Bruchpunkt- und Mischverteilungsmodelle

Volker Schmid

17. Juli 2017

Bruchpunktmodelle

Bayesianische Mischverteilungsmodelle

Bruchpunktmodelle

Bruchpunktmodelle

- Beim Bruchpunktmodell ändert sich ein Parameter eines Modells an einem Punkt sprungartig
- In der Regel Anwendung auf Zeitreihen
- Theoretisch diskrete oder stetige Zeit möglich

Beispiele:

- Aktienkurse
- Schlafdaten
- Krankheiten mit Schüben
- Stärke von Sportmannschaften
- Zeitreihen von Krankheitsfällen (Bruchpunkt: Ausbruch einer Epidemie)

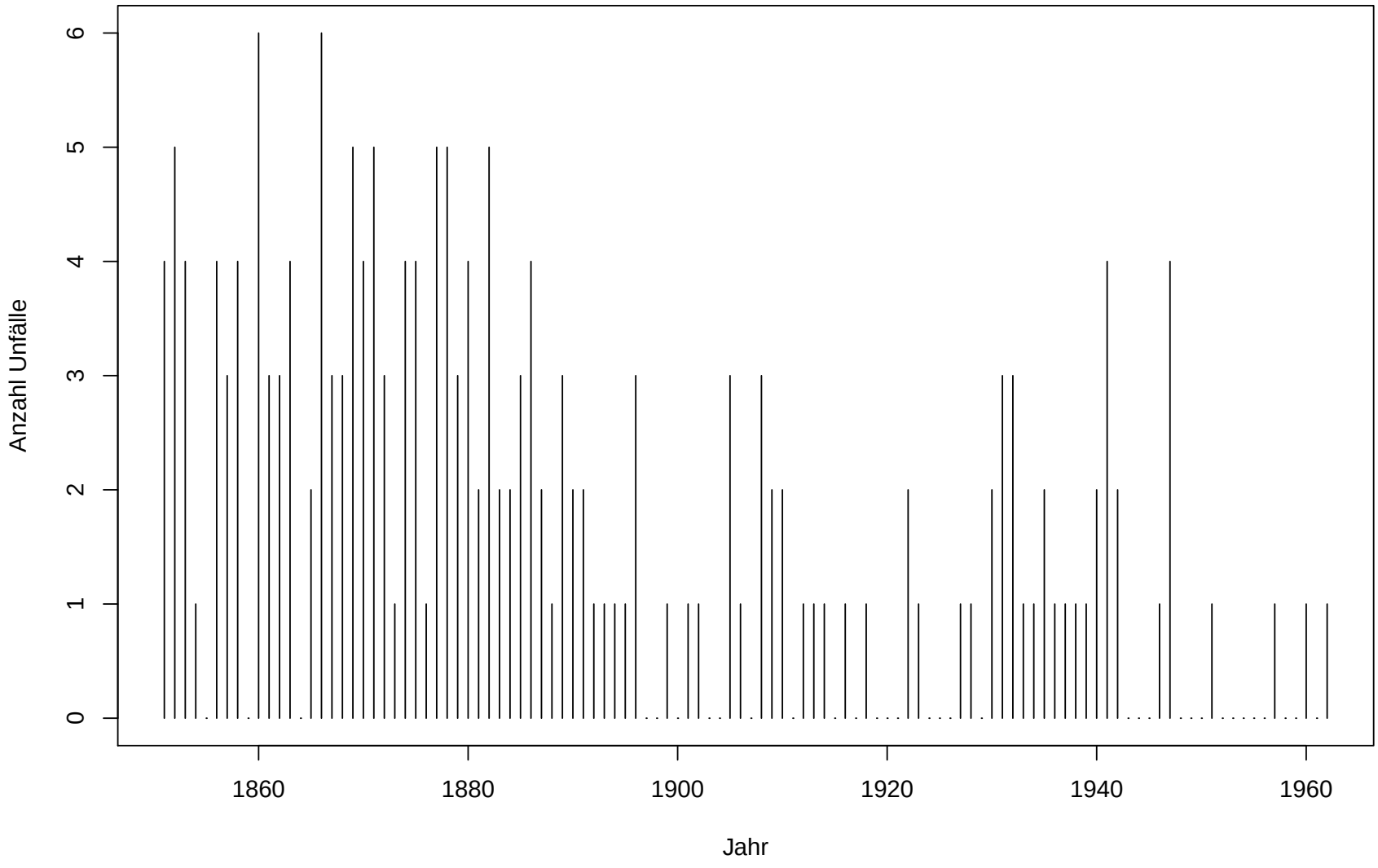
Beispiel: Unfälle in englischen Kohlebergwerken I

Der Datensatz {coal.txt} enthält die jährliche Anzahl von Unfällen in englischen Kohlebergwerken wärend der Jahre 1851-1962 (insgesamt 112 Jahre). Ein Plot der Daten zeigt einen deutlichen Rückgang der Unfälle ab etwa 1900.

```
coal <- read.table("../bayeskurs/data/coal.txt", header = T
```

```
plot(coal$year, coal$disasters, type = "h",
     xlab = "Jahr", ylab = "Anzahl Unfälle")
```

Beispiel: Unfälle in englischen Kohlebergwerken II



Bruckpunktmodell

Ein Bruchpunktmodell für diese Daten $\mathbf{y} = (y_1, \ldots, y_{112})$ hat folgende Form:

$$Y_i \sim \begin{cases} Po(\lambda_1), & i = 1, \ldots, \theta, \\ Po(\lambda_2), & i = \theta + 1, \ldots, 112, \end{cases}$$

Prioris:

- $\lambda_i \mid \alpha \sim Ga(3, \alpha)$ für $i = 1, 2$
- $\alpha \sim Ga(10, 10)$
- $\theta \sim U\{1, \ldots, 112\}$.

