Probabilistic Programming with CmdStanPy and plotnine

Mitzi Morris
Stan Development Team
Columbia University, New York NY

October 20, 2022

Probabilistic Programming

What is a probabilistic program?

- the program specifies a ***parametric model*** of the ***data generating process***.
- the inference engine uses the data to infer (i.e. learn) the parameters; it ***solves the inverse problem***.

Applications *already in widespread use* try to predict

- student ability from test results
- player / team abilities from pairwise matchups
- drug efficacy from clinical trial data
- population disease prevalence from tests of individuals
- election outcomes from voter surveys and census data

CmdStanPy: Python interface to Stan

What is Stan?

- Named after *Stanislaw Ulam* - inventor of Monte Carlo (MC) methods
  - estimation by repeated random sampling
- Imperative probabilistic programming language
  - Stan program defines a probability density
  - compiled to C++
- Stan algorithms
  - MCMC for full Bayesian inference (HMC-NUTS)
  - VB for approximate Bayesian inference
  - MLE for penalized maximum likelihood estimation

plotnine: a **grammar of graphics** for Python based on ggplot2

A plot has a common coordinate system and one or more layers.

Each layer is specified in terms of

- data - scalar, tuples, or tabular dataset
- a geometric plot object, e.g. x-y point and line plots, histograms, bar charts
- statistical transformations of the data
- *aesthetics* - a set of mappings from dataset columns to graph elements

Aesthetic mappings include size, shape, and color.

This increases the ways to show contrasts between dataset items.

Powerful paradigm, but relatively little documentation and examples in Python.

Rosetta Stone of Multilevel Modeling: Radon Levels in the Home

The data and models are taken from *Data Analysis Using Regression and Multilevel/Hierarchical Models*, by Andrew Gelman and Jennifer Hill

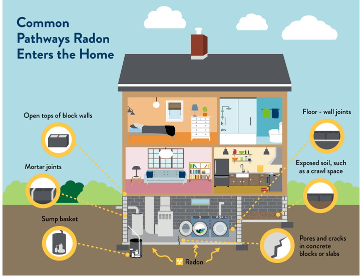
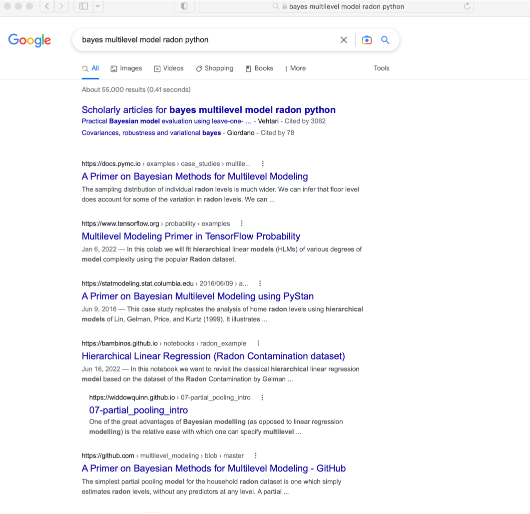


image source: Minnesota Department of Health



Today's Goal: Best Practices for Probabilistic Programming

How should (new) Stan user should approach a problem? Start with the data!

Preliminary data analysis is part of the design process

Estimates and predictions are the result of conditioning the model **on the data**.

- What do we want to estimate?
- What data is available?
- What are its size, shape, and tendencies?

EPA Radon Data for Minnesota

mn_radon - individual homes

	floor	county	fips	radon	log_radon	uranium	log_uranium	county_id
1	0	AITKIN	27001	2.2	0.79	0.50	-0.69	1
5	0	ANOKA	27003	2.5	0.92	0.43	-0.85	2
58	1	BECKER	27005	4.5	1.50	0.89	-0.11	3

