post.mcmc.list <- coda.samples(
  model = model,
  variable.names = names(inits),
  n.iter = n.iter,
  thin = n.thin
)
# おわり
```

burn in って何? → 「使いたくない」長さの指定



試行間で差がないかを収束診断 する



まあ、
いいかな.....

何やら
問題あり!

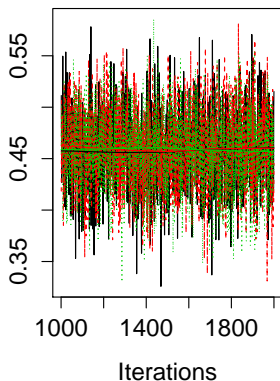
収束診断の \hat{R} (アールハット) 指数

- `gelman.diag(post.mcmc.list)` → 実演表示
- R-hat は Gelman-Rubin の収束判定用の指数
 - $\hat{R} = \sqrt{\frac{\hat{\text{var}}^+(\psi|y)}{W}}$
 - $\hat{\text{var}}^+(\psi|y) = \frac{n-1}{n}W + \frac{1}{n}B$
 - W : サンプル列内の variance の平均
 - B : サンプル列間の variance
 - Gelman et al. 2004. Bayesian Data Analysis. Chapman & Hall/CRC

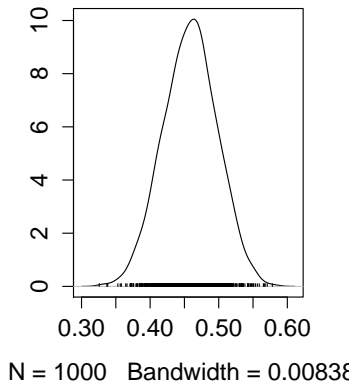
Gibbs sampling → 事後分布の推定

- `plot(post.mcmc.list)`

Trace of q



Density of q



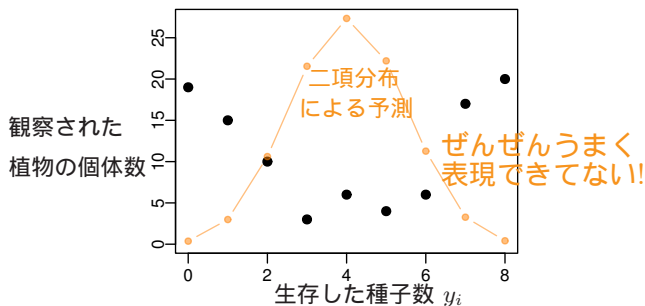
?? . GLMM と階層ベイズモデル

GLMM のベイズモデル化

階層ベイズモデルとなる

二項分布では説明できない観測データ!

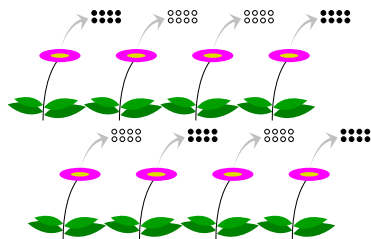
100 個体の植物の合計 800 種子中 **403 個** の生存が見られたので，平均生存確率は 0.50 と推定されたが.....



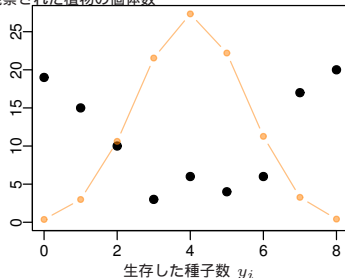
さっきの例題と同じようなデータなのに?
 (「統計モデリング入門」第 10 章の最初の例題)

個体差 → 過分散 (overdispersion)

極端な過分散の例



観察された植物の個体数



- 種子全体の平均生存確率は 0.5 ぐらいかもしれないが.....
