

一般化線形 (混合) モデル (3)

MCMC と階層ベイズモデル

久保拓弥 kubo@ees.hokudai.ac.jp

大阪大学「データ科学特論 I」<http://goo.gl/RrHZeY>

2013-08-27

ファイル更新時刻: 2013-08-27 08:51

この時間のハナシ I

① 簡単な GLM の最尤推定

MCMC 説明の準備のための復習

② 同じ統計モデルを MCMC であつかう

Markov chain Monte Carlo method

③ GLMM は階層ベイズモデルの一種

だったら MCMC で事後分布が得られるよね

④ MCMC のためのソフトウェア

事後分布からサンプリングしたい

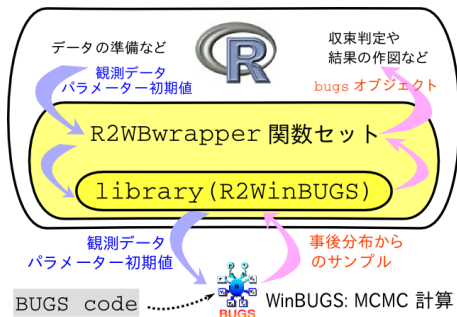
⑤ 階層ベイズモデルの推定

Gibbs sampling software を使ってみる

この時間のハナシ II

⑥ おわり

統計モデルを理解してデータ解析をする

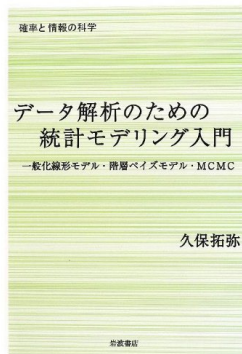


今日の内容と「統計モデリング入門」との対応

<http://goo.gl/Ufq2>

今日はおもに「第 8–10 章」の内容を説明します。

- 著者: 久保拓弥
- 出版社: 岩波書店
- 2012-05-18 刊行

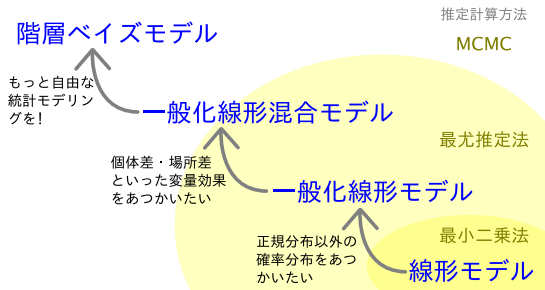


ねらい：統計モデルと推定方法をもっと発展させたい

より現実的・実戦的なデータ解析のために

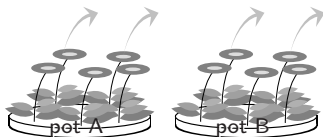
- 一般化線形モデル → 階層ベイズモデル
- 最尤推定 → Markov chain Monte Carlo (MCMC)

線形モデルの発展



なぜ？ 現実のデータ解析では推定がしんどいから

前の時間の最後に説明した問題



個体差も植木鉢差も
考慮して推定したい

$$\text{logit}q_i = \beta_1 + \beta_2 x_i + r_i + r_j$$

- この程度の簡単そうに見える統計モデルでもパラメーター推定 (数値的な最適化計算) はしんどい
 - 尤度方程式が重積分を含んでいるため
- 実際のデータ解析はもっとややこしくなりうる
 - 欠測値, 時系列構造, 空間構造.....などなど

あとづけメモ: このあたり, もう少しじっくり

この時間は

- 推定法を新しくする: 最尤推定 \rightarrow MCMC
- 統計モデルを新しくする: GLMM \rightarrow 階層
ベイズモデル

あとづけメモ: なぜ推定法を変えるのか?

- 統計モデルが複雑になると最尤推定法がしんどくなるから
- マルコフ連鎖モンテカルロ (MCMC) 法ならなんとかなりそう

あとづけメモ: なぜベイズモデル?

- MCMC 法で得られる結果は, 統計モデルがベイズ化されているほうが解釈しやすいような気がするから (これは久保の偏見かもしれない)
- GLMM はじつは階層ベイズモデルの一種なんので— モデルはベイズだけど最尤推定している— という解釈

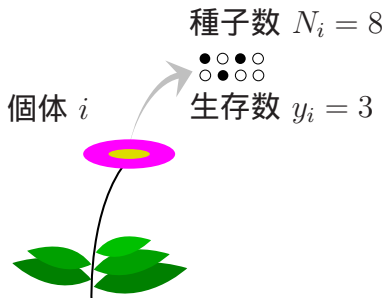
1. 簡単な GLM の最尤推定

MCMC 説明の準備のための復習

ロジスティック回帰

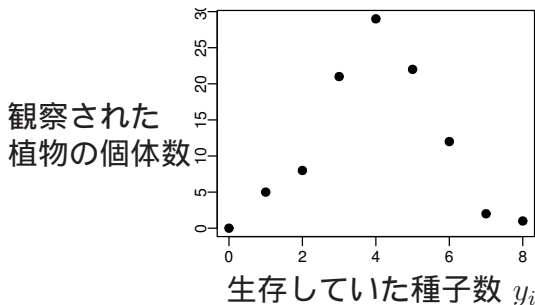
例題: 植物の種子の生存確率

- 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 100 個体, 合計 800 種子の生存の有無を調べた
- 問: この植物の生存確率はどのように統計モデル化できるか?



