Bayesian inference in biosciences and Statistical Data Science

autumn term 2019

Week 3: and Markov chain Monte Carlo

Jarno.Vanhatalo@helsinki.fi

The previous lecture

- R as a tool to practical computation
- Probability distributions
- Representing probability
- Different prior distributions

Aims of the week

- Monte Carlo approximation
- Generative model
- Markov chains
- Probabilistic programming and Stan

- ”Our goal in Bayesian computation is to obtain a set of independent draws θs, s=1,…,S from the posterior distribution, with enough draws S so that quantities of interest can be estimated with reasonably accuracy.”

Markov chain Monte Carlo

- ”Markov chain Monte Carlo (MCMC) simulation is a general method based on drawing values of θ from approximate distributions and then correcting those draws to better approximate the target posterior distribution p(θ|y). ”

- For a general posterior

$$p(\theta|y,\mathrm{n},\mathrm{I}) \propto p(y|\theta,\mathrm{n},\mathrm{I})p(\theta|\mathrm{I})$$

  there are no direct simulation algorithms
- Markov chain is an algorithm which, under certain conditions, samples from a general distribution
- Markov chain Monte Carlo (MCMC)
  - The samples from Markov chain are used for Monte Carlo approximation

Markov chain in a nutshell

- Initialize θ to θ^0
- Draw
  - $\theta^1 \sim q(\theta^1 | \theta^0)$
  - $\theta^2 \sim q(\theta^2 | \theta^1)$
  - …
  - $\theta^S \sim q(\theta^S | \theta^{S-1})$

$$\left.\begin{array}{l} \\ \\ \\ \\ \end{array}\right\}$$

Markov property:

$$q(\theta^i | \theta^0, \dots, \theta^{i-1}) = q(\theta^i | \theta^{i-1})$$

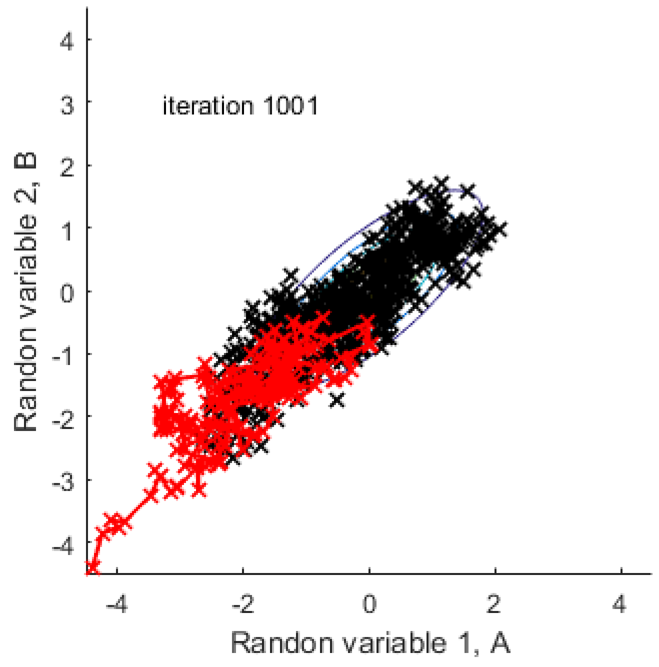
- With suitable q(.|.) The sequence θ^1, …, θ^S converges to the posterior distribution. That is:
  - After certain number of draws, T, the samples are from the posterior distribution

