Causality Intro

Causaility

We use explanatory models to quantitatively describe the outcomes of business activity. Analysts seek insight into future performance for planning, while assurance professionals seek explanations for historic performance - but the models are the same.

Inevitably, we arrive at the question of causation. There may be a strong correlation between a business driver and outcome, but does the driver **cause** the outcome? If not, we can't manage it to affect future performance, nor use it to validate past performance. There may be a strong correlation between Nicholas Cage films and business activity, but will getting Nicholas film contracts increase your clients revenues?

Causation often seems like an ambiguous question *to begin with*. Does an increase in credit card accounts cause an increase in Amazon sales? Does a shortage in steel cause an decrease in automobile margins? These may seem like hard questions to answer, but if you're an assurance professional, you'll write *"margin decrease is explained by. . . "* in your audit report. And if you're an analyst you'll recommend *"adding resources to deal with increasing revenues due to. . . "* So, we need to understand how these statements can be supported.

The process for establishing causation is two-step:

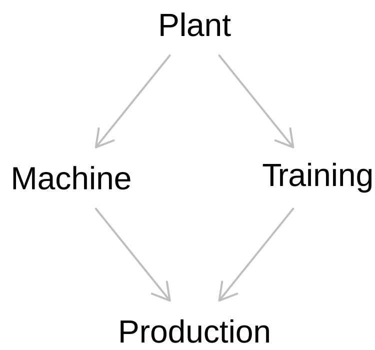
1. We model causal relationships using a structured DAG *(Directed Acyclic Graph)*. This model can be *communicated* with a freehand drawing - but it is a structured, rational analysis *(your Pearl text does an excellent job of presenting DAGs in Chapter 2)*.
2. We then test and confirm these relationships and refine our model - this is an iterative process. The tests measure outcomes as distributions, conditioned on the dimension values.

Simply put, the relationship can be structured in a DAG model *(as chains, forks and/or colliders)*, *AND* $P(outcome|X = x_1) <> P(outcome|X = x_2)$, then it's a causal relationship. This is an iterative process, beginning with ambiguous, but rational, beliefs - gradually disambiguating and quantifying until you've reached conclusion. Let's work through an example.

Data

In this exercise, we'll walk through example data using a DAG using the dagitty package.

```
library(dagitty)

g1 = dagitty('dag {

  Plant [pos = "2,1"]
  Machine [pos = "1,2"]
  Training [pos = "3,2"]
  Production [pos = "2,3"]

Plant -> Machine -> Production
Plant -> Training -> Production

             }')
plot(g1)
```



The DAG structure above, when combined with data, enables us to predict quantitatively the results of interventions without actually performing them *(Pearl pg 35)*.

> Intervention is often expensive, and dangerous *(e.g., giving a vaccine to people)*. In most business scenarios, we're not worried about people dying, but we do worry about other costs *(in this scenario, purchasing, or moving production machines)*.

In this exercise, we want to know how machines and training will affect production *without* intervention, so we estimate based on observation.

An intervention is **doing** something *(Pearl's "do" calculus)*, which often has a significant **effect**. Determining cause involves quantification of this **effect** through **Do Calculus**, which compares probability across intervention *do* conditions, or states - i.e., *(using integration - i.e., calculus)*. Fortunately, Bayesian modeling provides a valid method for measuring effects across multilevel conditions.

So, lets get started. We'll create data to reflect the 2 level graph above. We're building a model that can predict production for a given labor pool of hours. Beyond labor hours, production is affected by:

> **Plant** Different plants have different inefficiencies. **PEffect** *(which we don't usually know, but control for here)*, is the quantitative effect of those differences. Why? Different plants have different layouts, space, machine mix *(which we will measure separately)*, labor scheduling, decision-making, and overall culture. This goes beyond manufacturing - you will find the same effects and drivers in service industries too. I once did an analysis at GE that found our highest labor cost facility *(France)* was actually our most profitable because of culture *(they were very diligent about quoting, scheduling, and maintenance)*. Culture, in all it's ambiguity, makes a big difference and not considering it, in this context, would be incompetent.
>
> **Machine Type** The **MEffect** measures the effect of different machine types - i.e., given labor input in hours, different machines will produce different outputs. Note that different plants have different machine types *(based on management decisions and budget)*. Normally, you wouldn't have only one instance of a machine type in a plant, but we're trying to keep is simple so you can see how the analysis works. But just FYI, machines of the same type will vary based on maintenance and operators, so you often have to go down to the individual machine ID, and also account for the interaction effects of labor skill and experience, which brings up Training.
>
> **Training** The **TEffect** reflects the impact of a managment training program, which should increase productivity. This is at the discretion of plant managers, and involves a cost. So, some plants executed a training program others did not.