Posteriori

Die gemeinsame Posteriori-Dichte ist gegeben durch:

$$
\begin{aligned}
p(\lambda_1, \lambda_2, \alpha, \theta \mid \mathbf{y}) &\propto p(\mathbf{y} \mid \lambda_1, \lambda_2, \alpha, \theta)\, p(\lambda_1, \lambda_2, \alpha, \theta) \\
&= p(\mathbf{y} \mid \lambda_1, \lambda_2, \alpha, \theta)\, p(\lambda_1 \mid \lambda_2, \alpha, \theta) \\
&\quad \cdot p(\lambda_2 \mid \alpha, \theta)\, p(\alpha \mid \theta)\, p(\theta) \\
&= p(\mathbf{y} \mid \lambda_1, \lambda_2, \theta)\, p(\lambda_1 \mid \alpha)\, p(\lambda_2 \mid \alpha)\, p(\alpha) p\,(\theta) \\
&\propto \left(\prod_{i=1}^{\theta} \lambda_1^{y_i} \exp(-\lambda_1) \right) \left(\prod_{i=\theta+1}^{n} \lambda_2^{y_i} \exp(-\lambda_2) \right) \\
&\quad \cdot \underbrace{\alpha^3 \lambda_1^{3-1} \exp(-\alpha\lambda_1)}_{\lambda_1 \mid \alpha \sim Ga(3,\alpha)} \underbrace{\alpha^3 \lambda_2^{3-1} \exp(-\alpha\lambda_2)}_{\lambda_2 \mid \alpha \sim Ga(3,\alpha)} \\
&\quad \cdot \underbrace{\alpha^{10-1} \exp(-10\alpha)}_{\alpha \sim Ga(10,10)} \underbrace{\mathsf{I}(1 \leq \theta \leq 112)}_{\theta \sim \mathsf{U}\{1,\ldots,112\}} .
\end{aligned}
$$

Full Conditionals I

$$\lambda_1 \mid \lambda_2, \alpha, \theta, \mathbf{y} \propto \left(\prod_{i=1}^{\theta} \lambda_1^{y_i} \exp(-\lambda_1) \right) \lambda_1^{3-1} \exp(-\alpha\lambda_1)$$
$$= \lambda_1^{3+\sum_{i=1}^{\theta} y_i - 1} \exp\left(-\lambda_1(\theta + \alpha) \right)$$

Dies ist der Kern einer $Ga(3 + \sum_{i=1}^{\theta} y_i, \theta + \alpha)$-Verteilung.

$$\lambda_2 \mid \lambda_1, \alpha, \theta, \mathbf{y} \propto \left(\prod_{i=\theta+1}^{n} \lambda_2^{y_i} \exp(-\lambda_2) \right) \lambda_2^{3-1} \exp(-\alpha\lambda_2)$$
$$= \lambda_2^{3+(\sum_{i=\theta+1}^{n} y_i)-1} \exp\left(-\lambda_2(\alpha + [n - \theta]) \right)$$

Dies ist der Kern einer $Ga(3 + \sum_{\theta+1}^{112} y_i, \alpha + 112 - \theta)$, da $n = 112$.

Full Conditionals II

$$\begin{aligned}\alpha \mid \lambda_1, \lambda_2, \theta, \mathbf{y} &\propto \alpha^3 \exp(-\alpha\lambda_1)\alpha^3 \exp(-\alpha\lambda_2)\alpha^{10-1} \exp(-10\alpha) \\ &= \alpha^{16-1} \exp\big(-\alpha(10+\lambda_1+\lambda_2)\big)\end{aligned}$$

Dies ist der Kern einer $Ga(16, 10+\lambda_1+\lambda_2)$.

Full Conditionals III

$$
\begin{aligned}
\theta \mid \lambda_1, \lambda_2, \alpha, \mathbf{y} &\propto \left(\prod_{i=1}^{\theta} \lambda_1^{y_i} \exp(-\lambda_1) \right) \left(\prod_{i=\theta+1}^{n} \lambda_2^{y_i} \exp(-\lambda_2) \right) \\
&\quad \cdot \mathsf{I}(1 \leq \theta \leq 112) \\
&\propto \left(\prod_{i=1}^{\theta} \lambda_1^{y_i} \exp(-\lambda_1) \right) \left(\prod_{i=\theta+1}^{n} \lambda_2^{y_i} \exp(-\lambda_2) \right) \\
&\quad \cdot \frac{\left(\prod_{i=1}^{\theta} \lambda_2^{y_i} \exp(-\lambda_2) \right)}{\left(\prod_{i=1}^{\theta} \lambda_2^{y_i} \exp(-\lambda_2) \right)} \mathsf{I}(1 \leq \theta \leq 112) \\
&\propto \left(\prod_{i=1}^{\theta} \lambda_1^{y_i} \lambda_2^{-y_i} \exp(-\lambda_1) \exp(\lambda_2) \right) \mathsf{I}(1 \leq \theta \leq 112) \\
&= \exp\left(\theta(\lambda_2 - \lambda_1)\right) \left(\frac{\lambda_1}{\lambda_2} \right)^{\sum_{i=1}^{\theta} y_i} \mathsf{I}(1 \leq \theta \leq 112)
\end{aligned}
$$

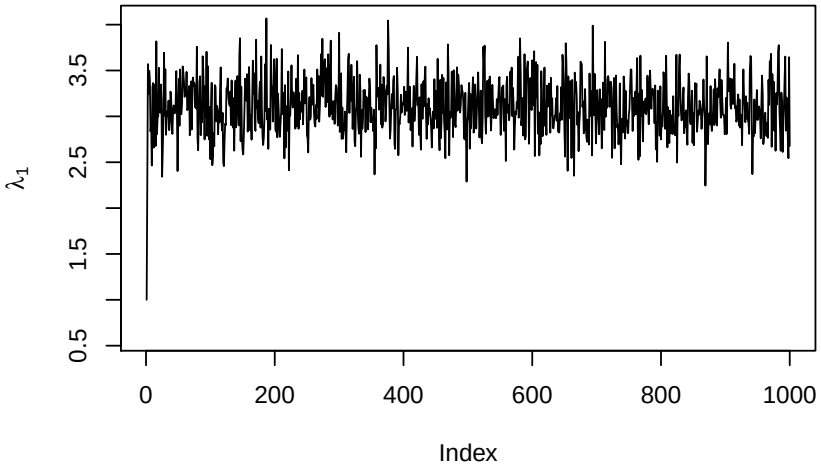
Full Conditionals IV

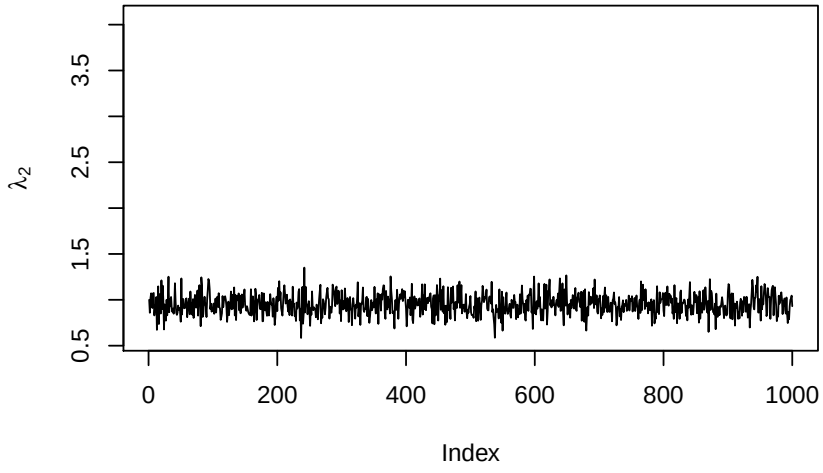
$$\theta \mid \lambda_1, \lambda_2, \alpha, \mathbf{y} = \exp\left(\theta(\lambda_2 - \lambda_1)\right) \exp\left\{\left(\sum_{i=1}^{\theta} y_i\right) \log\left(\frac{\lambda_1}{\lambda_2}\right)\right\} \cdot \mathsf{I}(1 \leq \theta \leq 112)$$

