PyMC3 model specification; regression models

ISTA 410 / INFO 510 - Bayesian Modeling and Inference

University of Arizona School of Information

September 8, 2021

Outline

Last time:

- Normal model, unknown variance
- Random sampling; applications in prior and posterior predictive checking

Today:

- A bit more model checking
- Introducing PyMC3
- Regression models as Bayesian models

Note: refer to sec. 4.2, 4.4 of *Rethinking*

Predictive sampling

Simulating future observations

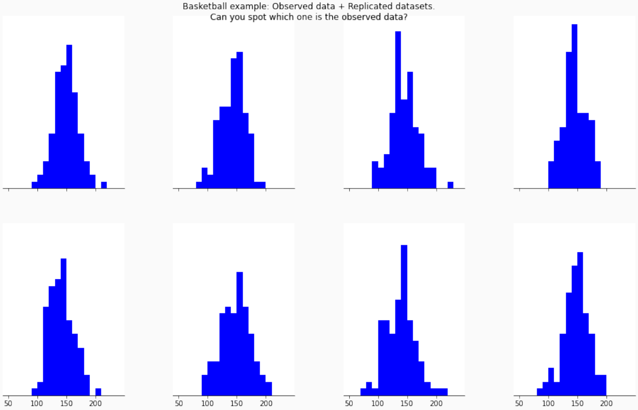
- Sampling from the posterior – produce plausible values of (μ, σ)
- Since the models are generative, we can also produce predictions of y
- Process:
  - Draw a pair $(\hat{\mu}, \hat{\sigma})$ from the posterior
  - Draw a value $y \sim N(\mu, \sigma)$

These samples come from the *posterior predictive distribution*

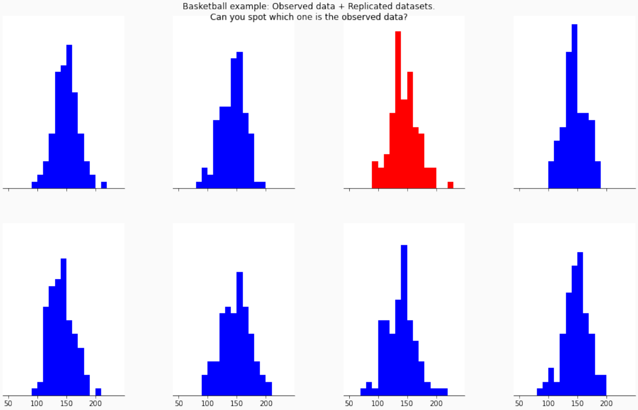
Why sample from the predictive distribution?

- Forecasting: application of models, especially in ML contexts, often involves
- Model evaluation: a good model should be able to produce simulated data that "resembles" real data (recall what we did with the kidney cancer example)
- Software testing: use predictive sampling with known models / fake data to ensure consistency
- Model design and prior evaluation: help understand structure of model and implications of the prior; power analysis

Checking the model by posterior predictive sampling



Checking the model by posterior predictive sampling



Predictive checking

Basic idea:

- Bayesian models are generative: they give a framework for generating data given parameter values
- So, we can generate data and check it for reasonableness
- Goals of predictive checking:
  - Prior: confirm that the prior model makes possible predictions
  - Posterior: confirm that the posterior (fitted) model makes predictions that resemble existing data

Example: speed of light measurements

Example

Simon Newcomb's speed-of-light experiment (1882):

- Place a mirror at the base of the Washington Monument
- Flash a light from the US Naval Observatory (where Newcomb worked) about 7442 m away
- Measure travel time

Simple model for estimating the travel time, assuming normal errors:

$$y_i \sim \mathrm{Normal}(\mu, \sigma)$$
$$p(\mu, \sigma) \propto (\sigma^2)^{-1}$$

Aside: why Gaussian?

Gaussian distributions are everywhere in statistics – why?

- Easy to calculate with
- Commonplace in nature
  - Adding together independent fluctuations leads to dampening
  - Damped fluctuations end up Gaussian
  - No information survives except mean/variance
- Very conservative assumptions
  - Given only mean and variance, Gaussian is the most conservative (*maximum entropy*) distributional assumption
  - Nature likes maximum entropy

Assessing the model

How can we assess the accuracy of this model?