Note that in the DAG above, training and machines are both "caused" by plants. Plant management decides on training and machines, and management differs across plants - this is often the case in business - CULTURE is often a cause and it's not undefined, fuzzy-wuzzy stuff. Here, plants also have budgets and have to decide how to allocate to machines or training *(you can't have it all in this game)*. This is a **fork**, followed by a **chain** and then a **collider** into Production. So, let's generate the data:

```r
Plant = data.frame(PlantID = c(1, 2, 3),
                   PBudget = c(100, 100, 75),
                   PEffect = c( 25, 25, 15))
Machine = data.frame(MachineType = c(1, 2, 3, 4),
                   MCost = c( 50,40, 35, 25),
                   MEffect = c( 45, 40, 35, 25))
Training = data.frame(TrainingID = c(1),
                      TCost = c(5),
                      TEffect = c(10))

Mu = 10
MHrs = 10
N = 800
WOrders = 800

# each observation is a work order (which authorizes labor / machine scheduling)

Data = data.frame(PlantID  = c(1, 1, 1, 2, 2, 3, 3, 3),
                  MachineType = c(1, 2, 3, 2, 3, 2, 3, 4),
                  Hrs = seq(1:MHrs), Production = seq(1:WOrders))

Data = Data %>% inner_join(Plant, by = "PlantID") %>% inner_join(Machine, by = "MachineType")
Data = Data %>% mutate(TrainingID= if_else(PlantID == 2, 1, 0), TEffect = if_else(PlantID == 2, 20, 0))


Data = Data %>% mutate(Hrs = sample(seq(from = 10, to =40, by = 1), nrow(Data), replace = TRUE),
                       Production =  Mu + (Hrs*(PEffect+MEffect+TEffect)))




# add randomness
Data$Production = Data$Production + rnorm(Data$Production,  0 , 500)

# prep or modeling
Data$Plant = factor(Data$PlantID)
Data$Machine = factor(Data$MachineType)
Data$Training = factor(Data$TrainingID)


# summarize
Metrics = Data %>% group_by(PlantID, MachineType, TrainingID) %>%
  summarise(Location = round(mean(Production),0), Scale = round(sd(Production),0))

knitr::kable(Metrics, caption = "Data Summary") %>%
  kable_styling(full_width = F, bootstrap_options = "striped", font_size = 9)
```

And let's check correlations:

> Something to keep in mind: in many scenarios where business DRIVERS are being analyzed, correlations can be quantitatively small - really small - especially in cases where there is a long time lag *(e.g., steel prices => automobile profit margins)*. Just becuase they're small doesn't mean they're unimportant - in fact, they may be the key to industry dominance and profit margins. Use your mind, not your spreadsheet.

Table 1: Data Summary

PlantID	MachineType	TrainingID	Location	Scale
1	1	0	1728	794
1	2	0	1567	796
1	3	0	1417	768
2	2	1	2119	928
2	3	1	2013	883
3	2	0	1327	603
3	3	0	1166	599
3	4	0	1105	600

Table 2: Correlation

	PlantID	PBudget	MachineType	MCost	TrainingID	Production
PlantID	1.00	-0.89	0.50	-0.55	0.00	-0.19
PBudget	-0.89	1.00	-0.45	0.49	0.45	0.33
MachineType	0.50	-0.45	1.00	-0.98	0.00	-0.20
MCost	-0.55	0.49	-0.98	1.00	0.00	0.20
TrainingID	0.00	0.45	0.00	0.00	1.00	0.36
Production	-0.19	0.33	-0.20	0.20	0.36	1.00