簡単すぎる例題: 生存確率は全個体で同じ (「個体差」なし)

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	0	5	8	21	29	22	12	2	1



No individual differences, no random effects

生存確率 q と二項分布の関係

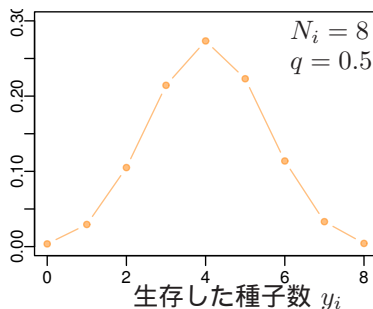
- 生存確率を推定するために**二項分布** という確率分布を使う
- 個体 i の N_i 種子中 y_i 個が生存する確率は二項分布

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

- ここで仮定していること
 - **個体差はない**
 - つまり **すべての個体で同じ生存確率 q**

二項分布で「 N_i 個中の y_i 個」型データをあつかう

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



ゆうど

尤度: 100 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 100 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

$$L(q|\{y_i\}) = \prod_{i=1}^{100} p(y_i | q)$$

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	0	5	8	21	29	22	12	2	1

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

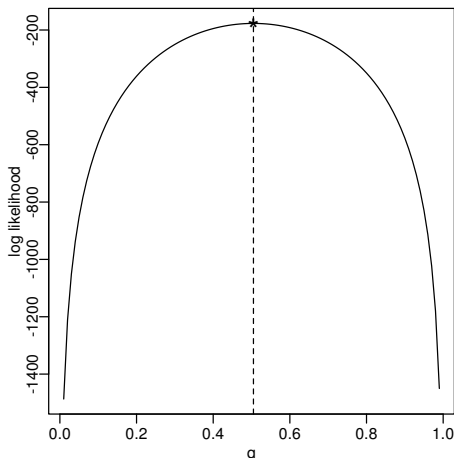
$$\begin{aligned}\log L(q \mid \text{データ}) &= \sum_{i=1}^{100} \log \binom{N_i}{y_i} \\ &+ \sum_{i=1}^{100} \{y_i \log(q) + (N_i - y_i) \log(1 - q)\}\end{aligned}$$

- 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) とは何か

- 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

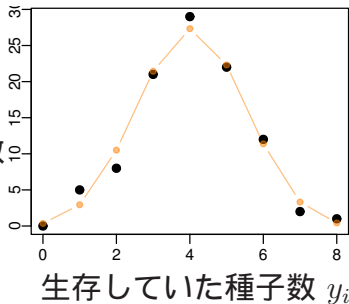
$$\hat{q} = \frac{\text{生存種子数}}{\text{調査種子数}} = \frac{404}{800} = 0.505$$



二項分布で説明された 8 種子中 y_i 個の生存

$$\hat{q} = 0.505 \text{ なので } \binom{8}{y} 0.505^y 0.495^{8-y}$$

観察された
植物の個体数



2. 同じ統計モデルを MCMC であつかう

Markov chain Monte Carlo method

最尤推定と MCMC はどのようにちがうのか?

ここでやること: 尤度と MCMC の関係を考える

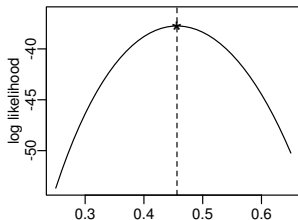
- さきほどの簡単な例題 (生存確率) のデータ解析を
- 最尤推定ではなく
- Markov chain Monte Carlo (MCMC) 法のひとつである **メトロポリス法** (Metropolis method) であつかう
- 得られる結果: 「パラメーターの値の分布」.....??

MCMC をもちださなくてもいい簡単すぎる問題
説明のためあえてメトロポリス法を適用してみる

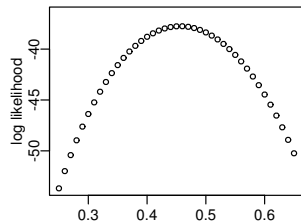
メトロポリス法を説明するための準備

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



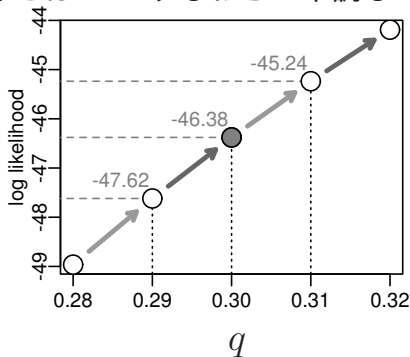
説明を簡単にするため

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

最尤推定 とは..... すなわち「単調な山のぼり」



- 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

メトロポリス法のルール: この例題の場合

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

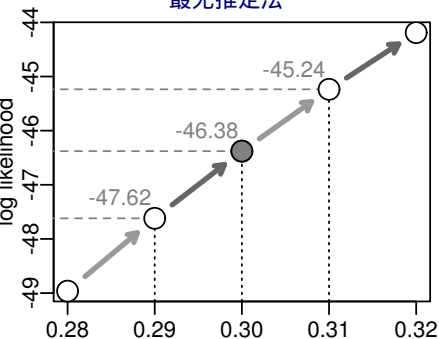
- $L(q_{\text{new}}) \geq L(q)$ (あてはまり改善): $q \leftarrow q_{\text{new}}$
- $L(q_{\text{new}}) < L(q)$ (あてはまり改悪):
 - 確率 $r = L(q_{\text{new}})/L(q)$ で $q \leftarrow q_{\text{new}}$
 - 確率 $1 - r$ で q を変更しない

④ 手順 2. にもどる

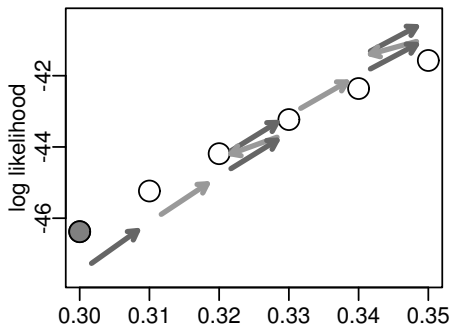
($q = 0.01$ や $q = 0.99$ でどうなるんだ, といった問題は省略)

メトロポリス法のルールで q を動かす

最尤推定法



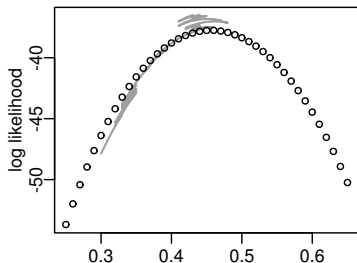
メトロポリス法 (MCMC)



