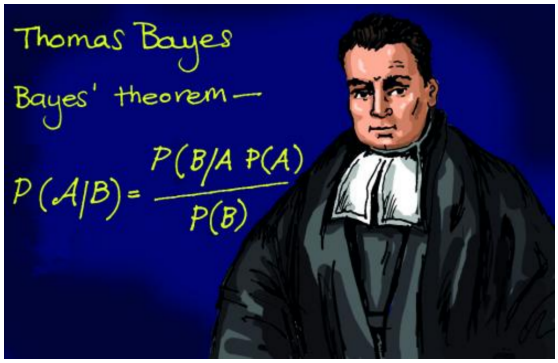
Lectures 15, 16 and 17:
Introduction to Bayesian Statistics models

Data Science 2
CS 109b, Stat 121b, AC 209b, E-109b

Mark Glickman Pavlos Protopapas

Bayesian statistics:

- Foundations and philosophy of Bayesian statistics
- Recipe to performing a Bayesian analysis
- Advantages and disadvantages to the Bayesian paradigm
- Modern approach to Bayesian statistics
- Example analyses: Hierarchical Models, Latent Dirichlet Allocation (LDA)



Are you a Bayesian?

In the following situations, what would you conclude about the (future) probability of success?

1. A music expert claims he can identify whether a piece of music was written by Mozart versus Haydn.

In 10 trials, the expert identifies the composer correctly 10 times.

2. A drunken friend claims he can predict the result of coin flips.

In 10 trials, he answers correctly 10 times.

3. A coffee connoisseur claims to be able to tell the difference between whether cream is poured in coffee first, or vice versa.

In 10 trials, the person answers correctly 10 times.

If you drew different conclusions in all three situations, then you are likely already Bayesian.

- One key feature of Bayesian statistics is that it is a "learning paradigm." You cannot (and should not) ignore knowledge you currently have in summarizing conclusions based on data.
- In the three example situations, we may have had strong prior beliefs that influenced our conclusions about the data. In classical settings, it is not straightforward to incorporate such beliefs.
- Bayesian statistics provides a formal way to incorporate prior information into an analysis.

Question:

What is the underlying difference between classical and Bayesian statistics?

Answer:

The definition of probability.

Example: Flip a fair coin; the probability the coin lands heads is $\frac{1}{2}$.

What does "the probability is $\frac{1}{2}$" mean?

Classical interpretation:

On repeated flips of the coin, the coin lands heads $\frac{1}{2}$ of the time in the long run.

Frequency definition of probability: Probability of an event is its **long-run frequency** of occurrence.

This definition is useful for describing the likelihood of potentially observable data.

Bayesian interpretation:

There is an equal probability of heads and tails due to the symmetry of the coin, so that $\frac{1}{2}$ is an assessment of my belief that the coin will land heads.

Subjective definition of probability: Probability of an event is one's **degree of belief** that the event will occur (1 means that it is certain to occur, and 0 means that it is certain not to occur).

This definition is useful for quantifying beliefs about non-data events (but also data events).

Classical statistics, which is founded on the frequency definition of probability, is usually called Frequentist statistics. Much of the theory developed by Jerzy Neyman in early 1900s.

Implications to a theory of statistical inference:

- Statistics founded on the Frequentist definition of probability can only assign probabilities to future data or potentially observable quantities.
- Statistics founded on the subjective definition of probability can consider probabilities of values of unknown parameters (as well as values of potentially observable quantities).

Variation of coin flipping:

Suppose yesterday, at 5:00pm, standing at home, I flipped a fair coin and recorded whether it landed heads.

What is the probability it landed heads?

Answer:

If you are a Bayesian, the answer is unequivocally $\frac{1}{2}$.

If you are a Frequentist, you are in a bit of a bind because I am not asking about a long-run frequency – I am asking about what happened specifically at 5:00pm yesterday.

A Frequentist would need to say either

- that the probability is either 0 or 1, but would not know which is correct, or
- need to resort to frequencies of events across different universes within a multiverse!

Because the goal of statistical inference is to infer unknown quantities from data, Frequentists often find themselves in awkward philosophical quandaries.

Example: Confidence intervals

Suppose $y_1, \ldots, y_n$ are a random sample of n values from $\text{N}(\mu, \sigma^2)$.

Formula for a 95% confidence interval is

$$(\bar{y} - t^*_{n-1}\frac{s}{\sqrt{n}},\ \bar{y} + t^*_{n-1}\frac{s}{\sqrt{n}}).$$

where $\bar{y}$ is the sample mean, s is the sample standard deviation, and t^*_{n-1} is a critical value from the t-distribution on $n-1$ degrees of freedom.

Before observing data, 95% of the time these endpoints will contain μ, the value we are trying to infer.

- Notice that the Frequentist definition applies to characterizing the **procedure**. That is, if we apply the procedure for determining 95% confidence intervals according to the formula, we will be correct 95% of the time.
- Once data are observed, the coverage probability is either 0 or 1.

Here is a dialog I once heard between a Frequentist and his employer where the Frequentist tried to explain confidence intervals.

A Conversation between a Frequentist Consultant and his insightful Employer

Employer: So, do you have the results of the drug trial analyses?

Frequentist: Yes, I do.

Employer: Let's hear it.

Frequentist: Okay. I computed a 95% confidence interval for the average decrease in diastolic blood pressure from taking the medication, and I got $(10.18,\ 14.92)$ mm Hg.

Employer: Great! What does that mean?

Frequentist: It means we should be 95% "confident" that the true mean decrease is between 10.18 and 14.92.

Employer: Oh, you mean that there is a 95% probability that the mean decrease is between 10.18 and 14.92 mm Hg. Got it!

Frequentist: Well, not exactly.

Employer: Huh?

Frequentist: It means that if we were to compute "95% confidence intervals" from many data samples, about 95% of them would contain the true mean decrease in blood pressure.

Employer: Um... But what about **this** data set?

Frequentist: I don't know.

Employer: What do you mean you don't know?

Frequentist: All I know is that in 95% of the analyses I perform, my 95% confidence intervals contain the parameter I'm estimating.

Employer: But I've hired you to draw conclusions from this data set!

Frequentist: Well, you can hope that your data set is "typical."

Employer: What are you talking about? What does that have to do with 95%?

Frequentist: If your data set was one of the 95% "typical" data sets you would observe, then the confidence interval would contain the mean decrease in blood pressure.

Employer: But what does that tell me about the confidence interval you reported to me?

Frequentist: Nothing.

Employer: Nothing?

Frequentist: Can I get my consulting fee? I have to go.

Employer: No way! You're FIRED!!!

Bayesians compute and report intervals as well, but they have different meaning because they are based on subjective probability.

- For example, a Bayesian can compute an interval in which there is 95% probability (as a degree of belief) that μ is in the interval.
- Note that in this case the endpoints of the interval are values computed from data.

Here's a conversation between the same employer, but with a Bayesian.

A Conversation between a Bayesian Consultant and his insightful Employer

Employer: So, do you have the results of the drug trial analyses?

Bayesian: Yes, I do.

Employer: Let's hear it.

Bayesian: Okay. I computed a 95% credible interval for the average decrease in diastolic blood pressure from taking the medication, and I got $(10.04,\ 14.83)$ mm Hg.

Employer: Great! What does that mean?

Bayesian: It means we should be 95% certain that the true mean decrease is between 10.04 and 14.83 mm Hg.

Employer: Oh, you mean that there is a 95% probability that the mean decrease is between 10.04 and 14.83. Got it!

Bayesian: Exactly!

THE END!

A p-value example: (Lindley and Philips, 1976)

A coin has a probability θ of landing heads. Want to test whether the coin is fair (versus the alternative that the coin lands heads with greater probability), that is, whether $\theta = \frac{1}{2}$.

$\mathbf{H}_o : \theta = \frac{1}{2}$

$\mathbf{H}_a : \theta > \frac{1}{2}$

Will test at the $\alpha = 0.05$ significance level.

The coin is flipped 12 times independently, and we observe 9 heads and 3 tails.

$$\text{H, T, H, H, H, H, H, H, T, H, H, T}$$

Want to compute a p-value and compare to 0.05.

- If $p \leq 0.05$, then we reject $\mathbf{H}_o$ in favor of $\mathbf{H}_a$.
- If $p > 0.05$, then we do not have enough evidence to reject $\mathbf{H}_o$.

Out of 12 independent coin tosses, let X be the number times the coin lands heads. So X follows a Binomial distribution.

$$\begin{aligned} p\text{-value} &= \mathrm{P}\left(X \geq 9 \,\middle|\, \theta = \frac{1}{2}\right) \\ &= \sum_{x=9}^{12} \binom{12}{x} \left(\frac{1}{2}\right)^{x} \left(1 - \frac{1}{2}\right)^{12-x} = \boxed{0.075}. \end{aligned}$$

Because p-value > 0.05, we cannot reject $\mathbf{H}_o$ at the 0.05 significance level.

But there is something I didn't tell you...

The way in which I obtained 9 heads and 3 tails was that I flipped the coins until the third tail appeared.

$$\text{H, T, H, H, H, H, H, H, T, H, H, T}$$

Let Y be the number of times I flip the coin until the third tail appears. Then Y has a negative-binomial distribution with

$$\mathrm{P}(Y = y \mid \theta) = \binom{y-1}{2} \theta^{y-3} (1-\theta)^3.$$

The p-value assuming $\theta = \frac{1}{2}$ under $\mathbf{H}_o$ is

$$\begin{aligned} p\text{-value} &= \mathrm{P}\left(Y \geq 12 \,\middle|\, \theta = \frac{1}{2}\right) \\ &= \sum_{y=12}^{\infty} \binom{y-1}{2} \left(\frac{1}{2}\right)^{y-3} \left(1 - \frac{1}{2}\right)^{3} = \boxed{0.0325}. \end{aligned}$$

With this p-value, we can reject $\mathbf{H}_o$.

What just happened?

- In each case, even though the data and coin flip model were identical, we obtained different conclusions.

- The difference came about in what was considered "as or more extreme" than the observed data.
- Thus, in the classical approach, because the conclusions of a hypothesis test depend on data that we do not observe, strange results can occasionally occur.

  This does not happen in Bayesian statistical inference.

Simple but practical example: Back to the coffee connoisseur

- Suppose p is the probability the coffee connoisseur correctly guesses whether cream is poured in coffee first, or vice versa.
- We observed 10 correct guesses out of 10 tries.
- The classical point estimate of p is the sample proportion, that is $\hat{p} = 10/10 = 1$ (with a standard error of 0).

This means that our best guess is that the coffee connoisseur is 100% accurate.

How can that be?

Bayes theorem (or Bayes Rule):

Before introducing the Bayesian approach to statistics, it is worthwhile reviewing Bayes theorem for discrete events.

Suppose A and B are two events where A temporally or causally comes before B.

Example: A is the event that you attend this lecture or watch the video, and B is the event that you obtain a perfect score on your Bayesian homework.

The use of Bayes Rule is to determine the probability of a "past" or "cause" event, given a current event has happened.

More formally, we use Bayes Rule to obtain $\mathrm{P}(A|B)$. This is a "detection" probability – given what we now know (B), what is the probability of a suspected event in the past (A)?

This is given by Bayes Rule:

$$\Pr(A|B) = \frac{\Pr(A)\Pr(B|A)}{\Pr(A)\Pr(B|A) + \Pr(A^c)\Pr(B|A^c)}$$

Usually $\Pr(A)$, $\Pr(B|A)$ and $\Pr(B|A^c)$ can be determined easily.

Example: Two coins, one fair and one double-headed, are placed in a box. One coin is chosen at random, and flipped. If it lands heads, what is the probability it is the double-headed coin?

Answer: Let

$$\begin{aligned} A &= \text{The double-headed coin is selected} \\ B &= \text{The coin selected lands heads} \end{aligned}$$

Can write down

$$\begin{aligned} \Pr(A) &= 1/2 \\ \Pr(B|A) &= 1 \\ \Pr(B|A^c) &= 1/2 \end{aligned}$$

So, by Bayes Rule,

$$\begin{aligned} \Pr(A|B) &= \frac{\Pr(A)\Pr(B|A)}{\Pr(A)\Pr(B|A) + \Pr(A^c)\Pr(B|A^c)} \\ &= \frac{(1/2)(1)}{(1/2)(1) + (1/2)(1/2)} = \frac{2}{3} \end{aligned}$$

The probability that the double-headed coin was selected (the past event) given that the coin landed heads (the current event) is $\frac{2}{3}$.

Application of Bayes Rule: Naive Bayes Spam Filter

Goal: Want to develop a way to assess the probability a new incoming e-mail is spam.

Assumptions:

- You have a large collection of e-mails to develop your method. From these e-mails, you can easily characterize as "spam" versus "not spam."
- You have a program that can search for occurrences of words in your e-mails.

Suppose we identify 10 word phrases common in spam. For example,

- earn money, click to remove, visit our website, stay in shape, act now, free membership

Let W_j be the event that the j-th phrase appears in the e-mail. Let W_j^c (complement of W_j) denote the event that the j-th phrase is *not* in the e-mail.

