Linear Model Walkthrough

Adam Fleischhacker

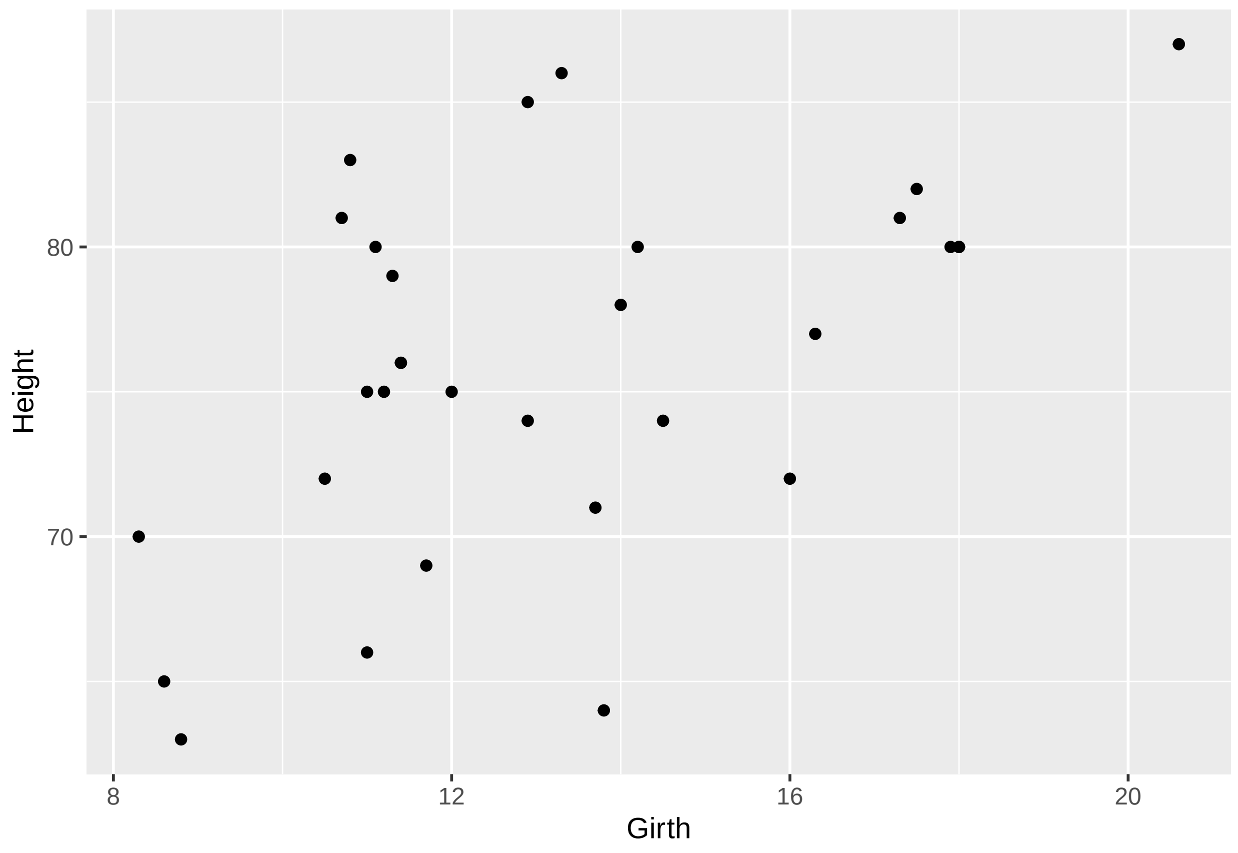
7/5/2019

Intro to Prediction of Tree Height

This walkthrough closely follows the lecture of Richard McElreath. We will use cherry tree heights instead of human heights. See this video for more info: https://www.youtube.com/watch?v=h5aPo5wXN8E&t=0s

