m4.3

Chris Parrish

June 16, 2016

Contents

m4.3

references:
- McElreath, Statistical Rethinking, chap 4, pp.95-114
- plotting multiple figures

!Kung model

Acknowledgments: The modeling code in this file is from chapter 4 of Richard McElreath's ***Statistical Rethinking****, but* `ggplot2` *rather than R base graphics is used to visualize the results.*

data

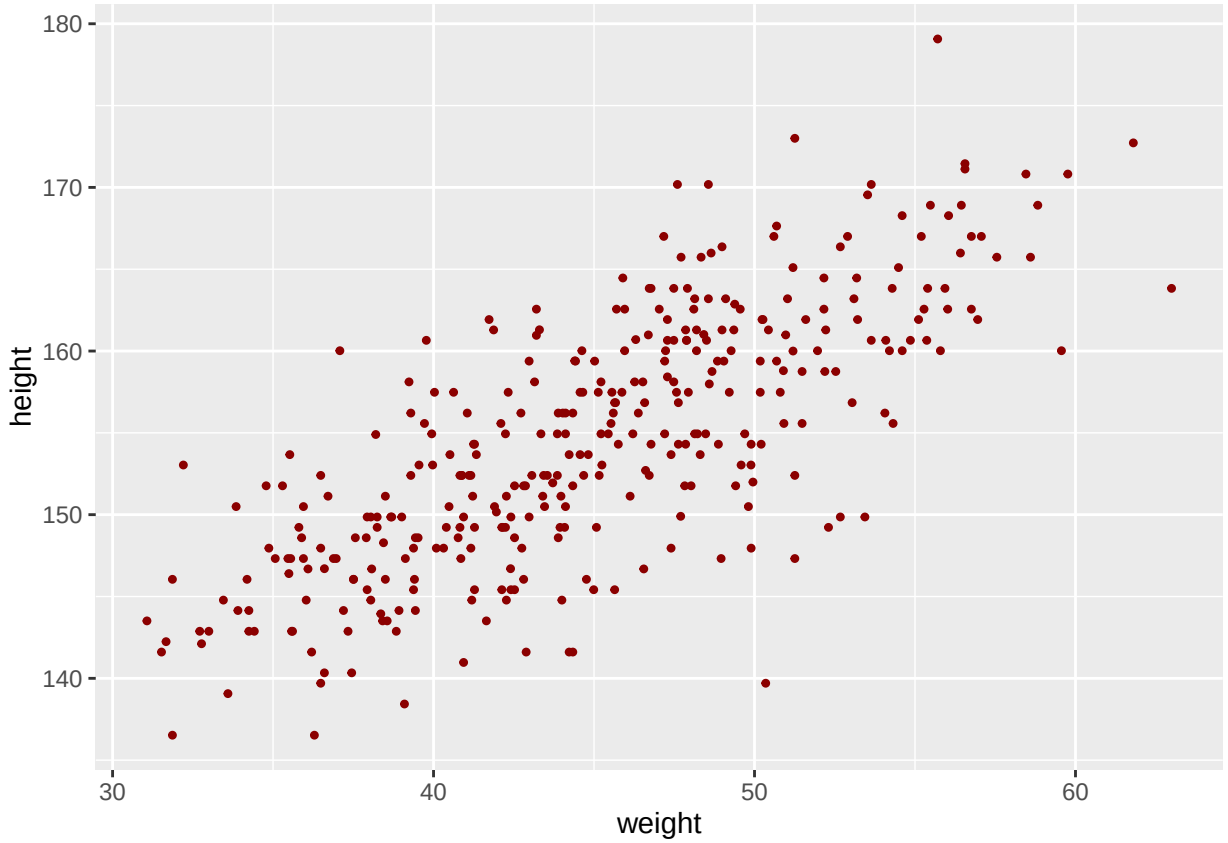
```
## R code 4.38
library(rethinking)
data(Howell1)
d <- Howell1
d2 <- d[ d$age >= 18 , c("weight", "height")]
head(d2)
```

```
##     weight  height
## 1 47.82561 151.765
## 2 36.48581 139.700
```

```
## 3 31.86484 136.525
## 4 53.04191 156.845
## 5 41.27687 145.415
## 6 62.99259 163.830
```

scatterplot with ggplot2

```
library(ggplot2)
ggplot(d2, aes(weight, height)) +
  geom_point(shape = 20, color = "darkred")
```



model

$$
\begin{aligned}
h_i &\sim Normal(\mu_i, \sigma) \\
\mu_i &= \alpha + \beta x_i \\
\alpha &\sim Normal(156, 100) \\
\beta &\sim Normal(0, 10) \\
\sigma &\sim Uniform(0, 50)
\end{aligned}
$$

map

fit the model with `map`

```
## R code 4.38
# fit model
m4.3 <- map(
    alist(
        height ~ dnorm( mu , sigma ) ,
        mu <- a + b*weight ,
        a ~ dnorm( 156 , 100 ) ,
        b ~ dnorm( 0 , 10 ) ,
        sigma ~ dunif( 0 , 50 )
    ) ,
    data=d2 )
```

```
## R code 4.40
precis( m4.3 )
```

```
##         Mean StdDev   5.5%  94.5%
## a     113.90   1.91 110.86 116.95
## b       0.90   0.04   0.84   0.97
## sigma   5.07   0.19   4.77   5.38
```

view parameter distributions with `ggplot2`

```
## R code 4.46
post <- extract.samples( m4.3 )
str(post)
```

```
## 'data.frame':    10000 obs. of  3 variables:
##  $ a    : num  112 114 117 109 112 ...
##  $ b    : num  0.943 0.888 0.838 1.013 0.958 ...
##  $ sigma: num  5.06 4.86 4.96 4.97 5.02 ...
```

