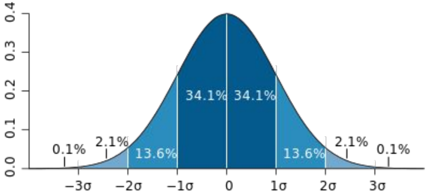
Using A Multiple-Temperature MCMC Model to More Efficiently Find the 95% Credible Interval

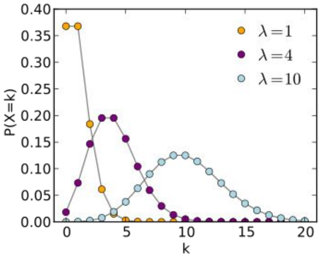
David Witten
Montgomery Blair High School

Background: Distributions

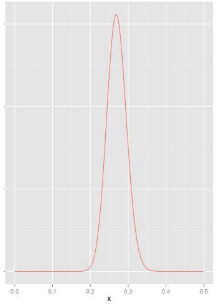
- Normal



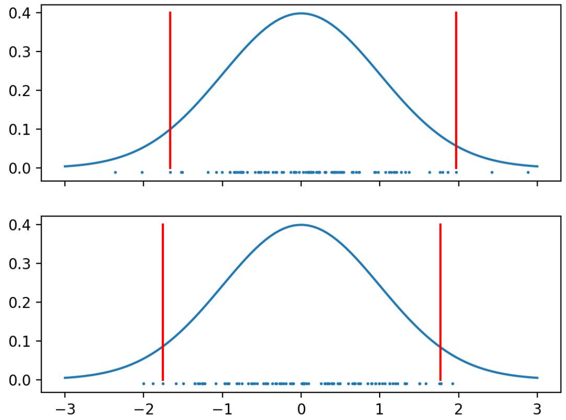
- Poisson

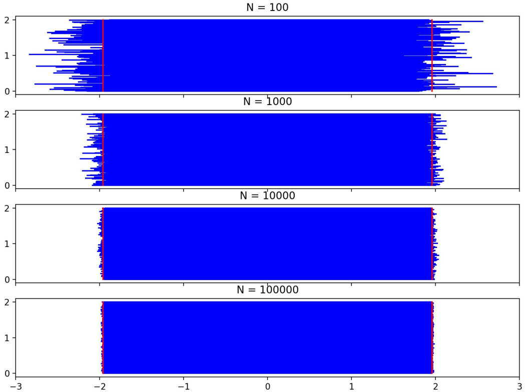


- Beta



Background





95% interval of a Normal is [–1.96, 1.96]

Middle 95% of 100 generated points

95% CI with 100, 1000, 10000, and 100000 data points

Goal: More efficiently estimate the 95% CI

Background: MCMC

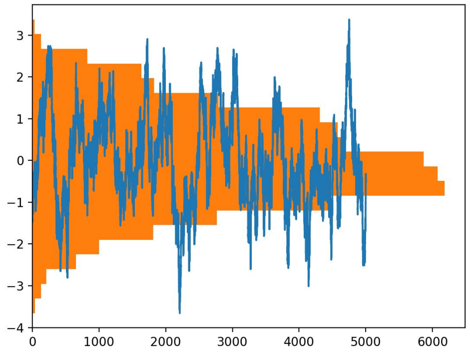
MCMC = Markov Chain Monte Carlo

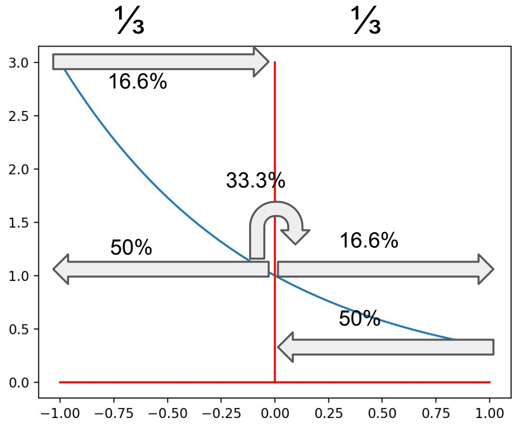
- Markov Chain = Sequence of events that only depend on the previous event
- Monte Carlo = Generated Random points
    - Reference to the Monte Carlo Casino

Background: MCMC (cont.)