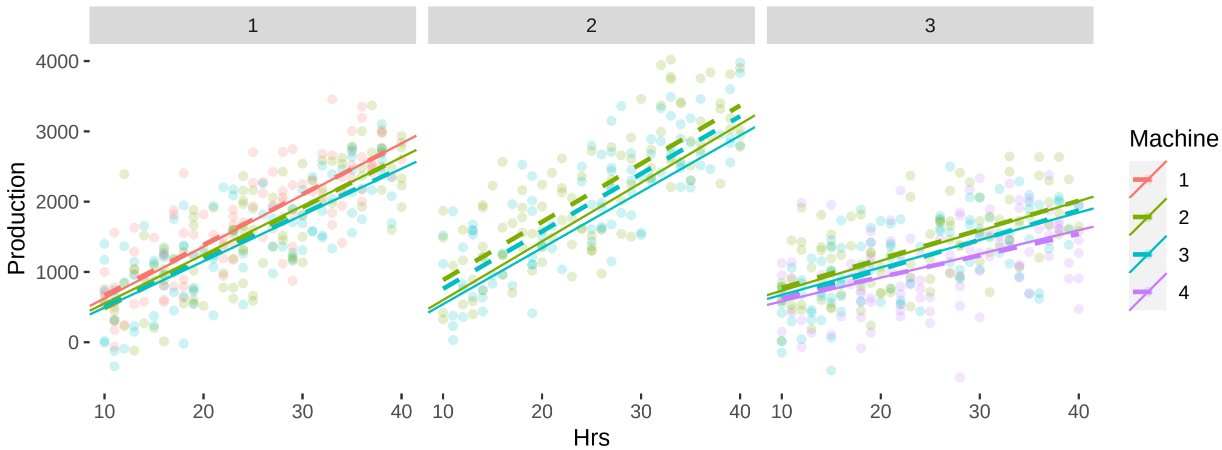
```
# check correlations
cData = round(cor(data.matrix(select(Data, PlantID, PBudget, MachineType, MCost, TrainingID, Production

knitr::kable(cData, caption = "Correlation") %>%
  kable_styling(full_width = F, bootstrap_options = "striped", font_size = 9)
```

Initial Models

Now, we'll run build some lmer models *(and add lm models in the plot)* for baseline:

```
# run quick lmer for orientation

HomeMod = lmer(Production ~ Hrs + Training + ( Hrs | Plant) + (Hrs | Machine) + (Hrs | TrainingID),
               data = Data)

# this will result in sigular fit bc small covariance

d1 = ranef(HomeMod)$Plant %>% rownames_to_column("Plant") %>% select(Plant, "I1" = "(Intercept)", "S1"
d2 = ranef(HomeMod)$Machine %>% rownames_to_column("Machine") %>% select(Machine, "I2" = "(Intercept)",
d3 = ranef(HomeMod)$Training %>% rownames_to_column("Training") %>% select(Training, "I3" = "(Intercept

I = fixef(HomeMod)[1]
S = fixef(HomeMod)[2]

Coef = crossing(d1, d2, d3) %>% mutate(Intercept = I + I1 + I2 + I3, Slope = S + S1 + S2 + S3)
Coef$Plant = factor(Coef$Plant)
Coef$Machine = factor(Coef$Machine)
Coef$Training = factor(Coef$Training)

#Data = Data %>% rowid_to_column("SID")

p =  ggplot(Data, aes(Hrs, Production, color = Machine)) +
```

```r
  geom_point(alpha = .2) +
  geom_smooth(method = "lm", se = F, linetype = "dashed", alpha = .6) +
  facet_wrap(~Plant) +
  theme(panel.background = element_rect(fill = "white"))

# match up with the data

CoefMap = Data %>% distinct(Plant, Machine, Training) %>%
  inner_join(Coef, by = c("Plant", "Machine", "Training"))

p = p + geom_abline(data = CoefMap,
                    aes(intercept = Intercept, slope =  Slope, color = Machine))
p
```



Note above, lm and lmer produce projections only for dimensions with existing data *(so, in this case, Plant 2 does not have a machine type 1 or 4, yet Plant 2 is the only plant that executed a training program. This confounds counterfactuals, or projections, like: "If Plant 1 adds training, how will that impact Production?")*. So, machine learning *(and Homer Simpson dashboards)* are not going to provide the answers you seek. STOP looking there!

