Modele Bayesowskie w JAGS

Łukasz Czekaj

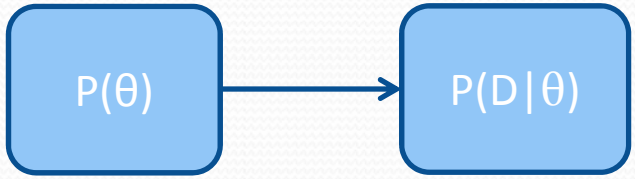
Plan prezentacji

- Modelowanie Bayesowskie
- Zastosowania
- Metody matematyczne
- Narzędzia
- Ocena jakości modeli

Modelowanie Bayesowskie

Budowanie modelu

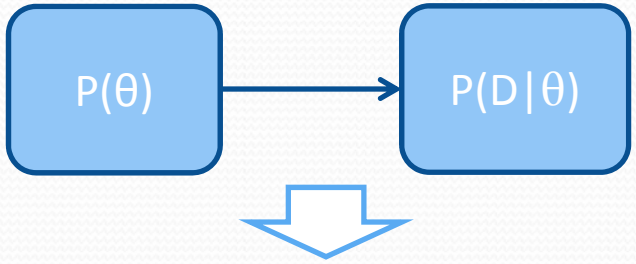
- Rozkłady apriori – hiperparametry
- Model generujący dane – prawdopodobieństwa warunkowe i struktura przyczynowa



Modelowanie Bayesowskie

Budowanie modelu

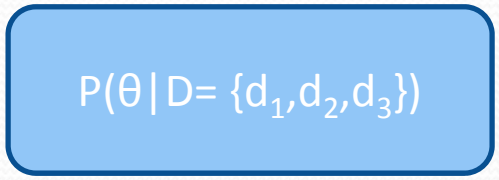
- Rozkłady apriori – hiperparametry
- Model generujący dane – prawdopodobieństwa warunkowe i struktura przyczynowa



- Dane

$D = \{d_1, d_2, d_3\}$

- Rozkład aposteriori

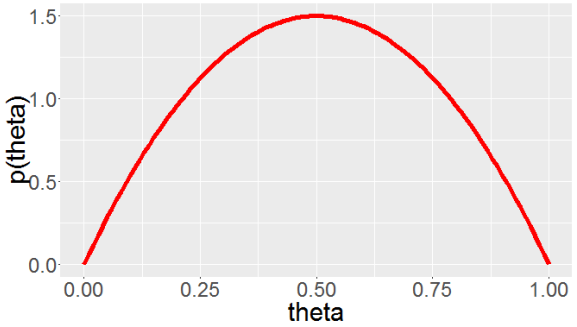


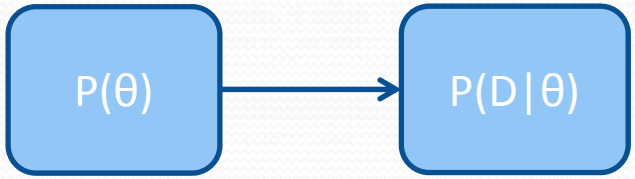
Modelowanie Bayesowski

Twierdzenie Bayesa

$p(\theta) = beta(2,2)$

$p(D=T|\theta) = \theta$

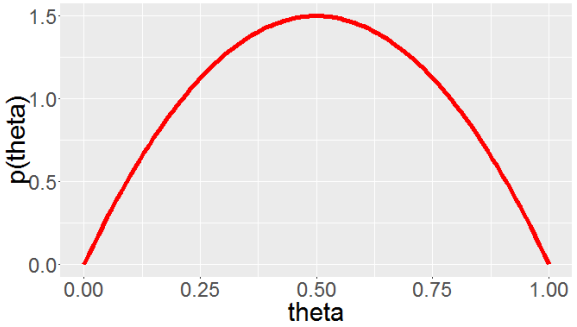


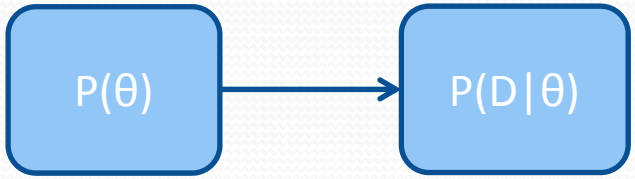


lp	D
1	H
2	H
3	H

Modelowanie Bayesowski

Twierdzenie Bayesa





lp	D
1	H
2	H
3	H

$$P(\theta|D)=p(D|\theta)p(\theta)/P(D)$$

$$p(\theta|D_1=H) \sim (1-\theta)p(\theta)$$

$$\sim \text{beta}(2,3)$$

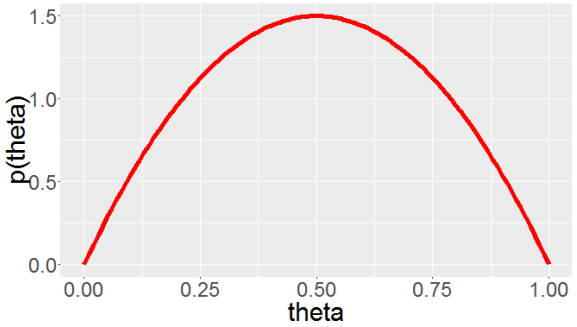
$$p(\theta|D_1=H, D_2=H) \sim (1-\theta)\, p(\theta|D_1=H)$$

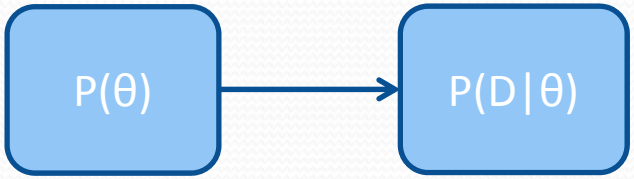
$$\sim (1-\theta)^2\, p(\theta)$$

$$\sim \text{beta}(2,4)$$

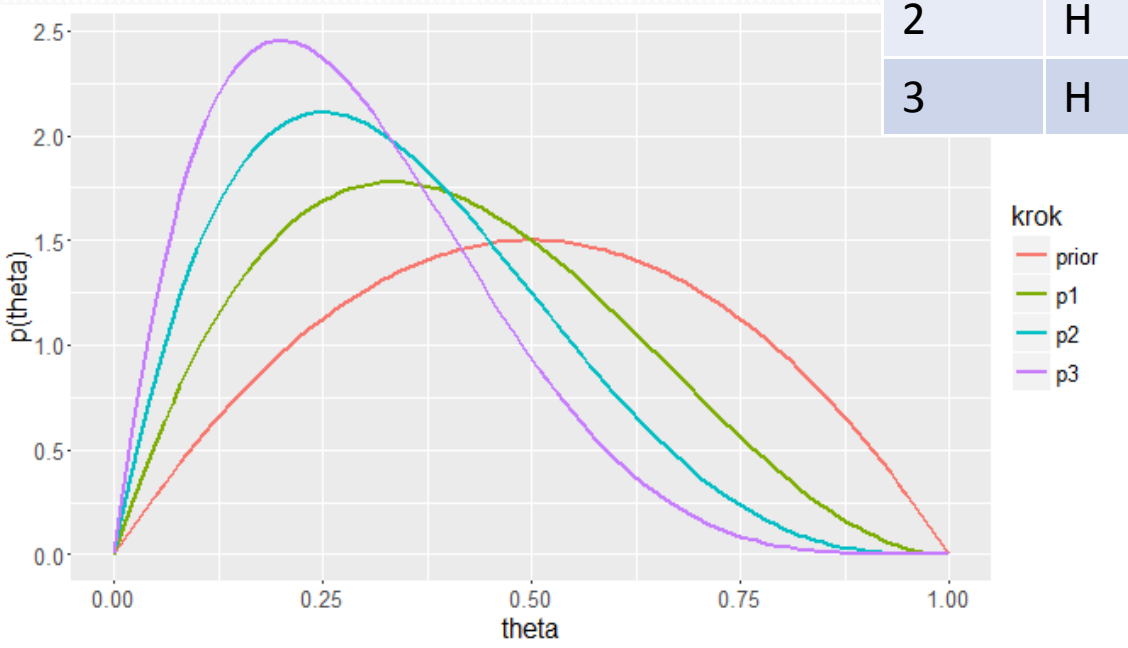
Modelowanie Bayesowski

Twierdzenie Bayesa





lp	D
1	H
2	H
3	H



Modelowanie Bayesowski

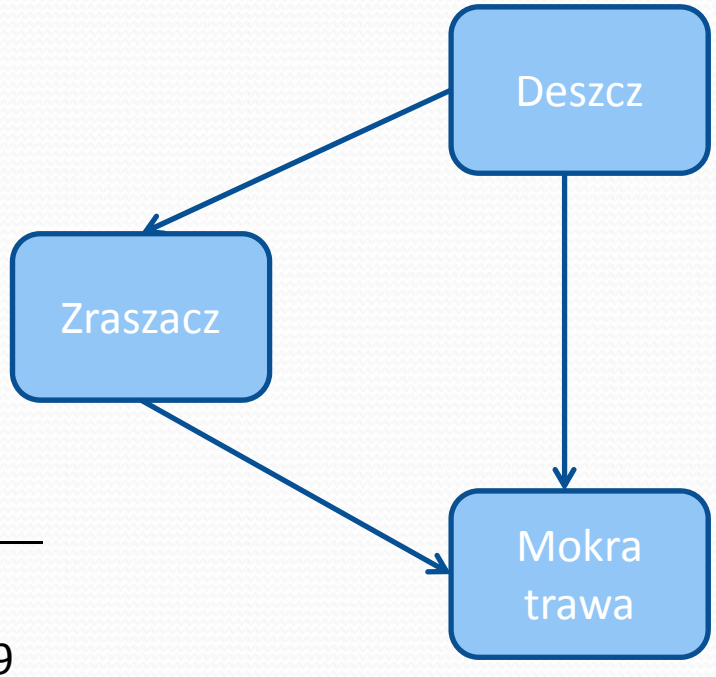
Sieci Bayesowskie

Lp	D	Z	MT
1	T	F	T
2	F	F	F
3	T	F	F
4	F	T	T

Modelowanie Bayesowski

Sieci Bayesowskie

Lp	D	Z	MT
1	T	F	T
2	F	F	F
3	T	F	F
4	F	T	T



	D
F	0.8
T	0.2

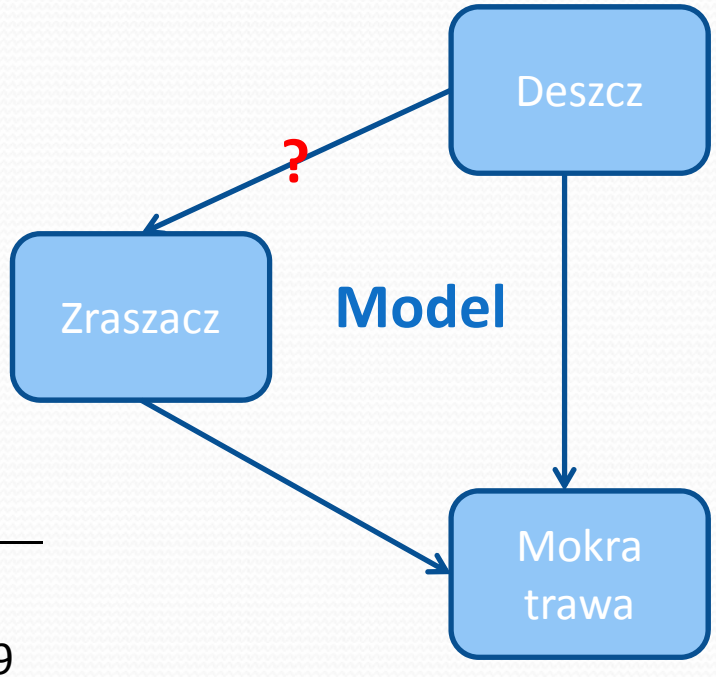
D\Z	T	F
F	0.4	0.6
T	0.01	0.99

Z,D\MT		T	F
F	F	0.0	1.0
F	T	0.8	0.2
T	F	0.9	0.1
T	T	0.99	0.01

Modelowanie Bayesowski

Sieci Bayesowskie (struktura)

Lp	D	Z	MT
1	T	F	T
2	F	F	F
3	T	F	F
4	F	T	T



	D
F	0.8
T	0.2

D\Z	T	F
F	0.4	0.6
T	0.01	0.99

Z,D\MT		T	F
F	F	0.0	1.0
F	T	0.8	0.2
T	F	0.9	0.1
T	T	0.99	0.01

Modelowanie Bayesowski

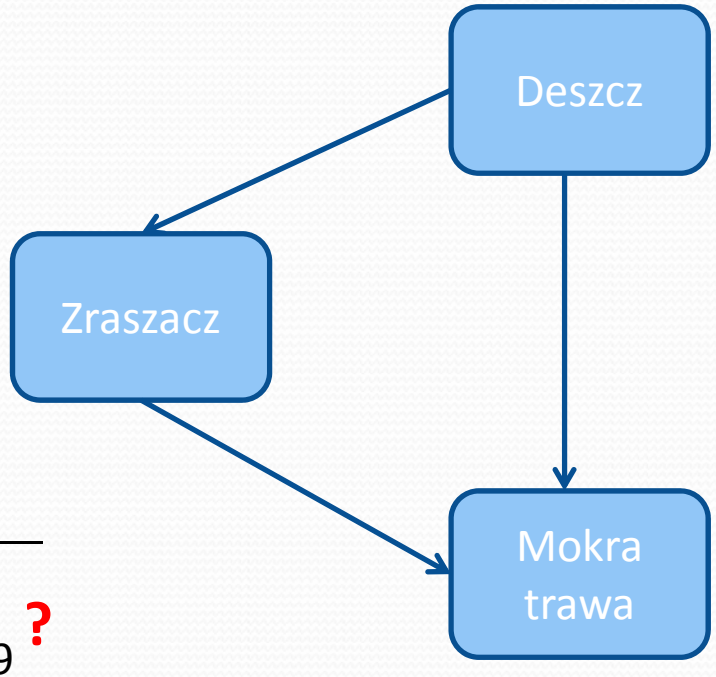
Sieci Bayesowskie (parametry)

Lp	D	Z	MT
1	T	F	T
2	F	F	F
3	T	F	F
4	F	T	T

Prawdopodobieństwa apriori

	D
F	0.8
T	0.2

?



Prawdopodobieństwa warunkowe

D\Z	T	F
F	0.4	0.6
T	0.01	0.99

?