Metropolis-Hastings Algorithm

$$f(x) = 3^{-x}$$



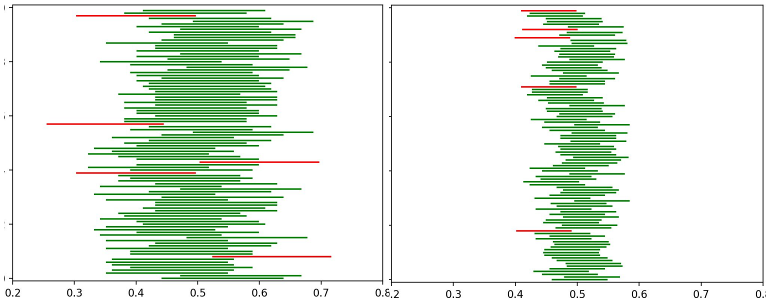


$$p(\theta \to \theta + 1) = 0.5 \min\left(\frac{P(\theta+1)}{P(\theta)}, 1\right)$$

$$p(\theta + 1 \to \theta) = 0.5 \min\left(\frac{P(\theta)}{P(\theta+1)}, 1\right)$$

$$\frac{p(\theta \to \theta + 1)}{p(\theta + 1 \to \theta)} = \frac{P(\theta+1)}{P(\theta)}$$

Background: Credible Interval



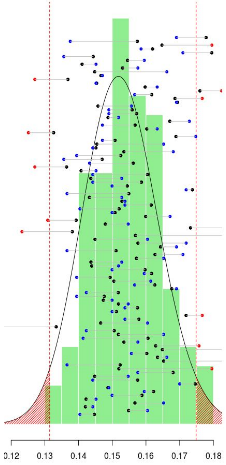
95% Confidence Interval:

95% of the 955 confidence intervals I create will contain the true value

95% Credible Interval:

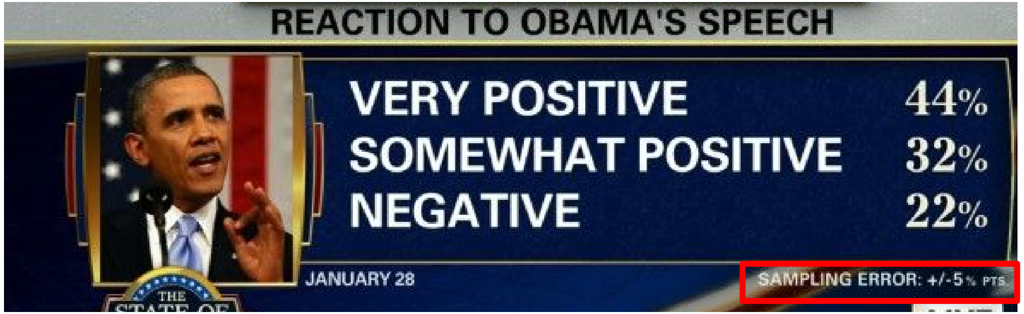
95% probability that the true value falls within a region, given some data

In simple cases, they're the **same**



Significance

- Credible intervals give an interval estimate of the parameter
    - Range of probable values
- Need 1,000,000 points to be accurate to 0.01
- This project reduced required samples by 50%