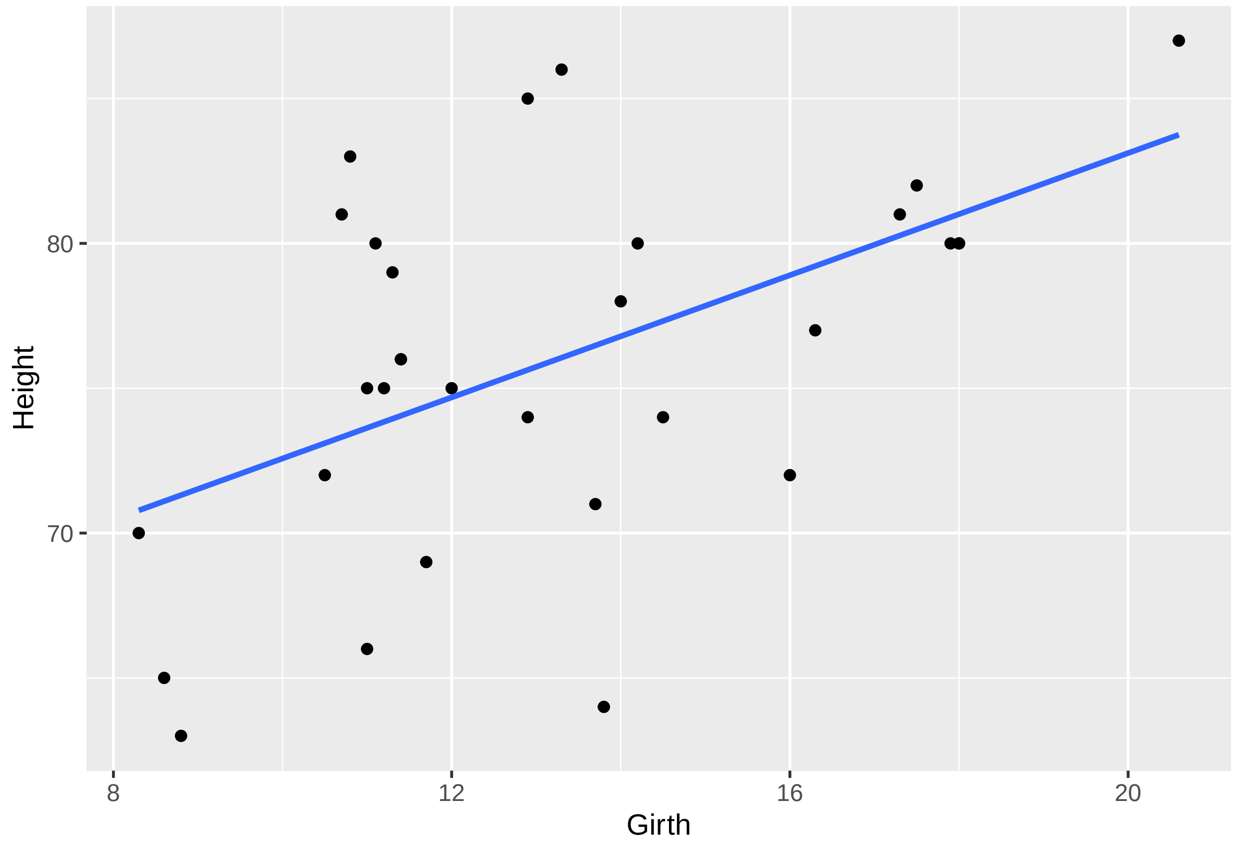
Here, is a graph of height versus tree girth for the cherry tree dataset built into R. Notice how if you know the girth of a tree, your prediction for height will change.

```r
library(tidyverse)
library(greta)
library(causact)

treesDF = trees %>% as_tibble()

# look at data
treesDF %>%
  ggplot(aes(x = Girth, y = Height)) +
  geom_point()
```



In a past course, you might have characterized how your prediction might change using a linear regression. It would return the so-called best line as shown here in blue:

```r
treesDF %>%
  ggplot(aes(x = Girth, y = Height)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE)
```

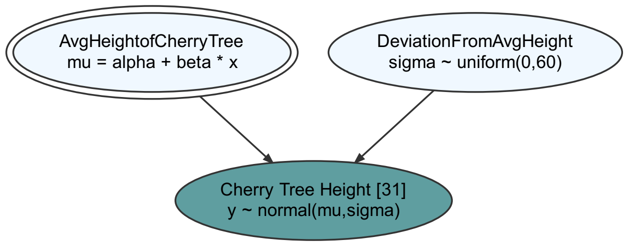


Limiting yourself to just one line can be a mistake with such little data for any given girth. Hence, in Bayesian analysis, we allocate plausibility over all the lines we can dream of.

