Tutorial 5

Statistical Computation and Analysis

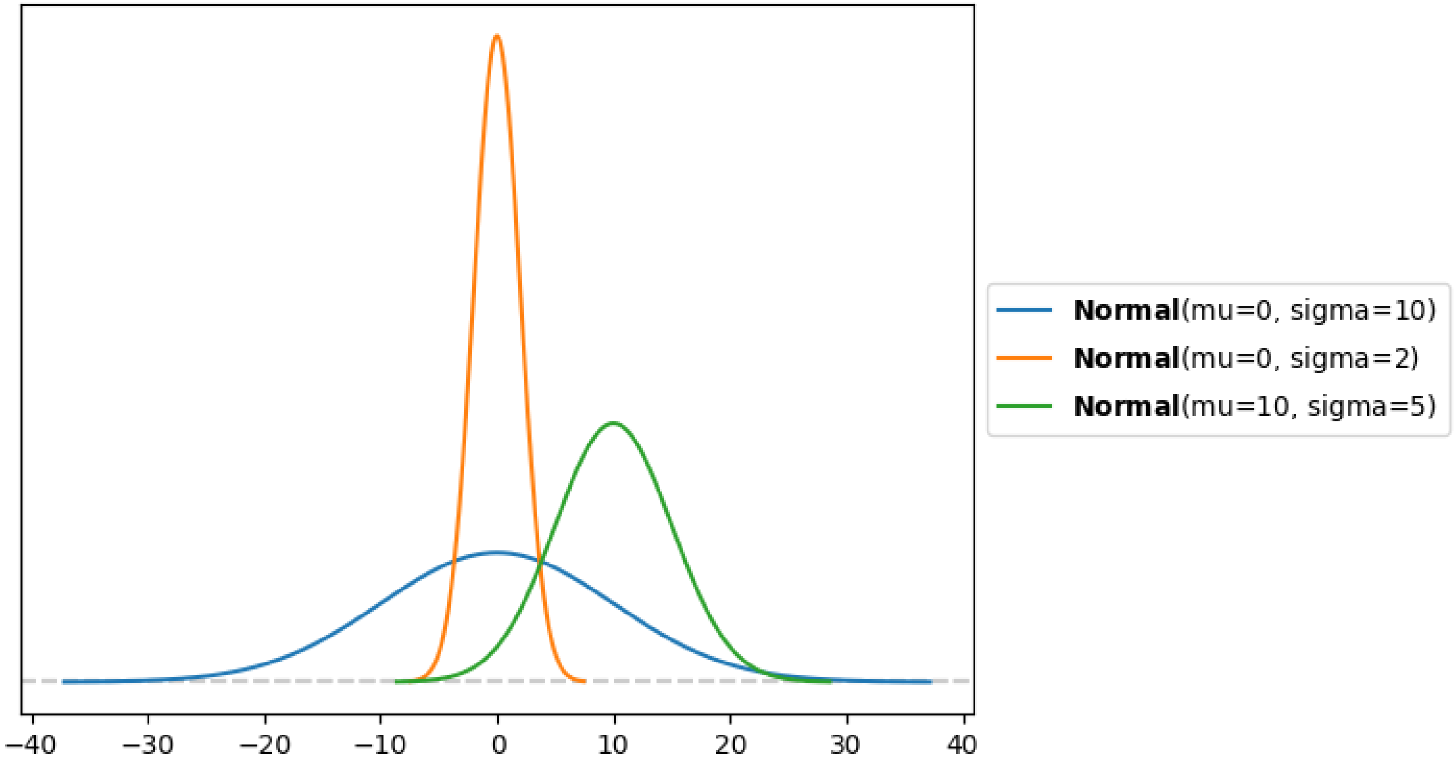
Spring 2025

Tutorial Outline

- Normal distribution
- Student’s t distribution
- Prior predictive checks
- Posterior predictive
- Groups comparison

Normal Distribution

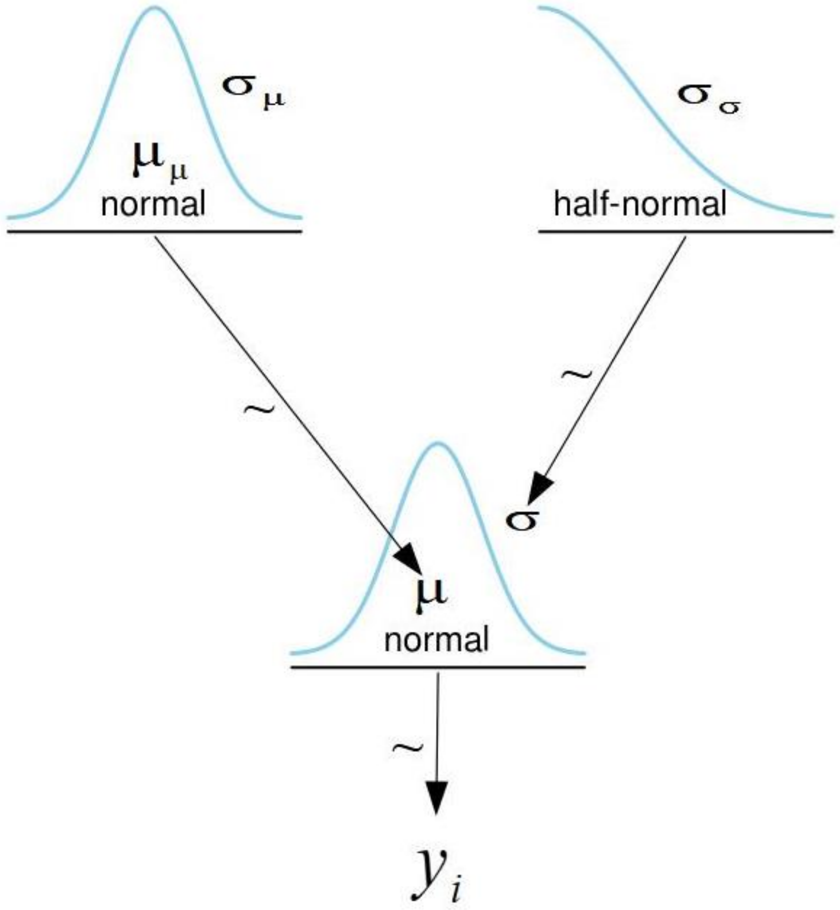
- Characterized by two parameters:
  - μ
  - σ



- If we model our data using a normal distribution, we will have a posterior distribution for each parameter.

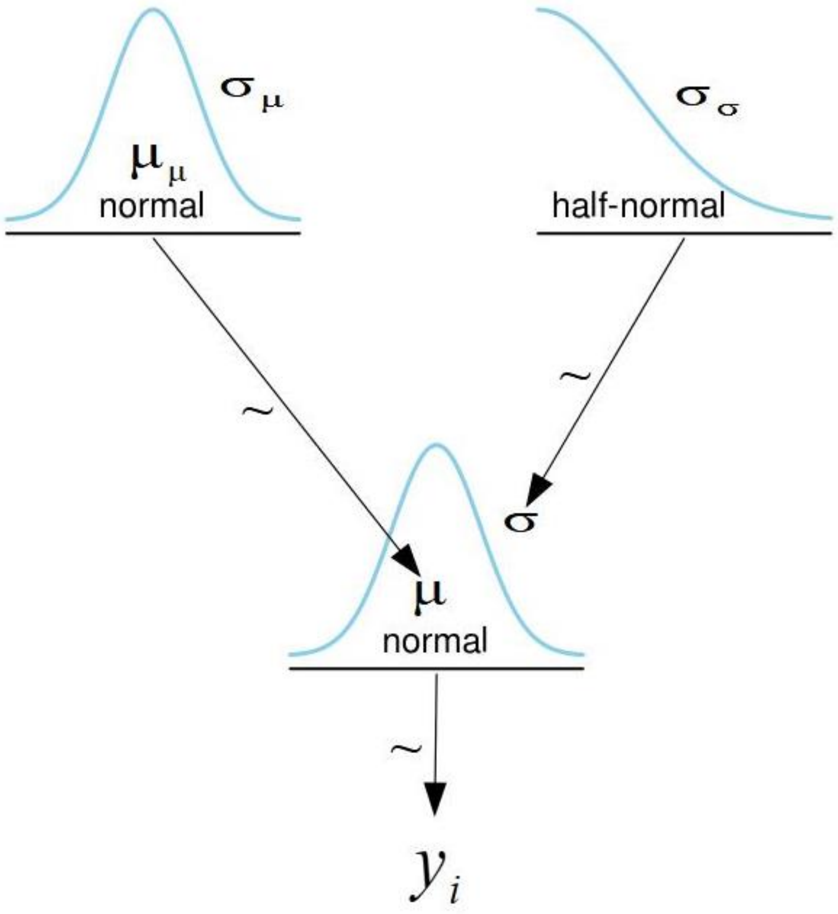
Normal Distribution

- Each parameter needs its own prior
  - Two priors
- We will use a normal prior for μ
  - Softer prior than uniform
- And a half normal prior for σ
  - Cannot be negative



Normal Distribution

Graphical model



Equations

$$y_i \sim N(\mu, \sigma)$$
$$\mu \sim N(\mu_\mu, \sigma_\mu)$$
$$\sigma \sim HalfNorm(\sigma_\sigma)$$

PyMC

```
coords = {"data": np.arange(len(data))}

with pm.Model(coords = coords) as model_g:
  m = pm.Normal('m', mu = mu_prior, sigma = sig_prior)
  sig = pm.HalfNormal('sig', sigma = sigma_prior)
  Y = pm.Normal('Y', mu=m, sigma=sig, observed=data, dims = 'data')
  idata_g = pm.sample(1000, chains = 4)
```

Data Collection

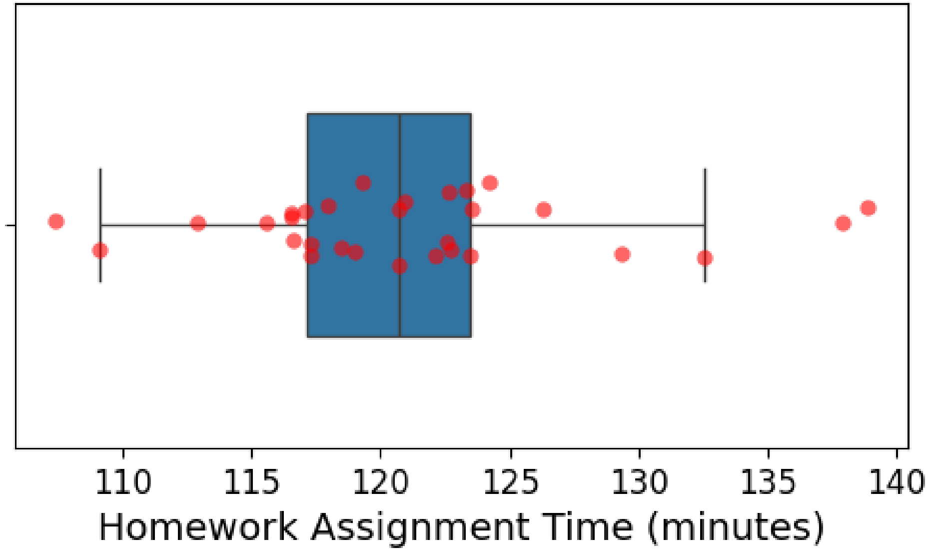
- How long does it take you to do a Statistics homework assignment in **minutes**?

https://docs.google.com/forms/d/e/1FAIpQLSeZ3_hrngoUDsQAM00WbY-097Q2D1t7ymABgFQDi3oavJzgnw/viewform?usp=dialog