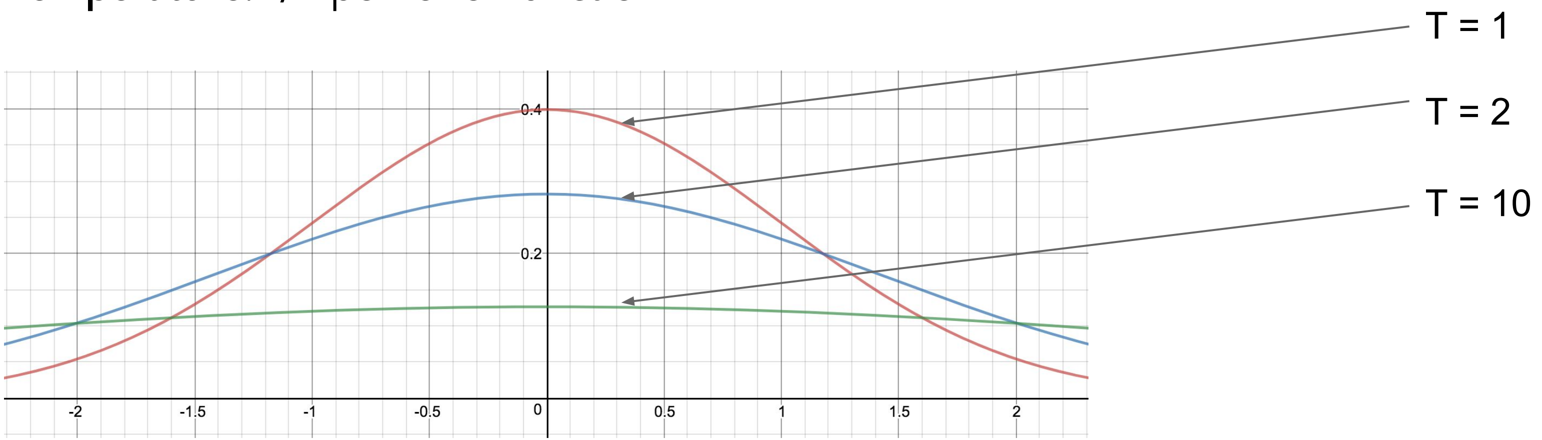


Temperature

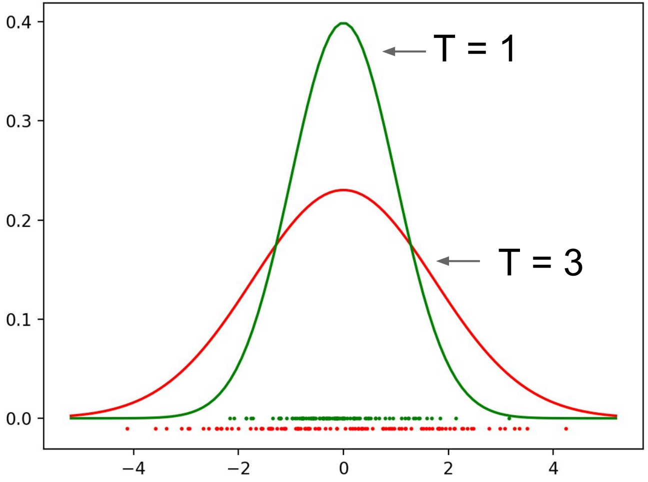
Concept from **Simulated Annealing**

- Used to the max/min of some function

Temperature: 1/T power of function



Benefit of Multiple Temperatures



At T = 1,

- Higher variation in the tails ✘
- Lower variation in the center ✔

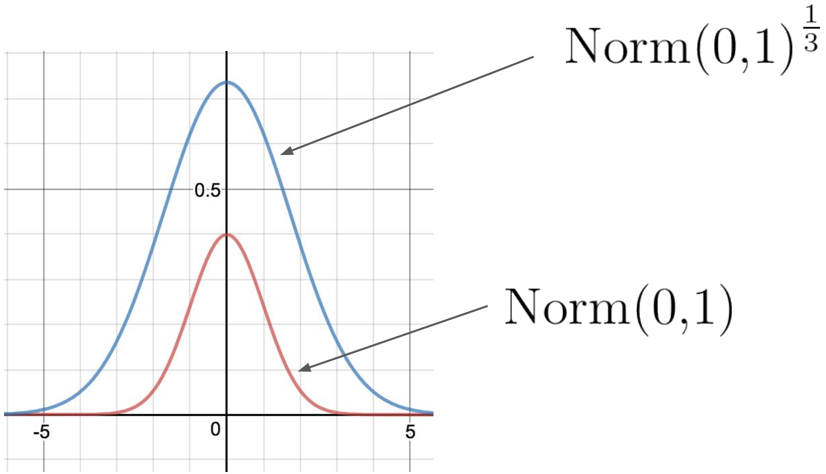
At T = 3,

- Lower variation in the tails ✔
- Higher variation in the center ✘

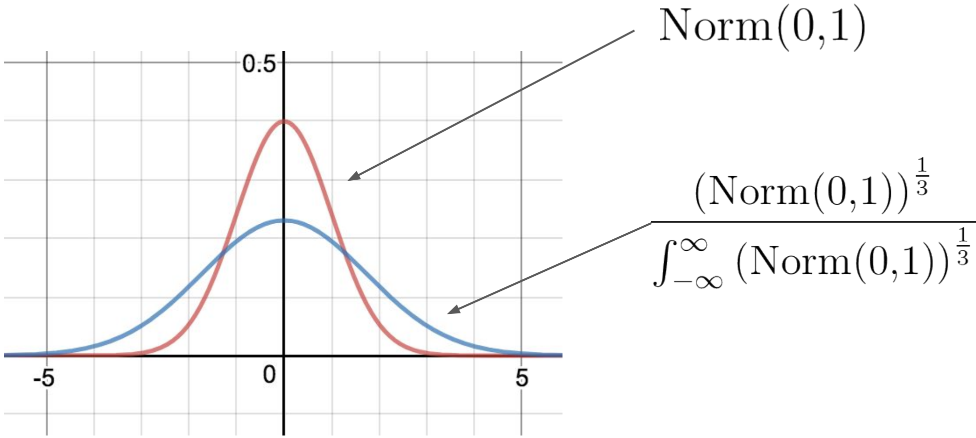
We want to combine them to keep the better parts of both

Finding the Absolute Ratio

What we have:



What we want:



$$\frac{\int f(x)\,\mathrm{d}x}{\int f^{\frac{1}{T}}(x)\,\mathrm{d}x} \approx \frac{1}{N}\sum_{i=1}^{N} \frac{f(x_i)}{f(x_i)^{\frac{1}{T}}}$$

