Rで学ぶベイズ理論

１１．４変化点もでる
１１．５頑健な回帰店モデル
１１．６キャリア軌跡を推定

茨城大学工学部情報工学科
倉持　辰洋

はじめに

- これらの章ではWinBUGSを実際に使用していく
  サンプルとするモデルは以下のモデルとする
  - 変化点モデル
  - 頑健な回帰モデル
  - キャリア奇跡を推定

変化点モデル（１）

- Carlin et al.(1992)で取り上げられているイギリスでの鉱山災害の頻度分析を取り上げる。データは1851年から1962年までのものである。
  - y_t：t年における災害数
    tは実際の年号から１８５０を引いた値
  -19世紀末に事故件数が減少しているよう
- ある年より前、t<τの時ポアソン分布に従うとする

変化点モデル（2）

- 平均の対数を次のようにする

$$\log \mu_t = \beta_0 \ \ (t < \tau), \ \ \log \mu_t = \beta_0 + \beta_1 \ \ (t \geq \tau)$$

これを $y_t \sim Pisson(\mu_t), \ \ \log(\mu_i) = \beta_0 + \beta_1 \times \delta(t - \tau)$

と表現する。δ()は引数が非負値なら1、そうでないなら0

- 未知パラメータは

回帰パラメータ: β_0, β_1

変化パラメータ: τ

- β_1、β_2に曖昧一様分布、τに区間（1,N)上の一様分布を割り当てる（Nは年号の数）

WinBUGSで変化点モデルを定義（1）

- 最初にBUGS言語でモデルを定義した短いスクリプトを記述する。まずそれぞれを以下のように定義する。
  - y_t ⇒ D[year]
  - 平均 ⇒ mu[year]
  - β_0, β_1⇒ b[1], b[2]
  - τ ⇒ changeyear
- D[year]は平均mu[year]のポアソン分布に従う

  ⇒ D[year] ~ dpois(mu[year])
- β_jに平均が0、精度が0.000001に等しい正規分布を割り当て

  ⇒ b[j] ~ dnorm(0.0 , 1.0E - 6)

WinBUGSで変化点モデルを定義（2）

- τが区間(1,N)で連続した一様分布である

  ⇒ changeyear ~ dunif(1, N)

- log(μ_i)の記述

  ⇒ $\log(mu[year]) <- b[1] + stap(year - cangeyear) * b[2]$

  “<-”はオブジェクトへの代入記号

  step()はδ（）関数に等しい

WinBUGSで変化点モデルを定義（3）

- これらのモデル記述を次のようにしテキストファイルcoalmining.bugに保存する

```
model
{
	for(year in 1 : N )
	{
		D[year] ~ dpois(mu[year])
		log(mu[year]) <- b[1] + step(year - changeyear) * b[2]
	}
	for (j in 1:2)
	{
		b[j] ~ dnorm(0.0, 1.0E-6)
	}
	changeyear ~ dunif(1, N)
}
```

データの入力

- データはRのコンソールに直接入力する
- 定数Nは年号の数、Dは観測頻度のベクトル。

```
> N <- 112
> D <- c(4,5,4,1,0,4,3,4,0,6,
+ 3,3,4,0,2,6,3,3,5,4,5,3,1,4,4,1,5,5,3,4,2,5,2,2,3,4,2,1,3,2,
+ 1,1,1,1,1,3,0,0,1,0,1,1,0,0,3,1,0,3,2,2,
+ 0,1,1,1,0,1,0,1,0,0,0,2,1,0,0,0,1,1,0,2,
+ 2,3,1,1,2,1,1,1,1,2,4,2,0,0,0,1,4,0,0,0,
+ 1,0,0,0,0,0,1,0,0,1,0,0)
> data <- list("N", "D")
```

- パラメータτと回帰係数のベクトルβのシミュレーション標本をモニタすることを表現

```
> parameters <- c("changeyear", "b")
```

- パラメータ(β1,β2)の初期値(0,0)、τの初期値50

```
> inits <- function() {list(b = c(0,0), changeyear = 50) }
```

WinBUGSの実行

- bugs()関数を使いWinBUGSを実行する

```
> coalmining.sim <- bugs (data, inits, parameters,
+ "coalmining.bug", n.chains = 3, n.iter = 1000, codaPkg = TRUE)
```

- codaPkg=TRUEを指定することによりbugs()関数の返り値がWinBUGSの出力ファイル名となる。これをcodaパッケージで利用する
- WinBUGSの出力ファイルよりMCMC法オブジェクトを作成する

```
> coalmining.coda <- read.bugs(coalmining.sim)
```