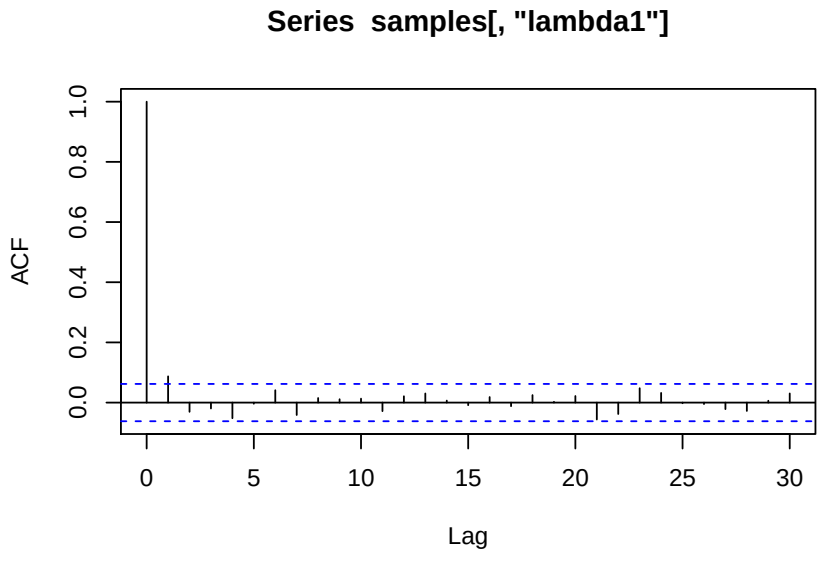
Ergebnisse I

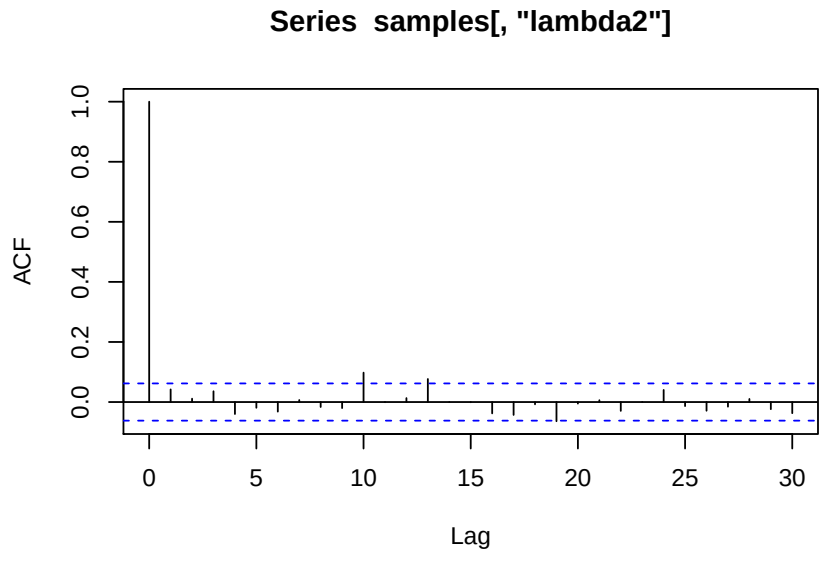
```
samples <- breakpoint.gibbs(1000, y = coal$disaster)

# plots
par(mfrow = c(2,2))
plot(samples[,"lambda1"], type = "l",
     ylab = expression(lambda[1]),
     ylim = c(min(samples[,1:2]), max(samples[,1:2])))
plot(samples[,"lambda2"], type = "l",
     ylab = expression(lambda[2]),
     ylim = c(min(samples[,1:2]), max(samples[,1:2])))
acf(samples[,"lambda1"])
acf(samples[,"lambda2"])
```

Ergebnisse II







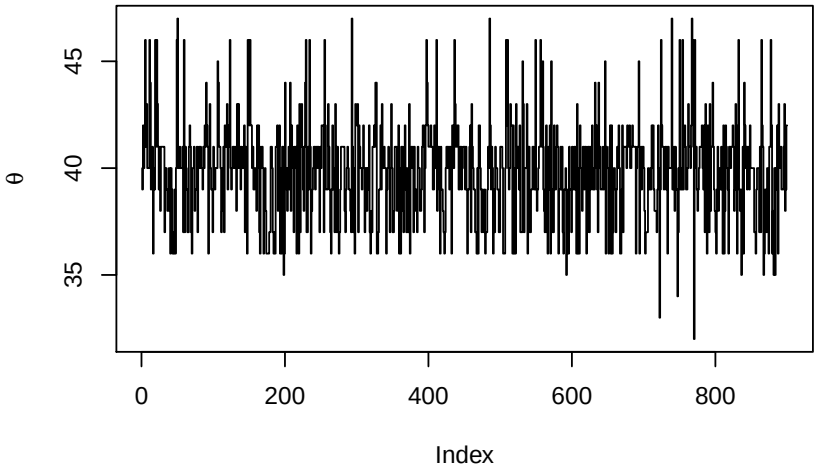


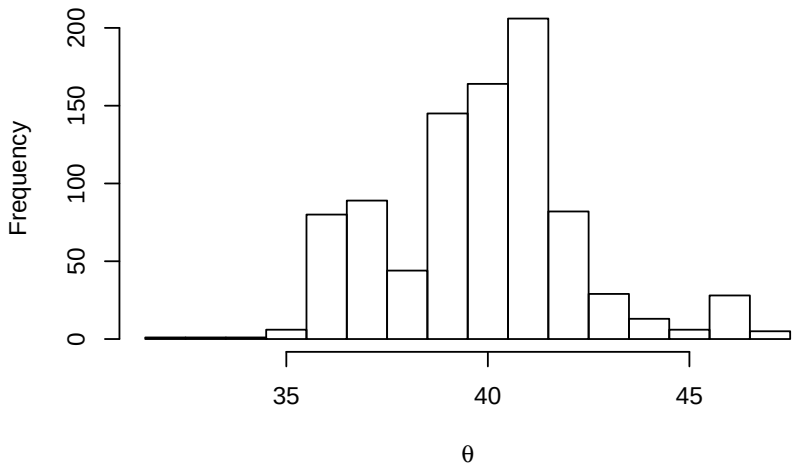
Betrachtung der Pfade: 1000 Realisationen reichen aus, fast kein Burn-in notwendig, kaum Autokorrelation.

