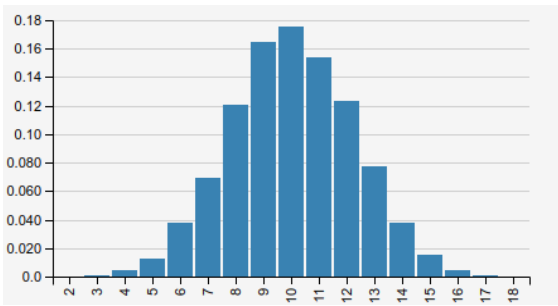
# Continualization of Probabilistic Programs With Correction

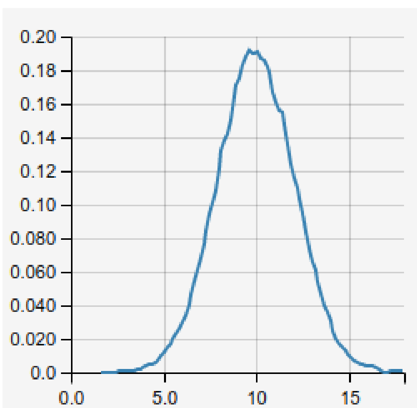
Jacob Laurel, Sasa Misailovic

University of Illinois at Urbana-Champaign

# Probabilistic Programs




```
var binomial = function(){
    return sample(Binomial({n: 20, p: 0.5}))
}
```




```
var gauss = function(){
    return sample(Gaussian({mu: 10, sigma: 2.1}))
}
```

*WebPPL probabilistic programming for the web*


HackPPL: A Universal Probabilistic Programming Language

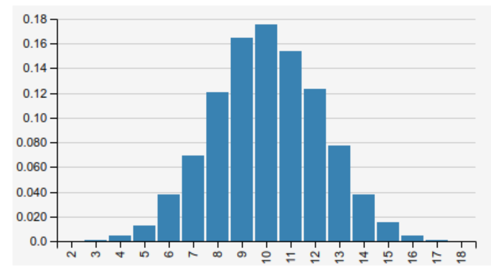
Stan

PYRO

# What Models can I write?

# Discrete Probabilistic Models

- Bayesian Learning often has ***Discrete*** structure
- Inference often ***harder***

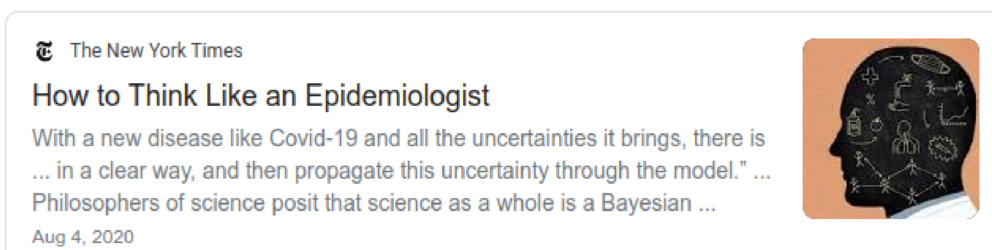




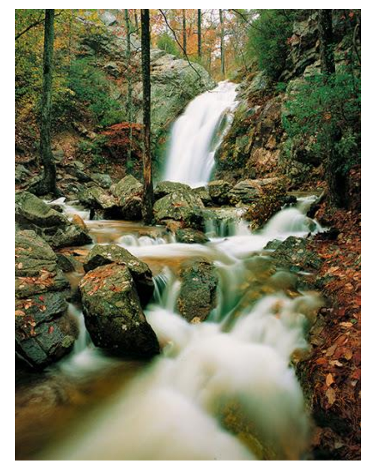
Population Models

XBOX LIVE

Ranking Systems



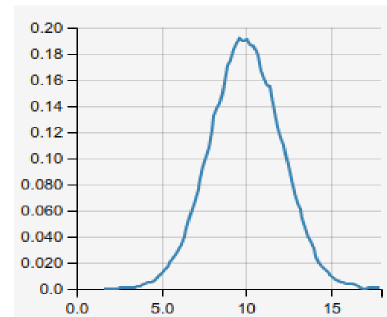

Disease Models

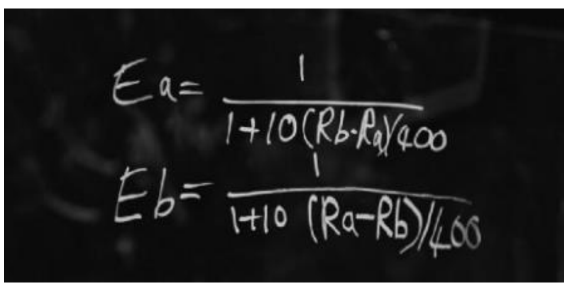


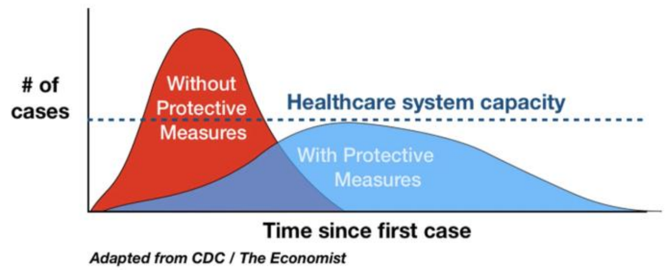
Ecology

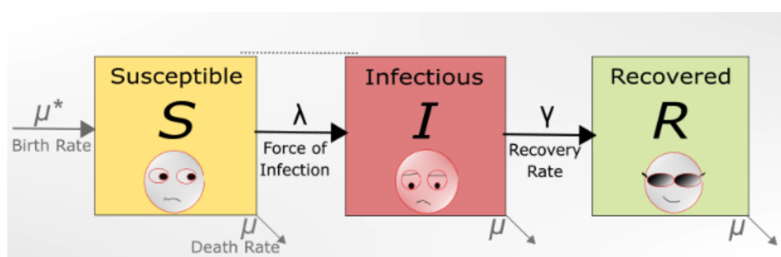
# Continuous Probabilistic Models

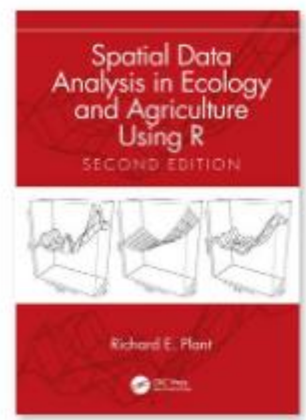
- Bayesian Learning often has ***Continuous*** structure
- Inference often ***easier***



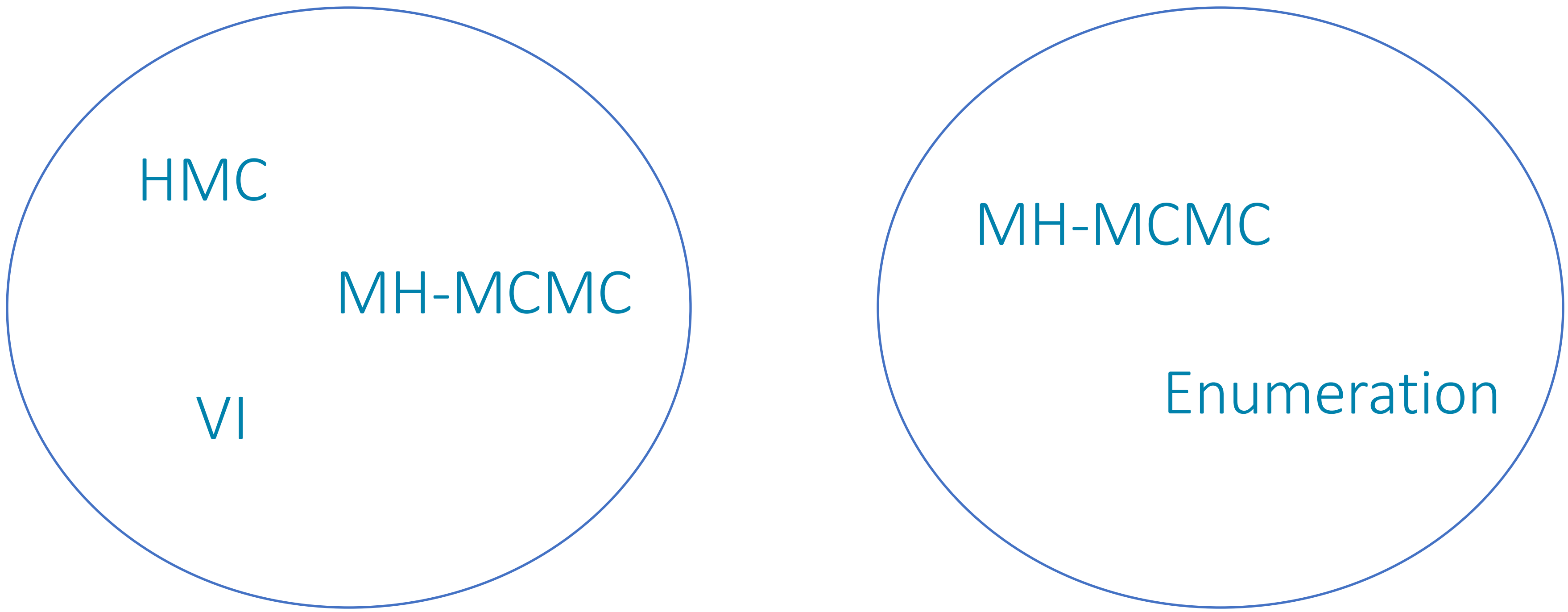



# Bayesian Inference Methods

Continuous Random Variables

Discrete Random Variables



# Bayesian Inference Methods

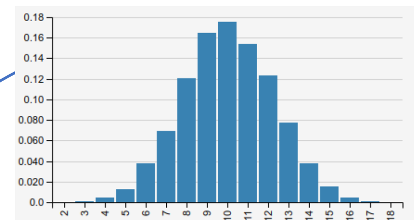
## Continuous Random Variables

HMC

MH-MCMC

VI

## Discrete Random Variables




```
Data := [12,8...]

Model {
  Prior = Uniform(20,5)
  Recruiters  = Poisson(prior)
  PerfGPA  = 4
  RegGPA = 4*Beta(7,3)
  GPA = Mix(PerfGPA,.05,RegGPA,.95)

  if (GPA == 4) {  Interviews=Bin(Recruiters,0.9)  }
  else if (GPA > 3.5) {  Interviews=Bin(Recruiters,0.6) }
  else { Interviews=Bin(Recruiters,0.5) }

  Offers=Bin(Interviews,0.4)
}

for  d in Data { factor(Offers,d) }
```