WinBUGSの実行（おまけ1）

- codaPkgオプションを含めないとbugs関数の出力はシミュレーション結果となる。print()関数とplot()関数を利用し要約とグラフを作成してみる。

```
> print( bugs(data,inits,parameters,"coalmining.bug",n.chains=3,n.iter=1000))
```

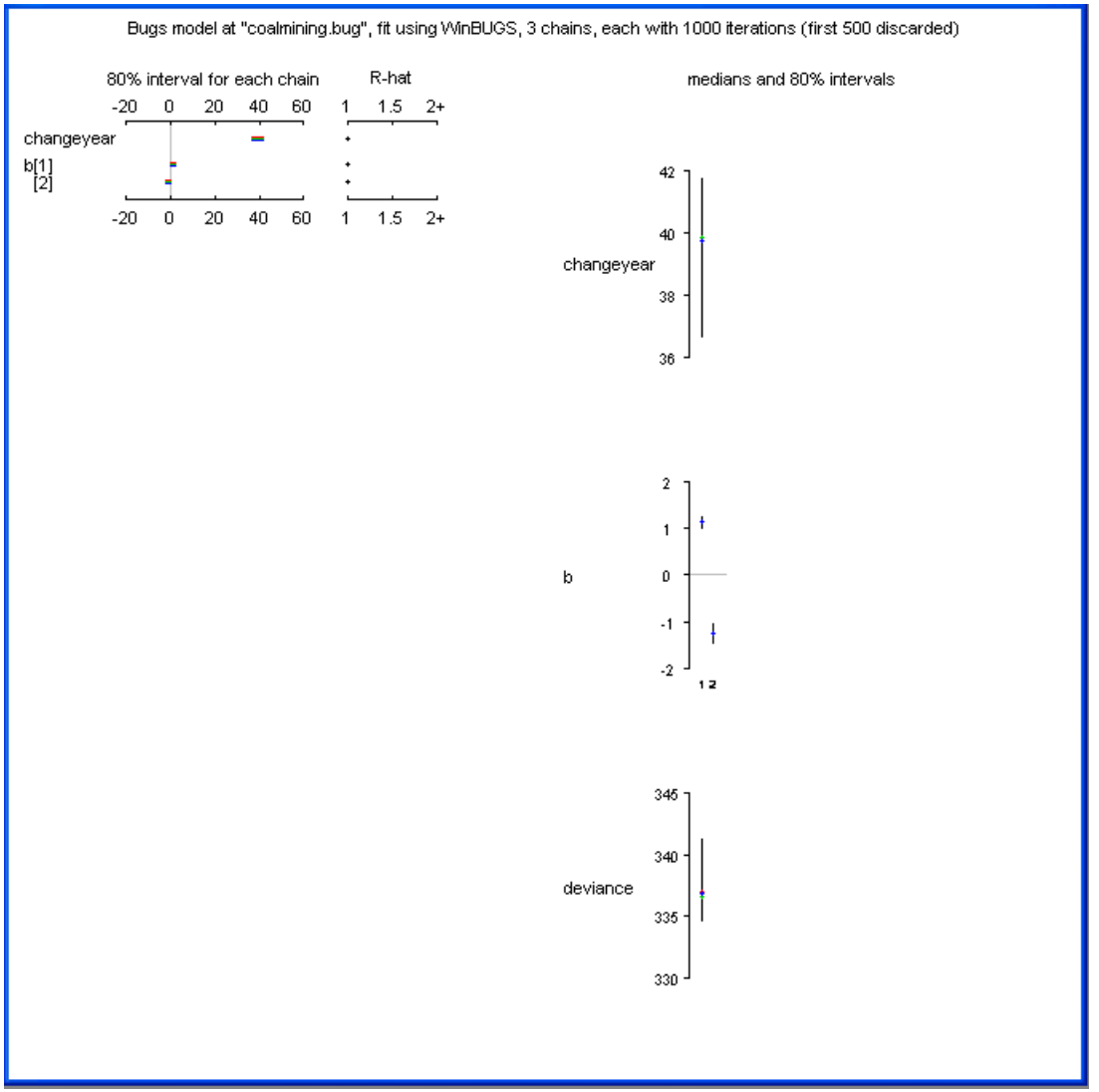
```
Inference for Bugs model at "coalmining.bug", fit using WinBUGS,
 3 chains, each with 1000 iterations (first 500 discarded)
 n.sims = 1500 iterations saved
             mean   sd  2.5%   25%   50%   75% 97.5% Rhat n.eff
changeyear   39.5  2.1  36.1  37.8  39.8  40.7  43.6    1  1500
b[1]          1.1  0.1   0.9   1.1   1.1   1.2   1.3    1   350
B[2]         -1.3  0.2  -1.6  - 1.4 -1.3  -1.2  -1.0    1  1300
deviance    337.5  2.6 334.2 335.6 336.8 338.6 344.0    1   820

For each parameter, n.eff is a crude measure of effective sample size,
and Rhat is the potential scale reduction factor (at convergence, Rhat=1).

DIC info (using the rule, pD = var(deviance)/2)
pD = 3.5 and DIC = 341.0
DIC is an estimate of expected predictive error (lower deviance is better).
```