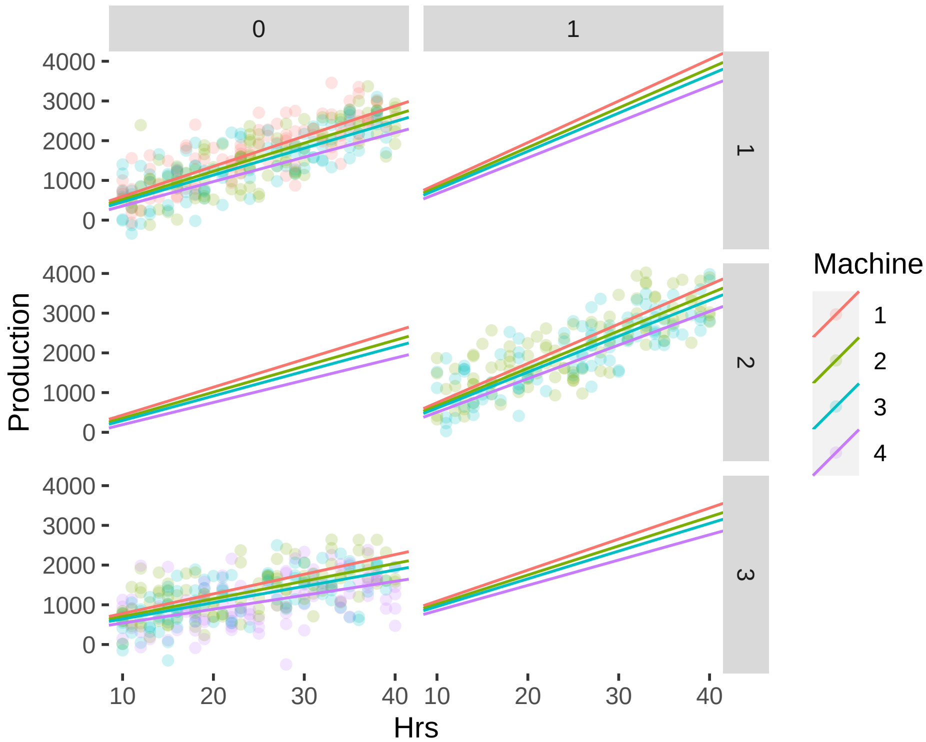
Bayesian Model

We need the **effects** for Plant, Machines and Training across **all** scenarios. So, let's build a model that can give us that metric. Lets build a mulitlevel regression model that projects production accross all levels:

```
stanMod1 <- '

data {

  int<lower=0> N;
  vector[N] x;
  vector[N] y;

  int<lower=0> J;
  int Plant[N];

  int<lower=0> K;
  int Machine[N];
```

```
  int<lower=0> L;
  int Training[N];

  vector[J] palpha;
  vector[K] palpha2;
  vector[L] palpha3;

  vector[J] pbeta;
  vector[K] pbeta2;
  vector[L] pbeta3;

}
parameters {

  vector[J] alpha;
  vector[K] alpha2;
  vector[L] alpha3;

  vector[J]  beta;
  vector[K]  beta2;
  vector[L]  beta3;

  real<lower=0> sigma;

}

transformed parameters {

vector[N] y_hat;
for (i in 1:N)
  y_hat[i] = alpha[Plant[i]] + alpha2[Machine[i]] + alpha3[Training[i]] + (beta[Plant[i]] * x[i]) + (be

}

model {

  target += normal_lpdf(alpha | palpha, 20);
  target += normal_lpdf(alpha2 | palpha2, 20);
  target += normal_lpdf(alpha3 | palpha3, 20);

  target += normal_lpdf(beta | pbeta, 50);
  target += normal_lpdf(beta2 | pbeta2, 50);
  target += normal_lpdf(beta3 | pbeta3, 50);

  target += normal_lpdf(sigma | 50, 50);

  target += normal_lpdf(y | y_hat, sigma);

}

'

fit <- stan(model_code = stanMod1,  data = list(
```

```
  N = nrow(Data),
  y = Data$Production,
  x = Data$Hrs,
  J = length(unique(Data$Plant)),
  K = length(unique(Data$Machine)),
  L = length(unique(Data$Training)),
  Plant = as.numeric(Data$Plant),
  Machine = as.numeric(Data$Machine),
  Training = as.numeric(Data$Training),
  palpha = as.numeric(coef(HomeMod)$Plant[,1]),
  palpha2 = as.numeric(coef(HomeMod)$Machine[,1]),
  palpha3 = as.numeric(coef(HomeMod)$TrainingID[,1]),
  pbeta = as.numeric(coef(HomeMod)$Plant[,2]),
  pbeta2 = as.numeric(coef(HomeMod)$Machine[,2]),
  pbeta3 = as.numeric(coef(HomeMod)$TrainingID[,2])
), refresh = 0)

sumFit <- data.frame(summary(fit))

# build on this

Intercept1 <- summary(fit, pars = c("alpha"), probs = c(0.1, 0.9))$summary
Intercept2 <- summary(fit, pars = c("alpha2"), probs = c(0.1, 0.9))$summary
Intercept3 <- summary(fit, pars = c("alpha3"), probs = c(0.1, 0.9))$summary

Slope1 <- summary(fit, pars = c("beta"), probs = c(0.1, 0.9))$summary