Area of $\mathrm{Norm}(0,1)$

Area of $\mathrm{Norm}(0,1)^{\frac{1}{3}}$

Ratio of the curves on the left

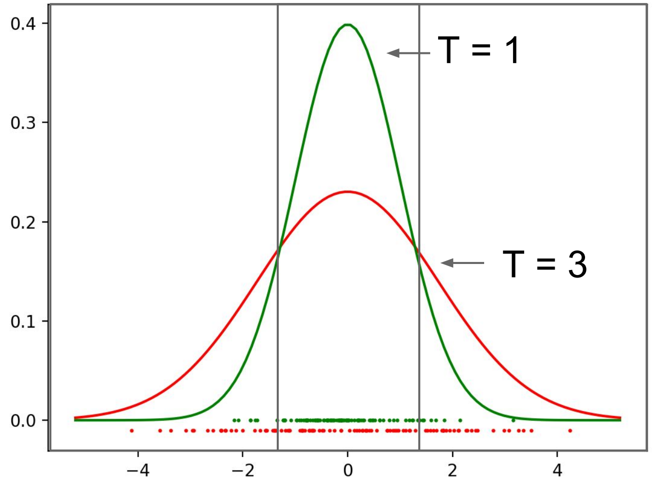
Algorithm

Weight:

- All points are weighted $\frac{f(x)}{f(x)^{\frac{1}{T}}}$

Note: At T = 1, this is 1

Algorithm:



Estimate area under tails

Estimate under center

$$1 \approx \frac{N_{t,T}}{N_{t,T}+N_{1,T}} \frac{\sum_{N_T \in \Omega_T} w_i}{\sum_{i=1}^{N_T} w_i} + \frac{N_{1,T}}{N_{t,T}+N_{1,T}} \frac{\sum_{N_1 \in \Omega_T} 1}{\sum_{i=1}^{N_1} 1} + \frac{N_{t,1}}{N_{t,1}+N_{1,1}} \frac{\sum_{N_T \in \Omega_1} w_i}{\sum_{i=1}^{N_T} w_i} + \frac{N_{1,1}}{N_{t,1}+N_{1,1}} \frac{\sum_{N_1 \in \Omega_1} 1}{\sum_{i=1}^{N_1} 1}$$