- External validation: Compare model predictions to new observations
    - Model estimates the speed of light inaccurately based on current estimates
    - But, this is more because of the experimental design and limitations of data collection

Assessing the model

How can we assess the accuracy of this model?

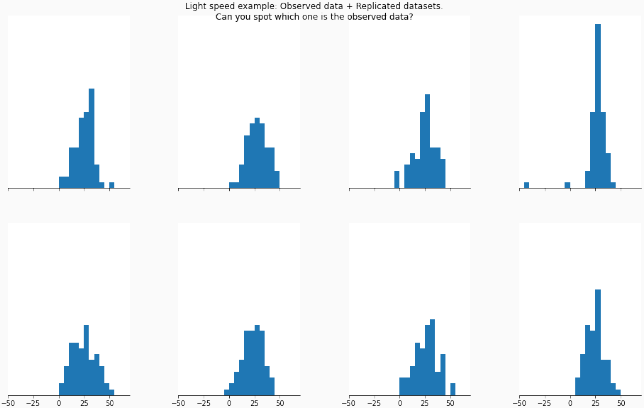
- External validation: Compare model predictions to new observations
    - Model estimates the speed of light inaccurately based on current estimates
    - But, this is more because of the experimental design and limitations of data collection
- Internal validation: Assess model accuracy / plausibility with the data we already have
    - Do the model predictions look right *relative to the data we have*?

Posterior predictive check for the speed-of-light

Simple posterior predictive check:

- Generate 66 observations from the posterior predictive distribution
- Repeat many times
- View histograms of these sets of 66 observations
- If we notice anything odd, drill down on that

Posterior predictive check for the speed-of-light model



Revising the model

What should we do?

- We see that the model reproduces some properties of the data, but the two outliers are not consistent with the model
- Problem: normal distribution has short tails, won't predict extreme outliers
- Model attempts to adjust by increasing σ, but this distorts the bulk

Updating the model:

- Replace the model likelihood with something that has heavier tails:

$$y_i \sim \mathrm{Normal}(\mu, \sigma)$$
$$y_i \sim \mathrm{StudentT}(\nu, \mu, \sigma)$$
$$y_i \sim \mathrm{Cauchy}(y_0, \gamma)$$

New candidate likelihoods

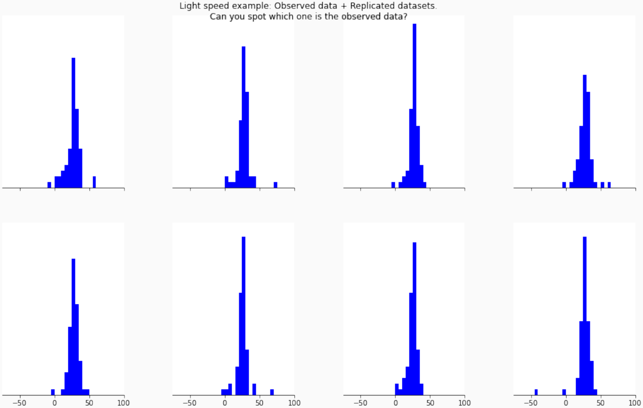
- Student's t-distribution:
  - Sampling distribution of a sample mean of $\nu + 1$ observations from a Gaussian distribution
  - Can be thought of as a mixture of Gaussians with different variances
  - Heavier tails than a Gaussian
- Cauchy distribution:
  - PDF:
$$p(y|y_0, \gamma) \propto \frac{1}{1 + \left(\frac{y - y_0}{\gamma}\right)^2}$$
  - Equivalent to StudentT with $\nu = 1$
  - Very heavy tails