- 植物個体ごとに種子の生存確率が異なる: 「個体差」
- 「個体差」があると overdispersion が生じる
- 「個体差」の原因は観測できない・観測されていない

モデリングやりなおし: 植物個体ごとに見えない “差” があると仮定

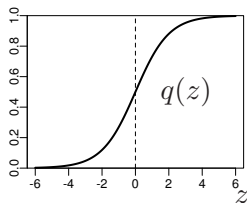
- 生存確率を推定するために **二項分布** という確率分布を使う
- 個体 i の N_i 種子中 y_i 個が生存する確率は二項分布

$$p(y_i | q_i) = \binom{N_i}{y_i} q_i^{y_i} (1 - q_i)^{N_i - y_i},$$

- ここで仮定していること
 - **個体差がある** ので個体ごとに生存確率 q_i が異なる

各個体の生存確率は logistic 回帰のモデルを使う

- 生存確率 $q_i = q(z_i)$ をロジスティック関数 $q(z) = 1/\{1 + \exp(-z)\}$ で表現



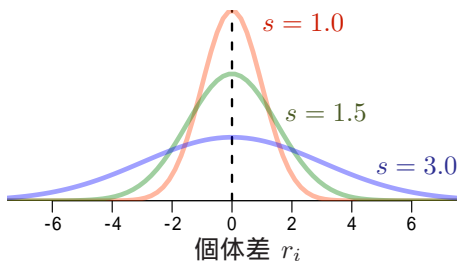
- 線形予測子 $z_i = a + r_i$ とする
 - パラメーター a : 全体の平均
 - パラメーター r_i : 個体 i の個体差 (ずれ)

個々の個体差 r_i を最尤推定 = データのよみあげ

パラメーター数 > サンプルサイズ

- 100 個体の生存確率を推定するためにパラメーター 101 個 (a と $\{r_1, r_2, \dots, r_{100}\}$) を推定すると.....
- 個体ごとに生存数 / 種子数を計算していることと同じ! (“データのよみあげ” と等価)

そこで，次のように考えてみる

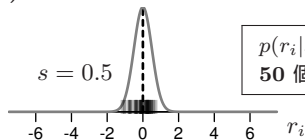
$\{r_i\}$ のばらつきは正規分布だと考えてみる

$$p(r_i | s) = \frac{1}{\sqrt{2\pi s^2}} \exp\left(-\frac{r_i^2}{2s^2}\right)$$

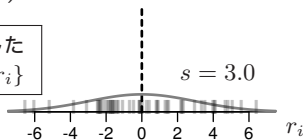
この確率密度 $p(r_i | s)$ は r_i の「出現しやすさ」をあらわしていると解釈すればよいでしょう。 r_i がゼロにちかい個体はわりと「ありがち」で、 r_i の絶対値が大きな個体は相対的に「あまりいない」。