- Let’s model this data using a normal distribution

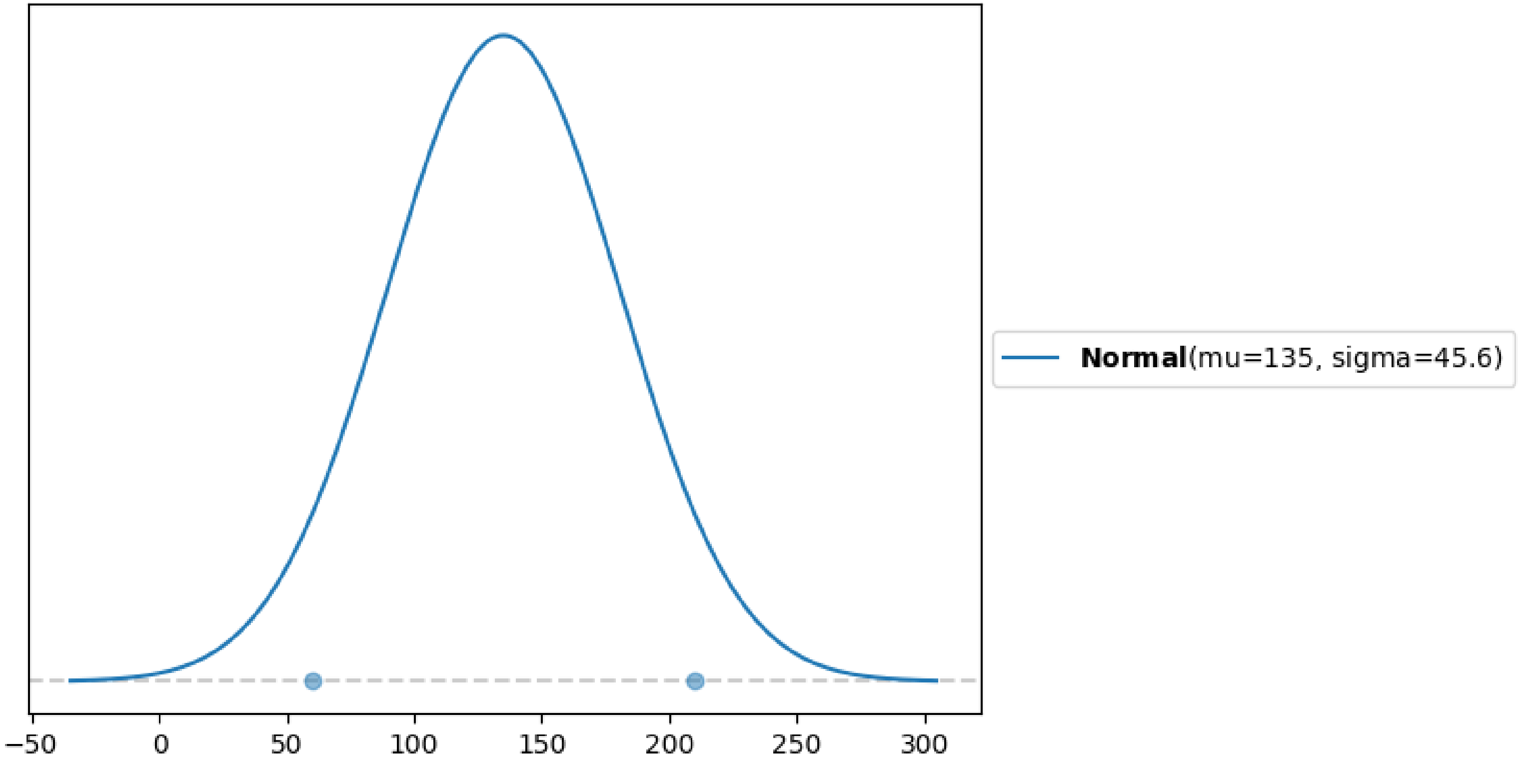
Plot Data (Simulated)

- Box plot
  - Line inside the box: median
  - The box: quartiles (25th and 75th percentiles)
  - Whiskers: Range excluding outliers
  - Dots: outliers (usually more than 1.5 IQR beyond IQR)



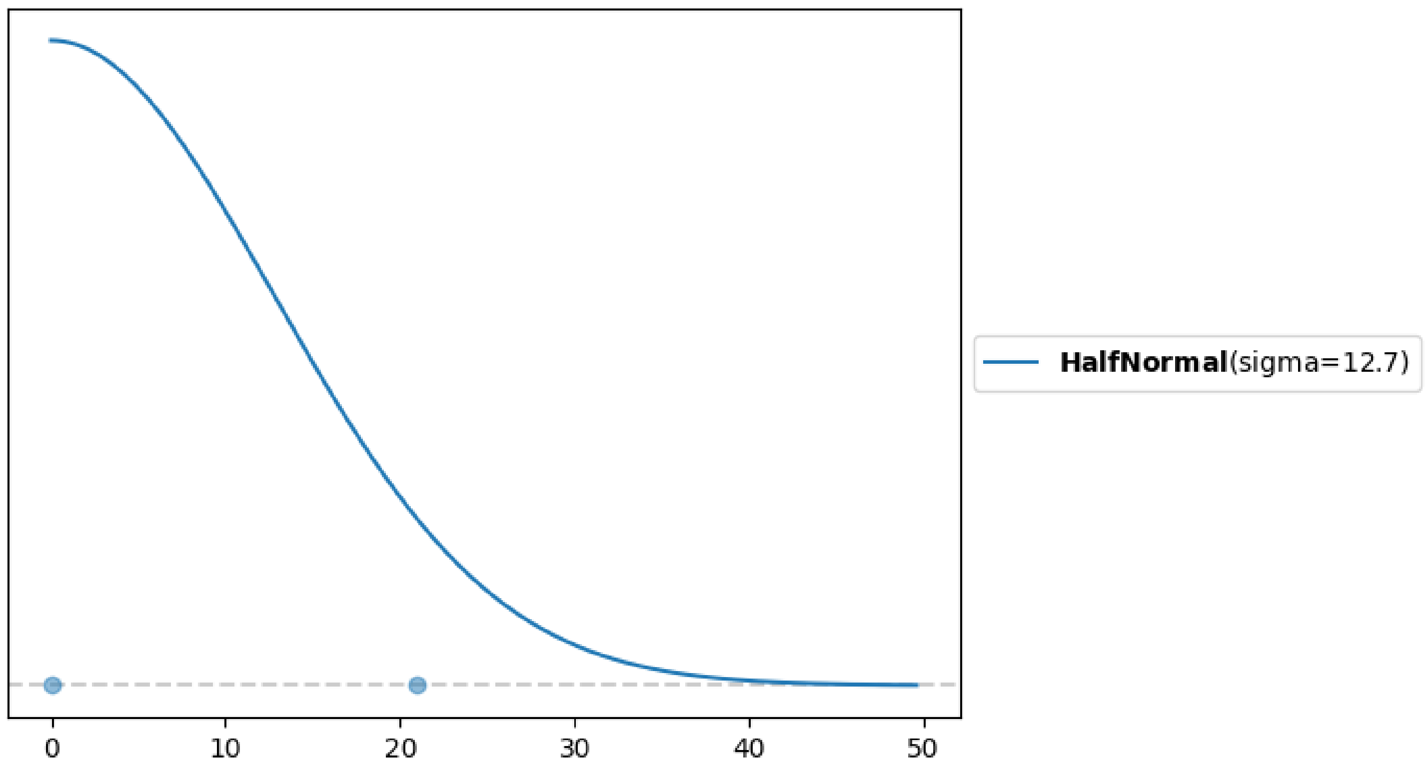
Priors

```
#mu
dist = pz.Normal()
pz.maxent(dist, 60, 210, 0.9)
```



Priors

```
#sigma
dist = pz.HalfNormal()
pz.maxent(dist, 0, 3*np.std(data, ddof = 1), 0.9)
```



Posterior

```
idata_g
```

arviz.InferenceData

▾ posterior

xarray.Dataset

▸ Dimensions: (**chain**: 4, **draw**: 1000)

▾ Coordinates:

chain	(chain)	int64	0 1 2 3
draw	(draw)	int64	0 1 2 3 4 5 ... 995 996 997 998 999

▾ Data variables:

m	(chain, draw)	float64	120.6 122.4 118.9 ... 122.9 120.3
sig	(chain, draw)	float64	6.5 8.054 8.178 ... 6.398 8.764

▸ Indexes: (2)

▸ Attributes: (6)

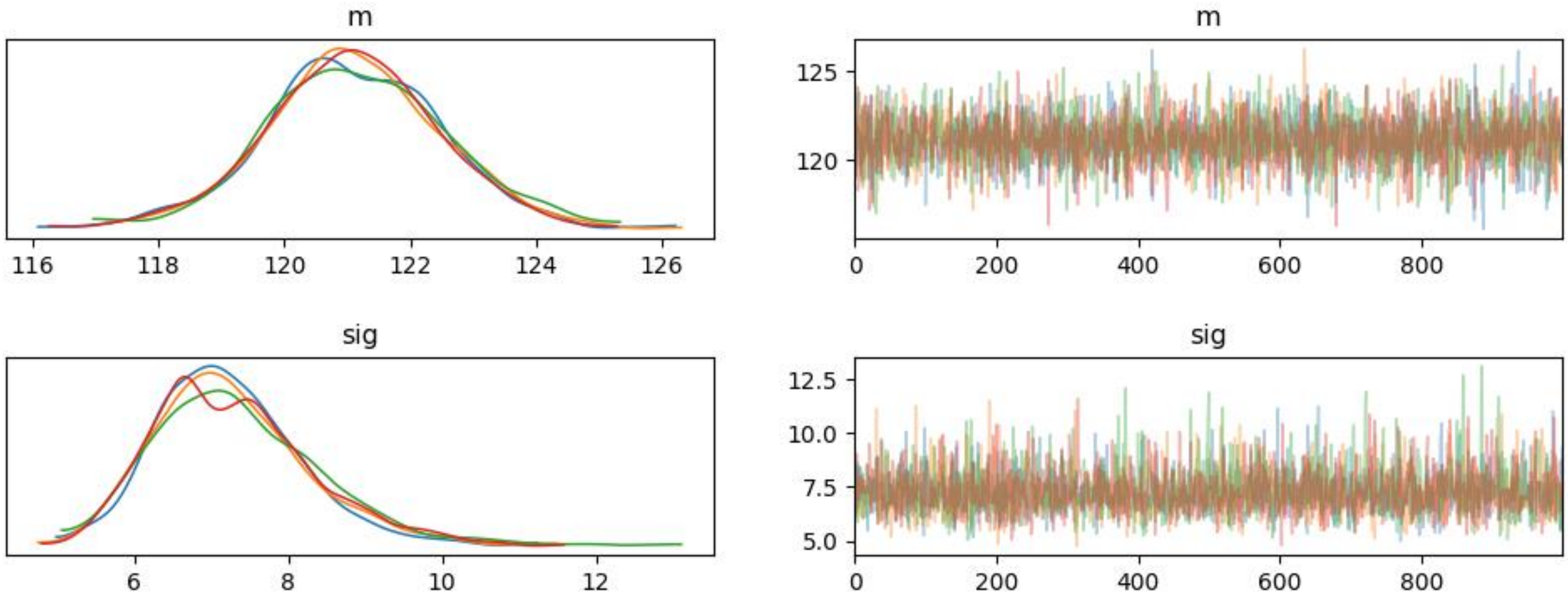
▸ sample_stats

▸ observed_data

- We have four chains for each the mu and the sigma
  - They were sampled **jointly**