Algorithm Further Explained

Estimate area under tails | **Estimate under center**

$$1 \approx \frac{N_{t,T}}{N_{t,T}+N_{1,T}} \frac{\sum_{N_T \in \Omega_T} w_i}{\sum_{i=1}^{N_T} w_i} + \frac{N_{1,T}}{N_{t,T}+N_{1,T}} \frac{\sum_{N_1 \in \Omega_T} 1}{\sum_{i=1}^{N_1} 1} + \frac{N_{t,1}}{N_{t,1}+N_{1,1}} \frac{\sum_{N_T \in \Omega_1} w_i}{\sum_{i=1}^{N_T} w_i} + \frac{N_{1,1}}{N_{t,1}+N_{1,1}} \frac{\sum_{N_1 \in \Omega_1} 1}{\sum_{i=1}^{N_1} 1}$$

% T=3

% T=1

% T=3

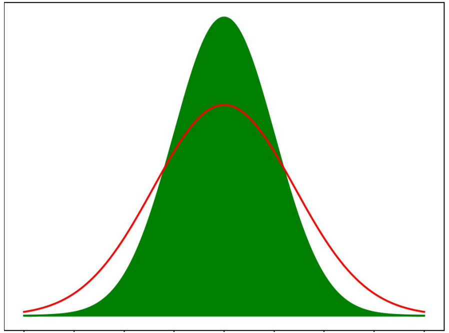
% T=1

T = 3 tail estimate

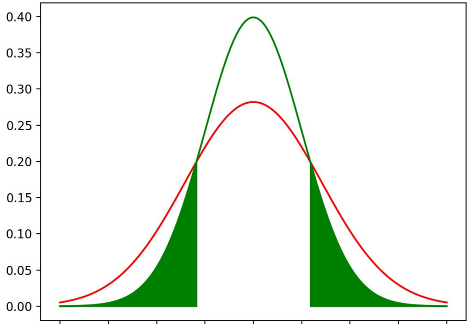
T = 1 tail estimate

T = 3 center estimate