ひとつの例示: 個体差 r_i の分布と 過分散 の関係

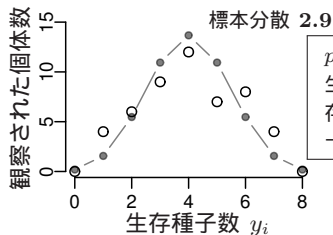
(A) 個体差のばらつきが小さい場合 (B) 個体差のばらつきが大きい場合



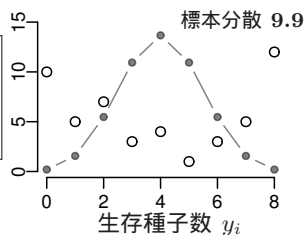
$p(r_i|s)$ が生成した
50 個体ぶんの $\{r_i\}$



確率 $q_i = \frac{1}{1 + \exp(-r_i)}$
の二項乱数を発生させる

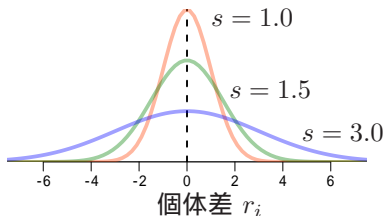


$p(y_i|q_i)$ が
生成した生
存種子数の
一例



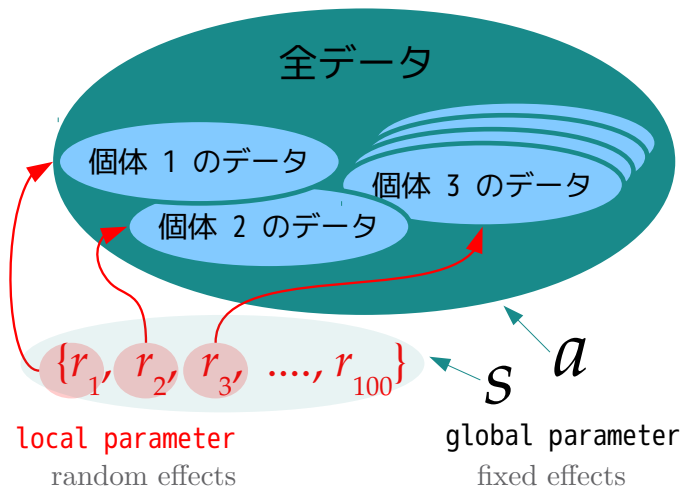
これは r_i の事前分布の指定, ということ

前回の講義で $\{r_i\}$ は正規分布にしたがうと仮定したが
ベイズ統計モデリングでは「100 個の r_i たちに
共通する事前分布として正規分布を指定した」
ということになる



$$p(r_i | s) = \frac{1}{\sqrt{2\pi s^2}} \exp\left(-\frac{r_i^2}{2s^2}\right)$$

統計モデルの大域的・局所的なパラメーター



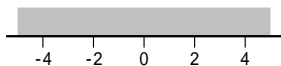
データのどの部分を説明しているのか?

パラメーターごとに適切な事前分布を選ぶ

無情報事前分布

a, s

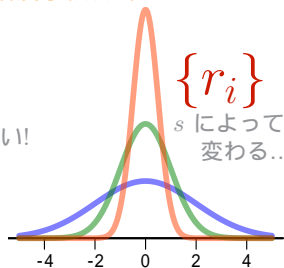
どんな値をとってもよい!



階層事前分布

$\{r_i\}$

s によって
変わる...



パラメーターの
種類

説明する範囲

事前分布

全体に共通する平均・ばらつき

大域的

無情報事前分布

個体・グループごとのずれ

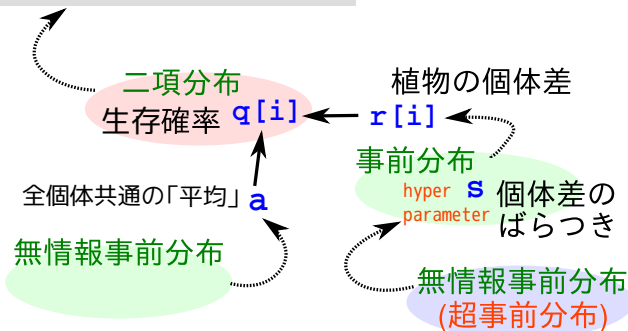
局所的

階層事前分布

階層ベイズモデル: Hierarchical and non-informative priors

超事前分布 → 事前分布という階層があるから

データ 種子8個のうち
 $y[i]$ が生存



矢印は手順ではなく、依存関係をあらわしている

?? 階層ベイズモデルの推定

ソフトウェア JAGS を試してみる

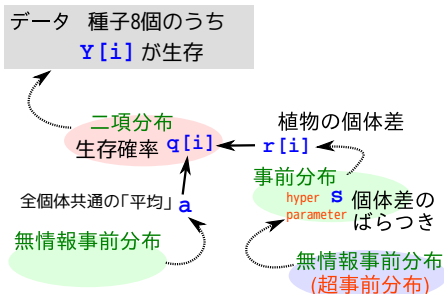