Posterior predictive check with StudentT

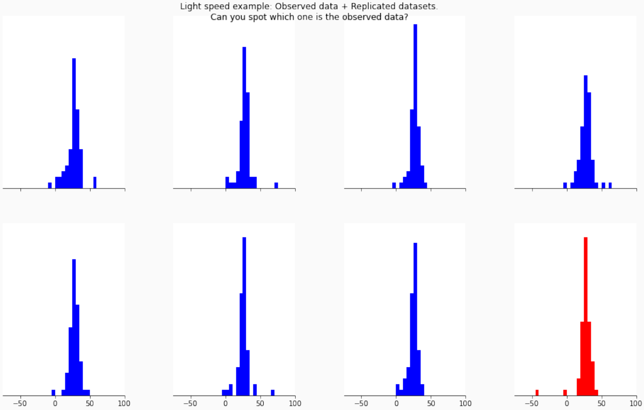
New model using a Student's *t* likelihood:

$$
\begin{aligned}
y_i &\sim \mathrm{StudentT}(\mu, \sigma, \nu) \\
\mu &\sim \mathrm{Normal}(0, 50) \\
\sigma &\sim \mathrm{Exponential}(1) \\
\nu &\sim \mathrm{Exponential}(1)
\end{aligned}
$$

Posterior predictive check with StudentT



Posterior predictive check with StudentT

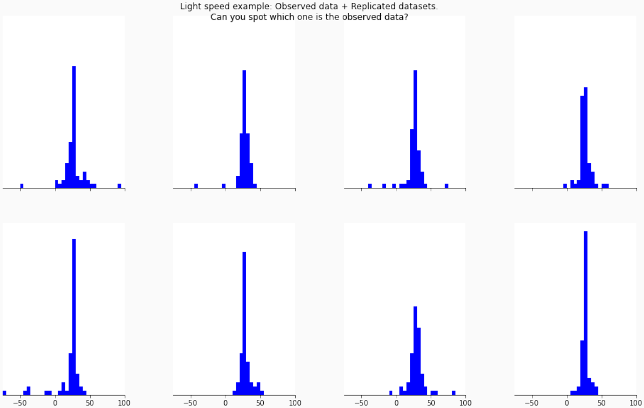


Posterior predictive check with Cauchy

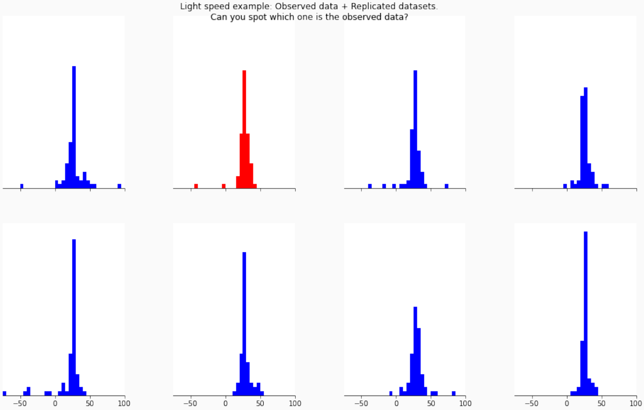
New model using a Cauchy likelihood:

$$\begin{aligned} y^{(i)} &\sim \mathrm{Cauchy}(y_0, \gamma) \\ y_0 &\sim \mathrm{Normal}(0, 50) \\ \gamma &\sim \mathrm{Exponential}(1) \end{aligned}$$

Posterior predictive check with Cauchy



Posterior predictive check with Cauchy



Summary

t and Cauchy likelihoods appear to do better, both at reproducing outliers and the main peak

- Normal peak is too wide, because it is trying to accommodate the outliers
- More flexible distributions can accommodate outliers without tuning the spread as much
- Choice of t or Cauchy is purely for distributional properties – like Gaussian, we're not choosing this for any physical reason
    - Know we need greater robustness vs. outliers, but we don't have a model for why

Regression models

Linear regression

We all know simple linear regression:

- Have a predictor variable x and a response variable y
- Pose a model equation of the form

$$\hat{y} = a + bx$$

- Seek a, b so that the mean squared error

$$\frac{1}{N} \sum_i (y_i - \hat{y}_i)^2$$

is minimized

How is this reproduced in our Bayesian modeling framework?