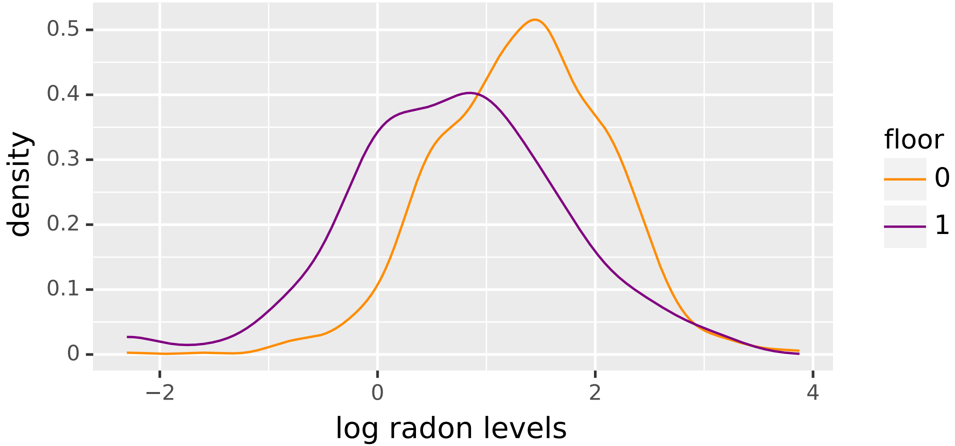
mn_uranium - county-level information

	county	fips	uranium	log_uranium	county_id	homes
0	AITKIN	27001	0.50	-0.69	1	4
1	ANOKA	27003	0.43	-0.85	2	52
2	BECKER	27005	0.89	-0.11	3	3

Best practice: put data on the log scale

Visualize radon levels

- Use plotnine.stat_density to compute the density of column `log_radon`.
- Map column `floor` to aesthetic ‘color’



plotnine code

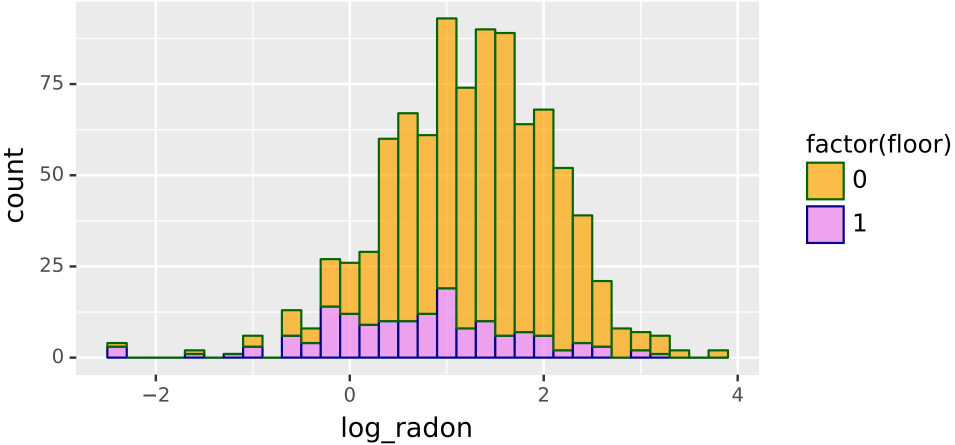
```
import matplotlib.pyplot as plt
import plotnine as p9
(p9.ggplot(data=mn_radon,
           mapping=p9.aes(x='log_radon', color='factor(floor)'))
    + p9.stat_density(geom='line')
    + p9.scale_color_manual(['darkorange','purple'], name='floor')
    + p9.xlab("log radon levels")
    + p9.theme(figure_size=(5,2.5))
)
```

all data and plots available at:

https://github.com/mitzimorris/pydata_paris_2022_10_20

Visualize measurements per floor

- Use plotnine.geom_histogram to show the amount of data per floor.
- Treat values in column `floor` as factors, map to aesthetic `color`



Percent home measurements by floor

The raw counts histogram reveals that most of the observations in the survey were taken on the basement level. Let's compute the exact percentages.

```
floor 0: 83%
floor 1: 17%
```

How many counties have data from both floors?

```
Number of counties: 85
Counties with measurements from floor 0: 85
Counties with measurements from floor 1: 60
```

Note: 2 counties with no data!