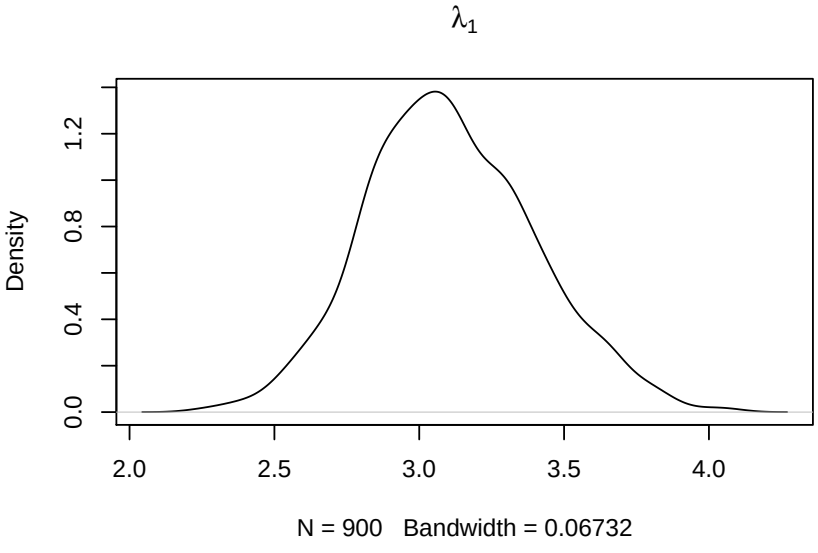
Ergebnisse III

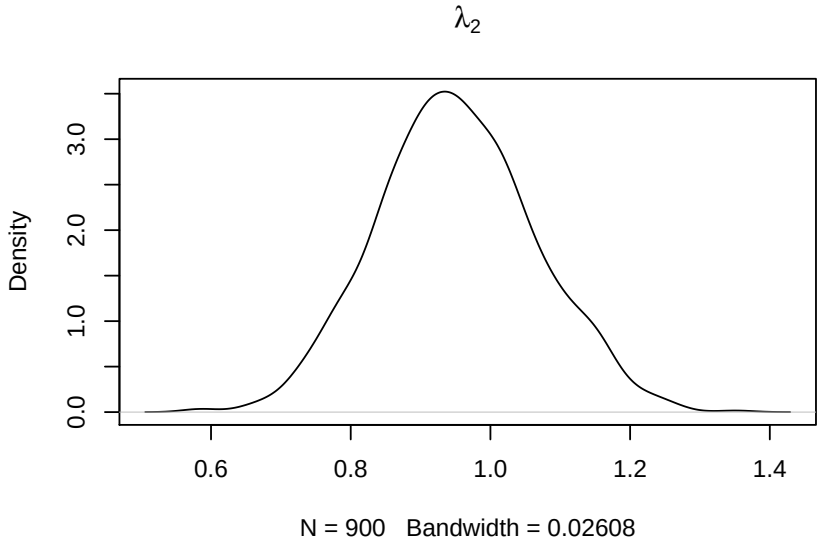
```
burnin <- 1:100 # hier großzügiger burn-in angewendet
par(mfrow = c(2,2))
# theta
plot(samples[-burnin, "theta"], type = "S",
     ylab = expression(theta))
hist(samples[-burnin, "theta"], xlab = expression(theta),
     breaks = ((min(samples[-burnin, "theta"])-1):
                  (max(samples[-burnin, "theta"])))+0.5,
     main = "" )
# lambda_1
plot(density(samples[-burnin, "lambda1"]), main =
       expression(lambda[1]))
# lambda_2
plot(density(samples[-burnin, "lambda2"]), main =
       expression(lambda[2]))
```

Ergebnisse IV









- Mittelwert und Median für alle Parameter:

Ergebnisse V

```
apply(samples[-burnin,], MAR = 2, mean)
```

```
##    lambda1    lambda2      alpha      theta
##  3.1088222  0.9511412  1.1418192 39.8566667
```

```
apply(samples[-burnin,], MAR = 2, median)
```

