# Bayesian Data Analysis for Statistical Causal Inference

A gentle challenge of Data Analysis Habits in Software Engineering Research

JulianFrattini/**bda4sci** Public

# Context & Goal

**Context**: Software engineering research aims to determine causal effects. Correlations serve for predictions, but do not inform interventions.

**Problem**: Many researchers are, however, ill-equipped to obtain valid answers to these causal questions.

**Problem**: This tutorial is aimed at academics that aim to tackle causal questions but lack the tools for it.

# Status Quo

Data Analysis in Software Engineering Research

# Data Analysis in Software Engineering Research

Data analysis in empirical SE research with quantitative data typically follows a process like:

1. **Formulate a hypothesis** that attributes an impact of an independent on a dependent variable
2. **Collect data** from a specific context
3. Select an **appropriate hypothesis test** depending on the properties of the variables
4. Perform the test and **calculate p-value and effect size**
5. **Report the results** and limit the conclusions based on the context factors

# Issues

This process is subject to several issues which are mostly rooted in two core problems.

**Lack of a causal inference framework**

**Simple frequentist analysis methods**

# Statistical Causal Inference

A rigorous approach to obtaining valid conclusions from data

# Overview

Most worthwhile research questions are of causal nature, but answers to such questions **cannot be computed from data alone**. Instead, addressing them necessitates knowledge about ***how* the data was generated**.

**Statistical causal inference:** inferring causal relationships from quantitative data

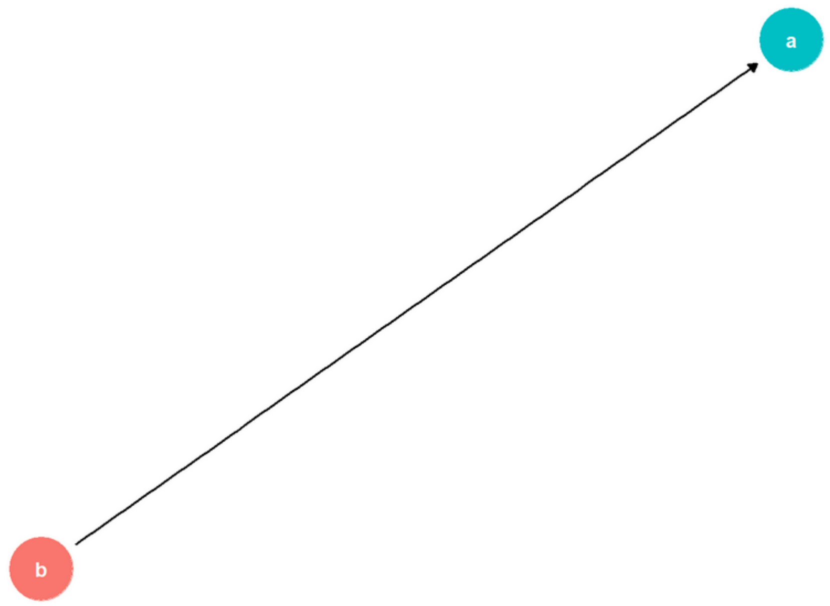
Pearl, J. (2009). Causal inference in statistics: An overview.

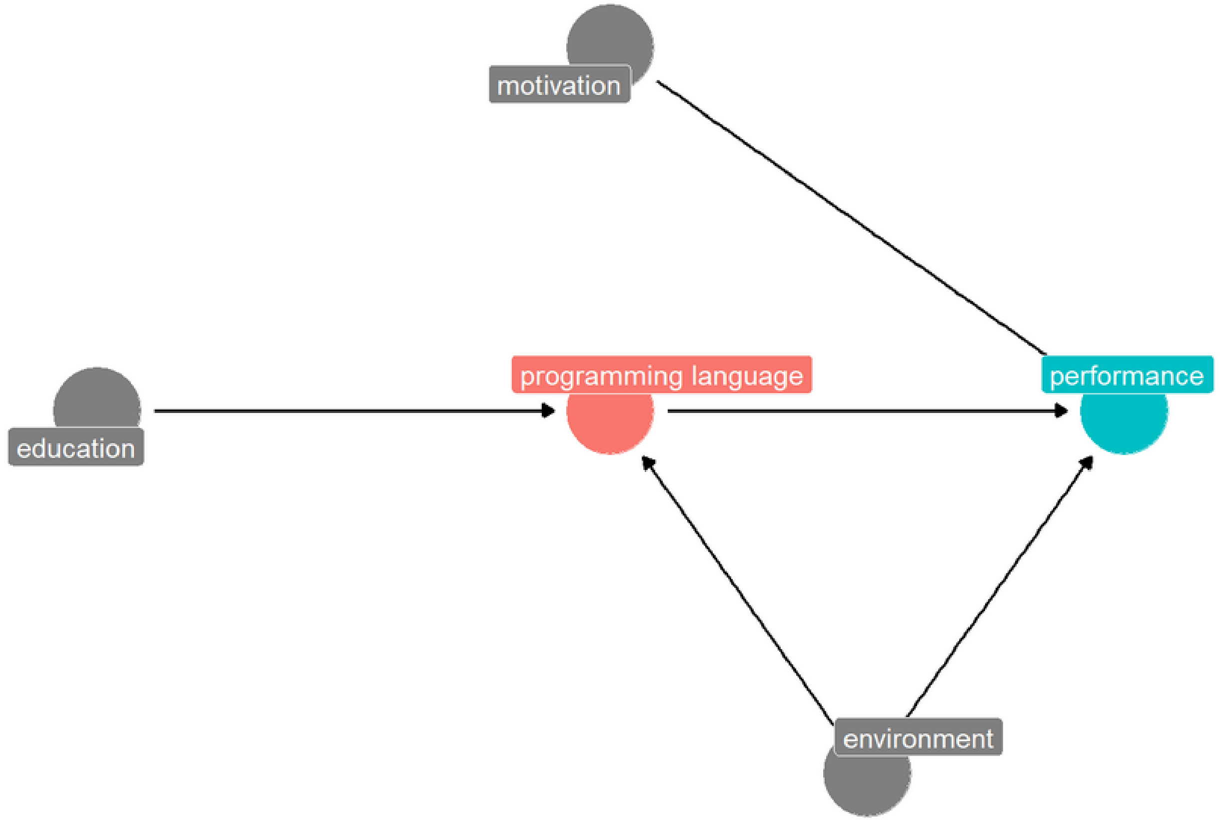
# Terminology

Exchange about causal inference necessitates the following terms:

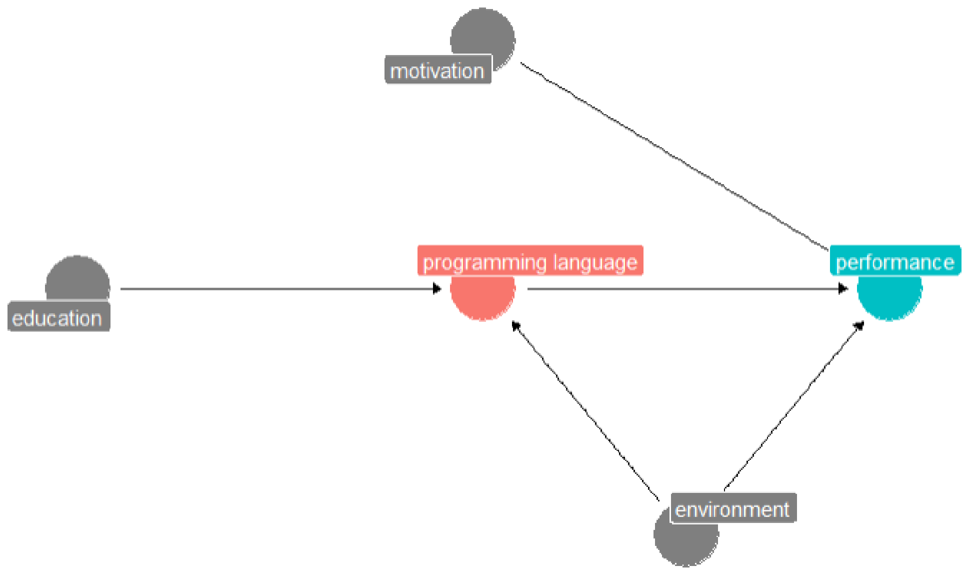
- **Factor**: variable of a specific type (e.g., categorical, continuous) projecting a construct onto a value
  - **Treatment** (or: main factor): independent variable of interest
  - **Outcome** (or: response variable): dependent variable of interest
- **Relationships**: association between two factors

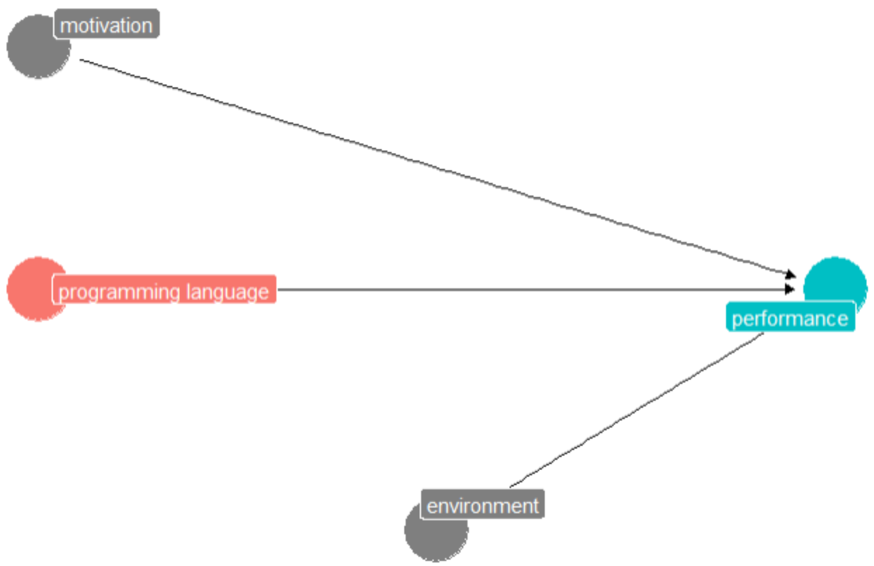
# Visualizing causal Assumptions via Directed Acyclic Graphs (DAGs)



# Experimental vs. Observational Studies



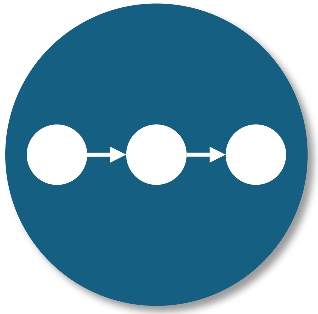
# Experimental vs. Observational Studies

Controlled experiments are **expensive** to conduct and controlling a treatment variable may be difficult without **perturbing the context**. Hence, we often need to resort to observational studies.

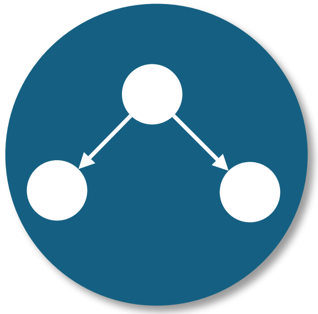
This, in turn, means that in the data generation process, the relationship between the treatment and outcome may be **confounded through different types of association.**

# Sources of Association

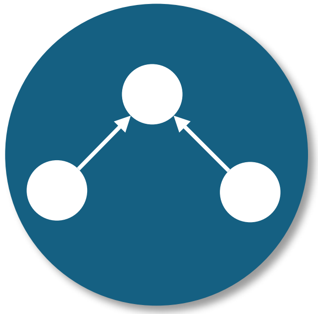
Since (1) most relationships of interest are rarely limited to only two variables and (2) these additional variables may interact with the relationship of interest in unforeseen ways, we need to be aware of *how* they can interact.



