Case Studies in Bayesian Data Science

1: Building a Non-Parametric Prior

David Draper

Department of Applied Mathematics and Statistics
University of California, Santa Cruz

Short Course (Day 5)
University of Reading (UK)

Building a Nonparametric Prior

Part 2 recap: Suppose in the future I'll observe **real-valued** $y = (y_1, \ldots, y_n)$ and I have **no covariate information**, so that my uncertainty about the y_i is **exchangeable**.

Then if I'm willing to regard y as part of an **infinitely exchangeable sequence** (which is like **thinking** of the y_i as having been **randomly sampled** from the **population** $(y_1, y_2, \ldots)$), then to be **coherent** my joint predictive distribution $p(y_1, \ldots, y_n)$ must have the **hierarchical** form

$$\begin{array}{rcl} F & \sim & p(F) \\ (y_i|F) & \stackrel{\text{IID}}{\sim} & F, \end{array} \qquad (1)$$

where F is the **limiting empirical cumulative distribution function** (CDF) of the infinite sequence $(y_1, y_2, \ldots)$.

How do I **construct** such a prior on F in a **meaningful** way?

Two main approaches have so far been fully developed: **Dirichlet processes** and **Pólya trees**.

Case study (introducing the **Dirichlet process**): **Fixing the broken bootstrap** (joint work with a former Bath MSc student, Callum McKail).

$\boxed{\text{Goal:}}$ **Nonparametric interval estimates** of the **variance** (or **standard deviation** (SD)).

One (**frequentist nonparametric**) approach: the **bootstrap**.

Best **bootstrap** technology at present for **nonparametric interval estimates** is BC_a method (e.g., Efron and Tibshirani, 1993) or the its computationally less intensive cousin, the ABC method; we work here with ABC (roughly **same performance**, **much faster**).

Bootstrap

(We also tried **iterated bootstrap** (e.g., Lee and Young, 1995) on problem below but it performed worse than BC_a and ABC.)

Bootstrap **propaganda**:

“One of the principal goals of bootstrap theory is to produce good confidence intervals automatically. “Good means that the bootstrap intervals should closely match exact confidence intervals in those special situations where statistical theory yields an exact answer, and should give dependably accurate coverage probabilities in all situations. ... **[The BC_a intervals] come close to [these] criteria of goodness**” (Efron and Tibshirani, 1993).

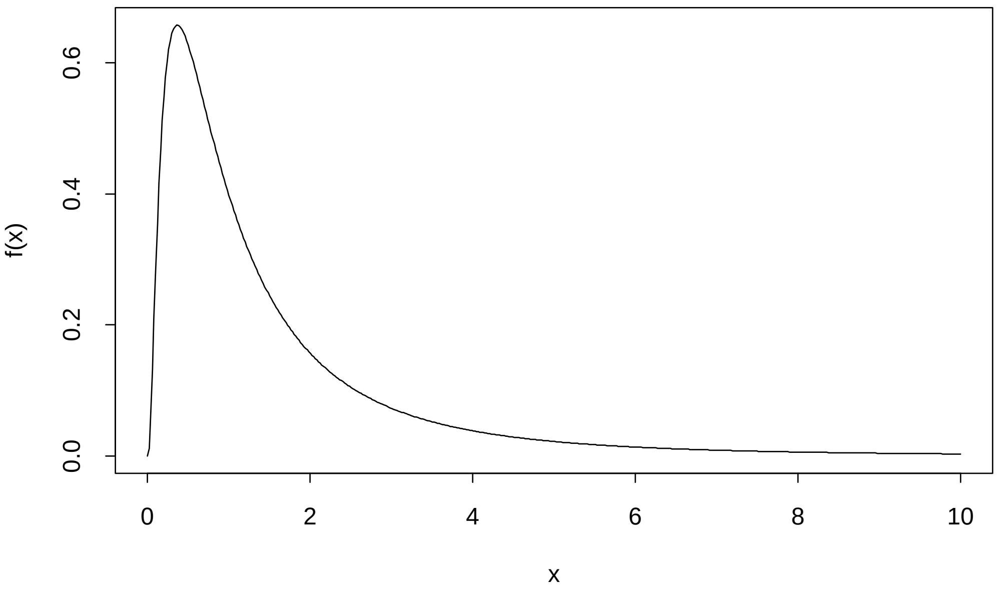


Figure 1. *Standard lognormal distribution* $LN(0,1)$, *i.e.,* $Y \sim LN(0,1) \iff \ln(Y) \sim N(0,1)$.

Lognormal distribution provides good test of bootstrap: it is highly skewed and heavy-tailed.

Bootstrap (continued)

Consider **sample** $y = (y_1, \ldots, y_n)$ from model

$$\begin{aligned} F &= LN(0,1) \\ (Y_1, \ldots, Y_n | F) &\overset{\text{IID}}{\sim} F, \end{aligned} \qquad (2)$$

and suppose **functional** of F of interest is

$$V(F) = \int [y - E(F)]^2 \, dF(y), \quad \text{where} \quad E(F) = \int y \, dF(y). \quad (3)$$

Usual **unbiased sample variance**

$$s^2 = \frac{1}{n-1} \sum_{i=1}^{n} (y_i - \bar{y})^2, \quad \bar{y} = \sum_{i=1}^{n} y_i, \qquad (4)$$

is (almost) **nonparametric MLE** of $V(F)$, and it serves as basis of ABC intervals; **population value** for $V(F)$ with $LN(0,1)$ is $e(e-1) = 4.67$.

n	**mean**	**median**	% $<$ 4.67	90th percentile
10	4.66 (0.49)	1.88 (0.09)	77.1 (1.2)	9.43 (0.75)
20	4.68 (0.34)	2.52 (0.09)	74.0 (1.4)	9.37 (0.60)
50	4.68 (0.21)	3.20 (0.08)	70.4 (1.5)	8.59 (0.43)
100	4.67 (0.15)	3.62 (0.08)	67.6 (1.5)	7.98 (0.31)
500	4.68 (0.07)	4.23 (0.05)	62.6 (1.4)	6.64 (0.13)

Table 1. *Distribution of sample variance for $LN(0,1)$ data, based on 1000 simulation repetitions (simulation SE in parentheses).*

s^2 achieves unbiasedness by being **too small most of the time and much too large some of the time**; does not bode well for bootstrap.

Bootstrap Calibration Failure

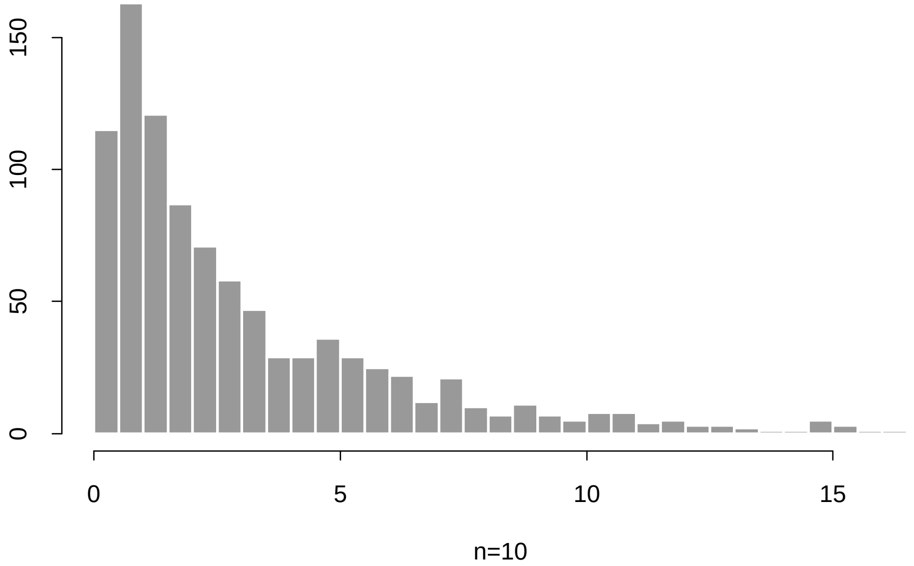


Figure 2. *Histogram of first 950 ordered values of the sample variance for* $n = 10$.