Graphical Model for All the Lines

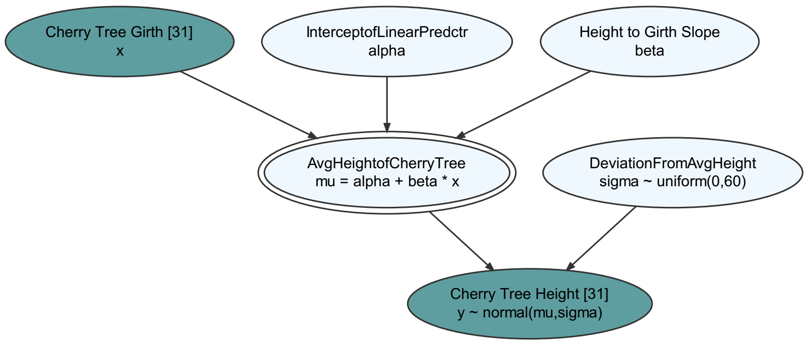
Let's use a graphical model to create our recipe for generating plausible lines. Here is the start:

```r
graph = dag_create() %>%
  dag_node("Cherry Tree Height","y",
           rhs = normal(mu,sigma),
           data = treesDF$Height) %>%
  dag_node("Avg Height of Cherry Tree","mu",
           child = "y",
           rhs = alpha + beta * x) %>%
  dag_node("Deviation From Avg Height","sigma",
           rhs = uniform(0,60),
           child = "y")
graph %>% dag_render()
```



We then realize, that our line equation has unknown variables on the right-hand side(rhs), so we add in some parents to describe our knowledge about plausible relationships:

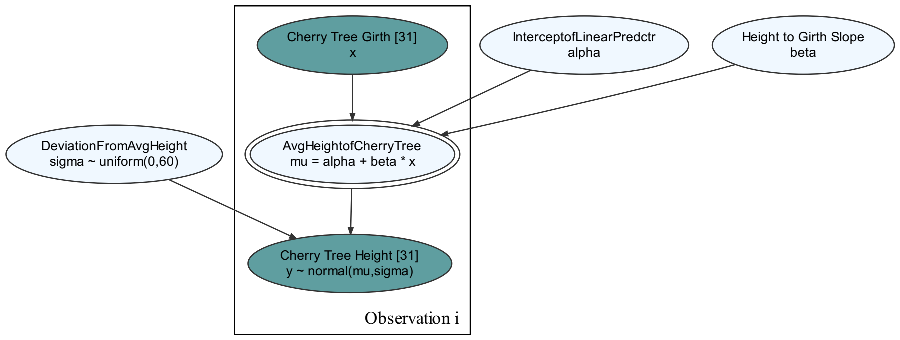
```
### add parents
graph2 = graph  %>%
  dag_node("Cherry Tree Girth","x",
           child = "mu",
           data = treesDF$Girth) %>%
  dag_node("Intercept of Linear Predictor","alpha",
           child = "mu") %>%
  dag_node("Height to Girth Slope","beta",
           child = "mu")
graph2 %>% dag_render()
```



Each oval in the graph represents one random variable. For example, there is one random variable representing the intercept of the linear predictor. For cherry tree height and girth though, we have 31 observations (i.e. see the [31] next to thsoe two node labels). At this point, we should ask ourselves, do we want each prediction of cherry tree height to have the same parameter `mu`? The answer is no, in fact, we want `mu` to vary based on the `girth` of the observation. Hence, we use plate notation to indicate that there is more than 1 `mu`, rather there should be 31 `mu` values; one for each oberved girth.