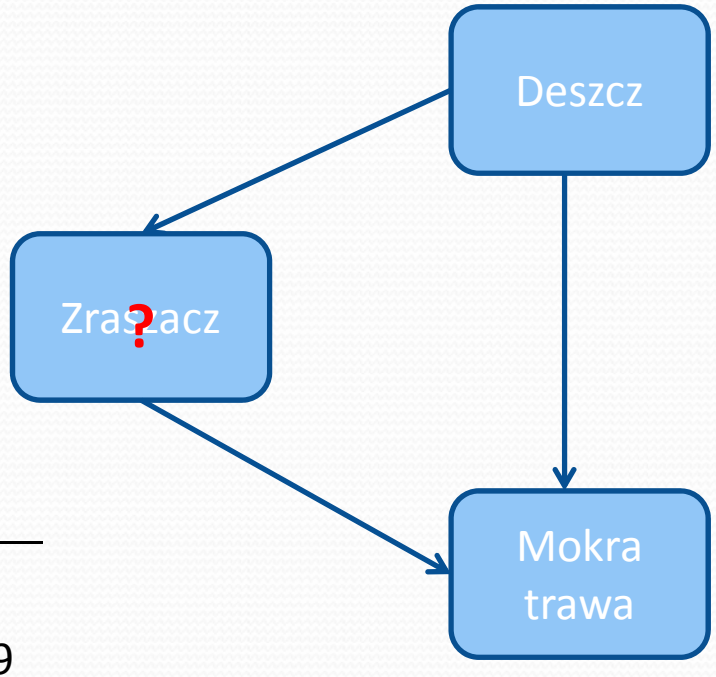
Z,D\MT		T	F
F	F	0.0	1.0
F	T	0.8	0.2
T	F	0.9	0.1
T	T	0.99	0.01

Modelowanie Bayesowski

Sieci Bayesowskie (zmienne ukryte)

Lp	D	Z	MT
1	T	F	T
2	F	F	F
3	T	F	F
4	F	T	T

Obserwowalność



	D
F	0.8
T	0.2

D\Z	T	F
F	0.4	0.6
T	0.01	0.99

Z,D\MT		T	F
F	F	0.0	1.0
F	T	0.8	0.2
T	F	0.9	0.1
T	T	0.99	0.01

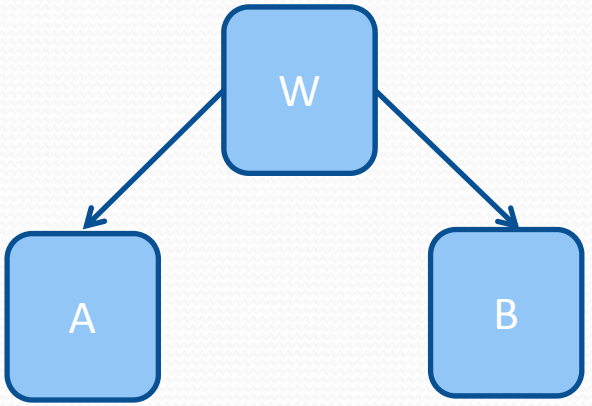
Modelowanie Bayesowski

Definicja i własności

- DAG (skierowany graf acykliczny)
- Prawdopodobieństwo łączne ma własność faktoryzacji

  $p(TM,Z,D) = p(TM|Z,D)\, p(Z|D)p(D)$
- Warunkowa niezależność

  $p(A \,\&\, B|W) = p(A|W)p(B|W)$



Modelowanie Bayesowskie

Rozkłady apriori

- Interpretacja
  - Populacja możliwych parametrów z którego pochodzą obecne wartości
  - Stan wiedzy o parametrach

Modelowanie Bayesowskie

Rozkłady apriori

- Nieinformatwne
  - Reprezentuje brak informacji o parametrze
  - Niezmienniczość względem reparametryzacji (zasada nierozróżnialności)
  - Jeffreys' Prior – „rozkład płaski", problem dla dużych wymiarów, rozkład normalny: $p(\mu, \sigma^2) \sim 1/\sigma^3$
  - Reference Prior – maksymalna odległość między a priori i a posteriori, rozkład normalny: $p(\mu, \sigma^2) \sim 1/\sigma^4$
- Informatywne
  - Kodujemy naszą wiedze z poprzednich doświadczeń, wiedzę dziedzinową

Modelowanie Bayesowski

Problem niemieckich czołgów

„Załóżmy, że przejęto 4 czołgi o numerach seryjnych 10, 256, 202 i 97. Zakładając, że czołgi są numerowane w kolejności w jakiej schodzą z taśmy produkcyjnej, ile jest obecnie czołgów?”

Think Bayes: Bayesian Statistics in Python, Allen B. Downey

https://en.wikipedia.org/wiki/German_tank_problem

Modelowanie Bayesowski

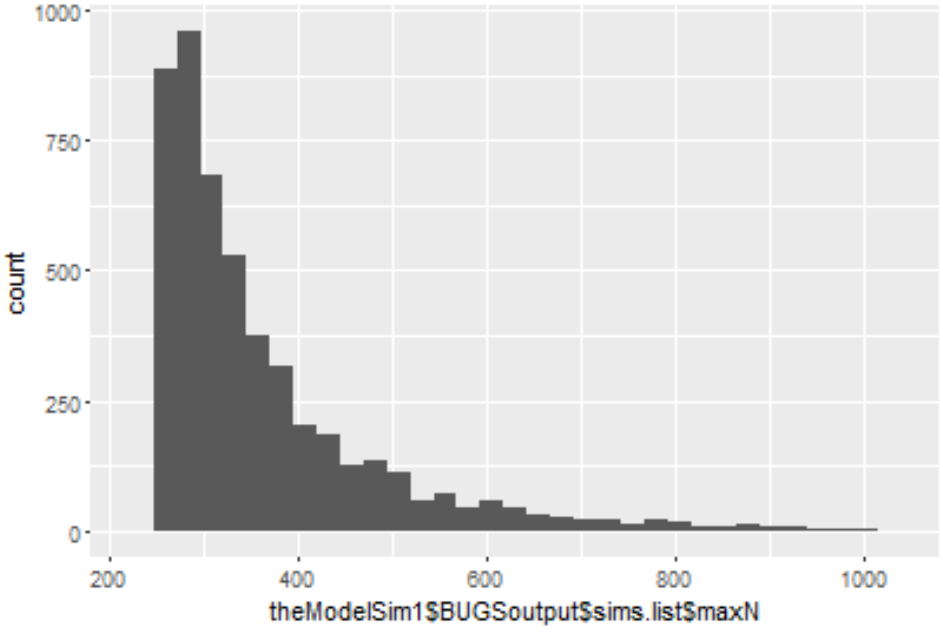
Problem niemieckich czołgów

```
library(R2jags)

observations <- c(10,256,202,97)

model1 <- list(
  model = function()
  {
    maxNTmp ~ dbetabin(1,1,1000-maxY)
    maxN <- maxNTmp+maxY
    for( i in 1:nY)
    {
      Y[i] ~ dbetabin(1,1,maxN)
    }
  },
  data = list(Y = observations,nY = length(observations),maxY=max(observations)),
  parameters = c("maxN","maxNTmp"),
  inits = NULL )

theModelsim1 <- jags(model1$data, model1$inits, model1$parameters, model1$model,
                     n.chains=5, n.iter=100000,
                     jags.module = c("glm","dic","mix"))
```

Modelowanie Bayesowski

Problem niemieckich czołgów

```
Compiling model graph
   Resolving undeclared variables
   Allocating nodes
Graph information:
   Observed stochastic nodes: 4
   Unobserved stochastic nodes: 1
   Total graph size: 36

Initializing model

   |++++++++++++++++++++++++++++++++++++
   |************************************
```

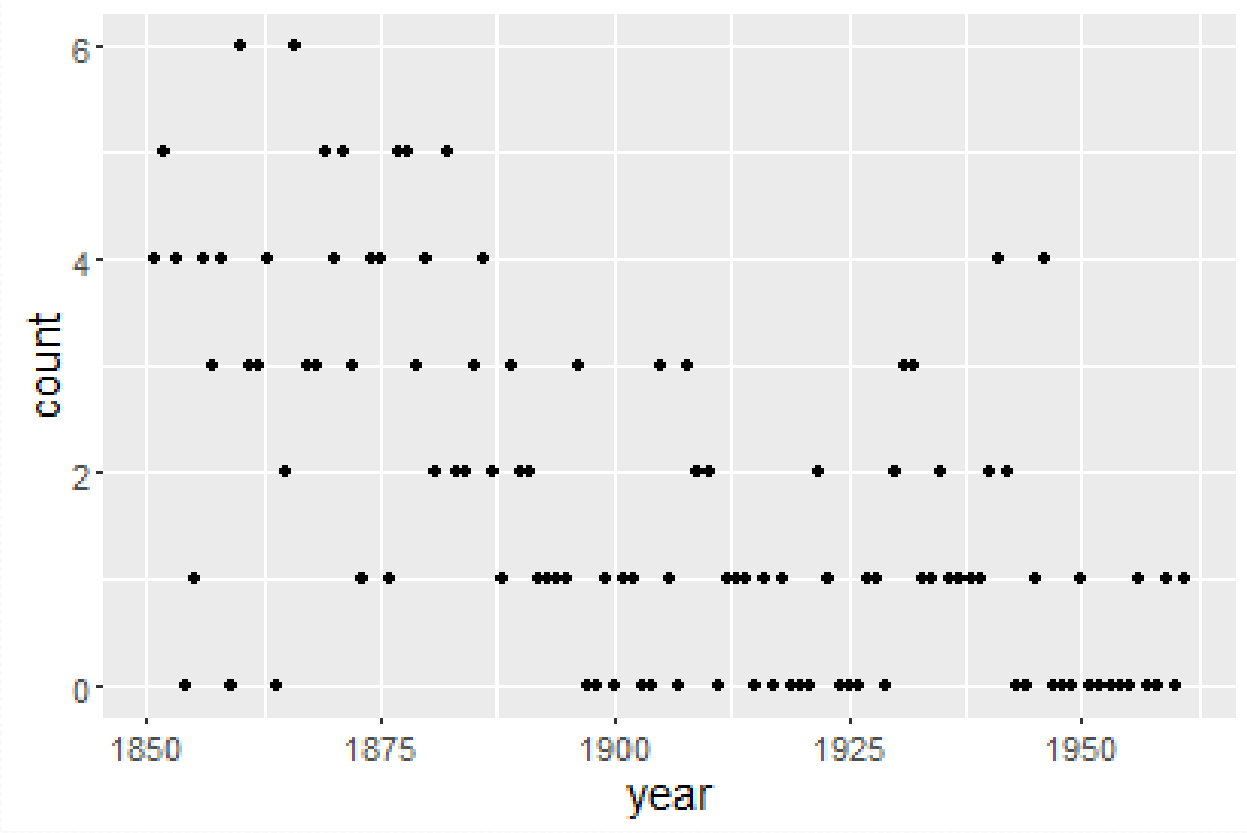


	mu.vect	sd.vect	2.5%	25%	50%	75%	97.5%	Rhat	n.eff
maxN	363.268	121.914	258.000	280.000	320.000	397.000	728.000	1.002	2800
maxNTmp	107.268	121.914	2.000	24.000	64.000	141.000	472.000	1.003	2300
deviance	46.839	2.212	44.455	45.107	46.172	47.892	52.733	1.002	2900

Modelowanie Bayesowski

Punkt przełączenia

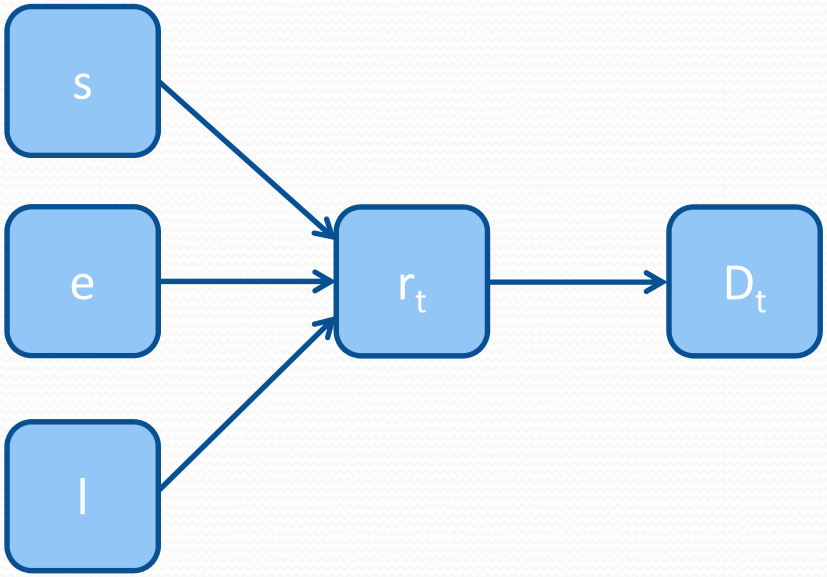
Wypadki w kopalniach w latach 1851-1951



https://pymc-devs.github.io/pymc/tutorial.html

Modelowanie Bayesowski

Punkt przełączenia



$D_t \sim \text{Poisson}(r_t)$

$r_t <- t > s \ ? \ l : e$

$s \sim \text{Dunif}(t_l, t_h)$

$e \sim \text{Exponential}(r_e)$

$l \sim \text{Exponential}(r_l)$

Modelowanie Bayesowski

Punkt przełączenia

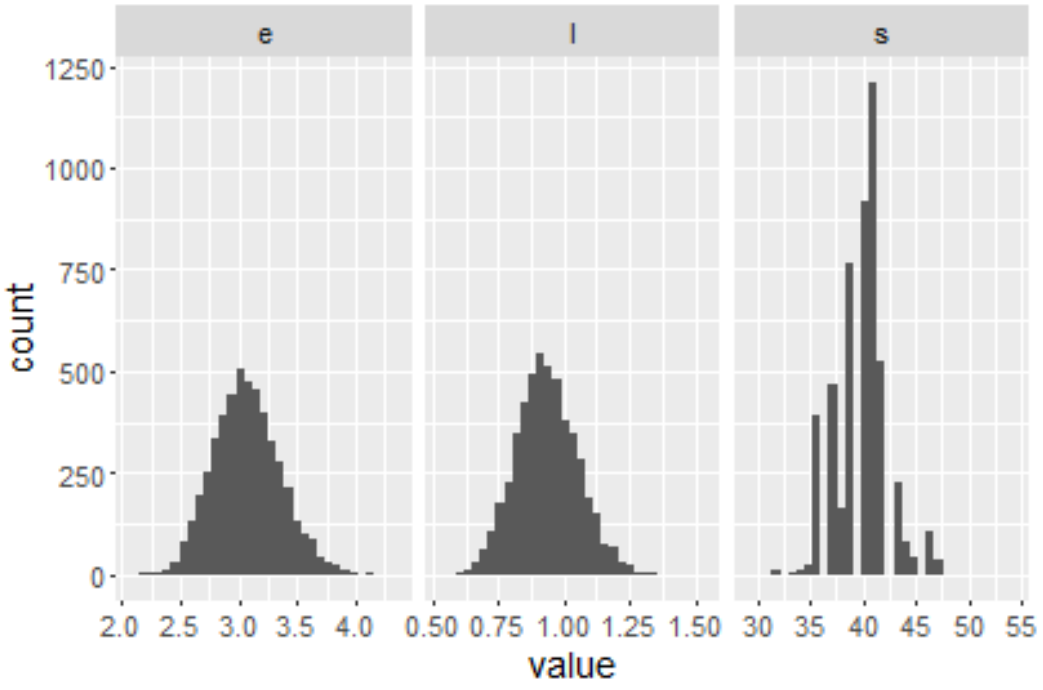
```
dfSP <- data.frame( count =
c(4, 5, 4, 0, 1, 4, 3, 4, 0, 6, 3, 3, 4, 0, 2, 6,
3, 3, 5, 4, 5, 3, 1, 4, 4, 1, 5, 5, 3, 4, 2, 5,
2, 2, 3, 4, 2, 1, 3, 2, 2, 1, 1, 1, 1, 3, 0, 0,
1, 0, 1, 1, 0, 0, 3, 1, 0, 3, 2, 2, 0, 1, 1, 1,
0, 1, 0, 1, 0, 0, 0, 2, 1, 0, 0, 0, 1, 1, 0, 2,
3, 3, 1, 1, 2, 1, 1, 1, 1, 2, 4, 2, 0, 0, 1, 4,
0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1 ),
year = 1851:1961)
```

```
modelSP <- list(
  model = function()
  {
    e ~ dexp(1)
    l ~ dexp(1)
    s ~ dbetabin(1,1,110)

    for( t in 1:length(D))
    {
      r[t] <- ifelse(t>s,l,e)
    }

    for( t in 1:length(D))
    {
      D[t] ~ dpois(r[t])
    }
  },
  data = list(D=dfSP$count),
  parameters = c("e","l","s"),
  inits = NULL )

theModelSP <- jags(modelSP$data, modelSP$inits, modelSP$parameters, modelSP$model,
                   n.chains=5, n.iter=100000,
                   jags.module = c("glm","dic","mix"))
```