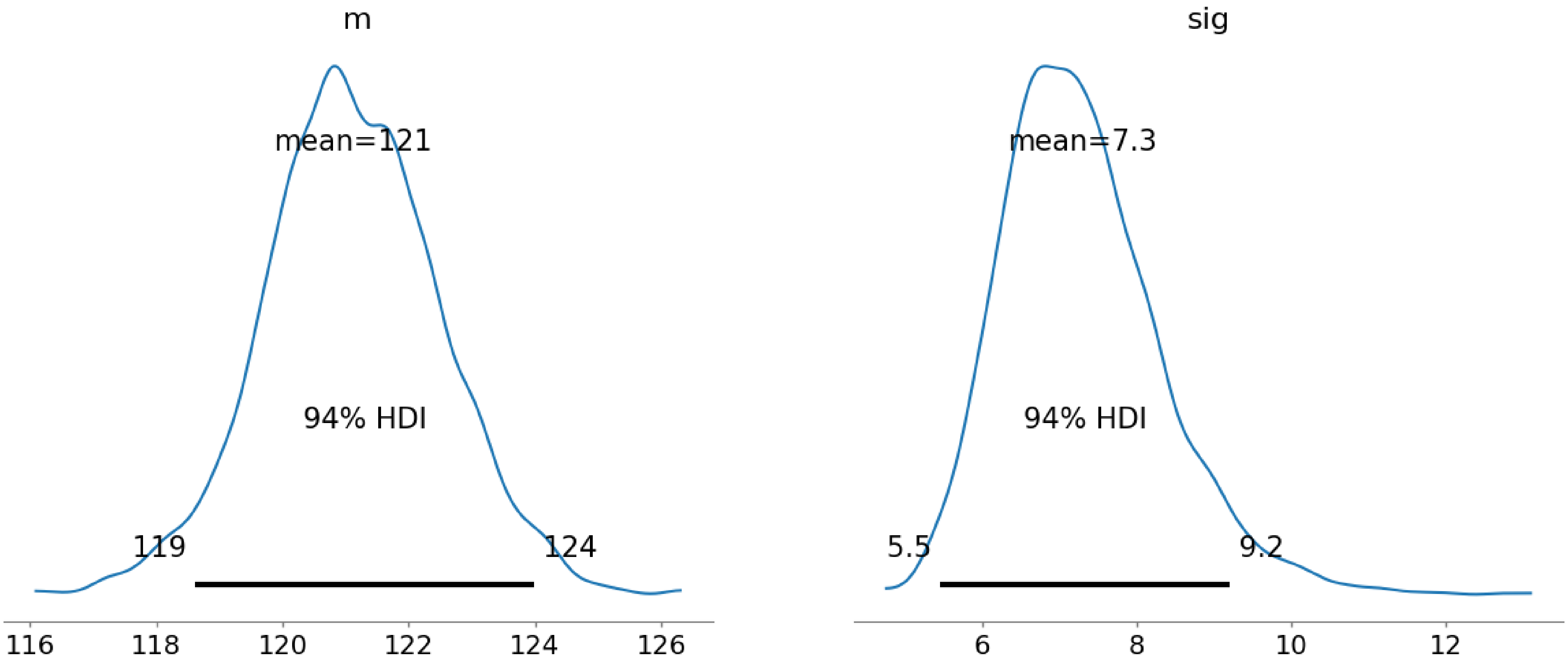
Posterior

- KDE
  - The chains overlap



Posterior

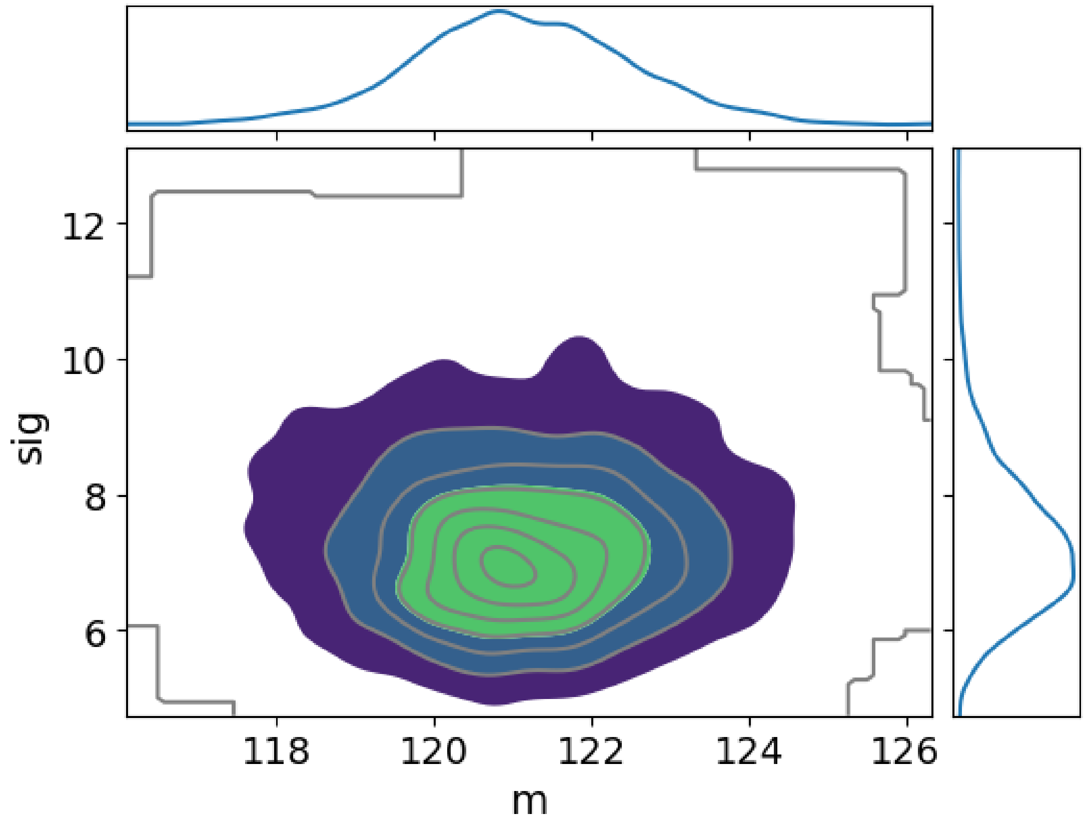
- 94% HDI



Posterior

- They were sampled **jointly**
- Every sample has each a value for mu and one for sigma
  - Are they correlated?
    - Not really (one doesn't grow or shrink

with the other)



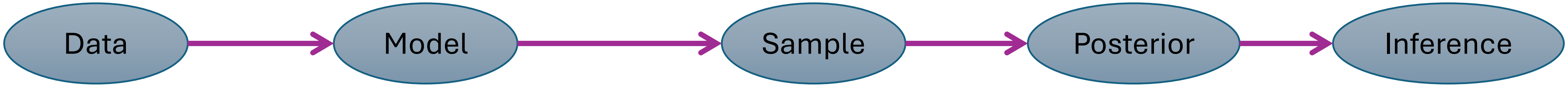
Posterior

- Let’s print the statistics

```
az.summary(idata_g, kind="stats").round(2)
```

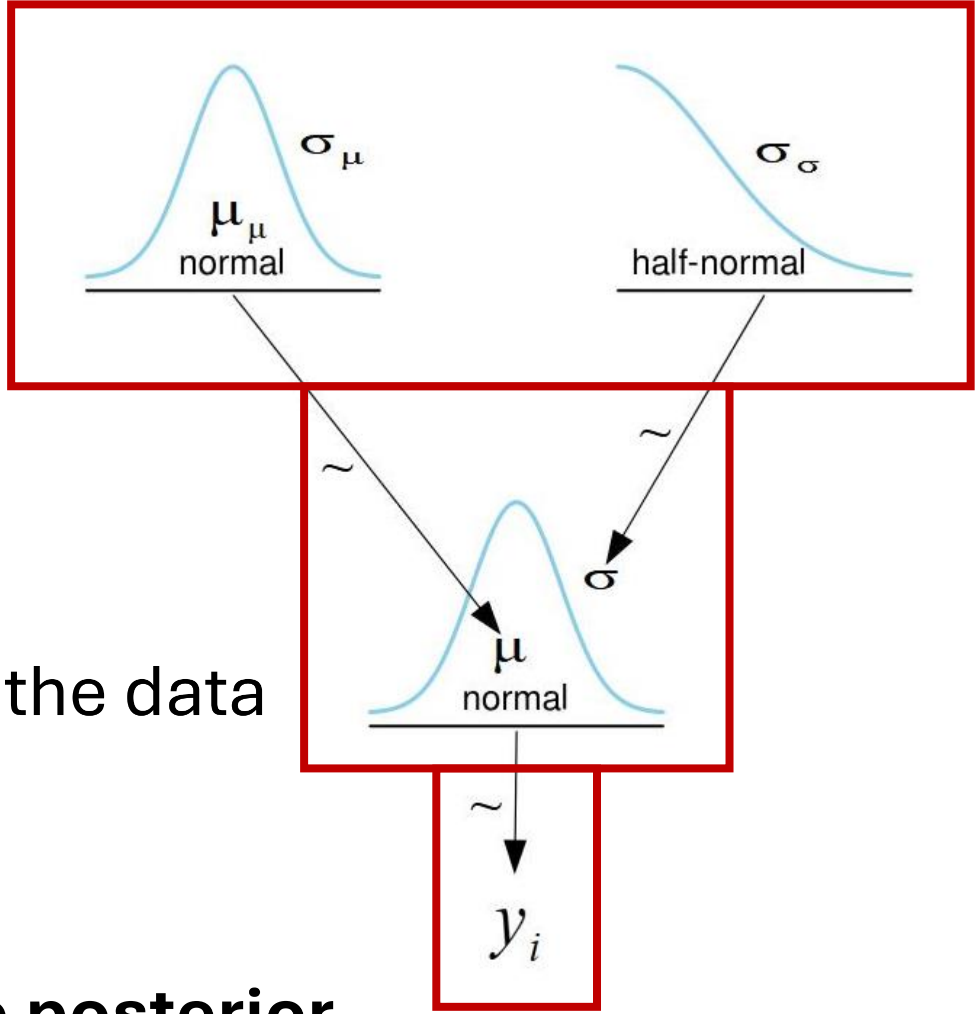
	mean	sd	hdi_3%	hdi_97%
m	121.11	1.39	118.62	123.98
sig	7.29	1.04	5.45	9.20

Bayesian Workflow

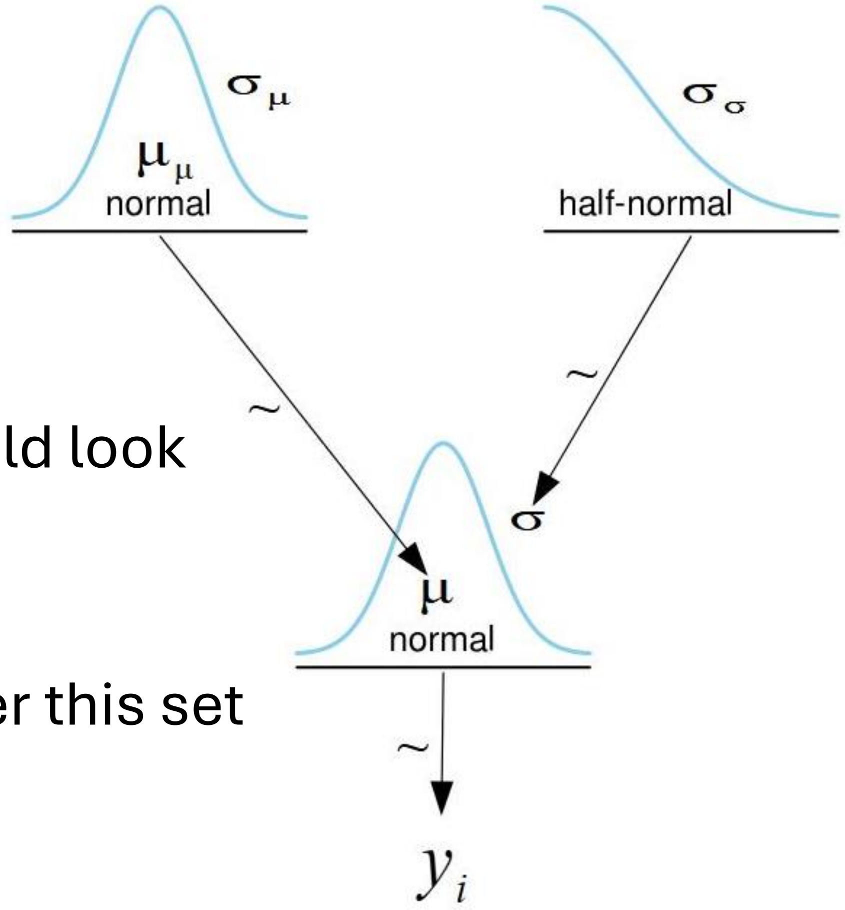


Prior Predictive Checks

- The priors are distributions
- Sample values of the priors
- Input them into the distribution describing the data
- Sample datasets
- **This is about checking the priors, not the posterior**



Prior Predictive Checks



- What does this give us?
  - We're more comfortable with what the data should look like than with what the priors should look like.
  - This shows us what the data would look like under this set of priors.
  - If we picked priors that make sense, but the data we collected is not possible given these priors:
    - Something's wrong with the priors
    - Something's wrong with the data

Prior Predictive Checks

```
with model_g:
  prior_predictive = pm.sample_prior_predictive(1000)

prior_predictive
```

▼ prior

xarray.Dataset

► Dimensions:	(**chain**: 1, **draw**: 1000)		
▼ Coordinates:			
chain	(chain)	int64	0
draw	(draw)	int64	0 1 2 3 4 5 ... 995 996 997 998 999
▼ Data variables:			
m	(chain, draw)	float64	97.79 101.1 54.88 ... 231.0 146.8
sig	(chain, draw)	float64	3.999 8.197 2.166 ... 16.87 6.452
► Indexes: (2)			
► Attributes: (4)			