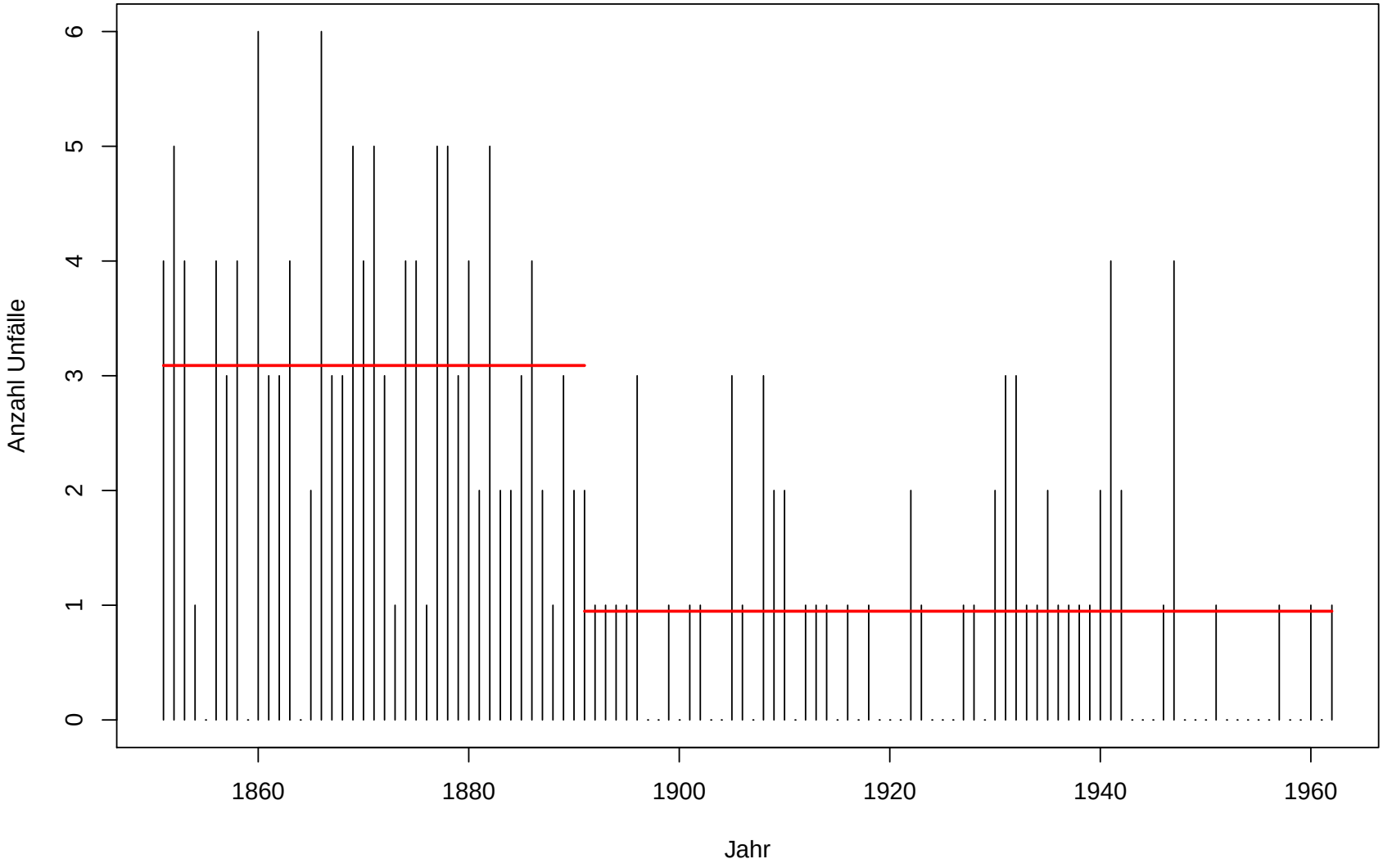
```
##    lambda1    lambda2      alpha      theta
##  3.0893246  0.9476406  1.1321178 40.0000000
```

Median-Modell im Histogramm:

Ergebnisse VI

```
plot(coal$year, coal$disasters, type = "h",
     xlab = "Jahr", ylab = "Anzahl Unfälle")
medianmodel <- apply(samples[-burnin,], MAR = 2, median)
jahr1 <- min(coal$year)
lines(c(jahr1, jahr1 + medianmodel["theta"]),
      rep(medianmodel["lambda1"], 2), col=2, lwd = 2)
lines(c(jahr1 + medianmodel["theta"], max(coal$year)),
      rep(medianmodel["lambda2"], 2), col=2, lwd = 2)
```

Ergebnisse VII



Bayesianische Mischverteilungsmodelle

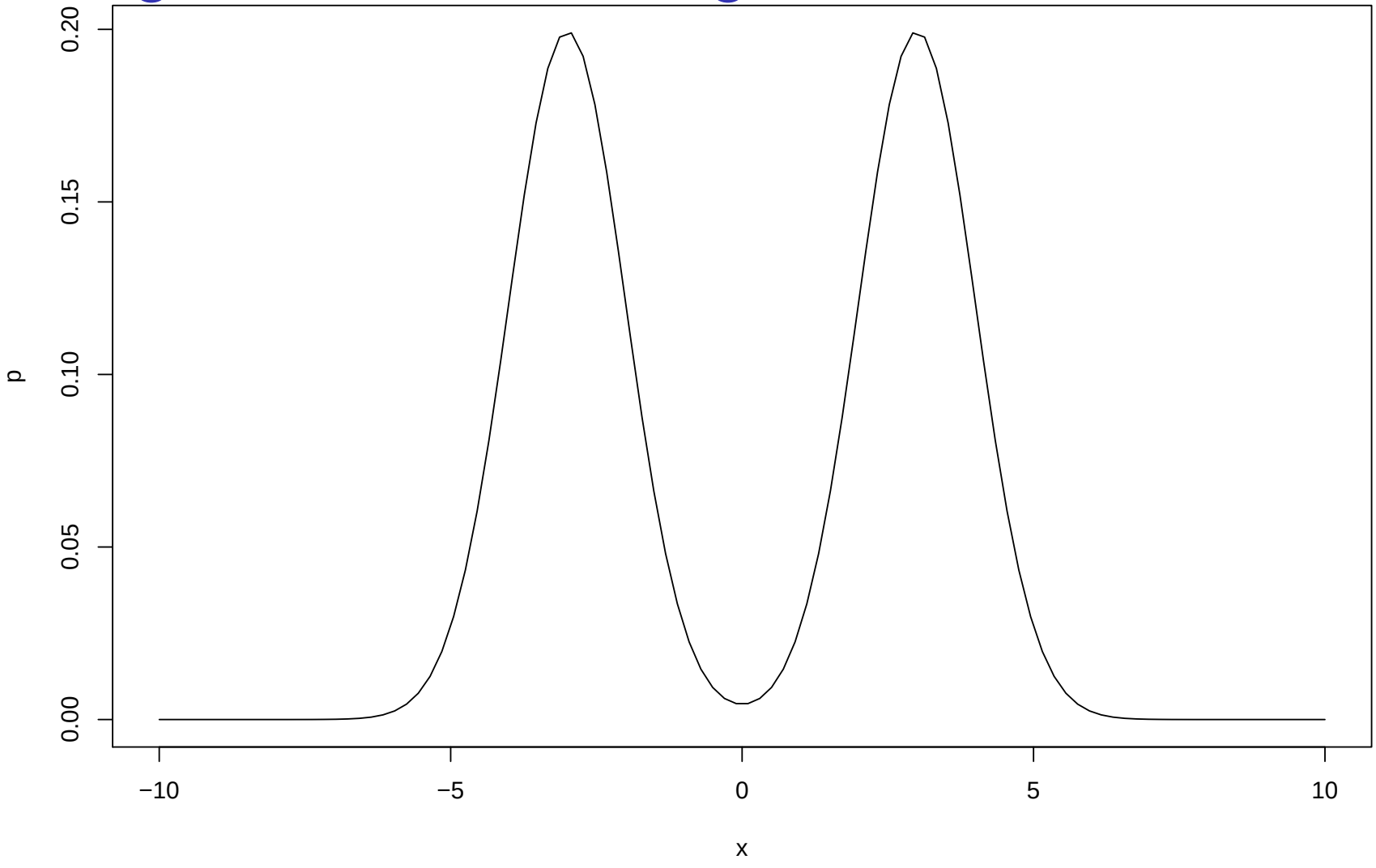
Mischverteilungen

- (Normal-)Verteilungsannahme basiert in der Regel auf der Annahme, dass die Beobachtungen identisch verteilt sind, es also eine Verteilung der Grundgesamtheit gibt
- Alternative Annahme: Es gibt mehrere Gruppen in der Grundgesamheit, die alle die selbe Verteilungsklasse haben, aber unterschiedliche Parameter
- Einfachster Fall: Mischung von Normalverteilungen