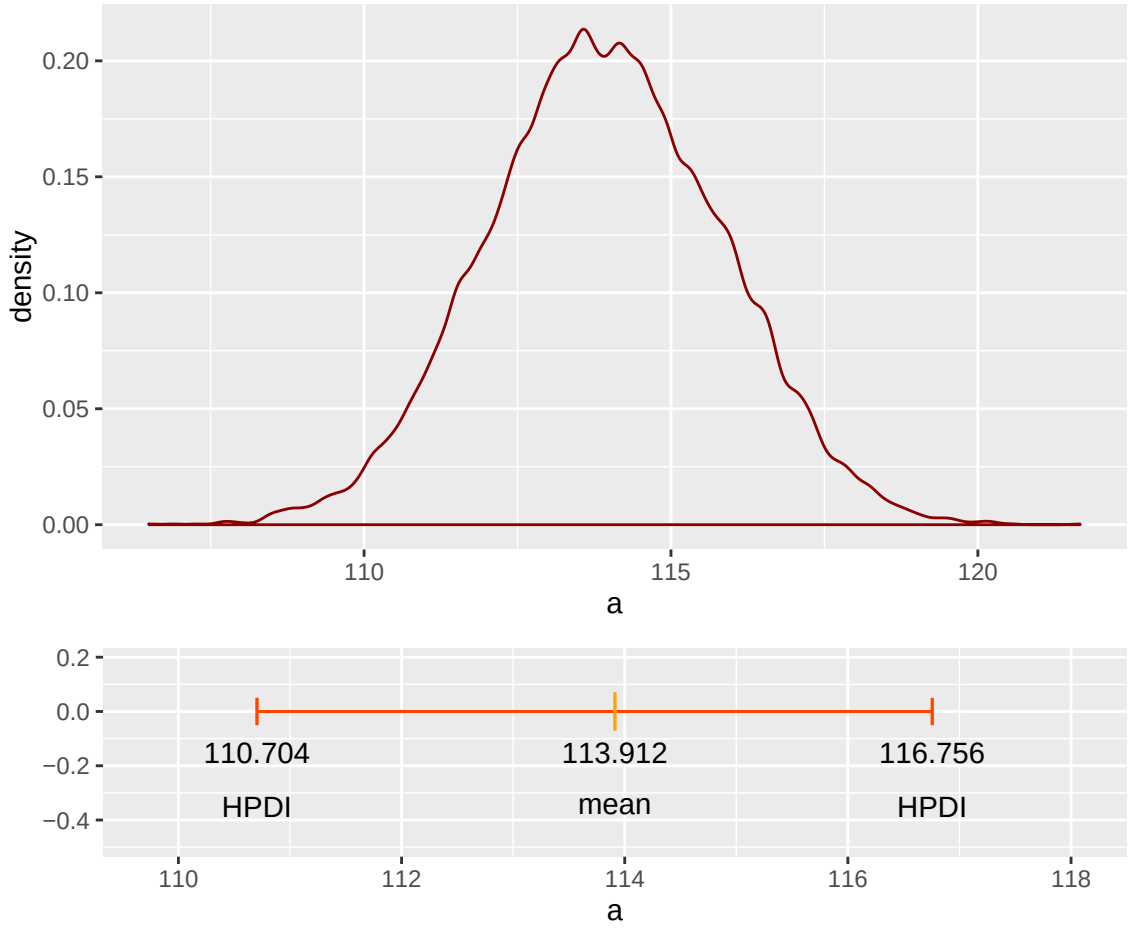
```
gg.draw.HPDI <- function(center, hpdi, parameter){
  me <- center - hpdi[1]              #  margin of error
  x.min <- center - 1.3 * me          #  boundaries for window onto CI
  x.max <- center + 1.3 * me
  data <- data.frame(x = 0:1, y = c(-0.5, 0.5))
  lwr.label <- paste("HPDI")
  phat.label <- paste("mean")
  upr.label <- paste("HPDI")
  ggplot(data, aes(x, y)) +
    coord_cartesian(xlim = c(x.min, x.max), ylim = c(-0.5, 0.2)) +
    geom_segment(x = hpdi[1], y = 0, xend = hpdi[2], yend = 0, color = "orangered") +
    geom_segment(x = hpdi[1], y = -0.05, xend = hpdi[1], yend = 0.05, color = "orangered") +
    geom_segment(x = hpdi[2], y = -0.05, xend = hpdi[2], yend = 0.05, color = "orangered") +
    geom_segment(x = center, y = -0.07, xend = center, yend = 0.07, color = "orange") +
    annotate("text", x = hpdi[1], y = -0.15, label = round(hpdi[1], 3)) +
    annotate("text", x = center, y = -0.15, label = round(center, 3)) +
    annotate("text", x = hpdi[2], y = -0.15, label = round(hpdi[2], 3)) +
    annotate("text", x = hpdi[1], y = -0.35, label = lwr.label, parse = TRUE) +
    annotate("text", x = center, y = -0.35, label = phat.label, parse = TRUE) +
```

```
    annotate("text", x = hpdi[2], y = -0.35, label = upr.label, parse = TRUE) +
    labs(x = parameter, y = "") +
    theme_gray()
}
```

```
library(cowplot)
parameter.dist <- function(parameter, values){
  d <- data.frame(vals = values)
  param.mu <- mean(d$vals)
  param.HPDI <- HPDI(d$vals)
  plot.dens <- ggplot(d, aes(vals)) +
    geom_density(adjust = 0.5, color = "darkred") +
    labs(x = parameter, y = "density") +
    theme_gray()
  plot.HPDI <- gg.draw.HPDI(param.mu, param.HPDI, parameter)
  image <- ggdraw() +
    draw_plot(plot.dens, 0, 0.32, 1, 0.67) +
    draw_plot(plot.HPDI, 0, 0, 1, .32)
  return(image)
}
```