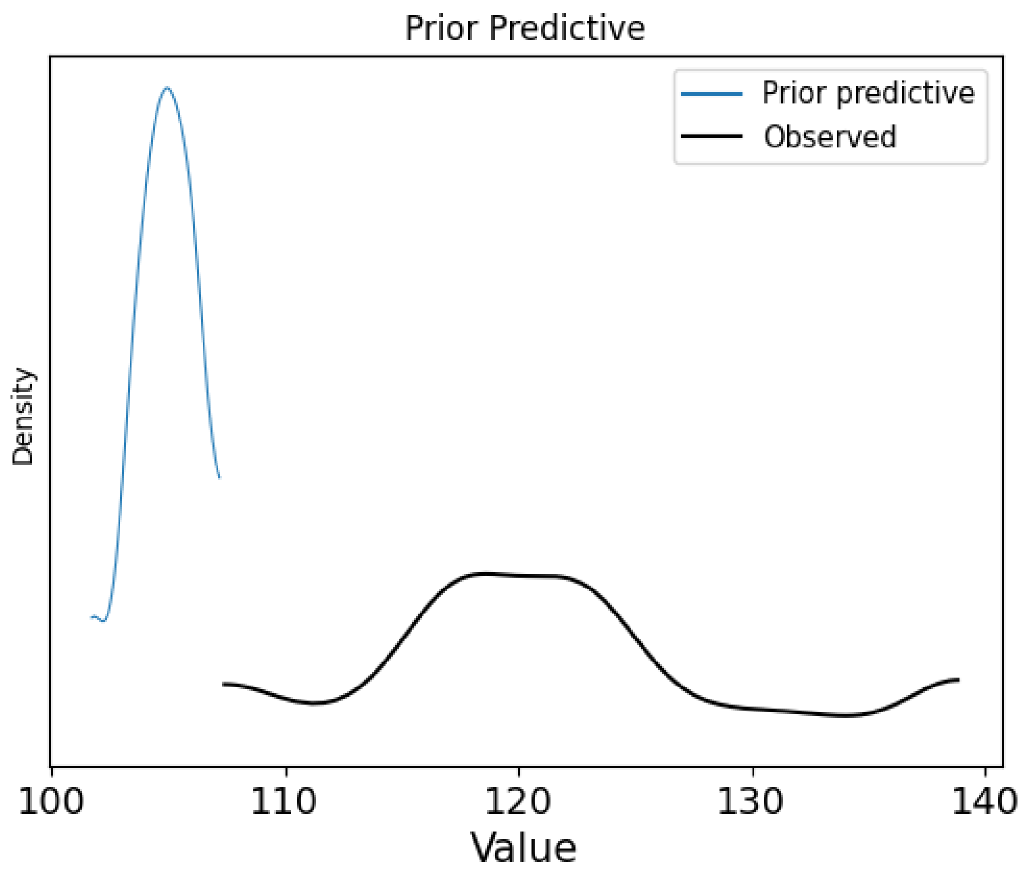
▼ prior_predictive

xarray.Dataset

► Dimensions:	(**chain**: 1, **draw**: 1000, **data**: 30)		
▼ Coordinates:			
chain	(chain)	int64	0
draw	(draw)	int64	0 1 2 3 4 5 ... 995 996 997 998 999
data	(data)	int64	0 1 2 3 4 5 6 ... 24 25 26 27 28 29
▼ Data variables:			
Y	(chain, draw, data)	float64	95.2 97.71 102.8 ... 140.0 154.3

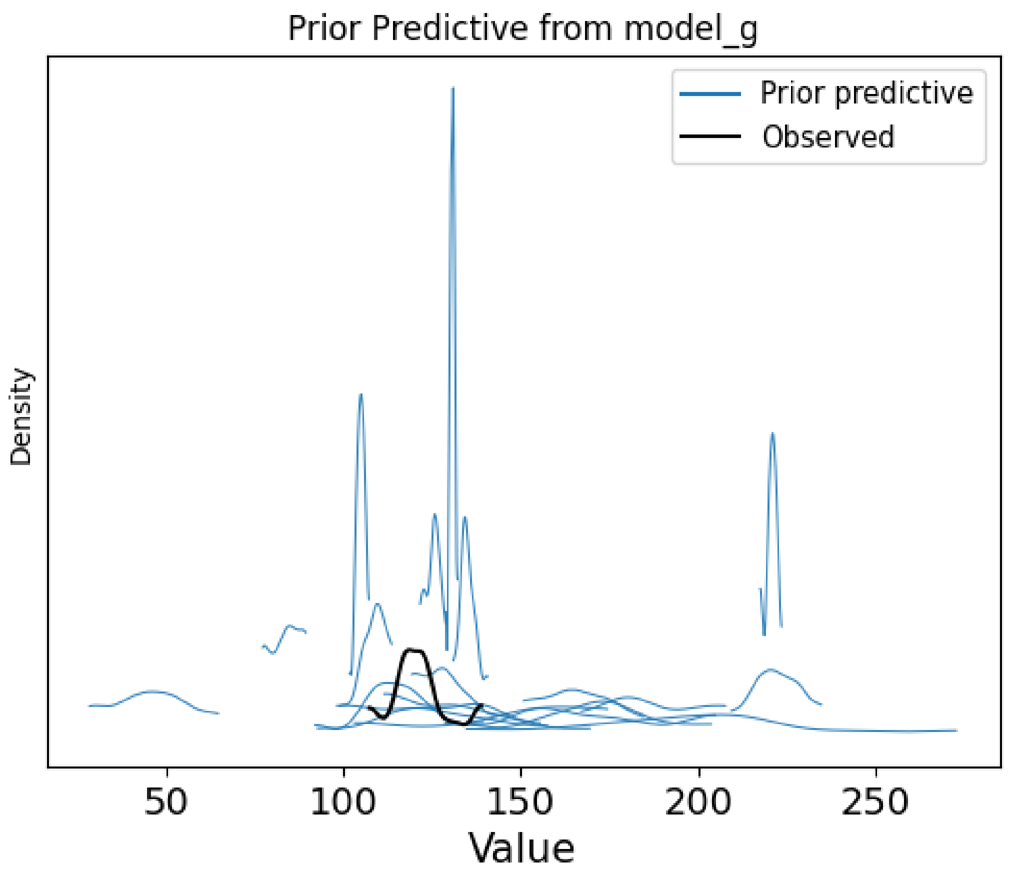
Prior Predictive Checks

```
ax = az.plot_ppc(prior_predictive, group='prior', num_pp_samples=1, mean=False, observed=True, random_seed=14)
ax.get_lines()[2].set_alpha(1.0)
plt.xlabel('Value')
plt.ylabel('Density')
plt.title('Prior Predictive')
```



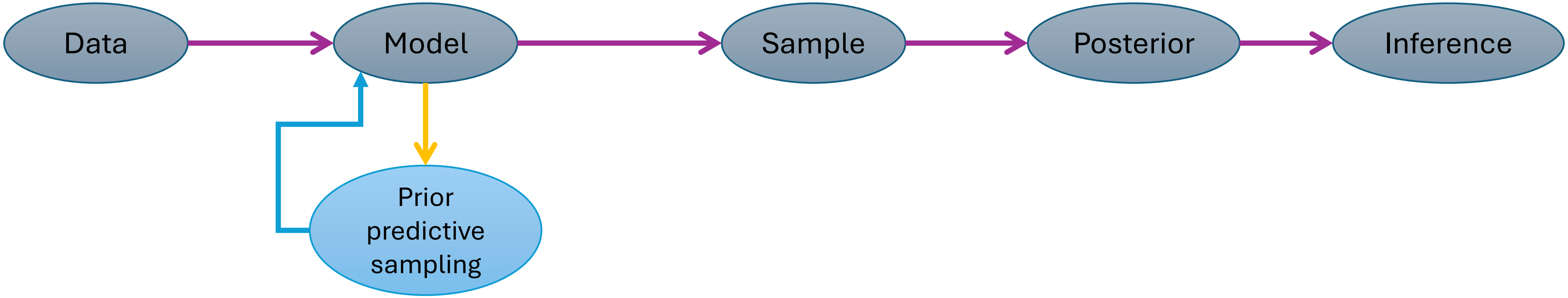
Prior Predictive Checks

```
plt.figure(figsize=(10, 6))
ax = az.plot_ppc(prior_predictive, group='prior', num_pp_samples=20, mean=False, observed=True, random_seed=14)
```

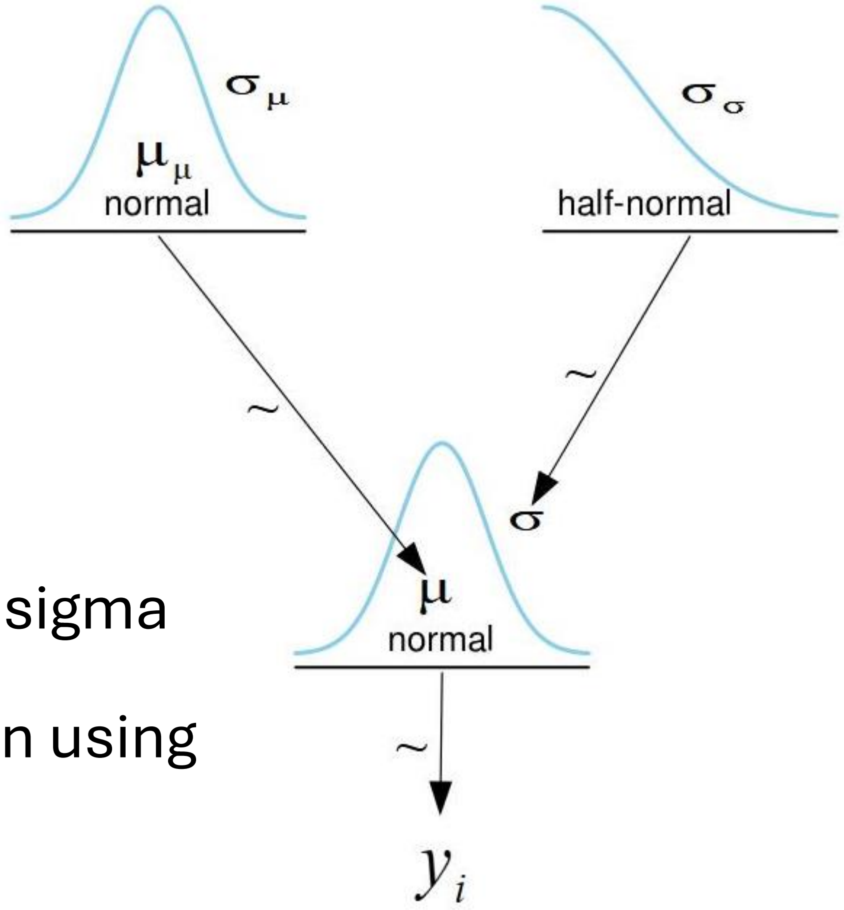


- The observed data should be possible.
- And the possible datasets should be possible.

Bayesian Workflow



Posterior Predictive Checks



- Check ability of model to explain data
- Pick parameters from posterior distribution
  - Meaning, we are still choosing values for mu and sigma
  - And then getting data from our normal distribution using those values.
  - However, in this case, mu and sigma are from the posterior and therefore take into account both the prior and the data as a result of the Bayesian Inference

Posterior Predictive Checks

```
pm.sample_posterior_predictive(idata_g, model=model_g, extend_inferencedata=True)
```

```
pm.sample_posterior_predictive(idata_g, model=model_g, extend_inferencedata=True)
```

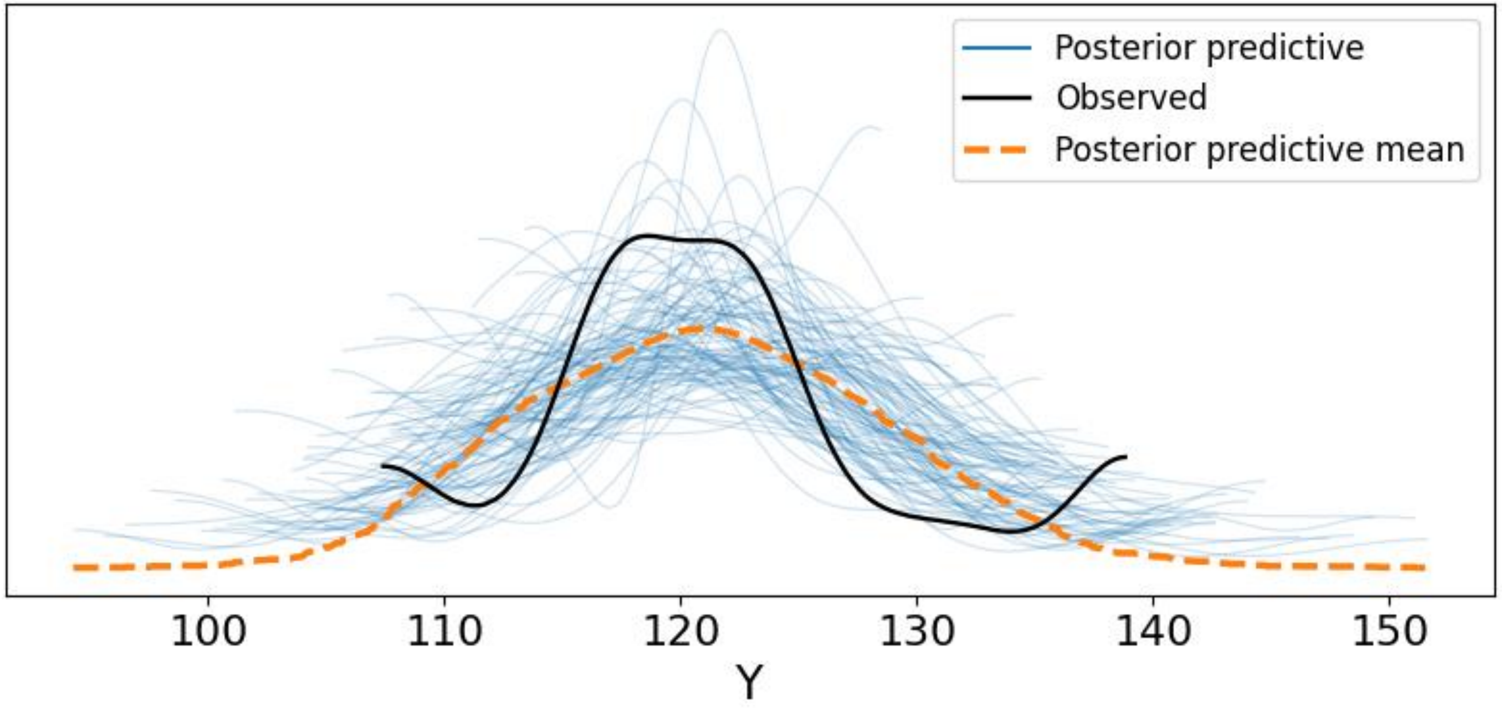
```
Sampling ... 100% 0:00:00 / 0:00:00
```