Modelowanie Bayesowski

Punkt przełączenia

```
Compiling model graph
   Resolving undeclared variables
   Allocating nodes
Graph information:
   Observed stochastic nodes: 111
   Unobserved stochastic nodes: 3
   Total graph size: 563
```

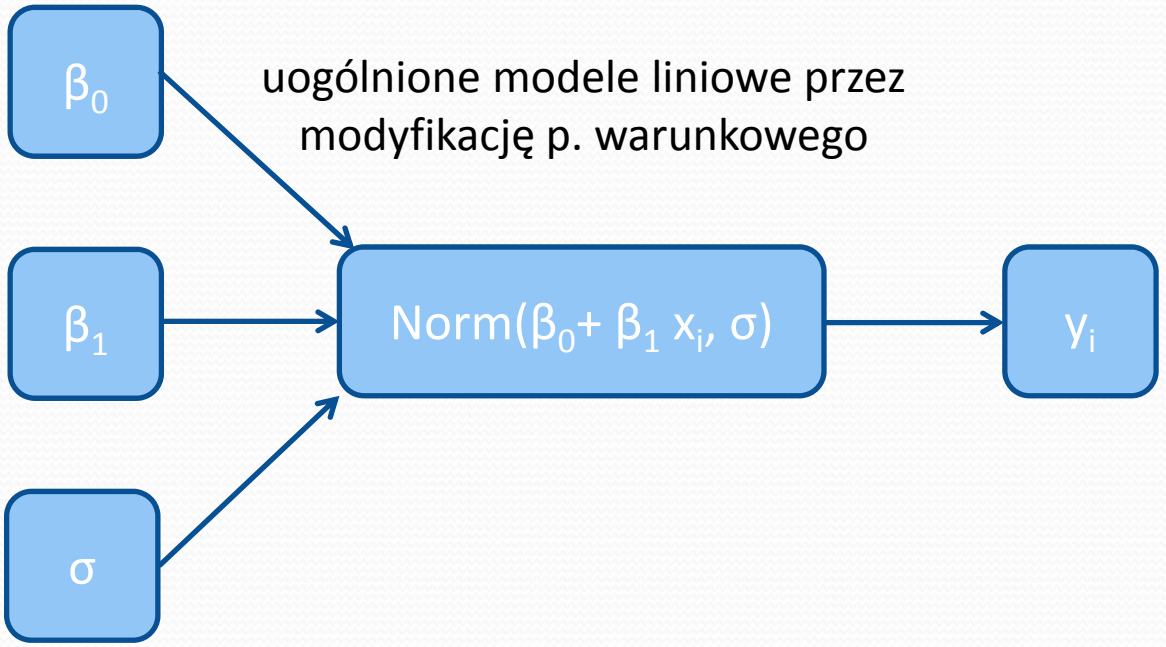


	mu.vect	sd.vect	2.5%	25%	50%	75%	97.5%	Rhat	n.eff
e	3.072	0.286	2.547	2.871	3.062	3.258	3.667	1.001	5000
l	0.937	0.119	0.715	0.855	0.931	1.015	1.187	1.001	5000
s	39.996	2.361	36.000	39.000	40.000	41.000	46.000	1.002	2800
deviance	339.360	2.637	336.092	337.512	338.677	340.577	346.032	1.001	5000

Modelowanie Bayesowski

Regesja liniowa

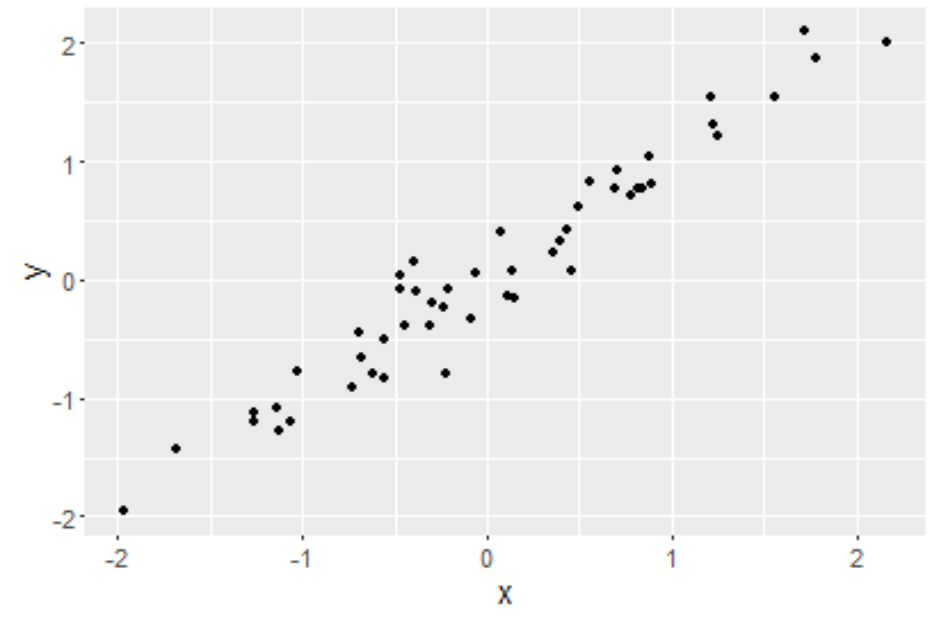
regularyzacja przez
rozkłady apriori



Modelowanie Bayesowski

Regesja liniowa

```
set.seed(123)
df <- data.frame(x = rnorm(50),
                 eps = rnorm(50,sd = 0.25))

df$y = with(df, x+eps)
```



Modelowanie Bayesowski

Regesja liniowa

```
modelNoInf <- list(
  model = function()
  {
    beta0 ~ dnorm(0,beta0Tau)
    beta1 ~ dnorm(0,beta1Tau)

    logSigma ~ dunif( -4 , 4 )
    sigma <- pow( 10, logSigma)
    tau <- pow(sigma,-2)
    #tau ~ dgamma(0.001,0.001)

    for( i in 1:length(y))
    {
      y[i] ~ dnorm(beta0+beta1*x[i],tau)
    }
  },
  data = list(x=df$x, y=df$y, beta0Tau = 1E-8, beta1Tau = 1E-8),
  parameters = c("beta0","beta1","sigma"),
  inits = NULL )

theModelNoInf <- jags(modelNoInf$data, modelNoInf$inits,
                      modelNoInf$parameters, modelNoInf$model,
                    n.chains=5, n.iter=100000,
                    jags.module = c("glm","dic","mix"))
```

Modelowanie Bayesowski

Regesja liniowa

Nieinformatywne rozkłady apriori

	mu.vect	sd.vect	2.5%	25%	50%	75%	97.5%	Rhat	n.eff
beta0	-0.066	0.041	-0.146	-0.092	-0.066	-0.039	0.015	1.001	5000
beta1	1.026	0.038	0.951	1.000	1.026	1.051	1.102	1.001	5000
sigma	0.277	0.029	0.227	0.256	0.274	0.294	0.341	1.001	5000
deviance	12.802	2.603	9.907	10.934	12.116	13.935	19.701	1.001	3900

Informatywne rozkłady apriori dla β (μ=-1, τ=100)

	mu.vect	sd.vect	2.5%	25%	50%	75%	97.5%	Rhat	n.eff
beta0	0.092	0.232	-0.369	-0.066	0.092	0.244	0.561	1.001	5000
beta1	-0.719	0.111	-0.939	-0.792	-0.721	-0.643	-0.502	1.001	4700
sigma	1.625	0.194	1.285	1.492	1.610	1.744	2.045	1.001	5000
deviance	189.662	6.690	176.099	185.236	189.910	194.162	202.533	1.001	5000

Regularyzacja przez rozkłady apriori:

Lasso (laplace priors), elastic net

Modelowanie Bayesowski

Modele hierarchiczne

Modelowanie trwałości pamięci – jak zapominamy z czasem?

- Grupa osób miała do zapamiętania 18 jednostek informacji
- Każda osoba była testowana podczas 10 sesji
- Model wykładniczego zaniku

Table 10.1 Memory retention data for 4 subjects and 10 time intervals.

	Time Interval									
Subject	1	2	4	7	12	21	35	59	99	200
1	18	18	16	13	9	6	4	4	4	?
2	17	13	9	6	4	4	4	4	4	?
3	14	10	6	4	4	4	4	4	4	?
4	?	?	?	?	?	?	?	?	?	?

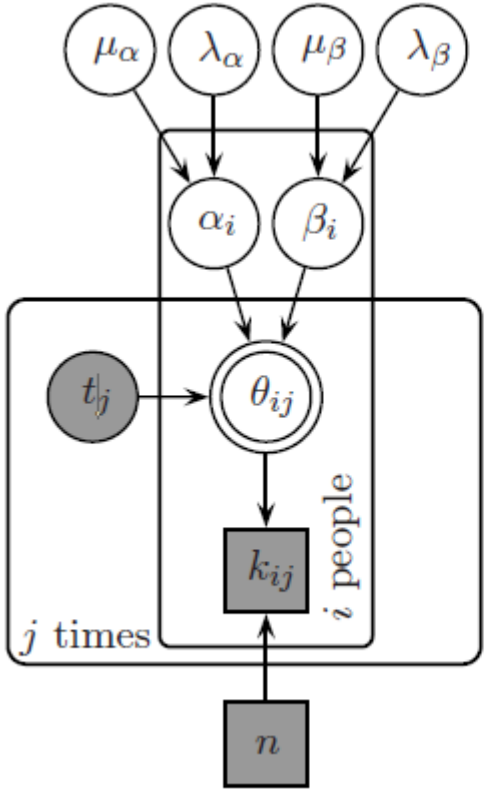
Bayesian Cognitive Modeling: A Practical Course,
Michael D. Lee, Eric-Jan Wagenmakers

Modelowanie Bayesowski

Modele hierarchiczne

współczynniki dla grupy

współczynniki indywidualne



$$
\begin{aligned}
\mu_\alpha &\sim \text{Beta}(1,1) \\
\lambda_\alpha &\sim \text{Gamma}(.001,.001) \\
\mu_\beta &\sim \text{Beta}(1,1) \\
\lambda_\beta &\sim \text{Gamma}(.001,.001) \\
\alpha_i &\sim \text{Gaussian}(\mu_\alpha, \lambda_\alpha)_{\mathcal{I}(0,1)} \\
\beta_i &\sim \text{Gaussian}(\mu_\beta, \lambda_\beta)_{\mathcal{I}(0,1)} \\
\theta_{ij} &\leftarrow \min\left(1, \exp\left(-\alpha_i t_j\right) + \beta_i\right) \\
k_{ij} &\sim \text{Binomial}(\theta_{ij}, n)
\end{aligned}
$$

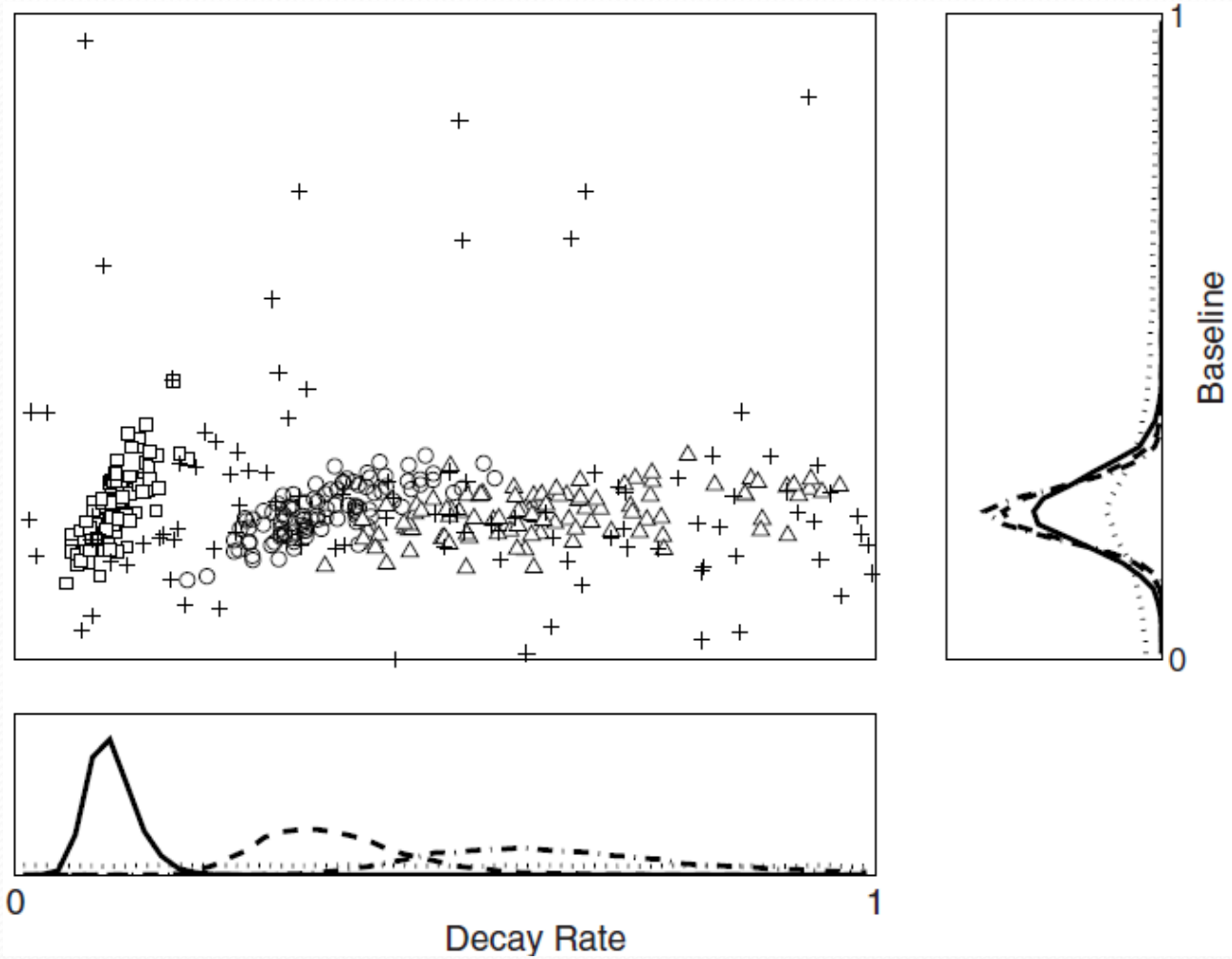
Modelowanie Bayesowski

Modele hierarchiczne

```
# Observed and Predicted Data
# i - subject
# j - test session
for (i in 1:ns){
  for (j in 1:nt){
    k[i,j] ~ dbin(theta[i,j],n)
    predk[i,j] ~ dbin(theta[i,j],n)
  }
}
# Retention Rate At Each Lag For Each Subject Decays Exponentially
for (i in 1:ns){
  for (j in 1:nt){
    theta[i,j] <- min(1,exp(-alpha[i]*t[j])+beta[i])
  }
}
# Parameters For Each Subject Drawn From Gaussian Group Distributions
for (i in 1:ns){
  alpha[i] ~ dnorm(alphamu,alphalambda)T(0,1)
  beta[i] ~ dnorm(betamu,betalambda)T(0,1)
}
# Priors For Group Distributions
alphamu ~ dbeta(1,1)
alphalambda ~ dgamma(.001,.001)T(.001,)
alphasigma <- 1/sqrt(alphalambda)
betamu ~ dbeta(1,1)
betalambda ~ dgamma(.001,.001)T(.001,)
betasigma <- 1/sqrt(betalambda)
```