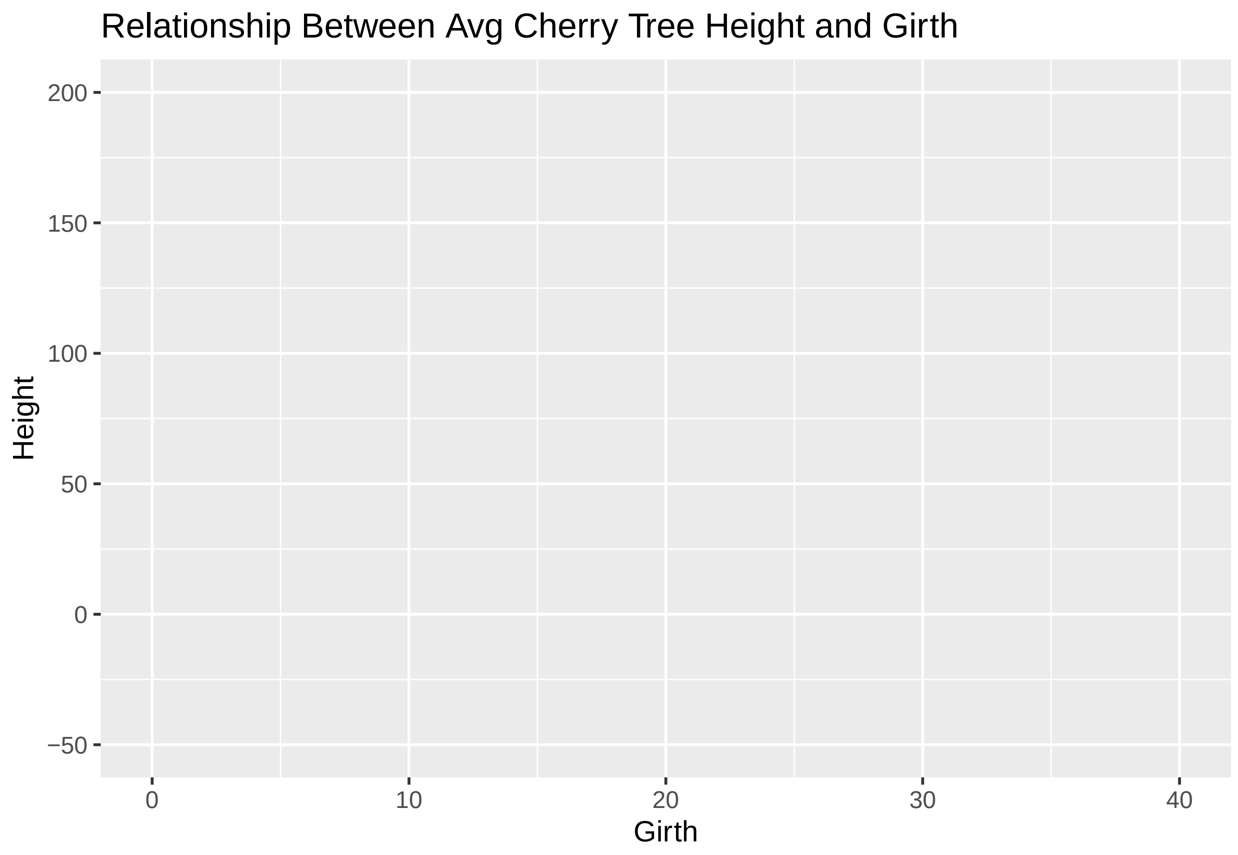
Here is the updated graphical model that makes this point clear:

```
### add plate for clarity
graph3 = graph2 %>%
  dag_plate("Observation","i",
            nodeLabels = c("y","mu","x"))
graph3 %>% dag_render()
```



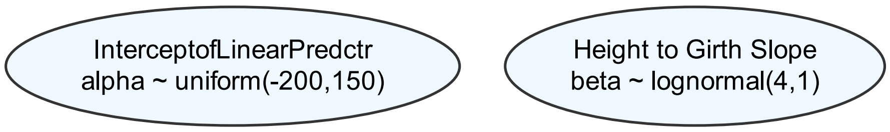
Notice, the only thing making this graphical model incomplete is the absence of prior for the two unobserved nodes, `alpha` and `beta` (observed nodes do not need priors unless you are modelling their generative process). We want to choose these so that only plausible lines are considered. Draw the plausible lines below and consider their equations:

```
### this is our skelton recipe for a line
### we just need to narrow down the list of
### possible alphas and beta values
### let's look at reasonable lines
treesDF %>%
  ggplot(aes(x = Girth, y = Height)) +
  xlim(0,40) + ylim(-50,200) +
  ggtitle("Relationship Between Avg Cherry Tree Height and Girth")
```



The best way to find good priors is to pick some and see their implications using simulation. Here is a first attempt (note: the `lognormal` distribution was chosen as it is strictly positive and often used to model uncertainty in standard deviation):