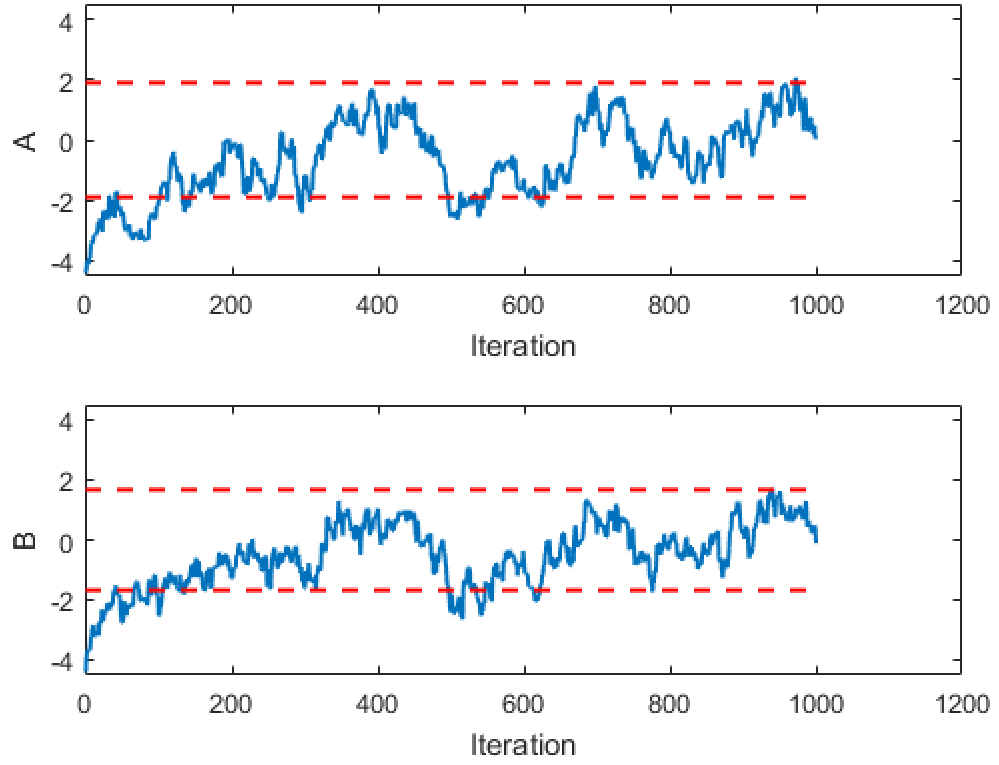
$$\theta^i \sim p(\theta | y) \text{ for all } i > T$$

- Burn-in = the early part of the chain until T
  - Has to be removed before using the samples