Mischung von Normalverteilungen I

```
x <- seq(-10, 10, length=100)
p <- 0.5*dnorm(x,-3,1) + 0.5*dnorm(x,3,1)
plot(x,p, type="l")
```

Mischung von Normalverteilungen II

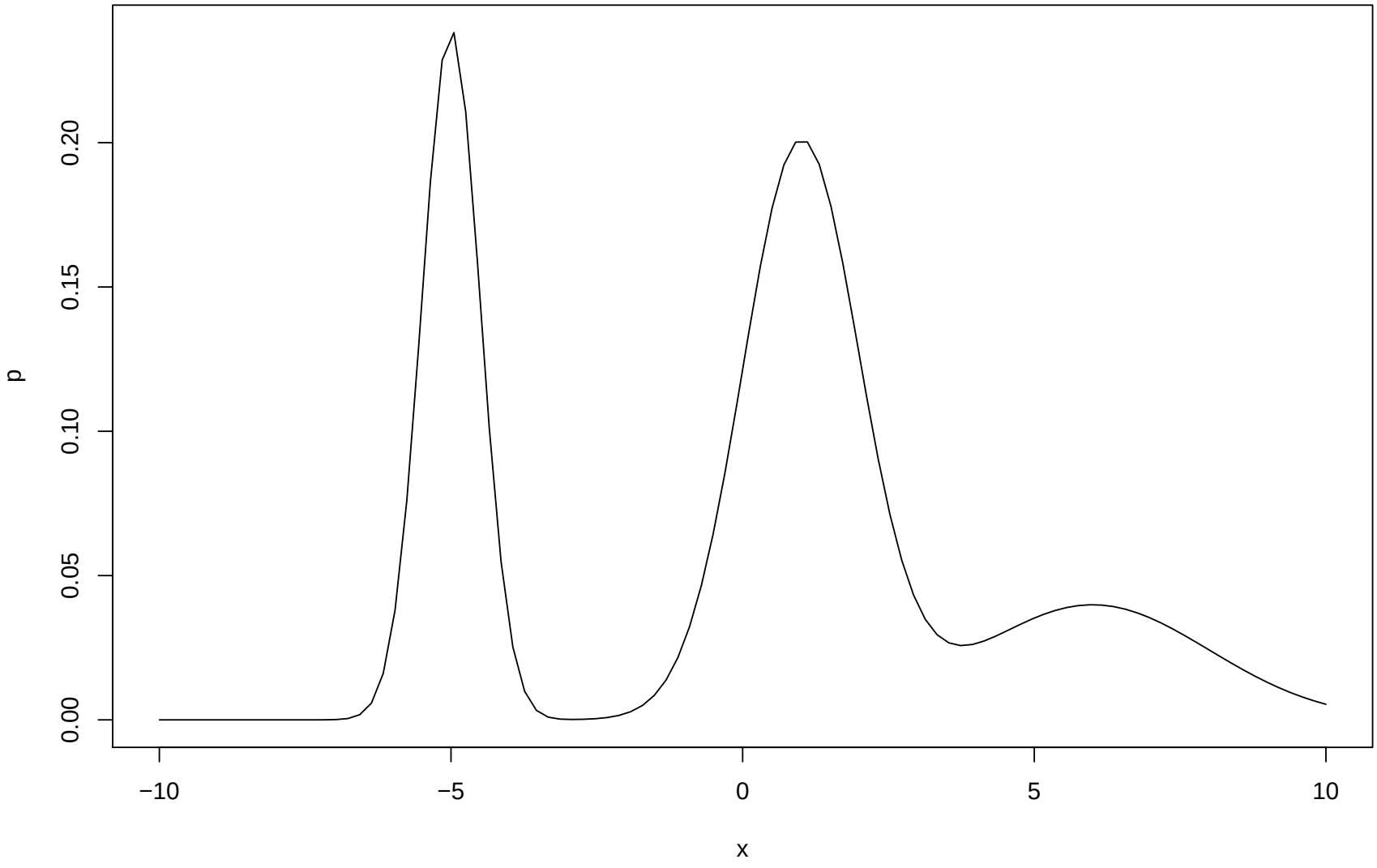


- Natürlich sind auch verschiedene Varianzen und verschiedene Gewichte möglich

Mischung von Normalverteilungen III

```
x <- seq(-10, 10, length=100)
p <- 0.3*dnorm(x,-5,0.5) + 0.5*dnorm(x,1,1) + 0.2*dnorm(x,6
plot(x,p, type="l")
```

Mischung von Normalverteilungen IV



Fragen

- Welche Parameter und welche Gewichte haben die einzelnen Verteilungen?
- Wieviele Mischverteilungen brauchen wir?
- Zu welcher Verteilung gehört welche Beobachtung

Latente Klassen

Idee der Modellierung: Führe latente Klassen S_i ein

$$
\begin{aligned}
x_i | S_i &\sim N(\mu_{S_i}, \sigma^2_{S_i}) \\
\mu_j &\sim N(\mu_0, \sigma^2_0) \\
\sigma^2_j &\sim IG(a, b) \\
p(S_i = j) &= \pi_j \\
(\pi_1, \ldots, \pi_K) &\sim Diri(\alpha, \ldots, \alpha)
\end{aligned}
$$