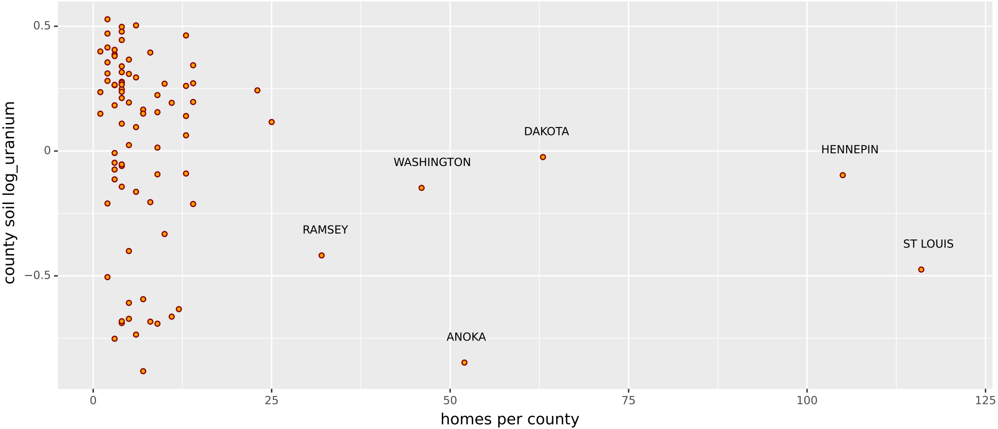
Radon and uranium levels

	radon	uranium	log_radon	log_uranium
mean	4.77	0.93	1.22	-0.13
std	4.48	0.32	0.85	0.37
min	0.10	0.41	-2.30	-0.88
25%	1.90	0.62	0.64	-0.47
50%	3.60	0.91	1.28	-0.10
75%	6.00	1.20	1.79	0.18
max	48.20	1.70	3.88	0.53

US EPA radon threshold: 4 pCi/L (picocuries per liter)

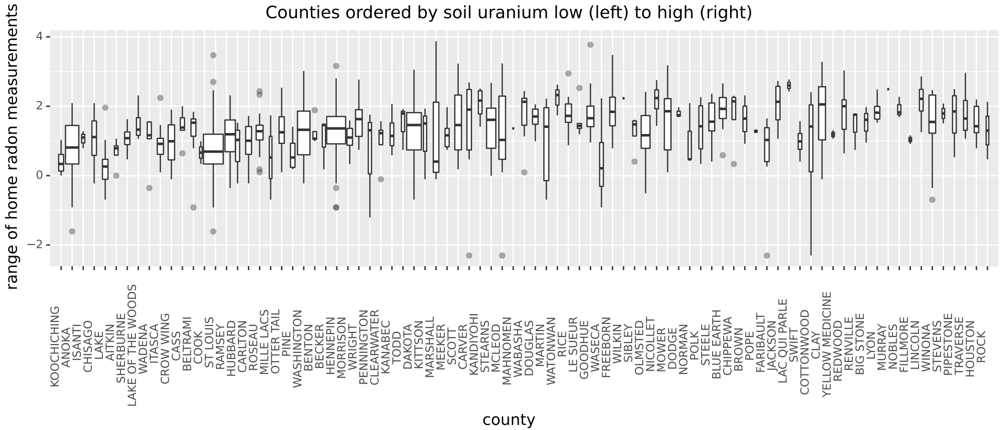
Soil uranium levels per county

Compare the background soil uranium (y-axis) to the number of measurements per county (x-axis). Add text labels to counties with more than 25 observations.



Radon levels per county

Show the range of measurements per county ordered by soil uranium, via plotnine.geom_boxplot



Preliminary Data Analysis

Takeaways:

- Within each county, the range of radon measurements is very wide.
- 70% of the counties (60 out of 85) have observations from both floors 0 and 1, the remaining 30% only have observations from floor 0 (basement).
- For most counties, there are fewer than 10 observations; 8 counties in metropolitan areas account for over half of the observations.