A normal model

At heart simple linear regression is a *normal model* just like our basketball model:

- Model mean and variance of normally (Gaussian) distributed observations
- Mean is an additive combination of weighted predictors
- Inference target is the predictor weights

Linear regression

This is easily reframed as a Bayesian model:

$$
\begin{aligned}
y_i &\sim \mathrm{Normal}(\mu_i, \sigma) \\
\mu_i &= a + bx_i \\
\sigma_i &\sim \text{(some prior)} \\
a &\sim \text{(some prior)} \\
b &\sim \text{(some prior)}
\end{aligned}
$$

Prior predictive simulations

Another way to make sense of this idea is to look at prior predictive simulations:

- Last time, we used posterior predictive simulations to assess model fit
- Prior predictive simulations can be used to assess reasonableness of priors

Example: CO2 vs. global temperature anomaly

Carbon dioxide and temperature

Our toy model for today: global average temperature as a function of atmospheric CO2, measured between 1959 and 2016

- $c =$ CO2 concentration in units of 100 ppm
- $T =$ global average temperature (Celsius) relative to 20th century average

Linear model:

$$T_i \sim \mathrm{Normal}(\mu_i, \sigma)$$
$$\mu_i = a + bc_i$$

Carbon dioxide and temperature

To specify the model, we also need to put some priors on a, b, σ.

Let's explore three options.

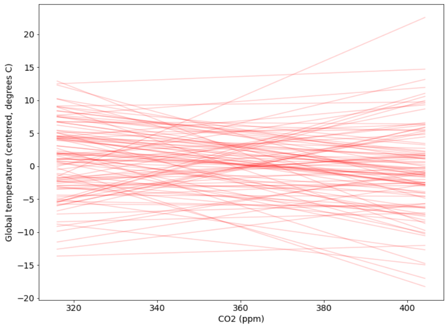
- Vaguest prior: everything is flat
- Vague prior: normal prior on a, b with a wide variance
- Weakly informative: normal on a, b with a narrower variance
- Slightly more informative: normal on a, log-normal on b

Why might we prefer some of these options over the others?

Prior predictive simulations

Vague prior:

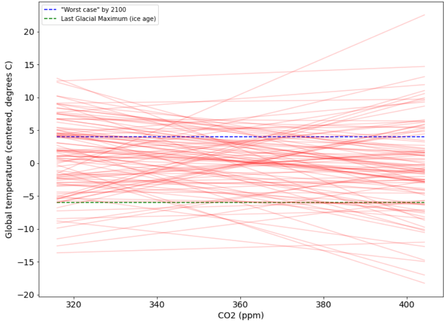
$$a \sim \mathrm{Normal}(0, 5)$$
$$b \sim \mathrm{Normal}(0, 10)$$



Prior predictive simulations

Vague prior:

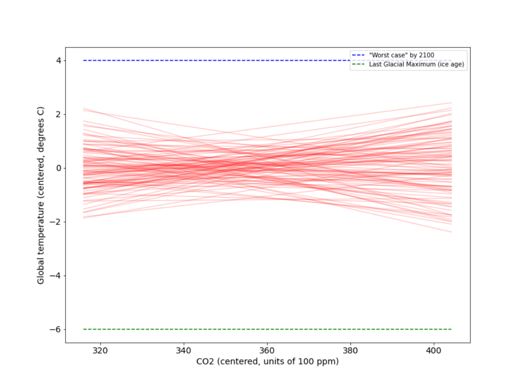
$$a \sim \mathrm{Normal}(0, 5)$$
$$b \sim \mathrm{Normal}(0, 10)$$



Prior predictive simulations

Weakly informative priors:

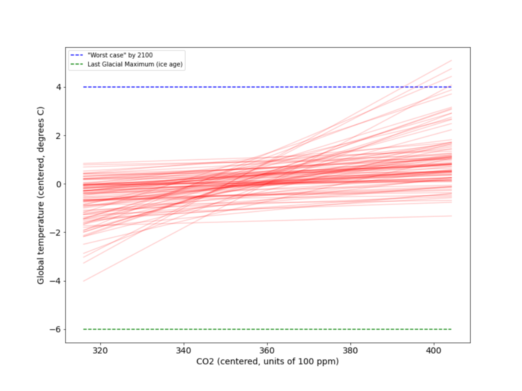
$$a \sim \mathrm{Normal}(0, 0.5)$$
$$b \sim \mathrm{Normal}(0, 1)$$