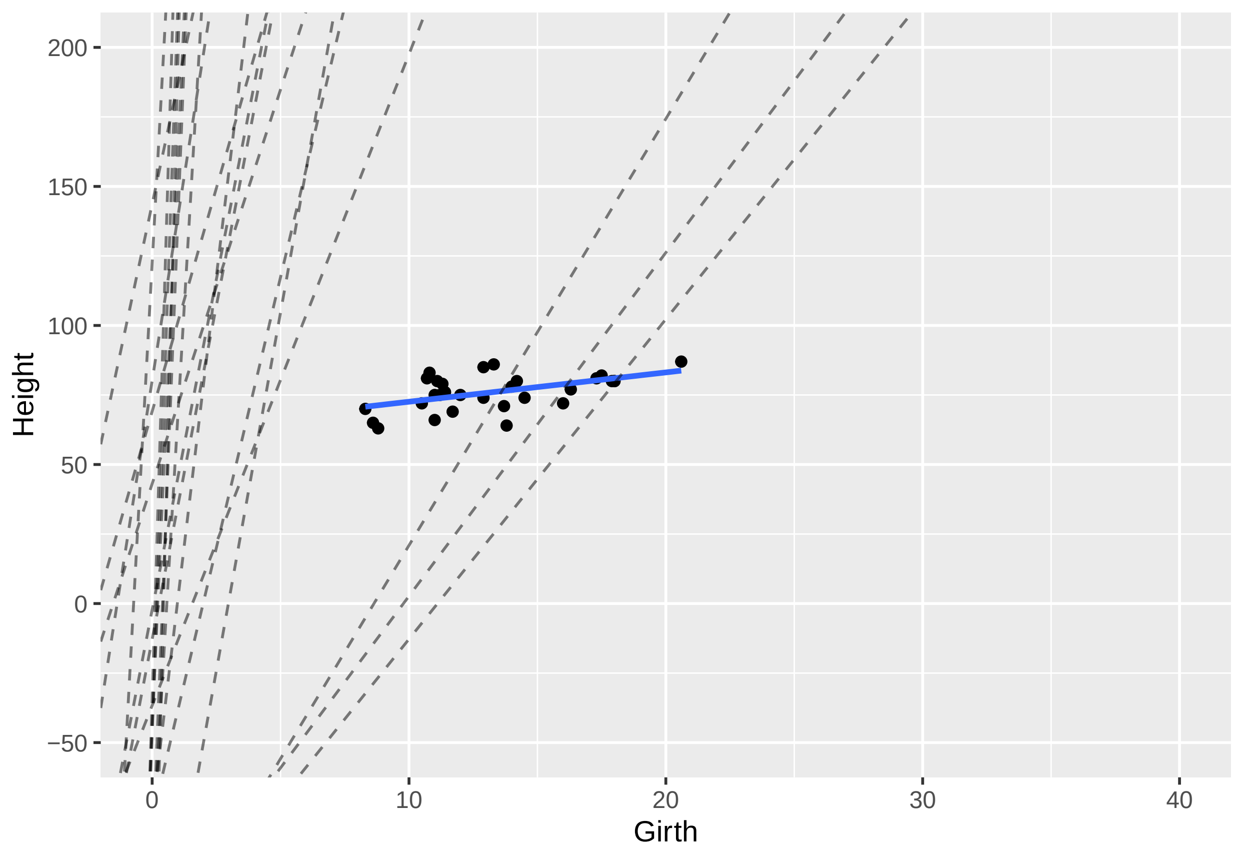
```
### simulate some average lines
priorGraph = dag_create() %>%
  dag_node("Intercept of Linear Predictor","alpha",
           rhs = uniform(-200,150)) %>%
  dag_node("Height to Girth Slope","beta",
           lognormal(4,1))
priorGraph %>% dag_render()
```



```r
priorGraph %>%
  dag_greta(mcmc=TRUE, warmup = 100)
```

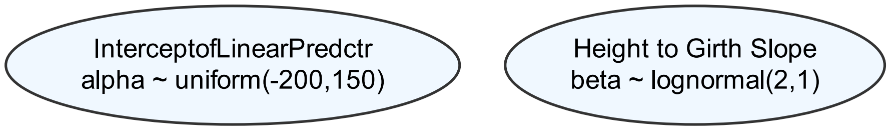
```
## ## The specified DAG corresponds to the following greta code:
## alpha  <- uniform(min = -200, max = 150)      #PRIOR
## beta   <- lognormal(meanlog = 4, sdlog = 1)   #PRIOR
## gretaModel <- model(alpha,beta)   #MODEL
## draws       <- mcmc(gretaModel,warmup = 100)   #POSTERIOR
## draws       <- replaceLabels(draws)   #POSTERIOR
## drawsDF     <- draws %>% as.matrix() %>% dplyr::as_tibble()   #POSTERIOR
## tidyDrawsDF <- drawsDF %>% tidyr::gather() %>%
##     addPriorGroups()   #POSTERIOR
```

```r
## draw 20 posterior samples
set.seed(123)
tempDF = drawsDF %>% sample_n(20)

## visualize 20 plausible lines from posterior
treesDF %>%
  ggplot(aes(x = Girth, y = Height)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE) +
  geom_abline(data=tempDF, aes(intercept = alpha, slope = beta), alpha = 0.5, linetype = "dashed")  +
  xlim(0,40) + ylim(-50,200)
```



In the above case, some of the so-called plausible prior lines seem to have a slope that is way too big and unrealistic. After some playing around with the priors, I found these priors to be more realistic:

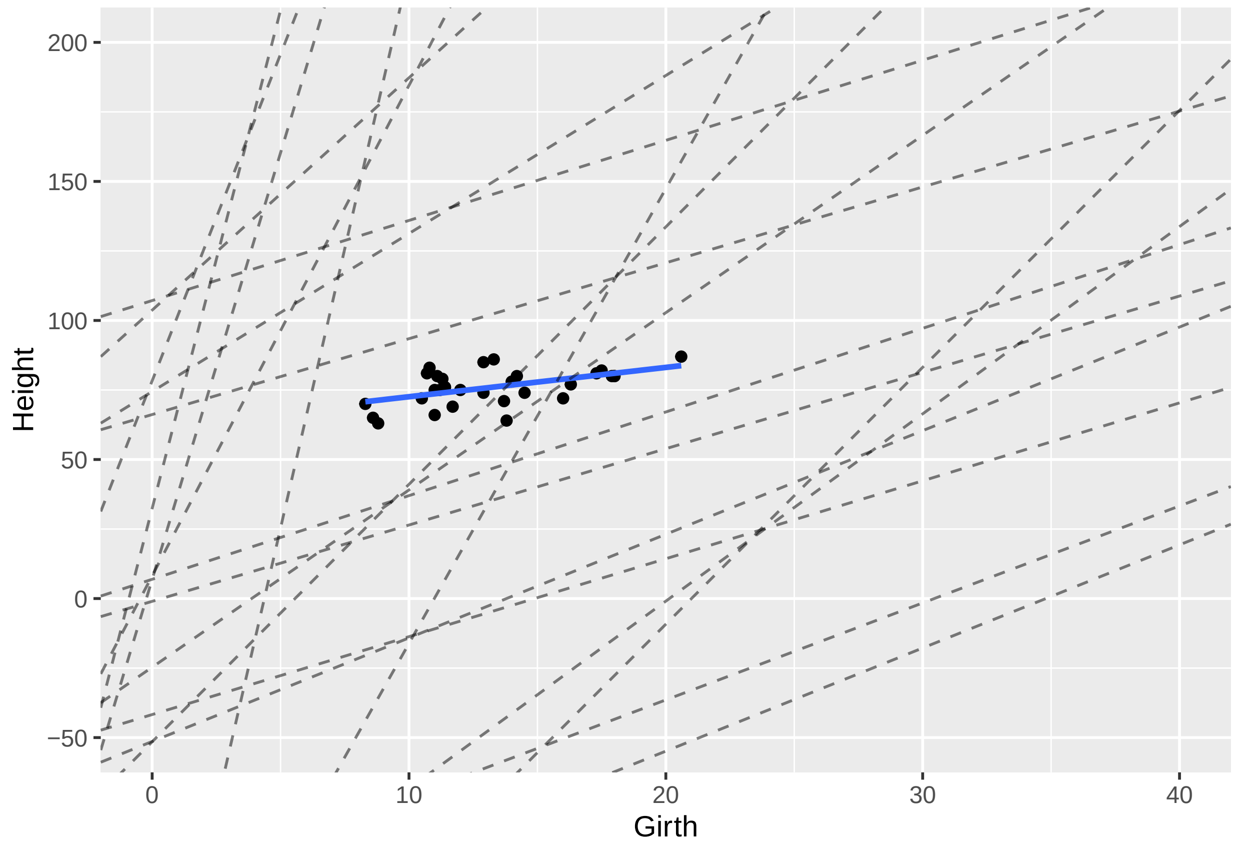
```
### simulate some average lines
priorGraph = dag_create() %>%
  dag_node("Intercept of Linear Predictor","alpha",
           rhs = uniform(-200,150)) %>%
  dag_node("Height to Girth Slope","beta",
           lognormal(2,1))
priorGraph %>% dag_render()
```



```r
priorGraph %>%
  dag_greta(mcmc=TRUE, warmup = 100)
```

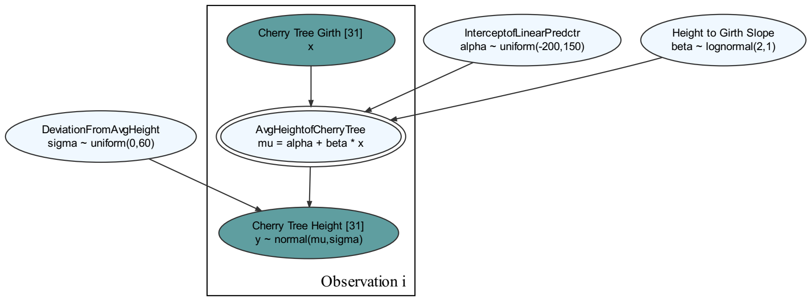
```
## ## The specified DAG corresponds to the following greta code:
## alpha  <- uniform(min = -200, max = 150)      #PRIOR
## beta   <- lognormal(meanlog = 2, sdlog = 1)   #PRIOR
## gretaModel <- model(alpha,beta)   #MODEL
## draws       <- mcmc(gretaModel,warmup = 100)   #POSTERIOR
## draws       <- replaceLabels(draws)   #POSTERIOR
## drawsDF     <- draws %>% as.matrix() %>% dplyr::as_tibble()   #POSTERIOR
## tidyDrawsDF <- drawsDF %>% tidyr::gather() %>%
##     addPriorGroups()   #POSTERIOR
```

```r
tempDF = drawsDF %>% sample_n(20)

treesDF %>%
  ggplot(aes(x = Girth, y = Height)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE) +
  geom_abline(data=tempDF, aes(intercept = alpha, slope = beta), alpha = 0.5, linetype = "dashed")  +
  xlim(0,40) + ylim(-50,200)
```