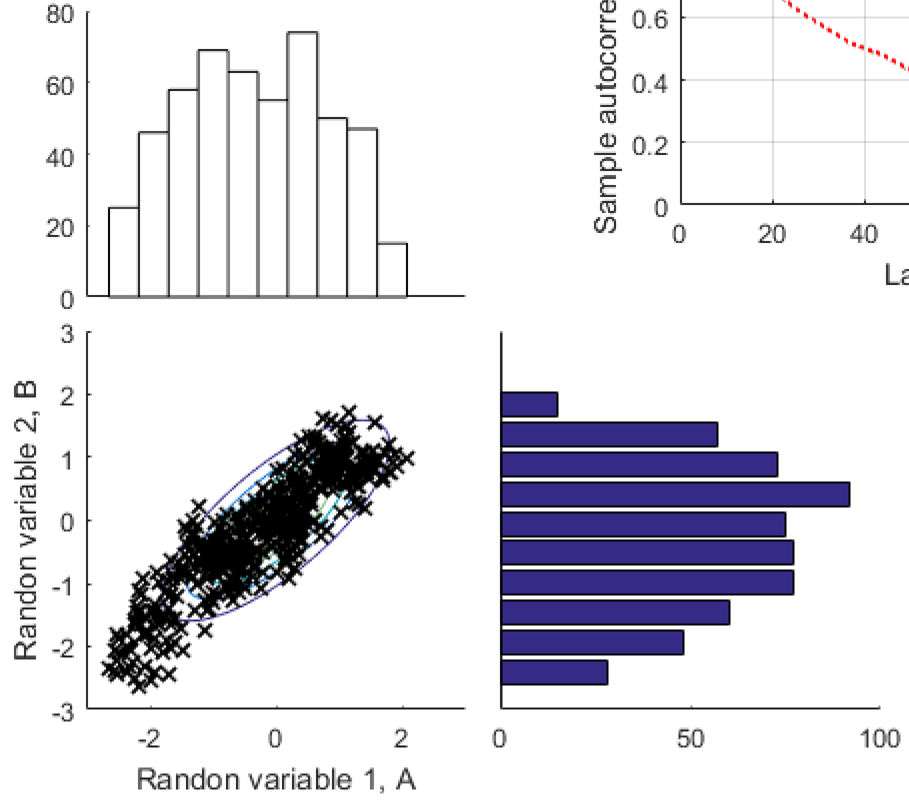
Example
- Markov chain

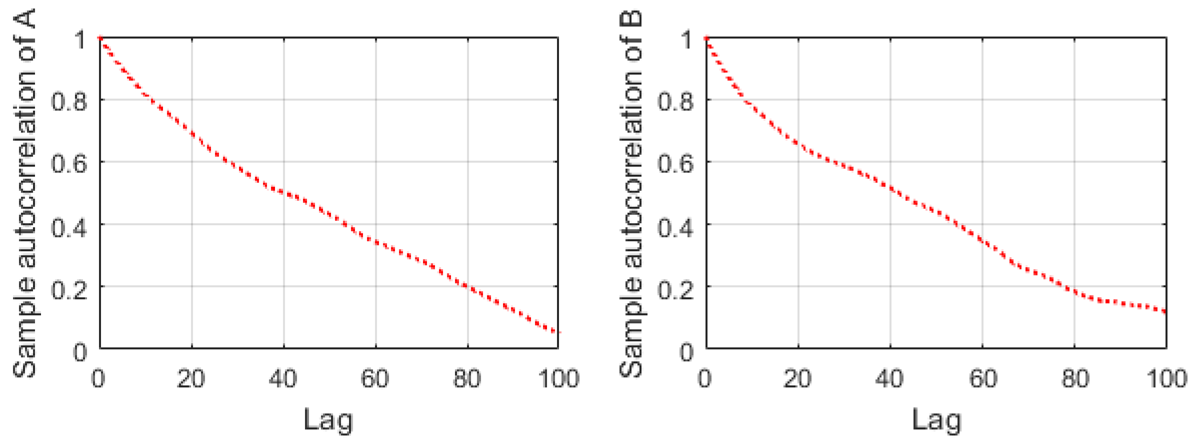
Markov chain example: sample chain in 2-D parameter space





Markov chain example: sample chain in 2-D parameter space





Markov chain Monte Carlo in practice

- Very general method
  - In theory, with infinite time, we are able to infer all models
  - Works for unnormalized distributions
- Theory does not say when Markov chain has converged
  - May take forever with hard models
  - There is no general tool to infer T
  - There are several methods to check for convergence that work in practice
- The samples θ^i are not independent
  - We need more samples than with direct Monte Carlo

Monitoring convergence

- The chain has converged to its target distribution, $p(\theta|y)$, when
  - ”it has forgotten its starting point θ^0”.
- If the proposal distribution $q\left(\theta^{(t)}\middle|\theta^{(t-1)}\right)$ **suggests too small steps**, the chain moves at each iteration but **the chain makes only small moves** (that is, $\theta^{(t)} \approx \theta^{(t+1)} \approx \theta^{(t+2)} \approx \cdots$). As a result
  - It takes long to converge
  - it takes long to cover the whole support of the posterior distribution
  - The autocorrelation between samples is high
- If the proposal distribution $q\left(\theta^{(t)}\middle|\theta^{(t-1)}\right)$ **suggests too large steps**, the canditate values are rejected with high probability and **the chain moves only seldomly** (that is $p(\theta^{(t)} \approx \theta^{(t+1)}) \approx 1$. As a result
  - It takes long to converge
  - it takes long to cover the whole support of the posterior distribution
  - The autocorrelation between samples is high

$\Rightarrow$ Buidling good samplers is challenging. Currently, Stan is among the best.