WinBUGSの実行（おまけ2）

```
> plot( bugs(data,inits,parameters,"coalmining.bug",n.chains=3,n.iter=1000))
```



codaの利用(1)

- MCNCオブジェクトをcodaを利用しシミューれション標本を要約、グラフの作成ができる
- Summary関数よりMCMCの実行結果の要約統計量が出力できる
- 出力結果のdeviance行は次の逸脱度関数の事後平均と事後標準偏差をあらわしている

$$D(\theta) = -2\log L(\theta) + 2\mathrm{h}(y)$$

L(θ)：尤度，y(θ)：データの標準化関数

D(θ)の事後平均はモデルの適合度の要約量となる

summary()関数の実行結果

```
> summary(coalmining.coda)

Iterations = 501:1000
Thinning interval = 1
Number of chains = 3
Sample size per chain = 500

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

                Mean      SD  Naive SE Time-series SE
b[1]           1.137 0.09564 0.002469      0.003904
b[2]          -1.260 0.15729 0.004061      0.006112
changeyear    39.532 2.06306 0.053268      0.082982
deviance     337.463 2.64416 0.068272      0.102482

2. Quantiles for each variable:

                2.5%      25%      50%      75%    97.5%
b[1]          0.9427    1.076    1.136    1.200   1.3255
b[2]         -1.5675   -1.362   -1.258   -1.153  -0.9544
changeyear   36.0748   37.788   39.805   40.740  43.6205
Deviance    334.2000  335.600  336.800  338.600 344.0000
```

codaの利用(2)

- MCNCオブジェクトよりcodaパッケージの関数からMCNC診断グラフを作成することが出来る
- xyplot()

  -すべてのパラメータと逸脱度関数についてのトレースプロットを作成
- acfplot()

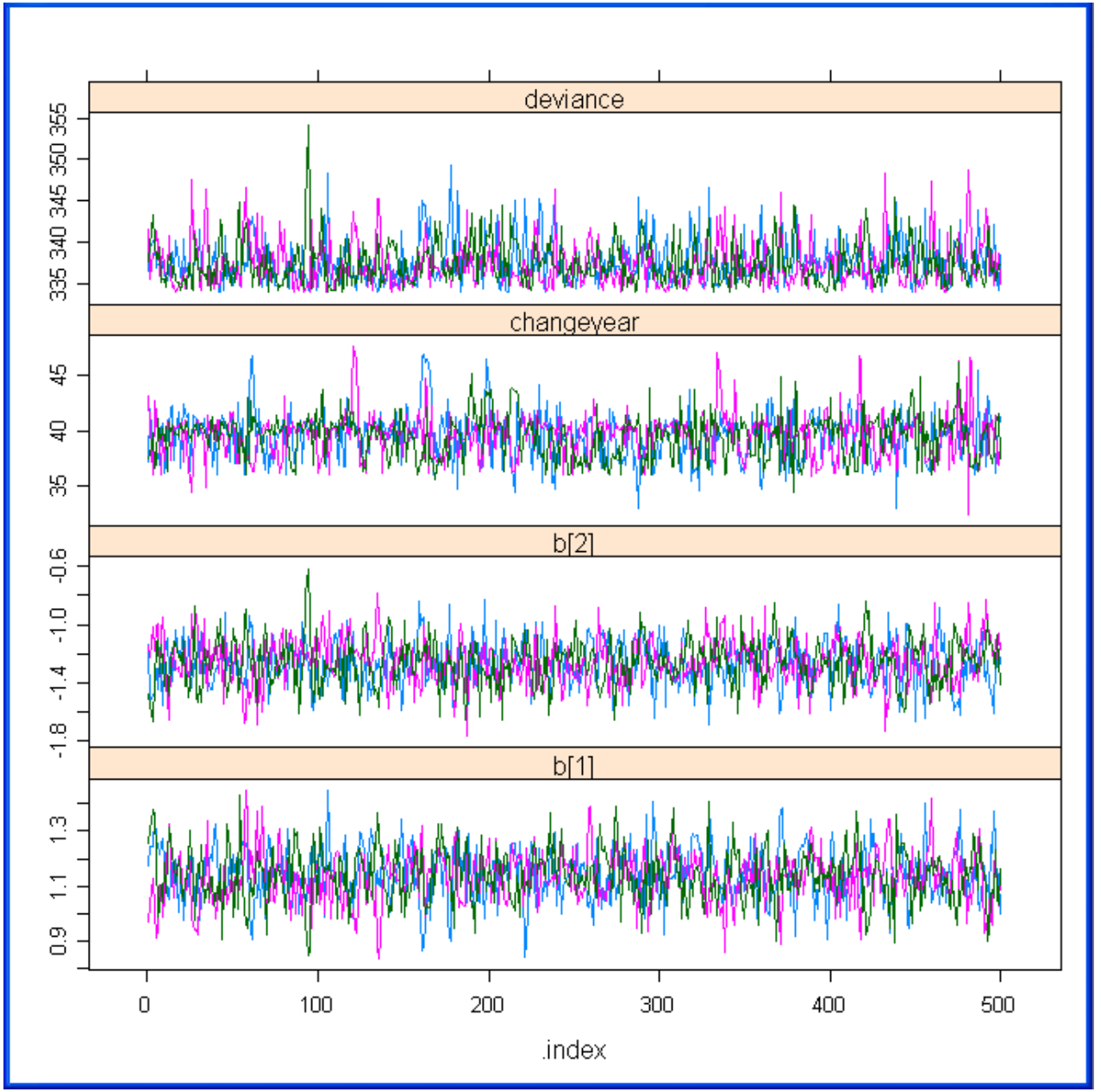
  -すべてのパラメータの自己相関グラフの作成
- densityplot()