Assume that an e-mail contains the 1st and 10th words, but none of the others. Want to compute or estimate

$$\Pr(\text{spam} \mid W_1, W_2^c, \ldots, W_9^c, W_{10}).$$

By Bayes Rule,

$$\Pr(\text{spam} \mid W_1, W_2^c, \ldots, W_9^c, W_{10}) =$$

$$\frac{\Pr(W_1, W_2^c, \ldots, W_9^c, W_{10} \mid \text{spam})\Pr(\text{spam})}{\Pr(W_1, W_2^c, \ldots, W_9^c, W_{10} \mid \text{spam})\Pr(\text{spam}) + \Pr(W_1, W_2^c, \ldots, W_9^c, W_{10} \mid \text{spam}^c)\Pr(\text{spam}^c)}.$$

We can empirically estimate $\Pr(\text{spam})$ from the sample proportion of spam you receive.

Also, $\Pr(\text{spam}^c) = 1 - \Pr(\text{spam})$.

Estimating

$$\Pr(W_1, W_2^c, \ldots, W_9^c, W_{10} \mid \text{spam})$$

and

$$\Pr(W_1, W_2^c, \ldots, W_9^c, W_{10} \mid \text{spam}^c)$$

can also in principle be accomplished empirically by finding the sample proportion of e-mails in your development sample with the 1st and 10th phrase in the note, but none of the others.

Problem: May be very few e-mails with exactly the 1st and 10th phrase in the note.

This problem is even worse if we considered, say, 100 spam phrases.

Possible solution: Naive Bayes

Use the following approximation:

$$\Pr(W_1, W_2^c, \ldots, W_9^c, W_{10} \mid \text{spam}) \approx \Pr(W_1|\text{spam}) \Pr(W_2^c|\text{spam}) \cdots \Pr(W_9^c|\text{spam}) \Pr(W_{10}|\text{spam})$$

$$\Pr(W_1, W_2^c, \ldots, W_9^c, W_{10} \mid \text{spam}^c) \approx \Pr(W_1|\text{spam}^c) \Pr(W_2^c|\text{spam}^c) \cdots \Pr(W_9^c|\text{spam}^c) \Pr(W_{10}|\text{spam}^c)$$

This approximation assumes that the 10 phrase occurrences are *conditionally independent* given the spam status of each e-mail.

This is a huge simplification because it ignores the possibility that some spam phrases tend to travel together.

But this approach has the following advantages:

- Estimates of $\Pr(W_j|\text{spam})$ and $\Pr(W_j|\text{spam}^c)$ are likely to be accurate.
- Easy to compute.
- If the occurrences of pairs, triples, etc., of spam phrases are not very highly dependent, assuming conditional independence can give accurate estimates of combinations of phrases that are not observed in the development sample of e-mails.

See `http://homes.cs.washington.edu/~pedrod/papers/cacm12.pdf` to see how Naive Bayes can outperform state-of-the-art learner.

Can implement Naive Bayes using functions in the `sklearn.naive_bayes` module.

From Bayes Rule to Bayesian statistics:

When we set up probability models for data, we specify

- Unknown model parameters
- How data are generated based on the model parameters

Can think of the model parameters coming first, and data coming second (based on the values of the model parameters).

Bayes theorem provides the tool to compute the probability of model parameter values based on the observed data.

In Bayesian statistics, all quantities (data, unknown parameters) have probability distributions to describe uncertainty.

Parameters: Uncertainty in model parameters are described through probability distributions $p(\theta)$. For instance, we may believe *a priori* that the mean adult male weight is uniformly distributed from 160 lbs to 200 lbs.

Data: Observations obtained from a data-generating mechanism are assumed to follow a probability distribution $p(y|\theta)$. For example, we may assume adult male weights are normally distributed around an unknown mean.

Note that assuming a probability distribution on a parameter does not mean that the parameter is "random." It is actually a fixed value, but our uncertainty about it is represented in the form of a probability distribution.

Summary of the Bayesian approach:

A typical Bayesian analysis can be outlined in the following steps.

1. Formulate a probability model for the data.
2. Decide on a prior distribution for the unknown model parameters.
3. Observe the data, and construct the likelihood function based on the data.
4. Determine the posterior distribution.
5. Summarize important features of the posterior distribution, or calculate quantities of interest based on the posterior distribution.

Keeping your eye on the ball:

The main goal of Bayesian inference is to obtain the **posterior distribution** of the unknown parameters. This is the probability distribution that describes the state of knowledge about the parameters once the data have been observed.

After the posterior distribution has been determined, just about any inferential question can be answered.

Simple example to illustrate steps in Bayesian analysis:

Suppose we are examining patterns of 30-day mortality of heart attack patients at a local hospital.

- Among 5 randomly selected heart attack patients, 1 died and 4 survived beyond 30 days.
- Let θ be the probability a patient dies within 30 days after admission for a heart attack.
- We want to make inferences about θ from the data.

First step: Formulate the probability model

For $i = 1, \ldots, 5$, let

$$Y_i = \begin{cases} 0 & \text{if the } i\text{-th patient survives 30 days} \\ 1 & \text{if the } i\text{-th patient dies within 30 days} \end{cases}$$

Probability model for the Y_i:

$$\Pr(Y_i = y \mid \theta) = \begin{cases} \theta & \text{for } y = 1 \\ 1 - \theta & \text{for } y = 0 \end{cases}$$

More compactly,

$$\mathrm{P}(Y_i = y \mid \theta) = p(y|\theta) = \theta^y (1 - \theta)^{1-y},$$

for $y = 0, 1$. This is a Bernoulli model for the data.

Second step: Choosing a prior distribution

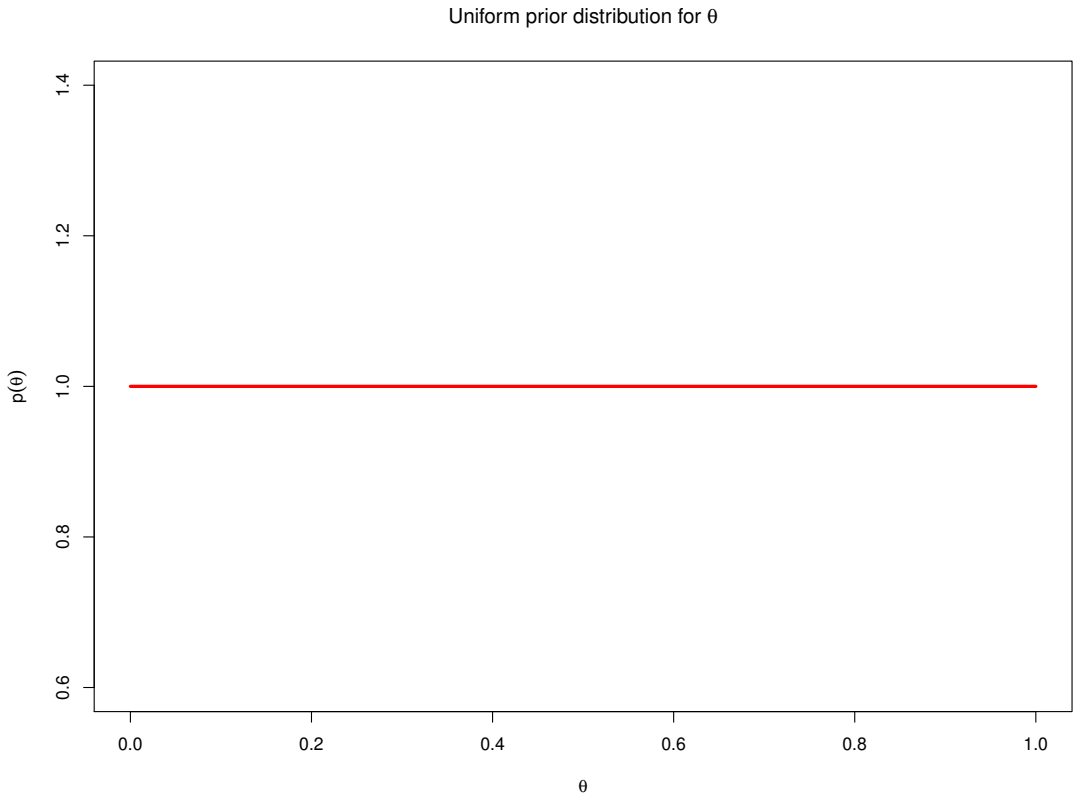
The prior distribution of the unknown parameter(s) represents the state of knowledge prior to observing the data.

For this setting, let us assume that all values of θ between 0 and 1 are equally believable.

Formally, this translates to

$$p(\theta) = 1$$

for $0 \leq \theta \leq 1$.



Third step: Construct the likelihood function

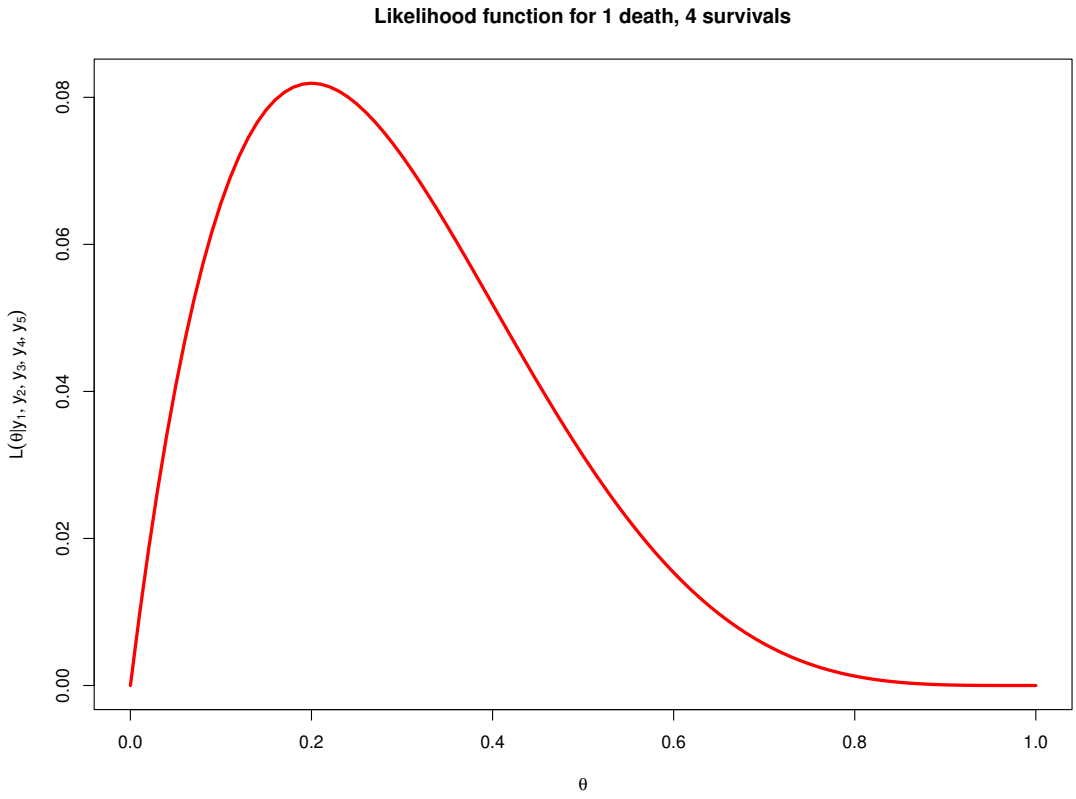
The likelihood function is the probability of the data (probability density, for continuous outcomes) conditional on the parameter(s), viewed as a function of the parameter(s).

$$p(y_1, y_2, y_3, y_4, y_5 \mid \theta) = \prod_{i=1}^{5} \theta^{y_i}(1-\theta)^{1-y_i} = \theta(1-\theta)^4.$$

Thus, the likelihood function is

$$L(\theta|\boldsymbol{y}) = \theta(1-\theta)^4,$$

where $\boldsymbol{y} = (y_1, y_2, \ldots, y_5)$ are the five binary outcomes.



Fourth step: Determine the posterior distribution

The posterior distribution is the probability distribution of the unknown parameter(s) conditional on the observed data.

This is determined by Bayes rule:

$$p(\theta \mid \boldsymbol{y}) = \frac{p(\theta)p(\boldsymbol{y} \mid \theta)}{\int p(\theta)p(\boldsymbol{y} \mid \theta)\, d\theta} = \frac{p(\theta)p(\boldsymbol{y} \mid \theta)}{p(\boldsymbol{y})}$$

$$\propto \quad p(\theta)p(\boldsymbol{y} \mid \theta) \propto p(\theta)L(\theta \mid \boldsymbol{y})$$

where "$\propto$" means "is proportional to."

The key tool in Bayesian statistics:

$$\boxed{p(\theta \mid y) \propto p(\theta)L(\theta \mid y)}$$

Posterior $\propto$ Prior $\times$ Likelihood

Applying the key tool in our setting:

We have

$$\begin{aligned} p(\theta) &= 1 \\ L(\theta \mid \boldsymbol{y}) &= \theta(1-\theta)^4 \end{aligned}$$

This means

$$p(\theta \mid \boldsymbol{y}) \propto p(\theta) L(\theta \mid \boldsymbol{y}) = (1)\theta(1-\theta)^4 = \theta(1-\theta)^4$$

This in turn means that

$$p(\theta \mid \boldsymbol{y}) = c\theta(1-\theta)^4$$

for some constant c (that does not depend on θ).