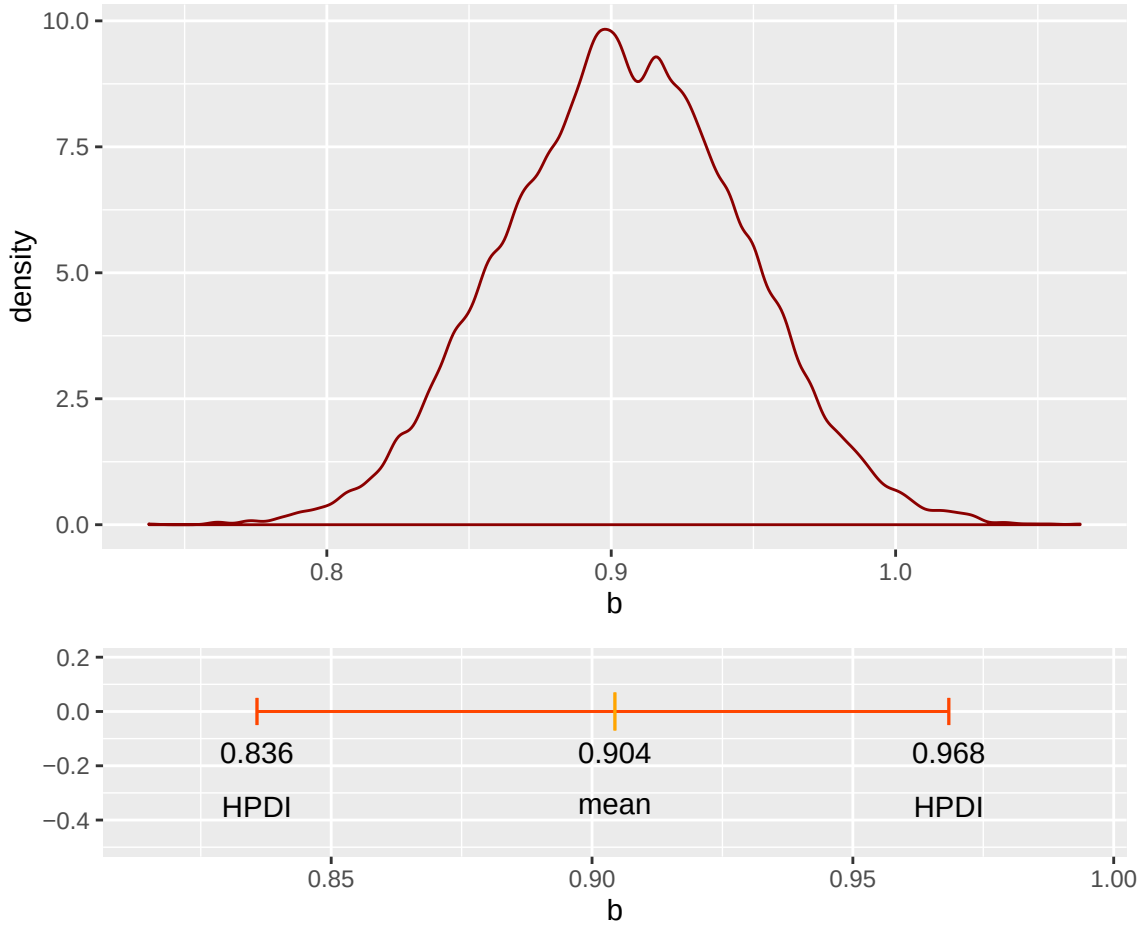
a

```
a.plot <- parameter.dist(parameter = "a", values = post$a)
a.plot
```



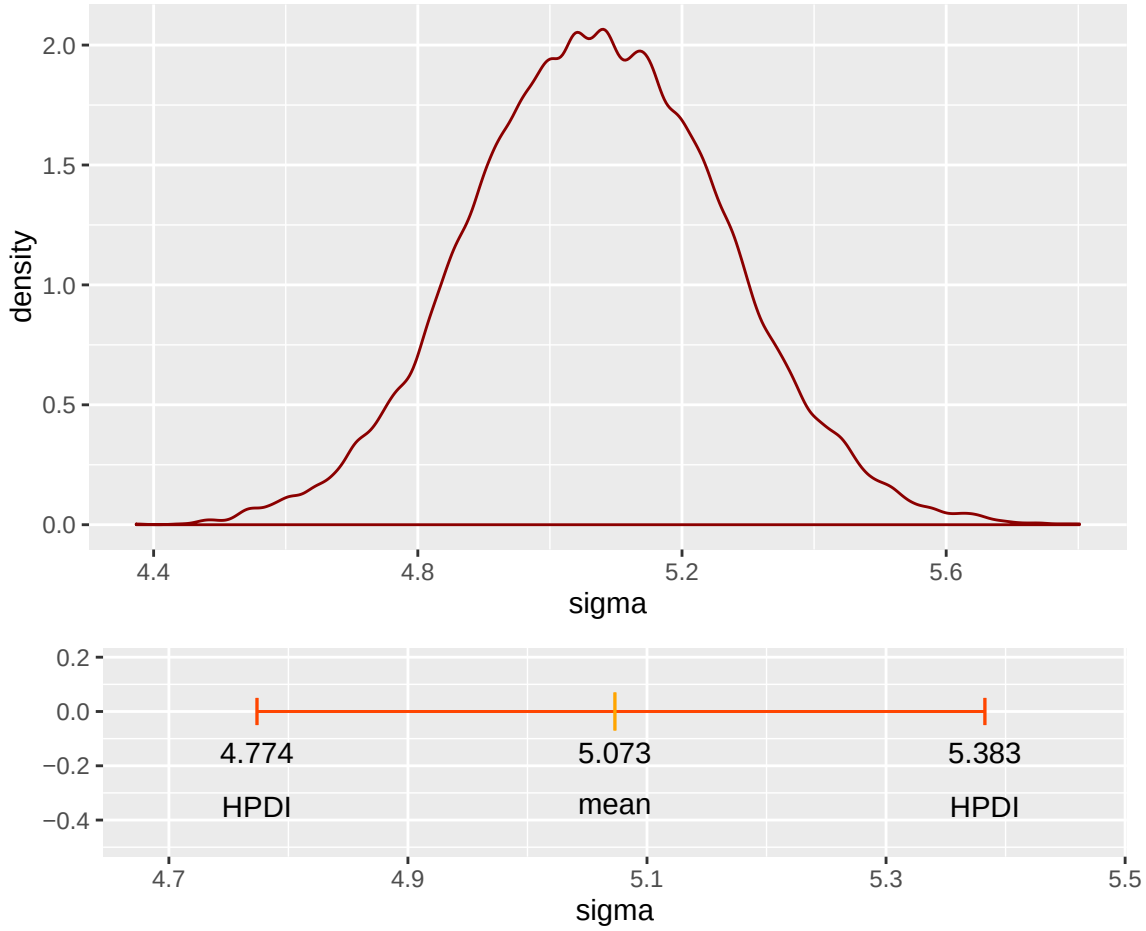
b

```
b.plot <- parameter.dist(parameter = "b", values = post$b)
b.plot
```