n	actual cov. (%)	mean length	median length	% on left	% on right
10	36.0 (1.5)	8.08 (0.68)	2.43 (1.13)	61.1 (1.5)	2.9 (0.5)
20	49.4 (1.6)	9.42 (0.77)	3.79 (1.32)	48.6 (1.6)	2.0 (0.4)
50	61.9 (1.5)	10.1 (0.61)	4.56 (0.75)	35.4 (1.5)	2.7 (0.5)
100	68.6 (1.5)	8.76 (0.49)	4.63 (0.72)	27.4 (1.4)	4.0 (0.6)
500	76.8 (1.3)	5.43 (0.21)	3.51 (0.27)	18.5 (1.2)	4.7 (0.7)

Table 2. *ABC, nominal 90% intervals,* $LN(0, 1)$ *data*

With $n = 10$, **nominal 90%** intervals only cover $\boxed{\textbf{36\%}}$ of the time, and even with $n = 500$ actual coverage is only up to **77%!**

Mistakes are almost always from interval lying entirely to **left** of true $V(F)$.

Bootstrap Failure (continued)

Bootstrap fails because it is based solely on empirical CDF $\widehat{F}_n$, which is **ignorant of right tail behavior beyond** $Y_{(n)} = \max_i Y_i$.

To improve **must bring in prior information about tail weight and skewness**.

Problem is of course not unsolvable **parametrically**: consider model

$$\begin{aligned} (\mu, \sigma^2) & \sim p(\mu, \sigma^2) \\ (Y_1, \ldots, Y_n | \mu, \sigma^2) & \overset{\text{IID}}{\sim} LN(\mu, \sigma^2), \end{aligned} \qquad (5)$$

and take proper but highly diffuse prior on (μ, σ^2).

Easy to use Gibbs sampling (even in `BUGS`) to show that Bayesian intervals are **well-calibrated** (but note interval lengths!):

	actual	**mean**	**median**	**% on**	**% on**
n	**cov. (%)**	**length**	**length**	**left**	**right**
10	88.7 (1.0)	$6 \cdot 10^5$ $(2 \cdot 10^5)$	194.6 (29.3)	4.9 (0.7)	6.4 (0.8)
20	89.3 (1.0)	145.2 (14.9)	39.6 (2.0)	4.9 (0.7)	5.8 (0.7)
50	89.1 (1.0)	17.8 (0.5)	12.6 (0.5)	5.3 (0.7)	5.6 (0.7)
100	90.6 (0.9)	8.9 (0.2)	7.6 (0.2)	4.0 (0.6)	5.4 (0.7)
500	89.9 (1.0)	3.0 (0.02)	3.0 (0.02)	5.5 (0.7)	4.6 (0.7)

Table 3. *Lognormal model,* $N(0, 10^4)$ *prior for* μ, $\Gamma(0.001, 0.001)$ *prior for* $\tau = \frac{1}{\sigma^2}$, *nominal 90%,* $LN(0, 1)$ *data*

Parametric Bayes Fails

But **parametric Bayesian inference** based on LN distribution is **horribly non-robust**:

n	**actual cov. (%)**	**mean length**	**% on left**	**% on right**
10	0.0 (0.0)	6.851 (0.03)	0.0 (0.0)	100.0 (0.0)
20	5.1 (0.7)	2.542 (0.01)	0.0 (0.0)	94.9 (0.7)
50	44.2 (1.6)	0.990 (0.004)	0.0 (0.0)	55.8 (1.6)
100	64.1 (1.5)	0.586 (0.002)	0.0 (0.0)	35.9 (1.5)
500	85.3 (1.2)	0.221 (0.0004)	1.1 (0.3)	13.6 (1.1)

Table 4. *Lognormal model,* $N(0, 10^4)$ *prior for* μ, $\Gamma(0.001, 0.001)$ *prior for* $\tau = \frac{1}{\sigma^2}$, *nominal 90%,* $N(0, 10)$ *data*

Need to bring in tail-weight and skewness **nonparametrically**.

Method 0: Appended ABC (ad hoc).

Given sample of size n, and using conjugate prior distribution, it is often helpful to think of prior as **equivalent to data set with effective sample size** m (for some m) which can be appended to the n data values.

The combined data set of $(m+n)$ observations can then be analyzed in frequentist way. Idea is to "teach" bootstrap about part of heavy tail of underlying lognormal distribution beyond largest data point $y_{(n)}$. Can try to do this by sampling m **"prior data points"** beyond a certain point, c, and then bootstrapping sample variance of $(m+n)$ points taken together.

Appended ABC Method

m	actual cov. (%)	mean length	% on left	% on right	mean variance
0	36.0 (1.5)	8.1 (0.7)	61.1 (1.5)	2.9 (0.5)	4.66 (0.49)
1	88.7 (1.0)	22.5 (1.6)	0.4 (0.2)	10.9 (1.0)	11.7 (0.66)
2	67.2 (1.5)	27.4 (1.8)	0.0 (0.0)	32.8 (1.5)	16.0 (0.73)
3	29.8 (1.4)	33.4 (1.9)	0.0 (0.0)	70.2 (1.4)	20.5 (0.74)

Table 5. *Appended ABC method*, $n = 10$, *nominal 90%*, $c = 5.60 = E(y_{(n)})$

m	actual coverage (%)	mean length	% on left	% on right	mean variance
0	76.8 (1.3)	5.43 (0.2)	18.5 (1.2)	4.7 (0.7)	4.68 (0.09)
1	82.4 (1.2)	10.5 (0.3)	0.0 (0.0)	17.6 (1.2)	6.67 (0.09)
2	53.7 (1.6)	13.9 (0.5)	0.0 (0.0)	46.3 (1.6)	8.58 (0.13)
3	20.7 (1.3)	16.7 (0.6)	0.0 (0.0)	79.3 (1.3)	10.5 (0.15)

Table 6. *Appended ABC method*, $n = 500$, *nominal 90%*, $c = 22.49 = E(y_{(n)})$

This sort of works with $m = 1$, but (a) highly imbalanced errors left-right and (b) coverage gets **worse** as n increases!

Method 1: Dirichlet process priors. To remove ad-hockery, work with **Bayesian nonparametric model**

$$\begin{aligned} F & \sim p(F) \\ (Y_1, \dots, Y_n | F) & \overset{\text{IID}}{\sim} F, \end{aligned} \qquad (6)$$

for some prior $p(F)$ on *infinite-dimensional space* $\mathcal{D}$ of all possible CDFs F. Use $p(F)$ to teach interval-generating method about tailweight and skewness.

Dirichlet Process Priors

Simplest $p(F)$ is class of **Dirichlet process priors** (Freedman, 1963; Ferguson, 1973).

Intuition: Freedman wanted to find **conjugate prior for empirical CDF $\widehat{F}_n$**.