  -パラメータの密度プロットの作成

xyplot()関数の実行結果

```
> xyplot(coalmining.coda)
```

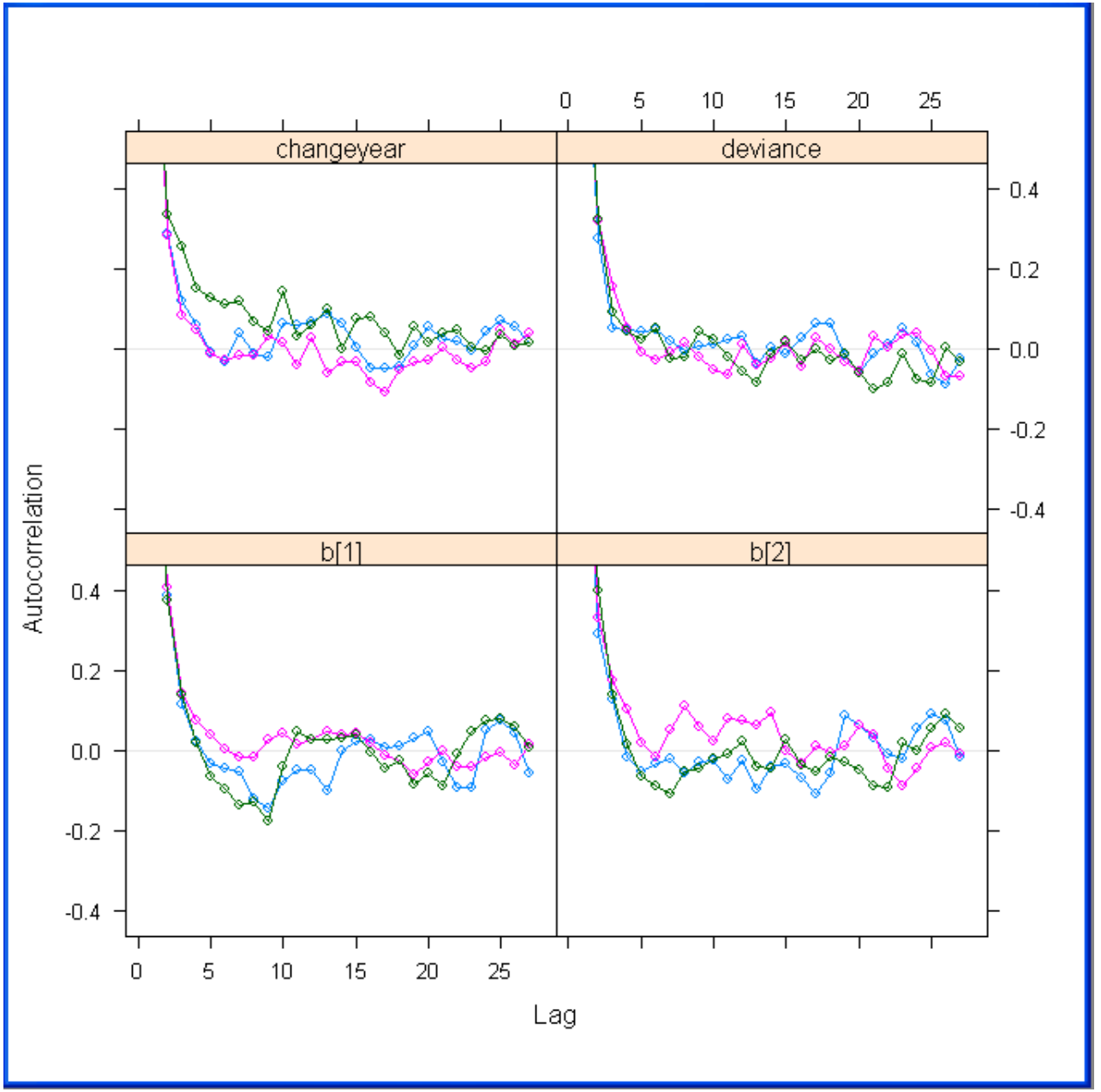
- 変化点問題のパラメータと逸脱度関数のトレースプロット



acfplot()関数の実行結果

```
acfplot(coalmining.coda)
```

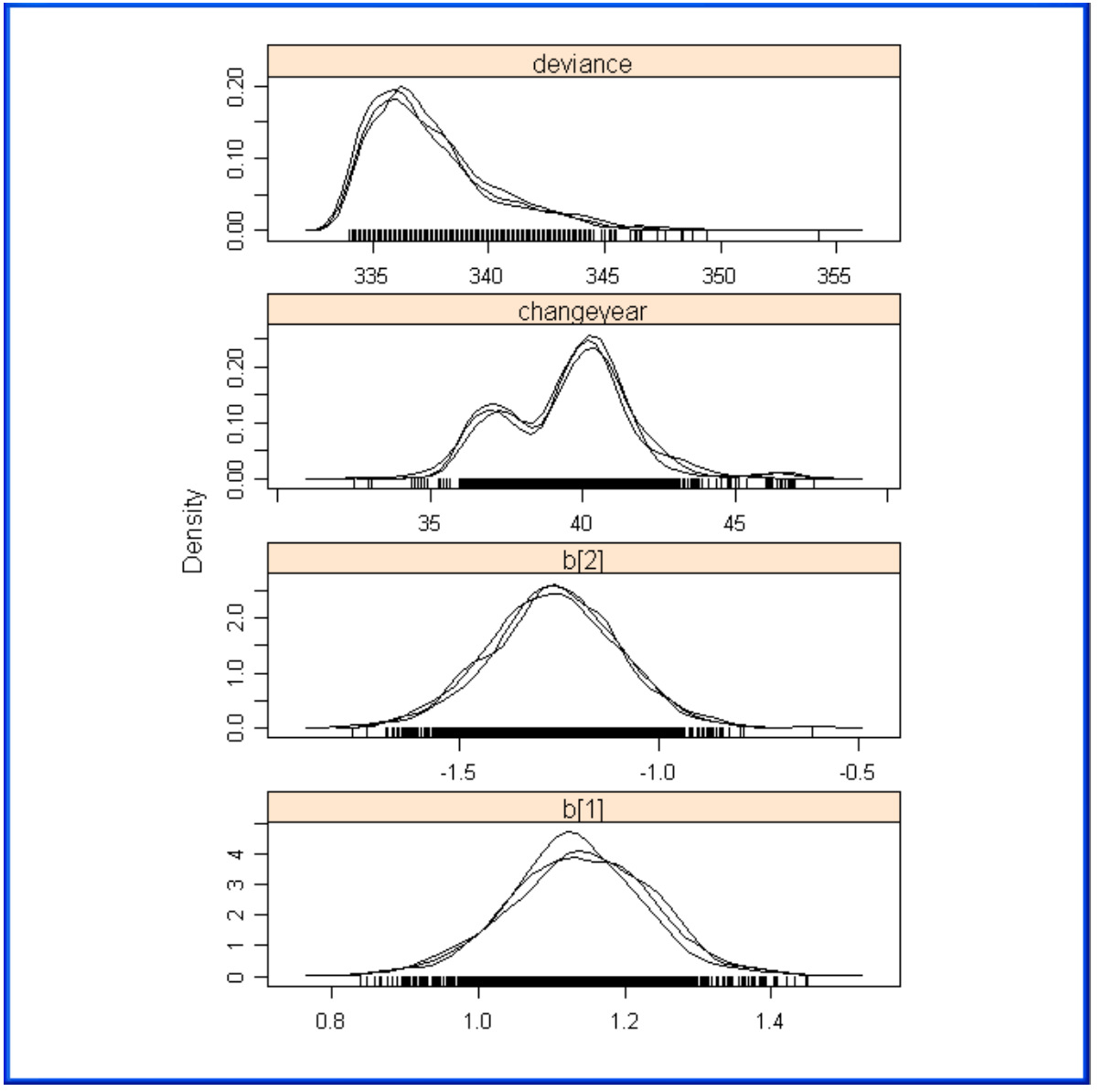
- 変化点問題のパラメータと逸脱度関数の自己相関プロット