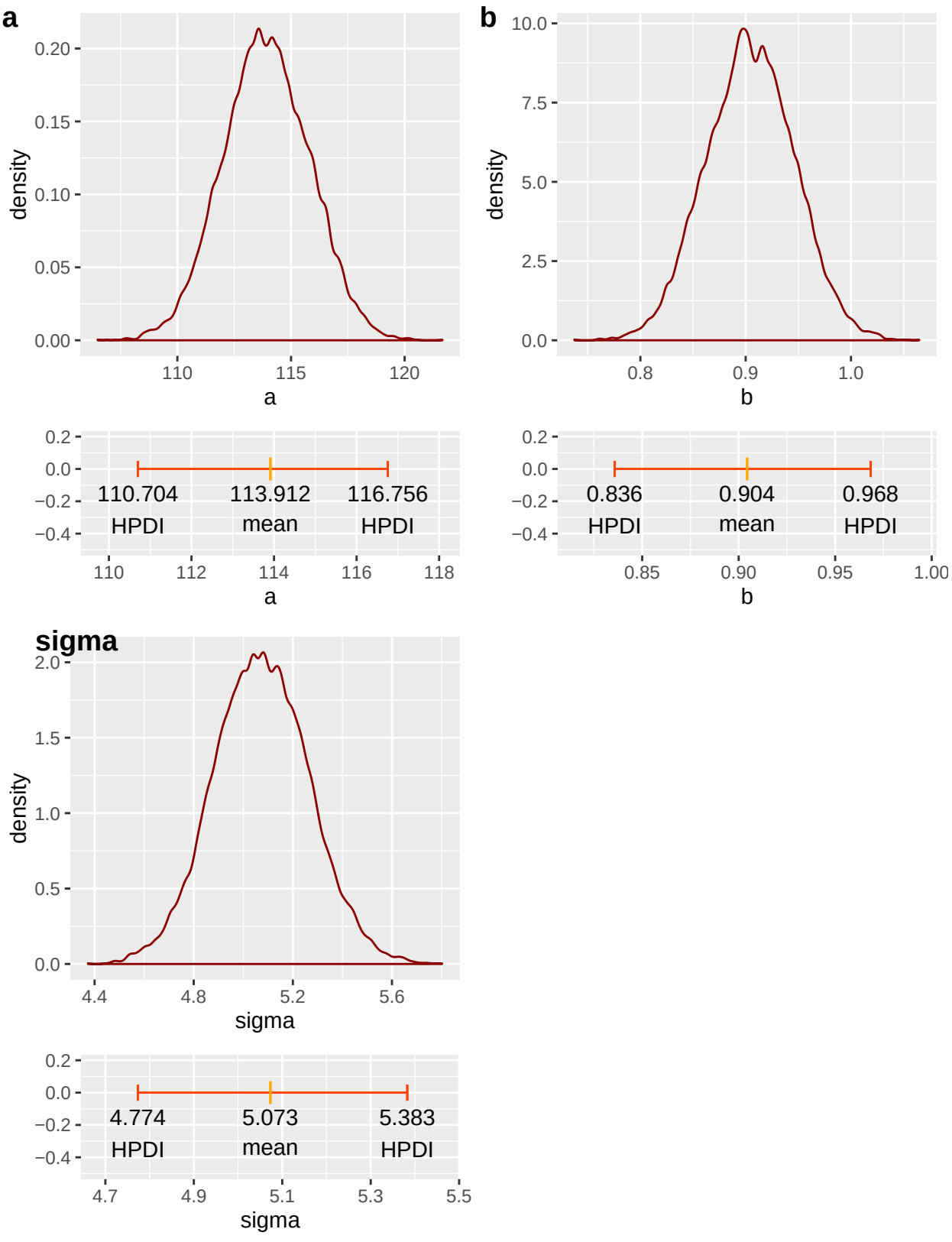
sigma

```
sigma.plot <- parameter.dist(parameter = "sigma", values = post$sigma)
sigma.plot
```



a, b, and *sigma*

```
plot_grid(a.plot, b.plot, sigma.plot,
          labels=c("a", "b", "sigma"), ncol = 2, nrow = 2)
```


a
density
0.20
0.15
0.10
0.05
0.00
110
115
120
a
0.2
0.0
−0.2
−0.4
110.704
HPDI
113.912
mean
116.756
HPDI
110
112
114
116
118
a
b
density
10.0
7.5
5.0
2.5
0.0
0.8
0.9
1.0
b
0.2
0.0
−0.2
−0.4
0.836
HPDI
0.904
mean
0.968
HPDI
0.85
0.90
0.95
1.00
b
sigma
density
2.0
1.5
1.0
0.5
0.0
4.4
4.8
5.2
5.6
sigma
0.2
0.0
−0.2
−0.4
4.774
HPDI
5.073
mean
5.383
HPDI
4.7
4.9
5.1
5.3
5.5
sigma

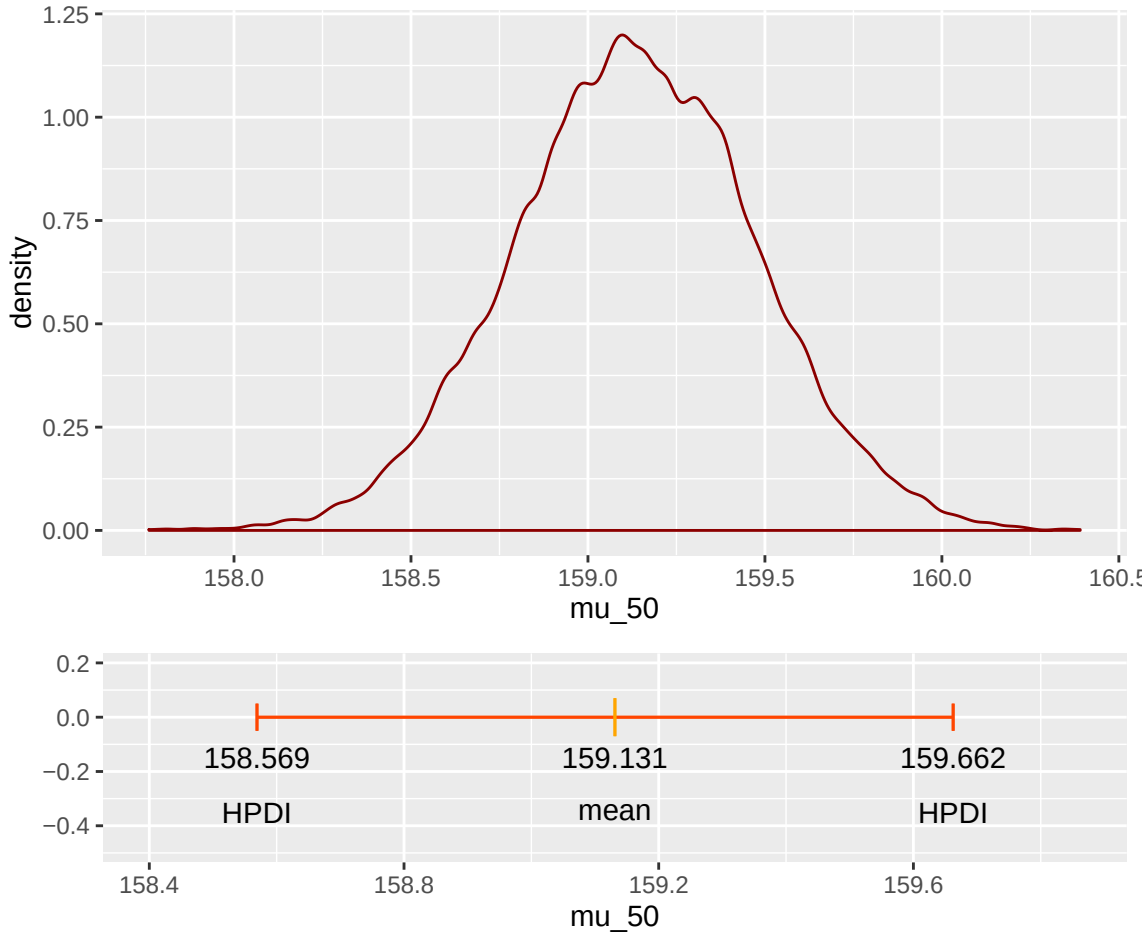
calculate μ_{50}

```
## R code 4.50
mu_at_50 <- post$a + post$b * 50
str(mu_at_50)
```

```
##  num [1:10000] 159 159 158 160 160 ...
```