IID sampling from $\widehat{F}_n =$ mass $\frac{1}{n}$ on each of $y_1, \ldots, y_n$ is **multinomial**:

Sort y_i into $k \leq n$ bins $b_1, \ldots, b_k$ ($k < n$ if ties) and let $n_j = \#(y_i$ in bin $b_j)$; then

$$Y^* \sim \widehat{F}_n \iff p(y^*) = c\, \theta_1^{n_1} \cdots \theta_k^{n_k}, \qquad (7)$$

$$\theta = (\theta_1, \ldots, \theta_k),\ \theta_j \geq 0,\ \textstyle\sum_{j=1}^k \theta_j = 1.$$

Conjugate prior for multinomial is **Dirichlet**: with $\alpha = (\alpha_1, \ldots, \alpha_k),\ \alpha_j \geq 0$,

$$\theta \sim D(\alpha) \iff p(\theta) = c\, \theta_1^{\alpha_1 - 1} \cdots \theta_k^{\alpha_k - 1}. \qquad (8)$$

Dirichlet Process Priors (continued)

So Freedman defined **Dirichlet process** as follows, e.g., for random variables on $\Re^1$ and with F having a density f:

Definition (Freedman, 1963). CDF $F \sim \mathcal{D}(\alpha)$ (F follows a **Dirichlet process** with parameter α, α itself a distribution) $\Longleftrightarrow$ for any (measurable) partition $A_1, \ldots, A_k$ of $\Re^1$, the random vector $[F(A_1), \ldots, F(A_k)]$ follows a Dirichlet distribution with parameter $[\alpha(A_1), \ldots, \alpha(A_k)]$, where $F(A_j)$ means the mass assigned to A_j by f.

Useful to express α in form $\alpha(\cdot) = cF_0(\cdot)$, where F_0 is the **centering** or **base distribution**—in the sense that $E(F) = F_0$—and c acts like a **prior sample size**.

With this way of writing α, **conjugate updating** becomes clear:

$$F \sim \mathcal{D}(cF_0),\ (Y_i|F) \overset{\text{IID}}{\sim} F \ \rightarrow\ (F|Y) \sim \mathcal{D}(c^* F^*)$$
$$c^* = c + n, \quad F^* = \frac{cF_0 + n\widehat{F}_n}{c+n}. \qquad (9)$$

Dirichlet Process Sampling

Sethuraman and Tiwari (1982) showed how to **sample** from a Dirichlet process:

$$F \sim \mathcal{D}(cF_0) \;\; \rightarrow \;\; F = \sum_{j=1}^{\infty} V_j \, \delta_{\theta_j}, \quad \text{where}$$

$$V_1 = W_1, V_j \;\; = \;\; W_j \prod_{k=1}^{j-1} (1 - W_k), j = 2, 3, ... \qquad (10)$$

Here $W_1, W_2, \ldots$ are IID Beta$(1, c)$, $\theta_1, \theta_2, \ldots$ are IID from F_0, and δ_{θ_j} is point mass at θ_j (this is the so-called **stick-breaking** algorithm, to be examined further in Parts 4–6).

This shows that **Dirichlet processes place all their mass on discrete CDFs**, which is in some contexts a drawback; **Dirichlet process mixture models** (Parts 4–6) solve this problem.

As c increases with $F \sim \mathcal{D}(cF_0)$, the sampling envelope around F_0 becomes tighter, because for any member A of a (measurable) partition of $\Re^1$,

$$V[F(A)] = \frac{F_0(A)[1 - F_0(A)]}{c + 1}, \qquad (11)$$

and increasing c **decreases variability** around F_0.

R Code for Dirichlet Process Sampling

```
rdir.ln <- function( m, cc ) {

  theta <- rlnorm( m, 0.0, 1.0 )

  v <- rep( 0, m )

  w <- rbeta( m, 1.0, cc )

  v[1] <- w[1]

  for ( j in 2:m ) {

    v[j] <- w[j] * v[j-1] * ( 1.0 - w[j - 1] ) / w[j - 1]

  }

  print( sum( v ) )

  temp <- cbind( v, theta )

  return( temp[order(temp[, 2], temp[, 1]), 1:2] )

}

rdir.Fstar <- function( m, cc, y ) {

  n <- length( y )

  theta <- rep( 0, m )

  for ( i in 1:m ) {

    U <- runif( 1 )

    S <- U < ( cc / ( cc + n ) )

    if ( S ) theta[i] <- rlnorm( 1, 0.0, 1.0 )

    else theta[i] <- sample( y, 1 )

  }
```

R Code (continued)

```
  v <- rep( 0, m )

  w <- rbeta( m, 1.0, cc )

  v[1] <- w[1]

  for ( j in 2:m ) {

    v[j] <- w[j] * v[j-1] * ( 1.0 - w[j - 1] ) / w[j - 1]

  }

  print( sum( v ) )

  temp <- cbind( v, theta )

  return( temp[order(temp[, 2], temp[, 1]), 1:2] )

}

test <- function( n, m, cc ) {

  y.1 <- rep( 0, n )

  y.2 <- rep( 0, n )

  for ( i in 1:n ) {

    sample <- rdir.ln( m, cc )

    y.1[i] <- sum( sample[,1] * sample[,2] )

    y.2[i] <- sum( sample[,1] * sample[,2]^2 )

  }

  return( c( mean( y.1 ), mean( y.2 - y.1^2 ) ) )

}
```

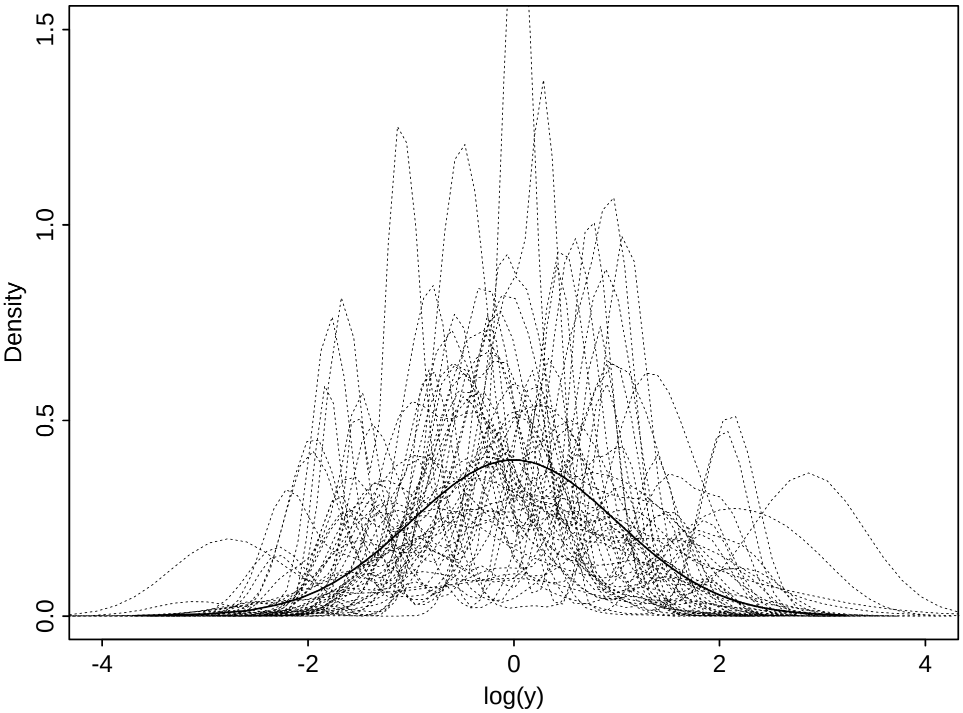
R Code (continued)

```
grid <- seq(0,8,length=500)

plot(grid,dlnorm(grid),type=’l’,lwd=2,ylim=c(0,0.8))

for ( i in 1:50) {

  temp <- rdir.ln(100,10)

  data <- rep( temp[,2], round(10000*temp[,1]) )

  temp <- density(log(data),width=(max(data)-min(data))/4)

  temp$x <- exp(temp$x)

  lines(temp,lty=2)

}

grid <- log(seq(0,25,length=1000))

plot(grid,dnorm(grid),type=’l’,lwd=2,ylim=c(0,1.5),
  xlim=c(-4,4),xlab=’log(y)’,ylab=’Density’)

for ( i in 1:50) {

  temp <- rdir.ln(100,5)

  data <- rep( temp[,2], round(10000*temp[,1]) )

  temp <- density(log(data),width=(max(data)-min(data))/8)

  lines(temp,lty=2)

}
```