densityplot()関数の実行結果

```
> densityplot(coalmining.coda, col = "black")
```

- τの密度が双峰形をしており、これは1850年から37年後か40年後に変化点があることを示唆している
- b[2]のグラフよりパラメータ$\beta_1 < 0$が明らかであり、変化点を越えると炭鉱設備の数が減少していることを示している



頑健な回帰モデル(1)

- 1996年と2000年それぞれの大統領選挙におけるフロリダ州での投票数をモデルとし関係を考察する

  -LearnBayesパッケージのelectionデータセットを利用。フロリダ州67地区の2000年の改革党候補パット・ブキャナンと1996年の改革党候補ロス・ペローへの投票数が記録されている

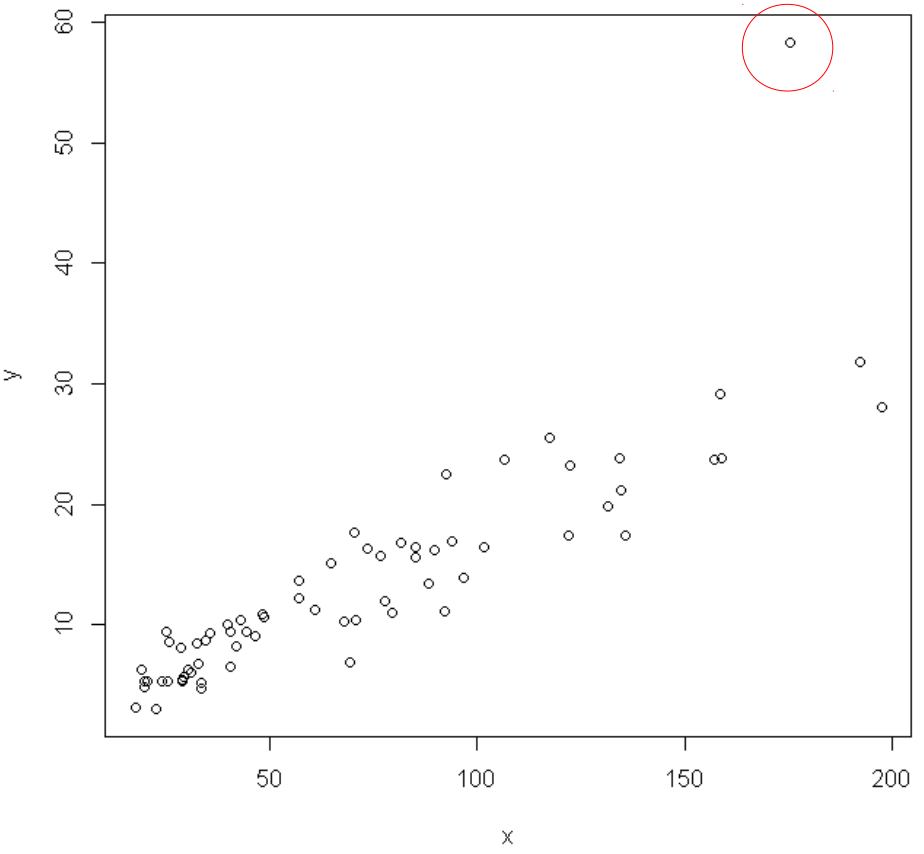
頑健な回帰モデル(2)

- 右図は各候補者の得票数をそれぞれ平方根に変換した値による散布図

  -ほぼ線形だが一つだけ外れ値がある

  ⇒パームビーチ郡でブキャナンへの得票数が以上に高かった

- y_i, x_iをそれぞれブキャナンとベローの得票数の平方根とする



頑健な回帰モデル(3)