**Mediator**

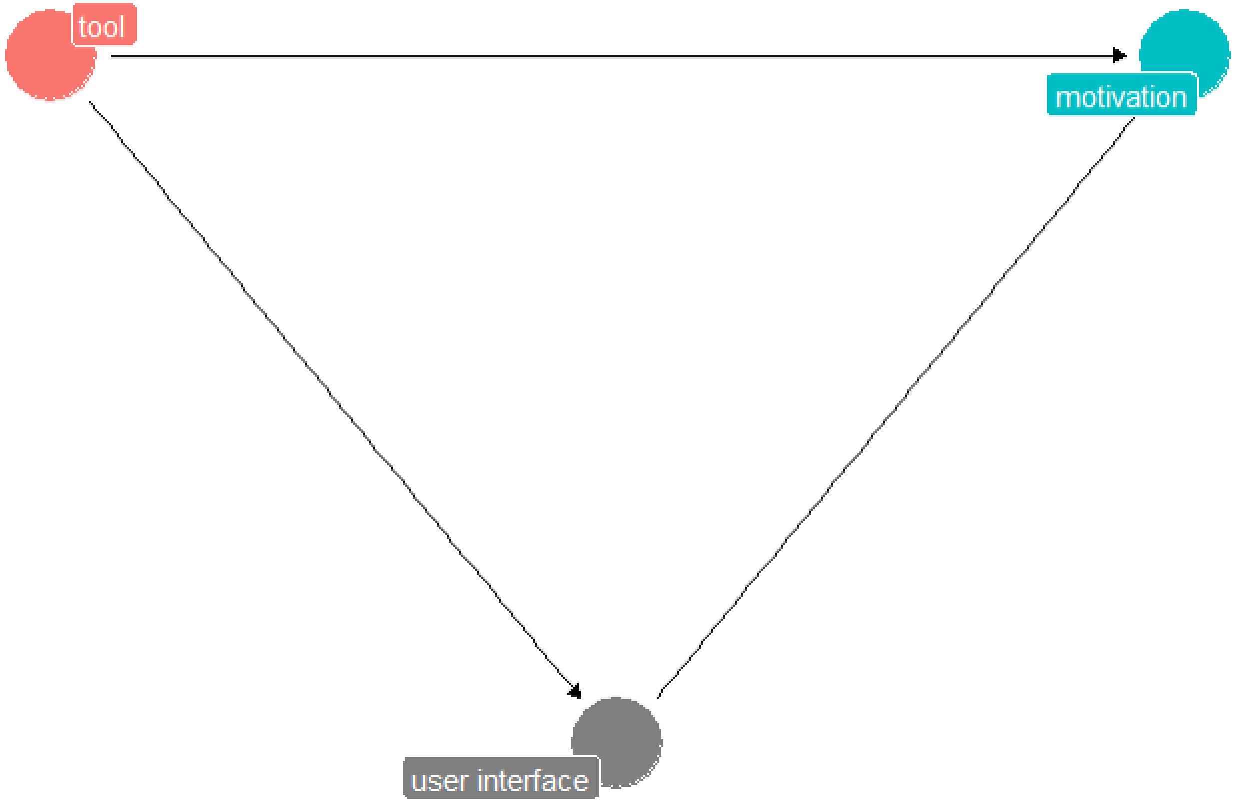


**Fork**



**Collider**

# Mediators



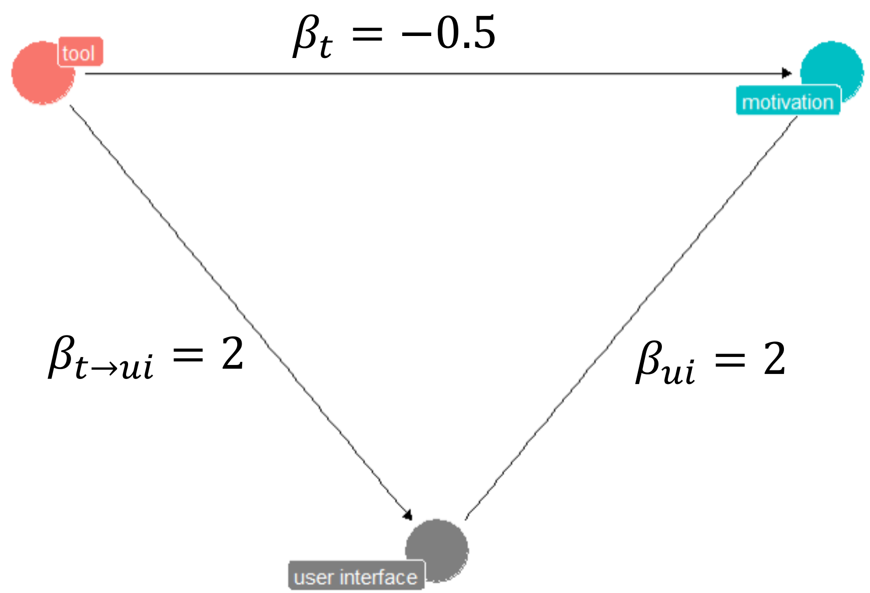
# Mediators

Mediators do not introduce a confounding bias to the causal analysis. However, they influence the distinction between the direct and total effect.

- **Direct effect**: immediate, isolated effect of the treatment on the outcome
- **Total effect**: direct effect plus all mediated effects

# Mediators

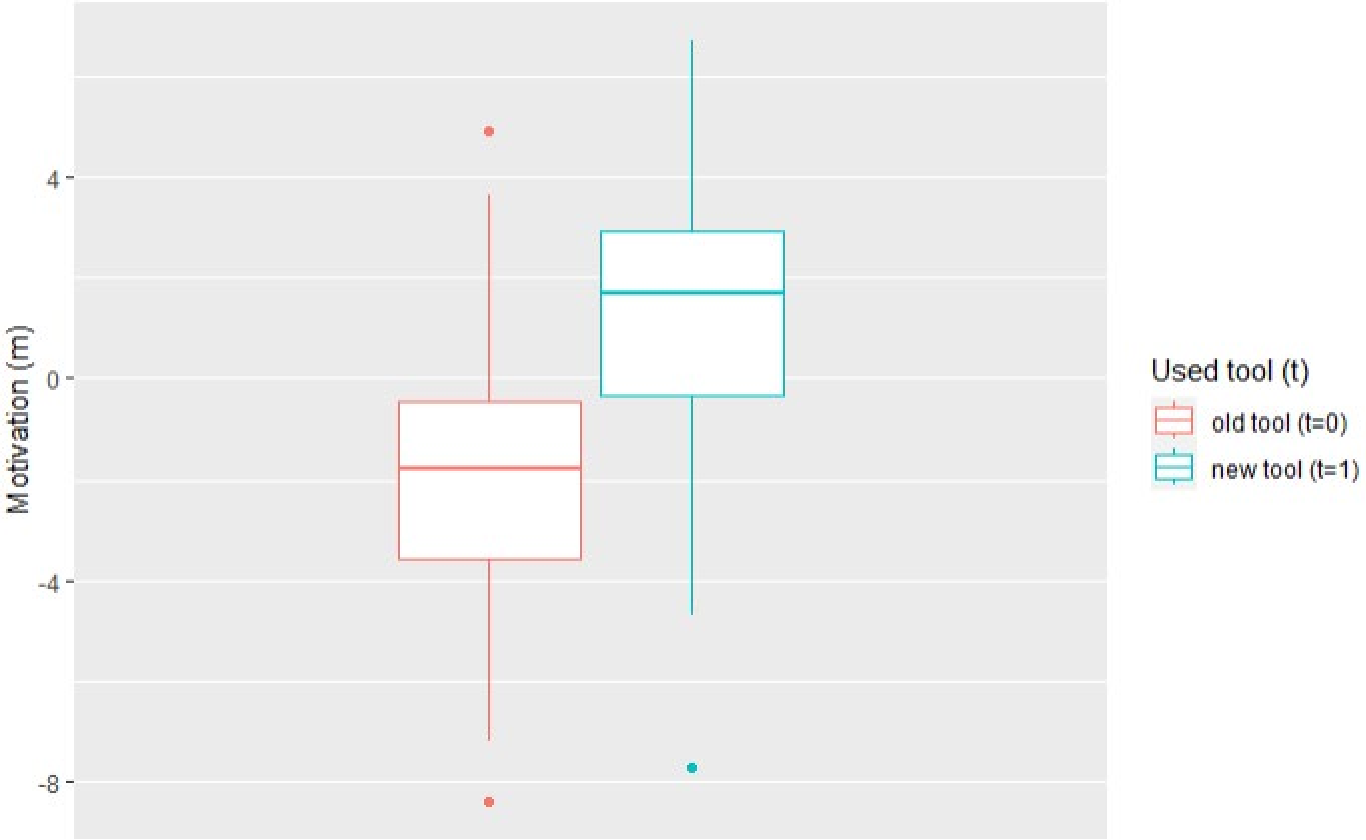
To demonstrate this distinction, consider the following assumption.




```
```{r simulation1}
n <- 500 # number of simulated units, i.e., tool users

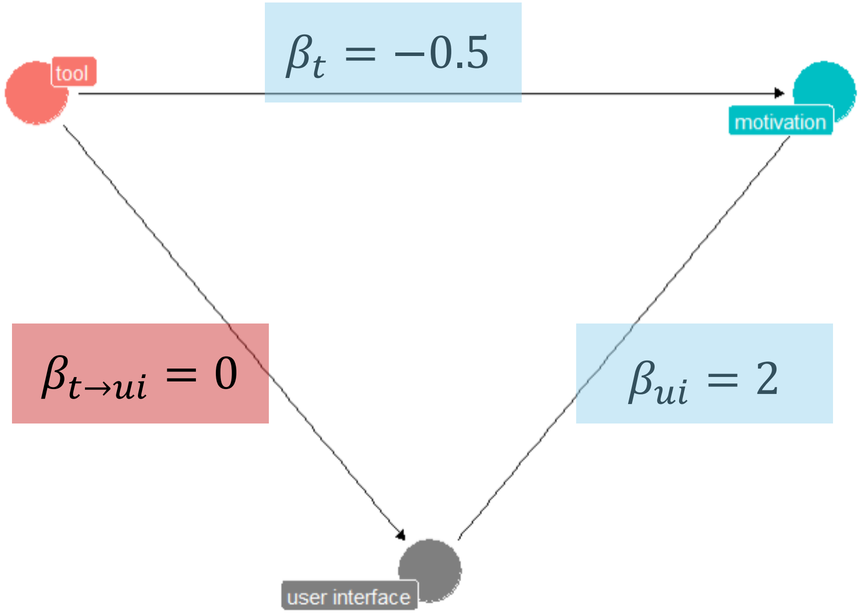
d1 <- data.frame(
  t = rep(c(0,1), n/2) # simulated values of t, i.e., alternating
                       # "old tool" (=0) and "new tool" (=1) assumption
) %>% mutate(
  ui = rnorm(n, 2*t - 1, 1), # simulated values of ui
  m = rnorm(n, -0.5*t + 2*ui, 1) # simulated values of m
)
```
```