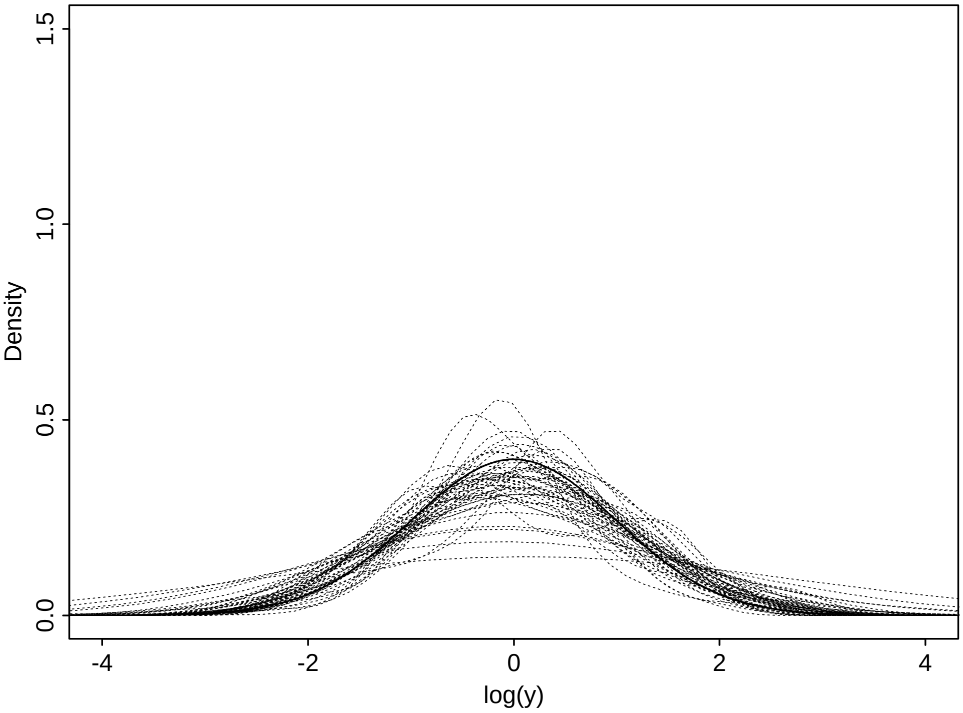
R Code (continued)

```
y = rlnorm( 100 ) + 1

grid <- log(seq(0,25,length=1000))

plot(grid,dnorm(grid),type=’l’,lwd=2,ylim=c(0,1.5),
  xlim=c(-4,4),xlab=’log(y)’,ylab=’Density’)

for ( i in 1:50) {

  temp <- rdir.Fstar(150,15,y)

  data <- rep( temp[,2], round(10000*temp[,1]) )

  temp <- density(log(data),width=(max(data)-min(data))/8)

  lines(temp,lty=2)

}
```

Dirichlet Process Sampling (continued)



Figures 3, 4. *Standard normal density (solid curve) and smoothed density traces of 50 draws (plotted on the log scale) from the Dirichlet process prior $\mathcal{D}(cF_0)$ with $c = 5$ (above) and 50 (below) and $F_0 =$ the standard lognormal distribution (dotted curves)*.



Bayesian Nonparametric Intervals

[show `R` **movies** now]

From (8), sampling draws from F^* with Dirichlet process prior is easy:

Generate $S \sim U(0,1)$, then sample from F_0 if $S \leq \frac{c}{c+n}$ and from $\hat{F}_n$ otherwise.

Direct **generalization** of bootstrap (also see Rubin, 1981): when $c = 0$ sample entirely from $\hat{F}_n$ (bootstrap), but when $c > 0$ tail-weight and skewness information comes in from F_0.

Now to simulate from posterior distribution of $V(F)$, just repeatedly draw F^* from $\mathcal{D}(cF_0 + n\hat{F}_n)$ and calculate $V(F^*)$.

c acts like **tuning constant**: successful nonparametric intervals for $V(F)$ will result if compromise c can be found that leads to well-calibrated intervals across broad range of underlying F.

Calibration Properties

Table 7. *Actual coverage of nominal 90% intervals for population variance, using Dirichlet process prior centered at $LN(0,1)$.*

n	Distribution	c	Actual Coverage	Mean Length	% on Left	% on Right
10	Gaussian	3.1	90.6	5.22	8.7	0.7
	Gamma	3.9	90.8	6.91	8.1	1.1
	Lognormal	4.3	89.7	8.70	9.2	1.1
20	Gaussian	3.7	89.3	4.26	10.0	0.7
	Gamma	6.6	90.0	6.40	9.5	0.5
	Lognormal	8.3	90.9	7.69	8.2	0.9
50	Gaussian	4.8	90.4	3.00	8.5	1.1
	Gamma	14.1	89.8	4.90	9.7	0.5
	Lognormal	20.2	90.4	6.68	8.7	0.9
100	Gaussian	5.5	91.1	2.19	7.1	1.8
	Gamma	18.0	90.1	3.93	9.1	0.8
	Lognormal	37.8	89.3	5.94	9.7	1.0

Compromise c are possible, e.g., with $n = 10$, $c \doteq 4.3$ produces actual coverage near 90% for the lognormal and coverage slightly in excess of 90% for lighter-tailed, less skewed data.

Note how much **narrower** intervals are than parametric Bayesian intervals in lognormal model (Table 3), e.g., with $n = 20$ parametric intervals had mean length 145.2 (versus 7.7 above)!

However, errors in Table 7 are still **badly asymmetric**.

Next Step

Table 7 is still **cheating**, though: $F_0 = LN(0,1)$, and mean and variance of data-generating distributions were chosen to match those of F_0 to avoid location and scale inconsistencies.

Solution: allow **base distribution** to be **indexed parametrically**, as in the model

$$
\begin{array}{rcl}
z_i & = & \ln(y_i), \\
(z_i|F) & \overset{\text{IID}}{\sim} & F \\
F & \sim & \mathcal{D}(cF_0), \\
F_0 & = & N(\mu, \sigma^2) \\
(\mu, \sigma^2) & \sim & p(\mu, \sigma^2) \\
c & \sim & p(c)
\end{array}
\qquad (12)
$$

This model may be **fit** via **MCMC**, using methods to be described in Parts 4–6.

Pólya Trees

Case study (introducing **Pólya trees**): **risk assessment** in **nuclear waste disposal**.

This **case study** will be examined in more detail in Part 8; it turns out that it also involves data that would be **parametrically** modeled as **lognormal**.

As Part 8 will make clear, in this problem would be good to be able to build a model that is **centered** at the lognormal, but which can **adapt** to other distributions when the data suggest this is necessary.