Bayesian Modeling

Bayesian estimation relates the **conditional probability** of the parameters (θ) given the data (y), $\mathrm{p}(\theta \mid y)$, to the **joint probability** of parameters and data, $\mathrm{p}(y, \theta)$

$$\mathrm{p}(\theta \mid \mathrm{y}) = \frac{\mathrm{p}(\mathrm{y}, \theta)}{\mathrm{p}(\mathrm{y})} \qquad \text{[def of conditional probability]}$$

$$= \frac{\mathrm{p}(\mathrm{y} \mid \theta)\,\mathrm{p}(\theta)}{\mathrm{p}(\mathrm{y})} \qquad \text{[rewrite joint probability as conditional]}$$

p(y) doesn't depend on θ - it's a constant for fixed y - we can drop it

$$\mathrm{p}(\theta \mid \mathrm{y}) \propto \mathrm{p}(\mathrm{y} \mid \theta)\,\mathrm{p}(\theta) \qquad \text{[unnormalized posterior density]}$$

The posterior is **proportional** to the **prior** times the **likelihood**

Model building and checking

Stan program encodes the model.

Fit the model to data to get estimates of model parameters and quantities of interest.

Systematically test the model predictions.

- Prior Predictive Checking - for model fit on no data.
- Posterior Predictive Checking - for model fit on all data.

Linear Regression

Find the linear function which best relates a scalar outcome (the dependent variable “y”) to one or more predictors (the independent variable “x”).

$y_i = \alpha + \beta x_i + \epsilon_i$

- α is the *intercept*, the offset from zero on the x-axis
- β is the *slope*, the multiplier applied to x.
- ϵ_i is the error term, assuming independent errors drawn from a normal distribution with location 0, scale σ.

Alternate formulation:

$y_i \sim \text{normal}(\alpha + \beta \, \mathrm{x}_i, \, \sigma), \quad \text{for } i = 1, \ldots, n$

Simple Linear Regression in Stan

```
data {
  int<lower=0> N; vector[N] x; vector[N] y;
}
parameters {
  real alpha; real beta; real<lower=0> sigma;
}
model {
  y ~ normal(alpha + beta * x, sigma);
  alpha ~ normal(0, 2.5);
  beta ~ normal(0, 2.5);
  sigma ~ normal(0, 5);
}
generated quantities {
  array[N] real y_rep = normal_rng(alpha + beta * x, sigma);
}
```

Stan prior choice wiki provides guidance on choice of priors and hyperpriors.

Simple Linear Regression for radon dataset

Outcome y is the log radon level.

Predictor x is the floor on which the measurement was taken.

Complete pooling - counties are interchangeable

$$\text{log\_radon}_i = \alpha + \beta\,\text{floor}_i + \epsilon_i$$

No-pooling - each county has its own intercept

$$\text{log\_radon}_{ij} = \alpha_j + \beta\,\text{floor}_i + \epsilon_i$$

Radon model - no pooling - per-county intercepts

```
data {
  int<lower=1> N;  // observations
  int<lower=1> J;  // counties
  vector[N] x;     // floor
  vector[N] y;     // radon
  array[N] int<lower=1, upper=J> county;
}
parameters {
  vector[J] alpha;   real beta;   real<lower=0> sigma;
}
model {
  y ~ normal(alpha[county] + beta * x, sigma);
  alpha ~ normal(0, 2.5);   beta ~ normal(0, 2.5);  sigma ~ normal(0, 5);
}
generated quantities {
  array[N] real y_rep = normal_rng(alpha[county] + beta * x, sigma);
}
```

Multilevel/hierarchical Regression

Multilevel regression models the dependency structures in the data along with the relation between outcome and predictors. This allows for ***partial pooling*** of information.

In this example: houses are grouped by county, counties are drawn from a common distribution whose scale determines the amount of information pooling.