# Mediators



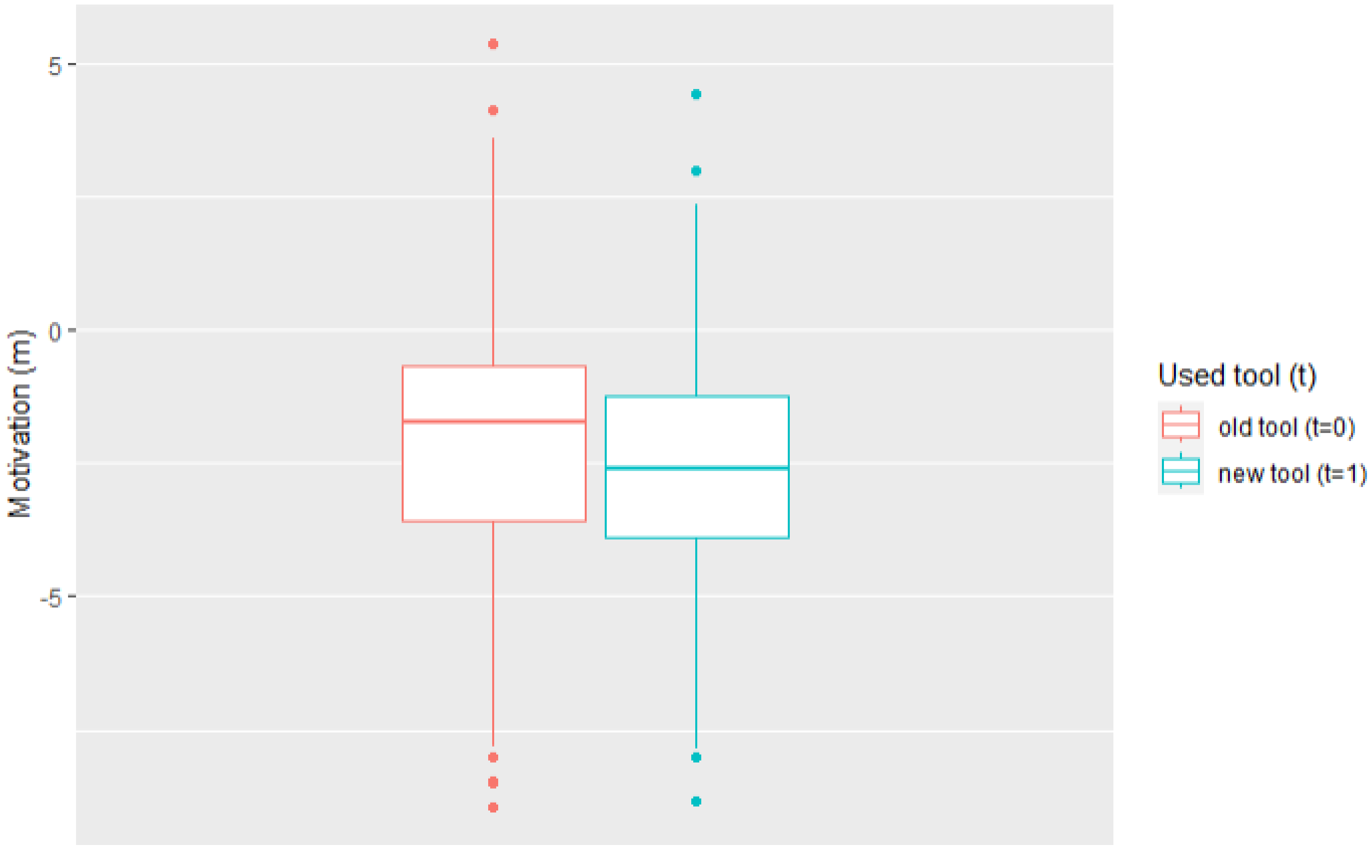
# Mediators

Assume that the indirect effect changes.

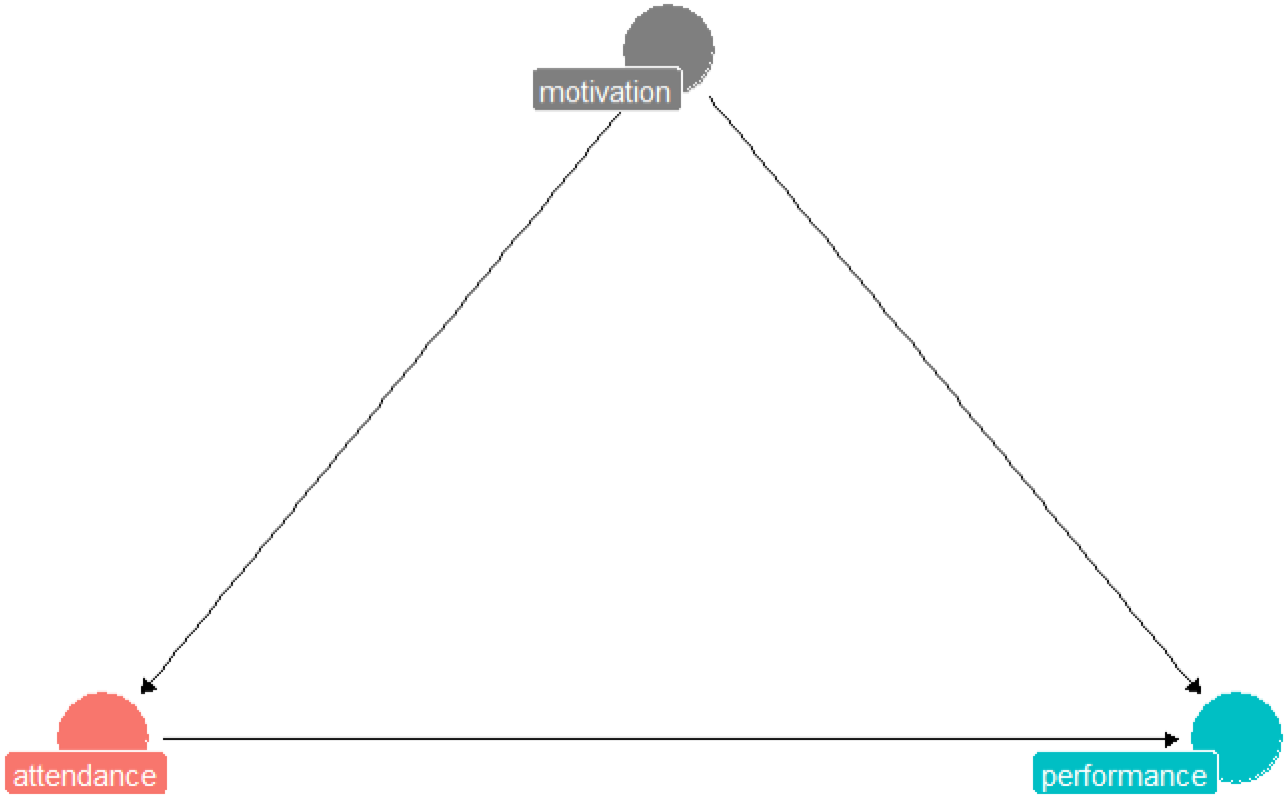



```
```{r simulation2}
d2 <- data.frame(
  t = rep(c(0,1), n/2) # simulated values of t
) %>% mutate(
  ui = rnorm(n, -1, 1), # simulated values of ui
  m = rnorm(n, -0.5*t + 2*ui, 1) # simulated values of m
)
```
```

