Computation methods

- Bayesian inference is based on reporting properties of posterior distributions (means, tail areas, etc.)
- This needs *integration* over posterior distribution.

- Conjugate priors can be found for limited cases → need computational methods to approximate posterior density and integrals of it.

Computation methods

- Approaches:
  - Exact analytic solution of posterior distribution available, algebraic solution to integrals (e.g. tail areas) – mainly for simplest problems.
  - Exact analytic solution of posterior, numeric solution to integrals (e.g. integrals of beta-distributions by R-functions)
  - Approximate but analytic solution: Normal distribution as an approximation
  - Monte Carlo approximation: generate large sample of random values from posterior distribution, evaluate empirical distribution of this sample.
  - Also: for some type of models, other numerical computations possible (e.g. INLA)

Normal approximation

- With larger data set (sample size large), posterior distributions tend to get more peaked → looks like normal distribution!
  - Approximate posterior distribution by N($E(\theta|X)$, $V(\theta|X)$), if you can just find out posterior mean and variance. Then compute integrals from this normal density as needed.
  - **Modal approximation:** focus on posterior mode instead of mean:

$$\pi(\theta \mid X) \approx N\left(\hat{\theta}, \left[I(\hat{\theta})\right]^{-1}\right)$$

  - Here: $\hat{\theta}$ is posterior mode
  - And $I(\theta) = -\frac{d^2}{d\theta} \log \pi(X \mid \theta)$ is called 'observed information'

Normal approximation

- Modal approximation is based on Taylor series expansion of log-posterior density function (≈log-likelihood if large sample) at mode

$$\log \pi(\theta \mid X) \approx \log \pi(X \mid \hat{\theta}) + \quad \leftarrow \text{constant wrt } \theta.$$

$$\left[\frac{d}{d\theta} \log \pi(X \mid \theta)\right]_{\theta=\hat{\theta}} \frac{(\theta - \hat{\theta})}{1!} + \quad \leftarrow =0 \text{ (derivative at mode)}$$

$$\left[\frac{d^2}{d\theta^2} \log \pi(X \mid \theta)\right]_{\theta=\hat{\theta}} \frac{(\theta - \hat{\theta})^2}{2!} + \ldots$$

$$+ \ldots \quad \leftarrow \sim 0 \text{ (higher order terms small if } \theta \text{ near mode)}$$

Intro to Monte Carlo method

- Example of a Monte Carlo sampler in 2D:
  - imagine a circle (radius L/2) within a square of LxL.
  - If points are randomly generated over the square, what's the probability to hit within circle?
  - By algebra: $\pi(L/2)^2/L^2 = \pi/4 = 3.14159\ldots/4$.
  - By simulation:

$$P(\theta \in S) \approx \frac{1}{K}\sum_{k=1}^{K} 1_{\{\theta \in S\}}(\theta^k)$$

  - This also provides a Monte Carlo approx of π.

Law of large numbers

- If θ^k (k=1,2,3,…) are i.i.d. (independent, identically distributed) with probability density $\pi(\theta)$, then

$$\overline{g(\theta)_K} = \frac{1}{K}\sum_{k=1}^{K} g(\theta_k) \to E(g(\theta)) = \int_{\Theta} g(\theta)\pi(\theta)d\theta$$

- Integrations can be done by Monte Carlo sampling.

Monte Carlo used for…

- To approximate mean, variance, probability, for a density of θ or $g(\theta)$.

$$\bar{\theta}_K = \frac{1}{K}\sum_{k=1}^{K}\theta_k \rightarrow E(\theta) \qquad \text{when} \qquad K \rightarrow \infty$$

$$\overline{\theta^2}_K = \frac{1}{K}\sum_{k=1}^{K}\theta_k^2 \rightarrow E(\theta^2)$$

$$\frac{1}{K}\sum_{k=1}^{K}(\theta_k - \bar{\theta}_K)^2 = \frac{1}{K}\sum_{k=1}^{K}\theta_k^2 - (\bar{\theta}_K)^2 \rightarrow E(\theta^2) - (E(\theta))^2 = V(\theta)$$

$$\frac{1}{K}\sum_{k=1}^{K} I_{\{0,\infty\}}(\theta_k) \rightarrow E(I_{\{0,\infty\}}(\theta)) = P(\theta > 0) \qquad \text{use indicator variables!}$$

→ Can do approximate Bayesian inference.