arviz.InferenceData

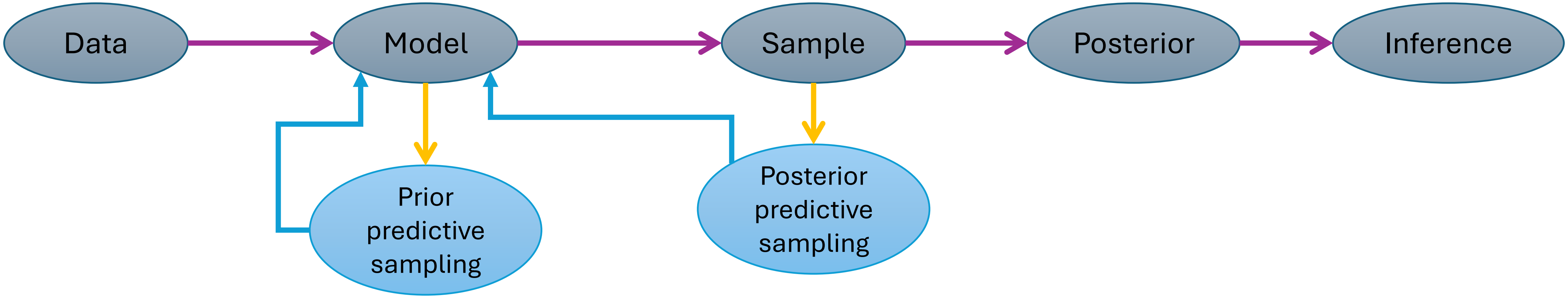
- ▶ posterior
- ▶ posterior_predictive
- ▶ sample_stats
- ▶ observed_data

Posterior Predictive Checks

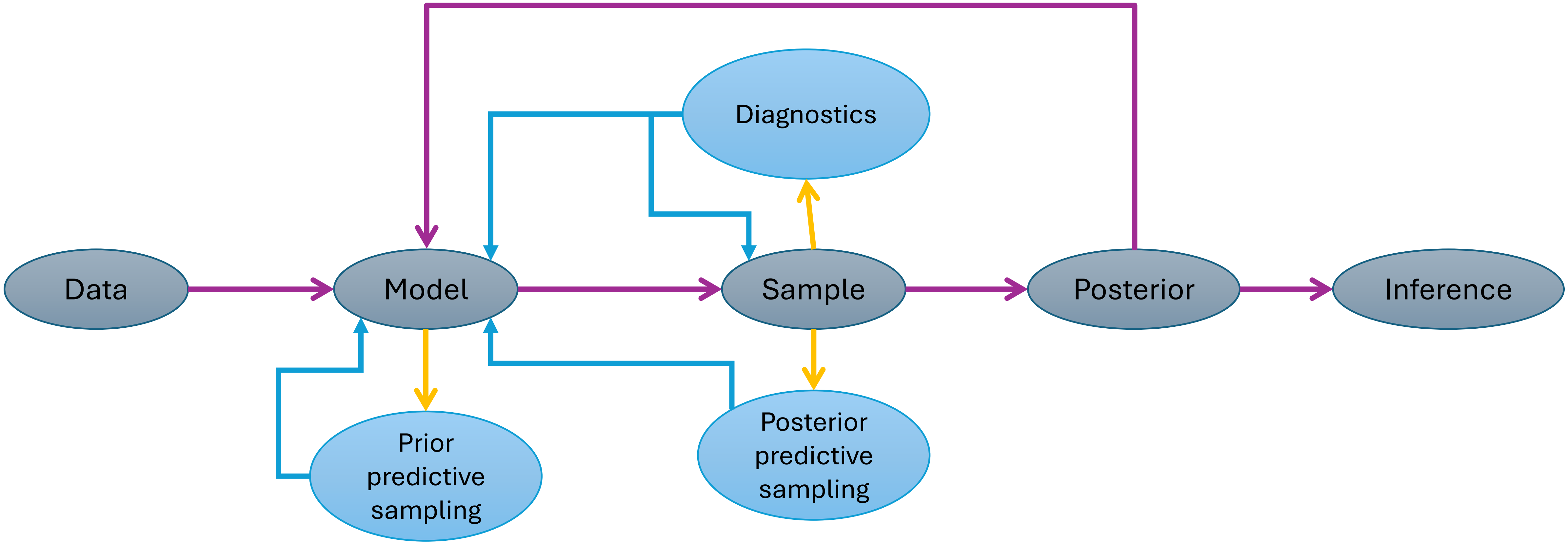
- The posterior predictive data has a shifted mode
- The posterior predictive data has larger variance
- The posterior predictive data has lighter tails
  - Pretty good
  - Outliers



Bayesian Workflow

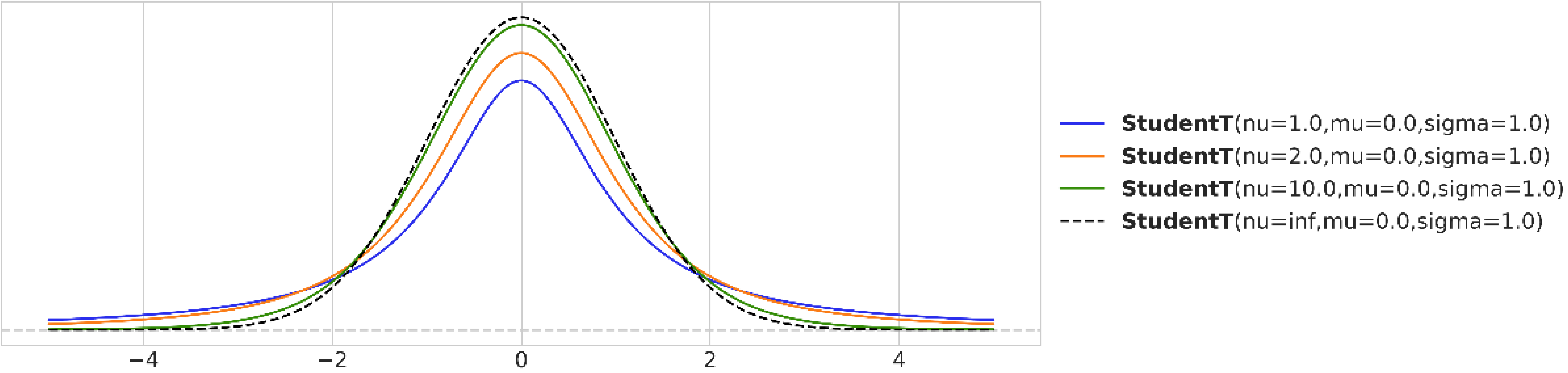


Bayesian Workflow



Student's t-distribution

- Similar to Normal distribution but with thicker tails
  - Has a parameter ν, which is the normality parameter (or degrees of freedom)
  - Converges to normal distribution as $\nu \to \infty$

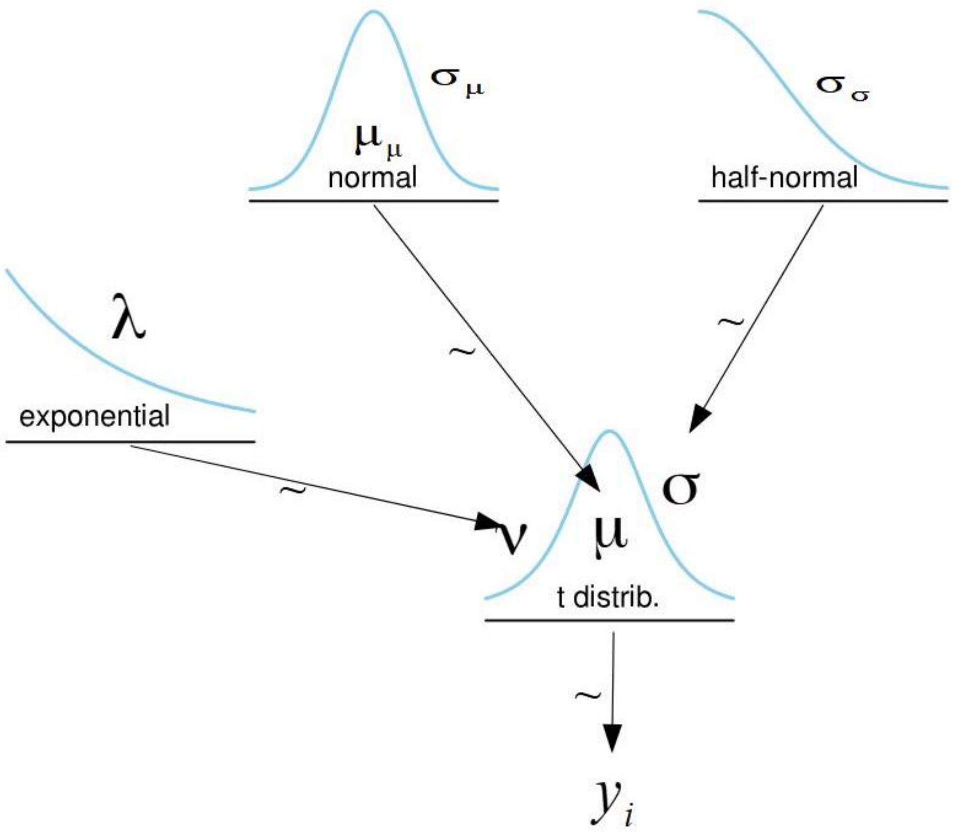


Student’s t-distribution

- The thicker tails allows for capturing the outliers without necessitating an increase in the variance.
  - Higher $\nu \Rightarrow$ Higher σ because the student’s t distribution will be more like a normal distribution and capture the outlier information less well.
- Repeat our analysis with the student’s t distribution

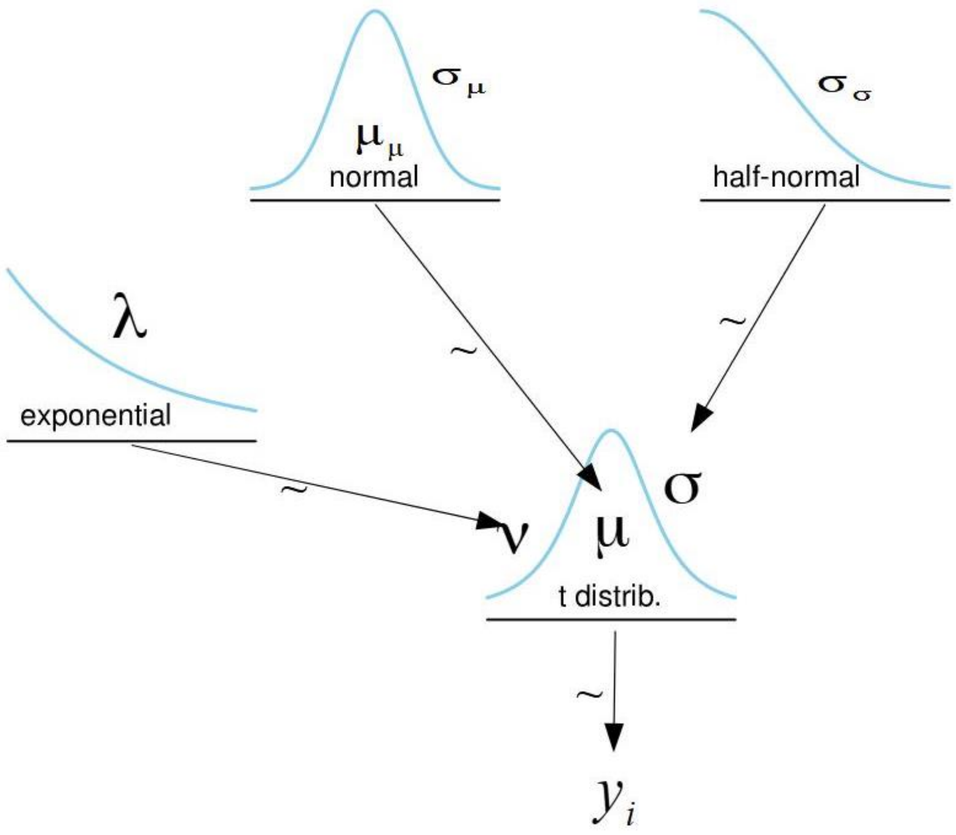
Student’s t-distribution

- The thicker tails allows for capturing the outliers without necessitating an increase in the variance.