# Mediators

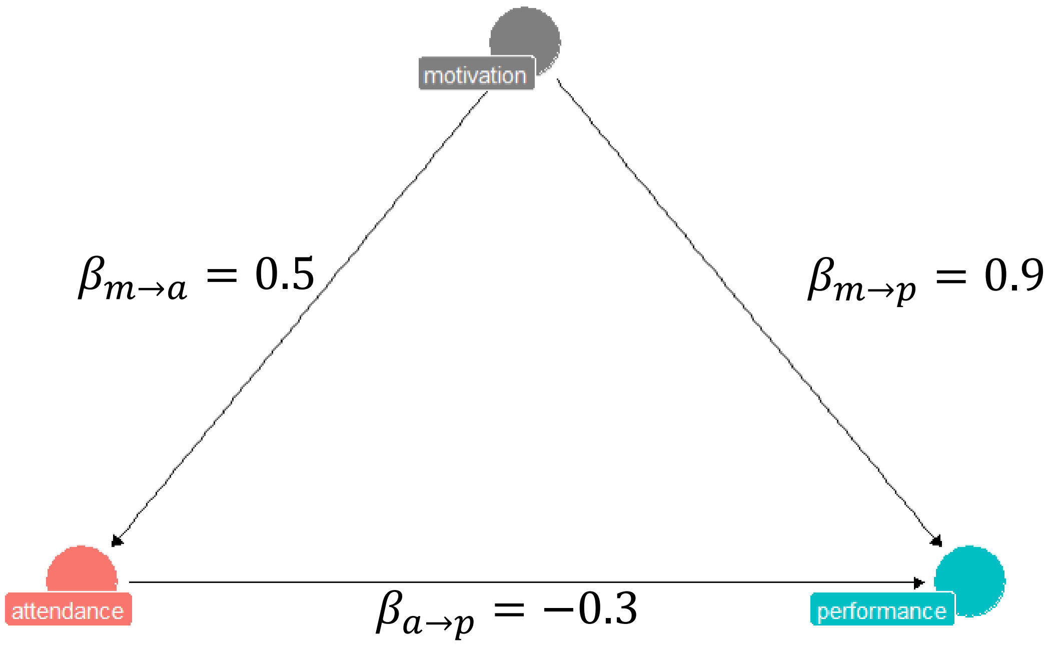


# Forks



# Forks

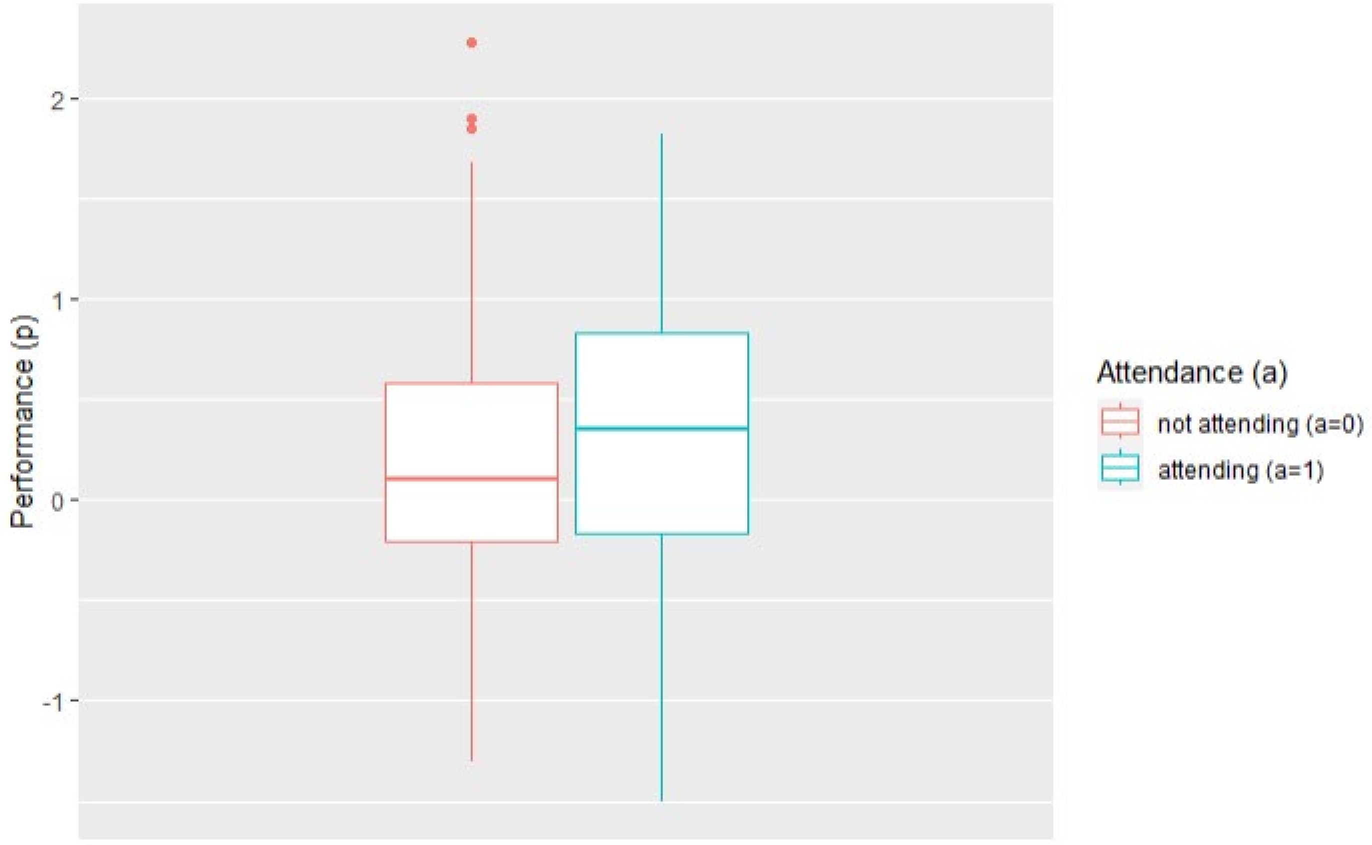
Assume the following ground truth.




```
```{r simulation}
n <- 500 # number of simulated units, i.e., students

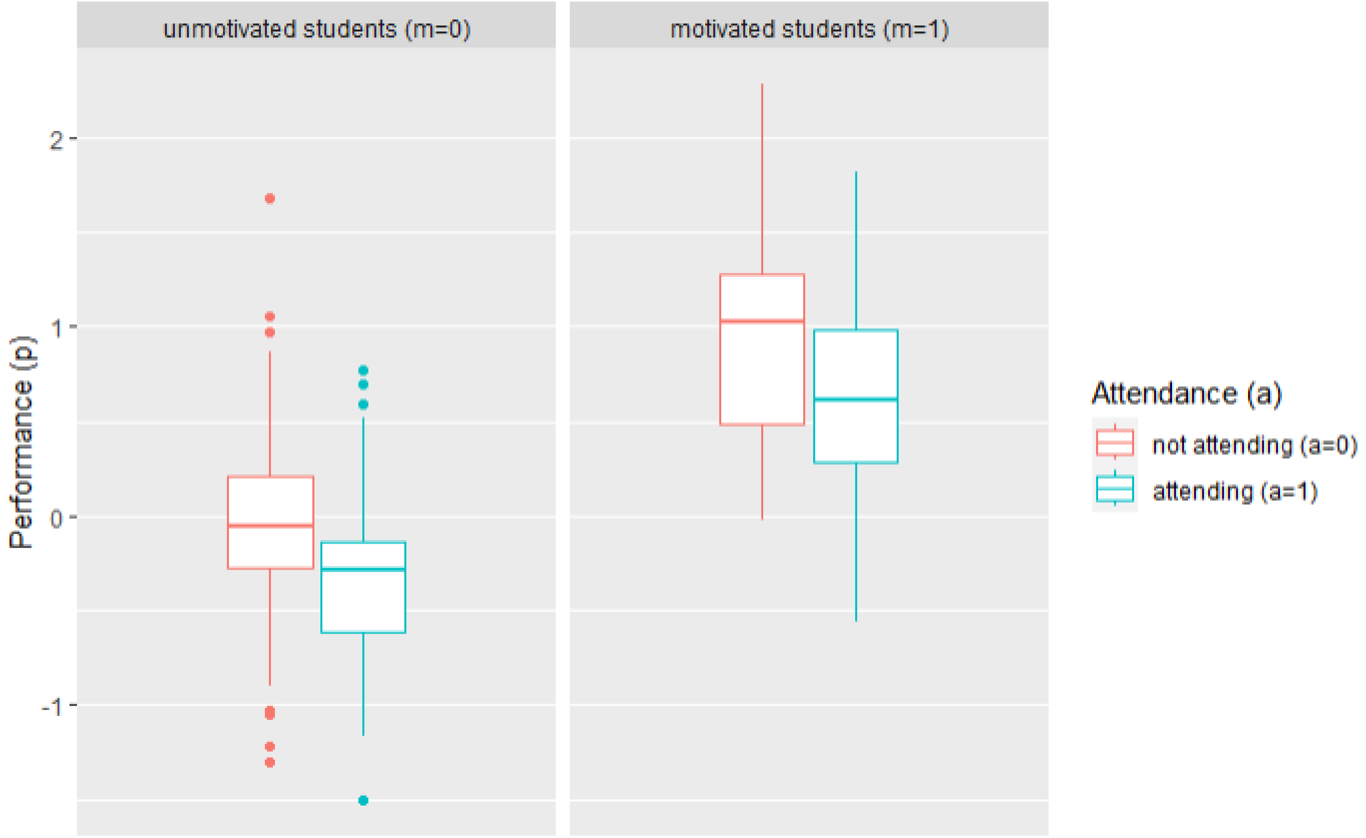
d <- data.frame(
  m = rbinom(n, 1, 0.5) # simulated values of m
) %>% mutate(
  # simulated values of a, which depend on the values of m
  a = rbinom(n, 1, 0.3+0.5*m)
  ) %>% mutate(
    # simulated values of p, which depend on both a and m
    p = rnorm(n, -0.3*a + 0.9*m, 0.5)
  )
```
```

# Forks

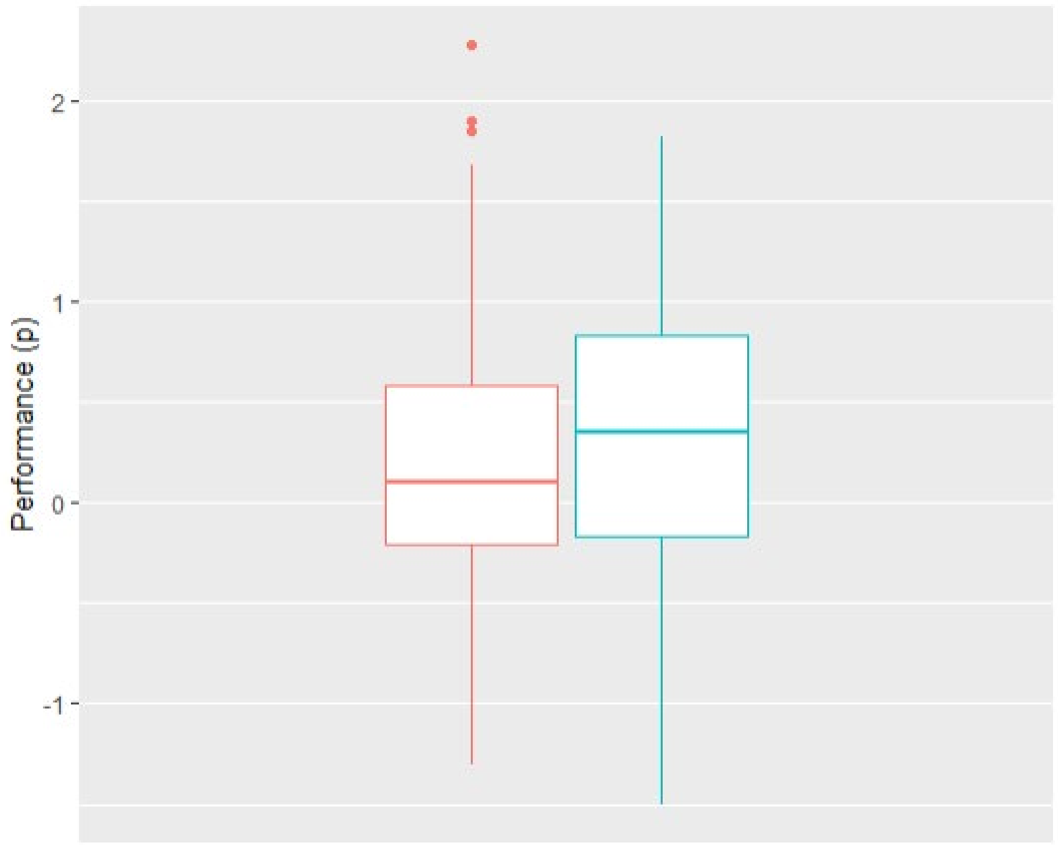


Performance (p)
2
1
0
-1
Attendance (a)
not attending (a=0)
attending (a=1)