…used for:

- Wanted: e.g. posterior mean

$$E(\theta \mid X) = \int \theta \pi(\theta \mid X) d\theta$$

- With conjugate priors, $\pi(\theta|X)$ would be a standard distribution.
  - Calculate directly, using known expressions.
  - Use statistical software, e.g. R to compute quantiles, etc. qbeta(c(0.025,0.975),2,5)
  - Even if we had solved the density, it can be difficult to evaluate E($g(\theta)$ |X) → Monte Carlo is easier.

Many Monte Carlo methods

- Monte Carlo is just a label for lots of methods, below some examples
- Each aims to produce a random sample from a target distribution (in bayesian inference: this is usually posterior distribution)

- Some methods produce **independent random samples** (i.i.d.).
- **Markov chain Monte Carlo** methods produce dependent samples. These are more generally applicable – and used in BUGS.

WinBUGS/OpenBUGS

- A tool that will do the Monte Carlo sampling for you (more generally MCMC)
- What you need to do?
  - **Write *logical definition*** of your model:
    - Prior and likelihood.
    - Model can be hierarchical with several layers.
  - Define what your data are. (fixed values).
  - The model should constitute a proper posterior distribution. (Or prior if no data).
  - Compile and run, monitor results, check convergence, analyze results.

WinBUGS/OpenBUGS

- Binomial model in BUGS
  - Recall the conjugate solution.
  - $\pi(\theta|X) = \pi(X \mid N,\theta)\ \pi(\theta)\ /c$
  - To compute posterior, we define $\pi(X \mid N,\theta)$ and $\pi(\theta)$. And we set a value for observed X (=data).
  - Note: we do not need to define or solve constant c!

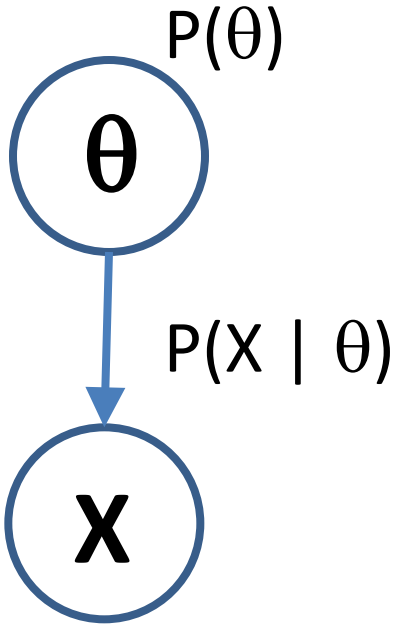
WinBUGS/OpenBUGS

- **In BUGS –language:**

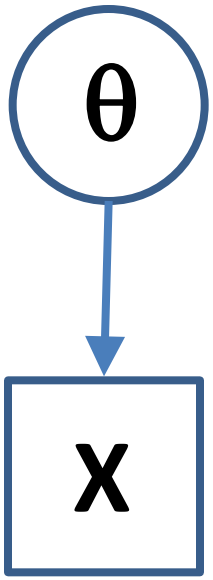
```
model{
X ~ dbin(theta,N)   # defines 'likelihood'
theta ~ dunif(0,1)  # defines prior
# or maybe:  ~dbeta(a,b)
}
# data given as a list:
list(X=3,N=20)
```