Multi-level model for the intercept term α, for I homes across J counties:

$$y_i \sim \mathrm{normal}(\alpha_{j[i]} + \beta\, x_i,\, \sigma) \quad \text{for} \;\; i = 1, \ldots, \mathrm{I}$$
$$\alpha_j \sim \mathrm{normal}(\mu_\alpha,\, \sigma_\alpha), \quad \text{for} \;\; j = 1, \ldots, \mathrm{J}$$

Multilevel Radon Model

Model block:

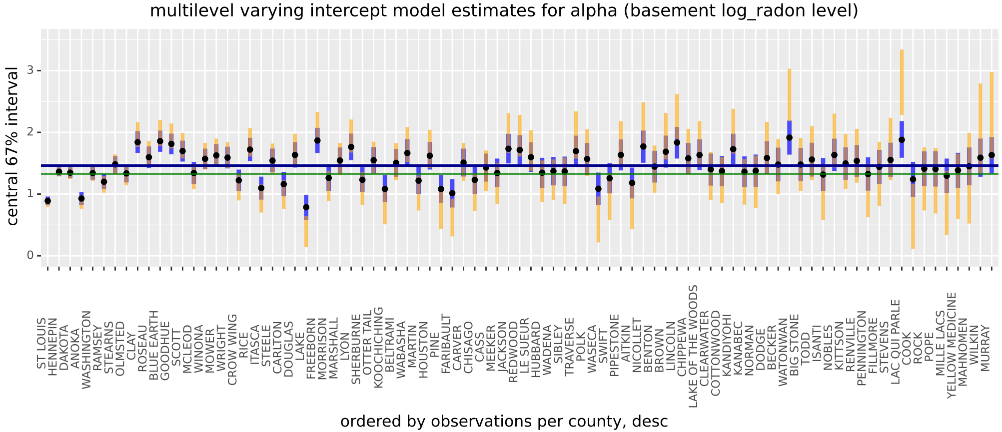
```
y ~ normal(alpha[county] + beta * x, sigma);
alpha ~ normal(mu_alpha, sigma_alpha); // partial-pooling
beta ~ normal(0, 2.5);
sigma ~ normal(0, 5);
mu_alpha ~ normal(0, 2.5);
sigma_alpha ~ normal(0, 5);
```

Compare to independent per-county intercepts

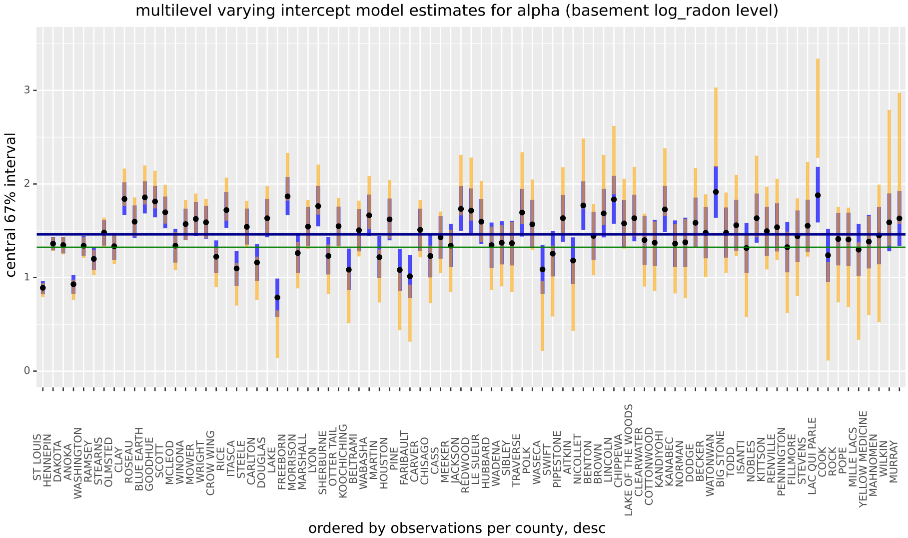
```
y ~ normal(alpha[county] + beta * x, sigma);
alpha ~ normal(0, 2.5);
beta ~ normal(0, 2.5);
sigma ~ normal(0, 5);
```