view the distribution of μ_{50} with ggplot2

```
parameter <- "mu_50"
values <- mu_at_50
parameter.dist(parameter, values)
```



use link

```
## R code 4.53
mu <- link( m4.3 )
```

```
## [ 100 / 1000 ]
[ 200 / 1000 ]
[ 300 / 1000 ]
```

```
[ 400 / 1000 ]
[ 500 / 1000 ]
[ 600 / 1000 ]
[ 700 / 1000 ]
[ 800 / 1000 ]
[ 900 / 1000 ]
[ 1000 / 1000 ]
```

```
str(mu)
```

```
##  num [1:1000, 1:352] 157 157 157 157 157 ...
```

compute 89% HPDI of μ

```
## R code 4.54
# define sequence of weights to compute predictions for
# these values will be on the horizontal axis
weight.seq <- seq( from=25 , to=70 , by=1 )

# use link to compute mu
# for each sample from posterior
# and for each weight in weight.seq
mu <- link( m4.3 , data=data.frame(weight=weight.seq) )
```

```
## [ 100 / 1000 ]
[ 200 / 1000 ]
[ 300 / 1000 ]
[ 400 / 1000 ]
[ 500 / 1000 ]
[ 600 / 1000 ]
[ 700 / 1000 ]
[ 800 / 1000 ]
[ 900 / 1000 ]
[ 1000 / 1000 ]
```

```
str(mu)
```

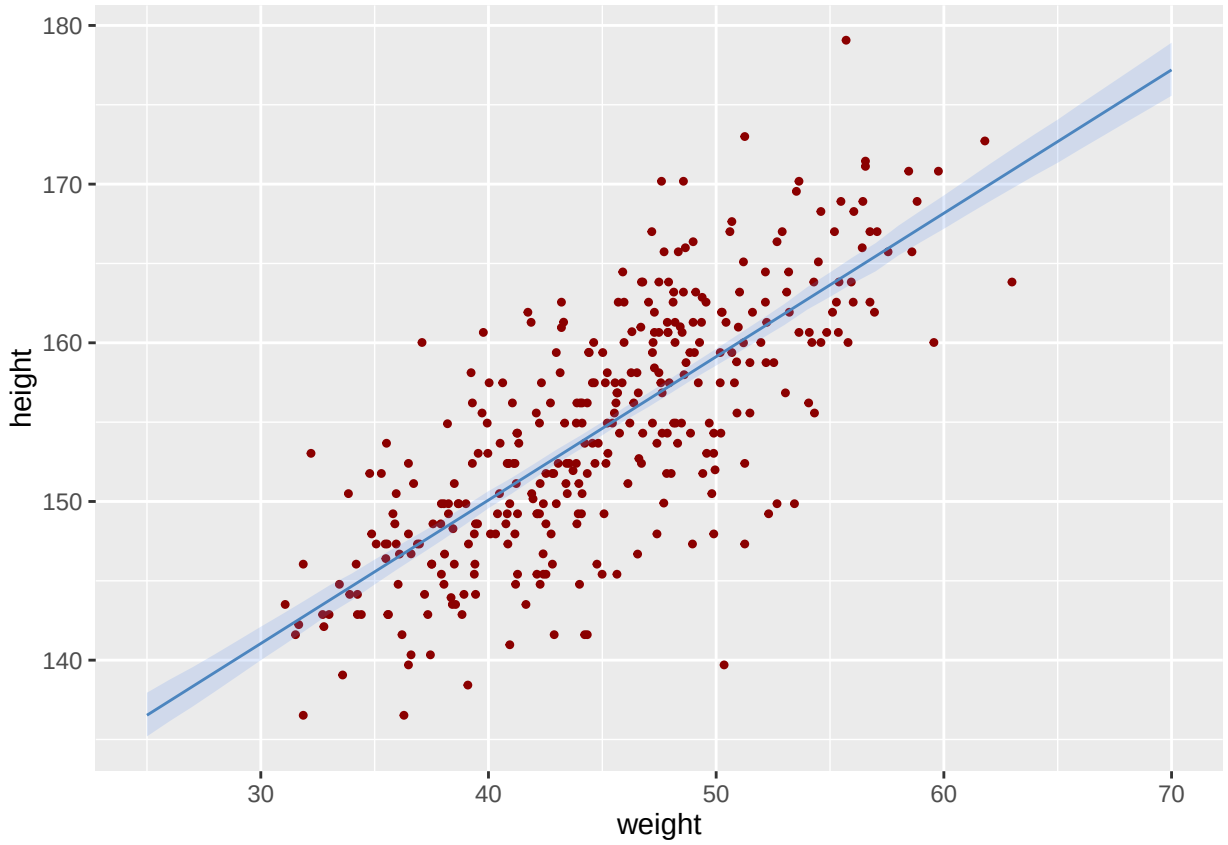
```
##  num [1:1000, 1:46] 137 136 138 137 136 ...
```

```
## R code 4.56
# summarize the distribution of mu
mu.mean <- apply( mu , 2 , mean )
mu.HPDI <- apply( mu , 2 , HPDI , prob=0.89 )
```