Student's t-distribution

Graphical model



Equations

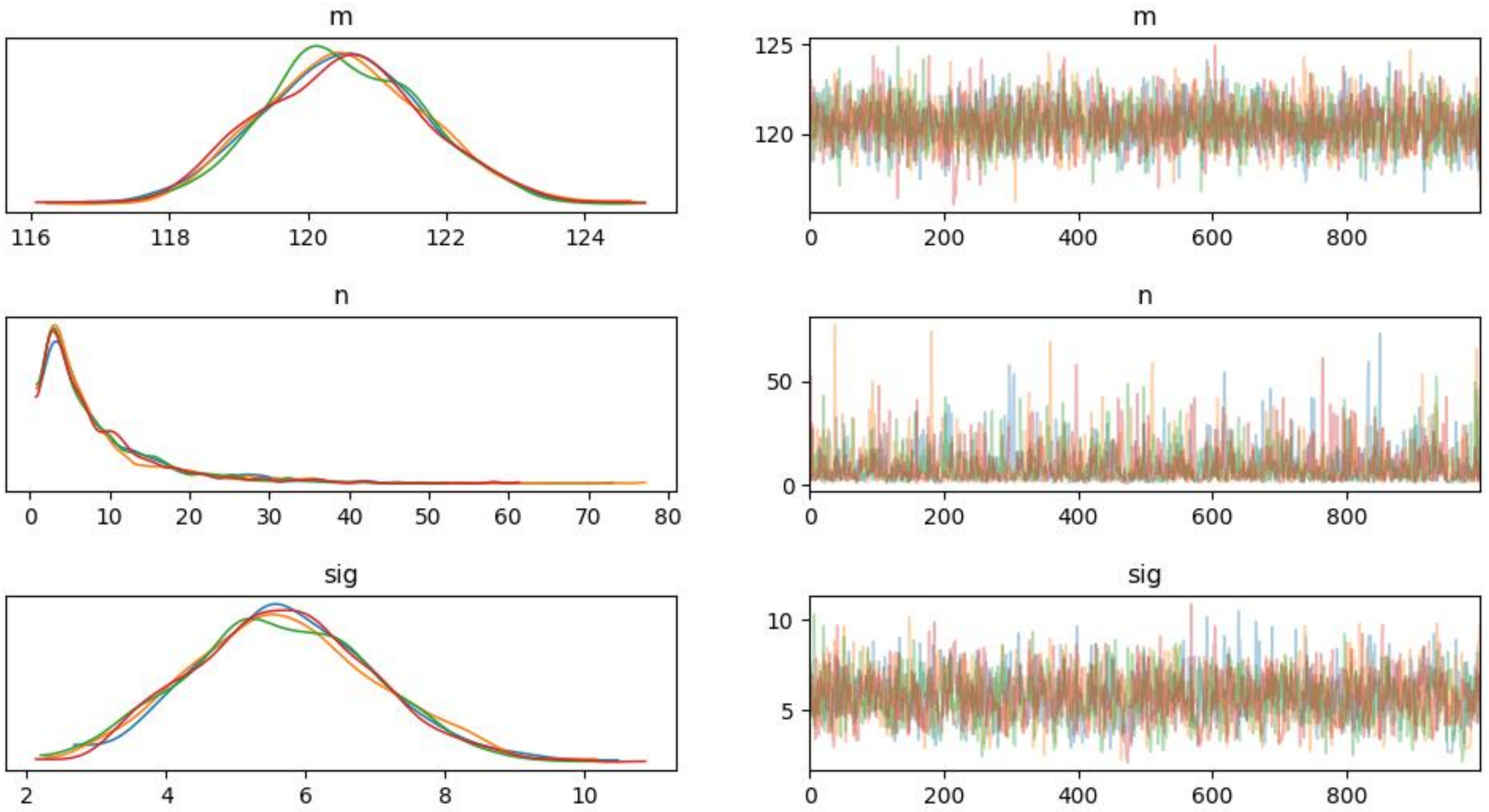
$$y_i \sim t(\nu, \mu, \sigma)$$
$$\mu \sim N(\mu_\mu, \sigma_\mu)$$
$$\sigma \sim HalfNorm(\sigma_\sigma)$$
$$\nu \sim Exp(\lambda)$$

PyMC

```
with pm.Model(coords = coords) as model_t:
  m = pm.Normal('m', mu = mu_prior, sigma = sig_prior)
  sig = pm.HalfNormal('sig', sigma = sigma_prior)
  n = pm.Exponential('n', 1/10)
  Y = pm.StudentT('Y', nu = n, mu=m, sigma=sig, observed=data, dims = 'data')
  idata_t = pm.sample(1000, chains = 4)
```

Posterior

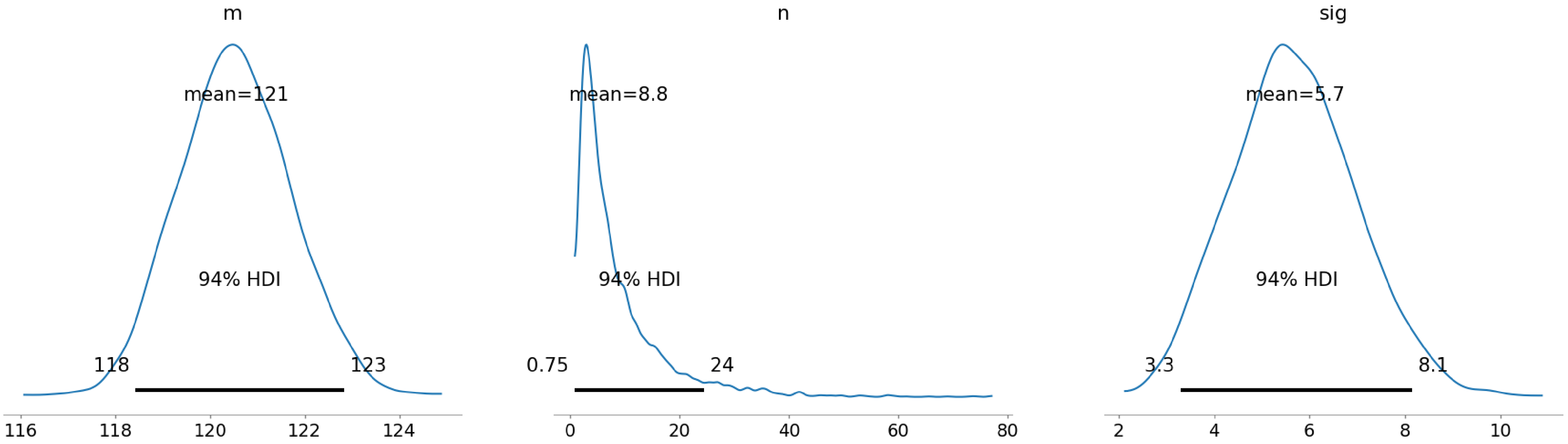
- KDE



Posterior

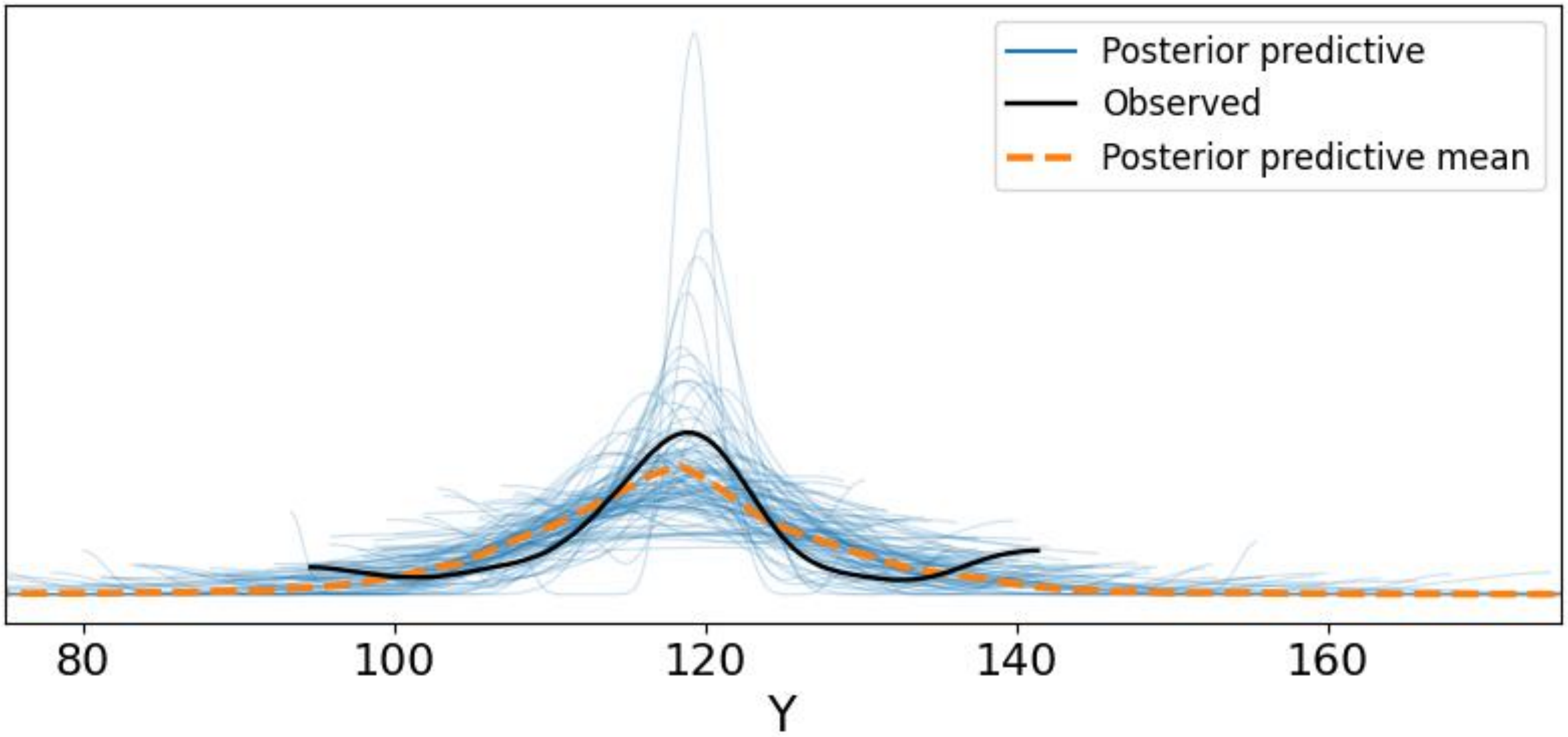
- 94% HDI

	mean	sd	hdi_3%	hdi_97%
m	120.54	1.19	118.42	122.83
n	8.75	8.50	0.75	24.46
sig	5.69	1.31	3.30	8.14