# Bayesian Inference Methods

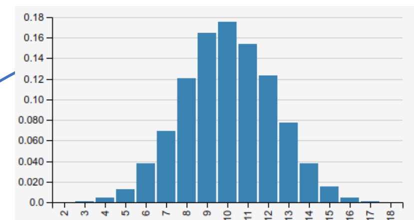
## Continuous Random Variables

HMC

MH-MCMC

VI

## Discrete Random Variables




M

```
Data := [12,8...]

Model {
  Prior = Uniform(20,5)
  Recruiters = Poisson(prior)
  PerfGPA = 4
  RegGPA = 4*Beta(7,3)
  GPA = Mix(PerfGPA,.05,RegGPA,.95)

  if (GPA == 4) { Interviews=Bin(Recruiters,0.9) }
  else if (GPA > 3.5) { Interviews=Bin(Recruiters,0.6) }
  else { Interviews=Bin(Recruiters,0.5) }

  Offers=Bin(Interviews,0.4)
}

for d in Data { factor(Offers,d) }
```

n

**Continualize!**

# Bayesian Inference Methods

## Continuous Random Variables

HMC

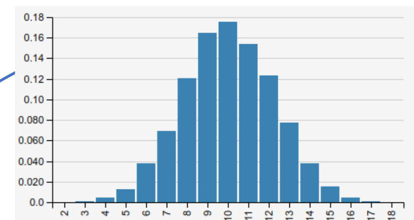
MH-MCM

VI

**Leios**

**Continualize!**

## Discrete Random Variables



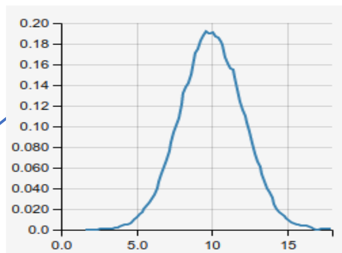

M

```
Data := [12,8...]

Model {
  Prior = Uniform(20,5)
  Recruiters = Poisson(prior)
  PerfGPA = 4
  RegGPA = 4*Beta(7,3)
  GPA = Mix(PerfGPA,.05,RegGPA,.95)

  if (GPA == 4) { Interviews=Bin(Recruiters,0.9) }
  else if (GPA > 3.5) { Interviews=Bin(Recruiters,0.6) }
  else { Interviews=Bin(Recruiters,0.5) }

  Offers=Bin(Interviews,0.4)
}

for d in Data { factor(Offers,d) }
```

n

# Bayesian Inference Methods

## Continuous Random Variables




```
Data := [12,8...]

Model {
  Prior = Uniform(20,5)
  Recruiters = Gauss(prior,sqrt(prior))
  PerfGPA = Gauss(4,0.1)
  RegGPA = 4*Beta(7,3)
  GPA = Mix(PerfGPA,.05,RegGPA,.95)

  if (3.9 < GPA <4.71) { Interviews=Gauss(Recruiters*0.9,sqrt(...)) }
  else if (GPA > 3.5) { Interviews=Gauss(Recruiters*0.6,sqrt(....)) }
  else { Interviews=Gauss(Recruiters*0.5,sqrt(...)) }

  Offers=Gauss(Interviews*0.4,sqrt(...))
}

for d in Data { factor(Offers,d) }
```

## Discrete Random Variables

MH-MCMC

Enumeration

**Leios**

**Continualize!**

# Easy right?

# Easy right?

194 Part III: From A to Binomial: Basic Probability Models

### *Making the continuity correction*

The normal approximation to the binomial is just what is says — an *approximation* — so before you move forward with your problem after you transform your X value into a z value and use the Z table (see the Appendix) to find your probability (see the previous section to find out how), you need to make an adjustment to get a close approximation. The adjustment is called a *continuity correction* — a correction you make when moving from a discrete distribution like the binomial to a continuous distribution like the normal (see Chapter 7 for more on discrete and continuous distributions). If you don't make the adjustment, your final answer will be a little larger or a little smaller than it should be.

Chapter 10

## Approximating a Binomial with a Normal Distribution

*In This Chapter*

- Using a normal distribution to approximate binomial probabilities
- Knowing when you can (and should) approximate a binomial
- Judging the sample and figuring the mean and standard deviation of the binomial
- Adding a continuity correction to the binomial

from "**Probability for Dummies**". Rumsey 2006

# Easy right?


94 Part III: From A to Binomial: Basic Probability Models

### *Making the continuity correction*

The normal approximation to the binomial is just what is says — an *approximation* — so before you move forward with your problem after you transform your X value into a z value and use the Z table (see the Appendix) to find your probability (see the previous section to find out how), you need to make an adjustment to get a close approximation. The adjustment is called a *continuity correction* — a correction you make when moving from a discrete distribution like the binomial to a continuous distribution like the normal (see Chapter 7 for more on discrete and continuous distributions). If you don't make the adjustment, your final answer will be a little larger or a little smaller than it should be.

Chapter 10

## Approximating a Binomial with a Normal Distribution

*In This Chapter*

- Using a normal distribution to approximate binomial probabilities
- Knowing when you can (and should) approximate a binomial
- Judging the sample and figuring the mean and standard deviation of the binomial
- Adding a continuity correction to the binomial

from "**Probability for Dummies**". Rumsey 2006

# Example

# Example

```
Data := [12,8...]

Model {
  Prior = Uniform(20,50)
  Recruiters = Poisson(prior)
  PerfGPA = 4
  RegGPA = 4*Beta(7,3)
  GPA = Mix(PerfGPA,.05,RegGPA,.95)

  if (GPA == 4) {  Interviews=Bin(Recruiters,0.9) }
  else if (GPA > 3.5) {  Interviews=Bin(Recruiters,0.6) }
  else { Interviews=Bin(Recruiters,0.5) }

  Offers=Bin(Interviews,0.4)
}

for  d in Data { factor(Offers,d) }
```

# Example

```
Data := [12,8...]

Model {
  Prior = Uniform(20,50)
  Recruiters = Poisson(prior)
  PerfGPA = 4
  RegGPA = 4*Beta(7,3)
  GPA = Mix(PerfGPA,.05,RegGPA,.95)

  if (GPA == 4) {  Interviews=Bin(Recruiters,0.9) }
  else if (GPA > 3.5) {  Interviews=Bin(Recruiters,0.6) }
  else { Interviews=Bin(Recruiters,0.5) }

  Offers=Bin(Interviews,0.4)
}

for  d in Data { factor(Offers,d) }
```

# Example

Data := [12,8...]

```
Model {
  Prior = Uniform(20,50)
  Recruiters = Poisson(prior)
  PerfGPA = 4
  RegGPA = 4*Beta(7,3)
  GPA = Mix(PerfGPA,.05,RegGPA,.95)

  if (GPA == 4) {  Interviews=Bin(Recruiters,0.9) }
  else if (GPA > 3.5) {  Interviews=Bin(Recruiters,0.6) }
  else { Interviews=Bin(Recruiters,0.5) }

  Offers=Bin(Interviews,0.4)
}
```

```
for  d in Data { factor(Offers,d) }
```

# Example

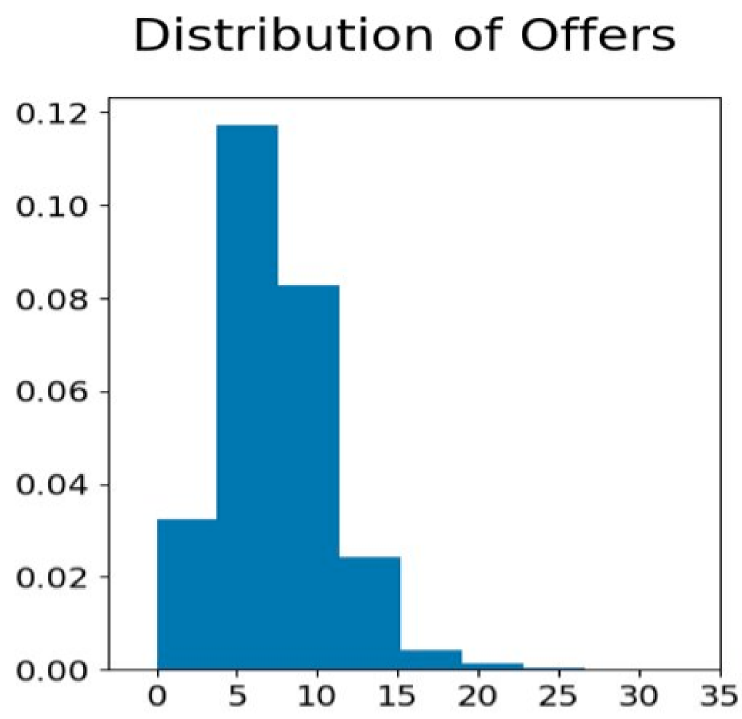
```
Data := [12,8...]

Model {
  Prior = Uniform(20,50)
  Recruiters = Poisson(prior)
  PerfGPA = 4
  RegGPA = 4*Beta(7,3)
  GPA = Mix(PerfGPA,.05,RegGPA,.95)

  if (GPA == 4) { Interviews=Bin(Recruiters,0.9) }
  else if (GPA > 3.5) { Interviews=Bin(Recruiters,0.6) }
  else { Interviews=Bin(Recruiters,0.5) }

  Offers=Bin(Interviews,0.4)
}

for d in Data { factor(Offers,d) }
```

# Example

```
Model {
  Prior = Uniform(20,50)
  Recruiters = Poisson(prior)
  PerfGPA = 4
  RegGPA = 4*Beta(7,3)
  GPA = Mix(PerfGPA,.05,RegGPA,.95)

  if (GPA == 4) {  Interviews=Bin(Recruiters,0.9) }
  else if (GPA > 3.5) {  Interviews=Bin(Recruiters,0.6) }
  else { Interviews=Bin(Recruiters,0.5) }

  Offers=Bin(Interviews,0.4)
}
```

# Example

```
Model {
  Prior = Uniform(20,50)
  Recruiters = Poisson(prior)
  PerfGPA = 4
  RegGPA = 4*Beta(7,3)
  GPA = Mix(PerfGPA,.05,RegGPA,.95)

  if (GPA == 4) {  Interviews=Bin(Recruiters,0.9) }
  else if (GPA > 3.5) {  Interviews=Bin(Recruiters,0.6) }
  else { Interviews=Bin(Recruiters,0.5) }

  Offers=Bin(Interviews,0.4)
}
```



# Example

```
Model {
  Prior = Uniform(20,50)
  Recruiters = Gauss(prior,sqrt(prior))
  PerfGPA = Gauss(4,β)
  RegGPA = 4*Beta(7,3)
  GPA = Mix(PerfGPA,.05,RegGPA,.95)

  if (GPA == 4) {  Interviews=Gauss(Recruiters*0.9,sqrt(0.09*Recruiters)) }
  else if (GPA > 3.5) {  Interviews=Gauss(Recruiters*0.6,sqrt(0.24*Recruiters))}
  else { Interviews=Gauss(Recruiters*0.5,sqrt(Recruiters*0.25))}

  Offers=Gauss(Interviews*0.4,sqrt(Interviews*0.24))
}
```

# Example

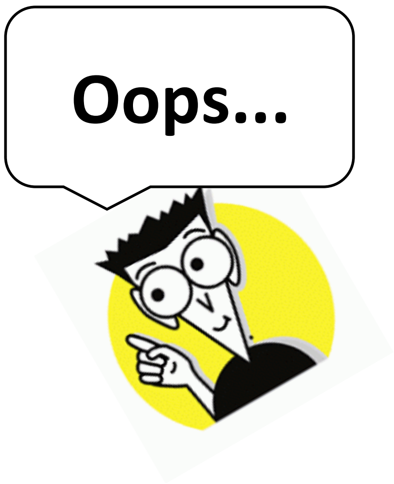
```
Model {
  Prior = Uniform(20,50)
  Recruiters = Gauss(prior,sqrt(prior))
  PerfGPA = Gauss(4,β)
  RegGPA = 4*Beta(7,3)
  GPA = Mix(PerfGPA,.05,RegGPA,.95)

  if (GPA == 4) {  Interviews=Gauss(Recruiters*0.9,sqrt(0.09*Recruiters)) }
  else if (GPA > 3.5) {  Interviews=Gauss(Recruiters*0.6,sqrt(0.24*Recruiters)) }
  else { Interviews=Gauss(Recruiters*0.5,sqrt(Recruiters*0.25)) }

  Offers=Gauss(Interviews*0.4,sqrt(Interviews*0.24))
}
```