メトロポリス法だと
「単調な山のぼり」
にはならない

対数尤度関数の「山」でうろうろする q の値

メトロポリス法 (そして一般の MCMC) は最適化ではない

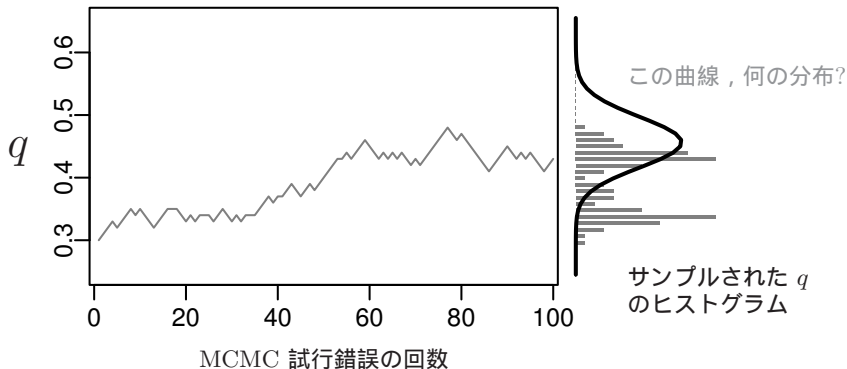


ときどきはでに落っこちる

何のためにこんなことをやるのか?

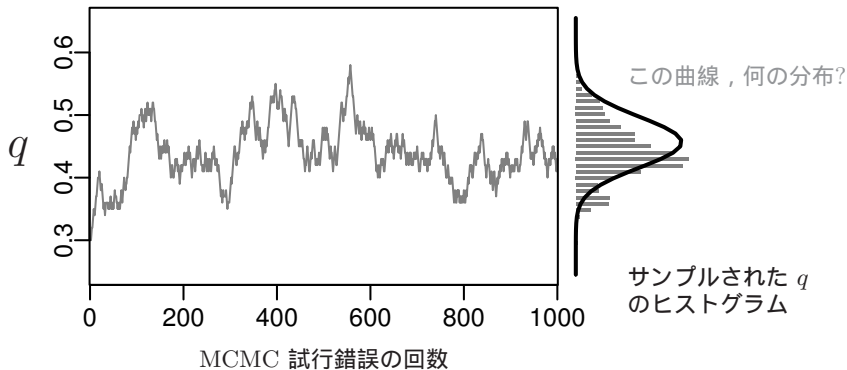
q の変化していく様子を記録してみよう

ステップごとに q の値をサンプリング



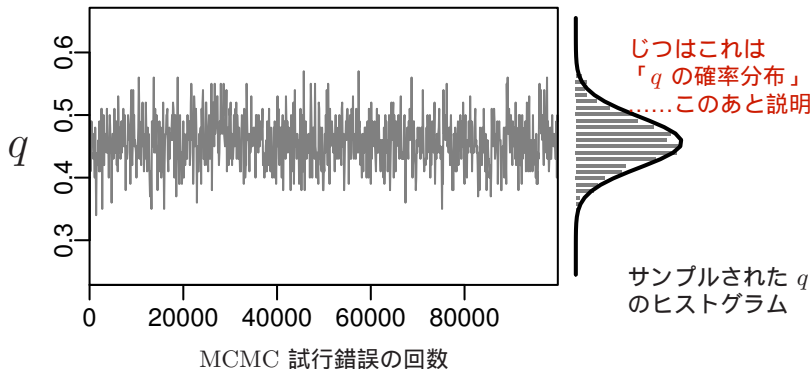
もっと試行錯誤してみたほうがいいのか?

もっと長くサンプリングしてみる



まだまだ.....?

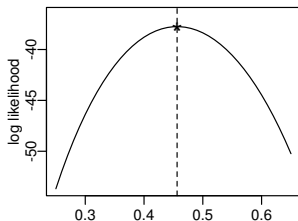
もっともっと長くサンプリングしてみる



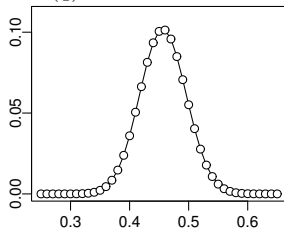
なんだか、ある「山」のかたちにとまってきたぞ？

MCMC は何をサンプリングしている？

既出の対数尤度 $\log L(q)$



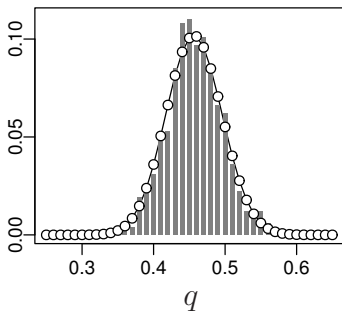
尤度 $L(q)$ に比例する確率分布



尤度に比例する確率分布からのランダムサンプル

マルコフ連鎖の定常分布は $p(q) = \frac{L(q)}{\sum_q L(q)}$ となる

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 サンプリングすると, $p(q)$ 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!

つはわれわれが知りたかったのは \hat{q} ではなく $p(q)$ だったのかも

MCMC という推定方法から
「パラメーター q の確率分布」
というちょっと奇妙な考えかたがでて
きた

「ふつう」の統計学では
「パラメーターの確率分布」といった
考えかたはしない, しかし

ベイズ統計学なら
「パラメーターの確率分布」はぜんぜん
自然な考えかただ

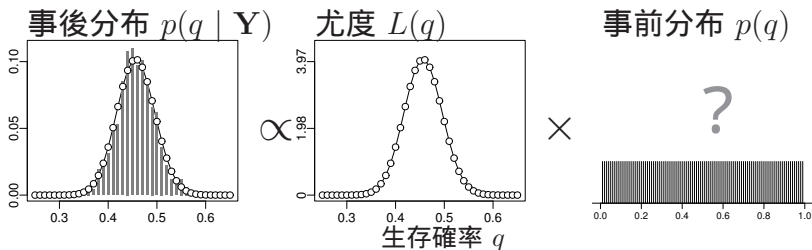
ベイズモデル: 尤度・事後分布・事前分布.....