A modeling approach based on **Pólya trees** (Lavine, 1992, 1994; Walker et al., 1998), first studied by Ferguson (1974), is one way forward.

The model in Part 8 will involve a **mixture** of a **point mass at 0** and positive values with a **highly skewed** distribution.

One way to write the **parametric Bayesian lognormal model** for the positive data values is

$$\begin{aligned} \log(Y_i) &= \mu + \sigma\, e_i \\ (\mu, \sigma^2) &\sim p(\mu, \sigma^2) \\ e_i &\overset{\text{IID}}{\sim} N(0, 1), \end{aligned} \tag{13}$$

for some prior distribution $p(\mu, \sigma^2)$ on μ and σ^2.

The Pólya trees idea is to replace the last line of (13), which expresses certainty about the distribution of the e_i, with a **distribution on the set of possible distributions** F for the e_i.

Pólya Trees (continued)

The new model is

$$\begin{array}{rcl} \log(Y_i) & = & \mu + \sigma\, e_i \\ (\mu, \sigma^2) & \sim & p(\mu, \sigma^2) \\ (e_i | F) & \overset{\text{IID}}{\sim} & F \quad \text{(mean 0, SD 1)} \\ F & \sim & PT\,(\Pi, \mathcal{A}_c)\,. \end{array} \tag{14}$$

Here (a) $\Pi = \{B_\epsilon\}$ is a **binary tree partition** of the real line, where ϵ is a binary sequence which locates the set B_ϵ in the tree.

You get to choose these sets B_ϵ in a way that **centers the Pólya tree on any distribution you want**, in this case the standard normal.

This is done by choosing the cutpoints on the line, which define the partitions, **based on the quantiles of** $N(0, 1)$:

Level	Sets	Cutpoint(s)
1	(B_0, B_1)	$\Phi^{-1}(\frac{1}{2}) = 0$
2	$(B_{00}, B_{01},$ $B_{10}, B_{11})$	$\Phi^{-1}(\frac{1}{4}) = -0.674, \Phi^{-1}(\frac{1}{2}) = 0,$ $\Phi^{-1}(\frac{3}{4}) = +0.674$
$\vdots$	$\vdots$	$\vdots$

(Φ is the $N(0, 1)$ CDF.) In practice this process has to stop somewhere; I use a tree **defined down to level** $M = 8$, which is like working with **random histograms**, each with $2^8 = 256$ bins.

Pólya Trees (continued)

And (b) Walker et al. (1998):

A helpful image is that of a **particle cascading through the partitions** B_ϵ. It starts [on the real line] and moves into B_0 with probability C_0 or into B_1 with probability $C_1 = 1 - C_0$. In general, on entering B_ϵ the particle could either move into $B_{\epsilon 0}$ or into $B_{\epsilon 1}$. Let it move into the former with probability $C_{\epsilon 0}$ or into the latter with probability $C_{\epsilon 1} = 1 - C_{\epsilon 0}$. For Pólya trees, these probabilities are random, **beta** variables, $(C_{\epsilon 0}, C_{\epsilon 1}) \sim \text{beta}(\alpha_{\epsilon 0}, \alpha_{\epsilon 1})$ with non-negative $\alpha_{\epsilon 0}$ and $\alpha_{\epsilon 1}$. If we denote the collection of α's by $\mathcal{A}$, a particular Pólya tree distribution is completely defined by Π and $\mathcal{A}$.

To make a Pólya tree distribution choose a continuous distribution with probability 1, the α's have to **grow quickly** as the level m of the tree increases. Following Walker et al. (1998) I take

$$\alpha_\epsilon = c\,m^2 \text{ whenever } \epsilon \text{ defines a set at level } m, \tag{15}$$

and this defines $\mathcal{A}_c$.

$c > 0$ is a kind of **tuning constant**: with small c the posterior distribution for the CDF of the e_i will be based almost completely on $\widehat{F}_n$, the empirical CDF (the "data distribution") for the e_i, whereas with large c the posterior will be based almost completely on the prior centering distribution, in this case $N(0,1)$.

Prior to Posterior Updating

Prior to posterior updating is easy with Pólya trees: if

$$\begin{aligned} F & \sim PT(\Pi, \mathcal{A}) \\ (Y_i | F) & \overset{\text{IID}}{\sim} F \end{aligned} \qquad (16)$$

and (say) Y_1 is observed, then the posterior $p(F|Y_1)$ for F given Y_1 is **also a Pólya tree** with

$$(\alpha_\epsilon | Y_1) = \left\{ \begin{array}{cc} \alpha_\epsilon + 1 & \text{if } Y_1 \in B_\epsilon \\ \alpha_\epsilon & \text{otherwise} \end{array} \right\}. \qquad (17)$$

In other words the updating follows a **Pólya urn scheme** (e.g., Feller, 1968): at each level of the tree, if Y_1 falls into a particular partition set B_ϵ, then 1 is added to the α for that set.

Figs. 5–7 show the **variation around** $N(0, 1)$ obtained by sampling from a $PT(\Pi, \mathcal{A}_c)$ prior for F as c varies from 10 down to 0.1, and Figs. 8–10 illustrate prior to posterior updating for the same range of c with a fairly skewed data set.

R code to perform these **Pólya-tree simulations** is given on the next several pages.

[show R **movie** now]

R Code For Pólya Trees

```
polya.sim1 <- function( M, n.sim, cc ) {

  b <- matrix( 0, M, 2^M - 1 )

  for ( i in 1:M ) {

    b[i,1:( 2^i - 1 )] <- qnorm( ( 1:( 2^i - 1 ) ) / 2^i )

  }

  alpha <- matrix( 0, M, 2^M )

  for ( i in 1:M ) {

    alpha[i, 1:( 2^i )] <- cc * rep( i^2, 2^i )

  }

  par( mfrow = c( 2, 1 ) )

  plot( seq( -3, 3, length = 500 ), dnorm( seq( -3, 3,
    length = 500 ) ),
    type = ’l’, xlab = ’y’, ylab = ’Density’, ylim = c( 0, 1.5 ) )
#     main = paste( ’c =’, cc, ’, n.sim =’, n.sim ) )

  F.star.cumulative <- rep( 0, 2^M )

  b.star <- c( -3, b[M,], 3 )

  for ( i in 1:n.sim ) {

    F.star <- rep( 1, 2^M )

    for ( j in 1:M ) {

      for ( k in 1:( 2^( j - 1 ) ) ) {

        C <- rbeta( 1, alpha[j, 2 * k - 1], alpha[j, 2 * k] )

        F.star[( 1 + ( k - 1 ) * 2^( M - j + 1 ) ):( ( 2 * k - 1 ) *
          2^( M - j ) )] <- F.star[( 1 + ( k - 1 ) * 2^( M - j + 1 ) ):
          ( ( 2 * k - 1 ) * 2^( M - j ) )] * C
```

R Code (continued)

```
      F.star[( ( 2 * k - 1 ) * 2^( M - j ) + 1 ):( k * 2^( M - j +
        1 ) )] <- F.star[( ( 2 * k - 1 ) * 2^( M - j ) + 1 ):( k *
        2^( M - j + 1 ) )] * ( 1 - C )

    }

  }

  F.star.cumulative <- F.star.cumulative + F.star

  n <- round( 10000 * F.star )

  y <- NULL

  for ( j in 1:2^M ) {

    y <- c( y, runif( n[j], b.star[j], b.star[j+1] ) )

  }

  lines( density( y ), lty = 2 )

  print( i )

}

F.star.cumulative <- F.star.cumulative / n.sim

n <- round( 10000 * F.star.cumulative )

y <- NULL

for ( i in 1:2^M ) {

  y <- c( y, runif( n[i], b.star[i], b.star[i+1] ) )

}

hist( y, nclass = 20, probability = T, ylab = ’Density’,
  xlim = c( -3, 3 ), ylim = c( 0, 0.5 ), xlab = ’y’ )

lines( seq( -3, 3, length = 500 ), dnorm( seq( -3, 3, length = 500 ) ) )
```