# Example

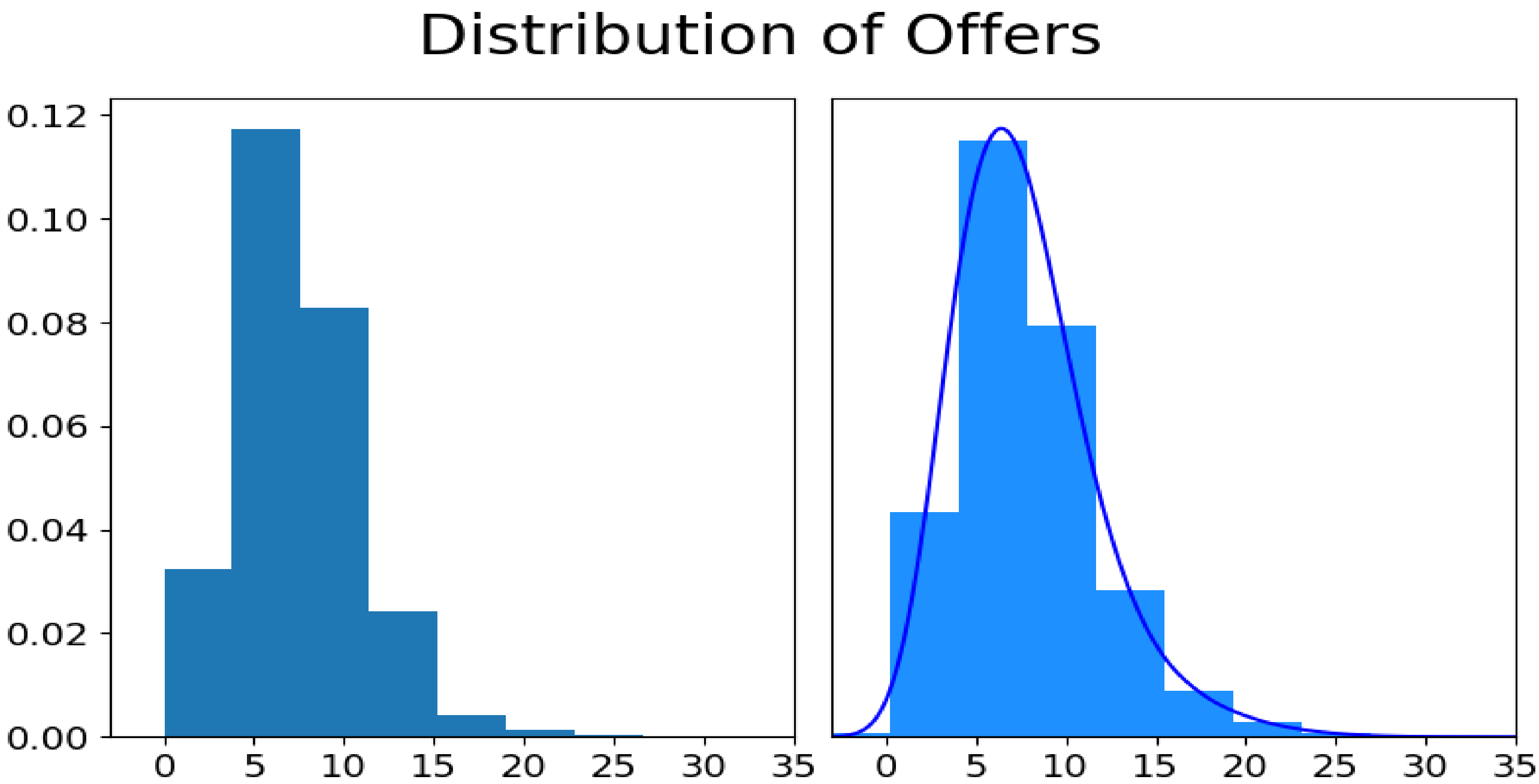
```
Model {
  Prior = Uniform(20,50)
  Recruiters = Gauss(prior,sqrt(prior))
  PerfGPA = Gauss(4,β)
  RegGPA = 4*Beta(7,3)
  GPA = Mix(PerfGPA,.05,RegGPA,.95)

  if (3.5 < GPA < 4.5) {  Interviews=Gauss(Recruiters*0.9,sqrt(0.09*Recruiters)) }
  else if (GPA > 3.5) {  Interviews=Gauss(Recruiters*0.6,sqrt(0.24*Recruiters)) }
  else { Interviews=Gauss(Recruiters*0.5,sqrt(Recruiters*0.25)) }

  Offers=Gauss(Interviews*0.4,sqrt(Interviews*0.24))
}
```

# Example

```
Model {
  Prior = Uniform(20,50)
  Recruiters = Gauss(prior,sqrt(prior))
  PerfGPA = Gauss(4,β)
  RegGPA = 4*Beta(7,3)
  GPA = Mix(PerfGPA,.05,RegGPA,.95)

  if (3.5 < GPA < 4.5) {  Interviews=Gauss(Recruiters*0.9,sqrt(0.09*Recruiters)) }
  else if (GPA > 3.5) {  Interviews=Gauss(Recruiters*0.6,sqrt(0.24*Recruiters)) }
  else { Interviews=Gauss(Recruiters*0.5,sqrt(Recruiters*0.25)) }

  Offers=Gauss(Interviews*0.4,sqrt(Interviews*0.24))
}
```