Model Comparison

- green - global intercept, complete-pooling model
- orange - per-county intercepts, no-pooling model
- blue - estimate from multilevel model; partial-pooling



Again

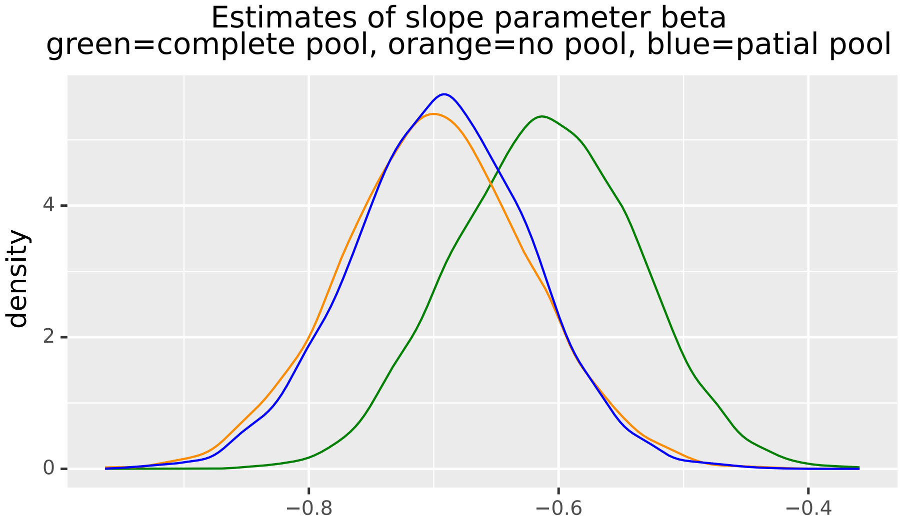


multilevel varying intercept model estimates for alpha (basement log_radon level)
central 67% interval
0
1
2
3
ST LOUIS
HENNEPIN
DAKOTA
ANOKA
WASHINGTON
RAMSEY
STEARNS
OLMSTED
CLAY
ROSEAU
BLUE EARTH
GOODHUE
SCOTT
MCLEOD
WINONA
MOWER
WRIGHT
CROW WING
RICE
ITASCA
STEELE
CARLTON
DOUGLAS
LAKE
FREEBORN
MORRISON
MARSHALL
LYON
SHERBURNE
OTTER TAIL
KOOCHICHING
BELTRAMI
WABASHA
MARTIN
HOUSTON
PINE
FARIBAULT
CARVER
CHISAGO
CASS
MEEKER
JACKSON
REDWOOD
LE SUEUR
HUBBARD
WADENA
SIBLEY
TRAVERSE
POLK
WASECA
SWIFT
PIPESTONE
AITKIN
NICOLLET
BENTON
BROWN
LINCOLN
CHIPPEWA
LAKE OF THE WOODS
CLEARWATER
COTTONWOOD
KANDIYOHI
KANABEC
NORMAN
DODGE
BECKER
WATONWAN
BIG STONE
TODD
ISANTI
NOBLES
KITTSON
RENVILLE
PENNINGTON
FILLMORE
STEVENS
LAC QUI PARLE
COOK
ROCK
POPE
MILLE LACS
YELLOW MEDICINE
MAHNOMEN
WILKIN
MURRAY
ordered by observations per county, desc

What about parameter beta?

	complete_pool_beta	no_pool_beta	partial_pool_beta
mean	-0.61	-0.70	-0.69
std	0.07	0.08	0.07
min	-0.85	-0.96	-0.94
25%	-0.66	-0.75	-0.74
50%	-0.61	-0.70	-0.69
75%	-0.56	-0.65	-0.65
max	-0.36	-0.42	-0.43

Visualize estimates of `beta`



Posterior Predictive Check

"Posterior predictive checks are the unit tests of probabilistic programming." – *Ben Goodrich*