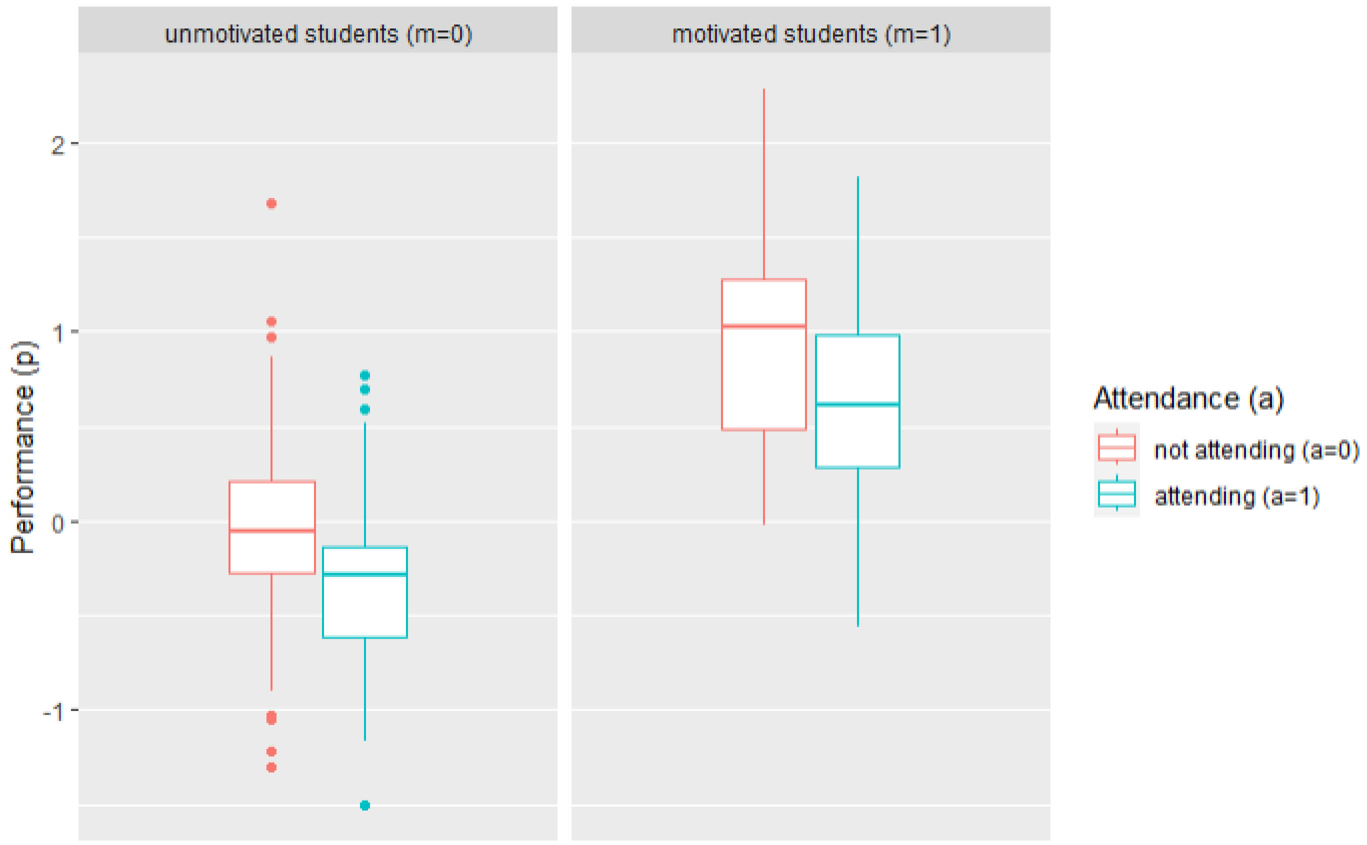
# Forks


unmotivated students (m=0)
motivated students (m=1)
Performance (p)
2
1
0
-1
Attendance (a)
not attending (a=0)
attending (a=1)

# Forks

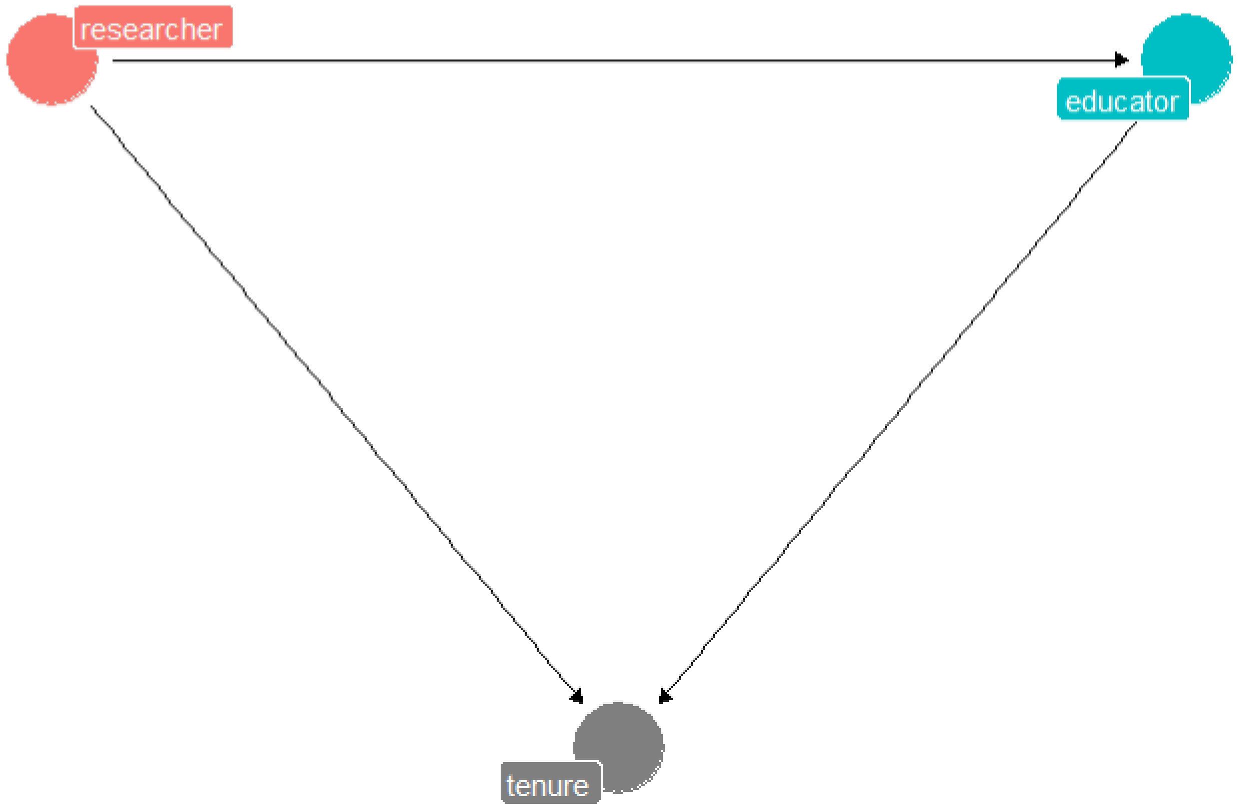


Performance (p)
2
1
0
-1

unmotivated students (m=0)
motivated students (m=1)
Performance (p)
2
1
0
-1
Attendance (a)
not attending (a=0)
attending (a=1)

# Colliders



# Colliders

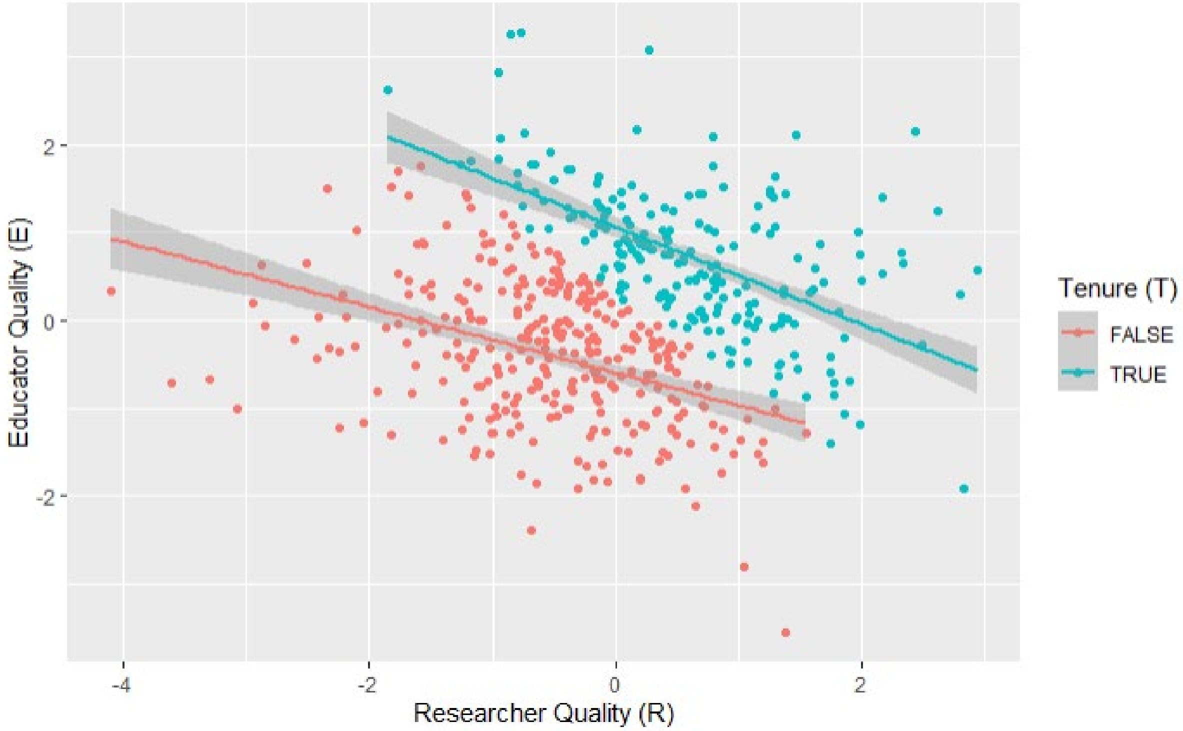
Assume that there is actually no effect of an academic's researching quality on their educational quality ($\beta_{r \to e} = 0$). However, If you are either a good researcher or a good educator, you will get tenure.

````
```{r simulation}
n <- 500 # number of simulated units, i.e., academics
threshold <- 0.3 # an arbitrary threshold, where any cumulative value of R and E that exceeds it means tenure

d <- data.frame(
  r = rnorm(n, 0, 1), # simulated values of R, which are normally distributed
  e = rnorm(n, 0, 1) # simulated values of E, which are also normally distributed and not influenced by R as per our assumption
) %>% mutate(
  t = ifelse(r+e>threshold, TRUE, FALSE) # simulated values of T, which are TRUE if the combined value of R and E exceed the threshold
)
```
````

# Colliders

For both academics that have tenure and those that do not, there is a negative association between researching and educational capability.

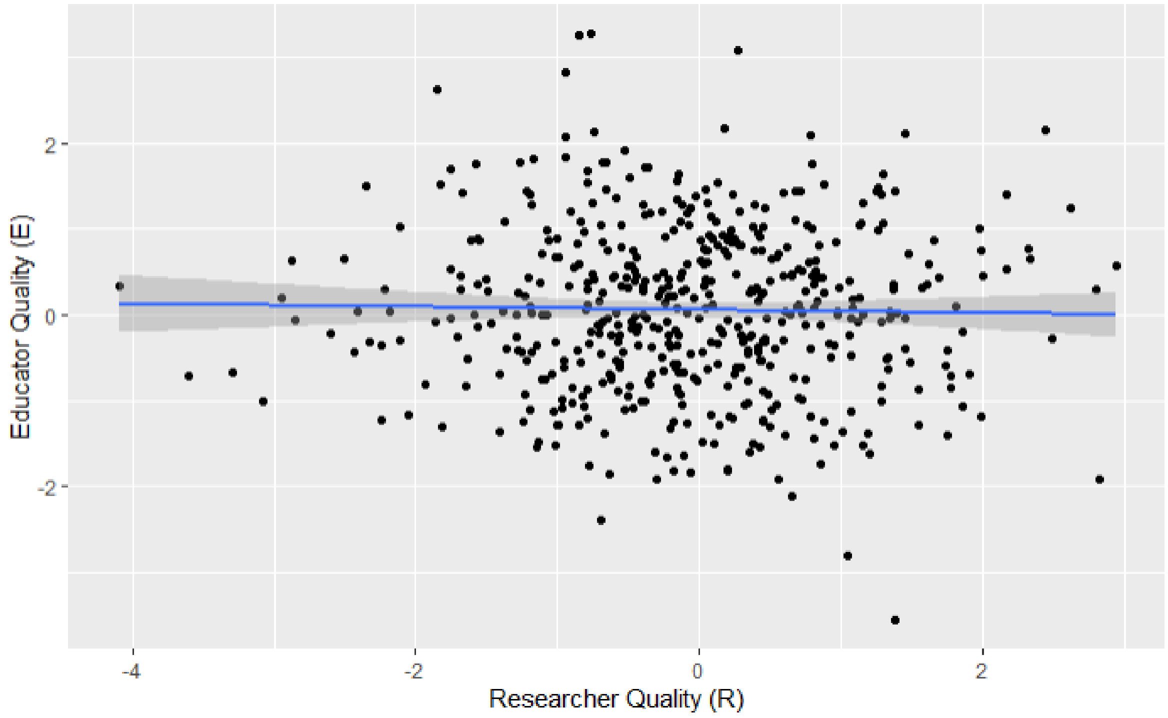


# Colliders

This conclusion should not be possible, as we manually defined that there is no relationship between the two variables.