Steps to monitor convergence (most important)

1. Initialize multiple chains to different locations and run them
    1. Try to spread the initial points more than the expected posterior deviation
    2. Initial points are not like priors, can be derived from the current dataset
2. Calculate statistics of samples between the chains and compare them
3. Visualize
    1. Trace plots of chains look (statistically) identical
4. Remove samples from the beginning until chains cannot be distinguished

“Convergence is diagnosed when the chains have `forgotten' their initial values, and the output from all chains is indistinguishable.”

Monitoring convergence

- Rhat statistics / PSRF diagnostics (Potential Scale reduction Factor / Gelman-Rubin diagnostic)
  - Based a comparison of within-chain and between-chain variances
  - Assumes that samples are (approximately) Gaussian distributed
  - Assumes that initial values have been over-dispersed
  - Upper confidence limit below 1.1 indicates “convergence”
  - See BDA3 pages 281-286

- Many more diagnostics but we use these in this course

Autocorrelation in Markov chain

- In Markov chain every sample depends on the sample before it

  -> Correlation between samples
- The autocorrelation of the sample chain tells
  - the correlation between samples k steps apart
  - how fast the chain forgets its earlier states
  - the effectiveness of the algorithm

Markov chain Monte Carlo in practice

- In practice, building efficient sampler, q(.|.), is hard for many interesting models
  - The chain may not converge in reasonable or finite time
  - The chain may not mix well (=high autocorrelation between samples)
- Often the most efficient computations can be achieved by combining different algorithms
  - Several free software available to conduct MCMC for a number of models
  - Choice of software compromise between efficiency and generality
    - general (and not always efficient): BUGS/OpenBugs, JAGS, (STAN)
      - (easy) to formulate wide range of models
      - Samples do not converge in feasible time for many ”difficult” models
    - Special (and efficient): GPstuff, (STAN), INLA…
      - Tailor-made for a certain model family (e.g., GPstuff for Gaussian process models)
      - Effiecient or at least feasible sampling for that kind of models

Probabilistic programming

- **Probabilistic programming** (PP) is a programming paradigm in which probabilistic models are specified and inference for these models is performed automatically. It represents an attempt to unify probabilistic modeling and traditional general purpose programming in order to make the former easier and more widely applicable. (Wikipedia)
  - https://en.wikipedia.org/wiki/Probabilistic_programming

Stan probabilistic programming language

- In this course we will use the Stan which is a state-of-the-art platform for statistical modeling and high-performance statistical computation.
- Standard distributions such as the normal, gamma, binomial, Poisson, and so forth, are programmed, and arbitrary distributions can be entered directly programmin the log density.
- Algebraic manipulations and functions such as exp and logit can also be included in the specifications.
- For help see Stan documentation
- http://mc-stan.org/users/documentation/index.html

Stan

- For each Stan call the user needs to define
- Model statements
- Parameters
- Data
- Number of chains, number of iterations per chain, and various control parameters (can be set by default)
- Starting values can be supplied or else they are generated from present default variables
- See A crash course to Stan’s syntax
- http://www.sumsar.net/files/posts/2017-bayesian-tutorial-exercises/stan_cheat_sheet2.12.pdf

Code blocks in Stan

A Stan model consist of code blocks. Each block is a place for a certain task. The bold red blocks below must be present in all Stan programs (even if they contain no arguments):

1. functions, where we define functions to be used in the blocks below
2. **data**, declares the data to be used for the model
3. transformed data, makes transformations of the data passed in above
4. **parameters**, defines the unknowns to be estimated, including any restrictions on their values.
5. transformed parameters, often it is preferable to work with transformations of the parameters and data declared above; in this case we define them here.
6. **model**, where the full probability model is defined.
7. generated quantities, generates a range of outputs from the model (posterior predictions, forecasts etc.).