It turns out

- $p(\theta) = 30\ \theta(1-\theta)^4$ integrates to 1, so that $c = 30$ in this case.
- This is the density for a Beta distribution with parameters 2 and 5 (one more than the exponents of θ and $(1-\theta)$, respectively), abbreviated

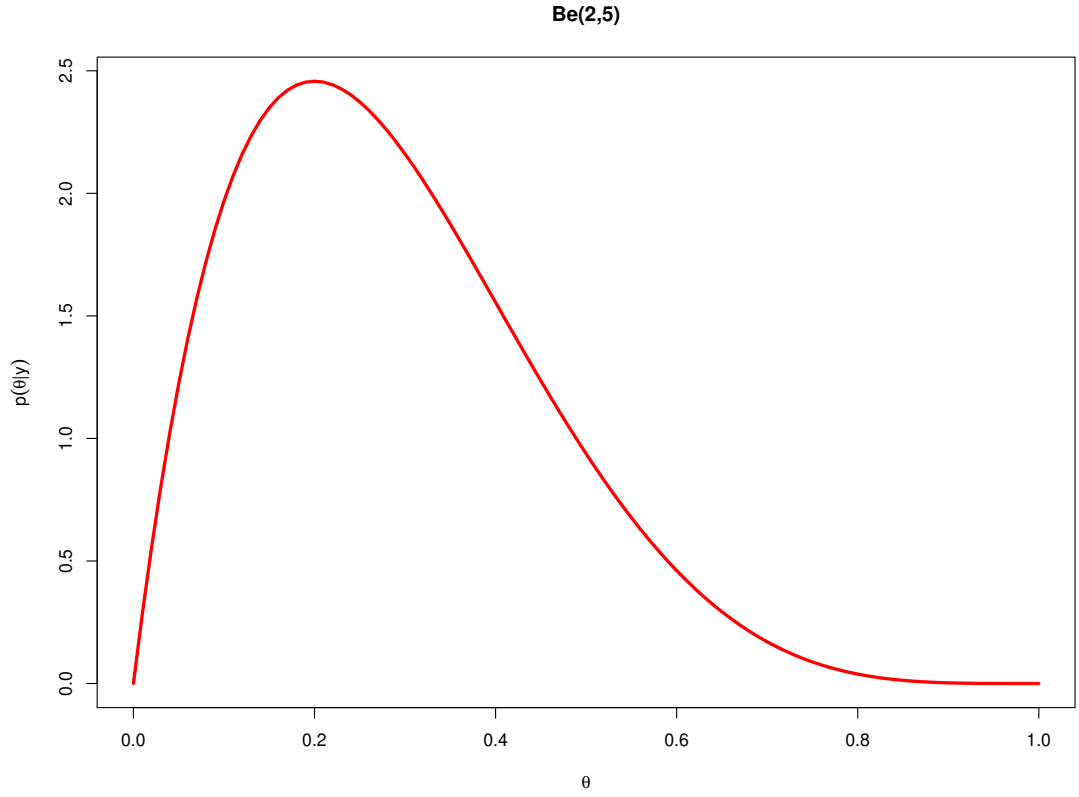
$$\theta \sim \text{Be}(2, 5).$$

This is a probability distribution used frequently by Bayesians.

For general Beta distributions, write $\text{Be}(\alpha, \beta)$.

Some properties of the Beta distribution:

$$\begin{aligned} \text{E}(\theta|\alpha,\beta) &= \frac{\alpha}{\alpha+\beta} \\ \text{Var}(\theta|\alpha,\beta) &= \frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)} \\ \text{Mode}(\theta|\alpha,\beta) &= \frac{\alpha-1}{\alpha+\beta-2} \end{aligned}$$



Fifth step: Summarizing the posterior distribution

- Posterior mean, $\mathrm{E}(\theta|\boldsymbol{y})$.
- Posterior mode (value of θ that maximizes $p(\theta \mid \boldsymbol{y})$).
- Central posterior interval for θ.
- Highest posterior density (HPD) region for θ – shortest interval with specified probability. Usually more difficult to compute than central intervals.

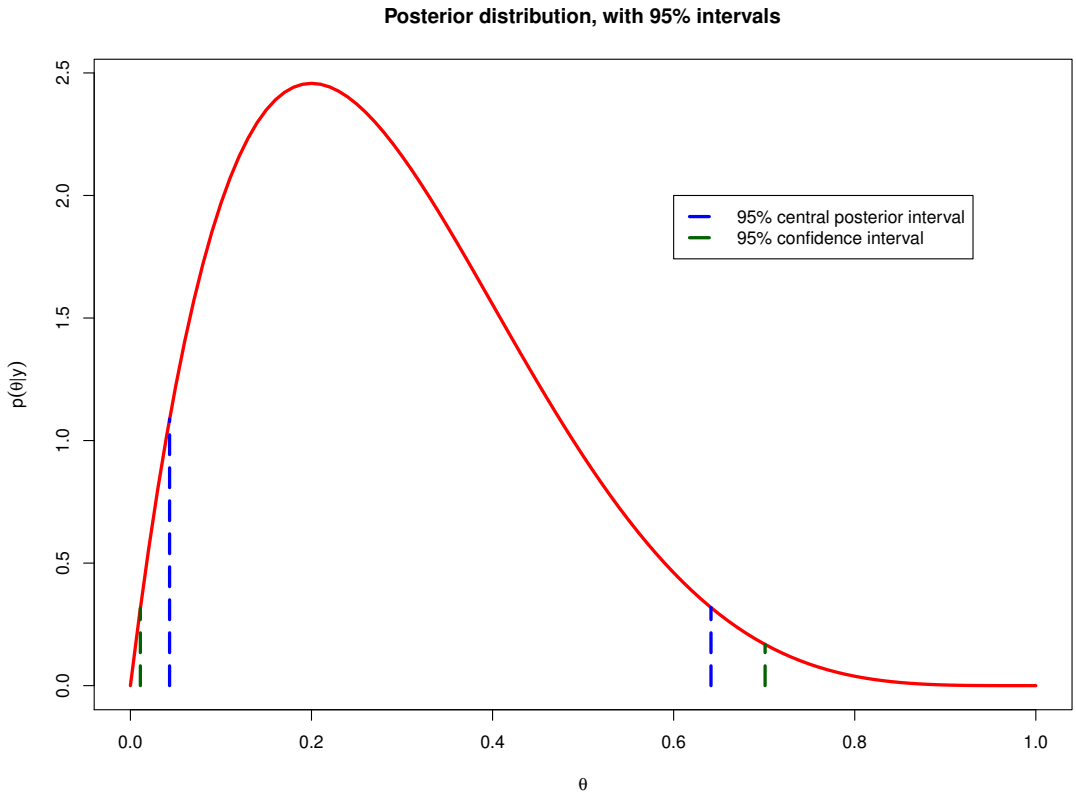
Posterior summaries for our setting:

Based on properties of the Beta distribution

- $\mathrm{E}(\theta|\boldsymbol{y}) = \frac{2}{2+5} \approx 0.286$
- $\text{mode}(\theta|\boldsymbol{y}) = \frac{2-1}{2+5-2} = 0.2$
- 95% central posterior interval: Can compute the 2.5%-ile and 97.5%-ile numerically.

$$(0.0433,\ 0.641)$$

Compare to the Frequentist 95% confidence interval $(0.011,\ 0.701)$.



Comments:

- The Bayesian approach recognizes the asymmetry in inferences about θ, whereas the standard Frequentist approach does not.
- The same approach outlined above applies identically to multi-parameter models.
- Frequentist approach can be improved, but need to use fancier machinery, e.g., the bootstrap, but even that is not often reliable with small samples.

Bayesian inference as a learning model:

The Bayesian approach facilitates the idea that one learns about the state of the world as data are accumulated.

Sequential nature of Bayesian analysis:

"Yesterday's posterior is today's prior"

Instead of performing a single analysis with the 5 observations, I could have performed (up to) 5

analyses in sequence:

Suppose $y_1 = 0$, $y_2 = 0$, $y_3 = 1$, $y_4 = 0$, and $y_5 = 0$.

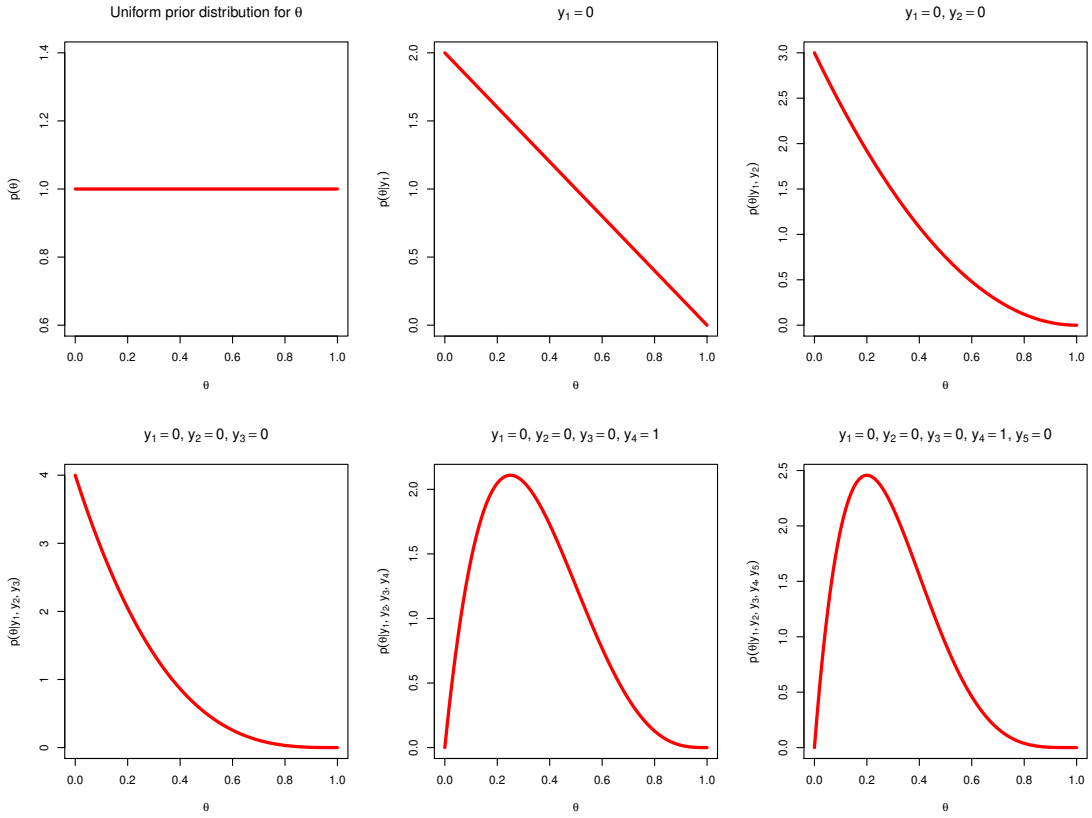
Start with $p(\theta)$.

Observe $y_1 = 0$, and determine $p(\theta \mid y_1 = 0)$.

Now use $p(\theta \mid y_1)$ as the prior distribution for the second observation.

Observe $y_2 = 0$, and determine $p(\theta \mid y_1 = 0, y_2 = 0)$.

And so on! You will obtain the same posterior distribution as analyzing the data simultaneously.



Modern Bayesian approach to data analyses:

The previous discussion laid out the foundations of the Bayesian approach, but more complex modeling situations can present great challenges.

- The posterior density for most real-life models can be time-consuming to determine.
- Even if one can write down the posterior density, it is almost always the case that it is difficult to summarize using standard analytic tools.

The modern approach: Summarizing posterior distributions through Monte Carlo simulation

Monte Carlo simulation:

The posterior distribution is a probability distribution, and the goal of a Bayesian analysis is to study the posterior distribution, or derive summaries from the posterior distribution.

Summarizing computer-simulated values from the posterior distribution is a legitimate alternative to analytic summaries.

Monte Carlo summaries of probability distributions:

- Rather than go through laborious calculations to obtain the mean, median, variance, percentiles, etc., of a probability distribution, one can calculate the sample-version of the distribution from the simulated sample.
- Usually need to simulate a very large sample in order to precisely approximate the generating distribution.

Example: Summarizing Be$(2, 5)$ distribution

Rather than report the analytically-determined summaries of the Beta posterior distribution for the 30-day mortality example, do the following:

1. Simulate 10,000 values from Be$(2, 5)$.
2. Report *sample* summaries from the distribution of 10,000 simulated values.

```
# simulate from Be(2,5)
theta = np.random.beta(2,5,10000) # generate 10,000 values from Be(2,5)
print(theta[0:10]) # first ten simulated values
print(np.mean(theta)) # sample mean
# 2.5% and 97.5% of empirical distn
print(np.percentile(theta,[2.5,97.5]))
```

```
[0.12471433 0.38933883 0.1296898  0.23946542 0.55924411 0.23182827
 0.13178348 0.29691004 0.27515282 0.31043925]
0.28478820646628294
[0.04471399 0.63485071]
```

Recall that E$(\theta|\alpha = 2, \beta = 5) = 0.286$ and the 95% central posterior interval was $(0.0433,\ 0.641)$.

Predictive inference:

One of the real strengths of the Bayesian setting is the ability to produce model-based predictions that accounts for the uncertainty in parameter inferences.

Goal: Want to determine $p(\tilde{y} \mid \boldsymbol{y})$, that is, the probability distribution for a new value $\tilde{y}$ given the data we already analyzed. This distribution is called the **posterior predictive distribution**.

Notice that θ is no longer in the expression – the posterior predictive distribution "averages out" the uncertainty about θ.