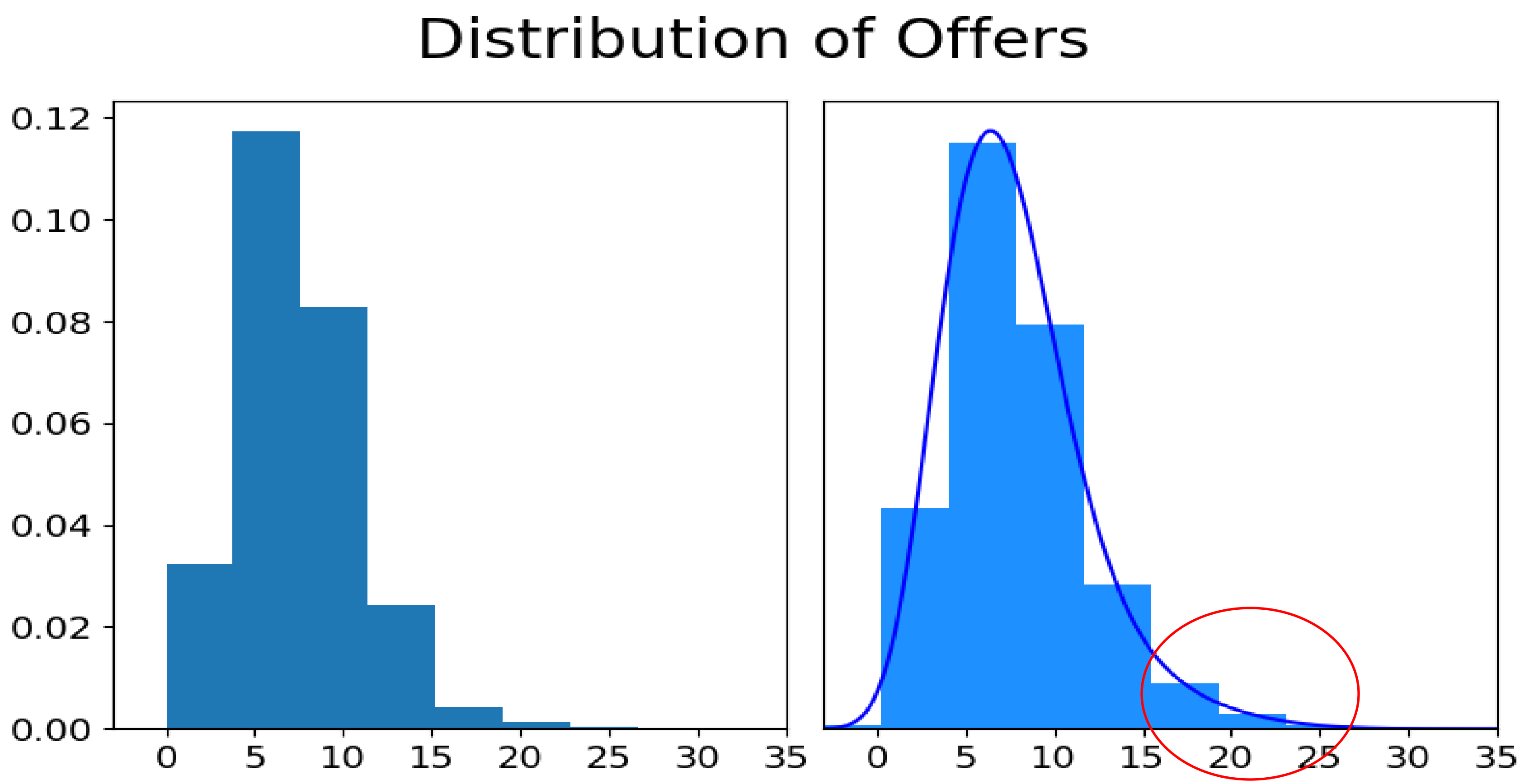
# Example




Original Model

Textbook Continualization

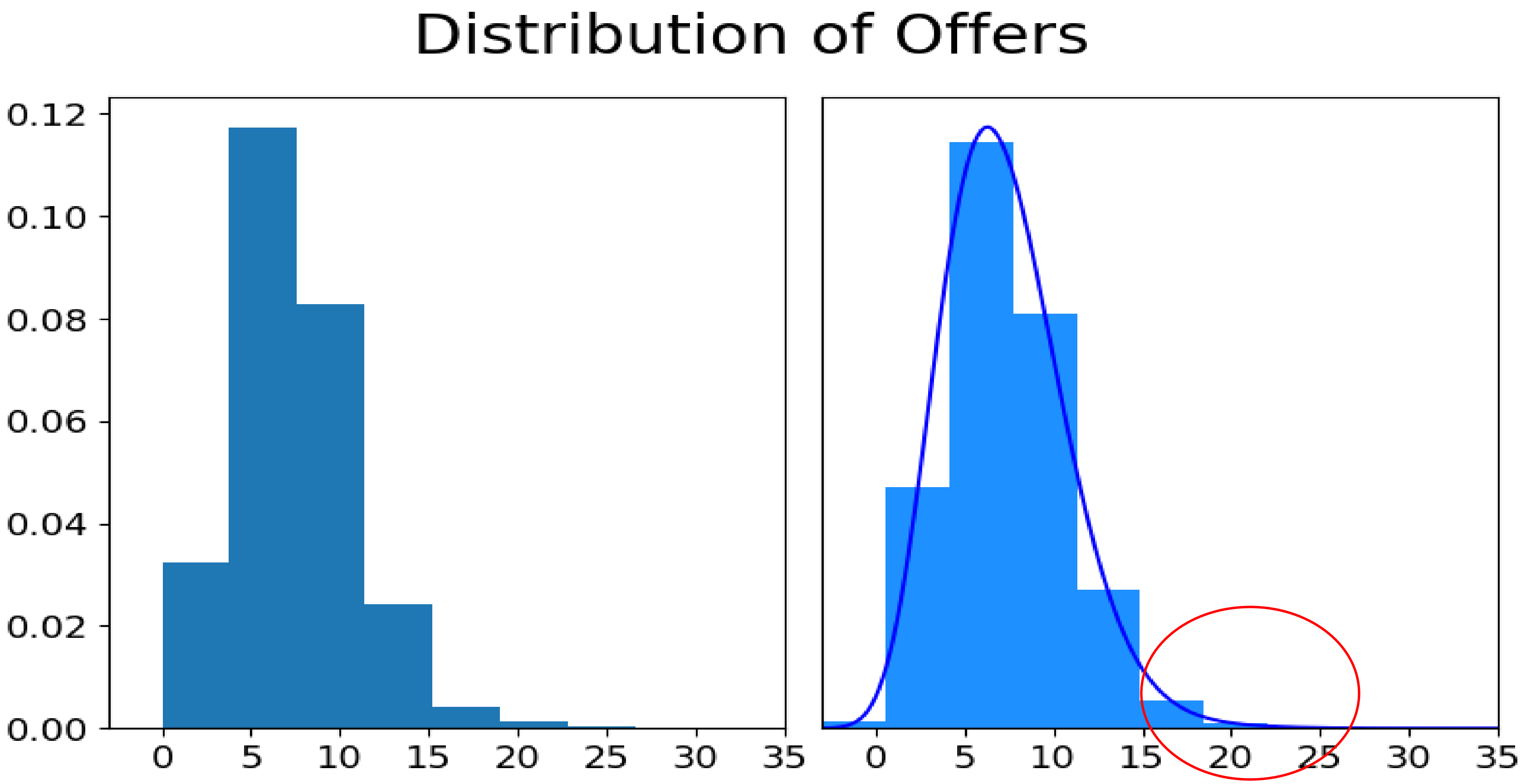
# Example




Original Model

Textbook Continualization

# Example




Original Model

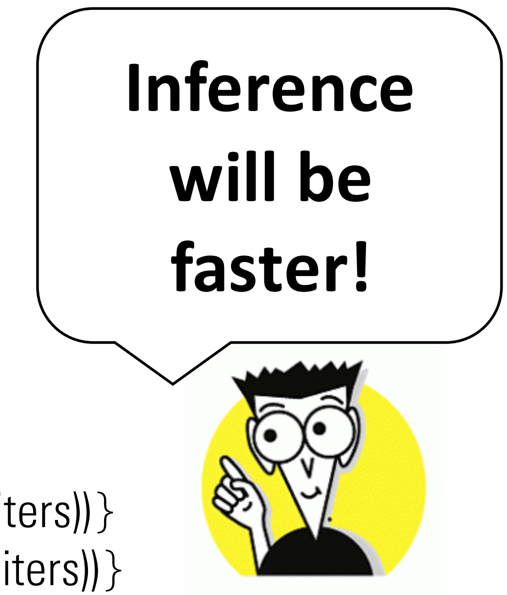
What we *really* want.

# Example

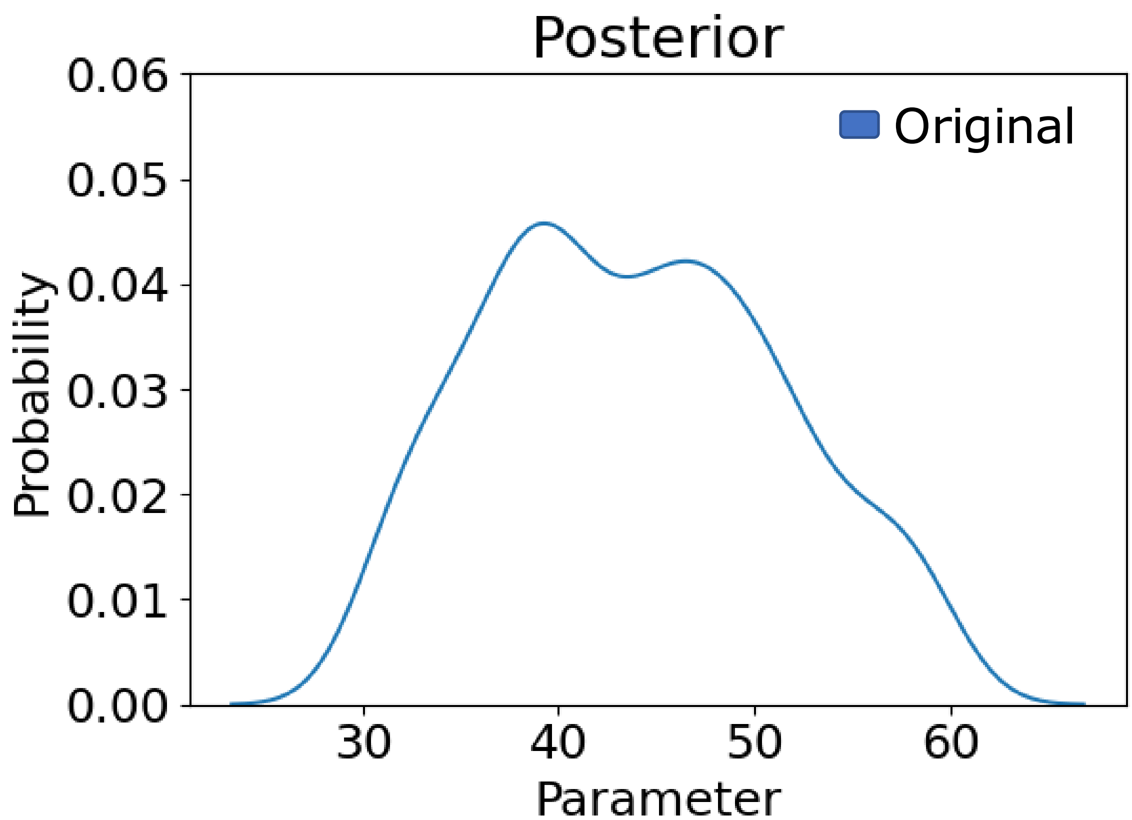
```
Model {
  Prior = Uniform(20,50)
  Recruiters = Gauss(prior,sqrt(prior))
  PerfGPA = Gauss(4,β)
  RegGPA = 4*Beta(7,3)
  GPA = Mix(PerfGPA,.05,RegGPA,.95)

  if (3.99 < GPA < 4.71) {  Interviews=Gauss(Recruiters*0.9,sqrt(0.09*Recruiters))}
  else if (GPA > 3.5001) {  Interviews=Gauss(Recruiters*0.6,sqrt(0.24*Recruiters))}
  else { Interviews=Gauss(Recruiters*0.5,sqrt(Recruiters*0.25))}

  Offers=Gauss(Interviews*0.4,sqrt(Interviews*0.24))
}
```

# Example

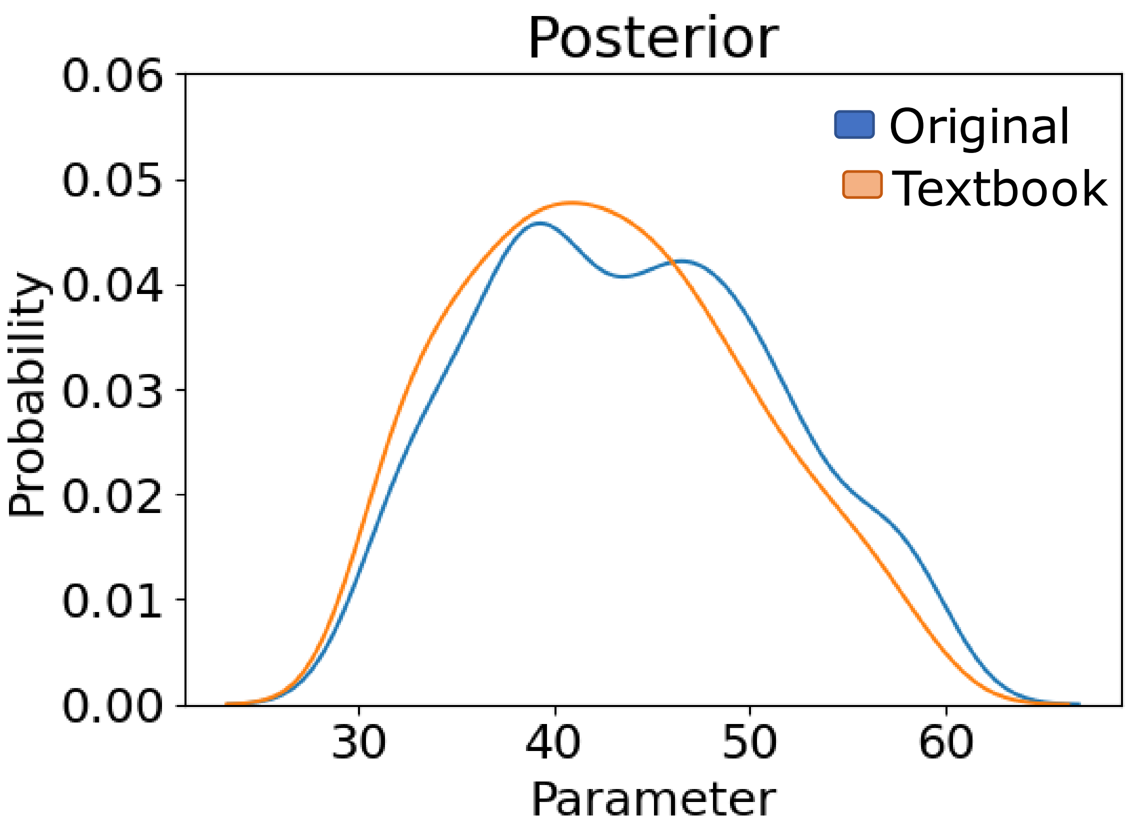
```
Data := [12, 8...]

Model {
  Prior = Uniform(20,50)
  Recruiters = Gauss(prior,sqrt(prior))
  PerfGPA = Gauss(4,β)
  RegGPA = 4*Beta(7,3)
  GPA = Mix(PerfGPA,.05,RegGPA,.95)

  if (3.99 < GPA < 4.71) { Interviews=Gauss(Recruiters*0.9,sqrt(0.09*Recruiters))}
  else if (GPA > 3.5001) { Interviews=Gauss(Recruiters*0.6,sqrt(0.24*Recruiters))}
  else { Interviews=Gauss(Recruiters*0.5,sqrt(Recruiters*0.25))}

  Offers=Gauss(Interviews*0.4,sqrt(Interviews*0.24))
}

for d in Data { factor(Offers,d)}
```