Dirichlet-Verteilung:

$$
p(\pi_1, \ldots, \pi_K) \propto \prod \pi_i^{\alpha_i - 1}
$$

Full conditionals

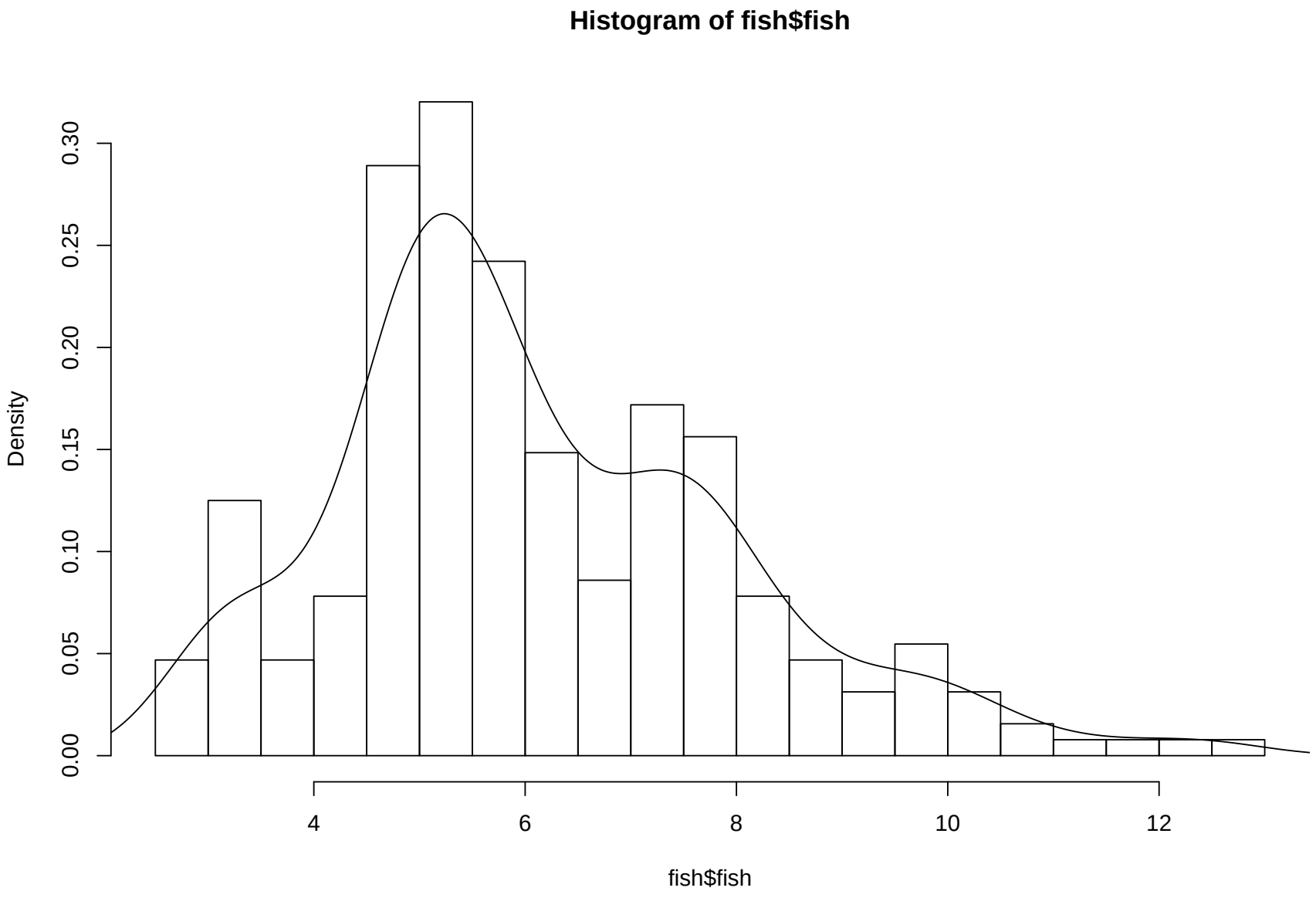
- Die full conditional von μ_j sind Normalverteilungen, wobei die x_i mit $S_i = j$ eingehen
- Analog ist die full conditional von σ_j^2 eine IG-Verteilung
- $p(S_i = j)$ berechnet sich aus Dichte von x_i mit μ_j und σ_j^2 (und Priori)
- Full Conditional von π ist Dirichlet mit $(\alpha + n_1, \ldots, \alpha + n_k)$, wobei n_j die aktuelle Anzahl der Beobachtungen in Klasse j ist

Beispiel I

Länge von 256 Fischen (aus D. M. Titterington, A. F. M. Smith and U.E. Makov (1985) Statistical Analysis of Finite Mixture Distributions. Wiley.)

```
data(fish, package="bayesmix")
hist(fish$fish, freq=FALSE, n=22)
lines(density(fish$fish))
```

Beispiel II


Histogram of fish$fish
Density
0.00
0.05
0.10
0.15
0.20
0.25
0.30
4
6
8
10
12
fish$fish

bayesmix-Paket I

```
library(bayesmix)
model <- BMMmodel(fish, k = 4,
  initialValues = list(S0 = 2),
  priors = list(kind = "independence",
  parameter = "priorsFish",
  hierarchical = "tau"))
```

```
print(model)
```

```
## Data for nodes: b0, B0inv, nu0Half, g0Half, g0G0Half, k
## Initial values for nodes: eta, mu, tau, S0
##
## Model specification in BUGS language:
##
## var
```

bayesmix-Paket II

```
##  b0,
##      B0inv,
##      nu0Half,
##      g0Half,
##      g0G0Half,
##      k,
##      N,
##      eta[4],
##  mu[4],
##  tau[4],
##  nu0S0Half,
##      S0,
##      e[4],
##  y[256],
##  S[256];
##
## model    {
```

bayesmix-Paket III

```
##  for (i in 1:N) {
##      y[i] ~ dnorm(mu[S[i]],tau[S[i]]);
##      S[i] ~ dcat(eta[]);
##  }
##  for (j in 1:k) {
##      mu[j] ~ dnorm(b0,B0inv);
##      tau[j] ~ dgamma(nu0Half,nu0S0Half);
##  }
##  S0 ~ dgamma(g0Half,g0G0Half);
##  nu0S0Half <- nu0Half * S0;
##
##  eta[] ~ ddirch(e[]);
## }
```

JAGS

- **JAGS** oder **Just another Gibbs sampler** ist eine Software zur MCMC-Simulation.
- Definition der Modelle entspricht der bei WinBUGS bzw. OpenBUGS
- Anbindung an R und andere Sprachen

Fortsetzung Beispiel I

```
control <- JAGScontrol(variables = c("mu", "tau", "eta",
    "S"), burn.in = 1000, n.iter = 5000, seed = 10)
```

```
z <- JAGSrun(fish, model = model, control = control)
```

```
## Compiling model graph
##    Declaring variables
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 256
##    Unobserved stochastic nodes: 266
##    Total graph size: 1047
##
## Initializing model
```