Can show analytically using tools of calculus that

$$p(\tilde{y} \mid \boldsymbol{y}) = \begin{cases} 5/7 & \text{if } \tilde{y} = 0 \\ 2/7 & \text{if } \tilde{y} = 1 \end{cases}$$

Thus we can conclude that there is a $2/7$ chance that a new (randomly selected) patient entering the hospital with a heart attack would die within 30 days.

Computing $p(\tilde{y} \mid \boldsymbol{y})$ via simulation:

This method highlights the power of simulation-based inference.

1. Generate (say) 10,000 simulated parameter values from the posterior distribution. Call these $\theta^{(1)}, \ldots, \theta^{(10000)}$.
2. For each $j = 1, \ldots, 10000$ separately, generate a value of $\tilde{y}^{(j)}$ from the probability model with parameter value $\theta^{(j)}$.
3. Summarize features of the 10,000 values $\tilde{y}^{(1)}, \ldots, \tilde{y}^{(10000)}$ for predictions.

Example with 30-day mortality:

The posterior distribution is $\theta \sim \text{Be}(2, 5)$. Also, $\Pr(\tilde{y} = 1|\theta) = \theta$.

```
# simulate y from posterior predictive distribution
theta = np.random.beta(2,5,10000) # generate 10,000 values from Be(2,5)
print(theta[0:10]) # first ten simulated values
# generate 10,000 binary values with different probabilities
y = np.random.binomial(1,theta,10000)
print(y[0:10]) # first eleven values
print(np.bincount(y)/10000)  # frequency of 0 and 1
```

```
[0.36309086 0.05705801 0.05409083 0.5388881  0.49666649 0.30191721
 0.3444435  0.08902907 0.08247285 0.21758449]
[0 0 0 1 0 0 1 1 0 0]
[0.7083 0.2917]
```

Deeper discussion on prior distributions:

A main difference operationally between Frequentist methods and Bayesian methods is the incorporation of a prior distribution.

What are the guidelines for choosing a prior distribution?

Two types of prior distributions: Informative versus non-informative

Informative prior distributions: The statistician uses his/her knowledge about the substantive problem, along with elicited expert opinion if possible, to construct a prior distribution that properly reflects prior beliefs about the unknown parameters.

Non-informative prior distributions: The statistician chooses a distribution in an attempt to be objective, acting as though no prior knowledge about the parameters exists before observing the data.

When one can construct a defensible informative prior distribution, this is the best and most scientific approach.

Criticisms of assuming an informative prior distribution:

1. Two different Bayesians could end up using two different informative prior distributions, and would therefore obtain two different posterior distributions.
2. Some argue informative prior distributions are not "objective" or "scientific."

Noninformative prior distributions:

Also termed "vague," "diffuse," and "objective."

It is often desirable to have prior distributions that

- formally express ignorance, and
- can be viewed as default choices when no prior knowledge is available.

Main desired property of choosing an objective prior distribution:

- Assigns "equal probability" (or at least approximately equal) to all values of the parameters.

That is, the prior density should be fairly flat, and that the likelihood should overwhelm the prior density with even small amounts of data.

Problems with objective prior distributions:

1. For most problems, there is no unique non-informative prior distribution.
2. Any method for constructing a non-informative prior distribution should be invariant to the scale of the parameter. This is a difficult property to satisfy in practice.
3. Common methods for constructing non-informative prior distributions result in "improper" probability distributions.

So being Bayesian is not necessarily the perfect solution.

Generative models:

One view of statistical prediction:

- Learn the relationship between a response y and predictors/features x through a machine learning algorithm.
- Apply the estimated relationship for making predictions.

An alternative view: :

- Propose a probabilistic mechanism in which data y are generated given x.
- Learn characteristics of this mechanism to be able to simulate values of y given x.

This latter view involves framing a prediction problem in the context of a generative model.

Most general view: A generative model is a probabilistic mechanism f that (possibly in conjunction with given features x) generates response values y, and that f itself is produced by a probabilistic mechanism.

What is the connection to Bayesian statistics?

Rather than keep f general, consider the following setup.

$$\begin{aligned} \theta &\sim p(\theta) && \text{Prior distribution} \\ y|\theta, x &\sim p(y|\theta, x) && \text{Data probability model} \end{aligned}$$

- As a special case of a generative model, $p(y|\theta, x)$ plays the role of f in which θ is the unknown aspect of the generating mechanism.
- $p(\theta)$ describes the probabilistic generation of θ

An appealing aspect to this approach is that it provides a pathway to generating simulated data once a model has been fit. Again, this is accomplished via the posterior predictive distribution:

As a reminder,

1. Obtain the posterior distribution, $p(\theta|y, x)$.
2. Simulate values of θ from $p(\theta|y, x)$.
3. For each simulated θ, generate $\tilde{y}$ from $p(\tilde{y}|\theta, \tilde{x})$ where $\tilde{x}$ is the set of features observed for the new value.

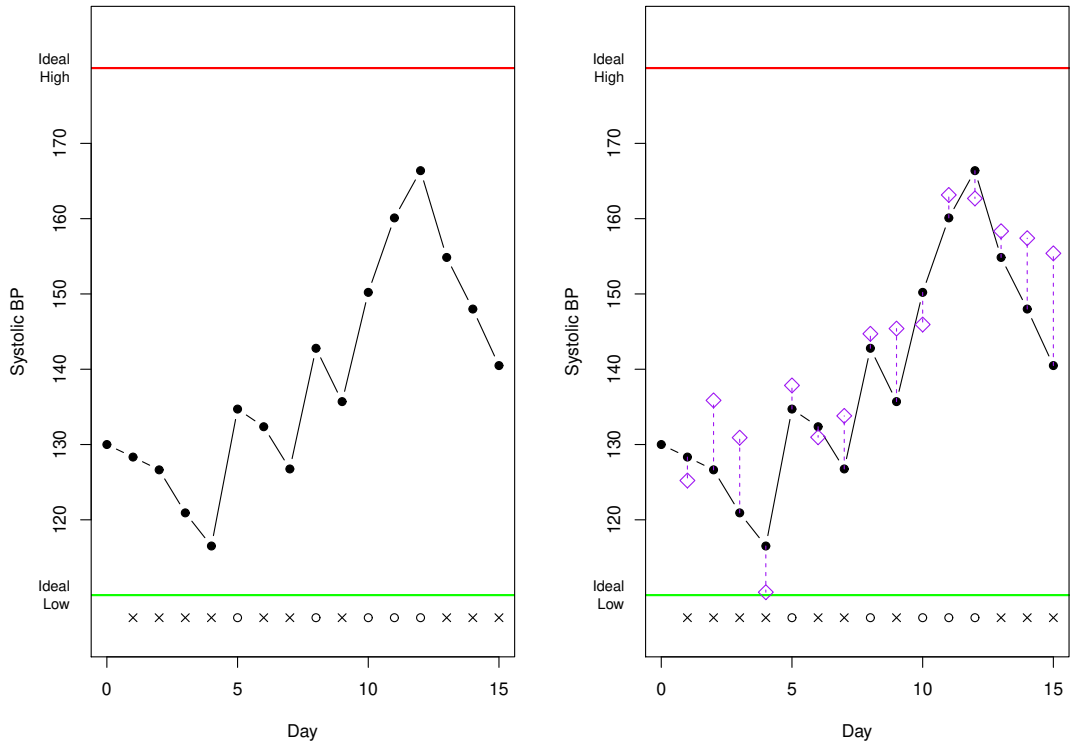
Most common type of generative models: Latent variable models

These models typically have complex probabilistic structure on θ. That is, we assume

$$\begin{aligned} \delta &\sim p(\delta) && \text{(prior distribution for hyperparameter)} \\ \theta|\delta &\sim p(\theta|\delta) && \text{(prior distribution conditional on } \delta) \\ y|\theta,\delta,x &\sim p(y|\theta,x) && \text{(model for data, conditionally independent of } \delta \text{ given } \theta) \end{aligned}$$

Examples of generative models:

- Hierarchical (linear) models – to be seen shortly
- Latent Dirichlet Allocation (LDA) – to be seen shortly
- Generative Adversarial Networks (GANs) – to be seen not so shortly
- Hidden Markov models (HMMs) – next slide!



Back to Monte Carlo simulation:

- When posterior densities become more complex, they become very difficult to summarize.

- If the posterior density is non-standard, then straight Monte Carlo simulation can be difficult as well.

In the late 1980s, a few statisticians (Alan Gelfand, Adrian Smith, and others) stumbled on an idea that revolutionized Bayesian statistics.

Main innovation:

Rather than directly simulate from $p(\boldsymbol{\theta}|\boldsymbol{y})$, simulate a random walk in the space of $\boldsymbol{\theta}$ which converges to $p(\boldsymbol{\theta}|\boldsymbol{y})$.

Markov chain Monte Carlo (MCMC) simulation:

- The key idea is to create a Markov chain whose stationary distribution is $p(\boldsymbol{\theta}|\boldsymbol{y})$. Values are then computer-simulated from the Markov chain.
- Once the Markov chain is run long enough, simulated values can be treated as coming from the posterior distribution.

There are actually many ways to set up such a Markov chain. The most straightforward way is to implement a **Gibbs sampler**.

Implementing an MCMC sampler to obtain simulated parameter values:

1. Run several parallel Gibbs samplers with different starting values (preferably widely-dispersed)
2. Simulate values from the Markov chains for a "burn-in" period (before the Markov chains have converged to the stationary distribution), and discard the burn-in simulations
3. Save simulated values after burn-in period. These will be the simulated values on which to perform inferential summaries.

If I were you, I would be left with a **big headache** because a lot of work still seems necessary to implement a Gibbs sampler.

- Still need to determine analytically the probability distributions for each step in a Gibbs sampler
- Still need to simulate from the probability distributions
- Still need to write code the performs the Markov chain simulation, checks for convergence, etc.

Fortunately, many packages now exist that do the hard work for you.

The two most well-known packages for implementing Bayesian analyses using MCMC are

- WinBUGS, OpenBUGS, and JAGS (BUGS = **B**ayesian inference **U**sing **G**ibbs **S**ampling).
- Stan, which is a lot like WinBUGS, but uses (by default) a different type of Markov chain sampler.

We will explore the use of JAGS ("Just another Gibbs Sampler") called from within python via the `pyjags` library.

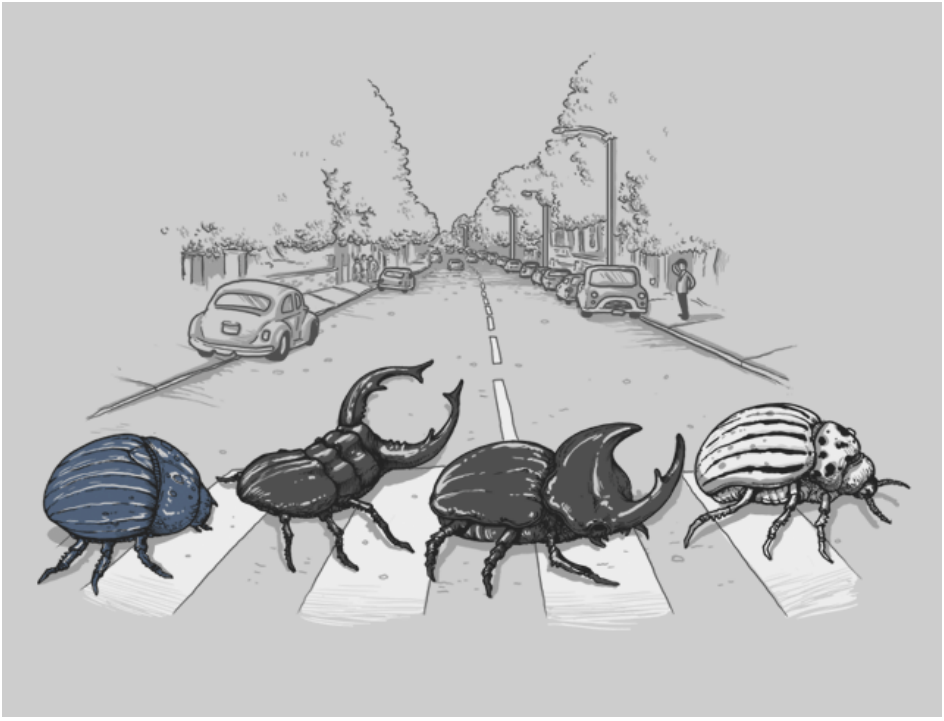
These programs actually figure out the conditional posterior distributions and automatically sample from them!

That's right – they automatically determine and implement the Markov chain!

All you need to do typically is

- specify the generative model for data, and
- specify logistical issues concerning MCMC simulation (starting values, burn-in period, how many iterations to run after burn-in)

Example: The Beetles!