Prior predictive simulations

More informative prior:

$$a \sim \mathrm{Normal}(0, 0.5)$$
$$b \sim \mathrm{LogNormal}(0, 1)$$



Connection to OLS

Compare to ordinary least squares:

- Assume centered data $\rightarrow$ Joint likelihood:

$$p(y_i|b,\sigma) \propto \exp\left(-\frac{1}{2}\sum_{i=1}^{N}\frac{(y_i - bx_i)^2}{\sigma^2}\right)$$

- Conditional on σ, likelihood is maximized by minimizing the mean squared error

What does the prior on b do?

Let's suppose we put a normal prior on the slope parameter b:

$$
\begin{aligned}
y_i &\sim \mathrm{Normal}(\mu_i, \sigma) \\
\mu_i &= b x_i \\
\sigma_i &\sim \text{(some prior)} \\
b &\sim \mathrm{Normal}(0, \tau^2)
\end{aligned}
$$

What does the prior on b do?

Putting this prior in, we get a posterior distribution (again conditional on σ):

$$p(b|y,\sigma) \propto \exp\left(-\frac{1}{2}\left(\sum_{i=1}^{N}\frac{(y_i - bx_i)^2}{\sigma^2} + \frac{1}{\tau^2}b^2\right)\right)$$

Suppose we seek the mode (maximum) of the posterior distribution.

- What quantity would we minimize?
- Does this look familiar?

Ridge regression

Ridge regression: a "penalized" version of OLS that minimizes the loss function

$$\mathcal{L}(b) = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 + \alpha b^2$$

- a form of regularization – reducing model variability to combat overfitting
- although we won't always use the MAP estimate in Bayesian inference, a weakly informative prior on the coefficients still performs this function of "shrinking" estimates

Priors summary

Priors on our regression coefficients:

- Keep prior predictions within the realm of possibility
- Perform regularization: encourage our model to be skeptical of extremely strong effects
- Our most common choice: $\mathrm{Normal}(0, \sigma_0)$ – useful default, can control strength of regularization with σ_0

Quadratic approximation

Quadratic approximation redux:

- Approximate posterior as a Gaussian
- Estimate two things:
  - Peak of posterior (maximum *a posteriori* aka MAP)
  - Standard deviations and correlations of parameters (covariance matrix)
- with flat priors, same as conventional maximum likelihood estimation

Using `quap`

In `Rethinking`:

- `quap`: R function that takes a list of model specification distributions/equations, returns a model
- Lets you take model specs as we write them and translate directly into code, ignore details of computation

On D2L and INFO510-public GitHub:

- `modelutils.py`: contains a version of `quap`
- Works inside the PyMC3 model context

The PyMC3 model context

Building a model in PyMC3:

- Like the `quap` in Rethinking:
    - Model spec in code resembles model spec in math notation
    - Backend sets up and performs computations
- Modeling separated from inference – you can approximate the posterior in a number of ways

The PyMC3 model context

Context: everything happens inside a `with` block

```
import pymc3 as pm

with pm.Model() as model:
    # define some parameters
    # define observed variables
    # do inference (quap, MCMC, etc.)

# later...
with model:
    # more stuff
```