- ベイズの公式 $p(q | \mathbf{Y}) = \frac{p(\mathbf{Y} | q) \times p(q)}{p(\mathbf{Y})}$
- $p(q | \mathbf{Y})$ は何かデータ (\mathbf{Y}) のもとで何かパラメーター (q) が得られる確率 (事後分布)
- $p(q)$ はあるパラメーター q が得られる確率 (事前分布)
- $p(\mathbf{Y} | q)$ パラメーターを決めたときにデータが得られる確率 (尤度に比例)
- $p(\mathbf{Y})$ はデータ \mathbf{Y} が得られる確率 (単なる規格化定数)

$$\begin{aligned}
 (\text{事後分布}) &\propto \frac{\text{尤度} \times \text{事前分布}}{(\text{データが得られる確率})} \\
 &\propto \text{尤度} \times \text{事前分布}
 \end{aligned}$$

ベイズ統計にむりやりこじつけてみると？

q の事前分布は一様分布，と考えるとつじつまがあう？



事前分布 っるのがよくわからない.....(あとで検討)

メトロポリスのアルゴリズム中の「 q の増減はランダム」に関連していないこともない (逃げ口上)

以上の説明は、「MCMC でパラメーター推定によって得られる結果はベイズ統計モデルのそれとして解釈しやすいかも」といったことを**ばくぜん**かつ**なんとなく**対応づけるひとつのこころみでありました.....

厳密な正当化とかそういったものではありません

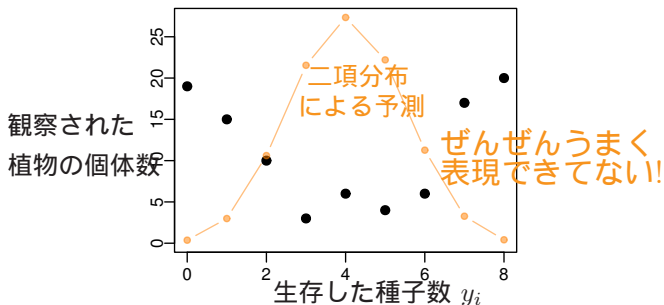
3. GLMM は階層ベイズモデルの一種

だったら MCMC で事後分布が得られるよね

事前分布をどう設定しよう？

また別の、二項分布では説明できない観測データ

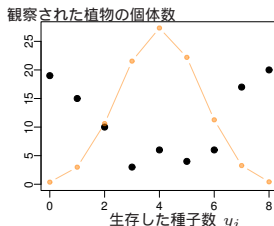
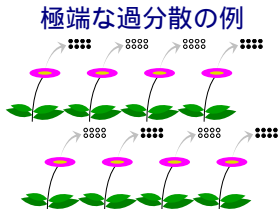
100 個体の植物の合計 800 種子中 **403 個** の生存が見られたので、
平均生存確率は 0.50 と推定されたが.....



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)



- 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

GLMM は fixed + random effects を考慮

線形モデルの伝統: 「個体ごとに異なる何かに由来する効果」を fixed/random effects にわけて統計モデル化する

- fixed effects 的な効果: 観測者がわざわざ設定・測定した要因 (実験処理, 植物のサイズなど), logit 変換された世界において生存確率の「効果の大きさ」を変える
 - この例題では fixed effects 的な要因なし
- random effects 的な効果: fixed effects 的ではない要因 (観測対象個体に関連する, 人間が設定・測定していないすべて)
 - logit 変換された世界において生存確率の「効果の大きさ」を変えずにばらつきだけを変えようとする

モデリングやりなおし: まず二項分布の再検討

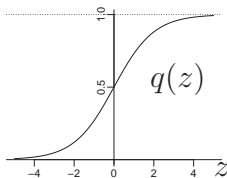
- 生存確率を推定するために **二項分布** という確率分布を使う
- 個体 i の N_i 種子中 y_i 個が生存する確率は二項分布

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

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

ロジスティック関数で表現する生存確率

- そこで生存する確率 $q_i = q(z_i)$ をロジスティック (logistic) 関数 $q(z) = 1/\{1 + \exp(-z)\}$ で表現 (logit link function)



- 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 を最尤推定するのはまずい

- 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」と同じ)
- こう仮定すると問題がうまくあつかえないだろうか?
 - 個体間の生存確率はばらつくけど, そんなにすごく異なる?
 - 観測データを使って, 「個体差」にみられるパターンを抽出したい (統計モデル化)

さて、
ここで統計モデル作りの手順
をみなおしてみましよう

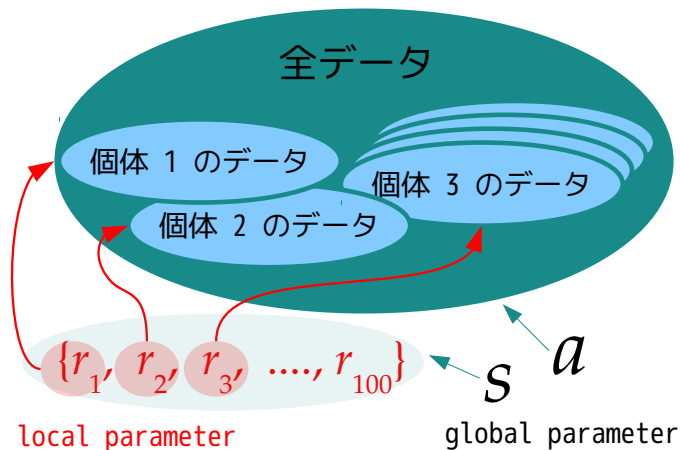
一般化線形モデル (GLM)

- 確率分布を決める: 二項分布 $y_i \sim \text{Binom}(8, q_i)$
- リンク関数を決める: $\text{logit}(q_i) = (\text{線形予測子 } z_i)$
- 線形予測子を決める: $z_i = a$