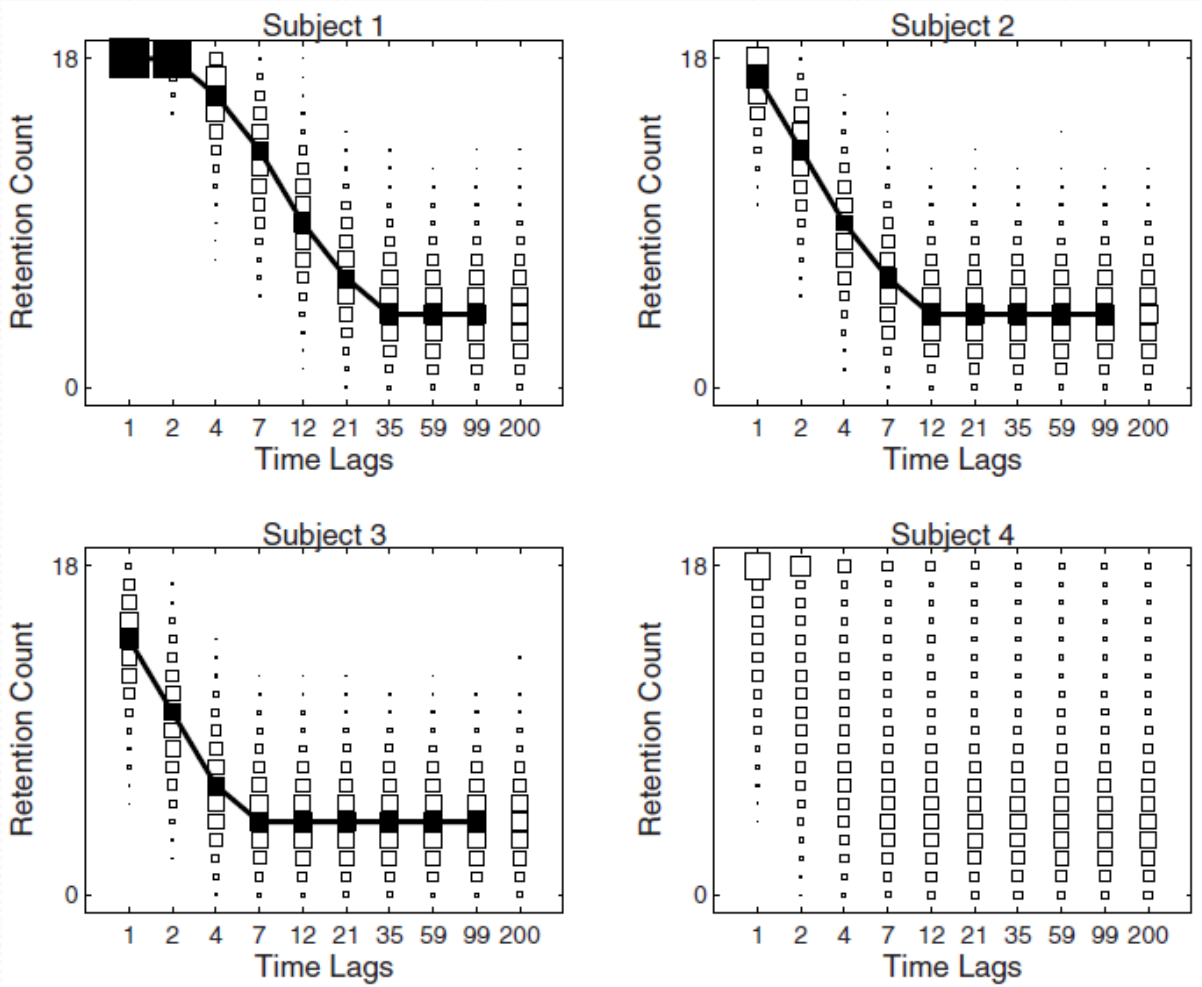
Modelowanie Bayesowski

Modele hierarchiczne



Modelowanie Bayesowski

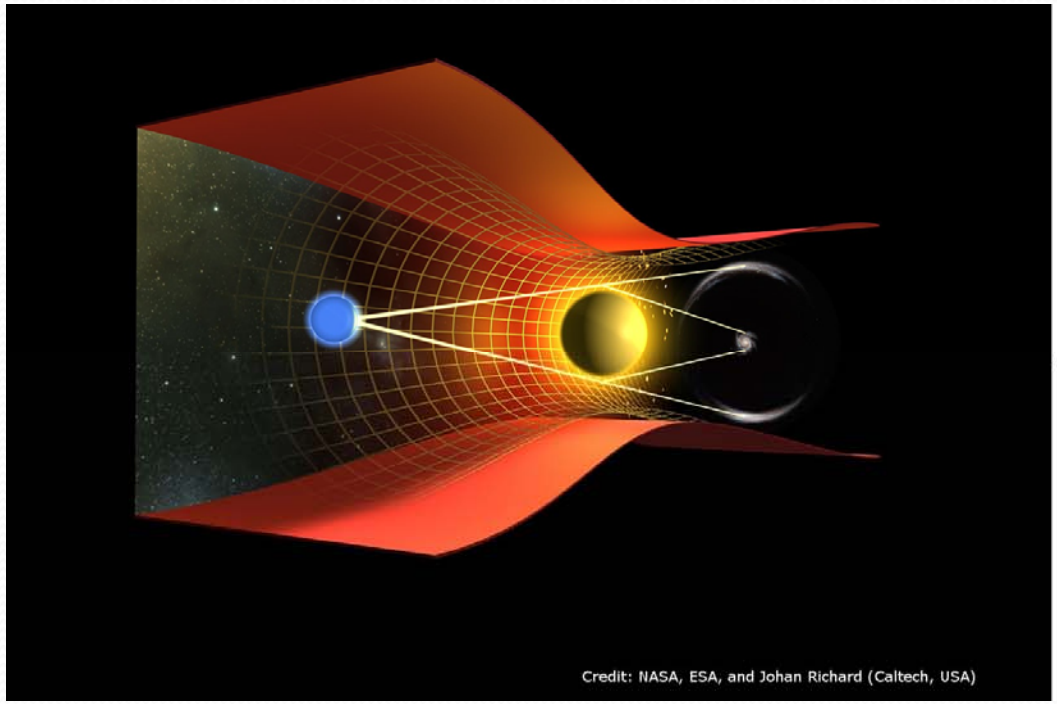
Modele hierarchiczne

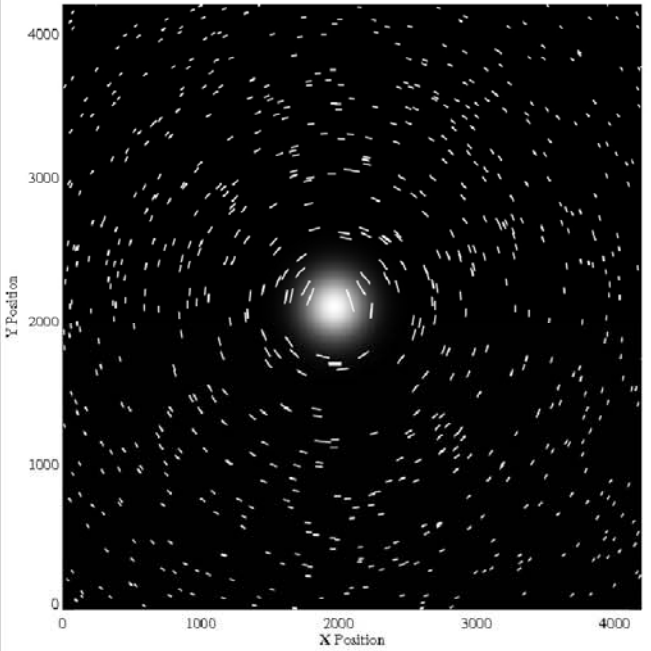


Zastosowania

Wykrywanie halo ciemnej materii

- http://timsalimans.com/observing-dark-worlds/
  https://arxiv.org/pdf/1306.0202.pdf

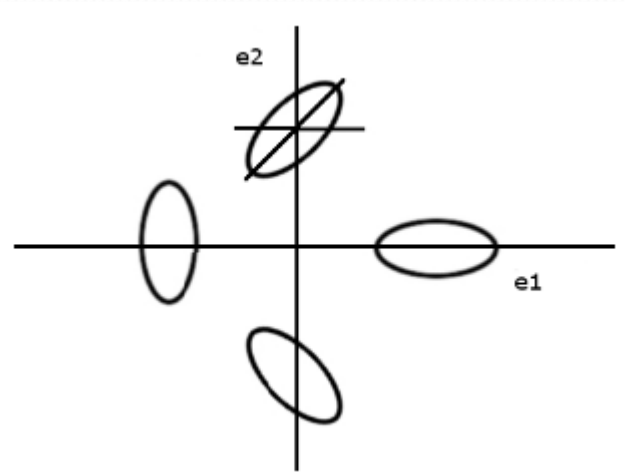




Zastosowania

Wyktywanie halo ciemnej materii

- Dane: 300-740 galaktyk na zdjęcie dla każdej galaktyki (x,y,e1,e2)
- Na każdym zdjęciu 1-3 halo; określić masę i położenie halo $P(\text{loc}_h, \text{mass}_h | \text{loc}_g, e_g)$
- Model generujący dane:



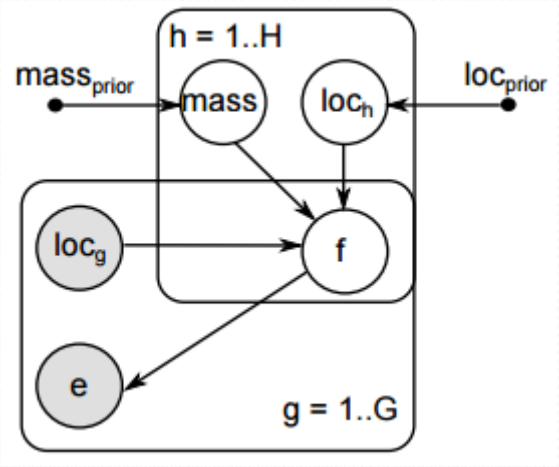
$$p(e_g|\text{loc}_g, \text{loc}_h, \text{mass}_h) = N(e_g|f(\text{loc}_g, \text{loc}_h, \text{mass}_h), \sigma^2)$$

$$f(\text{loc}_g, \text{loc}_h, \text{mass}_h) = \frac{\text{mass}_h}{||\text{loc}_g - \text{loc}_h||}$$

$$e_1 \sim N(-f\cos(2\phi), \sigma^2),\ e_2 \sim N(-f\sin(2\phi), \sigma^2),\ \phi = \text{atan}\frac{y_g - y_h}{x_g - x_h}$$

Zastosowania

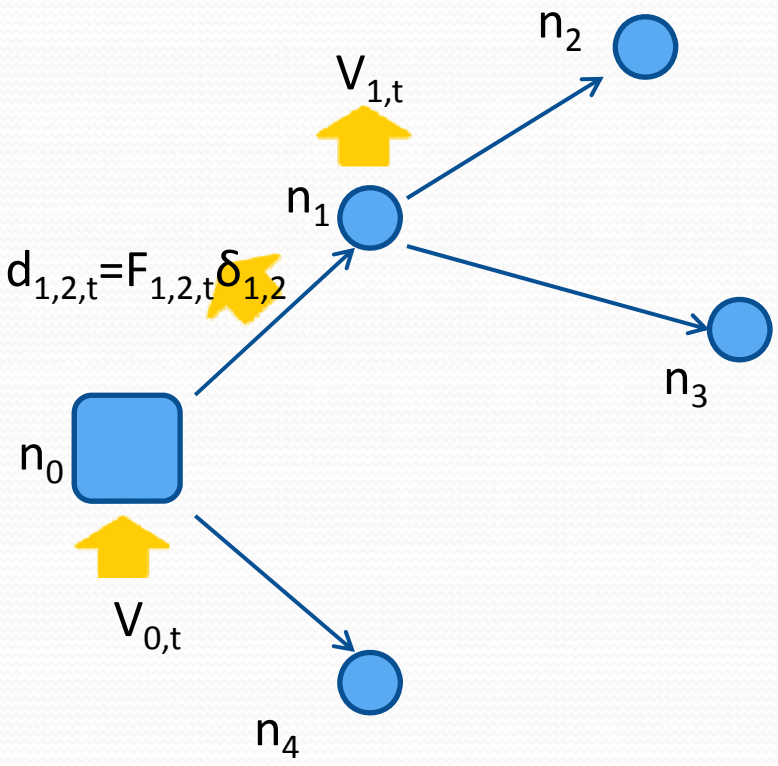
Wyktywanie halo ciemnej materii




```
model{
  for( i in 1 : G ) {
    for( h in 1 : H ) {
      dx[i , h]    <- gx[i] - loc[h , 1]
      dy[i , h]    <- gy[i] - loc[h , 2]
      dist[i , h]  <- sqrt(dx[i , h] * dx[i , h] +
                           dy[i , h] * dy[i , h])
      phi[i , h]   <- atan2(dy[i , h], dx[i , h])
      iDist[i , h] <- 1.0 / dist[i , h]
      f[i , h]     <- mass[h] * iDist[i , h]
      f1[i , h]    <- -f[i , h] * cos(2 * phi[i , h])
      f2[i , h]    <- -f[i , h] * sin(2 * phi[i , h])
    }
    mu1[i] <- sum(f1[i , ])
    mu2[i] <- sum(f2[i , ])
    e1[i] ~ dnorm(mu1[i], 0.05)
    e2[i] ~ dnorm(mu2[i], 0.05)
  }
  for( h in 1 : H ) {
    mass[h]    ~ dgamma(0.001, 0.001)
    loc[h , 1] ~ dunif(0, 4200)
    loc[h , 2] ~ dunif(0, 4200)
  }
}
```

Zastosowania

Analiza strat w sieci transportowej



Wz\sg	0,1	1,2	1,3	0,4
n_0	-	0	0	-
n_1	+	-	-	0

Dane:

V,A, niektóre f

Równanie bilansu dla węzła:

$V+d=Af$

Straty w węźle (proporcjonalne do przepływu):

$d = A^{+}(\delta.f)$

Założenia:

$f > 0,\ 1 > \delta > 0,\ \delta$ stałe w t

Zastosowania

Analiza strat w sieci transportowej

```
for( l in 1:nLinks )
{
    delta[l] ~ dbeta(aLinks,bLinks)
}

for( t in 1:nCases )
{
  for( l in 1:nLinks )
  {
    f[t,l] ~ dgamma(sLinks,rLinks)
    df[t,l] <- delta[l]*f[t,l]
  }

  fn[t,] <- A%*%f[t,]
  dn[t,] <- Aplus%*%df[t,]

  for( n in 1:nNodes )
  {
    v[t,n] ~ dnorm( fn[t,n] + dn[t,n], tauV)
  }
}
```

Kiedy stosować modele Bayesowskie?