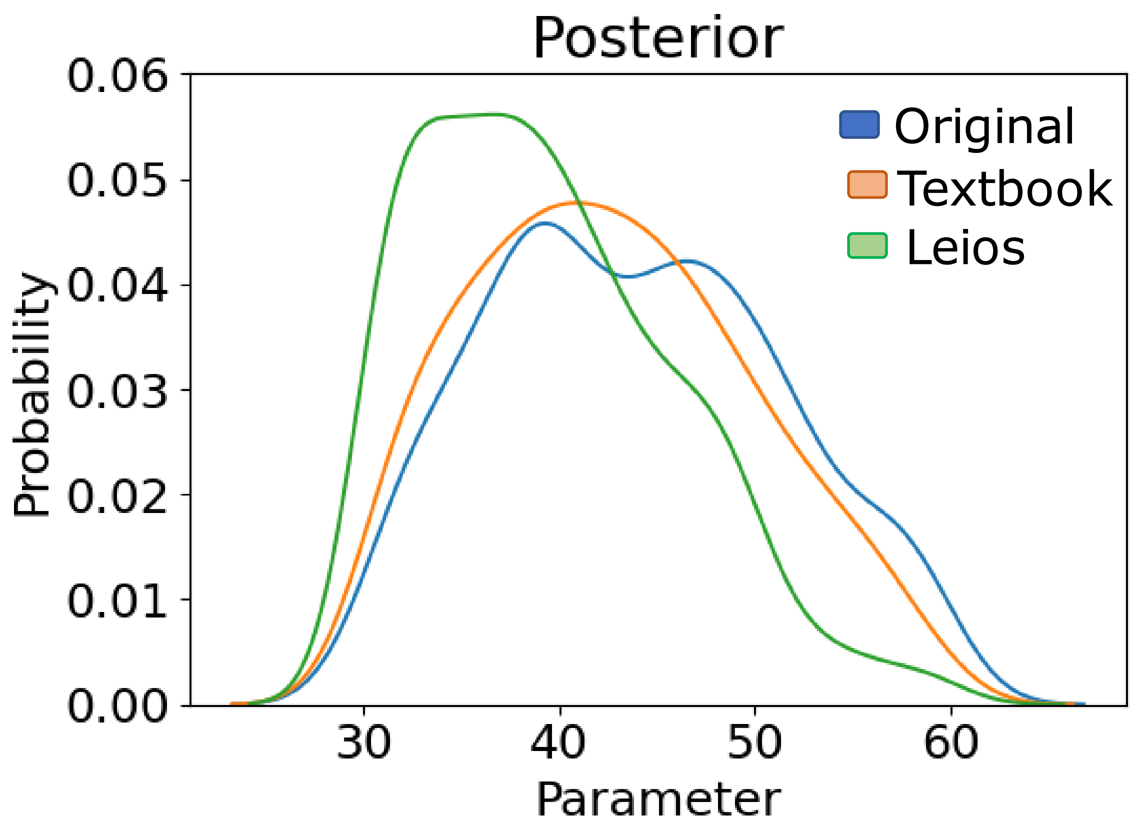
# Example - Results



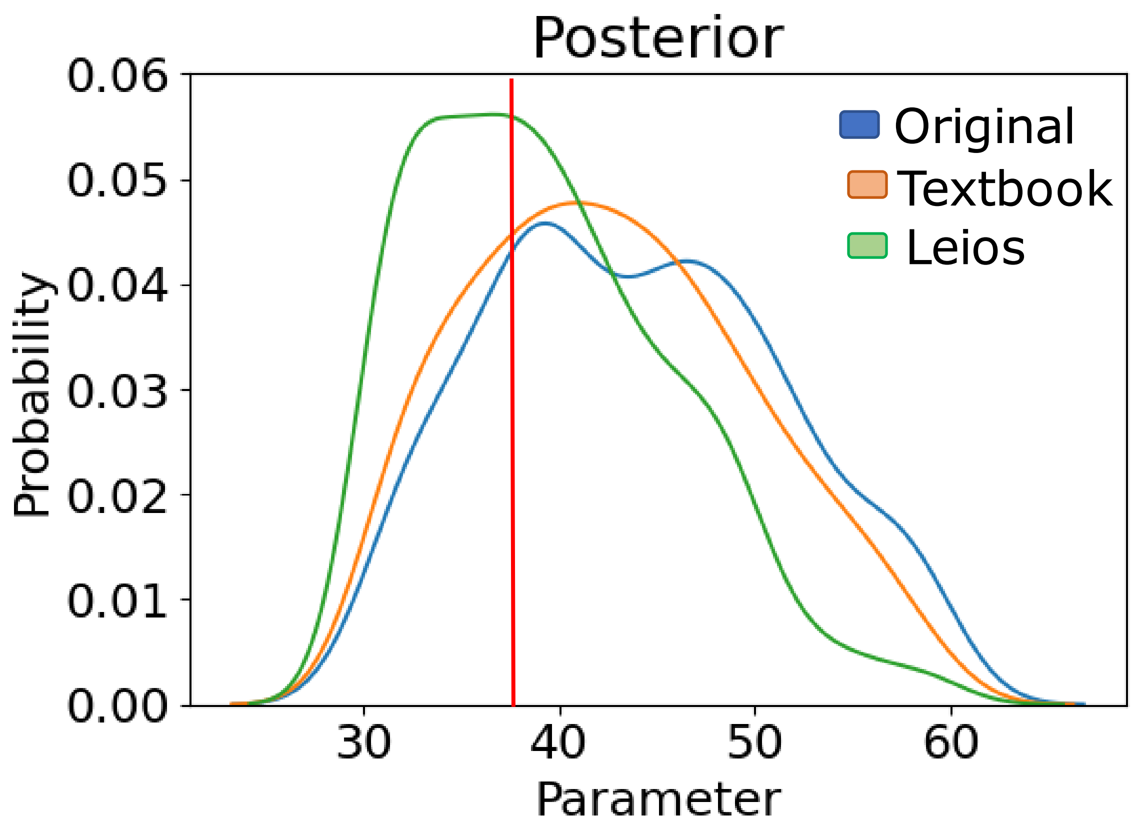
# Example - Results



# Example - Results

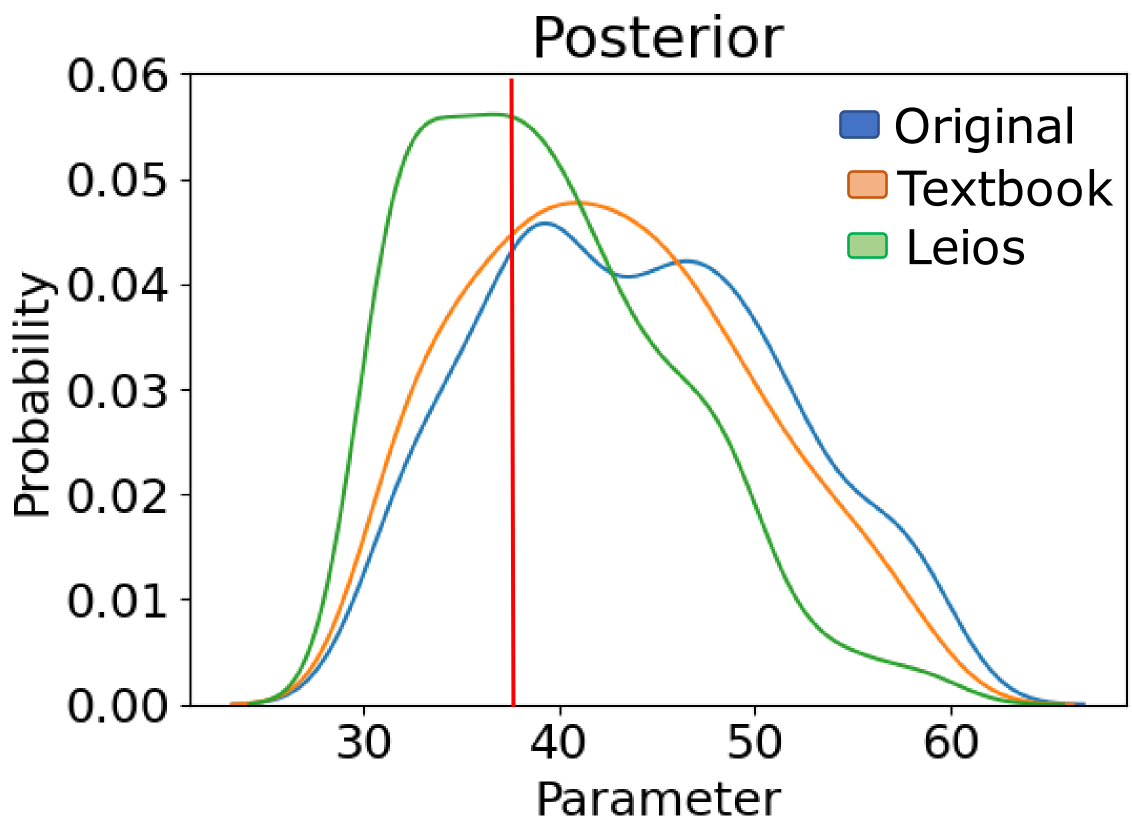


# Example - Results




Ground Truth Value: 37

# Example - Results




Ground Truth Value: 37 *and* we're 33% faster!

# How to get there?

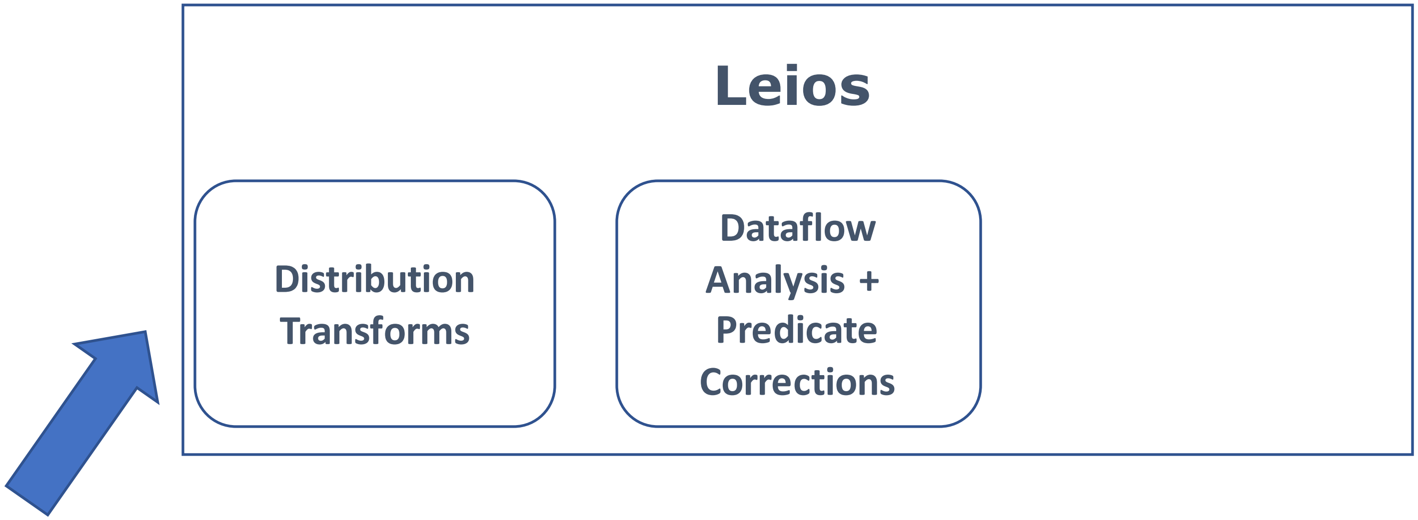
How to get there?

Use a **smarter** program analysis.