Slope2 <- summary(fit, pars = c("beta2"), probs = c(0.1, 0.9))$summary
Slope3 <- summary(fit, pars = c("beta3"), probs = c(0.1, 0.9))$summary


CoefMapP <- data.frame(Plant = unique(Data$Plant),   PIntercept = Intercept1[,1], PSlope = Slope1[,1])
CoefMapM <- data.frame(Machine = unique(Data$Machine), MIntercept = Intercept2[,1], MSlope = Slope2[,1]
CoefMapT <- data.frame(Training = unique(Data$Training), TIntercept = Intercept3[,1], TSlope = Slope3[,

CoefMapB = crossing(CoefMapP, CoefMapM, CoefMapT) %>%
  mutate(Intercept = PIntercept + MIntercept + TIntercept, Slope = PSlope + MSlope + TSlope)



p =  ggplot(Data, aes(Hrs, Production, color = Machine)) +
  geom_point(alpha = .2) +
  geom_abline(data = CoefMapB,
              aes(intercept = Intercept, slope =  Slope, color = Machine)) +
  facet_grid(Plant ~ Training) +
  theme(panel.background = element_rect(fill = "white"))
p
```



(above, showing regression models across plants and training on margins and machines in color)

This is one of the strengths of Bayesian Modleing - the ability to create effects for *(posterior)* scenarios where data *(likelihood)* does not exist!! *(note that each plant has a projection for ALL machines)* BIG DEAL! Now that we have a roungh estimated of the effects by level, we can build a counterfactual, or projection, for effects where machines, or training don't exist. Back to the question: "What would be the effect of training on Plant 1". Plant 2, where the training occurs, only has machines 2 and 3?

Bringing it together with the DAG

Now we have metrics for these relationships, and we can plug them back into the DAG and test for conditional effects. Let's work our way through the DAG in more detail, starting with the fork. First, review the DAG rules from Pearl as applied to our data:

Fork Rules

- Plant and Machine are dependent
- Plant and Training are dependent
- Machine and Training are likely dependent
- Machine and Training are independent, conditional on Plant

Chain Rules

- Plant and Machine are dependent
- Plant and Training are dependent
- Machine and Production are dependent
- Training and Production are dependent
- Plant and Production are likely dependent
- Plant and Production are independent, conditioned on Machine
- Plant and Production are independent, conditioned on Training