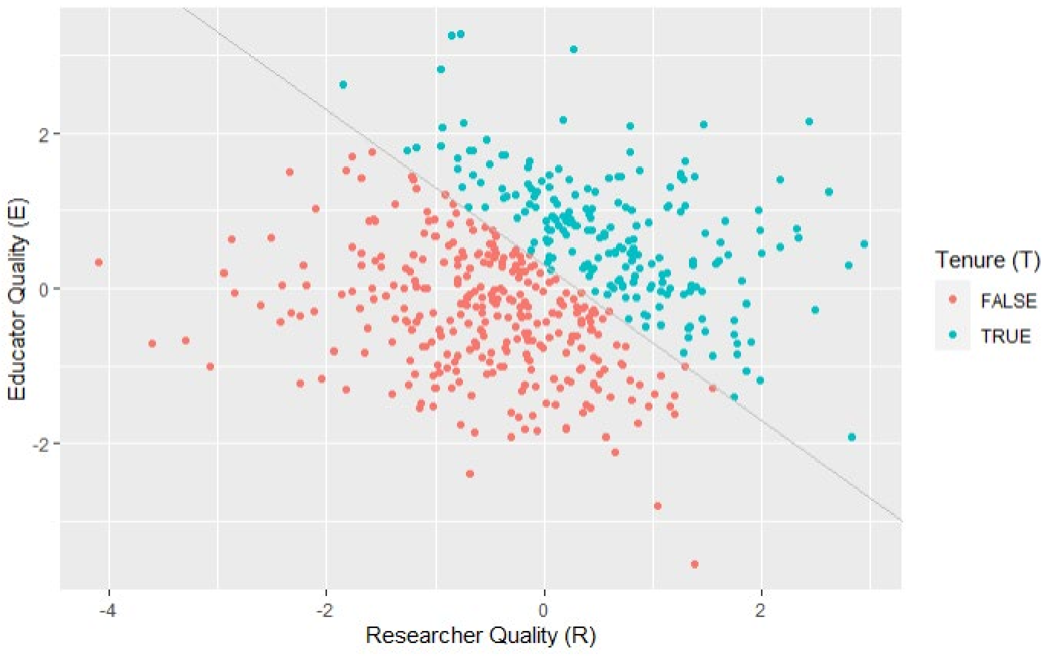
```
```{r simulation}
n <- 500 # number of simulated units, i.e., academics
threshold <- 0.3 # an arbitrary threshold, where any

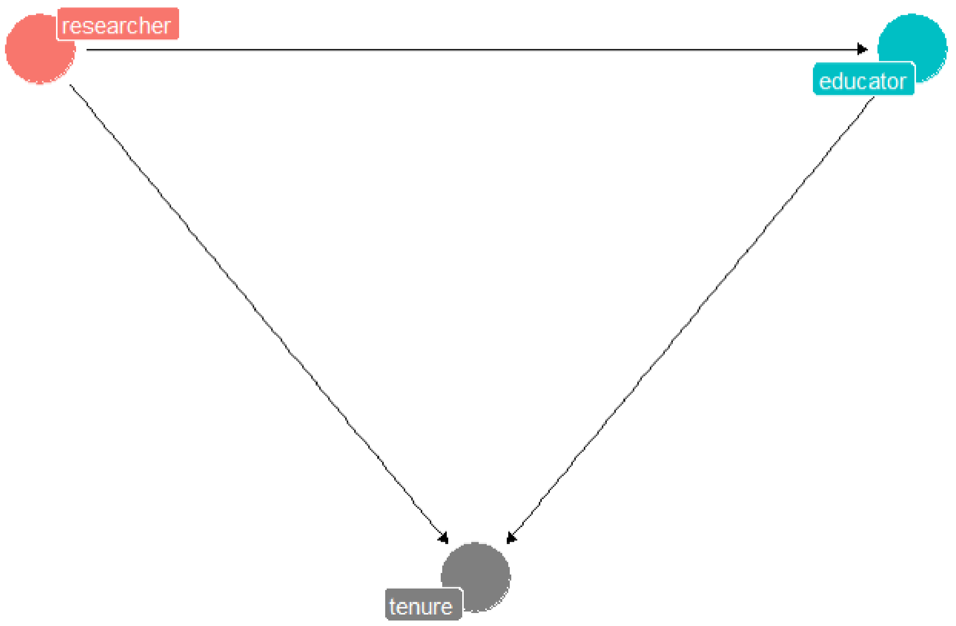
d <- data.frame(
  r = rnorm(n, 0, 1), # simulated values of R, which
  e = rnorm(n, 0, 1) # simulated values of E, which a
) %>% mutate(
  t = ifelse(r+e>threshold, TRUE, FALSE) # simulated
)
```
```



# Colliders

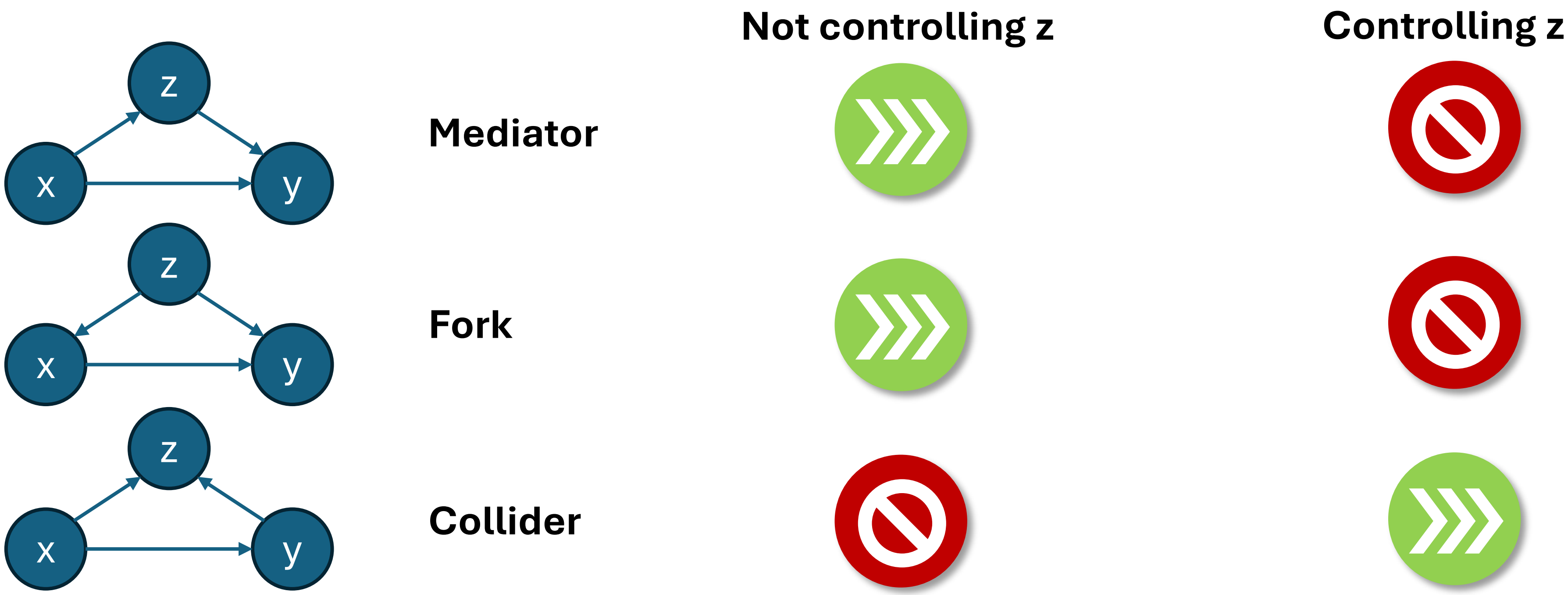
By controlling for the collider t, we introduced a spurious association between r and e.



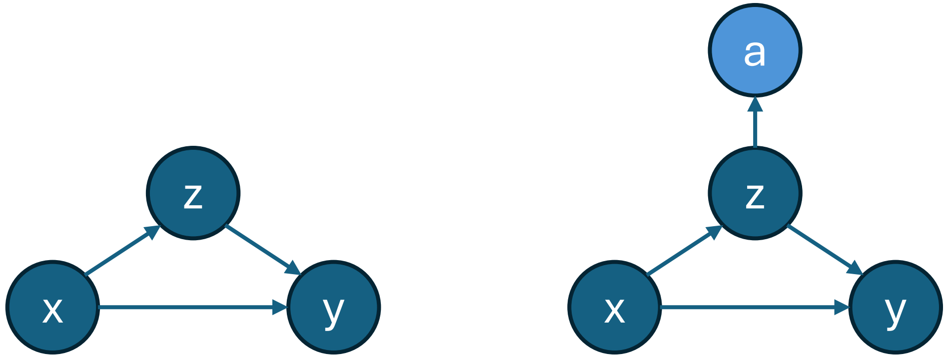
# Controlling Variables

Controlling variables has a different effect on the ”flow of information” depending on their relationship.



# Controlling Descendants

Controlling the descendant (i.e., child) of a variable has a comparable effect as controlling for the actual variable.

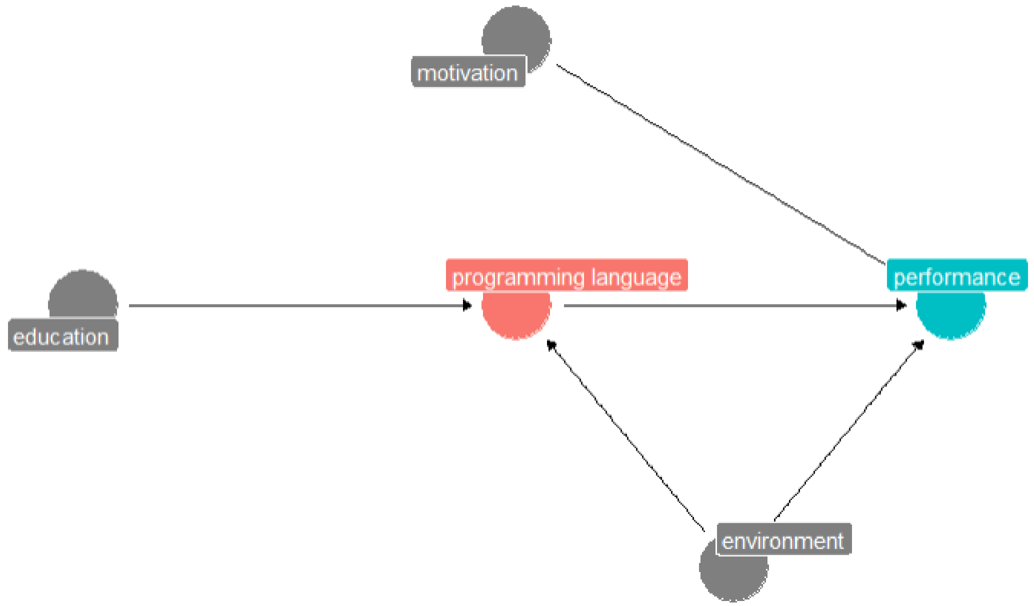



Controlling z blocks the path $x \to z \to y$, but controlling a has a similar (though maybe not as complete) effect.