R の “したうけ” として JAGS を使う

階層ベイズモデルを BUGS コードで記述する

```

model
{
  for (i in 1:N.data) {
    Y[i] ~ dbin(q[i], 8)
    logit(q[i]) <- a + r[i]
  }
  a ~ dnorm(0, 1.0E-4)
  for (i in 1:N.data) {
    r[i] ~ dnorm(0, tau)
  }
  tau <- 1 / (s * s)
  s ~ dunif(0, 1.0E+4)
}

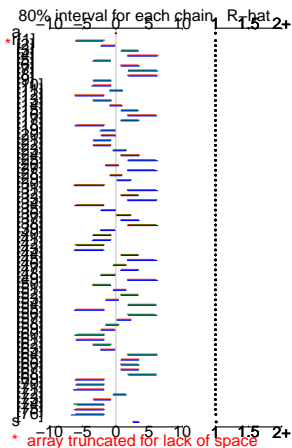
```



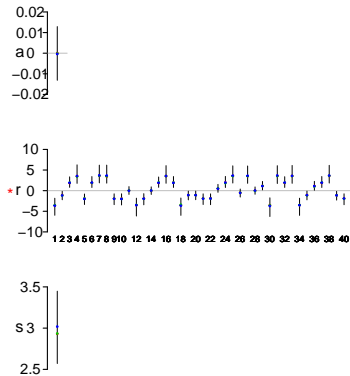
JAGS で得られた事後分布サンプルの要約

```
> source("mcmc.list2bugs.R") # なんとなく便利なので...
> post.bugs <- mcmc.list2bugs(post.mcmc.list) # bugs クラスに変換
```

3 chains, each with 4000 iterations (first 2000 discarded)

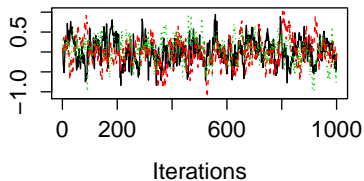


medians and 80% intervals

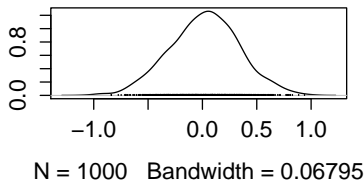


各パラメーターの事後分布サンプルを R で調べる

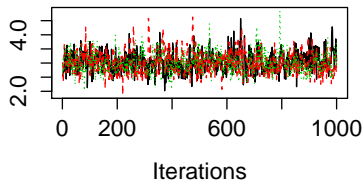
Trace of a



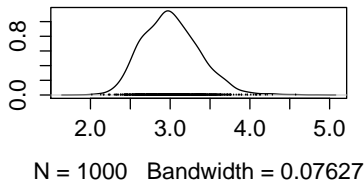
Density of a



Trace of s

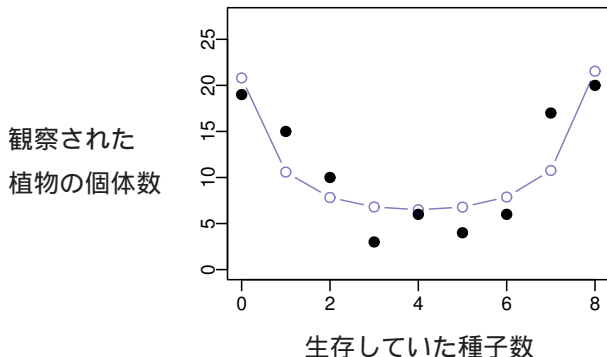


Density of s



得られた事後分布サンプルを組みあわせて予測

- `post.mcmc <- to.mcmc(post.bugs)`
- これは `matrix` と同じようにあつかえるので、作図に便利
-このあとごちゃごちゃと計算する必要あるけど、省略.....



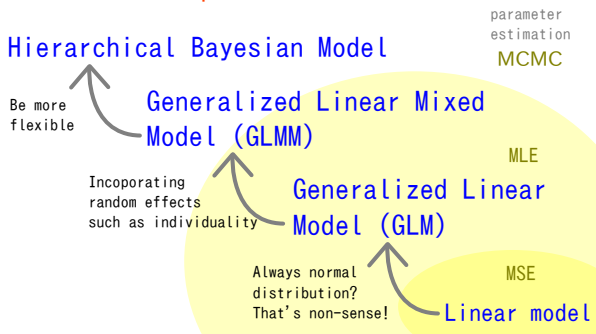
?? 階層ベイズモデル (HBM)? or GLMM?