Fortsetzung Beispiel II

```
zSort <- Sort(z, by = "mu")
zSort
```

```
##
## Call:
## JAGSrun(y = fish, model = model, control = control)
##
## Markov Chain Monte Carlo (MCMC) output:
## Start = 1001
## End = 6000
## Thinning interval = 1
##
##  Empirical mean, standard deviation and 95% CI for eta
##           Mean      SD    2.5%  97.5%
## eta[1] 0.1189 0.02483 0.07530 0.1731
## eta[2] 0.5065 0.04458 0.41275 0.5863
## eta[3] 0.2746 0.04544 0.19466 0.3742
```

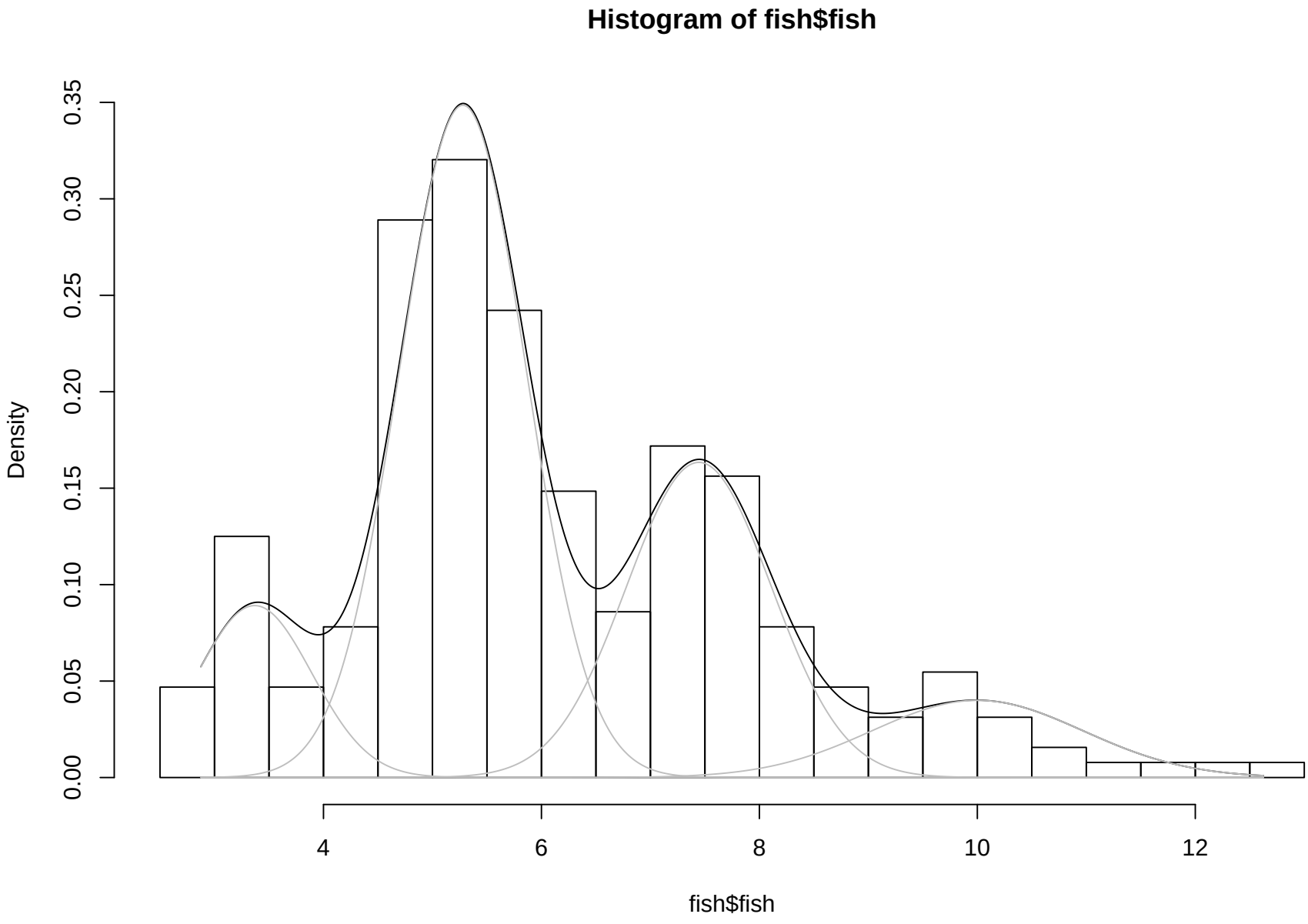
Fortsetzung Beispiel III

```
## eta[4] 0.1000 0.02937 0.05098 0.1658
##
##  Empirical mean, standard deviation and 95% CI for mu
##         Mean      SD  2.5%  97.5%
## mu[1] 3.382 0.15393 3.122  3.729
## mu[2] 5.278 0.07337 5.130  5.416
## mu[3] 7.437 0.14866 7.106  7.699
## mu[4] 9.979 0.39132 9.181 10.696
##
##  Empirical mean, standard deviation and 95% CI for sigma
##              Mean      SD   2.5%  97.5%
## sigma2[1] 0.2996 0.11811 0.1456 0.5876
## sigma2[2] 0.3504 0.08497 0.2357 0.5174
## sigma2[3] 0.4815 0.19657 0.2404 0.9607
## sigma2[4] 1.0023 0.34858 0.5354 1.8441
```

Medianmodell I

```
postmed<-apply(zSort$results, 2, median)
x<-seq(min(fish$fish), max(fish$fish), length=1000)
d1 <- postmed[257]*dnorm(x, postmed[261], sqrt(postmed[265]
d2 <- postmed[258]*dnorm(x, postmed[262], sqrt(postmed[266]
d3 <- postmed[259]*dnorm(x, postmed[263], sqrt(postmed[267]
d4 <- postmed[260]*dnorm(x, postmed[264], sqrt(postmed[268]
d <- d1 + d2 + d3 + d4
hist(fish$fish, freq=FALSE, n=22, ylim=c(0,max(d)))
lines(x, d)
lines(x, d1, col="grey")
lines(x, d2, col="grey")
lines(x, d3, col="grey")
lines(x, d4, col="grey")
```

Medianmodell II



Posteriori-Verteilung der Klassen I

```
S <- apply(zSort$results[,1:256],2,
     bioimagetools::table.n,4, percentage=TRUE)
barplot(S, col=c("red","green","blue","orange"))
```

Posteriori-Verteilung der Klassen II