一般化線形混合モデル (GLMM)

- 確率分布を決める: 二項分布 $y_i \sim \text{Binom}(8, q_i)$
- リンク関数を決める: $\text{logit}(q_i) = (\text{線形予測子 } z_i)$
- 線形予測子を決める: $z_i = a + r_i$
 - a は global parameter, r_i は random effects の local parameter
- r_i の確率分布を決める (平均ゼロの正規分布とか)

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？

ベイズな一般化線形混合モデル (Bayesian GLMM)

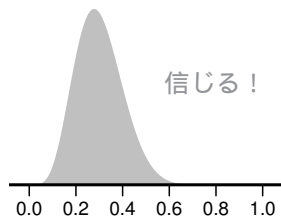
- 確率分布を決める: 二項分布 $y_i \sim \text{Binom}(8, q_i)$
- リンク関数を決める: $\text{logit}(q_i) = (\text{線形予測子 } z_i)$
- 線形予測子を決める: $z_i = a + r_i$
 - a は global parameter, r_i は random effects の local parameter
- r_i の事前分布を決める (平均ゼロの正規分布?)
- a の事前分布を決める (どうすればいいの?)

事前分布についてちゃんと検討してみよう

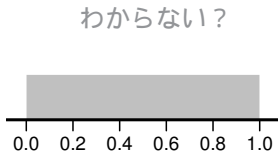
ベイズ統計モデルでよく使われる三種類の事前分布

たいていのベイズ統計モデルでは、ひとつのモデルの中で複数の種類の事前分布を混ぜて使用する。

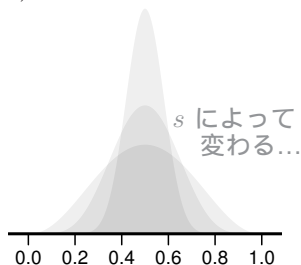
(A) 主観的な事前分布



(B) 無情報事前分布

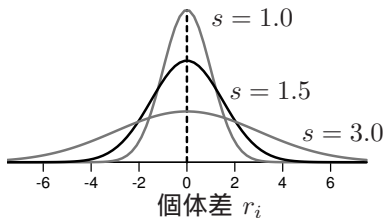


(C) 階層事前分布



r_i の事前分布は階層事前分布で良さそう

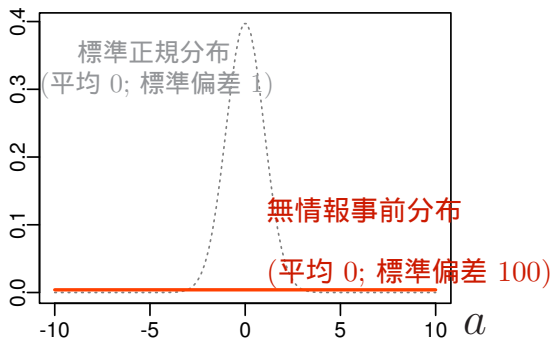
前回の授業で $\{r_i\}$ は正規分布にしたがうと仮定したが
ベイズ統計モデリングでは「100 個の r_i たちに
共通する事前分布として正規分布を指定した」
ということになる



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

個体差 $\{r_i\}$ のばらつき s の無情報事前分布

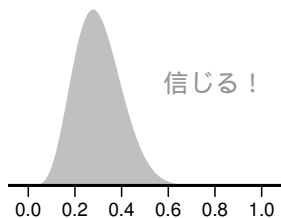
- s はどのような値をとってもかまわない
- そこで s の事前分布は **無情報事前分布** (non-informative prior) とする
- たとえば一様分布
 - とりあえず、ここでは $0 < s < 10^4$ の一様分布としてみる

全個体の「切片」 a の無情報事前分布

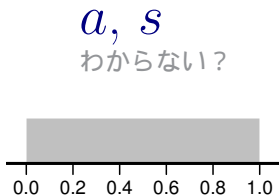
「生存確率の (logit) 平均 a は何でもよい」と表現している

パラメーターごとに適切な事前分布を選ぶ

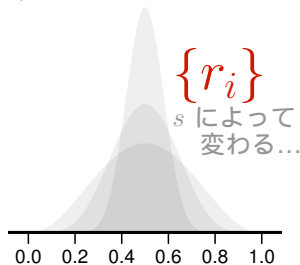
(A) 主観的な事前分布



(B) 無情報事前分布



(C) 階層事前分布



パラメーターの種類	説明する範囲	同じようなパラメーターの個数	事前分布
全体に共通する平均・ばらつき	大域的	少数	無情報事前分布
個体・グループごとのずれ	局所的	多数	階層事前分布

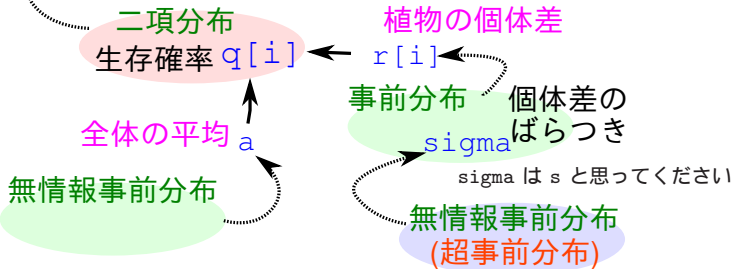
GLMM は階層ベイズモデル — なぜ?