Bayesian logistic regression

A dose-response study was performed that counted the number of beetles killed after 5-hour

exposure to carbon disulphide. The data are shown below

Concentration of carbon disulphide (x_i)	Number of beetles exposed (n_i)	Number killed (y_i)
1.6907	59	6
1.7242	60	13
1.7552	62	18
1.7842	56	28
1.8113	63	52
1.8369	59	52
1.8610	62	61
1.8839	60	60

Model for probability of beetle death:

For $i = 1, \ldots, N$, where $N = 8$ observations,

$$\begin{aligned} y_i &\sim \text{Bin}(n_i,\ p_i) \\ \text{logit}\ p_i &= \alpha + \beta x_i \end{aligned}$$

Not necessary, but slightly better computationally to center the x_i:

$$\text{logit}\ p_i = \alpha^* + \beta(x_i - \bar{x})$$

where

$$\begin{aligned} \bar{x} &= \frac{1}{N}\sum_{i=1}^{N} x_i \\ \alpha^* &= \alpha + \beta\bar{x} \end{aligned}$$

Prior distribution:

Assume non-informative but proper prior distribution that has independent components

$$\begin{aligned} \alpha^* &\sim \text{N}(0, 10000) \\ \beta &\sim \text{N}(0, 10000) \end{aligned}$$

By Bayes rule, the posterior density for α^* and β can be written as

$$\mathsf{p}(\alpha^*, \beta|\boldsymbol{y}) = c \cdot \text{N}(\alpha^*|0, 10000)\text{N}(\beta|0, 10000)\prod_{i=1}^{8} p_i^{y_i}(1-p_i)^{n_i - y_i}$$

where c is a normalizing constant, and the $\text{N}(\cdot|\cdot, \cdot)$ are normal densities with the given mean and variance.

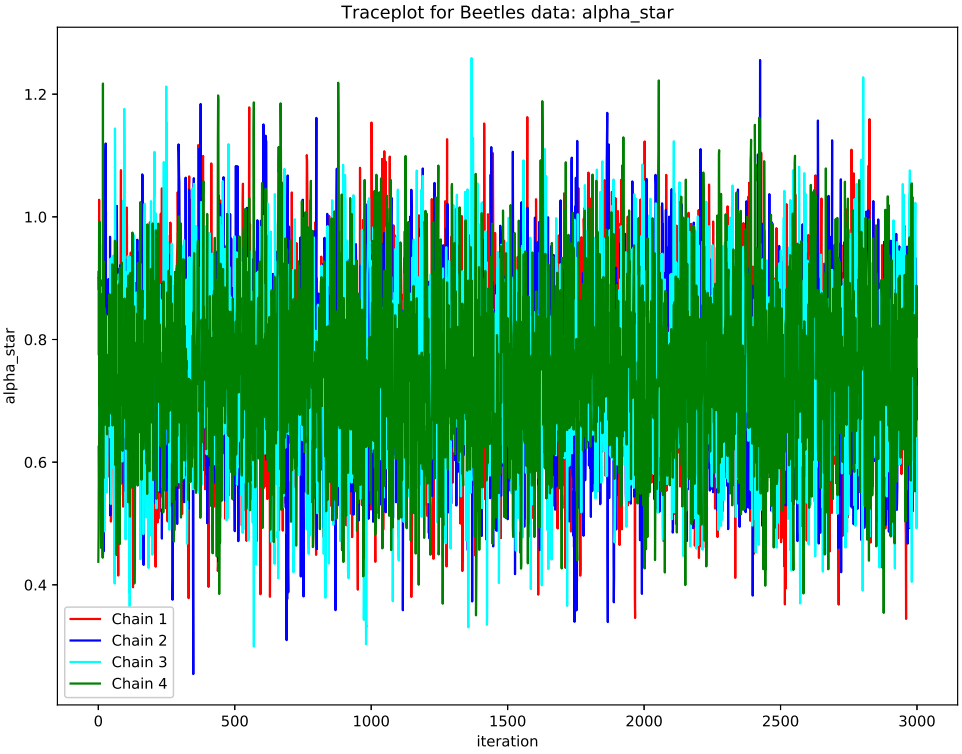
JAGS code: Written as a generative model

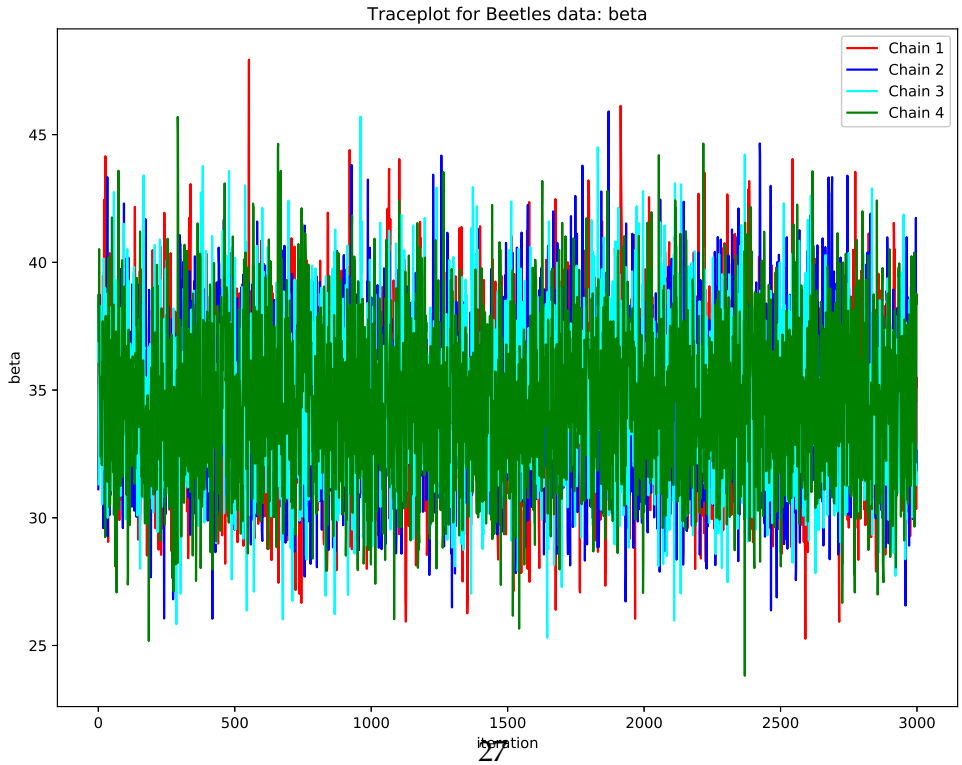
```
model {
mean_x = mean(x);
centered_x = x - mean_x;
alpha = alpha_star - beta*mean_x; ## original alpha
alpha_star ~ dnorm(0.0, 0.0001); ## prior for alpha_star
beta ~ dnorm(0.0, 0.0001); ## prior for beta
linpred = alpha_star + beta * centered_x;
for (i in 1:N)  {
    p[i] = ilogit(linpred[i]);
    yhat[i] = p[i]*n[i];  ## fitted values
    y[i] ~ dbin(p[i],n[i]) ## model for y
  }
}
```

python code calling JAGS code:

```
beetles_x = np.array([1.6907, 1.7242, 1.7552, 1.7842,
  1.8113, 1.8369, 1.8610, 1.8839])
beetles_n = np.array([59, 60, 62, 56, 63, 59, 62, 60])
beetles_y = np.array([6, 13, 18, 28, 52, 53, 61, 60])
beetles_N = 8

beetles_model = pyjags.Model(beetles_code,
  data=dict(x=beetles_x, n=beetles_n, y = beetles_y, N=beetles_N),
  chains=4)
#warmup/burn-in:
beetles_burnin = beetles_model.sample(500, vars=['alpha_star','beta'])
#simulations to save:
beetles_samples = beetles_model.sample(3000, vars=['alpha_star','beta'])
```


Traceplot for Beetles data: alpha_star
alpha_star
iteration
Chain 1
Chain 2
Chain 3
Chain 4
0.4
0.6
0.8
1.0
1.2
0
500
1000
1500
2000
2500
3000


Traceplot for Beetles data: beta
beta
iteration
Chain 1
Chain 2
Chain 3
Chain 4
25
30
35
40
45
0
500
1000
1500
2000
2500
3000

Numerical summaries:

```
            mean     sd    2.5%     25%     50%     75%   97.5%   n_eff   Rhat
alpha_star 0.749  0.139   0.485   0.654   0.746   0.842   1.029  5180.0  1.001
      beta 34.545  3.019  28.948  32.466  34.491  36.493  40.554  4655.0  1.000
```

Rhat:

The `Rhat` statistic is a measure that indicates whether the Markov chain was run long enough to reach convergence.

- Values near 1.0 indicate convergence.
- Large values (say 1.3 or greater) indicate either that the procedure has not been run long enough, or that the parameter itself may be difficult to obtain reasonable samples given strong autocorrelation in the sampler.

Hierarchical linear models:

A class of models that is particularly well-served by the Bayesian framework is hierarchical models.

Typical setup: Observe response data and predictor variables as in the usual regression setting.

Suppose now that the samples are clustered by a grouping variable.

Examples:

- Obtain a sample of patients seeing their doctor for chest pain. Want to measure the number of days until pain goes away as a function of patient characteristics and treatment given.

  The data may be clustered by clinic, and we might expect that the effects of the patient characteristics differ by clinic.
- Obtain a sample of children taking a standardized achievement test. Want to measure the relationship between test score and background variables about each child.

  The data may be clustered by school, and we might expect that the relationship between child characteristics and test score might depend on the school.

Example to implement: Reaction times and sleep deprivation

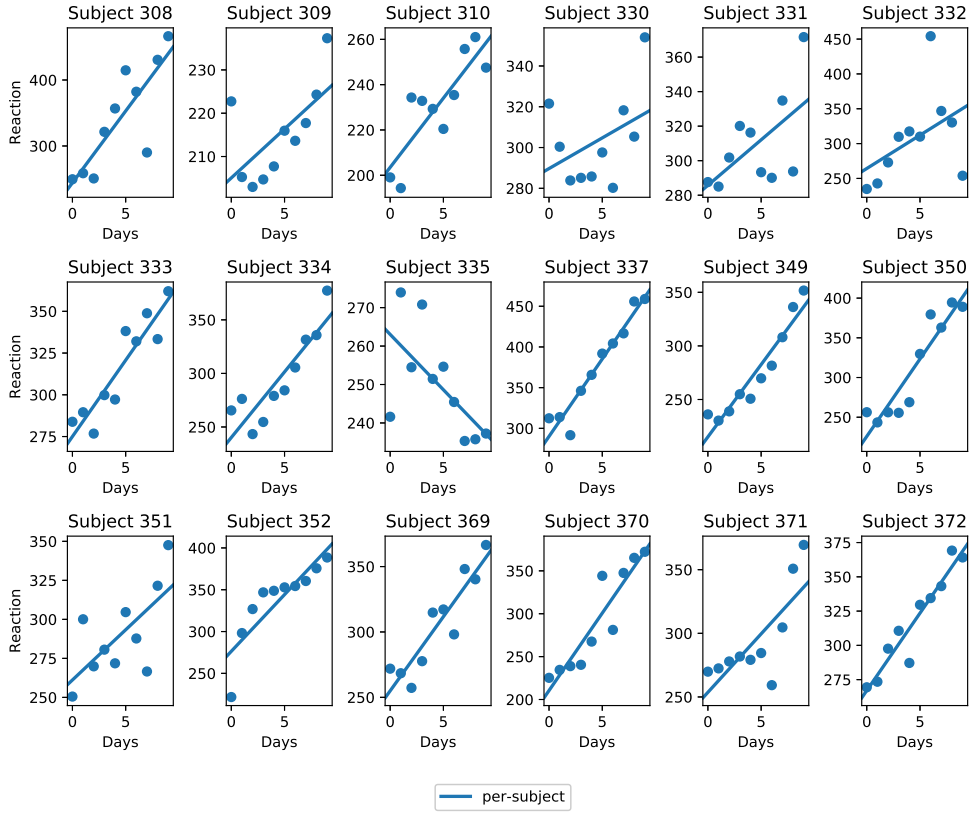
A study published in a 2003 issue of the *Journal of Sleep Research* measured the reaction time per day for 18 subjects in a sleep deprivation study.

On day 0, each subject had a normal amount of sleep. Starting that night, each subject was restricted to 3 hours of sleep.

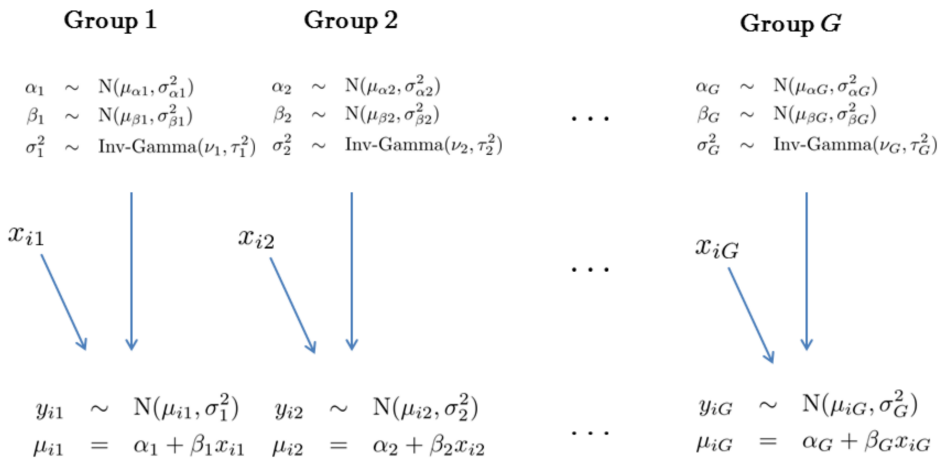
The response represents the average reaction time on a series of tests given to each subject across 10 days.