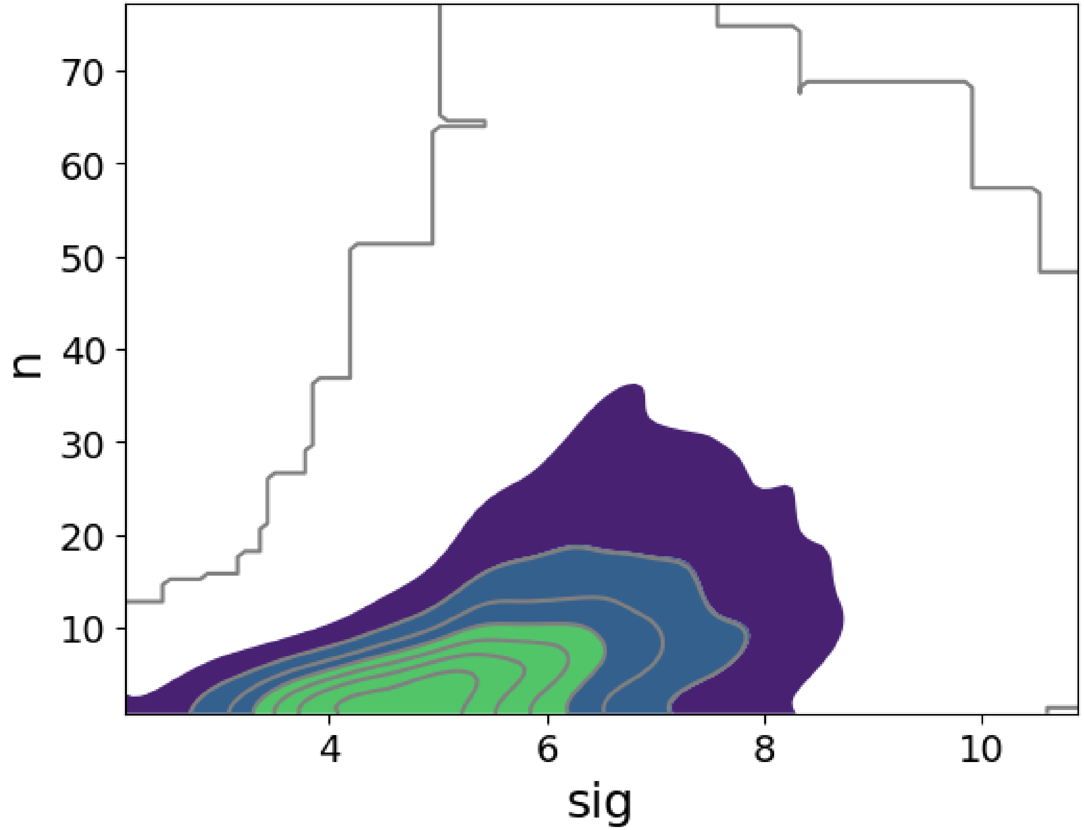
Posterior Predictive Checks

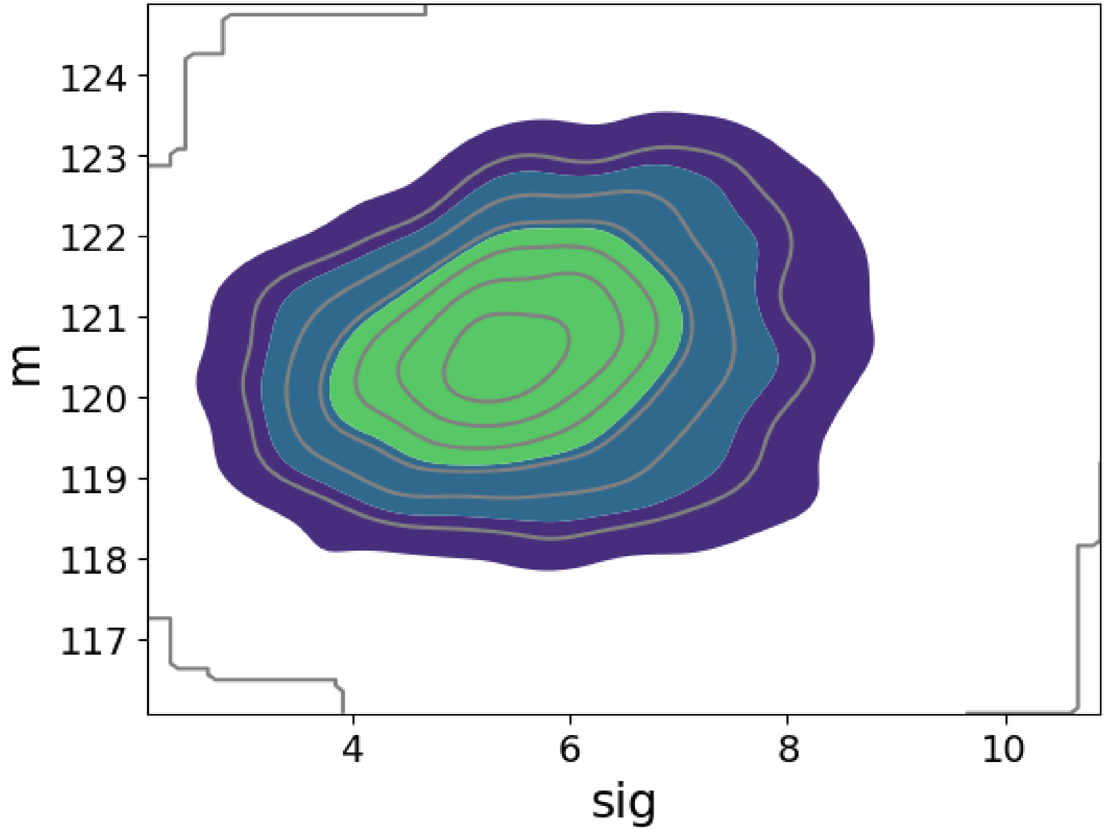
- Better fit than the normal distribution



Correlations Between Parameters

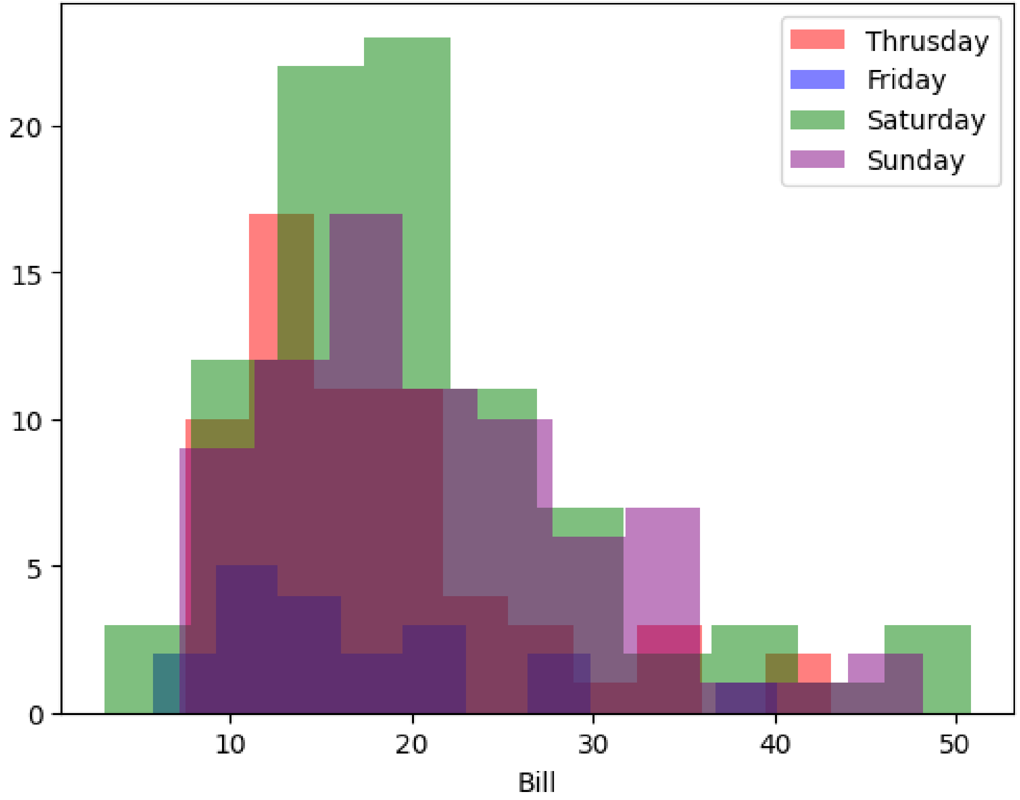
- Between nu and sigma
- But not between sigma and mu





Groups Comparison

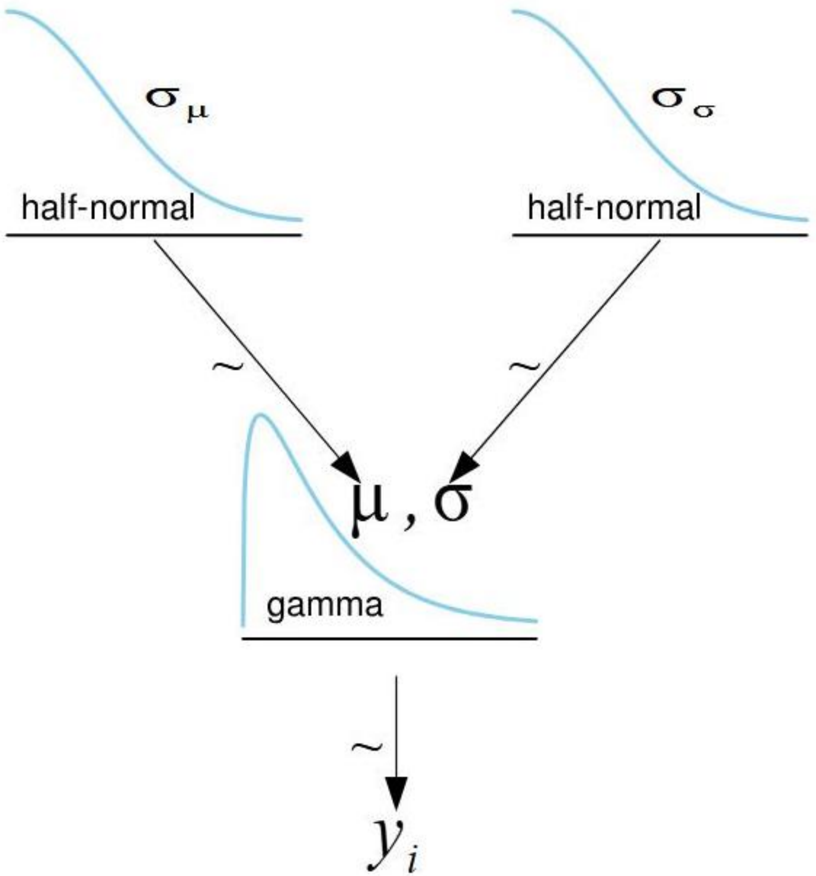
- Let's look at the tips dataset, but instead of comparing the tips on the different days, let's compare the bill values on the different days.



Groups Comparison

- Let's model the data for each day using a gamma distribution.

Graphical model



Equations

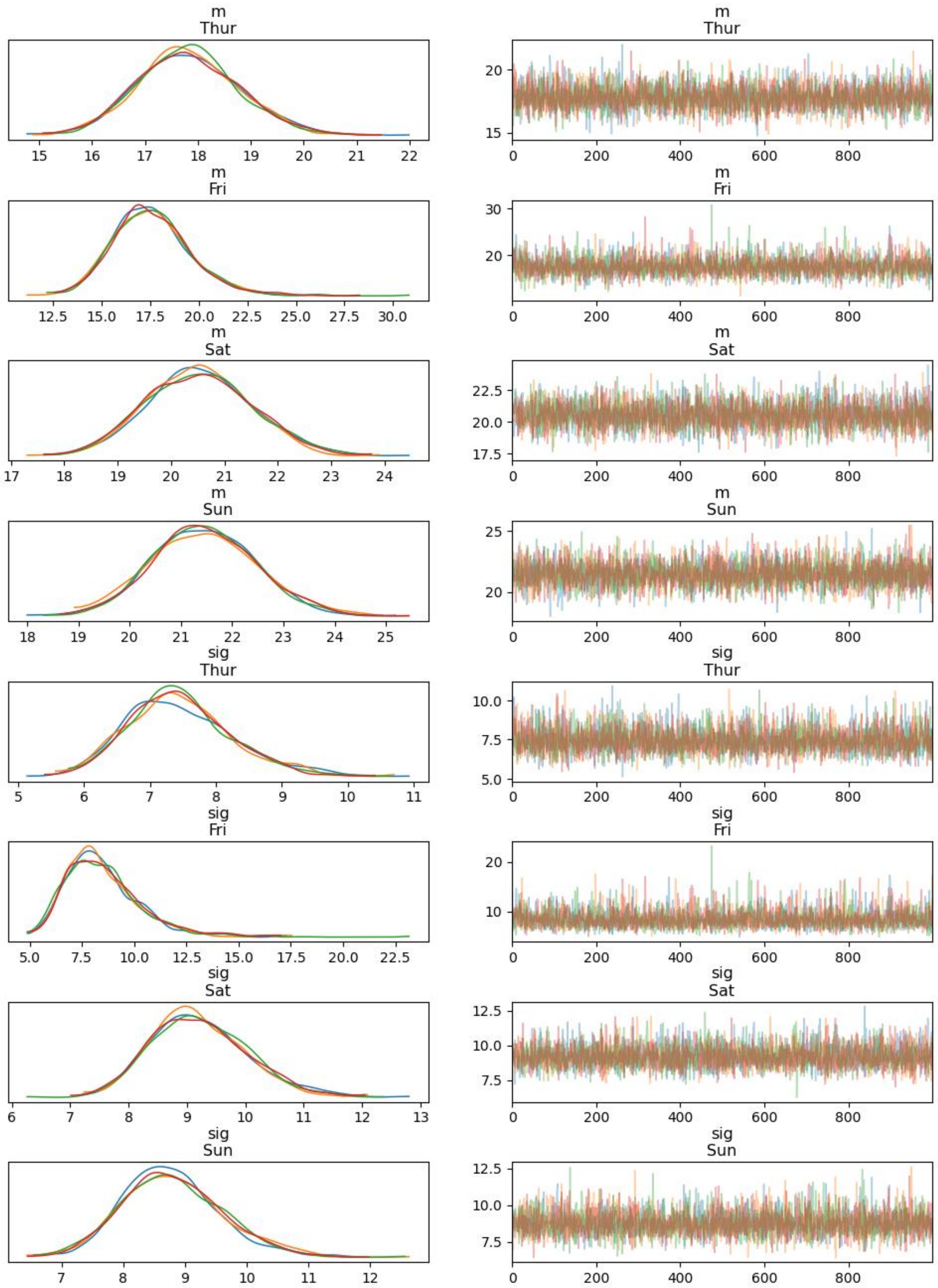
$$y_i \sim gamma(\mu, \sigma)$$
$$\mu \sim HalfNorm(\sigma_\mu)$$
$$\sigma \sim HalfNorm(\sigma_\sigma)$$