- $y_1,\ldots,y_n$は次の回帰モデルにしたがうと仮定する

$$y_i = \beta_0 + \beta_1 x_i + \epsilon_i$$

$\varepsilon_1,\ldots,\varepsilon_n$は平均0で尺度パラメータσ,自由度v=4のt分布からのランダム標本

- 上式の回帰モデルを正規分布による次の尺度混合形であらわす

$$\begin{aligned} y_i &\sim N\left(\beta_0 + \beta_1 x_i, \left(\tau\lambda_i\right)^{-1}/2\right), \\ \lambda_i &\sim gamma\left(2,2\right) \end{aligned}$$

- β_0とβ_1に一様分布、精度τには1/τに比例する標準的な無情報事前分布を割り当てる

WinBUGSで回帰モデルを定義(1)

- WinBUGSでは次のようにmodelを定義する
  - 観測値：y[1],...,y[N]
  - 観測平均：mu[1],...,mu[N]
  - 観測精度：p[1],...,p[N]
  - i番目の精度p[i]：tau*lam[i]
  - 尺度パラメータlam[i]にガンマ分布gamma(2,2)を割り当てる
- パラメータに非正規事前分布を割り当てるのは正式には不可能

  ⇒ b[1]に平均0で小さな精度0.001の世紀事前分布を割り当てることで一様事前分布に近似させる

  ⇒パラメータtauには形状パラメータと尺度パラメータに0.001という小さな値を設定したガンマ分布を割り当てる

WinBUGSで回帰モデルを定義(2)

- 定義をもとに作成したスクリプトをrobust.bugというファイルとして保存する

```
model
{
	for(i in i:N)
	{
		y[i] ~ dnorm(mu[i], p[i])
		p[i] <- tau*lam[i]
		lam[i] ~ dgamma(2,2)
		mu[i] <- b[1]+b[2]*x[i]
	}
	for(j in 1:2)
	{
		b[j] ~ dnorm(0,0.001)
	}
	tau ~ dgamma(0.001,0.001)
}
```

データの入力

- Rでデータを定義する。対の観測数N、反応のベクトルy、共変量ベクトルxを定義する

```
> data(election
> attach(election)
> y <- sqrt(buchanan)
> x <- sqrt(perot)
> N <- length(y)
```

- 初期値として回帰パラメータに0,0、精度パラメータτに1を設定。parameterを定義しτとベクトル{λi}、回帰係数のベクトルβiをモニタする対象として指定。

```
> data <- list("N","y","x")
> inits <- function() {list(b = c(0,0), tau = 1 ) }
> parameters <- c("tau","lam","b")
```

WinBGUSの実行

- bugs関数でモデルからシミュレーション結果を得る

```
> robust.sim <- bugs (data, inits, parameters, "robust.bug")
```

キャリア軌跡の推定(1)

- プロのアスリートはキャリアの半ばであるピークまで成績が上昇その後下がっていく。今回は野球選手の記録をモデルとしピーク年齢とピーク時の能力について推定する。LearnBayseパッケージのsluggerdataデータセットを利用する。

  - 現役j年目の打数 : n_j

  - 現役j年目のホームラン数 : y_j

  ホームラン率 y_j/n_j を選手の年齢x_jの関数として表現したい。

キャリア軌跡の推定(2)

- yiは二項分布binomial(nj,pj)に従うとする
- pjはj年目におけるホームランの確率で、次のロジスティック2次モデルに従うと仮定する

$$\log\left(\frac{p_j}{1-p_j}\right)=\beta_0+\beta_1 x_j+\beta_2 X_j^2$$

β2<0の時確率は次の値で最大化される

$$age_{PEAK}=\frac{-\beta_1}{2\beta_2}$$

確率の値のピークは次のようになる

$$PEAK=\beta_0-\frac{\beta_1^2}{4\beta_2}$$

キャリア軌跡の推定(3)

- 選手の現役期間は15～20年にすぎず2項分布の分散が相当に大きい

  ⇒　正確な推定値を得るのが難しい

- そこで同じようなキャリア軌跡を持つ野球選手のデータと組み合わせ、それによって改善された推定値を得るよう試みる

  - 交換可能モデルをあてはめる

キャリア軌跡の推定(4)

- 同等のキャリアの選手がk人いるとする
  - y_{ij} : 選手iのj年目のホームラン数
  - n_{ij} : 選手iのj年目の打数
  - x_{ij} : 選手iのj年目の年齢
- 確率{pij}は次のロジスティックモデルを満たすと仮定する(j=1,...,T_i)

$$\log\left(\frac{p_{ij}}{1-p_{ij}}\right)=\beta_{i0}+\beta_{i1}x_{ij}+\beta_{i2}x_{ij}^2$$