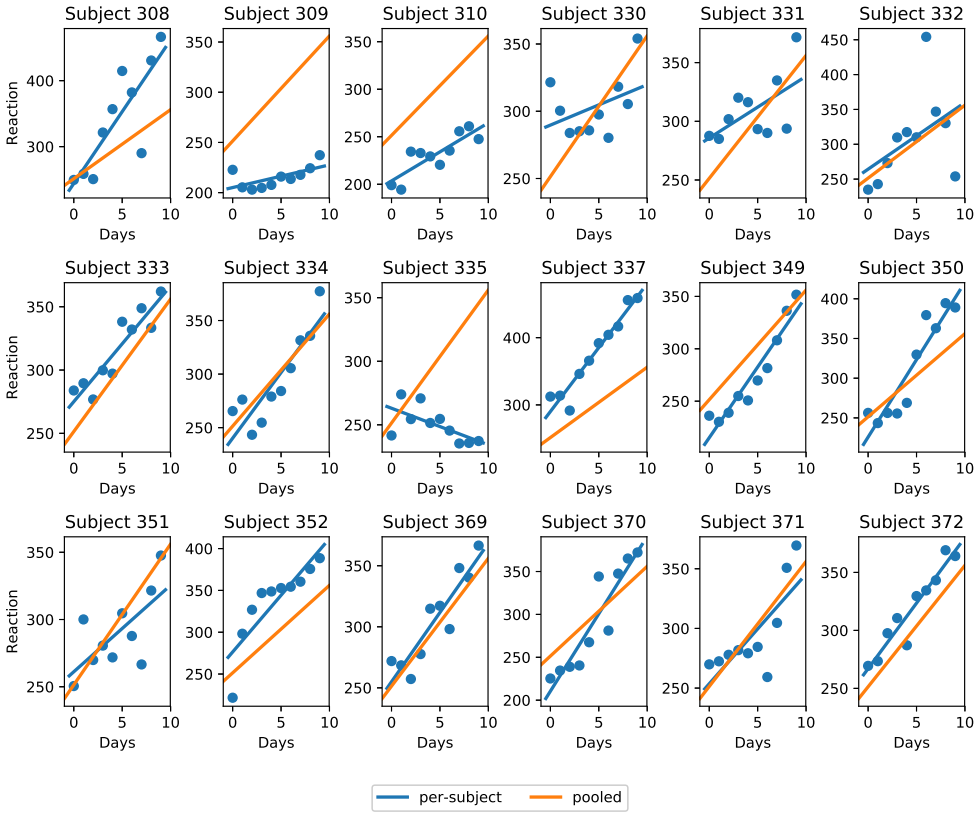
Consider the following two extremes:

- For each subject, fit a separate least-squares regression.
  - Good: Acknowledges that the relationship between x and y may differ within each group.
  - Bad: Very little data on which to estimate effects.
- Pool all the data together and fit a single least-squares regression.
  - Good: Makes full use of the entire data.
  - Bad: Does not recognize differences in effects within groups.



Separate least-squares regressions as a generative model:





Hierarchical modeling: A compromise between separate regressions and one overall regression

Suppose for group $g = 1, \ldots, G$,

$$y_{ig} = \boldsymbol{x}'_{ig}\boldsymbol{\beta}_g + \varepsilon_{ig} = \beta_{0g} + \beta_{1g}x_{i1g} + \cdots + \beta_{Jg}x_{iJg} + \varepsilon_{ig}$$

with $\varepsilon_{ig} \sim \mathrm{N}(0, \sigma^2)$.

Notice that the above model assumes a different $\boldsymbol{\beta}_g$ for each group g.

Now introduce the model component

$$\boldsymbol{\beta}_g \sim \mathrm{N}(\mu_\beta, \Sigma_\beta).$$

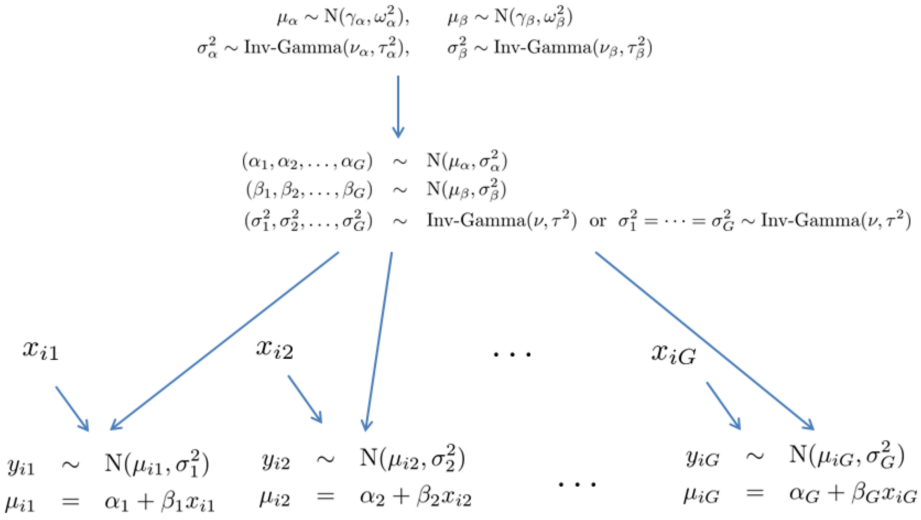
We will also assume a prior distribution on μ_β and Σ_β.

This normal "random effects" distribution $N(\mu_\beta, \Sigma_\beta)$ has an important role:

- The $\boldsymbol{\beta}_g$ are assumed distinct, but come from the same "family" (i.e., $N(\mu_\beta, \Sigma_\beta)$).
- The data across all groups inform the values of μ_β and Σ_β, which in turn means that the $\boldsymbol{\beta}_g$ are informed from other groups besides the data in group g. Sometimes called "partial pooling" of data across groups.
- Can think of the normal distribution acting like a rubber band around the $\boldsymbol{\beta}_g$. They can vary, but the random effects distribution keeps them from being too far apart.

- The random effects distribution "shrinks" the $\boldsymbol{\beta}_g$ to a common population mean.
- Shrinkage tends to be particularly noticeable when some groups have small numbers of observations relative to others.
- Assuming a population distribution on the the $\boldsymbol{\beta}_g$ with unknown covariance Σ_β can be viewed as a form of regularization.

Hierarchical least-squares regressions as a generative model:



JAGS code:

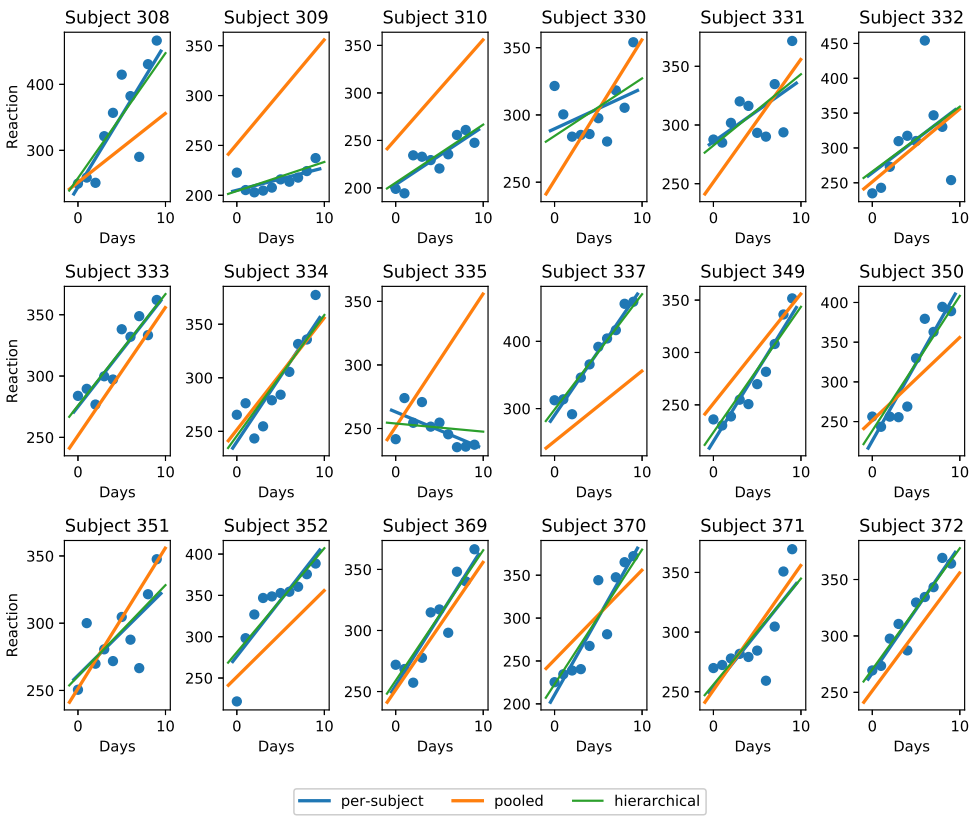
```
model {
mean_x = mean(x);
for (i in 1:N){
    linpred[i] = alpha[id[i]] + beta[id[i]]*x[i];
}
sig2_inv ~ dgamma(0.001, 0.001);
sig = sqrt(1/sig2_inv);
tau2_alpha_inv ~ dgamma(0.001, 0.0001);
tau_alpha = sqrt(1/tau2_alpha_inv);
tau2_beta_inv ~ dgamma(0.001, 0.001);
tau_beta = sqrt(1/tau2_beta_inv);
for (j in 1:N_ID){
    alpha[j] ~ dnorm(300, tau2_alpha_inv);
    beta[j] ~ dnorm(10, tau2_beta_inv);
}
for (i in 1:N){
    y[i] ~ dnorm(linpred[i], sig2_inv);
    }
}
```

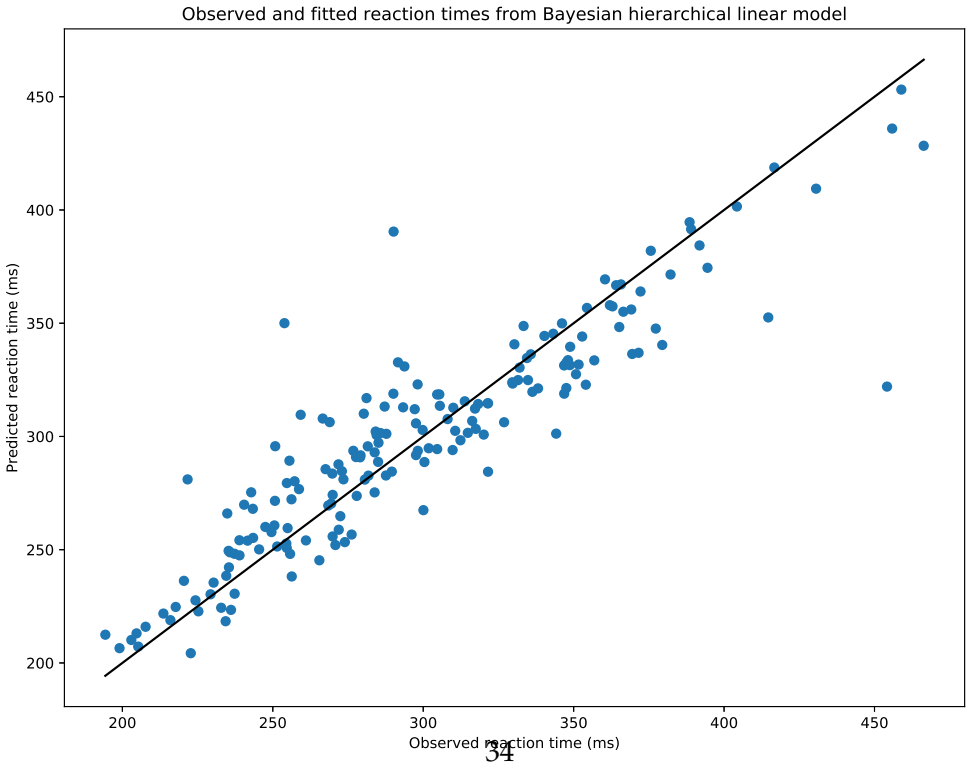
Python code: See pre-processing in ipynb file

```
sleepstudy_model = pyjags.Model(sleepstudy_code,
   data=dict(N = len(sleepstudy_id), N_ID = max(sleepstudy_id),
   y = flatten(np.array(sleepstudy[['Reaction']])),
   id = sleepstudy_id,
   x = flatten(np.array(sleepstudy[['Days']])) ), chains=3)
_ = sleepstudy_model.sample(2000, vars = []) #warmup/burn-in
sleepstudy_samples = sleepstudy_model.sample(5000,
   vars = ["alpha","beta","linpred","y"])
```

Numerical summaries:

```
           mean     sd    2.5%     25%     50%     75%   97.5%  n_eff  Rhat
alpha_1 258.541 14.087 231.090 248.876 258.499 267.955 286.338 2989.0 1.000
alpha_2 204.136 14.088 176.776 194.805 203.945 213.486 231.857 3003.0 1.001
alpha_3 205.991 13.747 178.495 196.808 206.083 215.260 233.073 3186.0 1.000
alpha_4 283.943 13.714 256.813 274.812 284.085 293.055 311.037 3116.0 1.002
alpha_5 282.310 13.846 255.212 273.091 282.422 291.593 309.477 2952.0 1.000
alpha_6 265.814 13.666 239.192 256.722 265.643 275.009 292.780 3099.0 1.001
...
beta_16  15.705  2.587  10.595  13.965  15.712  17.425  20.799 2864.0 1.001
beta_17   8.951  2.468   4.101   7.311   8.948  10.613  13.737 3106.0 1.002
beta_18  10.740  2.451   5.988   9.072  10.733  12.378  15.568 3346.0 1.000
```