Model: HBM and GLMM are equivalent

Estimation: NOT equivalent

一般化線形混合モデル

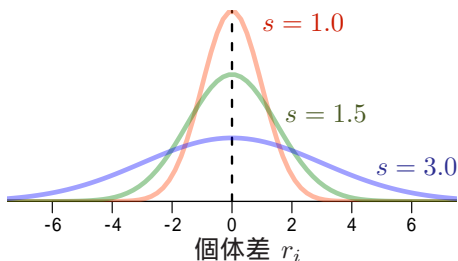
The development of linear models



Improve logistic regression model (GLM)!

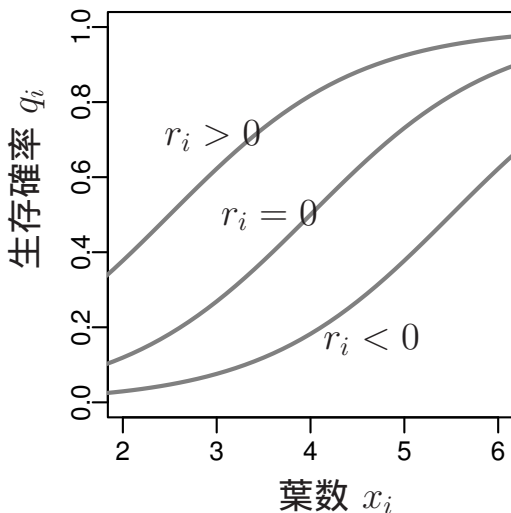
ロジスティック回帰のモデル

- 確率分布: 二項分布
- 線形予測子: $\beta_1 + \beta_2 x_i + r_i$
- リンク関数: logit リンク関数



1

 r_i^2

個体 i の個体差を r_i としてみよう

固定効果 と ランダム効果

Generalized Linear Mixed Model (GLMM)
で使う Mixed な 線形予測子: $\beta_1 + \beta_2 x_i + r_i$

- fixed effects: $\beta_1 + \beta_2 x_i$
- random effects: $+r_i$

fixed? random? よくわからん.....?

global parameter と local parameter

Generalized Linear Mixed Model (GLMM)
で使う Mixed な 線形予測子: $\beta_1 + \beta_2 x_i + r_i$

- fixed effects: $\beta_1 + \beta_2 x_i$
 - global parameter — for all individuals
- 全個体のばらつき s も global parameter
- random effects: $+r_i$
 - local parameter — only for individual i

maximum likelihood estimation of GLMM

データ $y_i \sim$ binomial distribution

$$p(y_i | \beta_1, \beta_2) = \binom{8}{y_i} q_i^{y_i} (1 - q_i)^{8 - y_i}$$

個体差 $r_i \sim$ Gaussian distribution

$$p(r_i | s) = \frac{1}{\sqrt{2\pi s^2}} \exp\left(-\frac{r_i^2}{2s^2}\right)$$

Integrate out $r_i!$

$$L_i = \int_{-\infty}^{\infty} p(y_i | \beta_1, \beta_2, r_i) p(r_i | s) dr_i$$

全データの尤度 — β_1, β_2, s の関数

$$L(\beta_1, \beta_2, s) = \prod_i L_i$$

- Model: HBM and GLMM are same
- Estimation: **NOT** same

Hierarchical Bayesian model (HBM)
is **better** because we can apply MCMC
estimation.

Maximum likelihood estimation (MLE)
is **NOT** easy!

次回予告

The next topic

階層ベイズモデルと時間変化モデル

Hierarchical Bayesian Model (HBM) & Time Change Model

The development of linear models