- Dysponujemy wiedzą o rozkładzie hiperparametrów, którą chcemy uwzględnić w modelu (np. współczynniki w regresji liniowej mogą przyjmować tylko wartości dodatnie, wyniki wcześniejszych eksperymentów)
- Złożona „fizyka” procesu; łatwo uwzględnić w modelu złożone transformacje, przekształcenia matematyczne, zależności hierarchiczne, zmienność w czasie (np. modele dynamiczne)
- Model korzysta z niestandardowych rozkładów prawdopodobieństwa
- Interesuje nas ocena niepewności otrzymanych wyników (np. rankingowanie, ocena ryzyka); rozkład prawdopodobieństwa vs przedział ufności; https://en.wikipedia.org/wiki/Credible_interval

Wyznaczanie rozkładów aposteriori

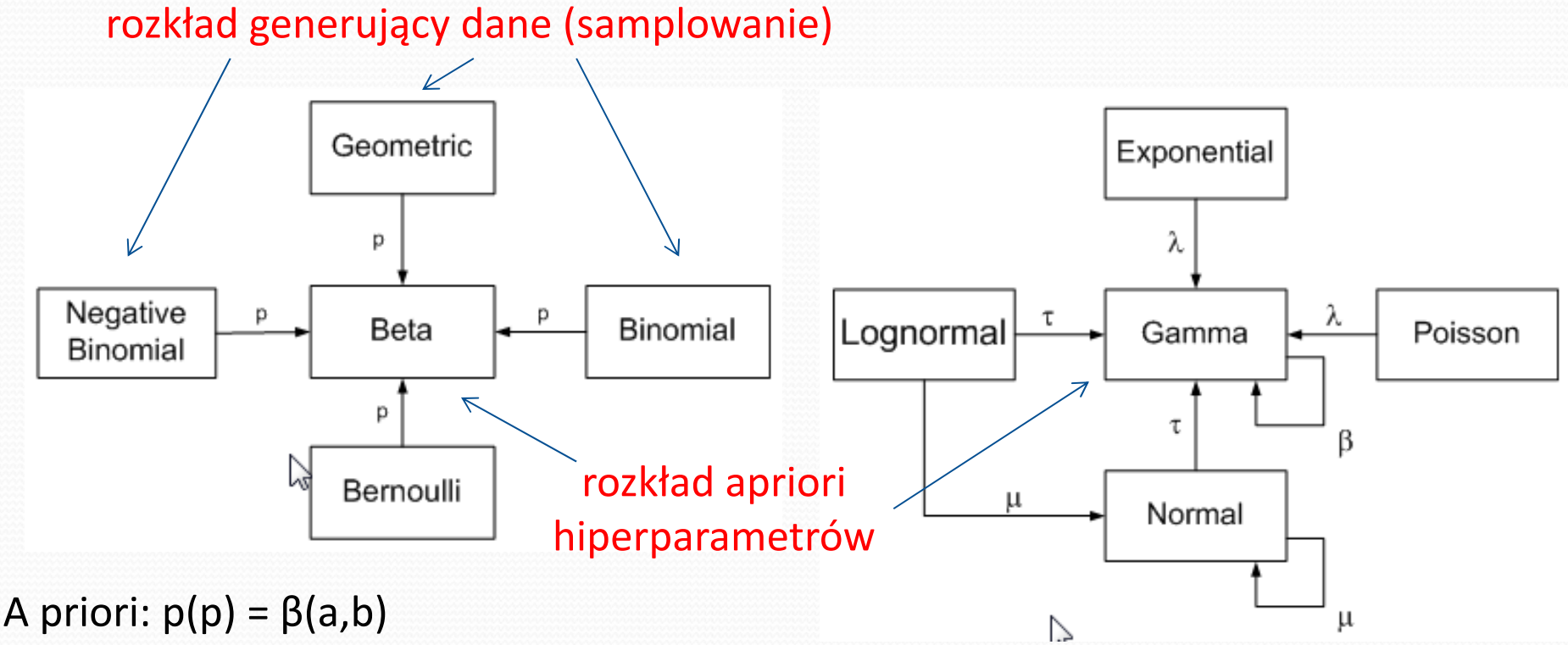
- Sprzężone rozkłady apriori
- Metody wariacyjne
- Samplowanie z rozkładów aposteriori $P(\theta|\{d_i\})$ (monte carlo markov chain)
  - Samplowane Gibbsa
  - Samplowanie Metropolis-Hastings
  - Samplowanie Hamiltonowskie

Information Theory, Inference and Learning Algorithms, David J. C. MacKay

Wyznaczanie rozkładów aposteriori

Sprzężone rozkłady apriori

- Metoda analityczna



A priori: $p(p) = \beta(a,b)$

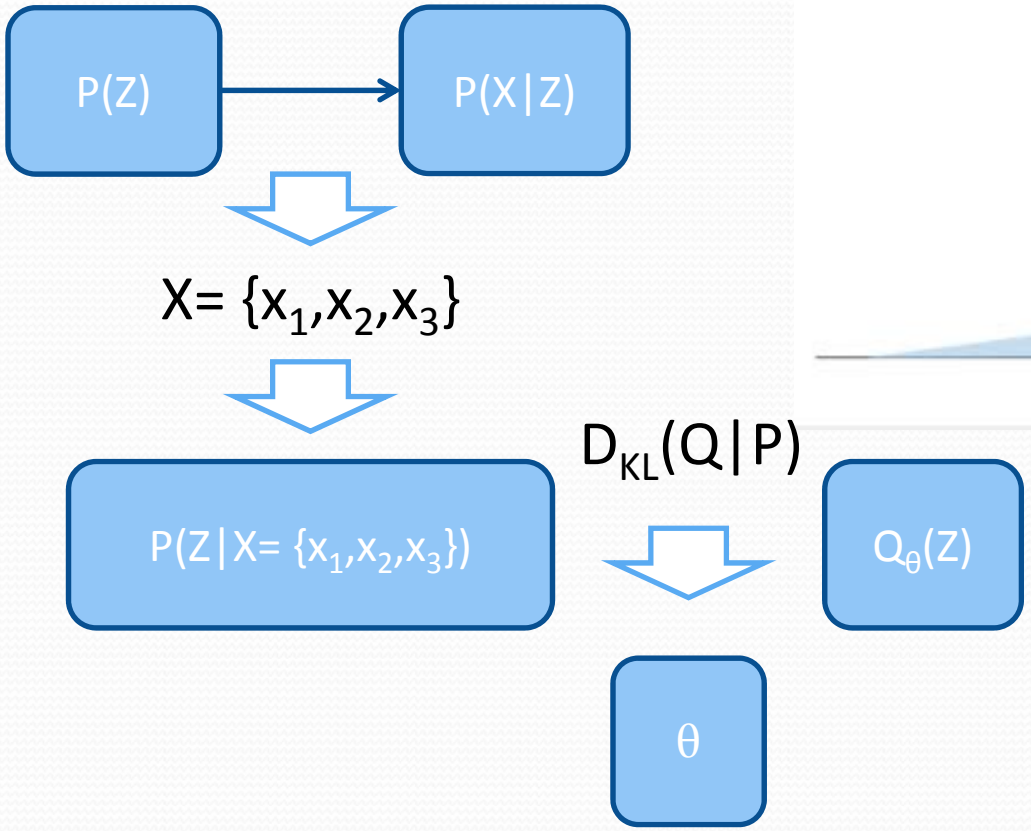
Rozkład generujący: $p(x|n,p) = \text{Bernouli}(x,n,p)$

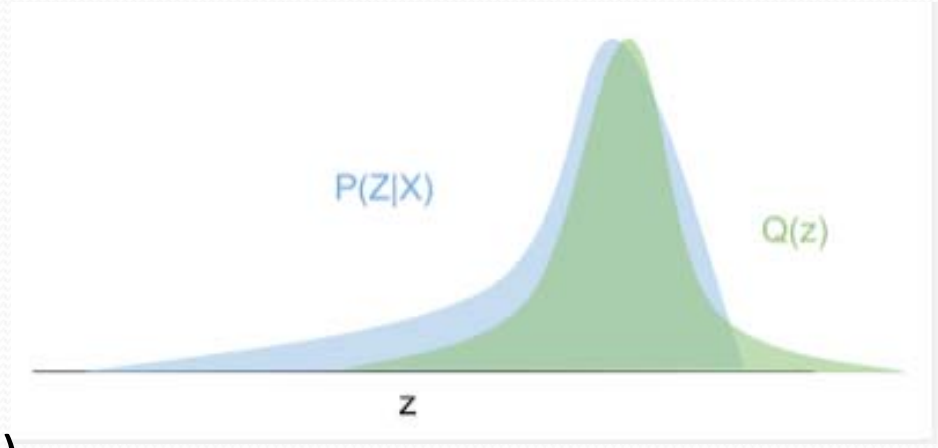
A posteriori: $p(p|x,n) = \beta(a+x,b+n-x)$

Wyznaczanie rozkładów aposteriori

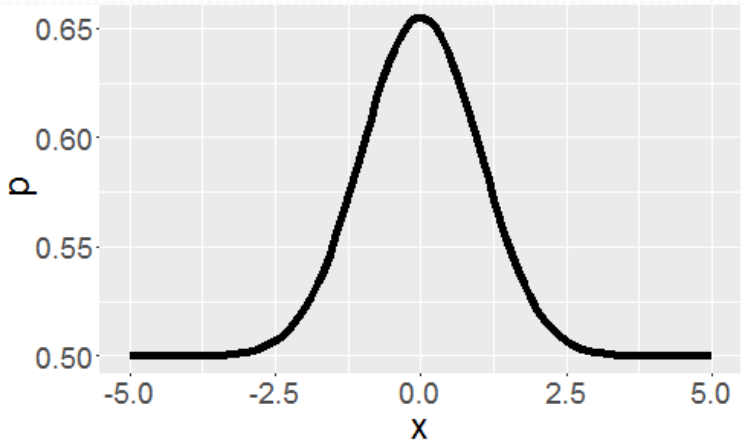
Metody wariacyjne

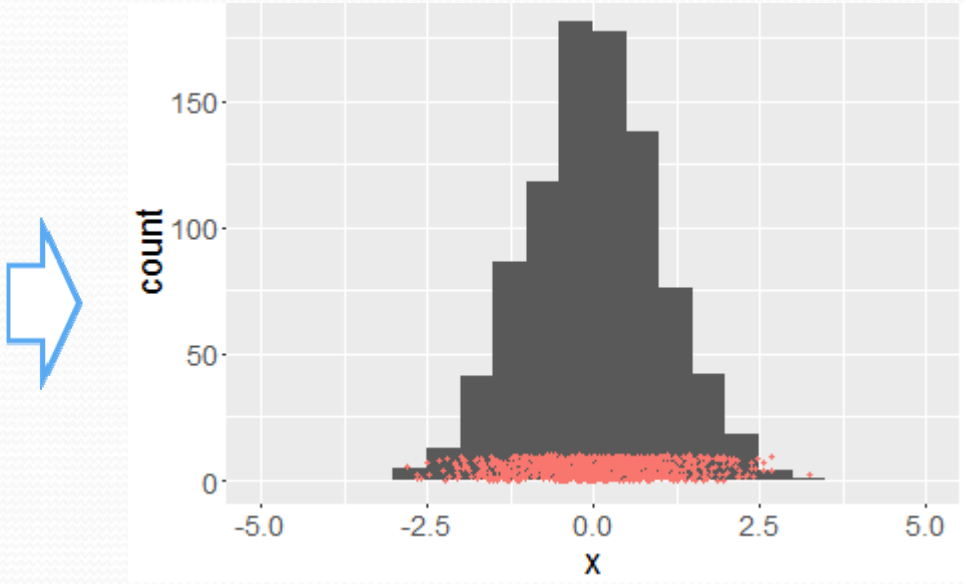
- Aproksymacja analityczna
  faktoryzacja rozkładów hiperametrów





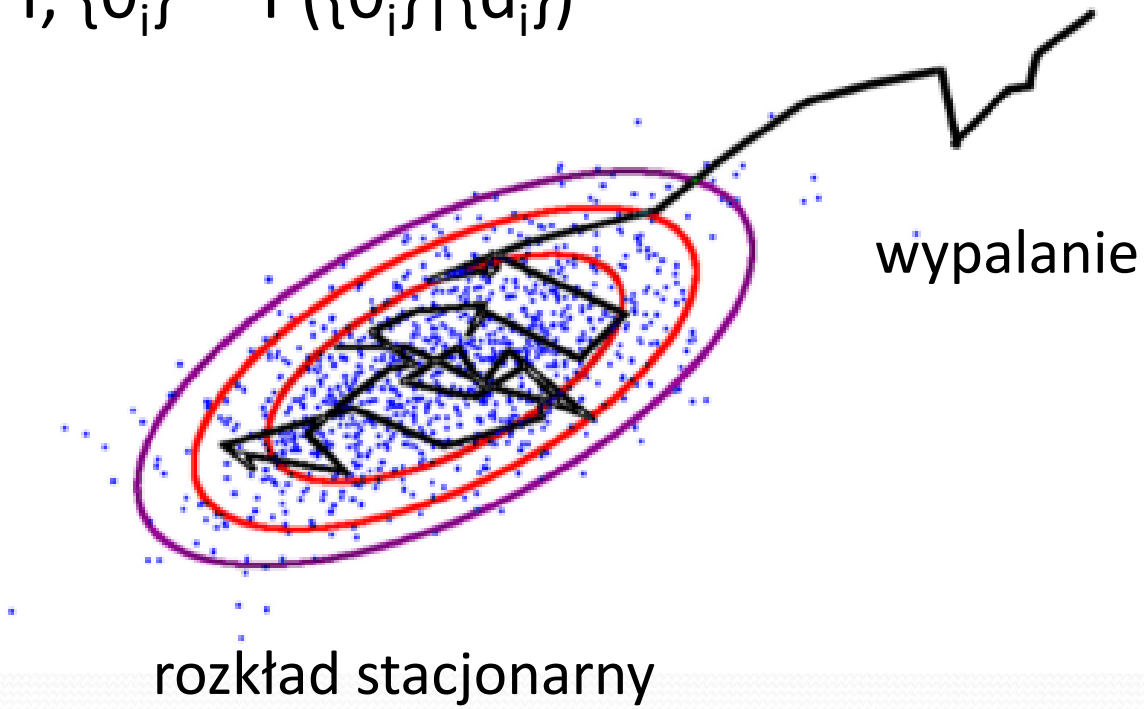
Wyznaczanie rozkładów aposteriori MCMC





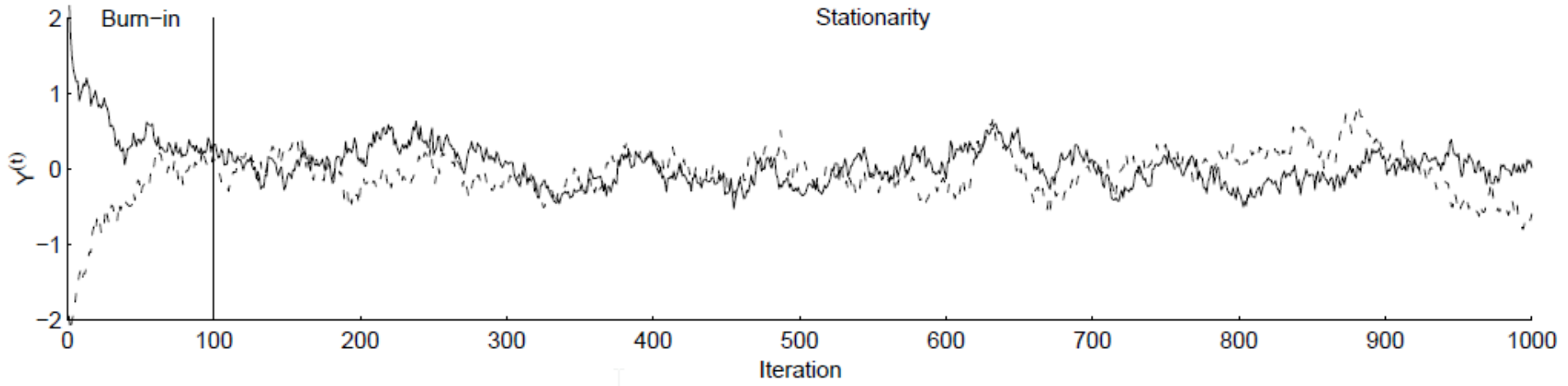
Wyznaczanie rozkładów aposteriori MCMC

- $P(\{\theta_i\}|\{d_i\}) \sim P(\{\theta_i\},\{d_i\})$
- $\{\theta_i\}^{t+1} = T(\{\theta_i\}^t)$
- Dla $t > T$, $\{\theta_i\}^t \sim P(\{\theta_i\}|\{d_i\})$