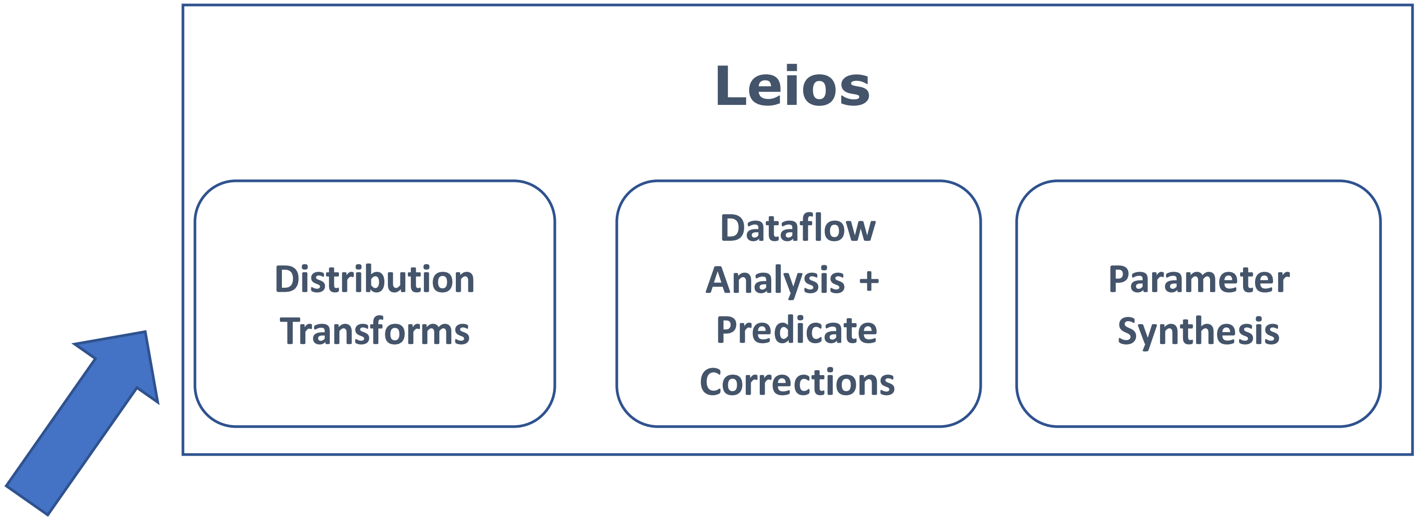
## Leios


```
Data := [12,8...]

Model {
  Prior = Uniform(20,5)
  Recruiters = Poisson(prior)
  PerfGPA = 4
  RegGPA = 4*Beta(7,3)
  GPA = Mix(PerfGPA,.05,RegGPA,.95)

  if (GPA == 4) {  Interviews=Bin(Recruiters,0.9) }
  else if (GPA > 3.5) {  Interviews=Bin(Recruiters,0.6) }
  else { Interviews=Bin(Recruiters,0.5) }

  Offers=Bin(Interviews,0.4)
}

for d in Data { factor(Offers,d) }
```

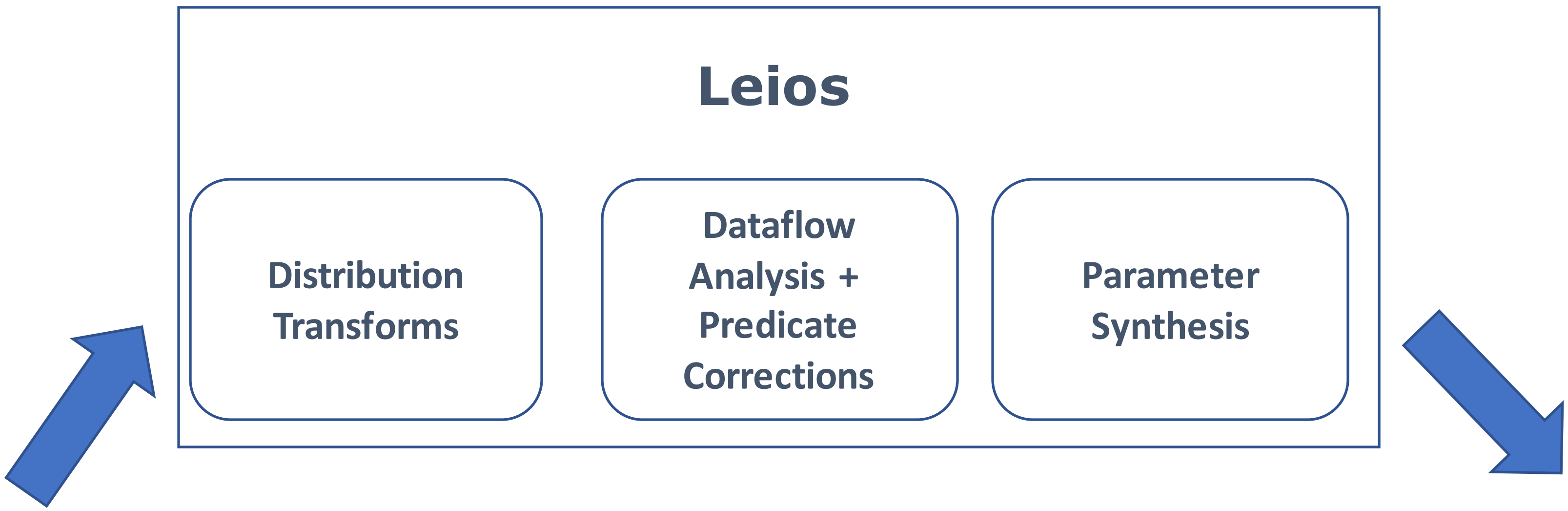

```
Data := [12,8...]

Model {
  Prior = Uniform(20,5)
  Recruiters = Poisson(prior)
  PerfGPA = 4
  RegGPA = 4*Beta(7,3)
  GPA = Mix(PerfGPA,.05,RegGPA,.95)

  if (GPA == 4) {  Interviews=Bin(Recruiters,0.9) }
  else if (GPA > 3.5) {  Interviews=Bin(Recruiters,0.6) }
  else { Interviews=Bin(Recruiters,0.5) }

  Offers=Bin(Interviews,0.4)
}

for d in Data { factor(Offers,d) }
```




```
Data := [12,8...]

Model {
  Prior = Uniform(20,5)
  Recruiters = Poisson(prior)
  PerfGPA = 4
  RegGPA = 4*Beta(7,3)
  GPA = Mix(PerfGPA,.05,RegGPA,.95)

  if (GPA == 4) {  Interviews=Bin(Recruiters,0.9) }
  else if (GPA > 3.5) {  Interviews=Bin(Recruiters,0.6) }
  else { Interviews=Bin(Recruiters,0.5) }

  Offers=Bin(Interviews,0.4)
}

for d in Data { factor(Offers,d) }
```




```
Data := [12,8...]

Model {
  Prior = Uniform(20,5)
  Recruiters = Poisson(prior)
  PerfGPA = 4
  RegGPA = 4*Beta(7,3)
  GPA = Mix(PerfGPA,.05,RegGPA,.95)

  if (GPA == 4) { Interviews=Bin(Recruiters,0.9) }
  else if (GPA > 3.5) { Interviews=Bin(Recruiters,0.6) }
  else { Interviews=Bin(Recruiters,0.5) }

  Offers=Bin(Interviews,0.4)
}

for d in Data { factor(Offers,d) }
```




```
Data := [12,8...]

Model {
  Prior = Uniform(20,5)
  Recruiters = Poisson(prior)
  PerfGPA = 4
  RegGPA = 4*Beta(7,3)
  GPA = Mix(PerfGPA,.05,RegGPA,.95)

  if (GPA == 4) { Interviews=Bin(Recruiters,0.9) }
  else if (GPA > 3.5) { Interviews=Bin(Recruiters,0.6) }
  else { Interviews=Bin(Recruiters,0.5) }

  Offers=Bin(Interviews,0.4)
}

for d in Data { factor(Offers,d) }
```

```
Data := [12,8...]

Model {
  Prior = Uniform(20,5)
  Recruiters = Gauss(prior,sqrt(prior))
  PerfGPA = Gauss(4,0.1)
  RegGPA = 4*Beta(7,3)
  GPA = Mix(PerfGPA,.05,RegGPA,.95)

  if (3.9 < GPA < 4.71) { Interviews=Gauss(Recruiters*0.9,sqrt(...)) }
  else if (GPA > 3.501) { Interviews=Gauss(Recruiters*0.6,sqrt(....)) }
  else { Interviews=Gauss(Recruiters*0.5,sqrt(...)) }

  Offers=Gauss(Interviews*0.4,sqrt(...))
}

for d in Data { factor(Offers,d) }
```

# Language Syntax

*Program* ::= *DataBlock? ; Model* { *Stmt* } *; ObserveBlock? ; return Var;*

*Stmt* ::= *skip* | *abort* | *Var := Expr* | *Var := Dist* | *CONST Var := Expr*
| *Stmt ; Stmt* | { *Stmt* } | *condition* ( *BExpr* ) | *while* ( *BExpr* ) *Stmt*
| *if* ( *BExpr* ) *Stmt else Stmt* | *for i = INT to INT Stmt*

*Expr* ::= *Expr ArithOp Expr* | *f* ( *Expr* ) | *REAL* | *INT* | *Var*

*BExpr* ::= *BExpr or BExpr* | *BExpr and BExpr* | *not BExpr* | *Expr Relop Expr*

*DataBlock* ::= [(*INT*)*] | [(*REAL*)*]

*ObserveBlock* ::= *for D in Data* { *factor* ( *Var* , *D* ) ; }

*Dist* ::= *Gaussian* | *Uniform* | *Binomial* | *Poisson* | *Bernoulli* | ...

*ArithOp* ∈ {+, -, *, /, **, ...} , *f* ∈ {*log, abs, exp, ...*} , *Relop* ∈ {<, ==, <=, ...}

# Language Syntax

*Program* ::= *DataBlock?* ; *Model* { *Stmt* } ; *ObserveBlock?* ; *return Var;*

*Stmt* ::= *skip* | *abort* | *Var* := *Expr* | *Var* := *Dist* | *CONST Var* := *Expr*
| *Stmt* ; *Stmt* | { *Stmt* } | *condition* ( *BExpr* ) | *while* ( *BExpr* ) *Stmt*
| *if* ( *BExpr* ) *Stmt else Stmt* | *for i* = *INT to INT* *Stmt*

*Expr* ::= *Expr ArithOp Expr* | *f* ( *Expr* ) | *REAL* | *INT* | *Var*

*BExpr* ::= *BExpr or BExpr* | *BExpr and BExpr* | *not BExpr* | *Expr Relop Expr*

*DataBlock* ::= [(*INT*)$^*$] | [(*REAL*)$^*$]

*ObserveBlock* ::= *for D in Data* { *factor* ( *Var* , *D* ) ; }

*Dist* ::= *Gaussian* | *Uniform* | *Binomial* | *Poisson* | *Bernoulli* | ...

*ArithOp* ∈ {+, -, *, /, **, ... } , *f* ∈ {*log, abs, exp, ...* } , *Relop* ∈ {<, ==, <=, ... }

# Step 1) Distribution Transforms

# Distribution Transforms

- Continuous relaxations for each latent

# Distribution Transforms

- Continuous relaxations for each latent

$$Binomial(n,p) \longrightarrow Gaussian(np, \sqrt{np(1-p)})$$

$$Binomial(n,p) \longrightarrow Gamma(n,p)$$

$$Poisson(\lambda) \longrightarrow Gaussian(\lambda, \sqrt{\lambda})$$

$$Poisson(\lambda) \longrightarrow Gamma(\lambda, 1)$$

$$DiscUniform(a,b) \longrightarrow Uniform(a,b)$$

$$C \longrightarrow Gaussian(C, \beta)$$

# Distribution Transforms

- Continuous relaxations for each latent

$$np \geq 30 \quad Binomial(n,p) \longrightarrow Gaussian(np, \sqrt{np(1-p)})$$

$$np < 30 \quad Binomial(n,p) \longrightarrow Gamma(n,p)$$

$$\lambda \geq 10 \quad Poisson(\lambda) \longrightarrow Gaussian(\lambda, \sqrt{\lambda})$$

$$\lambda < 10 \quad Poisson(\lambda) \longrightarrow Gamma(\lambda, 1)$$

$$DiscUniform(a,b) \longrightarrow Uniform(a,b)$$

$$C \longrightarrow Gaussian(C, \beta)$$

# Smoothing Likelihood

# Smoothing Likelihood

- Add Gaussian(0,β) to **smooth** observed value

# Smoothing Likelihood

- Add Gaussian(0,β) to **smooth** observed value

Why?

# Smoothing Likelihood

- Add Gaussian(0,β) to **smooth** observed value

Why?