超事前分布 → 事前分布という階層があるから

データ

8 個中の $Y[i]$ 個の種子が生存

σ は
hyper parameter



矢印は手順ではなく, 依存関係をあらわしている

4. MCMC のためのソフトウェア

事後分布からサンプリングしたい

Gibbs sampling

階層ベイズモデリング，その手順のまとめ

- 観測データを説明できそうな確率分布を選ぶ
- その確率分布の平均・分散などのモデリング
- パラメーターの**事前分布**を設定する
 - 階層的な事前分布 — 個体差・場所差など
 - 無情報事前分布 — いわゆる「処理の効果」など
- モデリングできたら，**事後分布**を推定する
 - 例: MCMC 計算によって事後分布からのサンプルを得る
- 事後分布を解釈する

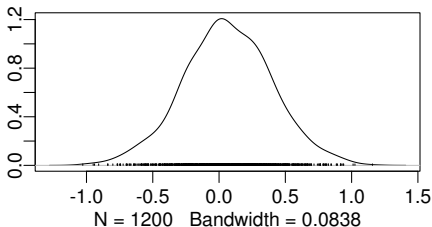
MCMC による事後分布からのサンプリング

- Markov Chain Monte Carlo : 単純な乱数を うまく つかって「あつかいづらい」確率分布からランダムサンプルを得る方法 (アルゴリズム)
- ある種のデータを解析するためには階層ベイズモデルが必要
- そういったベイズモデルを観測データに「あてはめ」てパラメータ推定するためには MCMC が役にたつ , ということにしたい (MCMC 利用法のひとつ)

「事後分布からのサンプル」って何の役にたつの？ (再)

```
> post.mcmc[, "a"] # 事後分布からのサンプルを表示
[1] -0.7592 -0.7689 -0.9008 -1.0160 -0.8439 -1.0380 -0.8561 -0.9837
[9] -0.8043 -0.8956 -0.9243 -0.9861 -0.7943 -0.8194 -0.9006 -0.9513
[17] -0.7565 -1.1120 -1.0430 -1.1730 -0.6926 -0.8742 -0.8228 -1.0440
... (以下略) ...
```

これらのサンプルの平均値・中央値・95% 区間を
調べることで事後分布の概要がわかる



どのようなソフトウェアで MCMC サンプリング?

① 自作プログラム

- 利点: 問題にあわせて自由に設計できる
- 欠点: 階層ベイズモデル用のプログラミング, けっこうめんどう

② R のベイズな package

- 利点: 空間ベイズ統計など便利な専用 package がある
- 欠点: 汎用性, とぼしい

③ 「できあい」の Gibbs sampler ソフトウェア

- 利点: 幅ひろい問題に適用できて, 便利
- 欠点: 「まちがいさがし」 (debug) がめんどう

R だけで何とかなる? — 最適化はしんどい

GLMM は階層ベイズモデルの一部

- R にはいろいろな GLMM 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)
- しかしながらもうちょっと複雑な GLMM では.....**無理**
(推定計算がうまくいかない)
 - `library(MCMCglmm)` を使いこなせば対処できるけど

5. 階層ベイズモデルの推定

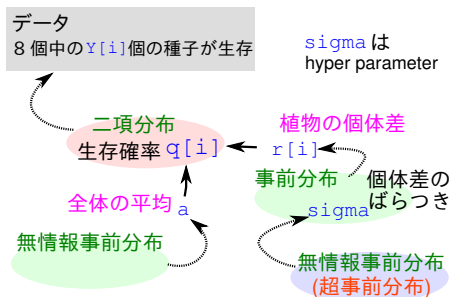
Gibbs sampling software を試してみる

かなりはしょった説明です すみません!

ここまでの用語の整理

- 階層ベイズモデル

$$(\text{事後分布}) \propto (\text{尤度}) \times (\text{事前分布}) \times (\text{超事前分布})$$



- 事後分布の推定計算方法: **Markov Chain Monte Carlo (MCMC) 法**

この例題の事後分布は？

$$p(a, \{r_i\}, s \mid \text{データ}) = \frac{\prod_{i=1}^{100} p(y_i \mid q(a + r_i)) p(a) p(r_i \mid s) p(s)}{\iint \cdots \int (\text{分子} \uparrow \text{そのまま}) dr_i ds da}$$

分母は何か**定数**になるので

$$p(a, \{r_i\}, s \mid \text{データ}) \propto \prod_{i=1}^{100} p(y_i \mid q(a + r_i)) p(a) p(r_i \mid s) p(s)$$

いろいろな MCMC の手段

いろいろな MCMC

- **メトロポリス法**: 試行錯誤で値を変化させていく MCMC
 - メトロポリス・ヘイスティングス法: その改良版
- **ギブス・サンプリング**: 条件つき確率分布を使った MCMC
 - 普通は複数の変数 (パラメーター・状態) のサンプリングのためにもちいる
- ここからあとで登場する MCMC はギブス・サンプリングと
考えてください

この事後分布から Gibbs sampling してみる

サンプリングの対象とするパラメーター以外は値を固定する

$$p(a \mid \cdots) \propto \prod_{i=1}^{100} p(y_i \mid q(a + r_i)) p(a)$$

$$p(s \mid \cdots) \propto \prod_{i=1}^{100} p(r_i \mid s) p(s)$$

$$p(r_1 \mid \cdots) \propto p(y_1 \mid q(a + r_1)) p(r_1 \mid s)$$

$$p(r_2 \mid \cdots) \propto p(y_2 \mid q(a + r_2)) p(r_2 \mid s)$$

⋮

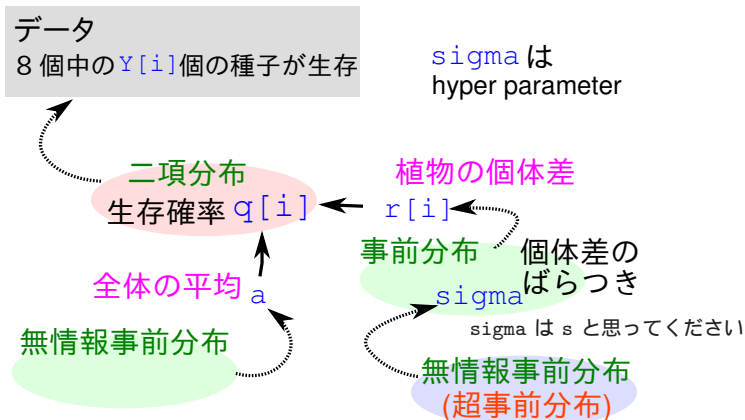
$$p(r_{100} \mid \cdots) \propto p(y_{100} \mid q(a + r_{100})) p(r_{100} \mid s)$$

便利な “BUGS” 汎用 Gibbs sampler たち

- BUGS 言語でベイズモデルを記述できるソフトウェア
 - WinBUGS — 評: とりあえずよく使われてます
 - OpenBUGS — 評: 予算が足りなくて停滞
 - JAGS — 評: じりじりと発展中, がんばってください
 - Stan — 評: 期待の新鋭
- リンク集: <http://hosho.ees.hokudai.ac.jp/~kubo/ce/BayesianMcmc.html>

BUGS 言語って何?