R Code (continued)

```
  lines( density( y ), lty = 2 )

  return( cat( "\007" ) )

}

y <- exp( rnorm( 100 ) ) - 2

polya.update <- function( M, n.sim, cc, y ) {

  b <- matrix( 0, M, 2^M - 1 )

  for ( i in 1:M ) {

    b[i,1:( 2^i - 1 )] <- qnorm( ( 1:( 2^i - 1 ) ) / 2^i )

  }

  b.star <- c( -3, b[ M, ], 3 )

  infinity <- 2 * abs( max( min( y ), max( y ), min( b ), max( b ) ) )

  alpha <- matrix( 0, M, 2^M )

  for ( i in 1:M ) {

    alpha[i, 1:( 2^i )] <- cc * rep( i^2, 2^i )

  }

  par( mfrow = c( 1, 1 ) )

  hist( y, xlim = c( -3, max( y ) ), ylim = c( 0, 1.5 ), xlab = ’y’,
    ylab = ’Density’, probability = T, nclass = 20,
    main = paste( ’n.sim =’, n.sim, ’, c =’, cc, ’, n = ’, length( y ) ) )

  for ( i in 1:n.sim ) {

    F.star <- rep( 1, 2^M )

    for ( j in 1:M ) {

      for ( k in 1:( 2^( j - 1 ) ) ) {
```

R Code (continued)

```
      C <- rbeta( 1, alpha[j, 2 * k - 1], alpha[j, 2 * k] )

      F.star[( 1 + ( k - 1 ) * 2^( M - j + 1 ) ):( ( 2 * k - 1 ) *
        2^( M - j ) )] <- F.star[( 1 + ( k - 1 ) * 2^( M - j + 1 ) ):
        ( ( 2 * k - 1 ) * 2^( M - j ) )] * C

      F.star[( ( 2 * k - 1 ) * 2^( M - j ) + 1 ):( k * 2^( M - j +
        1 ) )] <- F.star[( ( 2 * k - 1 ) * 2^( M - j ) + 1 ):( k *
        2^( M - j + 1 ) )] * ( 1 - C )

    }

  }

  n <- round( 10000 * F.star )

  y.star <- NULL

  for ( j in 1:2^M ) {

    y.star <- c( y.star, runif( n[j], b.star[j], b.star[j+1] ) )

  }

  lines( density( y.star ), lty = 1 )

}

for ( i in 1:M ) {

  n <- hist( y, breaks = c( - infinity, b[i,1:( 2^i - 1 )],
    infinity ), plot = F )$counts
  alpha[i, 1:( 2^i )] <- alpha[i, 1:( 2^i )] + n

}

for ( i in 1:n.sim ) {

  F.star <- rep( 1, 2^M )

  for ( j in 1:M ) {

    for ( k in 1:( 2^( j - 1 ) ) ) {
```

R Code (continued)

```
    C <- rbeta( 1, alpha[j, 2 * k - 1], alpha[j, 2 * k] )

    F.star[( 1 + ( k - 1 ) * 2^( M - j + 1 ) ):( ( 2 * k - 1 ) *
     2^( M - j ) )] <- F.star[( 1 + ( k - 1 ) * 2^( M - j + 1 ) ):
     ( ( 2 * k - 1 ) * 2^( M - j ) )] * C

    F.star[( ( 2 * k - 1 ) * 2^( M - j ) + 1 ):( k * 2^( M - j +
     1 ) )] <- F.star[( ( 2 * k - 1 ) * 2^( M - j ) + 1 ):( k *
     2^( M - j + 1 ) )] * ( 1 - C )

   }

  }

  n <- round( 10000 * F.star )

  y.star <- NULL

  for ( j in 1:2^M ) {

   y.star <- c( y.star, runif( n[j], b.star[j], b.star[j+1] ) )

  }

  lines( density( y.star ), lty = 2 )

 }

 return( "quack!" )

}
```

Sampling From the Prior

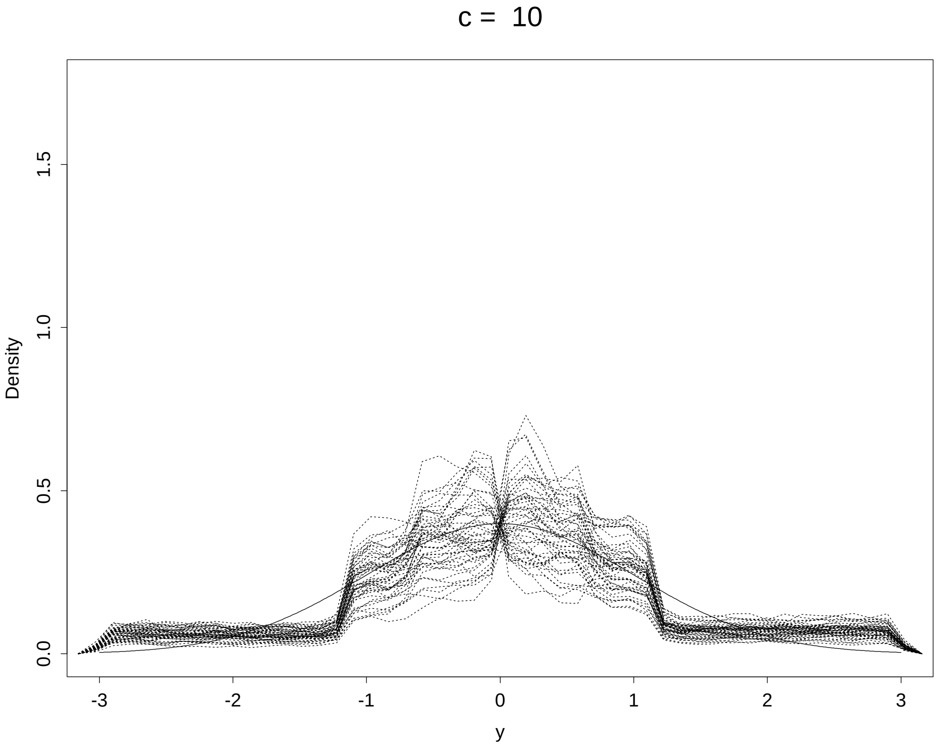


Figure 5. *Sampling from a $PT(\Pi, \mathcal{A}_c)$ prior for F centered at $N(0,1)$ (solid line) with $c = 10$. For large c the sampled distribution follows the prior pretty closely.*