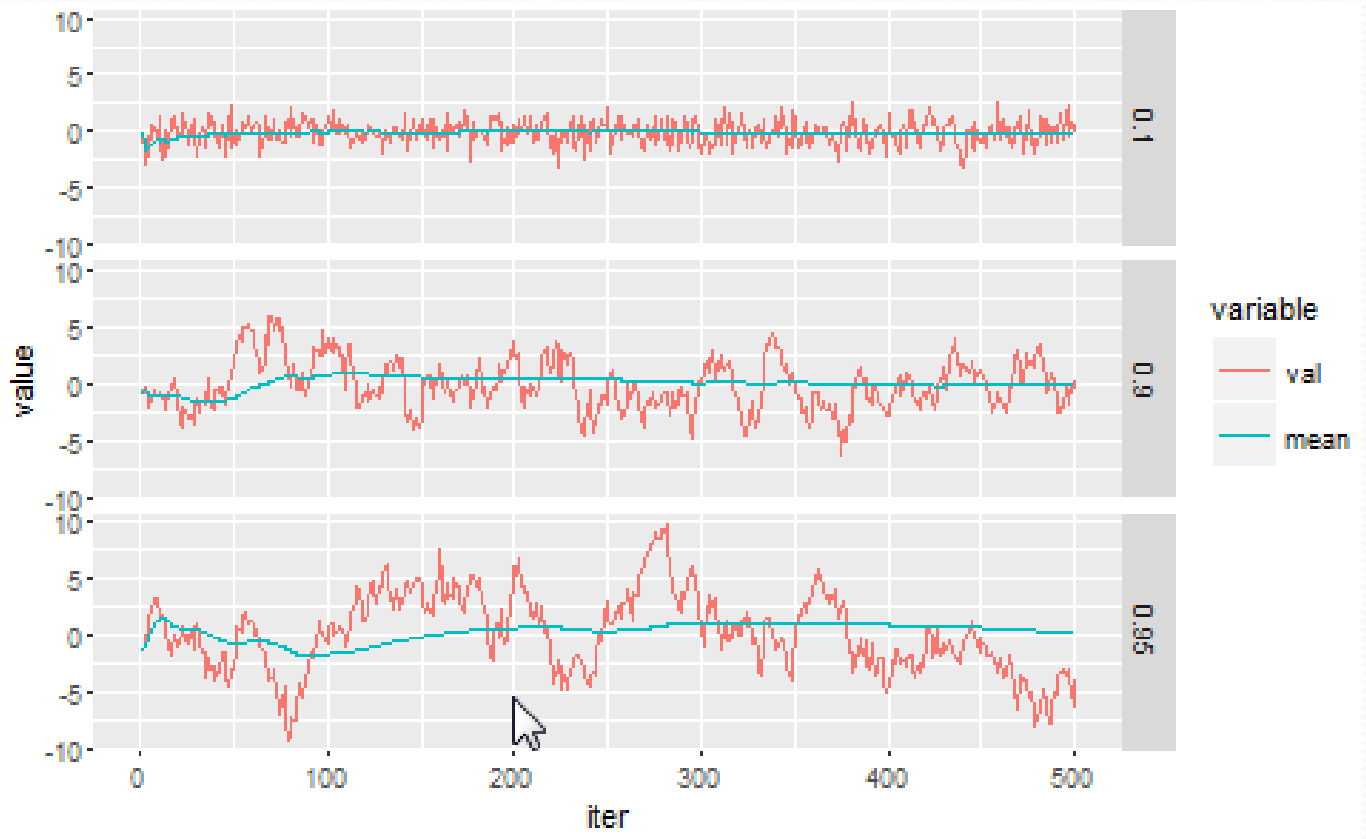
Wyznaczanie rozkładów aposteriori MCMC

- Wypalanie, rozkład stacjonarny, zbieżność



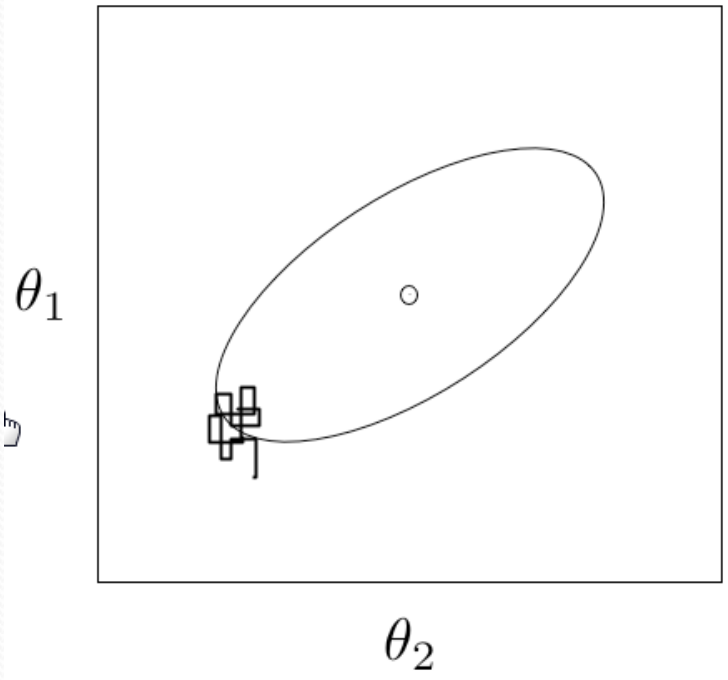
Wyznaczanie rozkładów aposteriori MCMC

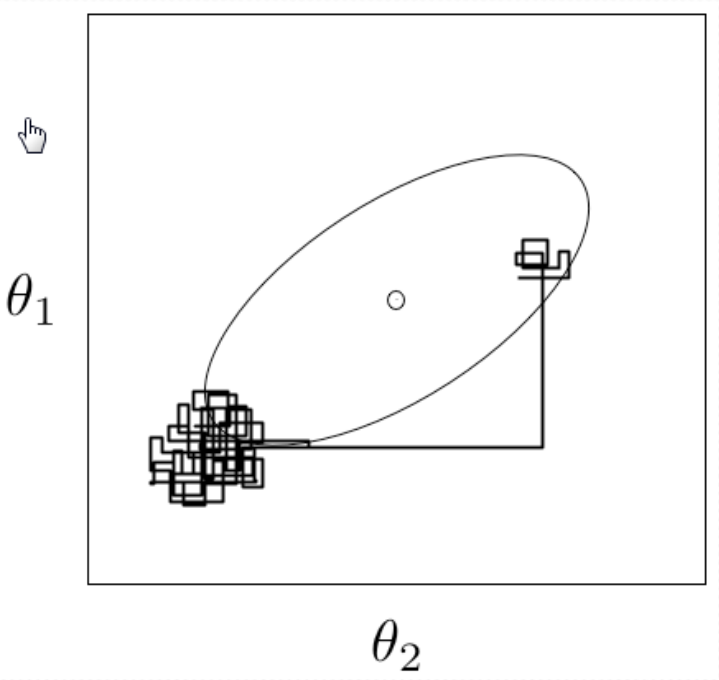
- Autokorelacje (thinning)



Wyznaczanie rozkładów aposteriori MCMC

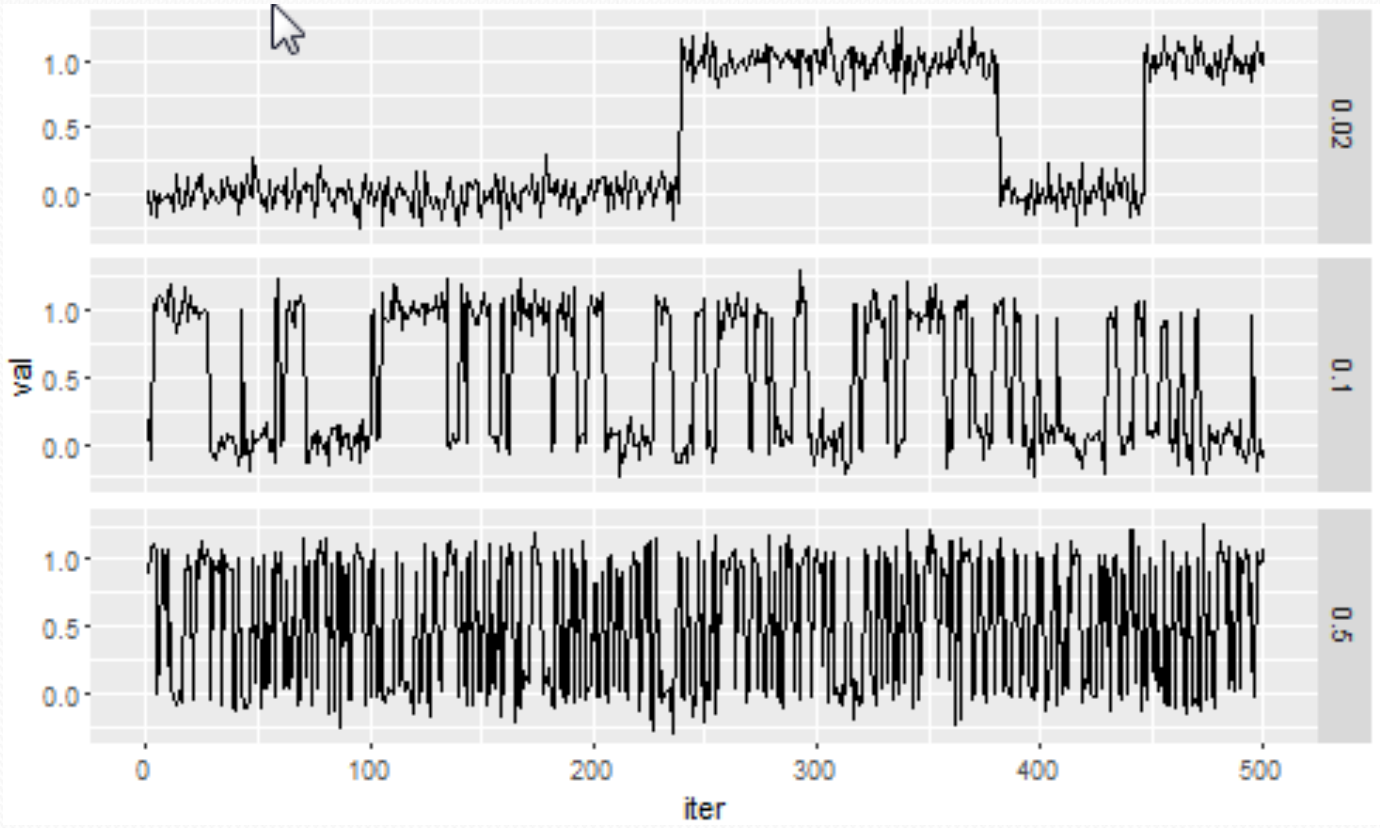
- Mieszanie i bistabilność





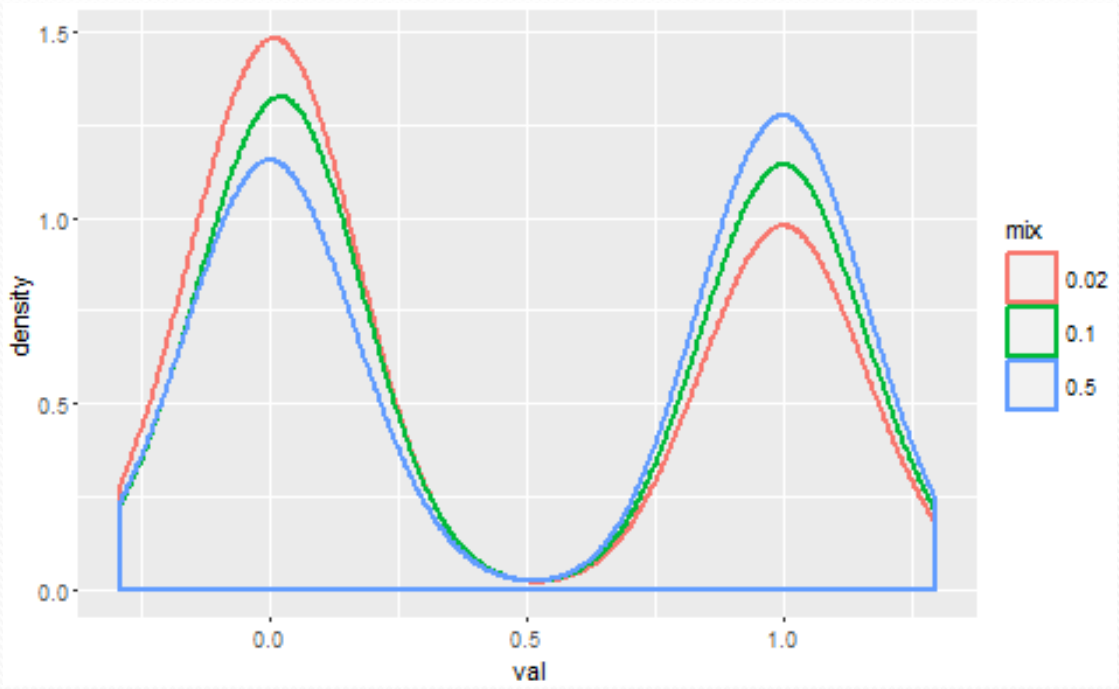
Wyznaczanie rozkładów aposteriori MCMC

- Mieszanie i bistabilność



Wyznaczanie rozkładów aposteriori MCMC

- Mieszanie i bistabilność



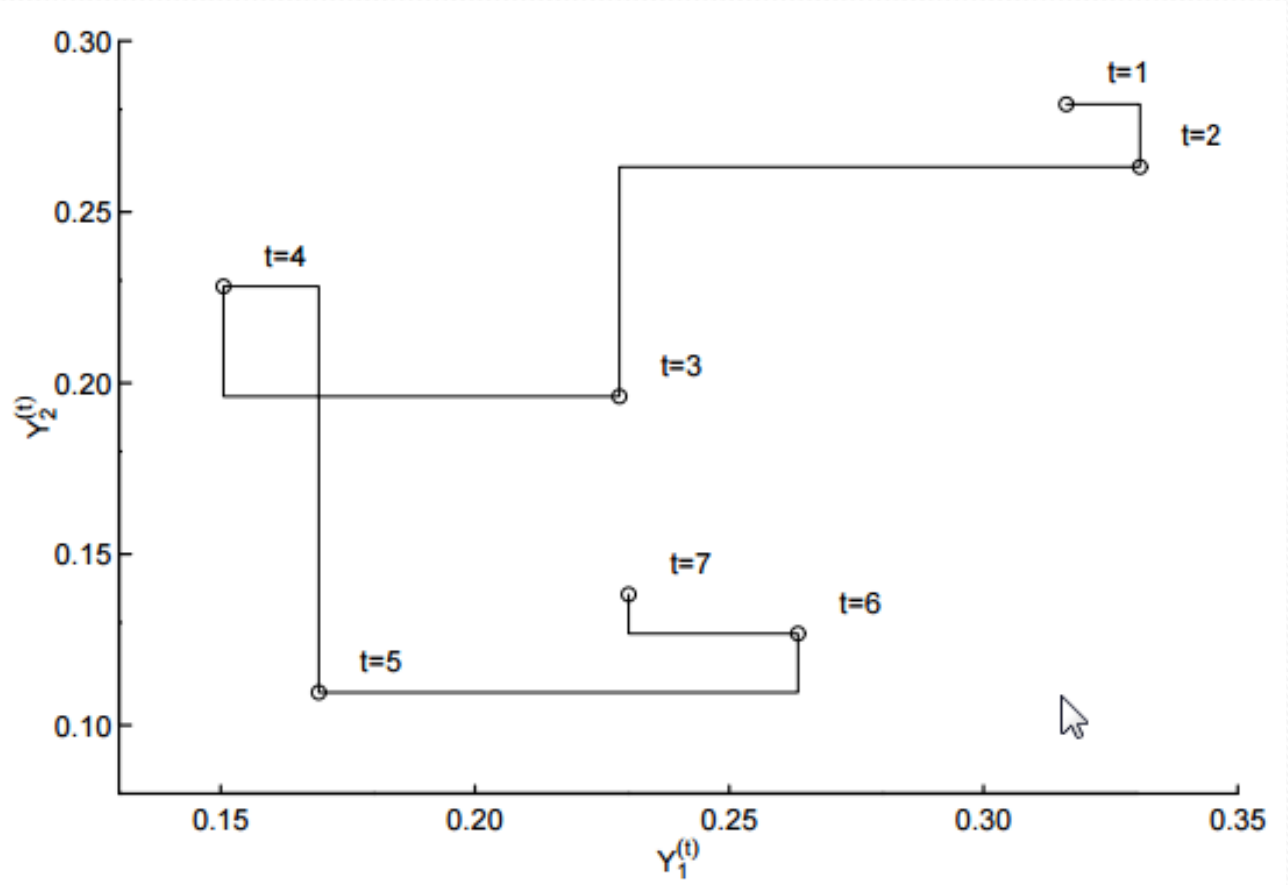
Wyznaczanie rozkładów aposteriori

Samplowanie Gibbsa

- $P(\theta_1, \theta_2 | \{d_i\}) = P(\theta_1 | \theta_2, \{d_i\}) P(\theta_2 | \{d_i\})$
- Ustalamy wartości początkowe i θ_1^0, θ_2^0, samplujemy naprzemiennie:
  - $\theta_1^{t+1} \sim P(\theta_1^{t+1} | \theta_2^t | \{d_i\})$
    $\theta_2^{t+1} \sim P(\theta_2^{t+1} | \theta_1^t | \{d_i\})$
- Zakładamy, że łatwo wygenerować $P(\theta_1 | \theta_2, \{d_i\})$; rozkłady sprzężone, modele DAC