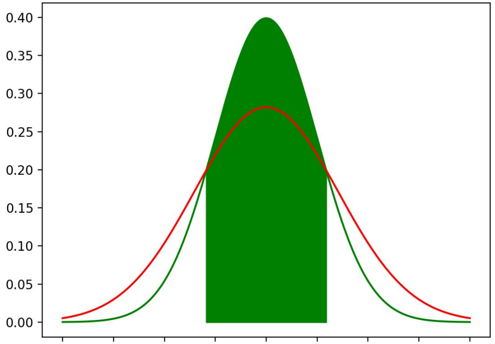
T = 1 center estimate



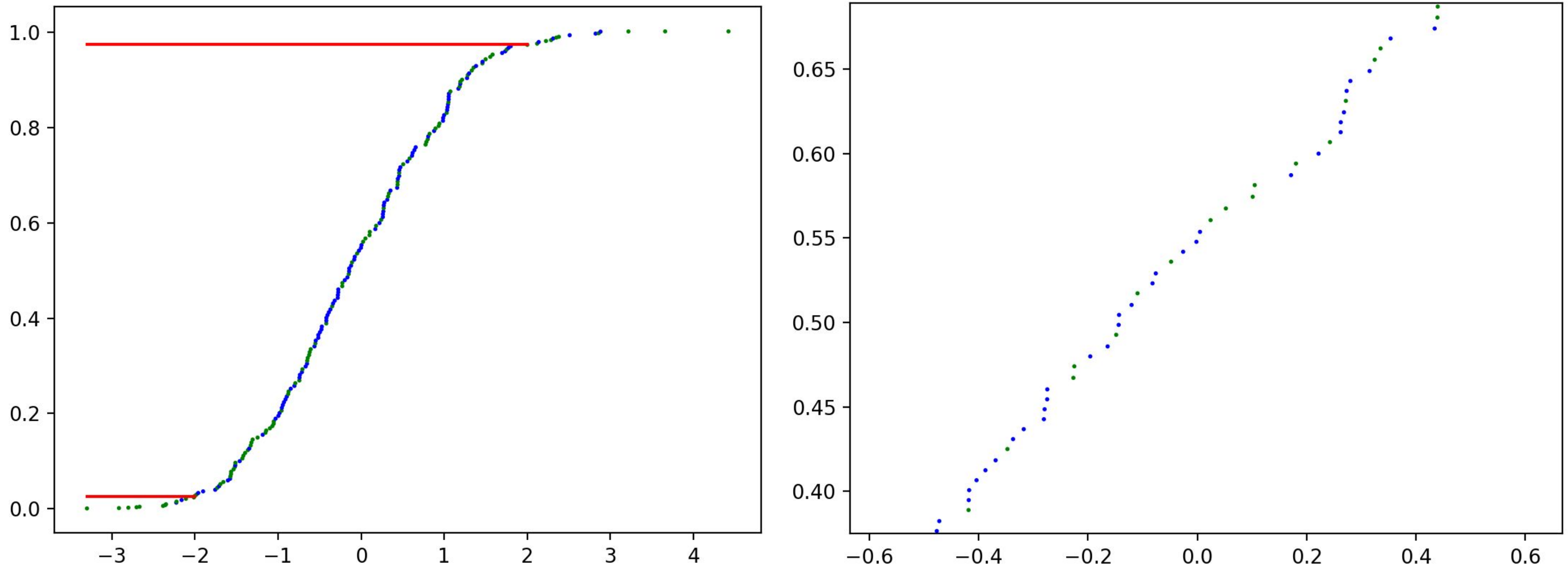
≈



+

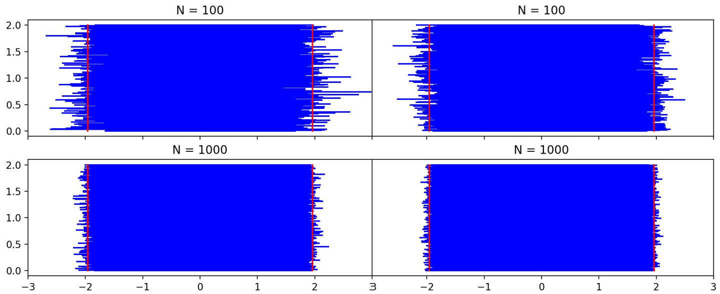


Cumulative Density Function



Results

Old New

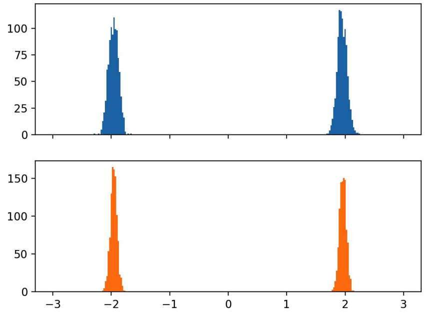


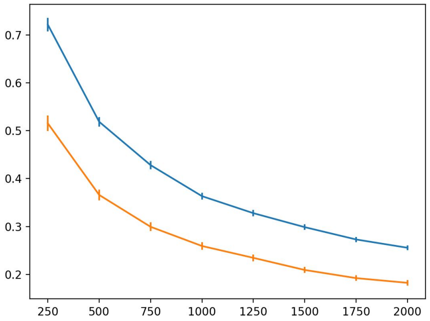
Using 50% partition

Distribution	Required to equal T = 1: 1000	Reduction
T(3)	498	50.2%
Beta(2,5)	420	58%
Normal(0,1)	500	50%

Old

New





Conclusion

- Simple, yet effective algorithm
  - Easy to implement
  - O(n)
- Around a 50% improvement

Acknowledgments

- Paul Kienzle
- NIST colleagues
- SHIP Program, NCNR, CHRNS