Subject 308
Subject 309
Subject 310
Subject 330
Subject 331
Subject 332
Subject 333
Subject 334
Subject 335
Subject 337
Subject 349
Subject 350
Subject 351
Subject 352
Subject 369
Subject 370
Subject 371
Subject 372
Reaction
Days
per-subject
pooled
hierarchical

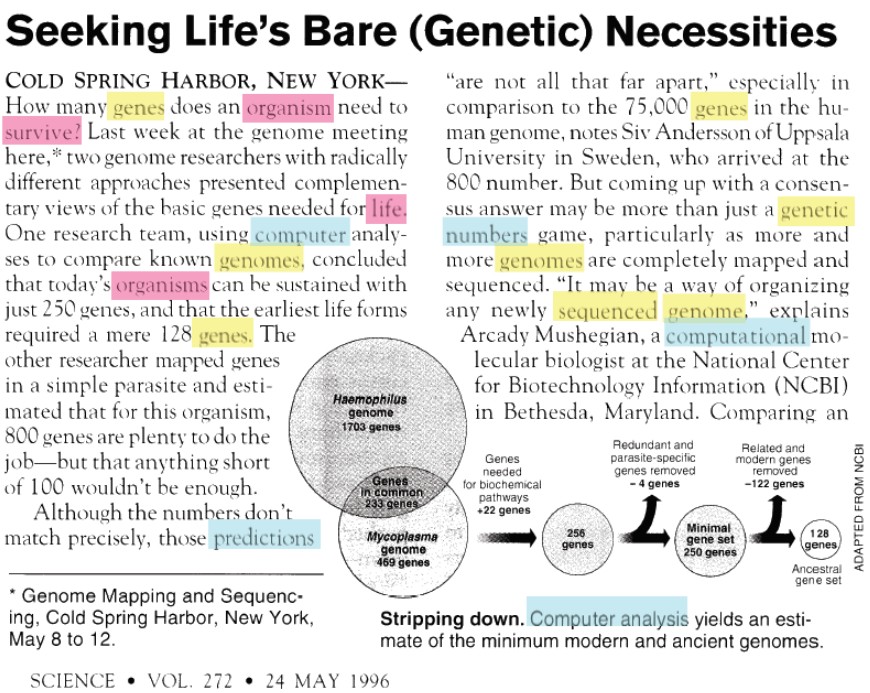

Observed and fitted reaction times from Bayesian hierarchical linear model
Predicted reaction time (ms)
Observed reaction time (ms)

Probabilistic topic models:

- Want to find *themes* (or *topics*) in documents. Useful for searching or browsing.
- We are not interested in supervised topic classification, and do not want to fix topics in advance.
- Want an approach that carries out topic discovery.
- Essentially we want to perform probabilistic/fuzzy clustering of words. As a side-effect, we may be able to cluster documents.

Latent Dirichlet Allocation (LDA):

- Documents exhibit multiple topics (but typically not many)
- LDA is a probabilistic model with a corresponding generative process (each document is generated by this process)
- A topic is a distribution over a fixed vocabulary (topics are assumed to be generated first, before the documents)
- Only the number of topics is specified in advance Sort of like K-means for text data.



Seeking Life's Bare (Genetic) Necessities

COLD SPRING HARBOR, NEW YORK— How many genes does an organism need to survive? Last week at the genome meeting here,* two genome researchers with radically different approaches presented complementary views of the basic genes needed for life. One research team, using computer analyses to compare known genomes, concluded that today's organisms can be sustained with just 250 genes, and that the earliest life forms required a mere 128 genes. The other researcher mapped genes in a simple parasite and estimated that for this organism, 800 genes are plenty to do the job—but that anything short of 100 wouldn't be enough.

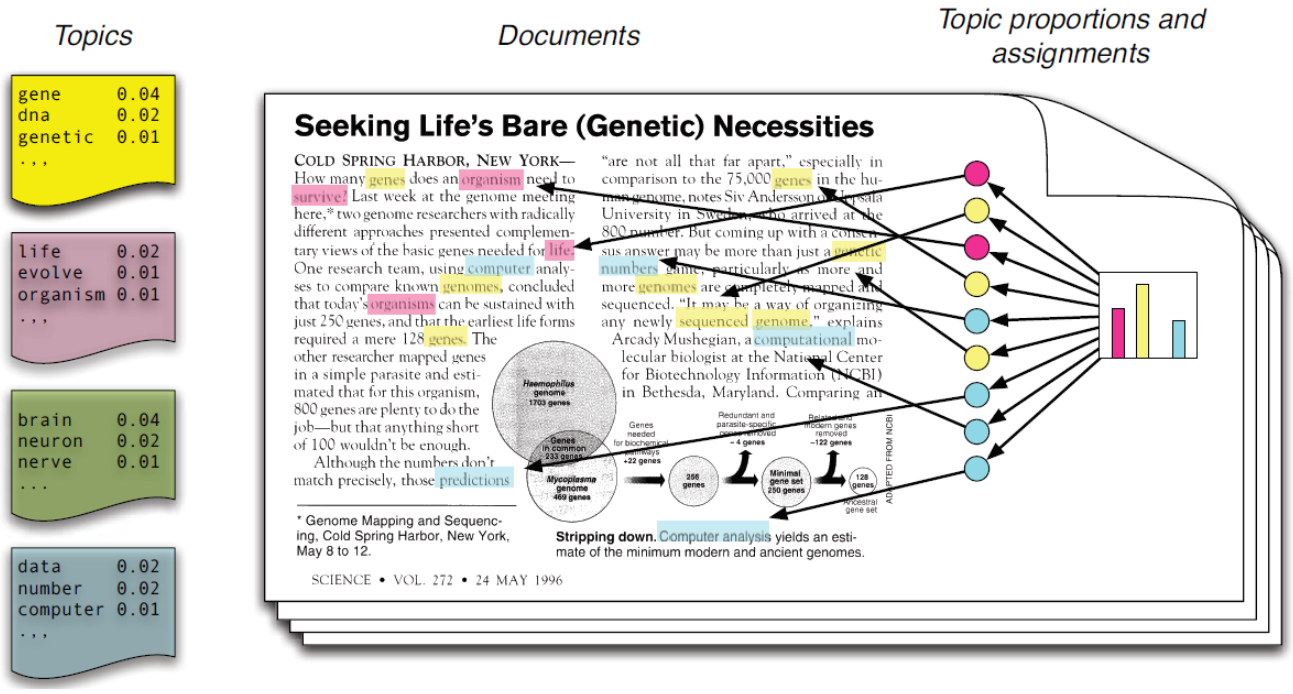
Although the numbers don't match precisely, those predictions "are not all that far apart," especially in comparison to the 75,000 genes in the human genome, notes Siv Andersson of Uppsala University in Sweden, who arrived at the 800 number. But coming up with a consensus answer may be more than just a genetic numbers game, particularly as more and more genomes are completely mapped and sequenced. "It may be a way of organizing any newly sequenced genome," explains Arcady Mushegian, a computational molecular biologist at the National Center for Biotechnology Information (NCBI) in Bethesda, Maryland. Comparing an

* Genome Mapping and Sequencing, Cold Spring Harbor, New York, May 8 to 12.

Stripping down. Computer analysis yields an estimate of the minimum modern and ancient genomes.

SCIENCE • VOL. 272 • 24 MAY 1996

Intuition for LDA:

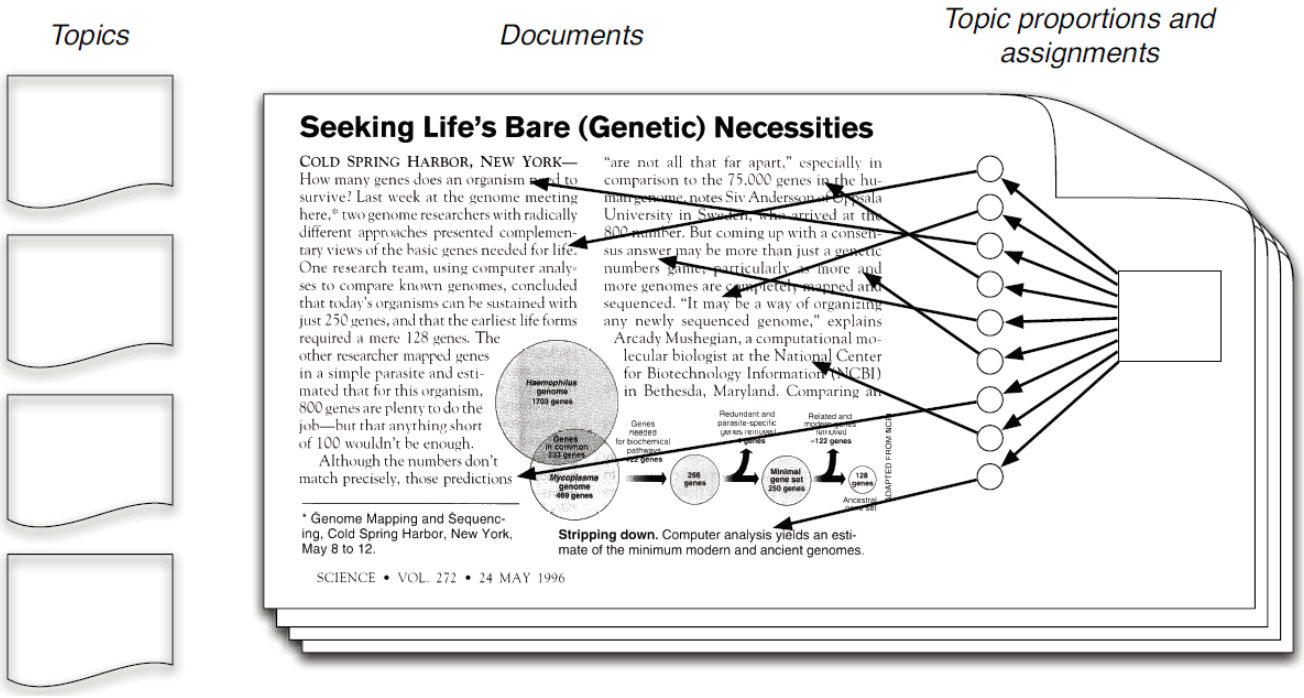


Intuition for LDA:

- Each topic is a distribution over words, and topics can share words (usually with different probabilities of occurrence)
- Each document is a mixture of corpus-wide topics
- Each word is drawn from one of those topics

Ultimately we want to infer the hidden variables.

Inference for LDA:



Example application: Harry Potter

Obtained text of all seven Harry Potter novels from
`https://github.com/bradleyboehmke/harrypotter`.

Treat chapters across the seven books as documents.

Want to discover clusters of words that "hang together" as topics, and understand how different topics are present across different chapters.

Sample text by chapter:

```
 1 Goblet Fire_17     "   Harry sat there, aware that every head in the Great ~
 2 Deathly Hallows_4  "Harry ran back upstairs to his bedroom, arriving at the~
 3 Deathly Hallows_2  "Harry was bleeding. Clutching his right hand in his lef~
 4 Prisoner Azkaban_8 "  FLIGHT OF THE FAT FADY  In no time at all, Defense Ag~
 5 Order Phoenix_38   "The Second War BeginsHE WHO MUST NOT BE NAMED RETURNS'I~
 6 Deathly Hallows_8  "Three o'clock on the following afternoon found Harry, R~
 7 Goblet Fire_35     "   Harry felt himself slam flat into the ground; his fa~
 8 Order Phoenix_12   "Professor UmbridgeSeamus dressed at top speed next morn~
 9 Prisoner Azkaban_4 "  THE LEAKY CAULDRON  It took Harry several days to get~
10 Deathly Hallows_17 "Harry, stop.\"\"What's wrong?\"They had only just reach~
```

LDA - the generative process:

Fix the number of topics in advance (like K-means clustering).

To generate a document:

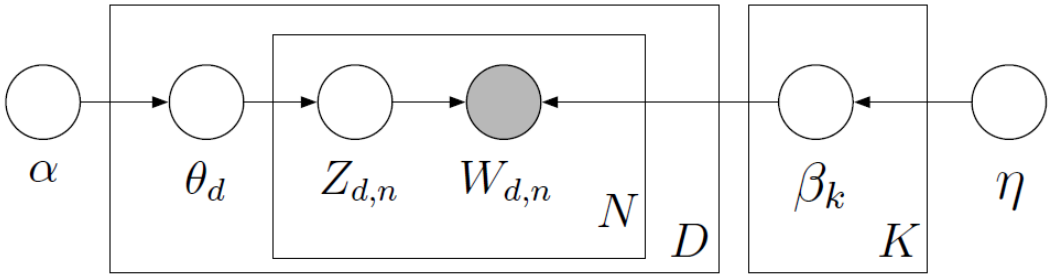
1. For each topic, generate/simulate probabilities of word occurrences.
2. Generate/simulate probabilities of topics within the document.
   (a) Draw a topic at random from these probabilities.
   (b) Draw a word at random from the distribution of word occurrences for that topic.

LDA - more technical description of generative process:

Assume V words in the vocabulary, D documents, and K topics

1. Generate vector $\boldsymbol{\beta}_k$ for each topic $k = 1, \ldots, K$ of length V from a Dirichlet$(\eta, \eta, \ldots, \eta)$ distribution. These are the probabilities over words in the dictionary for topic k.
2. For document $d = 1, \ldots, D$
   (a) Generate $\boldsymbol{\theta}_d$ of length K (topic probabilities) from a Dirichlet$(\alpha_1, \ldots, \alpha_K)$ distribution.
   (b) Generate $z_{id} \in \{1, \ldots, K\}$ according to probabilities $\boldsymbol{\theta}_d$ (topic of word i in document d).
   (c) Generate word $w_{id} \in \{1, \ldots, V\}$ according to probabilities in $\boldsymbol{\beta}_{z_{id}}$.