# Paths

Two nodes are connected via a **path**, i.e., a series of adjacent arrows that pass through each node at most once.

- **Causal path**: path where all arrows point from the treatment to the outcome
- **Non-causal path**: path where at least one arrow points from the outcome to the treatment
- **Backdoor path**: that enters the treatment



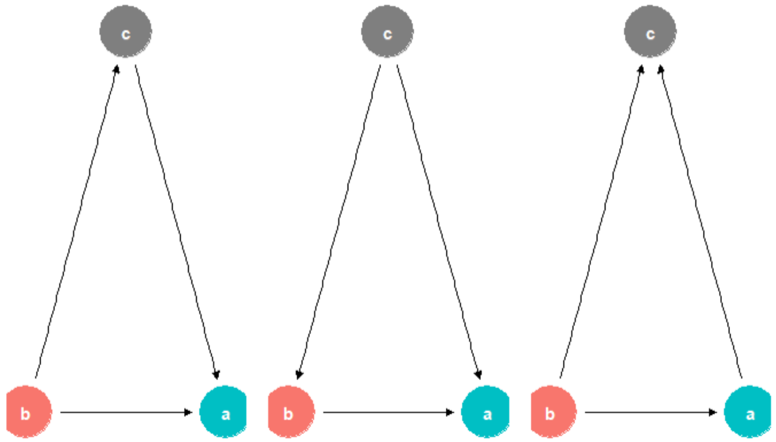
# The Backdoor Adjustment

To infer a causal relationship from observational data, we need to deconfound the relation of interest by **selecting a set of variables Z** that conform the backdoor criterion:

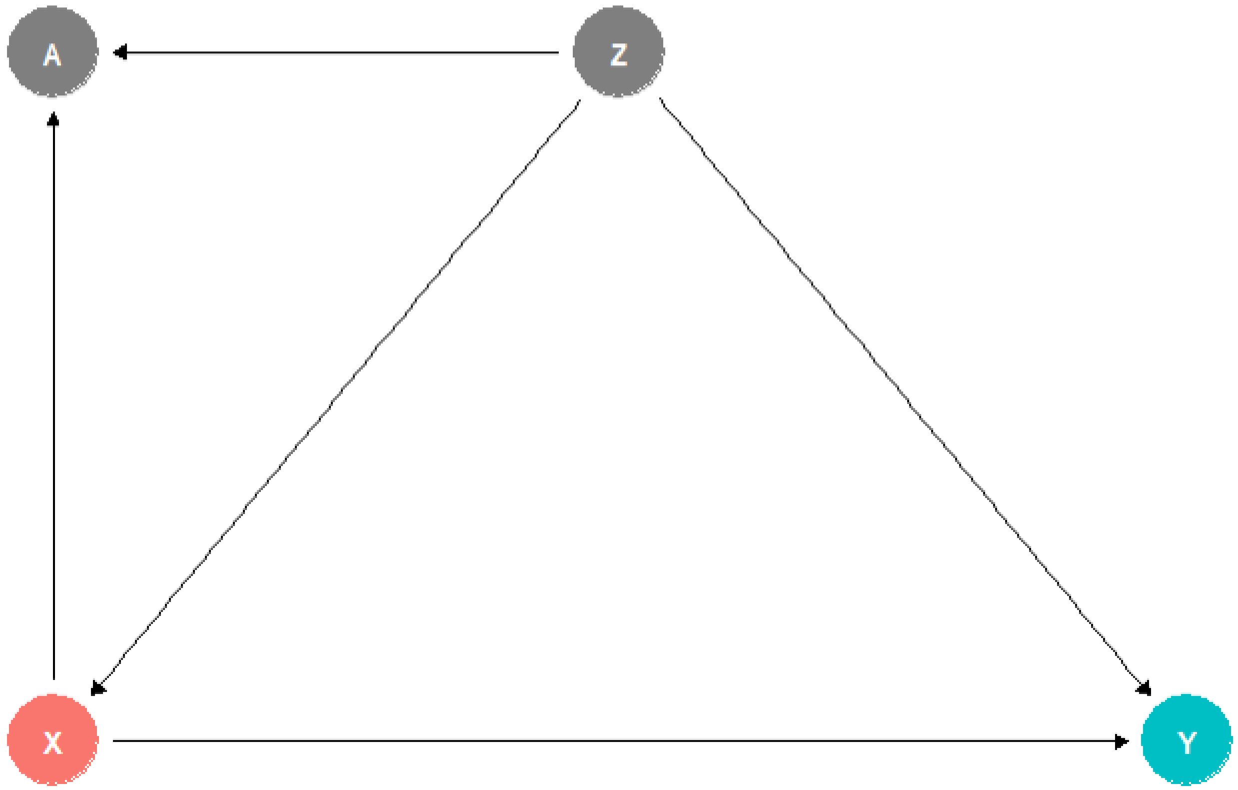
**Backdoor criterion:** Given an ordered pair of variables (X,Y) in a model, a set of confounder variables Z satisfies the backdoor criterion if

1. no confounder variable Z is a **descendent of X** and
2. Z **blocks every path** between X and Y that **contains an arrow into X**.