Likelihood sums over *N* observed data points

# Smoothing Likelihood (Example)

# Smoothing Likelihood (Example)

```
Data = […...]
X = DiscUniform(10,50)
Y = Binomial(X,0.5)

for i in range(N)
   factor(Data[i],Y)
```

# Smoothing Likelihood (Example)

```
Data = [.....]
X = DiscUniform(10,50)
Y = Binomial(X,0.5)

for i in range(N)
   factor(Data[i],Y)
```

Log-likelihood sum

$$\ln\left(P(X = x_0)\right) + \sum_{i=1}^{10} \ln\left(P(Y = data[i])\right)$$

# Smoothing Likelihood (Example)

```
Data = [.....]
X = DiscUniform(10,50)
Y = Binomial(X,0.5)

for i in range(N)
   factor(Data[i],Y)
```

Log-likelihood sum

$$\ln\left(P(X = x_0)\right) + \sum_{i=1}^{10} \ln\left(P(Y = data[i])\right)$$

$$\mathbf{ln}\ \left(\mathbf{1}_{x_0 \in [10,50]} \cdot \frac{1}{40}\right) + \sum_{i=1}^{10} \mathbf{ln}\ \left(\binom{x_0}{data[i]} \mathbf{0.5}^{data[i]} \cdot \mathbf{0.5}^{x_0 - data[i]}\right)$$

# Smoothing Likelihood (Example)

```
Data = [.....]
X = DiscUniform(10,50)
Y = Binomial(X,0.5)

for i in range(N)
   factor(Data[i],Y)
```

Log-likelihood sum

$$\ln\left(P(X=x_0)\right)+\sum_{i=1}^{10}\ln\left(P(Y=data[i])\right)$$

$$\ln\left(\mathbf{1}_{x_0\in[10,50]}\cdot\frac{1}{40}\right)+\sum_{i=1}^{10}\ln\left(\binom{x_0}{data[i]}0.5^{data[i]}\cdot 0.5^{x_0-data[i]}\right)$$

#Data * *O*(C(n,k))

# Smoothing Likelihood (Example)

```
Data = [.....]
X = DiscUniform(10,50)
Y = Binomial(X,0.5)

for i in range(N)
    factor(Data[i],Y)
```

```
Data = [.....]
X = DiscUniform(10,50)
Y = Binomial(X,0.5)
Z = Gaussian(Y,β)

for i in range(N)
    factor(Data[i],Z)
```

Log-likelihood sum

$$\ln\left(P(X=x_0)\right)+\sum_{i=1}^{10}\ln\left(P(Y=data[i])\right)$$

$$\ln\left(1_{x_0\in[10,50]}\cdot\frac{1}{40}\right)+\sum_{i=1}^{10}\ln\left(\binom{x_0}{data[i]}0.5^{data[i]}\cdot 0.5^{x_0-data[i]}\right)$$

# Smoothing Likelihood (Example)

```
Data = [.....]
X = DiscUniform(10,50)
Y = Binomial(X,0.5)

for i in range(N)
   factor(Data[i],Y)
```

```
Data = [.....]
X = DiscUniform(10,50)
Y = Binomial(X,0.5)
Z = Gaussian(Y,β)

for i in range(N)
   factor(Data[i],Z)
```

Log-likelihood sum

$$\ln\left(P(X = x_0)\right) + \sum_{i=1}^{10} \ln\left(P(Y = data[i])\right)$$

$$\ln\left(\mathbf{1}_{x_0\in[10,50]} \cdot \frac{1}{40}\right) + \sum_{i=1}^{10} \ln\left(\binom{x_0}{data[i]} 0.5^{data[i]} \cdot 0.5^{x_0 - data[i]}\right)$$

# Smoothing Likelihood (Example)

```
Data = [.....]
X = DiscUniform(10,50)
Y = Binomial(X,0.5)

for i in range(N)
   factor(Data[i],Y)
```

```
Data = [.....]
X = DiscUniform(10,50)
Y = Binomial(X,0.5)
Z = Gaussian(Y,β)

for i in range(N)
   factor(Data[i],Z)
```

New Log-likelihood sum

$$\ln\left(P(X = x_0)\right) + \ln\left(P(Y = y_0)\right) + \sum_{i=1}^{10} \ln\left(P(Z = data[i])\right)$$

# Smoothing Likelihood (Example)

```
Data = [.….]
X = DiscUniform(10,50)
Y = Binomial(X,0.5)

for i in range(N)
   factor(Data[i],Y)
```

```
Data = [.….]
X = DiscUniform(10,50)
Y = Binomial(X,0.5)
Z = Gaussian(Y,β)

for i in range(N)
   factor(Data[i],Z)
```

New Log-likelihood sum

$$\ln\left(P(X=x_0)\right)+\ln\left(P(Y=y_0)\right)+\sum_{i=1}^{10}\ln\left(P(Z=data[i])\right)$$

$$\ln\left(1_{x_0\in[10,50]}\cdot\frac{1}{40}\right)+\ln\left(\binom{x_0}{y_0}0.5^{y_0}\cdot 0.5^{x_0-y_0}\right)+\sum_{i=1}^{10}\left(-\frac{(data[i]-y_0)^2}{\beta}\right)+\ln\left(\frac{1}{\sqrt{2\pi\beta}}\right)$$

# Smoothing Likelihood (Example)

```
Data = [.....]
X = DiscUniform(10,50)
Y = Binomial(X,0.5)

for i in range(N)
   factor(Data[i],Y)
```

```
Data = [.....]
X = DiscUniform(10,50)
Y = Binomial(X,0.5)
Z = Gaussian(Y,β)

for i in range(N)
   factor(Data[i],Z)
```

New Log-likelihood sum

$$\ln\left(P(X = x_0)\right) + \ln\left(P(Y = y_0)\right) + \sum_{i=1}^{10} \ln\left(P(Z = data[i])\right)$$

$$\ln\left(1_{x_0 \in [10,50]} \cdot \frac{1}{40}\right) + \ln\left(\binom{x_0}{y_0} 0.5^{y_0} \cdot 0.5^{x_0 - y_0}\right) + \sum_{i=1}^{10} \left(-\frac{(data[i] - y_0)^2}{\beta}\right) + \ln\left(\frac{1}{\sqrt{2\pi\beta}}\right)$$

single $O$(C(n,k))

Much easier!

# Smoothing Likelihood (Example)

```
Data = [.….]
X = DiscUniform(10,50)
Y = Binomial(X,0.5)

for i in range(N)
   factor(Data[i],Y)
```

Full continualization:

```
Data = [.….]
X = Uniform(10,50)
Y = Gaussian(0.5*X,sqrt(X*0.25))
Z = Gaussian(Y,β)

for i in range(N)
   factor(Data[i],Z)
```

New Log-likelihood sum

$$\ln\left(P(X=x_0)\right)+\ln\left(P(Y=y_0)\right)+\sum_{i=1}^{10}\ln\left(P(Z=data[i])\right)$$

$$\ln\left(1_{x_0\in[10,50]}\cdot\frac{1}{40}\right)+\ln\left(\binom{x_0}{y_0}0.5^{y_0}\cdot0.5^{x_0-y_0}\right)+\sum_{i=1}^{10}\left(-\frac{(data[i]-y_0)^2}{\beta}\right)+\ln\left(\frac{1}{\sqrt{2\pi\beta}}\right)$$

**single** $O$(C(n,k)) — Much easier!

# Smoothing Likelihood (Example)

```
Data = [.....]
X = DiscUniform(10,50)
Y = Binomial(X,0.5)

for i in range(N)
   factor(Data[i],Y)
```

Full continualization:

```
Data = [.....]
X = Uniform(10,50)
Y = Gaussian(0.5*X,sqrt(X*0.25))
Z = Gaussian(Y,β)

for i in range(N)
   factor(Data[i],Z)
```

New Log-likelihood sum

$$\ln\left(P(X=x_0)\right)+\ln\left(P(Y=y_0)\right)+\sum_{i=1}^{10}\ln\left(P(Z=data[i])\right)$$

$$\ln\left(1_{x_0\in[10,50]}\cdot\frac{1}{40}\right)+\left(\frac{y_0-0.5x_0}{\sqrt{0.25\,x_0}}\right)^2+\ln\left(\frac{1}{\sqrt{2\pi\cdot 0.25x_0}}\right)+\sum_{i=1}^{10}\left(-\frac{(data[i]-y_0)^2}{\beta}\right)+\ln\left(\frac{1}{\sqrt{2\pi\beta}}\right)$$

Even easier.

Much easier!

# What about Program Control Flow?

# Predicate Correction

```
if (GPA == 4)
    Interviews = Bin(Recruiters,0.9)
else if  (GPA > 3.5)
    Interviews = Bin(Recruiters,0.6)
else
    Interviews = Bin(Recruiters,0.5)

Offers = Bin(Interviews, 0.4)
```

# Predicate Correction

```
if (GPA == 4)
    Interviews = Bin(Recruiters,0.9)
else if  (GPA > 3.5)
    Interviews = Bin(Recruiters,0.6)
else
    Interviews = Bin(Recruiters,0.5)

Offers = Bin(Interviews, 0.4)
```

```
if (4 -θ1 < GPA < 4 + θ2)
    Interviews = Gauss(Recruiters*0.9,. . . )
else if  (GPA > 3.5 + θ3)
    Interviews = Gauss(Recruiters*0.6,. . . )
else
    Interviews = Gauss(Recruiters*0.5,. . . )

Offers = Bin(Interviews, 0.4)
```

# Dataflow Analysis

- Do we have to change every predicate?

# Dataflow Analysis

- Do we have to change every predicate?

**No!**

# Dataflow Analysis

- Do we have to change every predicate?

**No!**

Only ones (transitively) affected by the approximations

# Step 3) Parameter Synthesis

# Parameter Synthesis

- Minimize Wasserstein distance to *original* program

- Only done *once* per model (cost amortized)

# Parameter Synthesis

```
if (4 -θ1 < GPA < 4 + θ2)
    Interviews = Gauss(Recruiters*0.9, . . . )
else if  (GPA > 3.5 + θ3)
    Interviews = Gauss(Recruiters*0.6, . . . )
else
    Interviews = Gauss(Recruiters*0.5, . . . )

Offers = Bin(Interviews, 0.4)
```

# Parameter Synthesis

$$\arg\min_{\theta_1,\theta_2,\theta_3}(Wasserstein\ Dist(P_{Orig}, P_{Cont.}))$$

```
if (4 -θ1 < GPA < 4 + θ2)
    Interviews = Gauss(Recruiters*0.9, . . . )
else if  (GPA > 3.5 + θ3)
    Interviews = Gauss(Recruiters*0.6, . . . )
else
    Interviews = Gauss(Recruiters*0.5, . . . )

Offers = Bin(Interviews, 0.4)
```

# Parameter Synthesis

$$\arg\min_{\theta_1,\theta_2,\theta_3}(Wasserstein\ Dist(P_{Orig}, P_{Cont.}))$$

Original Program

if (4 - $\theta_1$ < GPA < 4 + $\theta_2$)
    Interviews = Gauss(Recruiters*0.9, . . . )
else if (GPA > 3.5 + $\theta_3$)
    Interviews = Gauss(Recruiters*0.6, . . . )
else
    Interviews = Gauss(Recruiters*0.5, . . . )

Offers = Bin(Interviews, 0.4)

# Parameter Synthesis

$$\arg\min_{\theta_1, \theta_2, \theta_3} (Wasserstein\ Dist(P_{Orig}, P_{Cont.}))$$

Continualized Program

if (4 - $\theta_1$ < GPA < 4 + $\theta_2$)
    Interviews = Gauss(Recruiters*0.9, . . . )
else if (GPA > 3.5 + $\theta_3$)
    Interviews = Gauss(Recruiters*0.6, . . . )
else
    Interviews = Gauss(Recruiters*0.5, . . . )

Offers = Bin(Interviews, 0.4)

# Parameter Synthesis

$\arg\min_{\theta_1,\theta_2,\theta_3}(Wasserstein\ Dist(P_{Orig}, P_{Cont.}))$

How to optimize?

```
if (4 -θ1 < GPA < 4 + θ2)
    Interviews = Gauss(Recruiters*0.9, . . . )
else if  (GPA > 3.5 + θ3)
    Interviews = Gauss(Recruiters*0.6, . . . )
else
    Interviews = Gauss(Recruiters*0.5, . . . )

Offers = Bin(Interviews, 0.4)
```

# Parameter Synthesis - Optimization

- Parameterize $P_{Cont}$ with a fixed parameter

- Forward sample model (ignore observed data)

- Measure empirical Wasserstein Distance to $P_{Orig}$ samples:

$$X_i \sim P_{orig}, Y_i \sim P_{cont}$$

$$EWD(P_{orig}, P_{cont}) = \sum_{i=1}^{n} ||X_i - Y_i||$$

# Parameter Synthesis - Optimization

- Use Nelder-Mead to explore different parameters

- Allows us to uncover runtime errors  causing program to abort (e.g. negative variance)

# Isn't this as hard as Inference?

# Isn't this as hard as Inference?

**No!**

Isn't this as hard as Inference?