この階層ベイズモデルを BUGS 言語で記述したい



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.

```
model { # BUGS コードで定義された階層ベイズモデルの例
  for (i in 1:N.sample) {
    Y[i] ~ dbin(q[i], N[i])
    logit(q[i]) <- a + r[i]
  }
  a ~ dnorm(0, 1.0E-4)
  for (i in 1:N.sample) {
    r[i] ~ dnorm(0, tau)
  }
  tau <- 1 / (s * s)
  s ~ dunif(0, 1.0E+4)
}
```

なんとなく使われ続けている WinBUGS 1.4.3

- おそらく世界でもっともよく使われている Gibbs sampler
- **BUGS** 言語の実装
- 2004-09-13 に最新版 (ここで開発停止 → OpenBUGS)
- ソースなど非公開, 無料, ユーザー登録**不要**
- Windows バイナリーとして配布されている
 - Linux 上では WINE 上で動作
 - MacOS X 上でも Darwine など駆使すると動くらしい
- ヘンな GUI (Linux ユーザーの偏見)
- R ユーザーにとっては R2WinBUGS が快適 (後述)

WinBUGS は Gibbs sampling しているのか?

よくある質問: WinBUGS は Gibbs sampling してるの?

- 事前分布・尤度の組み合わせによって、サンプリング方法を自動的に変更している
 - 共役事前分布がない場合は、さまざまな数値的な方法を使う
- ユーザーはそのあたりをまったく指定する必要なし (指定できない)

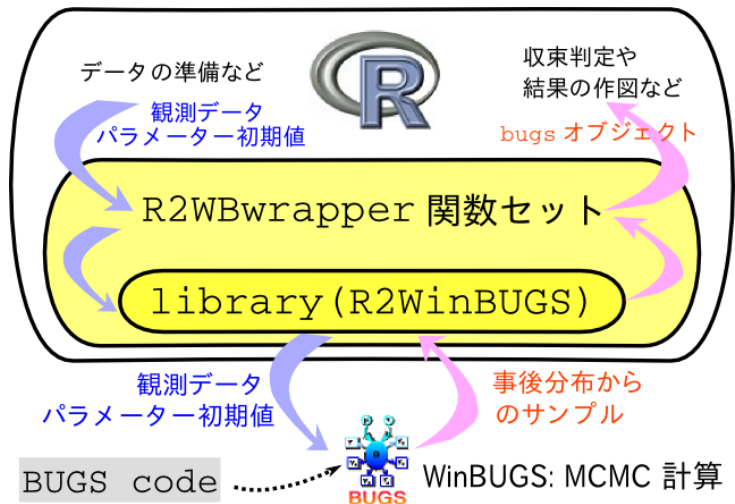
くわしくは WinBUGS のマニュアル読みましょう

<http://www.google.com/search?q=winbugs+user+manual>

今回説明する WinBUGS の使いかた (概要)

- WinBUGS を R から使う
 - R から WinBUGS をよびだし「このベイズモデルのパラメータの事後分布をこういうふうに MCMC 計算してね」と指示する
 - WinBUGS が得た事後分布からのサンプルセットを R がうけとる
- R の中では `library(R2WinBUGS)` package を使う
`R2WBwrapper` 関数 (久保作) を使う

概要: R2WBwrapper 経由で WinBUGS を使う



なんで WinBUGS を R 経由で使うの？

- WinBUGS のユーザーインターフェイスを使うのがめんど
うだから
- どうせ解析に使うデータは R で準備するから
- どうせ得られた出力は R で解析・作図するから
- R には R2WinBUGS という (機能拡張用) package があって, R
から WinBUGS を使うしくみが準備されてるから
 - R 上で `install.packages("R2WinBUGS")` でインストールで
きる

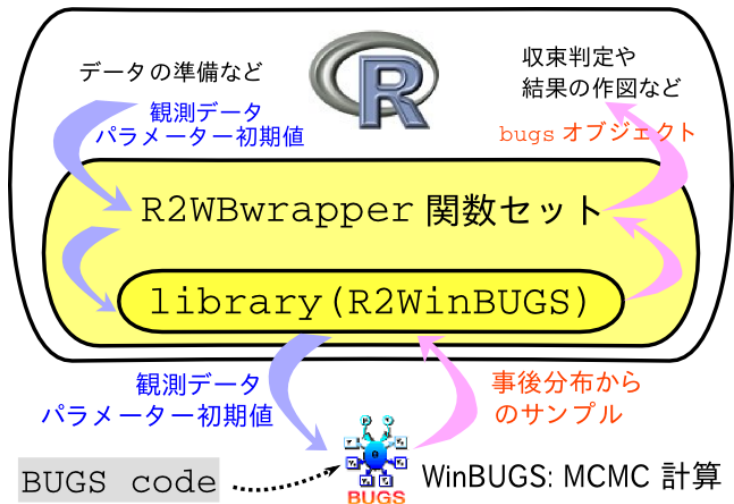
なんで R2WinBUGS をラップして使うの？

- R2WinBUGS 直接利用がめんどろだから
 - モデルをちょっと変更したらあちこち書きなおさないといけない
 - R2WBwrapper を使うとそのあたりがかなりマシになる
- Linux と Windows で「呼びだし」方法がびみょーに異なるため
 - R2WBwrapper を使うと自動的に OS にあわせた WinBUGS よびだしをする