Collider Rules

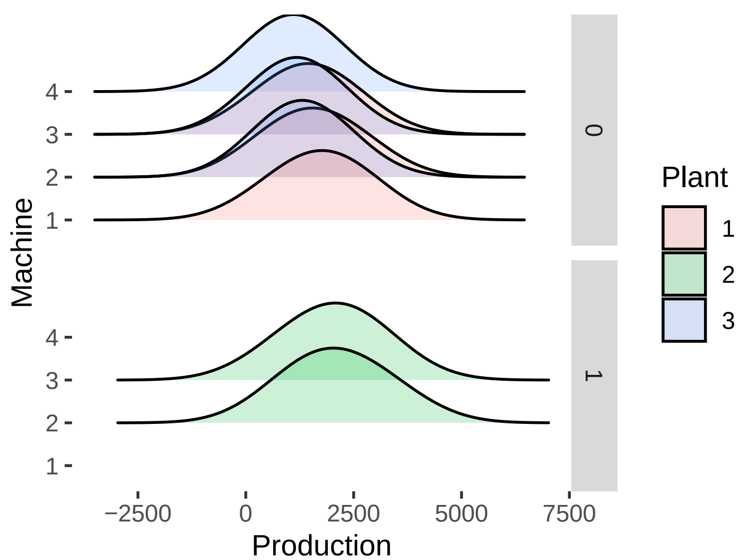
- Machine and Production are dependent *(inherited from chain rule)*
- Training and Production are dependent *(inherited from chain rule)*
- Training and Machine are independent
- Training and Machine are dependent, conditional on Production

So, that's our rules, now all we need to do is measure $P(beforeintervention|condition)$ <> $P(afterintervention|condition)$. How much difference counts as causal? Well that depends on your judgment! If you're measuring production like this example, the effect is obvious. But if you're measuring an ambiguous economic driver with a small impact on industry demand, or stock price, you might need to tune your significance threshold.

This exercise and the Stan model has already given us the answer, so you can already make credible inferences about causality.

First, let's look at the **Fork**. Comparing Training *(margins)*across machines and plants:

```
p3 <- ggplot(Data, aes(y =Machine , x = Production, fill = Plant)) +
  geom_density_ridges(bandwidth = 1000, alpha = .2) +
  facet_grid(rows = vars(Training)) +
  theme(panel.background = element_rect(fill = "white"))
p3
```



So there's visual evidence of a training effect, but since it's all in Plant 2, how much of that is plant vs. training assuming machine types can be projected across plants?

We can hold MachineType == 3 and compare Training==0 to Training==1 *(remember this includes all plant with machine 3, so plant effect has to be controlled for, but just for a start)*:

Do-Calculus

Our **Do-Calculus** computation compares probability *(integration across densities)* for 2 conditions *(states)* - training and no training:

```
ll = round(if_else(
  mean(filter(Data, MachineType == 3, Training==0)$Production) >
  mean(filter(Data, MachineType == 3, Training==1)$Production),
  mean(filter(Data, MachineType == 3, Training==0)$Production),
  mean(filter(Data, MachineType == 3, Training==1)$Production)),0)
```

```
ul = round(if_else(
  max(filter(Data, MachineType == 3, Training==0)$Production) <
  max(filter(Data, MachineType == 3, Training==1)$Production),
  max(filter(Data, MachineType == 3, Training==0)$Production),
  max(filter(Data, MachineType == 3, Training==1)$Production)),0)



P2 = integrate(approxfun(density(filter(Data, MachineType == 3, Training==0)$Production)),
               lower = ll,
               upper = ul,
               subdivisions=200)$value

P3 = integrate(approxfun(density(filter(Data, MachineType == 3, Training==1)$Production)),
               lower = 2000,
               upper = max(filter(Data, MachineType == 3)$Production),
               subdivisions=200)$value

P3 - P2
```

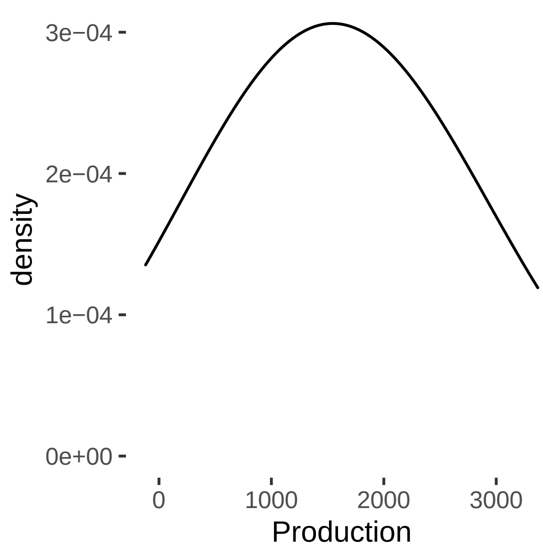
```
## [1] 0.3518013
```

So the difference here is ~ 20%, which is significant by any criteria. So, we know there's a causal effect here.