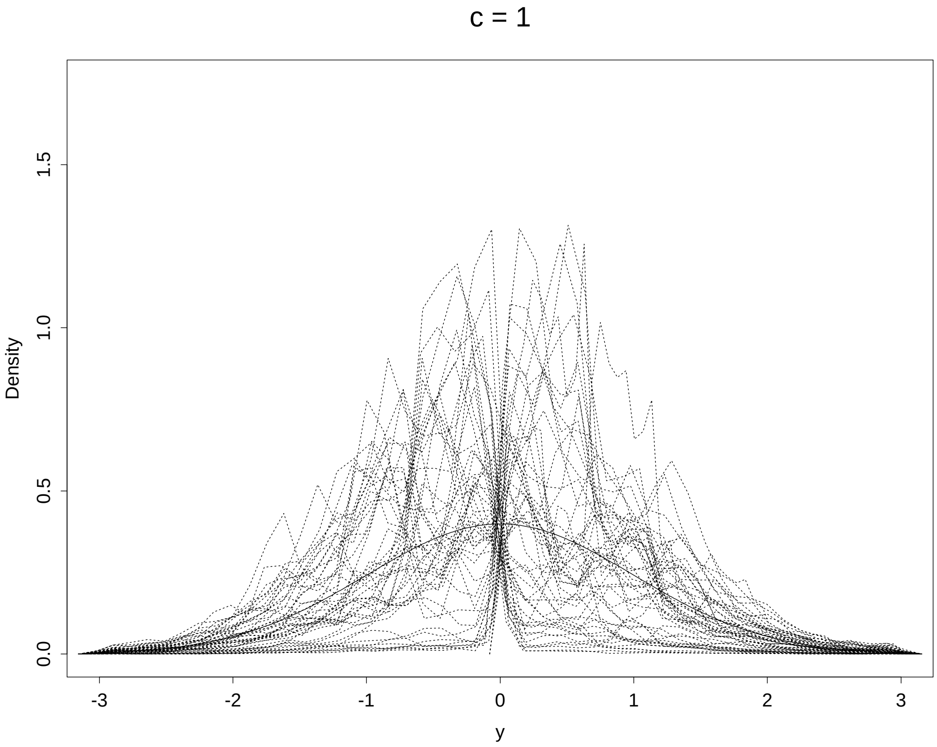


Figure 6. *Like Fig. 5 but with $c = 1$. The sampled F's are varying more around $N(0,1)$ with a smaller c.*

Pólya Tree Illustrations

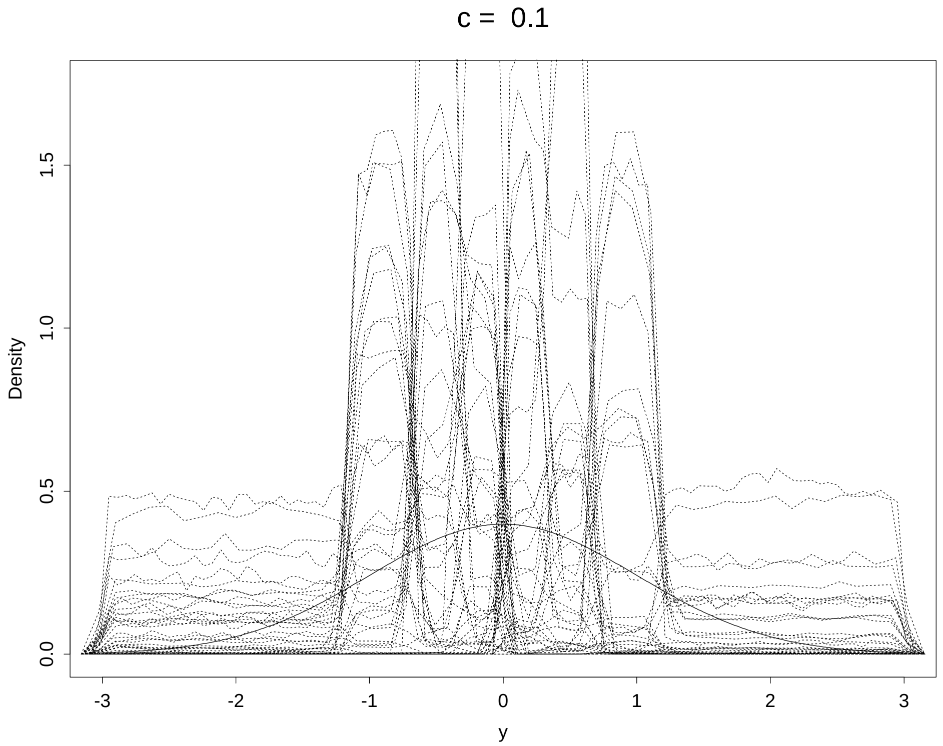


Figure 7. *Like Figs. 5 and 6 but with $c = 0.1$. With small c the sampled F bears little relation to the centering distribution.*

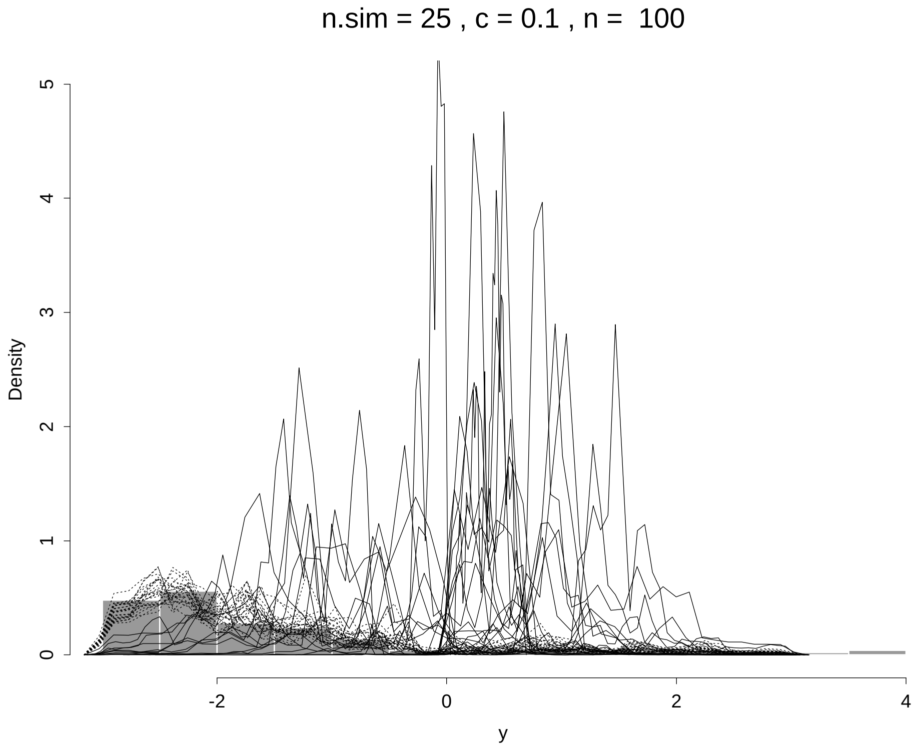


Figure 8. *Draws from the prior (solid lines); data (histogram, $n = 100$); and draws from the posterior (dotted lines), with $c = 0.1$. With c close to 0 the posterior almost coincides with the data.*