R2WBwrapper 経由で WinBUGS を使う

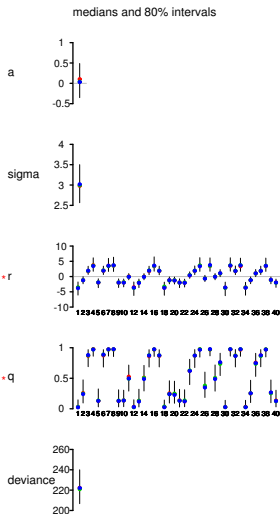
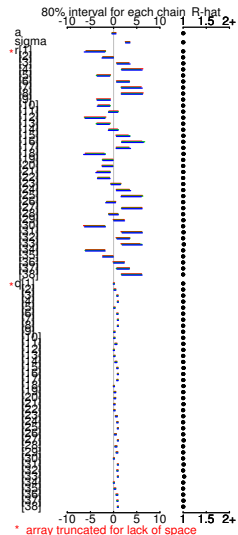
- ① BUGS 言語でかかれた model ファイルを準備する
- ② R2WBwrapper 関数を使う R コードを書く
- ③ R 上で 2. を実行
- ④ 出力された結果が bugs オブジェクトで返される
- ⑤ これを `plot()` したり `summary()` したり.....オブジェクトに変換して, いろいろ事後分布の図なんかを描いてみたり.....

(時間があればこの過程を実演します)



WinBUGS で得られた事後分布サンプルの要約

/kubo/public_html/stat/2010/ism/winbugs/model.bug.txt*, fit using WinBUGS, 3 chains, each with 1300 iteration

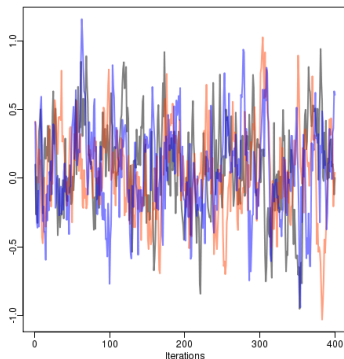
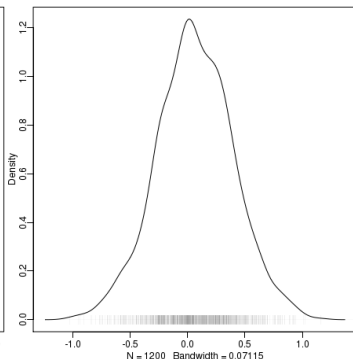


bugs オブジェクトの post.bugs を調べる

- `print(post.bugs, digits.summary = 3)`
- 事後分布の 95% 信頼区間などが表示される

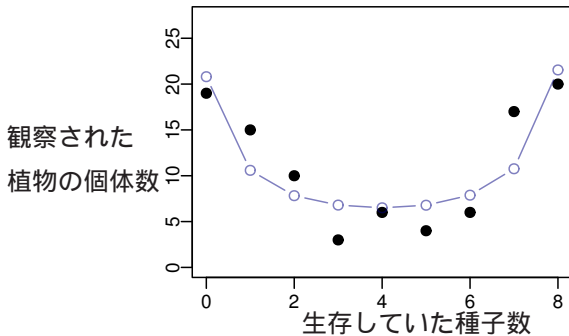
	mean	sd	2.5%	25%	50%	75%	97.5%	Rhat	n.eff
a	0.031	0.357	-0.718	-0.187	0.041	0.268	0.682	1.034	72
sigma	3.060	0.376	2.365	2.807	3.029	3.288	3.830	1.002	1200
r[1]	-3.890	1.903	-8.238	-4.918	-3.514	-2.546	-1.174	1.001	1200
r[2]	-1.190	0.905	-3.137	-1.763	-1.159	-0.559	0.438	1.007	290
r[3]	2.062	1.128	0.185	1.296	1.931	2.730	4.611	1.002	1200
r[4]	3.985	1.860	1.058	2.635	3.745	5.105	8.520	1.021	130
r[5]	-2.049	1.077	-4.458	-2.679	-1.971	-1.276	-0.255	1.008	270
r[6]	1.995	1.061	0.137	1.266	1.922	2.629	4.300	1.002	900
r[7]	3.886	1.765	1.144	2.664	3.583	4.894	8.223	1.008	320
r[8]	3.862	1.763	1.142	2.590	3.591	4.814	7.993	1.011	330
r[9]	-2.093	1.136	-4.532	-2.788	-1.978	-1.313	-0.130	1.003	540
r[10]	-1.993	1.082	-4.358	-2.631	-1.905	-1.250	-0.158	1.000	1200
r[11]	-0.049	0.786	-1.654	-0.555	-0.032	0.466	1.462	1.006	320
r[12]	-3.849	1.788	-8.204	-4.874	-3.547	-2.598	-1.144	1.001	1200
r[13]	-2.005	1.115	-4.593	-2.640	-1.908	-1.254	-0.069	1.001	1200

各パラメーターの事後分布サンプルを R で調べる

 a のサンプリングの様子 a の事後確率密度の推定

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



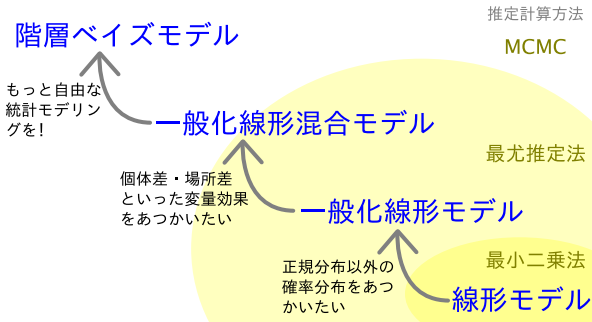
時間があれば個体差+場所差の例を紹介

6. おわり

統計モデルを理解してデータ解析をする

ここでこの統計モデリング 授業は終了です

線形モデルの発展



- データ解析の背後には統計モデルがある
- 統計モデルを理解して使おう
- データにあわせて統計モデルを設計する