Directed Acyclic Graph: DAG

- **Graphical representation: DAG**
  - Describes conditional distributions



When X is unknown

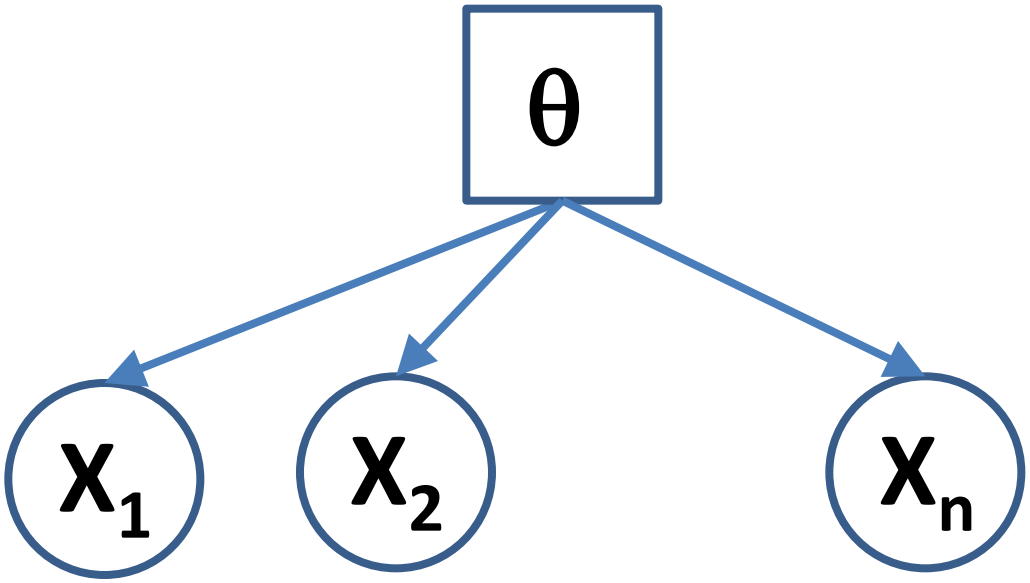


Can solve:
$P(\theta|X)$

When X is observed (fixed) as data

DAG

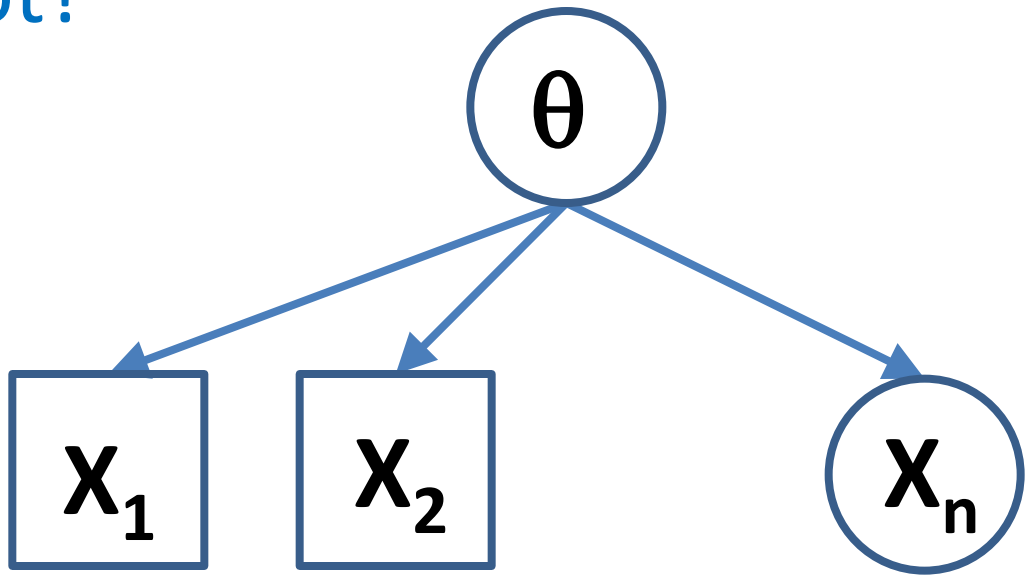
- What happens if θ is fixed, X are not?



  - X will be independent of each other, given θ. → Simulate each X independently with given parameter(s).

DAG

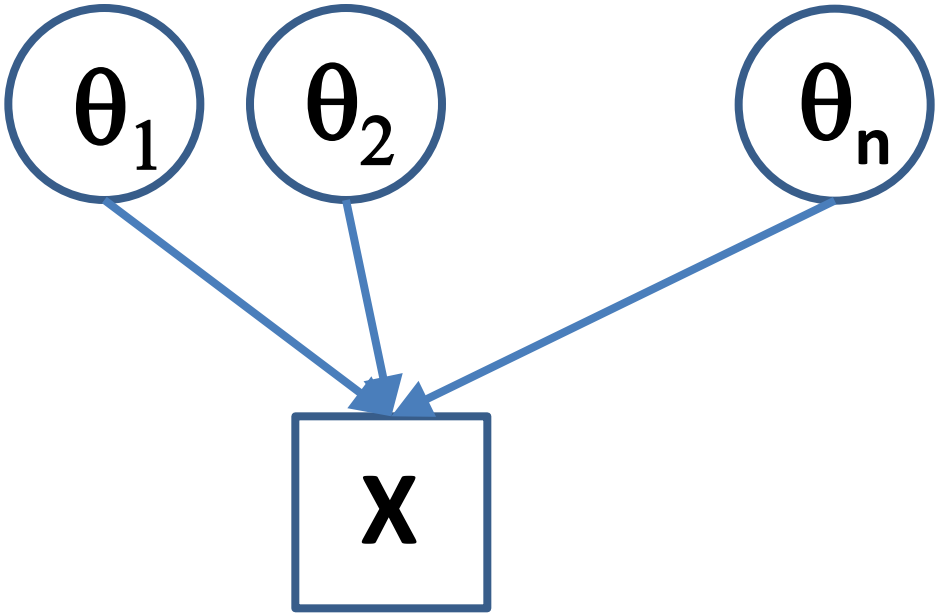
- What happens if *some X* are fixed, θ is not?