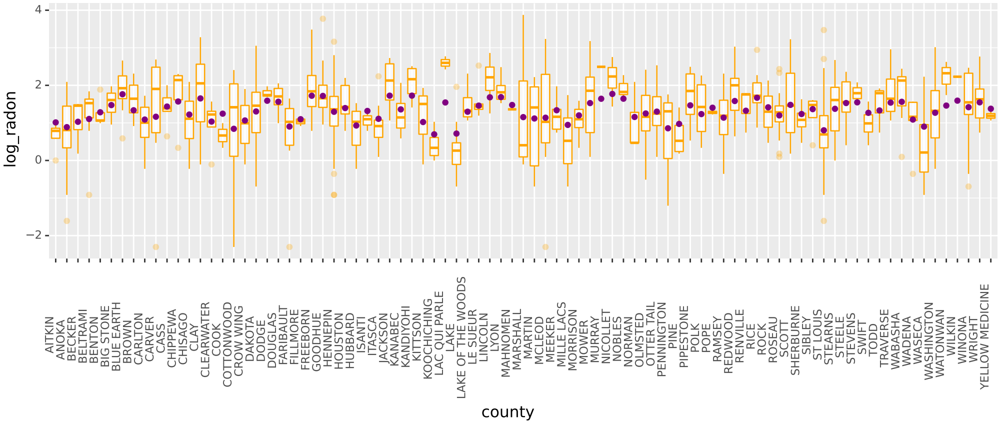
Simulate a new data set using the fitted model parameters:

```
generated quantities {
  array[N] real y_rep =
      normal_rng(alpha[county] + beta * x, sigma);
}
```

- Each iteration of the sampler generates a new dataset `y_rep` (replicate)
- Compute statistics on dataset `y_rep`; compare with those on `y`.

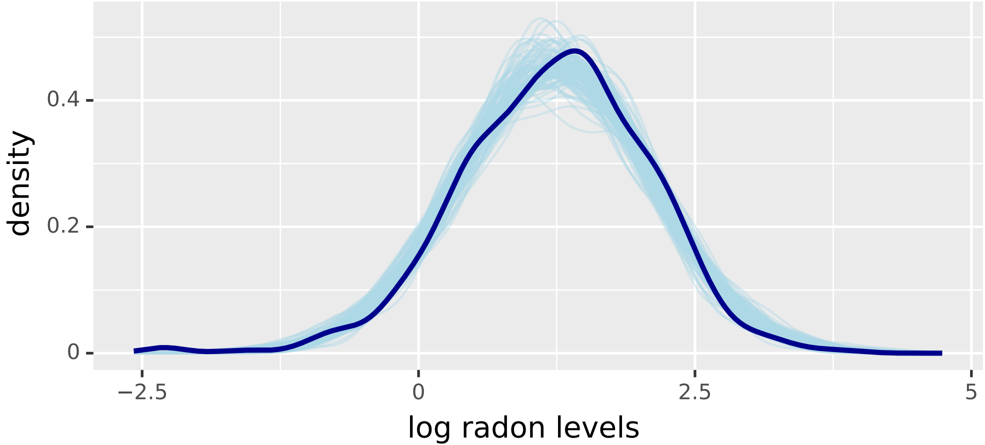
Posterior Predictive Test - Partial Pooling Model

Estimated mean home radon level per county overlaid on boxplot of y



Posterior Predictive Density Plot - Partial Pooling Model

Overlay the density plots of a random sample of `y_rep` with density plot of `y`



Concluding remarks

Data visualization drives design, testing, and documentation.

Prior and posterior predictive checks validate the model specification.

Plotnine provides a rich set of visualizations to demonstrate and communicate data, inferences, and predictions.

We welcome feedback and ideas for ways to turn these plots into easy-to-use tools.

References and Resources

Stan Tutorials YouTube Playlist Maggie Lieu - a series of introductory videos on Bayesian modeling with Stan

Stan User's Guide

Statistical rethinking by Richard McElreath - an intro-stats/linear models course taught from a Bayesian perspective.

- GitHub page
- Course page

Making Plots With plotnine - plotnine tutorial notebook.

CmdStanPy Documentation

Many Thanks!

Thanks! to members of the Stan Team

Thanks! to PyData Paris and OVH

Any Questions?