LDA generative model as a diagram:



Dirichlet distribution:

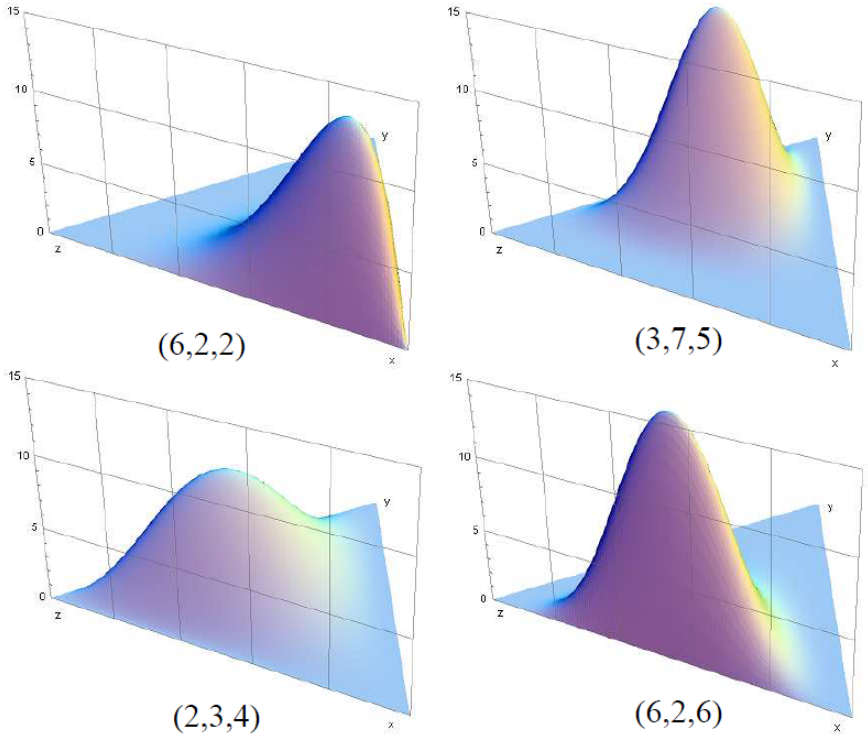
If $p_1, p_2, \ldots, p_M$ are M probabilities that sum to 1, then the Dirichlet distribution is a common choice of a prior distribution describing beliefs about the locations of the p_m.

Dirichlet distribution:

$$\mathsf{p}(p_1, \ldots, p_M | \alpha_1, \ldots, \alpha_M) = \frac{\Gamma(\sum_{m=1}^{M} \alpha_m)}{\prod_{m=1}^{M} \Gamma(\alpha_m)} \prod_{m=1}^{M} p_m^{\alpha_m - 1}$$

for $p_m \geq 0$ all m, and $\sum_{m=1}^{M} p_m = 1$.

Dirichlet Distribution Example



Parameters: $(\alpha_1, \alpha_2, \alpha_3)$

Statistical inference:

The posterior distribution can be summarized via MCMC simulation (see Blei et al., 2003 for details).

Most interested in finding the summaries for

$$\begin{aligned} \boldsymbol{\beta}_k &= (\beta_{k1}, \ldots, \beta_{kV}) && \text{topic probabilities of individual terms} \\ \boldsymbol{\theta}_d &= (\theta_{d1}, \ldots, \theta_{dK}) && \text{topic proportions in a document} \\ z_{id} & && \text{topic assignment distribution of a word } i \text{ in document } d \end{aligned}$$

Straightforward to generate samples from the posterior distribution for these quantities, and then calculate sample means as Monte Carlo estimates of the true posterior means.

But what about choosing the number of topics, K?

Blei and Lafferty (2009) mention cross-validation strategies, and choosing the number based on information outside the data. Other approaches involve treating K as an unknown parameter and inferring it during model fitting (e.g., Teh et al., JASA, 2007).

Chen et al. (2015) recommend choosing K based on the harmonic mean of the estimated log-likelihood of each model.

Our approach: UMass confirmation method

Compute

$$C_{UMass} = \frac{2}{N(N-1)} \sum_{i=2}^{N} \sum_{j=1}^{i-1} \log \left(\frac{P(w_i, w_j) + \varepsilon}{P(w_j)} \right)$$

where N is the selected number of top words in topics, ε is a small constant to avoid a log of zero.

The term $P(w_j)$ is the frequency of the occurrence of word w_j in Wikipedia, and $P(w_i, w_j)$ is the frequency of the co-occurrence of words w_i and w_j in Wikipedia articles.

Larger coherence values suggest greater co-occurrence of words within documents using Wikipedia as a gold standard.

See Röder, M., Both, A., Hinneburg, A. (2015, February). Exploring the space of topic coherence measures. In Proceedings of the eighth ACM international conference on Web search and data mining (pp. 399-408). ACM.

Performing LDA in python: gensim library

See `https://radimrehurek.com/gensim/index.html`.

The process requires three steps

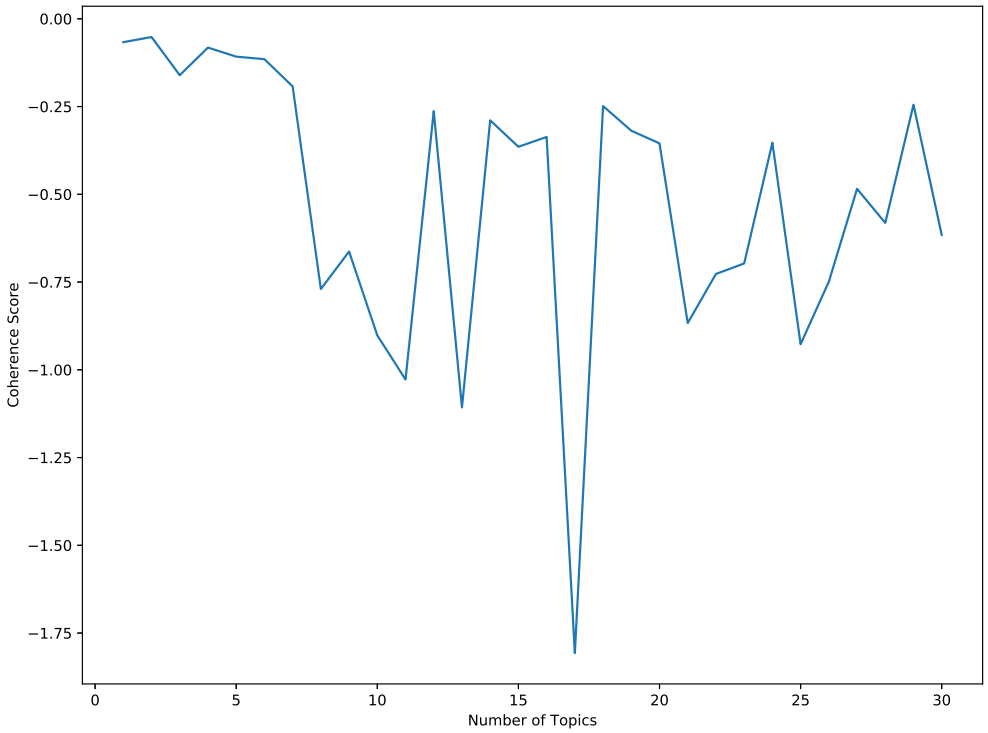
- Preparing the corpus, tokenizing the text, determining the count of each word within a document.
- Deciding (or determining) the number of topics, K.
- Fitting the model on the text data.

Various summaries can then be obtained from the result of the fit.

Working with text data:

There are endless ways to manipulate text data (in python).

Can use the `NLTK` toolkit. See the example in the ipynb lecture notes code.



LDA model fitting for Harry Potter novels:

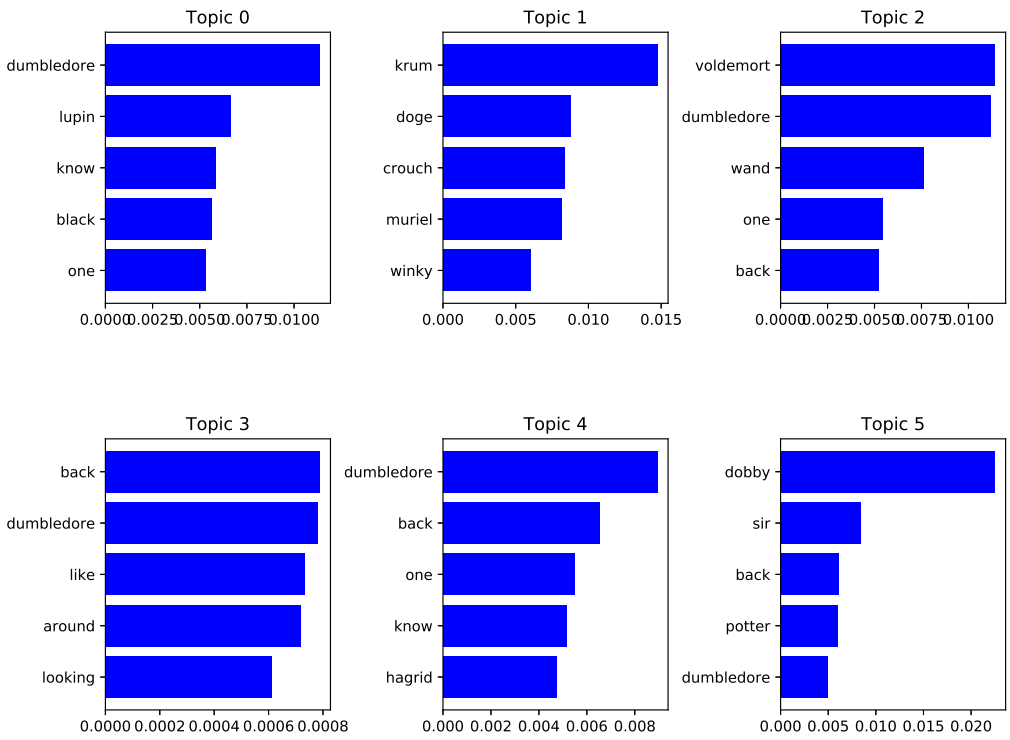
Optimization analysis suggests $K = 3$ topics, but we will use 16.

A few topics: :

```
0: dumbledore lupin know black one looked like snape sirius back around see never face righ
1: krum doge crouch muriel winky auntie fleur hic luna know skeeter marquee oh hagrid dumbl
2: voldemort dumbledore wand one back riddle death around know like though looked eyes stil
3: back dumbledore like around looking professor one see eyes know voice though snape look
```

Corresponding βs: :

```
0: 0.011360412 0.006647794 0.0058578085 0.0056657353 0.00531025 0.005158881 0.004821639 0.0
1: 0.014736434 0.0087334905 0.008363549 0.008150502 0.005997632 0.0046619778 0.004424218 0
2: 0.0114096925 0.011179432 0.007624367 0.0054342975 0.0052318317 0.004827917 0.0045865234
3: 0.0007882023 0.0007830554 0.0007326782 0.0007203899 0.0006111204 0.00053769234 0.000516
```



Highest θ by document (chapter):

First column is document number (chapter), second is topic, third is $\hat{\theta}$ normalized.

```
133   9     0.999759
69    9     0.999731
194   4     0.999712
118   12    0.999699
100   6     0.999689
147   0     0.999688
29    8     0.999682
179   0     0.999637
46    10    0.999635
138   14    0.999630
```

Other tasks:

- Visualizing documents - color words with topic having the highest estimated probability
- Identifying similar documents - can compute Hellinger distance between documents as a

dissimilarity measure:

$$D_{d,f} = \sum_{k=1}^{K} \left(\sqrt{\hat{\theta}_{d,k}} - \sqrt{\hat{\theta}_{f,k}} \right).$$

LDA extensions:

- Dynamic topic models (LDA where topic probabilities evolve over time)
- Correlated topic models (LDA topic structures may be related across topics)
- Supervised LDA (have a response variable for each document and "train" the LDA to fit to the response)
- Plenty of new work in this area!

Some references:

- D. Blei, A. Ng, and M. Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993-1022, January 2003.
- D. Blei and J. Lafferty. Topic Models. In A. Srivastava and M. Sahami, editors, *Text Mining: Theory and Applications.* Taylor and Francis, 2009.
- B. Chen, X. Chen, and W. Xing. "Twitter Archeology" of Learning Analytics and Knowledge Conferences. In Proceedings of the Fifth International Conference on Learning Analytics And Knowledge (pp. 340-349). New York, NY, 2015.

Final thoughts about Bayesian statistics:

- Precise modeling leads to statistical inferences through a leak-proof route
- Given a probability model, Bayesian analysis makes full use of all the data
- Statistical inferences that are unacceptable must come from inappropriate modeling assumptions, not a problem in the underlying inferential mechanism

- Less attention need to be given to mathematical convenience of models, and more attention on scientific merit
- Awkward problems that Frequentists face (e.g., choice of estimators, adjustments for certain types of data) do not arise for Bayesians
- Modern computational methods (MCMC sampling from the posterior distribution) enables fitting even complex data models.