See http://mc-stan.org/users/documentation/tutorials.html

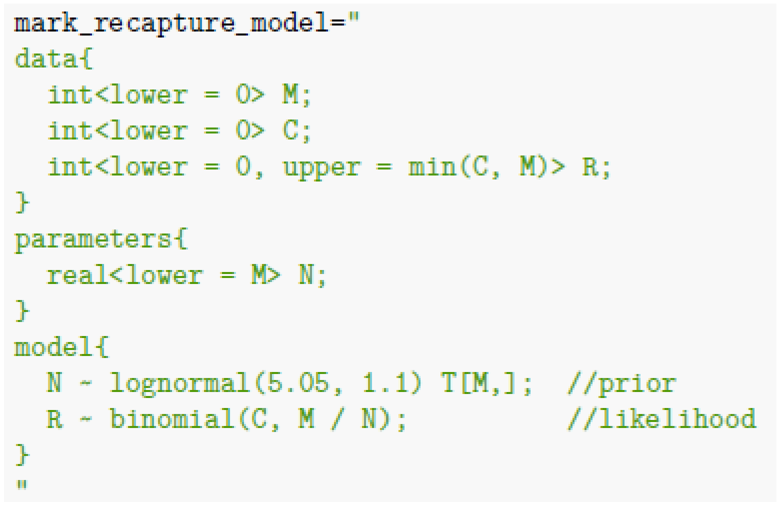
MCMC workflow (with Stan)

- Write your model on the paper
- Code it to Stan model file
- Simulate from posterior with Rstan
- Check for convergence
  - If not: increase adaptation and length of chain (multiple times)
  - If still not: check the model is correct
  - If still not: consult somebody
- Check for autocorrelation
  - Let C be the number after which autocorrelation is under 5%
  - Sample at least C × 1000 samples (rule of thump)
  - (optionally) thin with C
- Calculate the results in R

Model definition in Stan

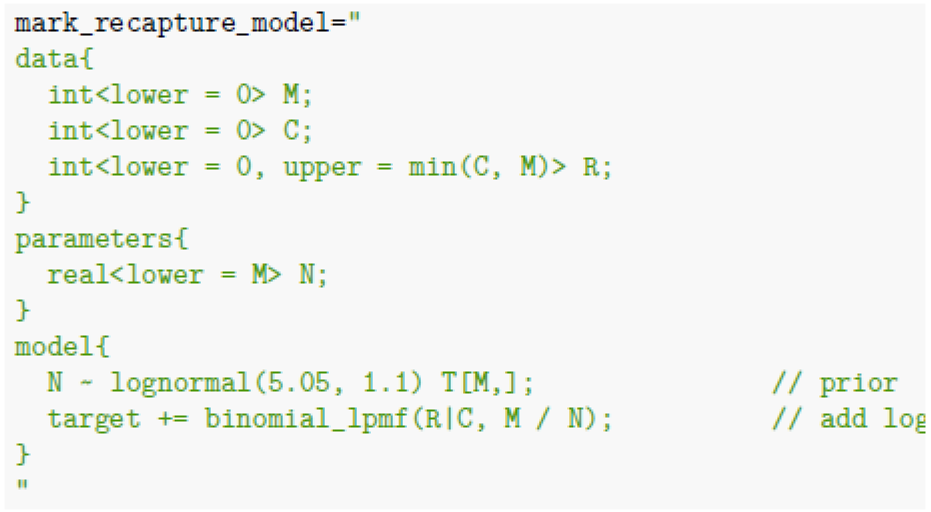
- A model in stan can be used either by describing a generative model using sampling statement ”~” notation or through ”target” variable where target denotes (unnormalized) log posterior density
  - See example_mark_recapture.Rmd for an example
  - See Stan manual section 7.3 and 7.4 (https://mc-stan.org/docs/2_18/reference-manual/)

Example: Sampling statement



```
mark_recapture_model="
data{
  int<lower = 0> M;
  int<lower = 0> C;
  int<lower = 0, upper = min(C, M)> R;
}
parameters{
  real<lower = M> N;
}
model{
  N ~ lognormal(5.05, 1.1) T[M,];  //prior
  R ~ binomial(C, M / N);          //likelihood
}
"
```

Example: Increment log density



```
mark_recapture_model="
data{
  int<lower = 0> M;
  int<lower = 0> C;
  int<lower = 0, upper = min(C, M)> R;
}
parameters{
  real<lower = M> N;
}
model{
  N ~ lognormal(5.05, 1.1) T[M,];            // prior
  target += binomial_lpmf(R|C, M / N);       // add log
}
"
```