Simulations

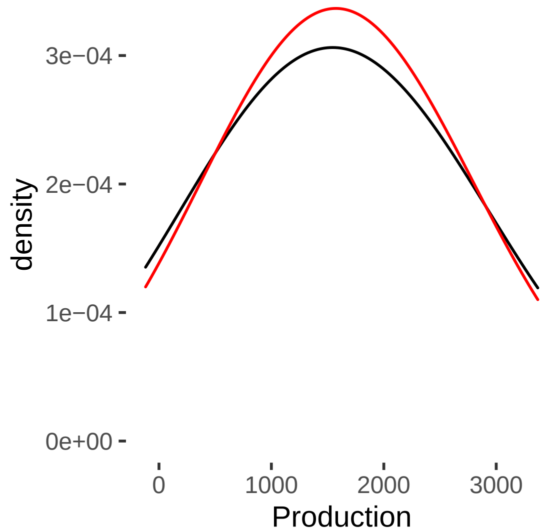
Another good way to analyze causal scenarios is to run a simulations *(since we have estimated effects from Stan, we can generate simulations all day)* . First, let's look at actual data for Machine 2 in Plant 1 as a baseline:

```
Plant1Machine2 = filter(Data, Machine == 2, Plant == 1)
pSim = ggplot(Plant1Machine2, aes(x = Production)) +
  geom_density(bw = 1000, alpha = .2) +
  theme(panel.background = element_rect(fill = "white"))
pSim
```



Now let's simulate this scenario:

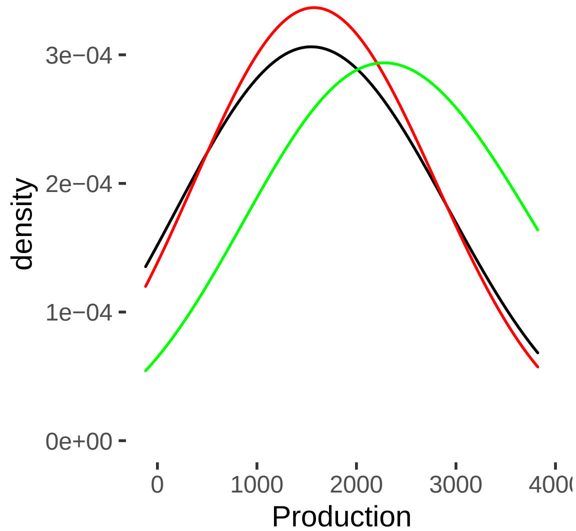
```
Plant1Machine2T0Coef = filter(CoefMapB, Plant == 1, Machine == 2, Training == 0)
Plant1Machine2T1Coef = filter(CoefMapB, Plant == 1, Machine == 2, Training == 1)
SimP1M2T0 = data.frame(Production = Plant1Machine2T0Coef$Intercept + (Plant1Machine2T0Coef$Slope *  Plan
SimP1M2T1 = data.frame(Production = Plant1Machine2T1Coef$Intercept + (Plant1Machine2T1Coef$Slope *  Plan

pSim = pSim +
  geom_density(data = SimP1M2T0, aes(x = Production), bw = 1000, alpha = .2, color = "red")
pSim
```



(showing that the simulation distribution is the same and the actual, above)

NOW, let's use the Stan effects to simulate the same scenario **WITH** training:

```
pSim = pSim +
  geom_density(data = SimP1M2T1, aes(x = Production), bw = 1000, alpha = .2, color = "green")
pSim
```



As you can see, there is a marked effect - an increase in produciton.

We'll work through some more exercises in the next document.