That looks much better. We now update our graphical model with the new prior on slope:

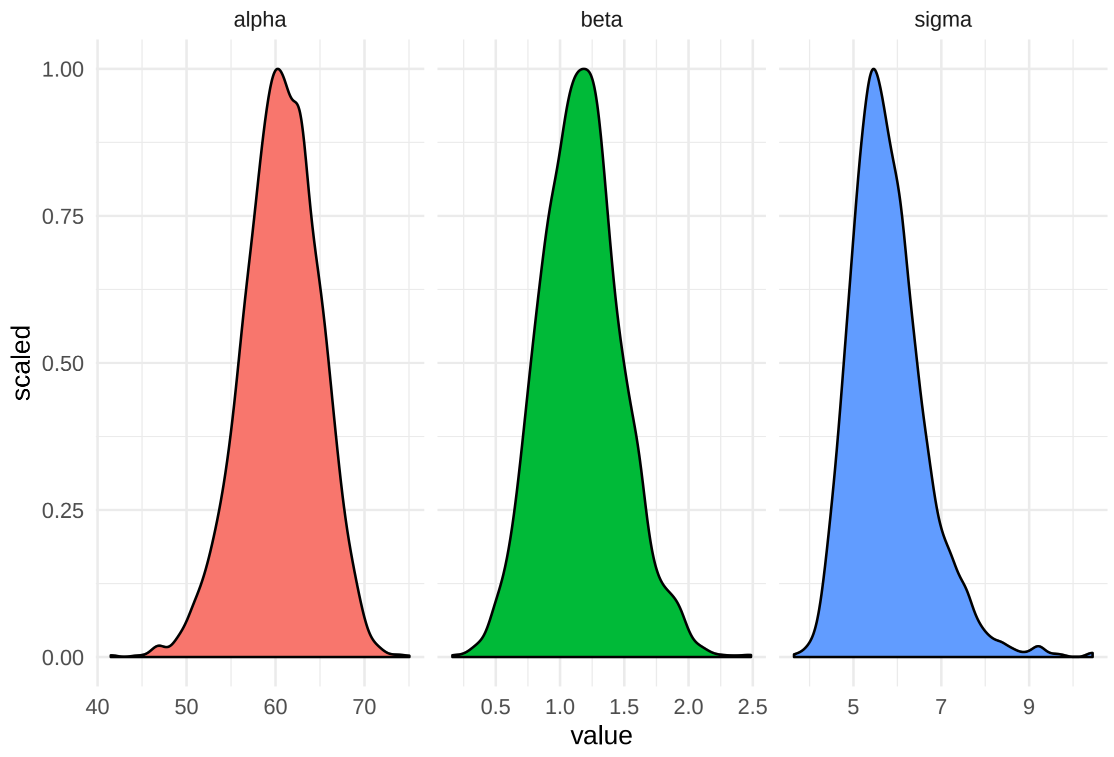
```
graph6 = dag_create() %>%
  dag_node("Cherry Tree Height","y",
           rhs = normal(mu,sigma),
           data = treesDF$Height) %>%
  dag_node("Avg Height of Cherry Tree","mu",
           child = "y",
           rhs = alpha + beta * x) %>%
  dag_node("Cherry Tree Girth","x",
           child = "mu",
           data = treesDF$Girth) %>%
  dag_node("Deviation From Avg Height","sigma",
           rhs = uniform(0,60),
           child = "y")  %>%
  dag_node("Intercept of Linear Predictor","alpha",
           child = "mu",
           rhs = uniform(-200,150)) %>%
  dag_node("Height to Girth Slope","beta",
           child = "mu",
           lognormal(2,1))  %>%  # new line
  dag_plate("Observation","i",
            nodeLabels = c("y","mu","x"))

graph6 %>% dag_render()
```



```
## use greta
graph6 %>%
  dag_greta(mcmc=TRUE)
```