  - Unknown X will be dependent on known X. (we can learn from the 'siblings'). → solve posterior for θ, simulate X_n from that.

DAG

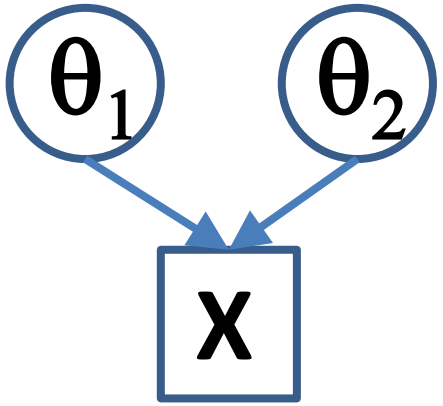
- What happens if X is fixed, θ are not?



- Given X, unknown θ will be dependent on each other. E.g. X ~ Bin(1-(1-θ_1)(1–θ_2),N), so that data X constrain the possible values of θ.

DAG

- Identifiability of θ_1 & θ_2 ?



- **In general:** if for different parameter values $\theta \neq \theta'$ the likelihood function is different $L(\theta) \neq L(\theta')$, then parameter(s) θ is (are) identifiable from data.

DAG

- E.g. X ~ Bin(1-(1-θ_1)(1$-\theta_2$),N) parameter θ = (θ_1, θ_2) not identifiable from data X&N.
- Identifiability in posterior?
  - Only if prior evidence exists so that $\pi(\theta) \neq \pi(\theta')$.
  - In principle: identifiability not a problem for conducting Bayesian inference, as long as the posterior still is a proper distribution!
  - But could lead to computational problems in practice, e.g. poor convergence in BUGS.

About BUGS language

- **Declarative language: don't try to think procedural programming.**
- **Directly corresponds to a DAG**
- **(1) Nodes are either 'stochastic' or 'deterministic':**
  - They depend on parents (=other nodes): either as

    Child ~ ddistribution(Parents) [stochastic]

    Or:

    Child <- function(Parents) [deterministic]
- **(2) Or nodes are 'founder nodes' which are constants. E.g. parameters of prior distribution (no parents), or fixed design variable N in binomial modeling.**
- **When data are assigned to any Child node → Bayesian inference about parents.**

About BUGS language

- Every model is a logical definition which corresponds to defining likelihood and prior in Bayes theorem
- A logical definition can be expressed in several equivalent ways:

```
X ~ dbin(theta,N)
theta ~ dunif(0,1);   X <- 3; N <- 20
```

is same as (assuming x is given as data)

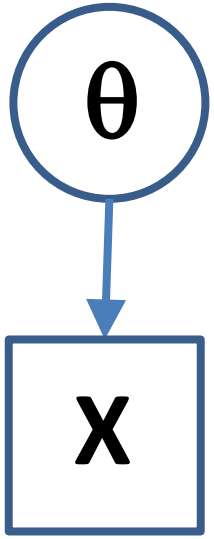
```
theta ~ dbeta(a,b);
a <- X+1 ;   b <- N-X+1; X <- 3; N <- 20
```

(if x was not given as data, the latter would not be defined, and the former would produce predictive distribution for X & the prior for theta)

About BUGS language

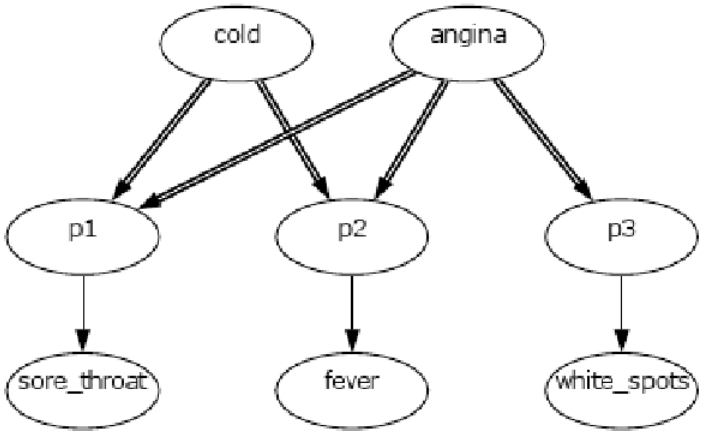
- It is good practice to keep model definition and data separated.
- Data = ”everything that is given as constant”
- Model = defined functions and/or distributions (for all nodes in DAG):

```
model{
X ~ dbin(theta,N)
theta ~ dbeta(a,b);
}
# data:
List(a=1,b=1,X=3,N=20)
```