summarize the distribution of μ with ggplot2

```
data <- data.frame(w = weight.seq,
                   mu.mean = mu.mean,
                   mu.HPDI.lwr = mu.HPDI["|0.89", ],
                   mu.HPDI.upr = mu.HPDI["0.89|", ])
ggplot(data, aes(w, mu.mean)) +
  geom_point(data = d2, aes(weight, height), shape = 20, color = "darkred") +
  geom_line(color = "steelblue") +
  geom_ribbon(aes(ymin = mu.HPDI.lwr, ymax = mu.HPDI.upr),
```

```
            fill = "cornflowerblue", alpha = 0.2) +
  labs(x = "weight", y = "height") +
  theme_gray()
```



link algorithm

link calculates $\hat{y}$

```
## R code 4.58
post <- extract.samples(m4.3)
str(post)
```

```
## 'data.frame':    10000 obs. of  3 variables:
##  $ a    : num  114 114 114 115 114 ...
##  $ b    : num  0.898 0.908 0.886 0.861 0.895 ...
##  $ sigma: num  5 4.77 5.08 4.87 5.07 ...
```

```
mu.link <- function(weight) post$a + post$b*weight
weight.seq <- seq( from=25 , to=70 , by=1 )
mu <- sapply( weight.seq , mu.link )
str(mu)
```

```
##  num [1:10000, 1:46] 137 136 137 137 137 ...
```

```
mu.mean <- apply( mu , 2 , mean )
str(mu.mean)
```

```
##  num [1:46] 137 137 138 139 140 ...
```

```
mu.HPDI <- apply( mu , 2 , HPDI , prob=0.89 )
str(mu.HPDI)
```

```
##  num [1:2, 1:46] 135 138 136 139 137 ...
##  - attr(*, "dimnames")=List of 2
##   ..$ : chr [1:2] "|0.89" "0.89|"
##   ..$ : NULL
```

percentile intervals

```
## R code 4.62
sim.height <- sim( m4.3 , data=list(weight=weight.seq) , n=1e4 )
```

```
## [ 1000 / 10000 ]
[ 2000 / 10000 ]
[ 3000 / 10000 ]
[ 4000 / 10000 ]
[ 5000 / 10000 ]
[ 6000 / 10000 ]
[ 7000 / 10000 ]
[ 8000 / 10000 ]
[ 9000 / 10000 ]
[ 10000 / 10000 ]
```

```
str(sim.height)
```

```
##  num [1:10000, 1:46] 134 132 136 139 138 ...
```

```
height.PI <- apply( sim.height , 2 , PI , prob=0.89 )
str(height.PI)
```

```
##  num [1:2, 1:46] 128 145 129 146 130 ...
##  - attr(*, "dimnames")=List of 2
##   ..$ : chr [1:2] "5%" "94%"
##   ..$ : NULL
```

```
head(height.PI)
```

```
##         [,1]     [,2]     [,3]     [,4]     [,5]     [,6]     [,7]
## 5%  128.4340 129.1266 130.1699 130.9631 131.9219 132.9107 133.7521
## 94% 144.7233 145.7814 146.6826 147.5857 148.2953 149.3230 150.2616
##         [,8]     [,9]    [,10]    [,11]    [,12]    [,13]    [,14]
## 5%  134.6228 135.5791 136.4930 137.2698 138.3458 139.2608 139.9981
## 94% 150.7597 151.9467 152.8769 153.8578 154.5821 155.6248 156.4262
##        [,15]    [,16]    [,17]    [,18]    [,19]    [,20]    [,21]
## 5%  140.9385 142.0000 142.5864 143.9155 144.6003 145.5392 146.4773
## 94% 157.2736 158.0608 159.2621 160.0736 160.8948 161.7421 162.7083
##        [,22]    [,23]    [,24]    [,25]    [,26]    [,27]    [,28]
## 5%  147.4890 148.2830 149.1086 149.9239 151.1745 152.0686 152.7986
## 94% 163.6162 164.7431 165.4548 166.4721 167.2578 168.3626 168.9514
##        [,29]    [,30]    [,31]    [,32]    [,33]    [,34]    [,35]
## 5%  153.7504 154.7193 155.4604 156.4633 157.3483 158.2110 158.8959
## 94% 170.2124 170.9774 171.6764 172.9199 173.6821 174.4642 175.2513
##        [,36]    [,37]    [,38]    [,39]    [,40]    [,41]    [,42]
## 5%  159.9854 160.8828 161.9524 162.6885 163.5577 164.5856 165.2777
```