More Pólya Tree Illustrations

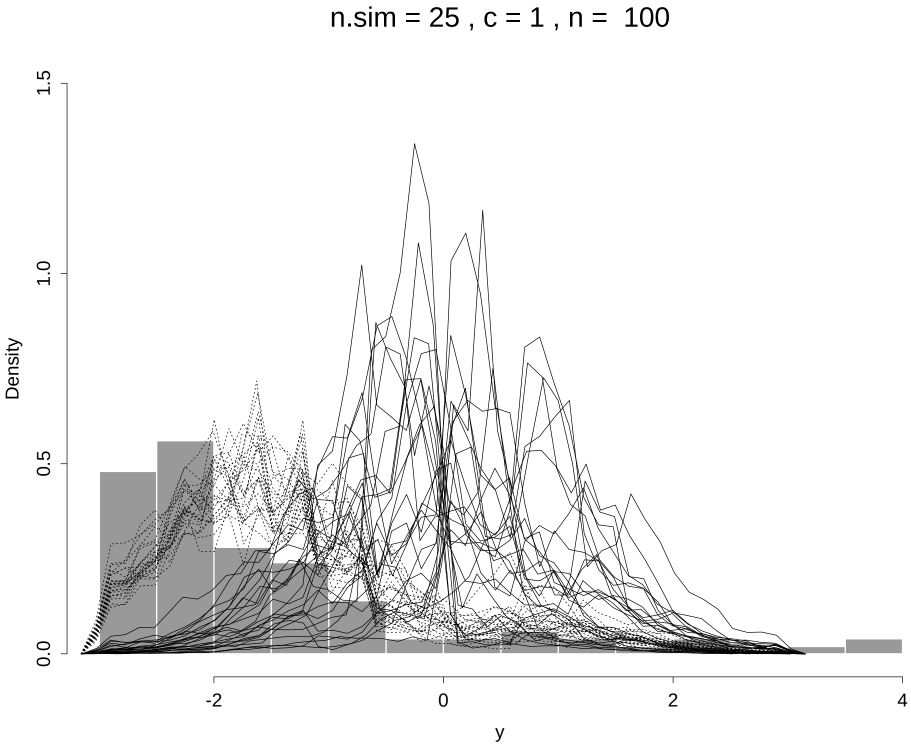


Figure 9. *Like Fig. 8 but with $c = 1$. The posterior is now a compromise between the prior and the data.*

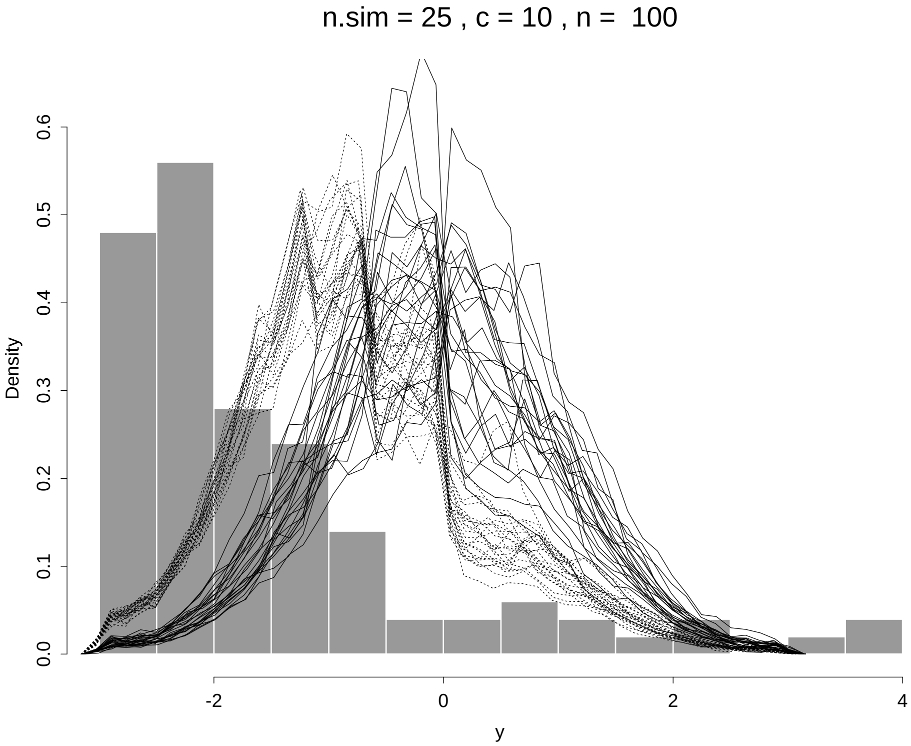


Figure 10. *Like Figs. 8 and 9 but with $c = 10$. Now the posterior is much closer to the prior.*

Part 3 References

Efron B, Tibshirani RJ (1993). *An Introduction to the Bootstrap*. London: Chapman & Hall.

Escobar MD, West M (1995). Bayesian density estimation and inference using mixtures. *Journal of the American Statistical Association*, **90**, 577–588.

Ferguson TS (1973). A Bayesian analysis of some non-parametric problems. *Annals of Statistics*, **1**, 209–230.

Freedman DA (1963). On the asymptotic behavior of Bayes estimates in the discrete case I. *Annals of Mathematical Statistics*, **34**, 1386–1403.

Lee SMS, Young GA (1995). Asymptotic iterated bootstrap confidence intervals. *Annals of Statistics*, **23**, 1301-1330.

Rubin DB (1981). The Bayesian bootstrap. *Annals of Statistics*, **9**, 130-134.

Sethuraman J, Tiwari R (1982). Convergence of Dirichlet measures and the interpretation of their parameter. *Proceedings of the Third Purdue Symposium on statistical Decision Theory and Related Topics*, SS Gupta, JO Berger (Eds.). New York: Academic press.

Walker SG, Damien P, Laud, PW, Smith AFM (1997) Bayesian nonparametric inference for random distributions and related functions. Technical Report, Department of Mathematics, Imperial College.

Part 3 References (continued)

Draper D (1997). Model uncertainty in "stochastic" and "deterministic" systems. In *Proceedings of the 12th International Workshop on Statistical Modeling*, Minder C, Friedl H (eds.), Vienna: *Schriftenreihe der Österreichischen Statistichen Gesellschaft*, **5**, 43–59.

Draper D (2005). *Bayesian Modeling, Inference and Prediction*. New York: Springer-Verlag, forthcoming.

Efron B, Tibshirani RJ (1993). *An Introduction to the Bootstrap*. London: Chapman & Hall.

Feller W (1968). *An Introduction to Probability Theory and Its Applications, Volume I*, Third Edition. New York: Wiley.

Ferguson TS (1974). Prior distributions on spaces of probability measures. *Annals of Statistics*, **2**, 615–629.

Gilks WR, Richardson S, Spiegelhalter DJ (1996). *Markov Chain Monte Carlo in Practice*. London: Chapman & Hall.

Johnson NL, Kotz S (1970). *Distributions in Statistics: Continuous Univariate Distributions, Volume 1*. Boston: Houghton-Mifflin.

Lavine M (1992). Some aspects of Pólya tree distributions for statistical modeling. *Annals of Statistics*, **20**, 1203–1221.

Lavine M (1994). More aspects of Pólya trees for statistical modeling. *Annals of Statistics*, **20**, 1161–1176.

PSAC (Probabilistic System Assessment Code) User Group (1989). *PSACOIN Level E Intercomparison*. Nuclear Energy Agency: Organization for Economic Co-operation and Development.

Part 3 References (continued)

Sinclair J (1996). Convergence of risk estimates obtained from highly skewed distributions. AEA Technology briefing.

Sinclair J, Robinson P (1994). The unsolved problem of convergence of PSA. Presentation at the 15th meeting of the NEA Probabilistic System Assessment Group, 16–17 June 1994, Paris.

Spiegelhalter DJ, Thomas A, Best NG, Gilks WR (1997). *BUGS: Bayesian inference Using Gibbs Sampling, Version 0.6*. Cambridge: Medical Research Council Biostatistics Unit.

Walker SG, Damien P, Laud PW, Smith AFM (1998). Bayesian nonparametric inference for random distributions and related functions. Technical report, Department of Mathematics, Imperial College, London.

Woo G (1989). Confidence bounds on risk assessments for underground nuclear waste repositories. *Terra Nova*, **1**, 79–83.