PyMC

```
#creating the model
coords = {"days": categories, "days_flat":categories[idx]}

with pm.Model(coords=coords) as comparing_groups:
  m = pm.HalfNormal("m", sigma = 20, dims="days")
  sig = pm.HalfNormal("sig", sigma=30, dims="days")

  y = pm.Gamma("y", mu=m[idx], sigma=sig[idx], observed = bill, dims = "days_flat")

  idata_cg = pm.sample(1000, chains = 4)
  idata_cg.extend(pm.sample_posterior_predictive(idata_cg))
```

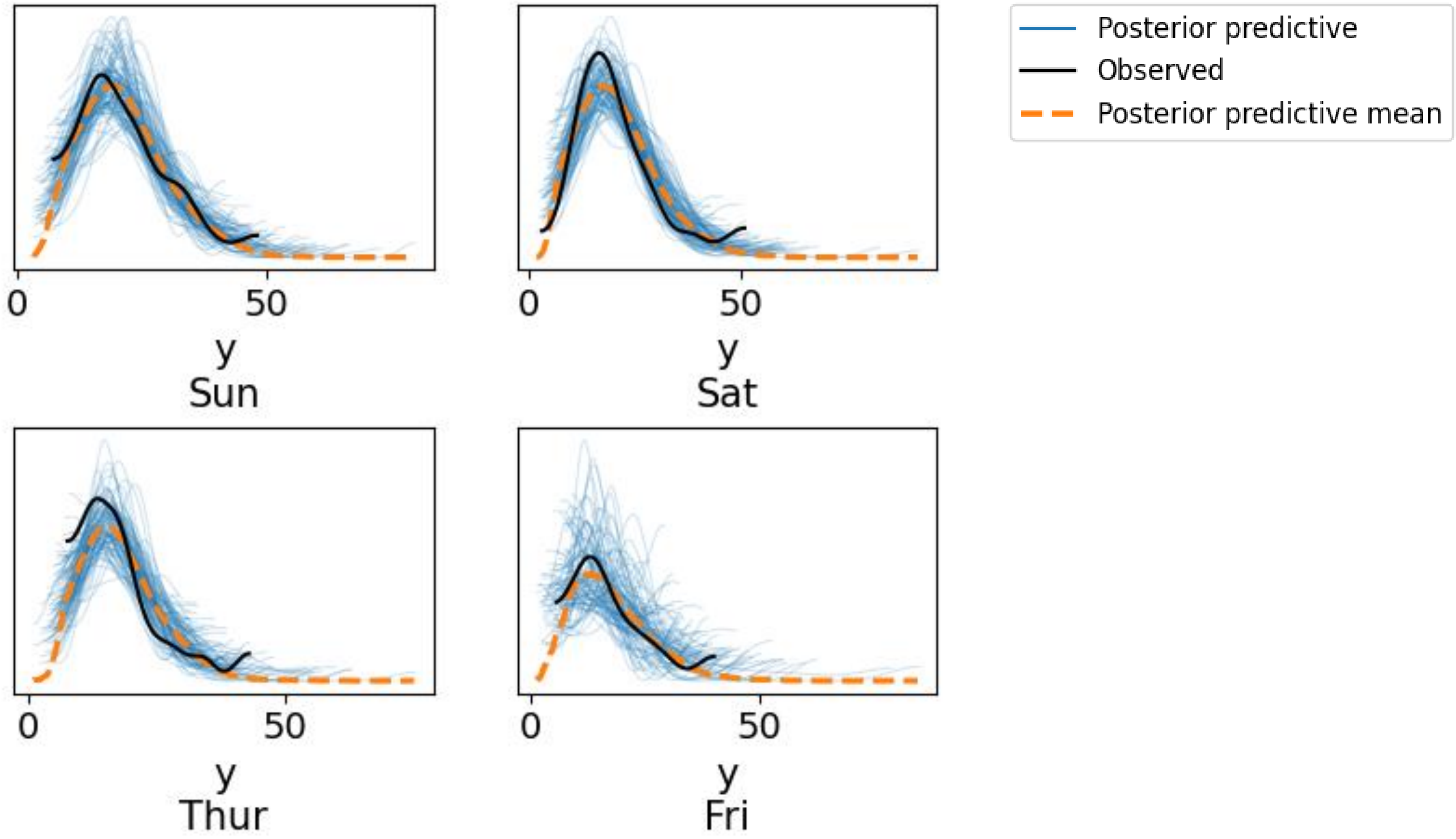
Groups Comparison

- Look at KDE plots



Groups Comparison

- Posterior predictive checks



Groups Comparison

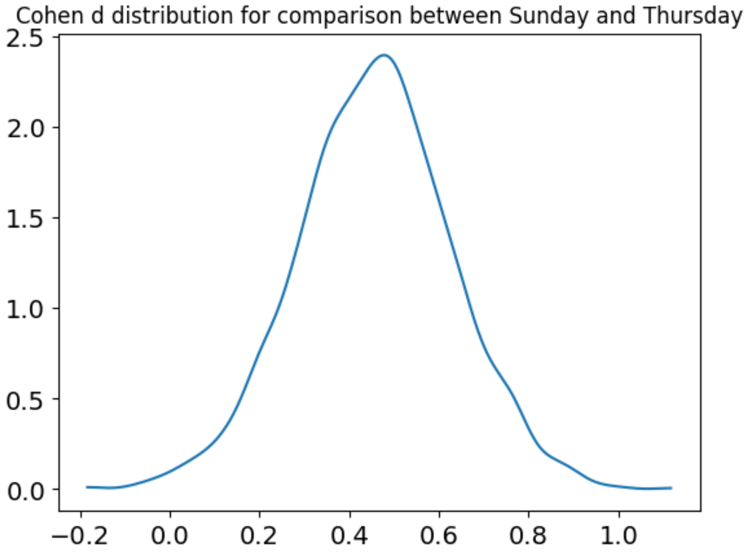
Effect size	
0.2	Small
0.5	Medium
0.8	Large

- Cohen's d
- $cohen's\ d = \frac{(\mu_2 - \mu_1)}{\sqrt{\frac{\sigma_1^2 + \sigma_2^2}{2}}}$
- As we have a distribution for each, we can compute the cohen's d distribution.
- If we are interested in a single value, we can compute the mean, median or mode of the distribution.

Groups Comparison

- The distribution for comparing Sunday and Thursday

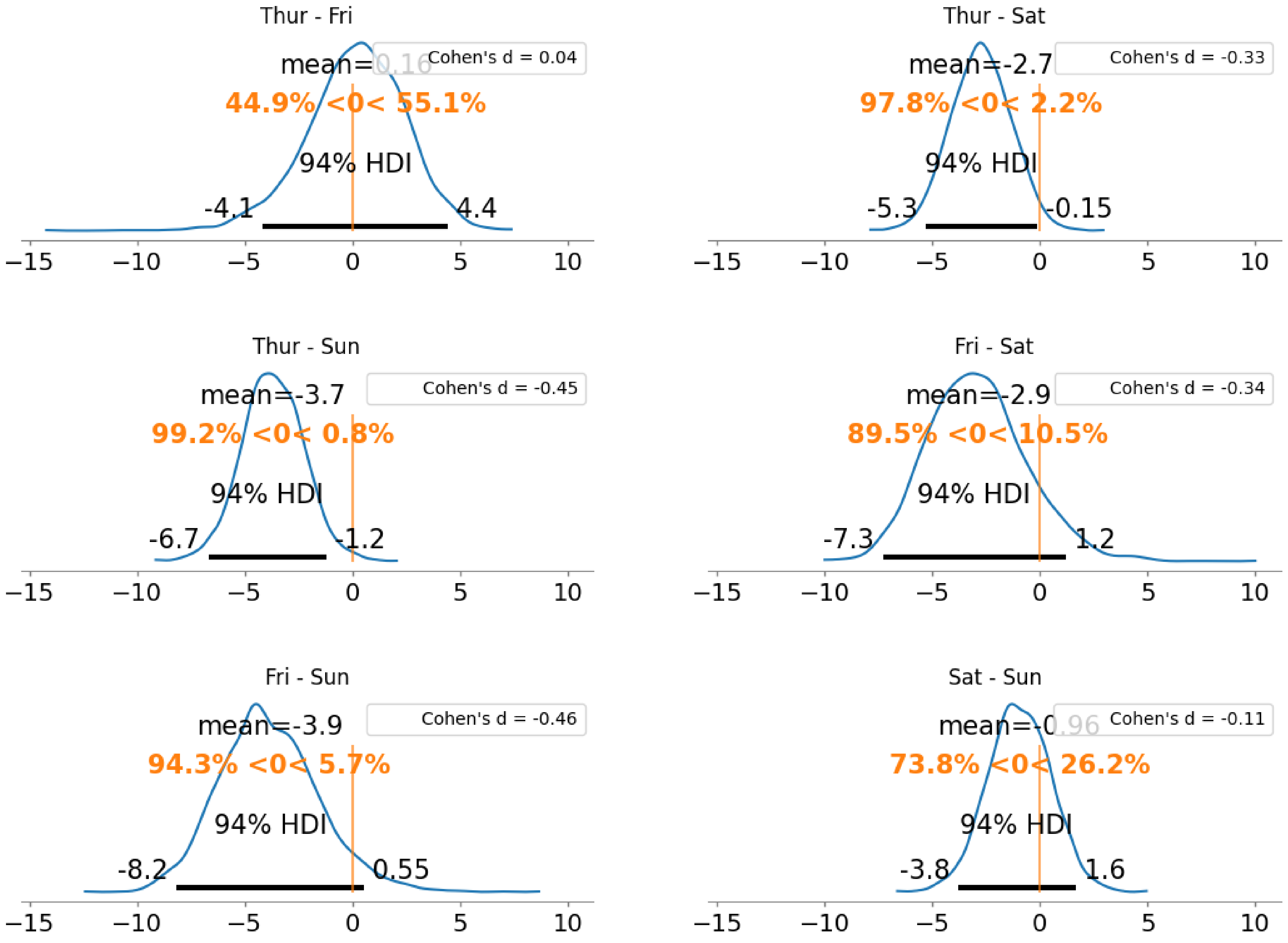
```
means_diff = cg_posterior["m"].sel(days='Sun') - cg_posterior['m'].sel(days='Thur')

d_cohen = (means_diff /
           np.sqrt((cg_posterior["sig"].sel(days='Sun')**2 +
                   cg_posterior["sig"].sel(days='Thur')**2) / 2))


az.plot_dist(d_cohen)
plt.title('Cohen d distribution for comparison between Sunday and Thursday')
```



Groups Comparison

```
comparisons = [(categories[i], categories[j]) for i in range(4) for j in range(i+1, 4)]
#print(comparisons)
cg_posterior = az.extract(idata_cg)

for (i, j) in comparisons:
    means_diff = cg_posterior["m"].sel(days=i) - cg_posterior['m'].sel(days=j)

    d_cohen = (means_diff /
               np.sqrt((cg_posterior["sig"].sel(days=i)**2 +
                        cg_posterior["sig"].sel(days=j)**2) / 2)).mean().item()

    print(f"{i} - {j}: cohen's d = {round(d_cohen, 2)}")
```

```
Thur - Fri: cohen's d = 0.04
Thur - Sat: cohen's d = -0.33
Thur - Sun: cohen's d = -0.45
Fri - Sat: cohen's d = -0.34
Fri - Sun: cohen's d = -0.46
Sat - Sun: cohen's d = -0.11
```



- Distributions are the difference of means.

- The orange shows how much of the distribution is above and below the set value (in this case 0 – no difference)