Wyznaczanie rozkładów aposteriori

Samplowanie Gibbsa



Wyznaczanie rozkładów aposteriori

Samplowanie Metropolis-Hastings

- Nie umiemy generować próbek z rozkładu $P(\theta_1 | \theta_2, \{d_i\})$;
- Umiemy generować kandydatów $\{\theta_i\}$.
- Umiemy policzyć funkcję $g(\{\theta_i\}) \sim p(\{\theta_i\}|\{d_i\})$
- Algorytm:
  - Inicjalizacja (wybieramy losowy punkt $\{\theta_i\}^0$)
  - Generowanie kandydata $\{\theta_i\}'$
  - Akceptacja kandydata
    $\{\theta_i\}^{t+1}=\{\theta_i\}'$ z prawdopodobieństwem
    $\max(1, g(\{\theta_i\}')/g(\{\theta_i\}^t))$, w przeciwnym razie
    odrzucamy kandydata i pozostajemy z $\{\theta_i\}^{t+1}= \{\theta_i\}^t$

Narzędzia

- JAGS/WinBUGS (from R via R2jags/R2WinBugs)
- STAN
- PyMC
- PyMC3
- Infer.NET

JAGS

opis modelu/składnia

- Definicja

  model {

  …

  #komentarz

  }

- Węzły/zmienne
  - zmienne stochastyczne

    x ~ dnorm(0,1)
  - zmienne deterministyczne

    Y <- ifelse(x>0,1,0)

JAGS

tablice

- Indeksowanie i pętle

```
for( i in 1:N)
{
    Y[i] ~ dnorm(0,1)
}
```

- Tablice wielowymiarowe Y[i,j],Y[i,],Y[,j],Y[i,1:10]
- Długość tablicy length(Y),
  wymiary dim(Y) (to trzeba wrzucić do bloku danych)
  for(i in 1:length(Y))...

JAGS

operacje wbudowane, funkcje

- Operatory logiczne
- Operatory arytmetyczne
- log,exp,step,ifelse, round
- min,max,sum,mean,
- sort,rank,order,inverse,transpose
- interp.lin

JAGS

rozkłady generujące

- dbeta,dexp,dnorm, dunif
- dbern,dbin, dcat
- ddirch, dmnorm
- Obcinanie rozkładów T(L,U)
  dnorm(0,1)T(0,)

JAGS

dane

- Dane: stałe oraz zmienne stochastyczne,
  dane dla zmiennych deterministycznych generują błąd
- Brak danych NA

```
data {
        Y<-c(1,NA,2,3,4)
        mu<-5.1
}
```

JAGS

JAGS w R: R2jags

- R jako frontend dla JAGS
  - definiowanie modelu
  - dostarczanie danych
  - sterowanie wykonaniem
  - przetwarzanie wyników

JAGS

JAGS via R: R2jags

- R jako frontend dla JAGS
  - definiowanie modelu
  - dostarczanie danych

```
model = function()
{
  maxNTmp ~ dbetabin(1,1,1000-maxY)
  maxN <- maxNTmp+maxY
  for( i in 1:nY)
  {
    Y[i] ~ dbetabin(1,1,maxN)
  }
}

observations <- c(10,256,202,97)
data = list(Y = observations,nY = length(observations),maxY=max(observations))
```

JAGS

JAGS via R: R2jags

- R jako frontend dla JAGS
  - sterowanie wykonaniem (jags, jags.parallel)

```
jags(data, inits, parameters.to.save, model.file="model.bug",
  n.chains=3, n.iter=2000, n.burnin=floor(n.iter/2),
  n.thin=max(1, floor((n.iter - n.burnin) / 1000)),
  DIC=TRUE, working.directory=NULL, jags.seed = 123,
  refresh = n.iter/50, progress.bar = "text", digits=5,
  RNGname = c("Wichmann-Hill", "Marsaglia-Multicarry",
              "Super-Duper", "Mersenne-Twister"),
  jags.module = c("glm","dic")
  )
```

JAGS

JAGS via R: R2jags

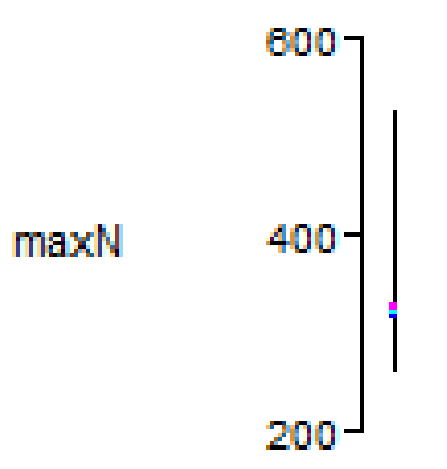
- R jako frontend dla JAGS
  - przetwarzanie wyników: print(model)

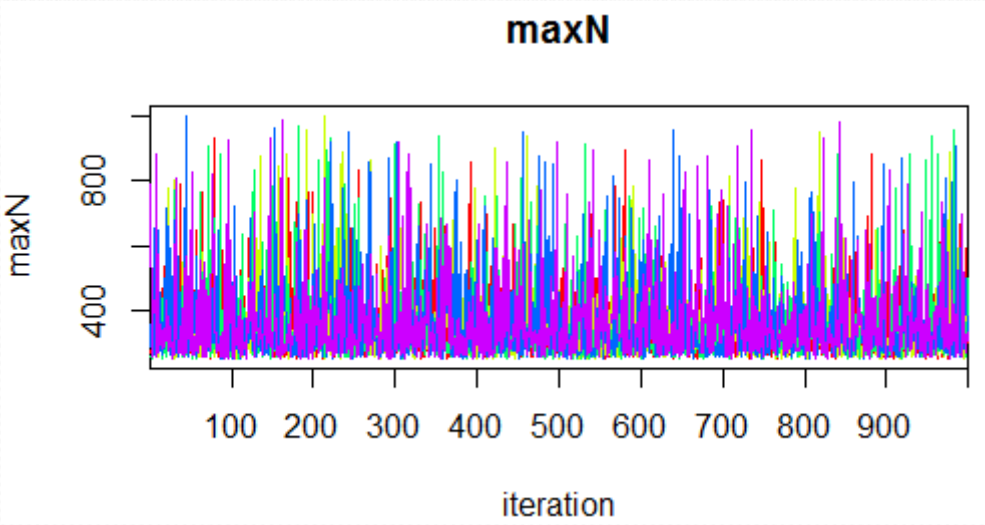
```
Inference for Bugs model at "C:/Users/Lukasz/AppData/Local/Temp/Rtmpeeeuka/model2f0c56e
 5 chains, each with 1e+05 iterations (first 50000 discarded), n.thin = 50
 n.sims = 5000 iterations saved
          mu.vect sd.vect    2.5%     25%     50%     75%   97.5%  Rhat n.eff
maxN      365.126 126.094 258.000 280.000 321.000 399.000 746.000 1.001  4300
maxNTmp   109.126 126.094   2.000  24.000  65.000 143.000 490.000 1.001  4100
deviance   46.863   2.260  44.455  45.107  46.196  47.932  52.929 1.001  4300

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 = 2.6 and DIC = 49.4
DIC is an estimate of expected predictive error (lower deviance is better).
```

JAGS

JAGS via R: R2jags

- R jako frontend dla JAGS
  - przetwarzanie wyników: plot(model), traceplot(model)

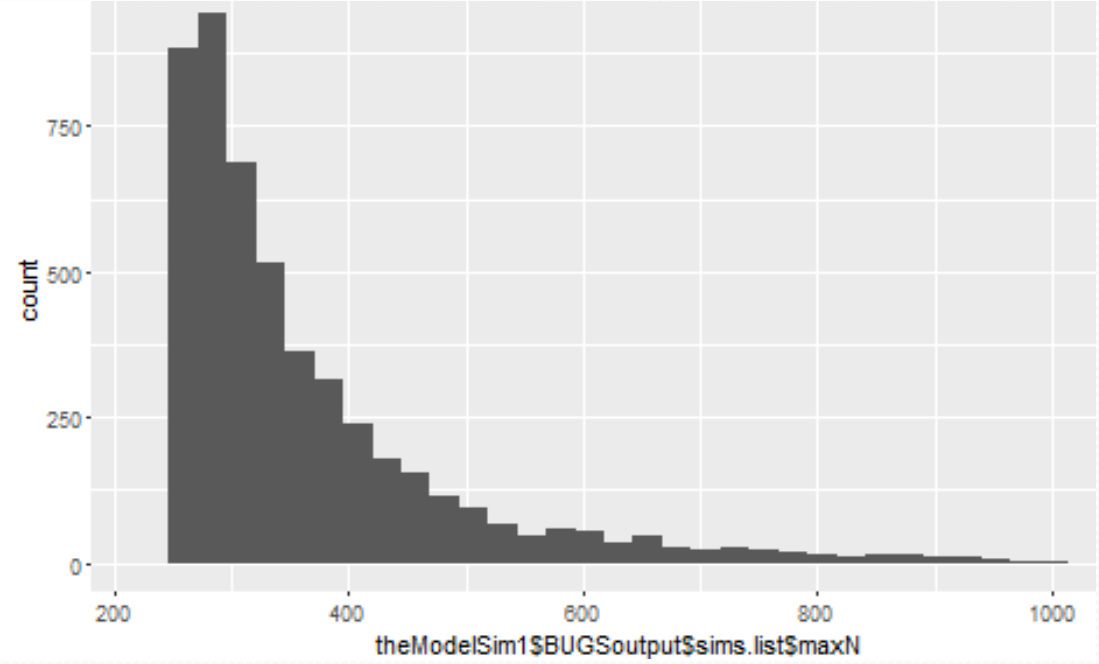




JAGS

JAGS via R: R2jags

- R jako frontend dla JAGS
  - qplot(theModelSim1$BUGSoutput$sims.list$maxN,geom = "histogram")



JAGS

etapy wykonywania modelu

- Kompilacja (błędy)
  - Graf zawiera cykle
  - Brak parametrów (nazwy niezdeklarowane są traktowane jako stałe/dane)
  - Błędy w nazwach funkcji, rozkładów
  - Niewłaściwy format paramtru (np. tabela gdy oczekiwany jest skalar)
- Inicjalizacja
  - Wartości parametrów
  - Generatory liczb losowych, samplery
- Adaptacja samplerów i burn-in
- Monitoring

Ocena jakości modeli (CODA)

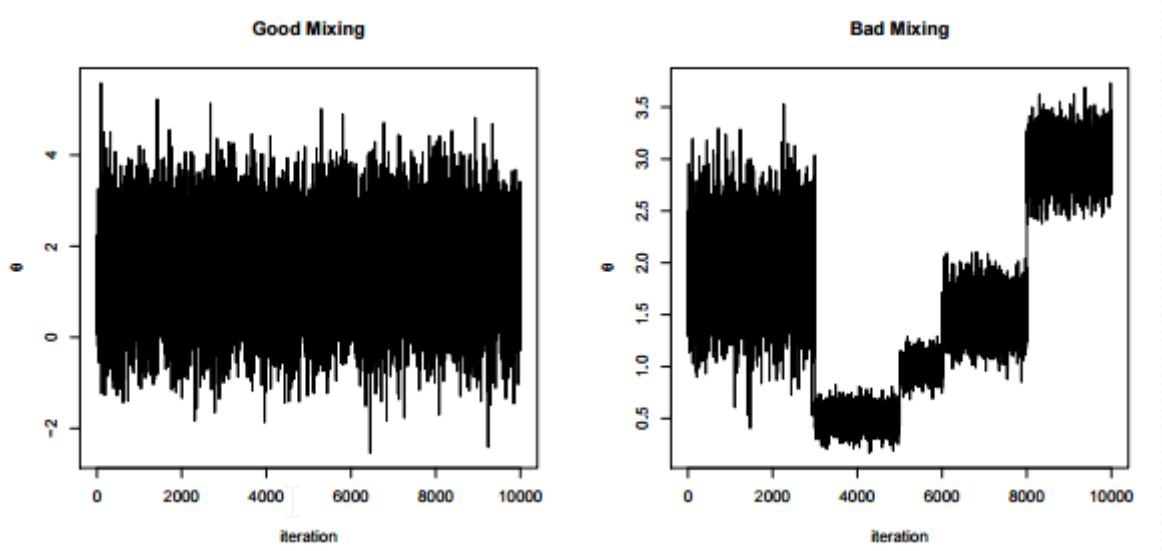
- Testowanie zbieżność
- Efektywny rozmiar próbki i błąd średnich
- Ocena jakości modelu, poziom dopasowania