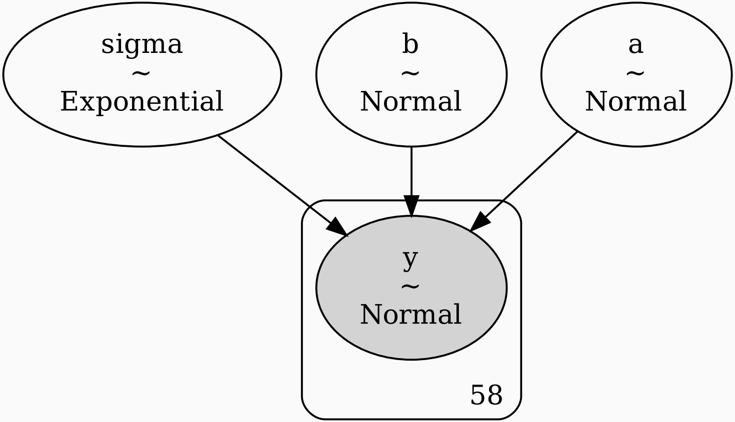
Specifying the model

$$T_i \sim \mathrm{Normal}(\mu_i, \sigma)$$
$$\mu_i = a + bc_i$$
$$a \sim \mathrm{Normal}(0, 1)$$
$$b \sim \mathrm{Normal}(0, 1)$$
$$\sigma \sim \mathrm{Exponential}(1)$$

```
with pm.Model() as model:
    a = pm.Normal('a', 0, 1)
    b = pm.Normal('b', 0, 1)
    sigma = pm.Exponential('sigma', 1)
    mu = a + b * c
    T = pm.Normal('t', mu, sigma,
                  observed=T_obs)
```

`c`, `T_obs` hold our actual data

Plate notation



Getting results

Inside the model context:

```
from modelutils import quap
with model:
    qp = quap()
```

Return value qp is an object containing:

- a table summarizing the marginal distributions of parameters
- the MAP estimates and covariance matrix
- methods for getting the summary, plotting parameter estimates, posterior sampling

Getting results

Currently implemented:

- `qp.summary()` – return the table of marginal distributions
- `qp.get_mode()`, `qp.get_cov()` – return a dictionary of posterior modes or a covariance matrix repsectively
- `qp.plot_forest()` – forest plot of parameter values
- `qp.extract_samples()` – sample from the posterior

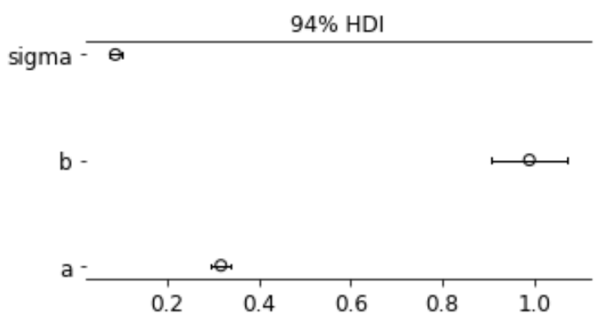
Getting results

```
qp.summary()
```

	mean	sd	hdi_3%	hdi_97%
a	0.317	0.011	0.296	0.339
b	0.990	0.044	0.907	1.073
sigma	0.087	0.008	0.072	0.102

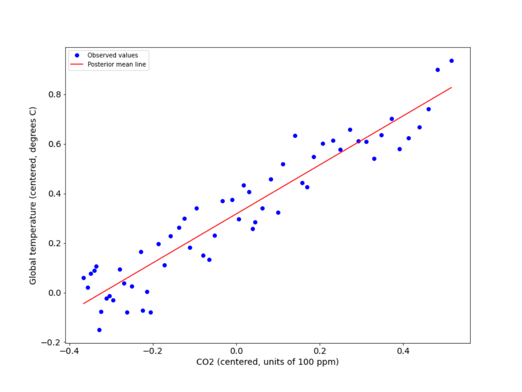
Getting results

```
qp.plot_forest()
```



Getting results

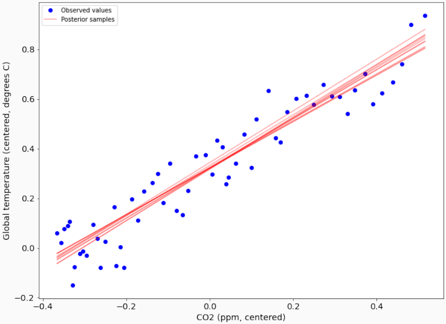
```
modes = qp.get_mode()
y = modes['a'] + modes['b'] * x
```



Putting uncertainty onto the graph

The posterior is full of lines:

```
samples = qp.extract_samples()
```



Summary

Today:

- Linear models
- Quadratic approximation in practice with PyMC3

Next week:

- Extending the linear model framework
- Multiple regression
- Intro to causal diagrams