キャリア軌跡の推定(5)

- βi = (βi0, βi1, βi2)：選手iの回帰ベクトル

  - 交換可能とする確信を表現するためβ1,...,βkは平均μβで分散共分散行列Vの共通の多変量正規事前分布からのランダムな標本とする

$$\beta_i | \mu_\beta, R \sim N_3(\mu_\beta, V) \quad (i = 1, \ldots, k)$$

- 次に超パラメータに曖昧事前分布を割り当てる。

$$\mu_\beta \sim c, \quad V \sim inverse\,Wishart(S^{-1}, v)$$

$inverse\,Wishart(S^{-1}, v)$ は尺度行列S、自由度vの逆ウィシャート分布である。WinBUGSでは精度行列Pに置くことで表現できる

$$P = V^{-1} \sim Wishart(S, v)$$

WinBUGSでの定義(1)

- 最初にcareertraj.setup()関数を使いデータセットsluggerdataから利用する行列を抽出する

```
> data(sluggerdata)
> s <- careertraj.setup(sluggerdata)
> N <- s$N; T <- s$T; y <- s$y; n <- s$n; x <- s$x
```

オブジェクトN : 選手の数

ベクトルT : 各選手の現役シーズン数

行列y : i番目の行が選手iの各シーズンのホームラン数

行列n : 全選手の打数

行列x : 各シーズンでの年齢

WinBUGSでの定義(2)

- Career.bugファイルに以下のようにモデルを記述する

```
model
{
	for(i in 1 : N)
	{
		beta[i, 1:3] ~ dmnorm(mu.beta[], R[ , ])
		for(j in 1 : T [i])
		{
			y[i, j] ~ dbin(p[i, j], n[i, j])
			logit(p[i, j]) <- beta[i, 1] + beta[i, 2] * x[i, j] + beta[i, 3] * x[i, j] * x[i, j]
		}
	}
	mu.beta[1:3] ~ dmnorm(mean[1:3], prec[1:3, 1:3])
	R[1:3, 1:3] ~ dwish(Omega[1:3, 1:3], 3)
}
```

データの入力

- 超パラメータの値を定義

```
> mean <- c(0,0,0)
> Omega <- diag(c(.1, .1, .1))
> prec <- diag(c(1.0E-6, 1.0E-6, 1.0E-6))
```

- β, μβ, Rの初期推定値を与える

```
> beta0 <- matrix(c(-7.69, .350, -.0058), nrow = 10, ncol = 3, byrow = TRUE)
> mu.beta0 <- c(-7.69, .350, -.0058)
> R0 <- diag(c(.1,.1,.1))
```

- dataの行に変数のリストを指定、inits()関数で初期値を設定、parameterは行列betaだけをモニタすることをあらわす

```
> data <- list("N", "T", "y", "n", "x", "mean", "Omega", "prec")
> inits <- function() {list(beta = beta0, mu.beta = mu.beta0, R = R0)}
> parameters <- c("beta")
```

シミュレーション標本の作成

- bugs()関数を実行する

```
> career.sim <- bugs (data, inits, parameters, "career.bug",
+ n.chains = 1, n.iter = 50000, n.thin = 1)
```

βのシミュレーション標本は要素career.sim$sims.list$betaに含まれる。beta[, i, j]にはβ_{ij}のシミュレーション標本が含まれる

- ピーク年齢のシミュレーション標本を作成する

```
> peak.age <- matrix(0, 50000, 10)
> for(i in 1:10)
+ peak.age[,i] = -career.sim$sims.list$beta[, i, 2]/2/
+ career.sim$sims.list$beta[, i, 3]
```

グラフ化

- codaパッケージの関数を利用する
- はじめにdimname()関数を使いシミュレーション標本の行列の各列に選手名をラベルする

```
> dimnames(peak.age)[[2]] <- c("Aaron", "Greenberg", "Killebrew",
+ "Mantle", "Mays", "McCovey", "Ott", "Ruth",
+ "Schmidt", "Sosa")
```

- densityplot()関数で10名の線湯のピーク年齢の推定密度を構築する

グラフ化（2）

```
> densityplot(as.mcmc(peak.age), plot.points = FALSE)
```

- 右図が10名の野球選手のピーク年齢パラメータの密度推定値
- おおよそ30代前半がホームランを打つ能力ががピークとなっている