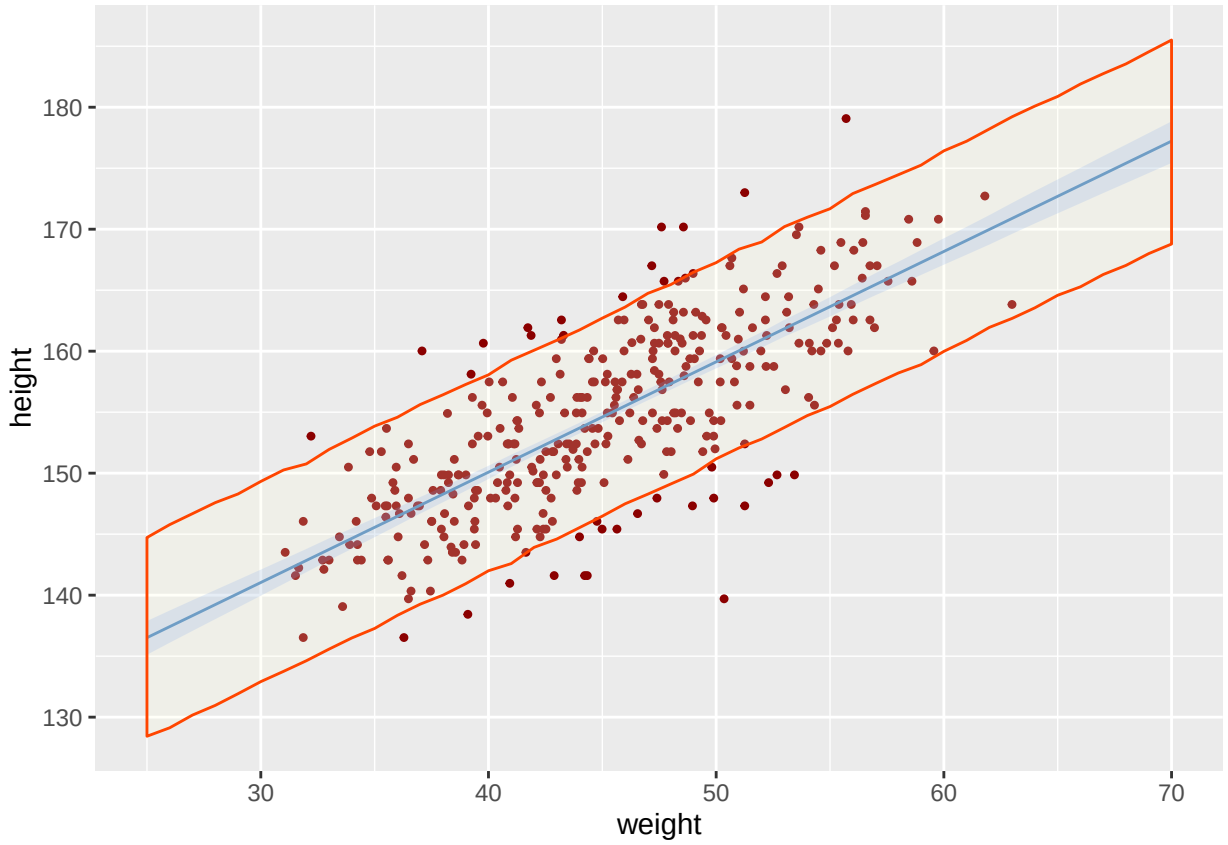
```
## 94% 176.4258 177.2142 178.2172 179.2207 180.0935 180.8835 181.9007
##        [,43]    [,44]    [,45]    [,46]
## 5%  166.2882 167.0326 168.0011 168.7834
## 94% 182.7519 183.5518 184.5415 185.5079
```

summarize percentile intervals with ggplot2

The image indicates that about 38 points are outside the 89% PI band.
There are 352 points in the data set $d2$, and (352 - 38) / 352 = 0.892.

```
data <- data.frame(w = weight.seq,
                   mu.mean = mu.mean,
                   mu.HPDI.lwr = mu.HPDI["|0.89", ],
                   mu.HPDI.upr = mu.HPDI["0.89|", ],
                   height.PI.lwr = height.PI["5%", ],
                   height.PI.upr = height.PI["94%", ])

ggplot(data, aes(w, mu.mean)) +
  geom_point(data = d2, aes(weight, height), shape = 20, color = "darkred") +
  geom_line(color = "steelblue") +
  geom_ribbon(aes(ymin = mu.HPDI.lwr, ymax = mu.HPDI.upr),
              fill = "cornflowerblue", alpha = 0.2) +
  geom_ribbon(aes(ymin = height.PI.lwr, ymax = height.PI.upr),
              color = "orangered", fill = "lightyellow", alpha = 0.2) +
  labs(x = "weight", y = "height") +
  theme_gray()
```



sim algorithm

sim calculates y

```
## R code 4.63
post <- extract.samples(m4.3)
weight.seq <- 25:70
sim.height <- sapply( weight.seq , function(weight)
    rnorm(
        n=nrow(post) ,
        mean=post$a + post$b*weight ,
        sd=post$sigma ) )
height.PI <- apply( sim.height , 2 , PI , prob=0.89 )
head(height.PI)
```

```
##        [,1]     [,2]     [,3]     [,4]     [,5]     [,6]     [,7]     [,8]
## 5%  128.320 129.1095 130.0340 131.1087 131.9342 133.0782 133.8159 134.6971
## 94% 144.813 145.6874 146.5737 147.3186 148.2850 149.2946 150.1149 150.9611
##         [,9]    [,10]    [,11]    [,12]    [,13]    [,14]    [,15]
## 5%  135.4826 136.7334 137.3390 138.3764 139.3494 140.0306 140.9762
## 94% 151.8003 152.9592 153.8681 154.6144 155.5897 156.4258 157.4189
##        [,16]    [,17]    [,18]    [,19]    [,20]    [,21]    [,22]
## 5%  142.0103 142.8746 143.8271 144.6879 145.4382 146.4070 147.5165
## 94% 158.0226 159.0148 160.1006 160.8281 161.6769 162.5938 163.3907
##        [,23]    [,24]    [,25]    [,26]    [,27]    [,28]    [,29]
## 5%  148.4292 148.9907 150.1088 151.1529 151.8846 152.8757 153.8992
## 94% 164.4611 165.5646 166.3206 167.3467 168.0801 169.1321 170.1165
##        [,30]    [,31]    [,32]    [,33]    [,34]    [,35]    [,36]
## 5%  154.5384 155.5753 156.4107 157.2985 158.1049 159.0437 159.9750
## 94% 170.9893 171.7931 172.8534 173.5888 174.3561 175.5258 176.2579
##        [,37]    [,38]    [,39]    [,40]    [,41]    [,42]    [,43]
## 5%  160.8814 162.0300 162.5882 163.6252 164.4532 165.5530 166.3306
## 94% 177.2973 178.0273 179.1853 180.0646 180.8666 181.7355 182.7949
##        [,44]    [,45]   [,46]
## 5%  167.1136 167.9276 168.843
## 94% 183.8281 184.7787 185.489
```