Bayesian Modeling Using WinBUGS, Ioannis Ntzoufras

Bayesian Data Analysis, Andrew Gelman and John B. Carlin

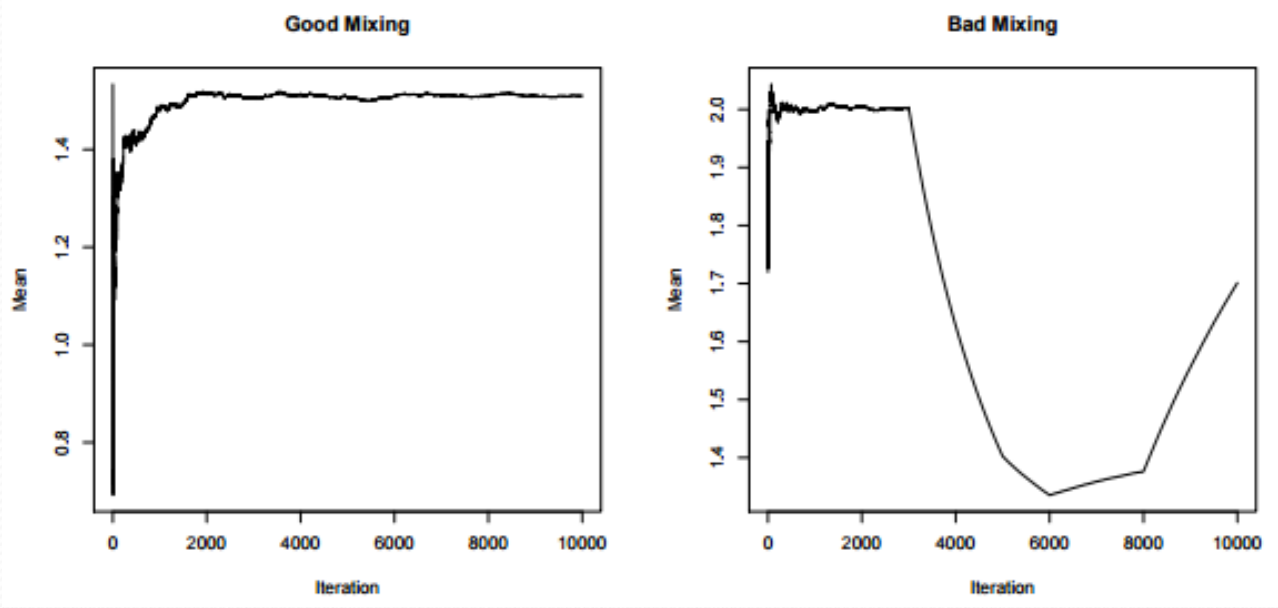
Ocena jakości modeli (CODA)

- Autokorelacje, mieszanie, inspekcja wizualna



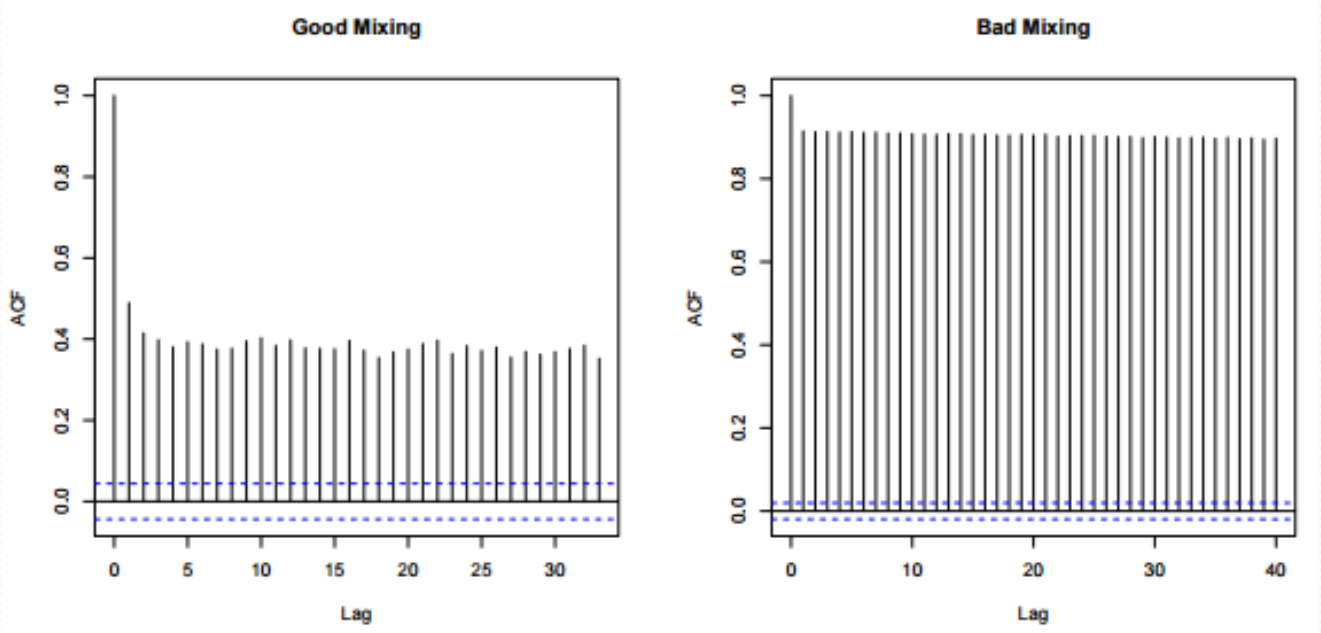
Ocena jakości modeli (CODA)

- Autokorelacje, mieszanie, inspekcja wizualna



Ocena jakości modeli (CODA)

- Autokorelacje, mieszanie, inspekcja wizualna



Ocena jakości modeli (CODA)

- Test Geweke – średnie z dwóch nie przecinających się części łańcucha (pierwsze 0.1 i ostanie 0.5 łańcucha) nie powinny się różnić statystycznie (Z-score)

$$z = \frac{\bar{\theta}_a - \bar{\theta}_b}{\sqrt{Var(\theta_a) + Var(\theta_b)}}$$

- Analogicznie możemy porównywać histogramy/qqplot

Ocena jakości modeli (CODA)

- Gelman-Rubin
  - m łańcuchów, n próbek
  - Test ANOVA (wariancja między łańcuchami do wariancji wewnątrz łańcuchów), Rhat < 1.1

$$B = \frac{n}{m-1}\sum_{j=1}^{m}(\bar{\theta}_{.j} - \bar{\theta}_{..})^2$$

$$W = \frac{1}{m}\sum_{j=1}^{m}\left[\frac{1}{n-1}\sum_{i=1}^{n}(\theta_{ij} - \bar{\theta}_{.j})^2\right]$$

$$\hat{\mathrm{Var}}(\theta|y) = \frac{n-1}{n}W + \frac{1}{n}B$$

$$\hat{R} = \sqrt{\frac{\hat{\mathrm{Var}}(\theta|y)}{W}}$$

Ocena jakości modeli (CODA)

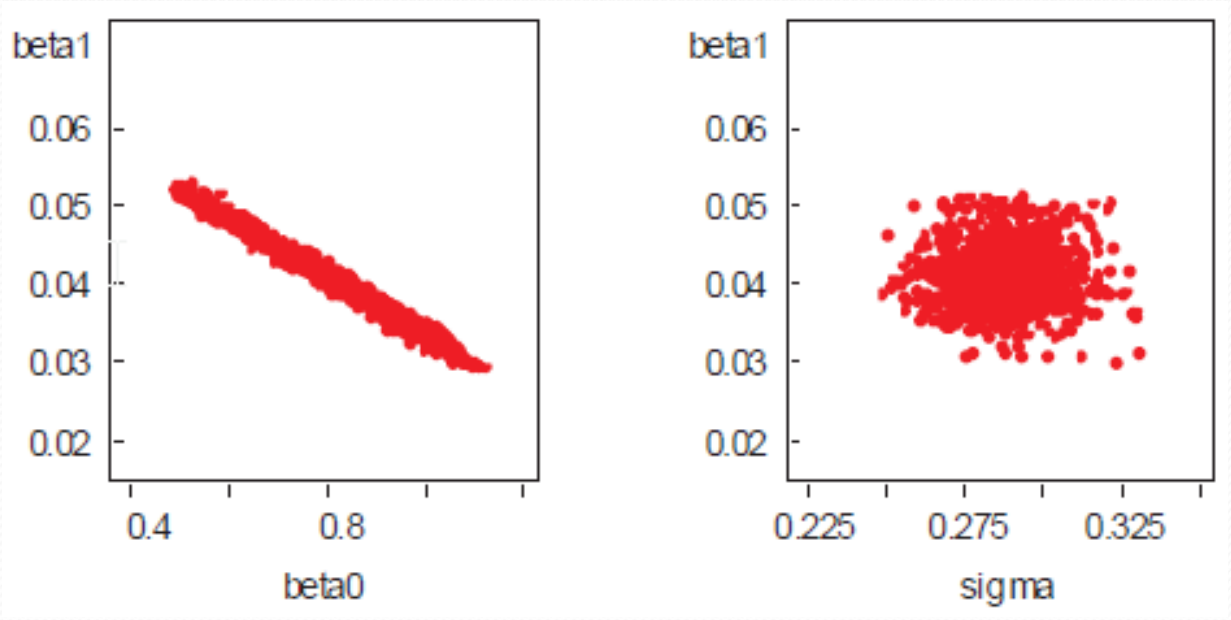
- Efektywny rozmiar próbki

  $n_{eff} = n/(1+(n-1)\rho)$

- Dokładność średnich $\sigma_e = \sigma_m/\mathrm{sqrt}(n)$
- $n_{eff} \sim 400$

Ocena jakości modeli (CODA)

- Korelacje wzajemne między parametrami



Ocena jakości modeli (CODA)

DIC

- Deviance (czy dane pasują do modelu, wyznaczane dla rozkładu aposteriori):
  $D(\theta)=-2 \log p(Y|\ \theta)$;
- Kryterium informacyjne
  (DIC - kwadratowe przybliżenie $D(\theta)$):

$$AIC = D(\hat{\theta}) + 2p$$

$$\mathrm{DIC} = D(\bar{\boldsymbol{\theta}}) + 2p_D$$

- Efektywna liczba parametrów p_D (w sensie AIC)

Ocena jakości modeli (CODA)

Predictive distriubtion

- Generowanie danych przy użyciu dopasowanego modelu, rozkład aposteriori dla prognozy:

$$p(\tilde{y}|y) = \int p(\tilde{y}|\theta) f(\theta|y) d\theta$$

- Rozbieżność, Bayesian p-value

$$p = Pr[D(x_{\text{sim}}|\theta) > D(x_{\text{obs}}|\theta)]$$

- Wartości odstające, cross-validation

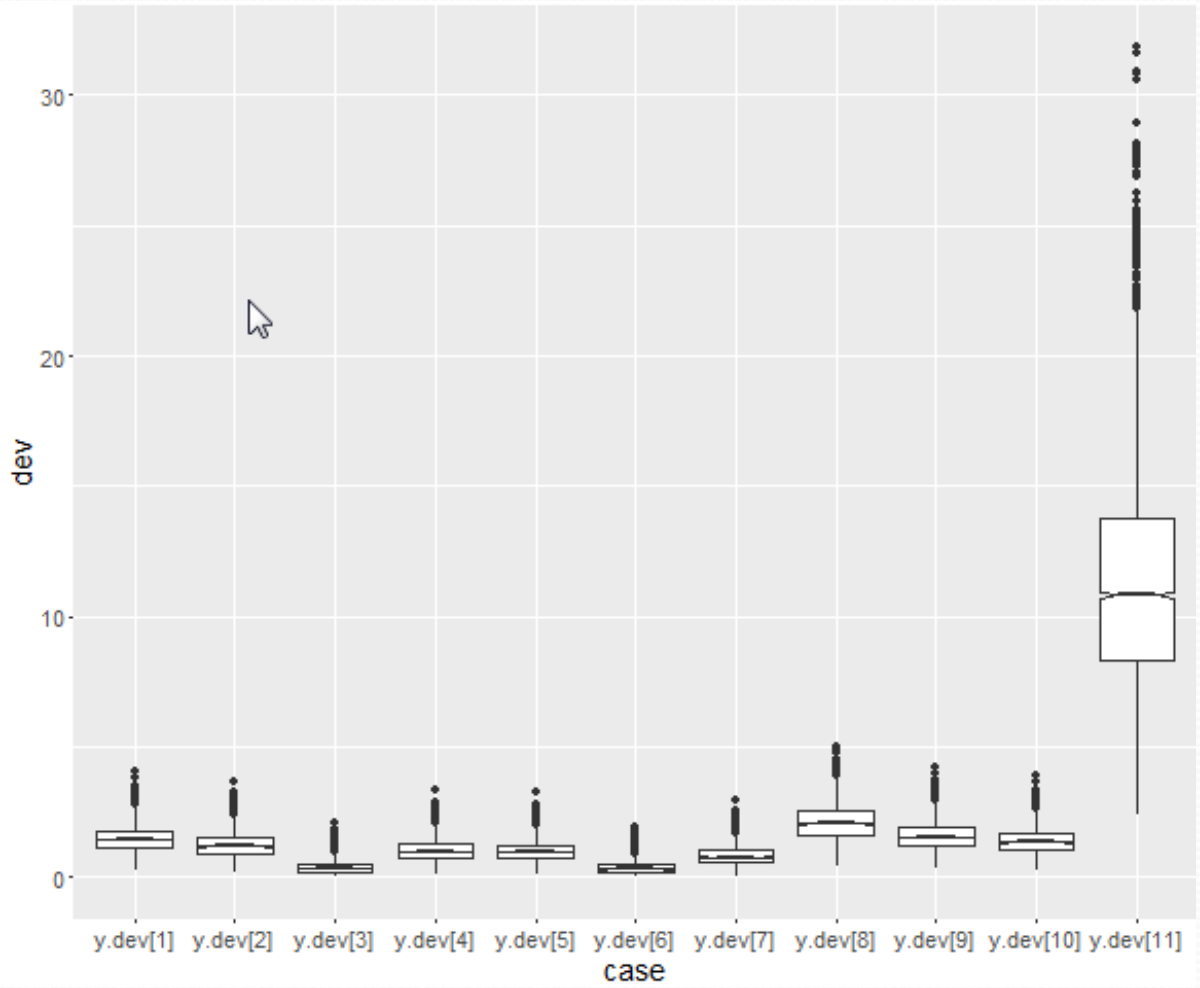
Ocena jakości modeli

Wykrywanie wartości odstających

```
set.seed(123)
df<-data.frame(y=c(rnorm(10,10,1),4))


model <- list(
  model = function()
  {
    alpha ~ dnorm(0,1E-8)

    logsigma ~ dunif( -4 , 4 )
    sigma <- pow( 10, logsigma)
    tau <- pow(sigma,-2)

    for( i in 1:length(y))
    {
      y[i] ~ dnorm(alpha,tau)
      y.dev[i] <- -2*log(pnorm(y[i],alpha,tau))
    }
  },
  data = list(y=df$y),
  parameters = c("alpha","sigma","y.dev"),
  inits = NULL )
```

Ocena jakości modeli

Wykrywanie wartości odstających



Ocena jakości modeli

Testowanie zgodności rozkładów

```
alpha ~ dnorm(0,1E-8)

logSigma ~ dunif( -4 , 4 )
sigma <- pow( 10, logSigma)
tau <- pow(sigma,-2)

for( i in 1:length(y))
{
  y[i] ~ dnorm(alpha,tau)
  y.sim[i] ~ dnorm(alpha,tau)
}

mu <- mean(y[])
s <- sd(y[])
mu.sim <- mean(y.sim[])
s.sim <- sd(y.sim[])

for( i in 1:length(y))
{
  m4[i] <- pow((y[i]-mu)/s,4)
  m4.sim[i] <- pow((y.sim[i] - mu.sim)/ifelse(s.sim<=0,1,s.sim),4)
}

kurt <- mean(m4[])

kurt.sim <- ifelse(s.sim<0,-1000,mean(m4.sim[]))

p.val <- step(kurt.sim-kurt)
```

```
df<-data.frame(y=c(rcauchy(20,10,1)))
```

	mu.vect	sd.vect
alpha	9.803	0.934
kurt	4.418	0.000
kurt.sim	2.452	0.695
p.val	0.019	0.136
sigma	4.164	0.716
deviance	113.156	2.057

Dalsze tematy

- Approximated Bayesian Computation (ABC)
- Probabilistic Programming Languages (Edward, PyMC3)
- Deep Bayesian Learning