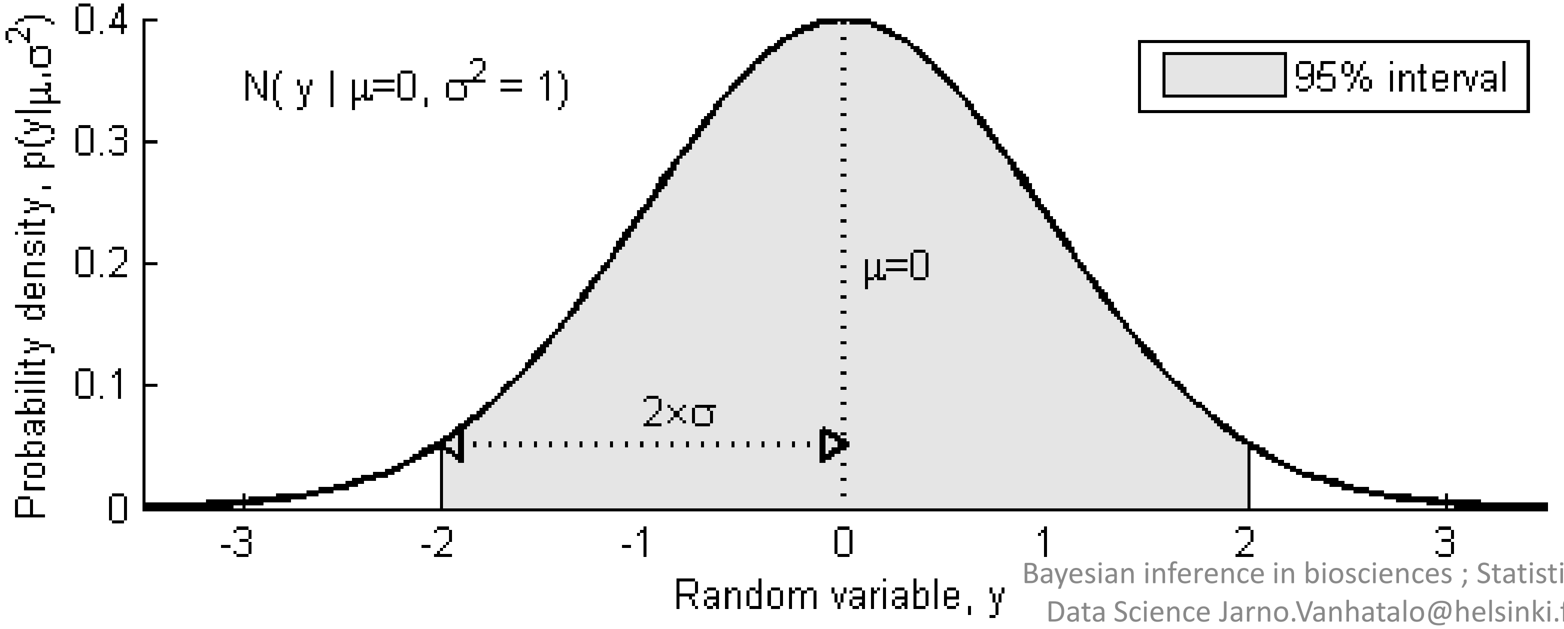
Gaussian distribution

$$p(y|\mu,\sigma^2) = N(y|\mu,\sigma^2)$$
$$= \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(y-\mu)^2/2\sigma^2}$$

y continues real number (-∞, ∞)