stan

fit the model with stan

```
# fit model
m4.3s <- map2stan(
    alist(
        height ~ dnorm( mu , sigma ) ,
        mu <- a + b*weight ,
        a ~ dnorm( 156 , 100 ) ,
        b ~ dnorm( 0 , 10 ) ,
        sigma ~ dunif( 0 , 50 )
    ) ,
    data=d2 )
```

```
##
## SAMPLING FOR MODEL 'height ~ dnorm(mu, sigma)' NOW (CHAIN 1).
```

```
##
## Chain 1, Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 1, Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 1, Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 1, Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 1, Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 1, Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 1, Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 1, Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 1, Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 1, Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 1, Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 1, Iteration: 2000 / 2000 [100%]  (Sampling)#
## #  Elapsed Time: 0.297749 seconds (Warm-up)
## #                0.292481 seconds (Sampling)
## #                0.59023 seconds (Total)
## #
##
## SAMPLING FOR MODEL 'height ~ dnorm(mu, sigma)' NOW (CHAIN 1).
##
## Chain 1, Iteration: 1 / 1 [100%]  (Sampling)#
## #  Elapsed Time: 3e-06 seconds (Warm-up)
## #                5e-05 seconds (Sampling)
## #                5.3e-05 seconds (Total)
## #

## Computing WAIC

## Constructing posterior predictions

## [ 100 / 1000 ]
[ 200 / 1000 ]
[ 300 / 1000 ]
[ 400 / 1000 ]
[ 500 / 1000 ]
[ 600 / 1000 ]
[ 700 / 1000 ]
[ 800 / 1000 ]
[ 900 / 1000 ]
[ 1000 / 1000 ]
```

```
## R code 4.40
precis( m4.3s )
```

```
##         Mean StdDev lower 0.89 upper 0.89 n_eff Rhat
## a     113.73   1.86     111.02     117.05   237    1
## b       0.91   0.04       0.83       0.97   237    1
## sigma   5.11   0.19       4.82       5.40   404    1
```

view distributions of model parameters with ggplot2

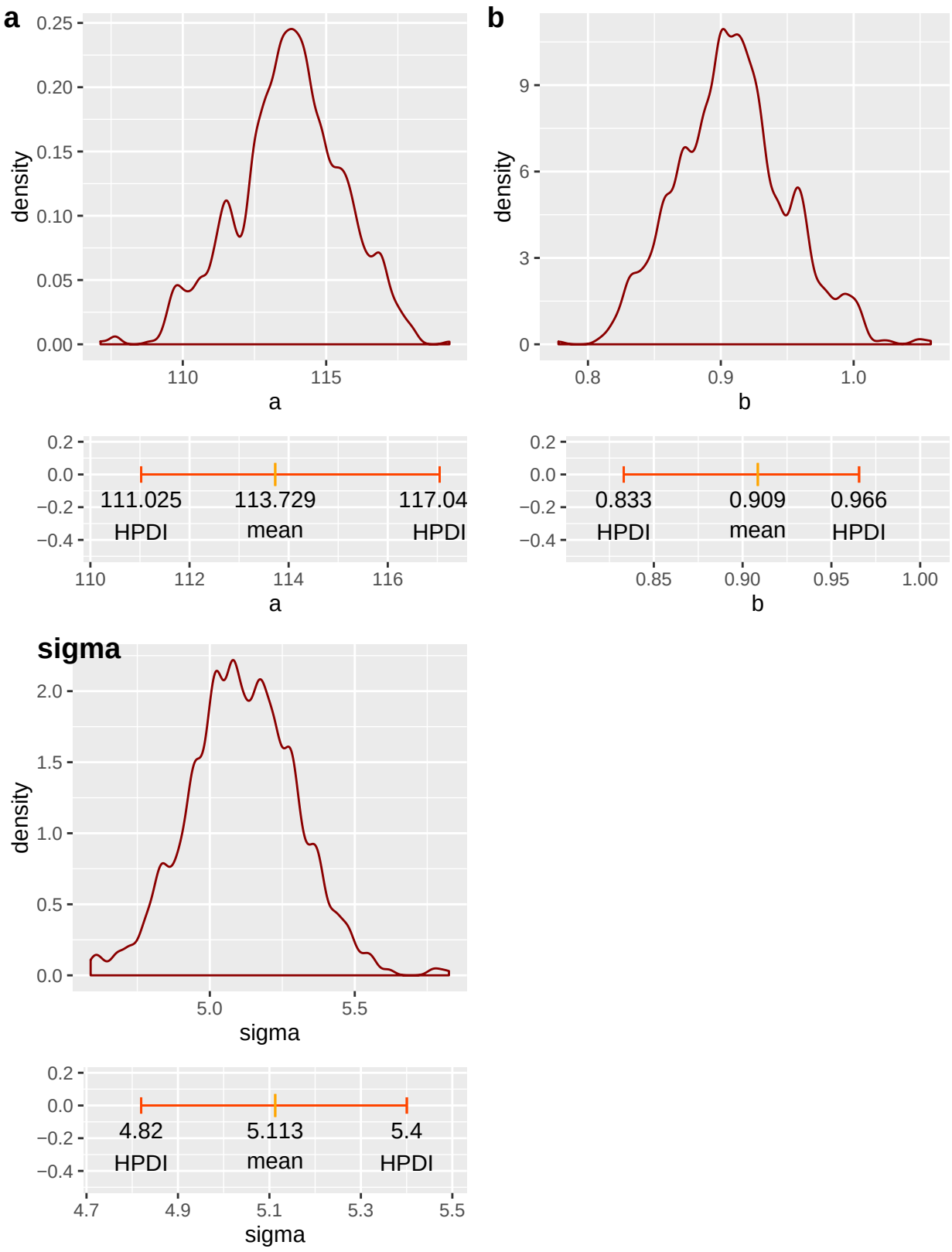
```
## R code 4.46
post <- extract.samples( m4.3s )
str(post)
```

```
## List of 3
##  $ a    : num [1:1000(1d)] 111 116 113 114 113 ...
##  $ b    : num [1:1000(1d)] 0.965 0.884 0.925 0.912 0.919 ...
##  $ sigma: num [1:1000(1d)] 5.1 4.99 5.07 5.17 5.12 ...
```

$a, b,$ and $sigma$

```
a.plot <- parameter.dist(parameter = "a", values = post$a)
b.plot <- parameter.dist(parameter = "b", values = post$b)
sigma.plot <- parameter.dist(parameter = "sigma", values = post$sigma)
```

```
plot_grid(a.plot, b.plot, sigma.plot,
          labels=c("a", "b", "sigma"), ncol = 2, nrow = 2)
```


a
density
0.25
0.20
0.15
0.10
0.05
0.00
110
115
a
0.2
0.0
−0.2
−0.4
111.025
HPDI
113.729
mean
117.04
HPDI
110
112
114
116
a
b
density
9
6
3
0
0.8
0.9
1.0
b
0.2
0.0
−0.2
−0.4
0.833
HPDI
0.909
mean
0.966
HPDI
0.85
0.90
0.95
1.00
b
sigma
density
2.0
1.5
1.0
0.5
0.0
5.0
5.5
sigma
0.2
0.0
−0.2
−0.4
4.82
HPDI
5.113
mean
5.4
HPDI
4.7
4.9
5.1
5.3
5.5
sigma