WinBUGS/OpenBUGS

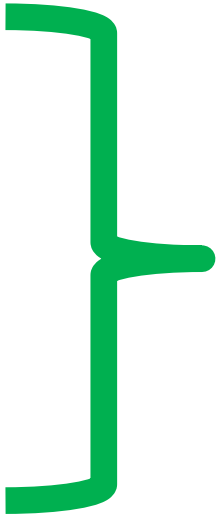
- Bayesian inference & diagnosis in BUGS



- Parent nodes → Child nodes: from cause to effect.
- May contain stochastic & deterministic nodes.
  - Here: effects are stochastic, depend on parameters determined by causes (which again are stochastic).

WinBUGS/OpenBUGS

- Cold = 1/0
- Angina = 1/0
- Sore_throat = 1/0
- Fever = 1/0
- White_spots = 1/0



Stochastic nodes

- Angina can cause any of the symptoms
- Cold can cause only fever or sore throat
- (=assumptions!)

WinBUGS/OpenBUGS

- P(Sore_throat) = p1
- P(Fever) = p2
- P(White_spots) = p3

Deterministic nodes

- p1 <- cold * (1 - angina) * 0.1 +
  (1 - cold) * angina * 0.8 +
  cold * angina * 0.95 +
  (1 - cold) * (1 - angina) * 0.05

- p2 and p3 similarly defined….

WinBUGS/OpenBUGS

- Resulting code, generated from the graphical model:

```
model{
    angina ~ dbern(0.5)
    cold ~ dbern(0.5)
    fever~ dbern(p2)
    sore_throat ~ dbern(p1)
    white_spots ~ dbern(p3)
    p1 <- cold * (1 - angina) * 0.1 + (1 - cold) * angina * 0.8 + cold * angina * 0.95 + (1 - cold) * (1 - angina) * 0.05
    p2 <- cold * (1 - angina) * 0.1 + (1 - cold) * angina * 0.7 + cold * angina * 0.85 + (1 - cold) * (1 - angina) * 0.01
    p3 <- angina
}
```

WinBUGS/OpenBUGS

- The model is joint distribution of all unknown variables
- The two causes are given prior probabilities (here: P(angina)=0.5, P(cold)=0.5 independently)
- The symptoms are given conditional probabilities, given by parameters p_1, p_2, p_3, because they are Bernoulli variables. These probabilities should depend on the causes, according to a given model.
- Goal: to evaluate P(angina,cold | symptoms).
- This model could be constructed graphically with 'Doodle-BUGS' or directly writing the code.

WinBUGS/OpenBUGS

- Including full data in a full likelihood.
- With several data points $X_1+\ldots+X_n$:
  - Write the likelihood using ***sufficient statistics***, if this can be found.
    - E.g. instead of the product of n Bernoulli-likelihoods we can write one binomial likelihood,
    - or instead of n exponential likelihoods we can write one likelihood with gamma-density [because if $X_i \sim \exp(\theta)$, then $X_1+\ldots+X_n = Y \sim \text{Gamma}(n,\theta)$]
  - Or just write full likelihood as ***product of*** $\mathbf{P(X_i \mid \theta)}$ using for-loops in BUGS:

    for(i in 1:n){ x[i] ~ ddistr(parameters) }

WinBUGS/OpenBUGS

- Write also likelihood terms for **censored data, if needed**.
- The posterior distribution may no longer have analytic solution, because conjugacy may not exist, but BUGS can simulate the posterior.
- Censored data models in OpenBUGS:
  - for(i in 1:n){ x[i] ~ ddistr(parameters) C(,B[i]) }
    If the observation was x[i]<B[i]
  - for(i in 1:n){ x[i] ~ ddistr(parameters) C(A[i],) }
    If the observation was x[i]>A[i]
  - for(i in 1:n){ x[i] ~ ddistr(parameters) C(A[i],B[i]) }
    If the observation was A[i] < x[i] < B[i]

  - Note: the corresponding x[i] should be written as NA in the data, whereas exactly observed x[i] are given the observed values in data listing.

WinBUGS/OpenBUGS

- Model → Specification Tool
  - Check model (check syntax)
  - Load data (values for observed variables)
  - Compile (check if model + data makes a posterior)
  - Gen inits (initial values for the MCMC sampler)
- Model → update tool
  - Update = run some MCMC iterations
- Inference → Sample monitor tool
  - Specify which parameters to analyse (=see their marginal distributions), & Update more iterations.

Demo with OpenBUGS