μ mean

σ^2 variance



Gaussian observation model

- Noisy observations

$$y_i = \mu + \varepsilon_i$$

  - ε_i: i'th observation (measurement) error
  - μ: true value (e.g. weight, volume, speed, etc.)
  - y_i: i'th observation
- Central limit theorem,
  - Total error a sum of many (zero mean) independent sources
    - Similar scales and ”light-tailed” distributions
  - > error is approximately Gaussian

$$\varepsilon_i \sim N(0, \sigma^2) \Rightarrow y_i \sim N(\mu, \sigma^2)$$

- For logarithm of a positive variable

$$\log(y_i) \sim N(\mu, \sigma^2)$$

  - Total error is product of many independent sources

Gaussian observation model

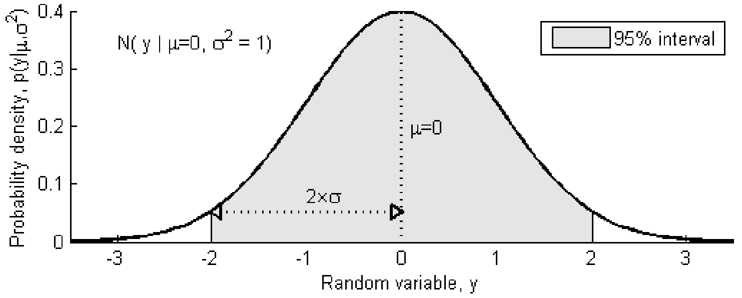
- The observation model / likelihood function in case of *n* <u>independent</u> (identically distributed) observations $y = \{y_1, \dots y_n\}$

$$p(y|\mu, \sigma^2) = N(y_1|\mu, \sigma^2) \times \dots \times N(y_n|\mu, \sigma^2)$$
$$= \prod_{i=1}^{n} N(y_i|\mu, \sigma^2)$$

Gaussian observation model

The functional form of $N(y|\mu, \sigma^2)$ is convenient

- Easy to calculate even analytically
  - Important fact before efficient computers
- Relation to minimum squared error approach
- Linear regression ($\mu = ax_1 + bx_2 + \cdots$) analytically tractable
- <u>Even though common, not always appropriate!</u>



Gaussian observation model

- Very practical and common in many models
    - Often assumptions are valid
    - Even if not totally valid, good for approximation
    - Important part of hierarchical models (later)
    - Important part of mixture models
    - Can be used to build more robust Student-*t* model
- BDA3 devotes much time on analytical solutions
    - Analytical solutions for posterior distribution with ’completing square’ technique
    - Very useful to understand but not treated during this course
    - Here, we analyze with Stan

Gaussian observation model

- Observation model

$$y_i | \mu, \sigma^2 \sim N(\mu, \sigma^2)$$

- Different possibilies for parameters
  1. σ^2 is known, μ is unknow
  2. μ is known, σ^2 is unknow
  3. μ and σ^2 are unknow

Gaussian observation model

- Observation model

$$y_i|\mu,\sigma^2 \sim N(\mu,\sigma^2)$$

- Prior

$$\mu \sim N(\mu_0,\tau^2)$$
$$\sigma^2 \sim \mathrm{Inv}-\boldsymbol{\mathcal{X}}^{\mathbf{2}}(\upsilon_0,\sigma_0^2)$$

  - Mean and variance independent *a priori*

$$p(\mu,\sigma^2) = p(\mu)p(\sigma^2)$$

  - In BDA3 example where mean and variance are *a prior* dependent

$$p(\mu,\sigma^2) = p(\mu|\sigma^2)p(\sigma^2)$$

Gaussian observation model

- Observation model $y_i | \mu, \sigma^2 \sim N(\mu, \sigma^2)$

- Scaled Inverse-Chi squared prior for <u>variance</u>

$$\sigma^2 \sim \mathrm{Inv} - \boldsymbol{\chi}^2 \ (\upsilon_0, \sigma_0^2)$$

  - Same as inverse-Gamma distribution (BDA3 p. 579)

$$\sigma^2 \sim \mathrm{Inv} - \mathrm{gamma}(\alpha, \beta)$$

- Gamma prior for <u>precision</u>

$$1/\sigma^2 \sim \mathrm{gamma}(\alpha, \beta)$$

This week

- Monte Carlo method
- Markov chains
- Stan: a program to conduct Markov chain sampling for Bayesian models
- Gaussian observation model

Next week

- Gaussian observation model
- Linear regression