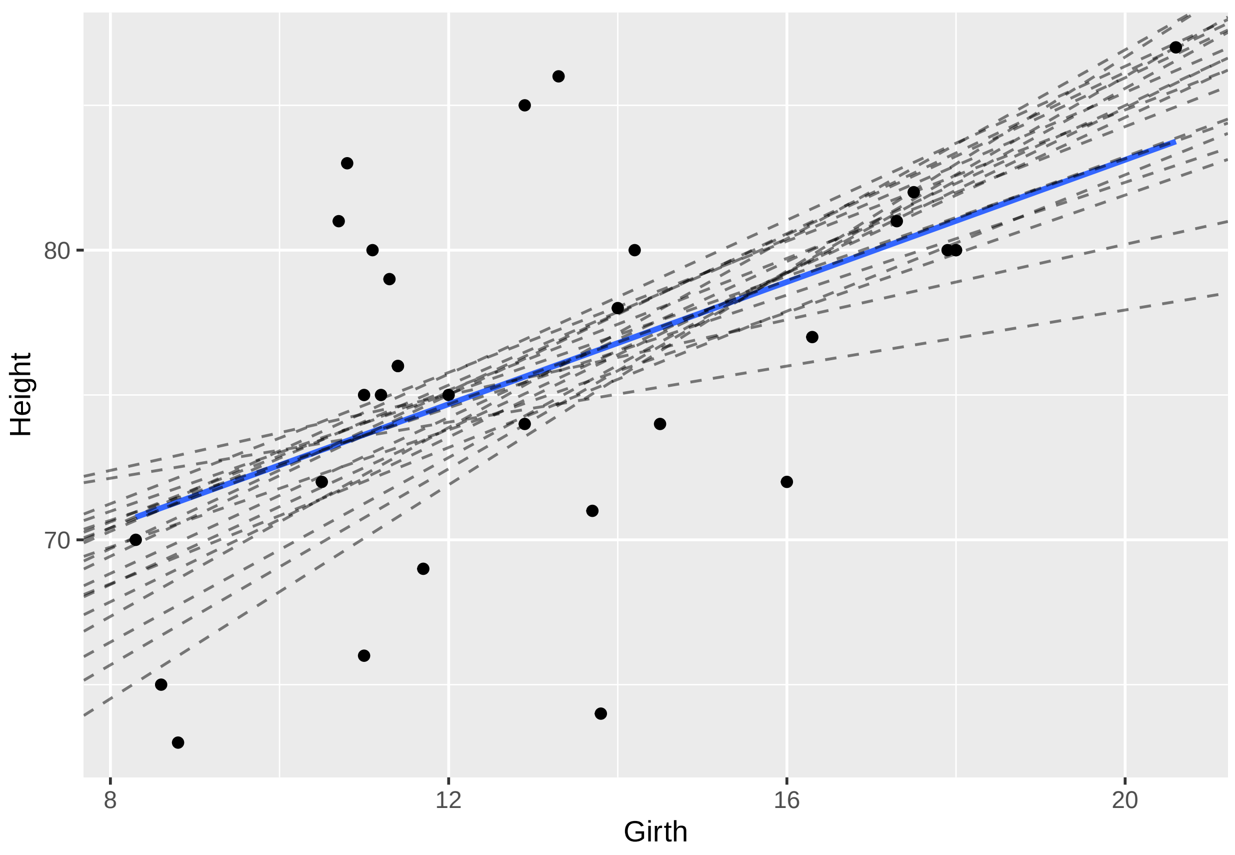
```
## ## The specified DAG corresponds to the following greta code:
## x <- as_data(treesDF$Girth)    #DATA
## y <- as_data(treesDF$Height)   #DATA
## sigma  <- uniform(min = 0, max = 60)          #PRIOR
## alpha  <- uniform(min = -200, max = 150)      #PRIOR
## beta   <- lognormal(meanlog = 2, sdlog = 1)   #PRIOR
## mu     <- alpha + beta * x   #OPERATION
## distribution(y) <- normal(mean = mu, sd = sigma)   #LIKELIHOOD
## gretaModel <- model(sigma,alpha,beta)   #MODEL
## draws       <- mcmc(gretaModel)   #POSTERIOR
## draws       <- replaceLabels(draws)   #POSTERIOR
## drawsDF     <- draws %>% as.matrix() %>% dplyr::as_tibble()   #POSTERIOR
## tidyDrawsDF <- drawsDF %>% tidyr::gather() %>%
##     addPriorGroups()   #POSTERIOR
```

```
## posterior
drawsDF %>% dagp_plot()
```



Now, we conclude with a posterior predictive check to make sure that the plausible lines generate by our process can at least explain the data we have seen:

```
## posterior predictive check
tempDF = drawsDF %>% sample_n(20)

treesDF %>%
  ggplot(aes(x = Girth, y = Height)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE) +
  geom_abline(data=tempDF, aes(intercept = alpha, slope = beta), alpha = 0.5, linetype = "dashed")
```


Height
80
70
8
12
16
20
Girth