**No!**

Cost is **amortized.**

# Measure-Theoretic Semantics

- Program **state**: $\sigma \in \mathbb{R}^n$
- Sub-probability **measures**: $\mu : \mathcal{B}(\mathbb{R}^n) \to [0, 1]$
- Program **transforms** sub-probability measures:

$$[\![Program]\!] : \mu \to \mu$$

- Distributions interpreted as **Kernels**

$$[\![Dist]\!] : \sigma \to \mu$$

# Measure-Theoretic Semantics

- Absolute Continuity: A sub-probability measure μ is absolutely continuous w.r.t to the Lebesgue measure **λ,** iff

$$\forall S \in \mathcal{B}(\mathbb{R}^n) \ : \ \lambda(S) = 0 \implies \mu(S) = 0$$

# Measure-Theoretic Semantics

Sampling: $[\![ x_i = \text{Dist}(e_1, \ldots, e_k) ]\!](\mu)$

$$= \lambda S.\, \int_{\mathbb{R}^n} \mu(d\sigma) \cdot \delta_{x_1} \circ \ldots \circ [\![ \text{Dist}(e_1, \ldots, e_k) ]\!](\sigma) \circ \delta_{x_{i+1}} \ldots (S)$$

Where $x_i$'s marginal is:

$$\mu_{x_i} = \lambda S.\, \int_{\sigma \in \mathbb{R}^n} \mu(d\sigma) [\![ \text{Dist}(e_1, \ldots, e_k) ]\!](\sigma)$$

# Measure-Theoretic Semantics

$[\![ x_i = \text{Gauss(a,b)} ]\!](\mu)$ Example:

$$\mu_{x_i} = \lambda S.\int_{\sigma \in \mathbb{R}^n} \mu(d\sigma) \int_{x_i \in \mathbb{R} \cap S} \frac{1}{\sqrt{2\pi b(\sigma)}} e^{-\left(\frac{x_i - a(\sigma)}{b(\sigma)}\right)^2} dx_i$$

$[\![ x_i = \text{Binomial(n,p)} ]\!](\mu)$ Example:

$$\mu_{x_i} = \lambda S.\int_{\sigma \in \mathbb{R}^n} \mu(d\sigma) \sum_{k=1}^{n(\sigma)} \binom{n(\sigma)}{k} p(\sigma)^k (1 - p(\sigma))^{n(\sigma)-k} \delta_k(S)$$

# Measure-Theoretic Semantics

$[\![ x_i = \text{Gauss(a,b)} ]\!](\mu)$ Example:

$$\mu_{x_i} = \lambda S. \int_{\sigma \in \mathbb{R}^n} \mu(d\sigma) \int_{x_i \in \mathbb{R} \cap S} \frac{1}{\sqrt{2\pi b(\sigma)}} e^{-\left(\frac{x_i - a(\sigma)}{b(\sigma)}\right)^2} dx_i$$

$[\![ x_i = \text{Binomial(n,p)} ]\!](\mu)$ Example:

$$\mu_{x_i} = \lambda S. \int_{\sigma \in \mathbb{R}^n} \mu(d\sigma) \sum_{k=1}^{n(\sigma)} \binom{n(\sigma)}{k} p(\sigma)^k (1 - p(\sigma))^{n(\sigma)-k} \boxed{\delta_k(S)}$$

# Measure-Theoretic Semantics

Sequencing:

$$[\![P_1 ; P_2]\!](\mu) = [\![P_2]\!]([\![P_1]\!](\mu))$$

if-then-else:

$$\mu_B = \lambda S.\, \mu(S \cap B)$$

$$\mu_{\neg B} = \lambda S.\, \mu(S \cap \neg B)$$

$$[\![\text{if (B) } P_1 \text{ else } P_2]\!](\mu) = [\![\, P_1 \,]\!](\mu_B) + [\![\, P_2]\!](\mu_{\neg B})$$

# Measure-Theoretic Semantics

Factor:

$$[\![\text{factor}(x_i\ ,t)]\!](\mu) = \lambda S.\int_{\mathbb{R}^n} \mathbf{1}_S \cdot g(t,\sigma)\cdot \mu(ds)$$

Where $g(t,\sigma)$ is a **smooth** function

# Theoretical Implications

**Theorem 1**: In the transformed program the marginal sub-probability measure of each latent is absolutely continuous at each point the variable is defined

# Evaluation - Benchmarks

| Program | Prior | Likelihood | Correction? | $T_{cont}$ (s) |
|---|---|---|---|---|
| GPA | | | | |
| Election | | | | |
| Fairness | | | | |
| SVM Fairness | | | | |
| TrueSkill | | | | |
| Disease | | | | |
| SVE | | | | |
| Beta Binomial | | | | |
| Exam | | | | |
| Plankton | | | | |

# Evaluation - Benchmarks

| Program | Prior | Likelihood | Correction? | $T_{cont}$ (s) |
|---|---|---|---|---|
| GPA | Uniform | | | |
| Election | Disc Uniform | | | |
| Fairness | Disc Uniform | | | |
| SVM Fairness | Binomial | | | |
| TrueSkill | Poisson | | | |
| Disease | Disc Uniform | | | |
| SVE | Uniform | | | |
| Beta Binomial | Beta | | | |
| Exam | Uniform | | | |
| Plankton | Disc Uniform | | | |

# Evaluation - Benchmarks

| Program | Prior | Likelihood | Correction? | $T_{cont}$ (s) |
|---|---|---|---|---|
| GPA | | Discrete | | |
| Election | | Bernoulli | | |
| Fairness | | Bernoulli | | |
| SVM Fairness | | Continuous | | |
| TrueSkill | | Bernoulli | | |
| Disease | | Discrete | | |
| SVE | | Hybrid | | |
| Beta Binomial | | Discrete | | |
| Exam | | Discrete | | |
| Plankton | | Discrete | | |

# Evaluation - Benchmarks

| Program | Prior | Likelihood | Correction? | $T_{cont}$ (s) |
|---|---|---|---|---|
| GPA | | | y | |
| Election | | | y | |
| Fairness | | | y | |
| SVM Fairness | | | y | |
| TrueSkill | | | y | |
| Disease | | | n | |
| SVE | | | n | |
| Beta Binomial | | | n | |
| Exam | | | n | |
| Plankton | | | n | |

# Evaluation - Benchmarks

| Program | Prior | Likelihood | Correction? | $T_{cont}$ (s) |
|---|---|---|---|---|
| GPA | | | | 3.6 |
| Election | | | | 1.1 |
| Fairness | | | | 1.8 |
| SVM Fairness | | | | 1.6 |
| TrueSkill | | | | 1.1 |
| Disease | | | | 0.006 |
| SVE | | | | 0.009 |
| Beta Binomial | | | | 0.006 |
| Exam | | | | 0.008 |
| Plankton | | | | 0.006 |

# Evaluation - Methodology

- Fix a true parameter and generate data
- Place flat prior over parameter
- Infer/recover true value given generated data
- Measure how close: $\left| \frac{\text{True Val.} - \text{Inferred Val.}}{\text{True Val.}} \right|$
  - Leios
  - Original Model + Likelihood Smoothing (Naïve)
  - Original Model

# Evaluation - Methodology

- Fix a true parameter and generate data
- Place flat prior over parameter
- Infer/recover true value given generated data
- Measure how close: $\left| \frac{\text{True Val. - Inferred Val.}}{\text{True Val.}} \right|$
    - Leios
    - Original Model + Likelihood Smoothing (Naïve)
    - Original Model

Improvement due to:
**Continuous Approximations**

# Evaluation - Methodology

- Fix a true parameter and generate data
- Place flat prior over parameter
- Infer/recover true value given generated data
- Measure how close: $\left| \frac{\text{True Val.} - \text{Inferred Val.}}{\text{True Val.}} \right|$
  - Leios
  - Original Model + Likelihood Smoothing (Naïve)
  - Original Model

Improvement due to:
**Likelihood Smoothing**

# MCMC (β=0.1)

| **Program** | $T_{Orig}$ | $E_{Orig}$ | $T_{Naive}$ | $E_{Naive}$ | $T_{Leios}$ | $E_{Leios}$ |
|---|---|---|---|---|---|---|
| GPA | 0.806 | 0.090 | 0.631 | 0.070 | 0.605 | 0.058 |
| Election | x | x | 3.232 | 0.051 | 0.616 | 0.036 |
| Fairness | 4.396 | 0.057 | 0.563 | 0.056 | 0.603 | 0.093 |
| SVM Fairness | x | x | 0.626 | 0.454 | 0.980 | 0.261 |
| TrueSkill | 3.668 | 0.009 | 0.494 | 0.059 | 0.586 | 0.053 |
| Disease | 4.944 | 0.009 | 1.350 | 0.013 | 0.490 | 0.008 |
| SVE | x | x | 0.522 | 0.045 | 0.516 | 0.091 |
| Beta Binomial | 1.224 | 0.028 | 0.564 | 0.024 | 0.459 | 0.013 |
| Exam | 3.973 | 0.087 | 0.504 | 0.126 | 0.527 | 0.133 |
| Plankton | 0.570 | 0.017 | 0.457 | 0.080 | 0.453 | 0.042 |

# Variational Inference (β=0.1)

| Program | $T_{Orig}$ | $E_{Orig}$ | $T_{Naive}$ | $E_{Naive}$ | $T_{Leios}$ | $E_{Leios}$ |
|---|---|---|---|---|---|---|
| GPA | x | x | x | x | 3.11 | 0.20 |
| Election | x | x | x | x | 1.76 | 0.07 |
| Fairness | x | x | x | x | 1.81 | 0.72 |
| SVM Fairness | x | x | x | x | 1.80 | 0.20 |
| TrueSkill | x | x | x | x | 1.81 | 0.12 |
| Disease | x | x | x | x | 1.73 | 0.24 |
| SVE | 0.677 | 0.684 | 1.478 | 3.095 | 1.47 | 0.58 |
| Beta Binomial | x | x | x | x | 1.60 | 0.83 |
| Exam | x | x | x | x | 0.60 | 0.22 |
| Plankton | x | x | x | x | 3.43 | 0.29 |

# Leios Takeaways


Leios makes Discrete Inference *much* easier


Approximation error is small price to pay!