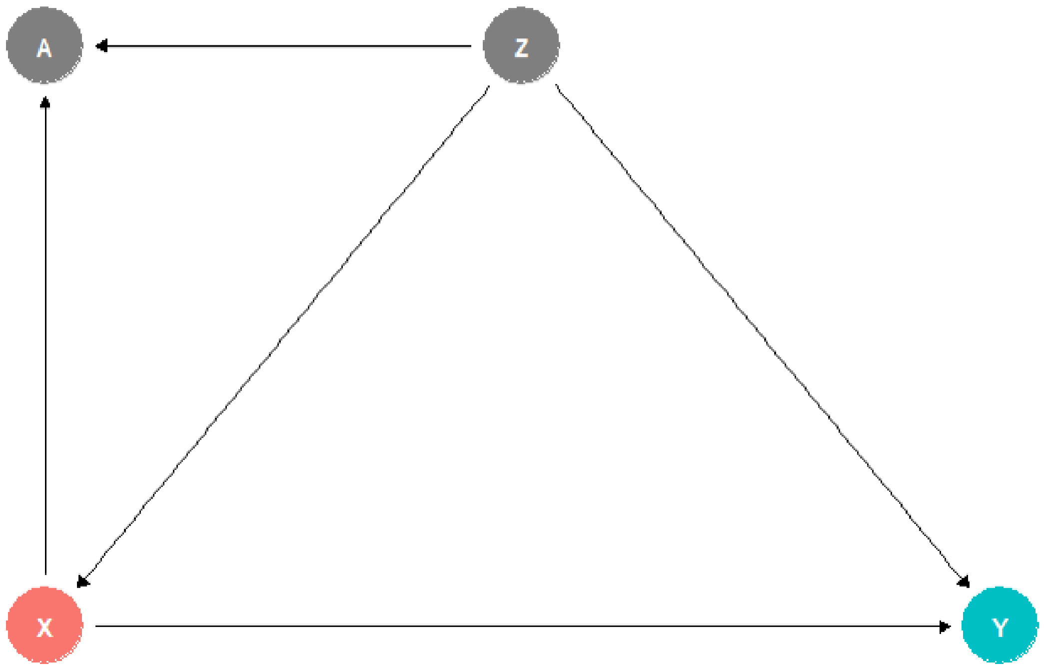
# The Backdoor Adjustment

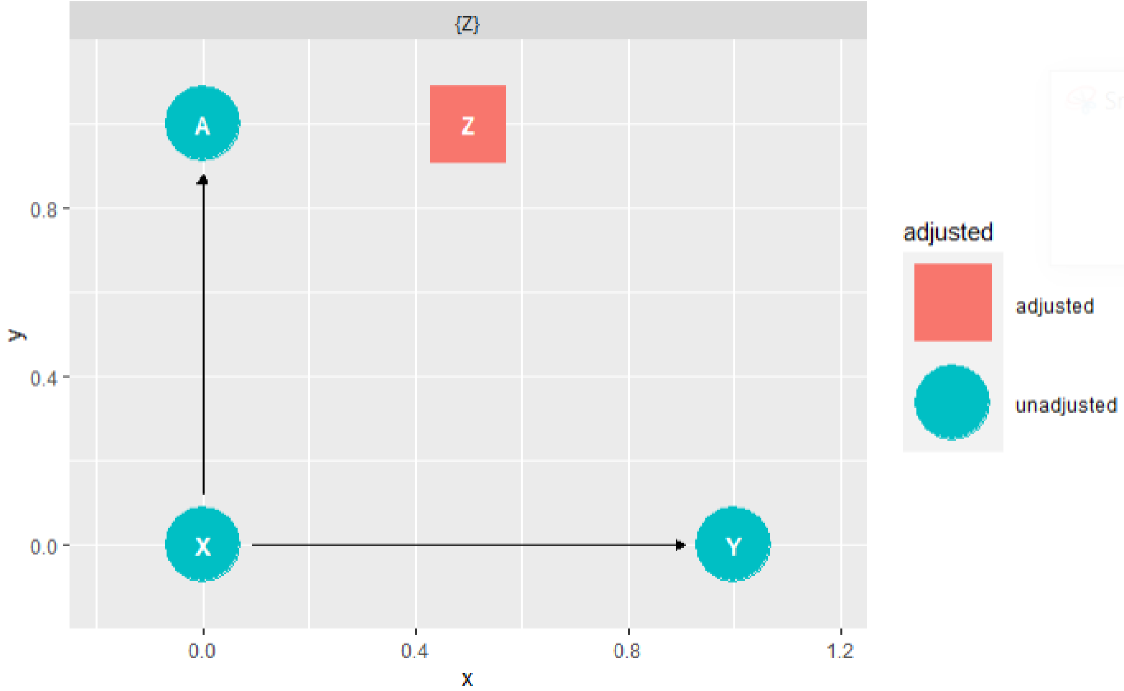



# The Backdoor Adjustment

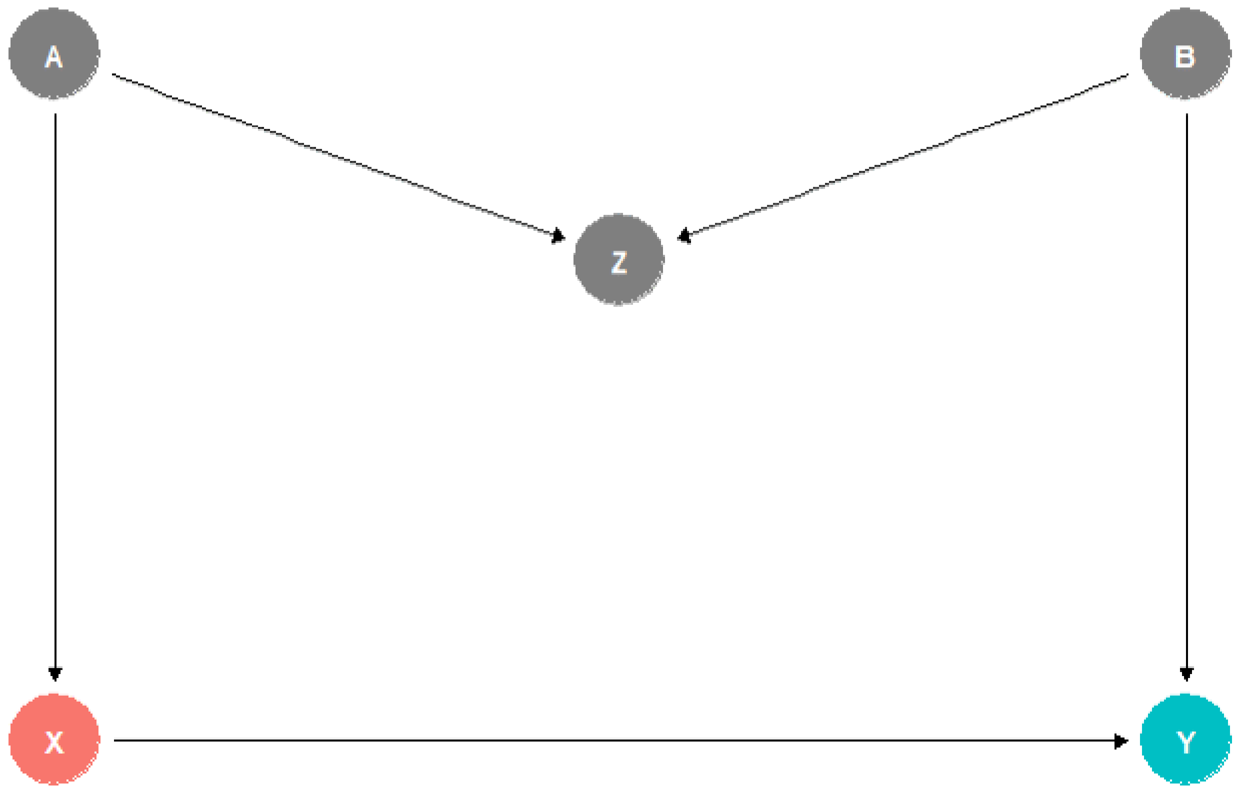


# The Backdoor Adjustment

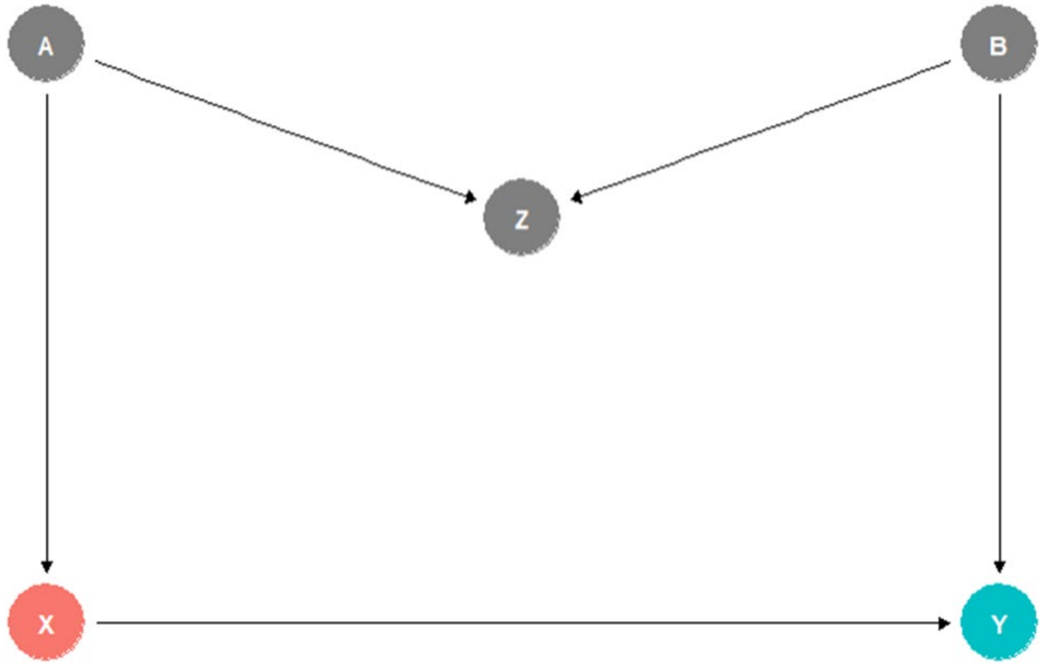


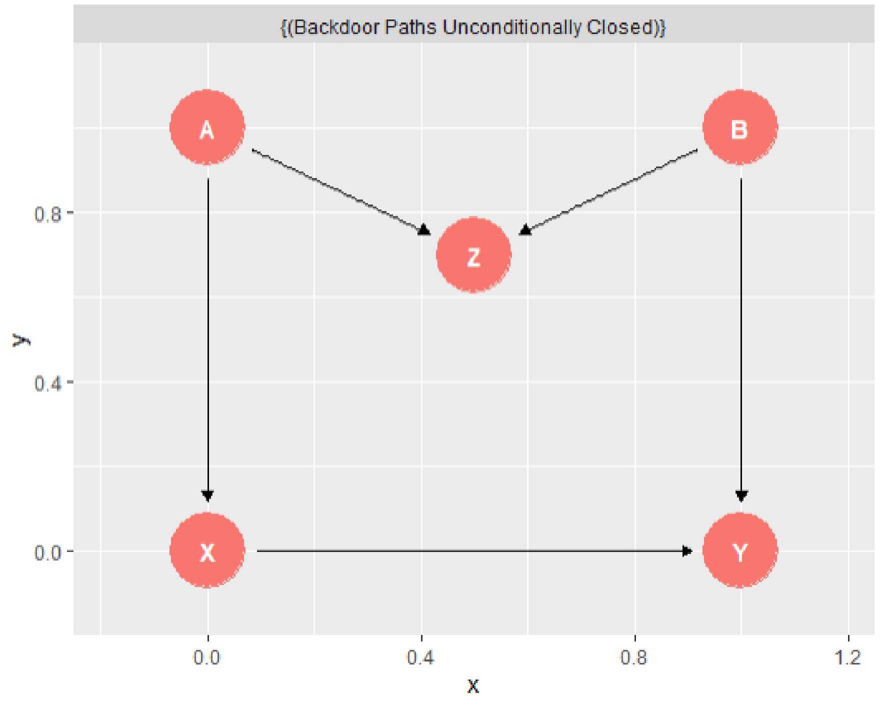



# The Backdoor Adjustment

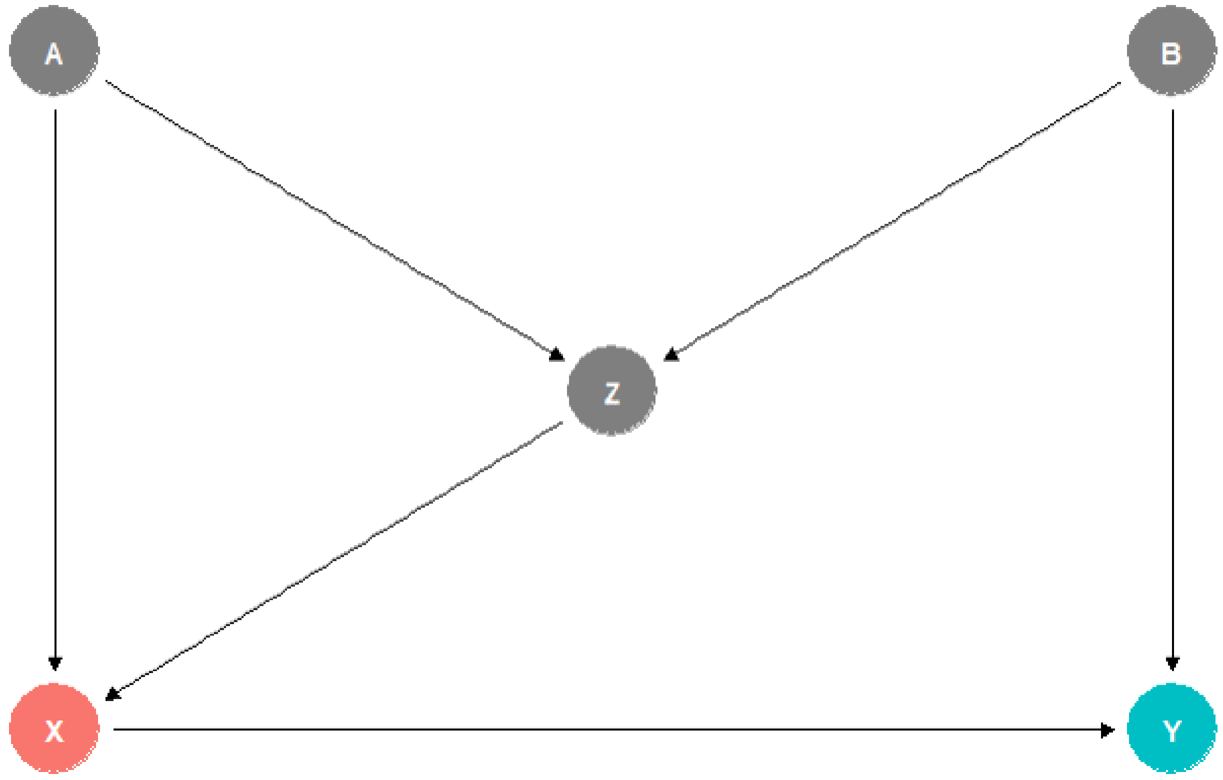


# The Backdoor Adjustment

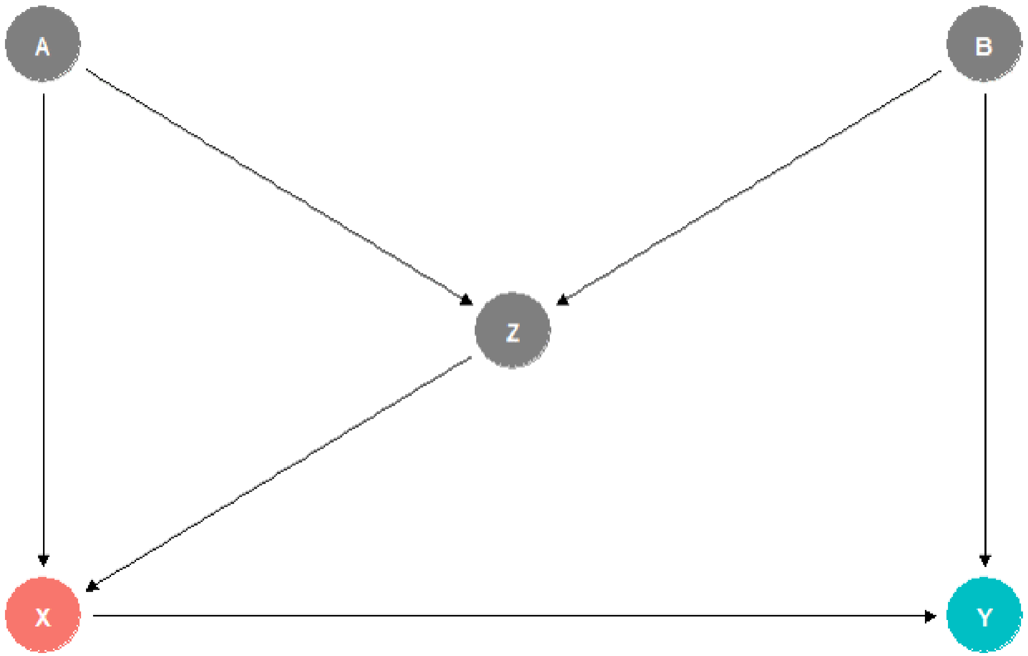


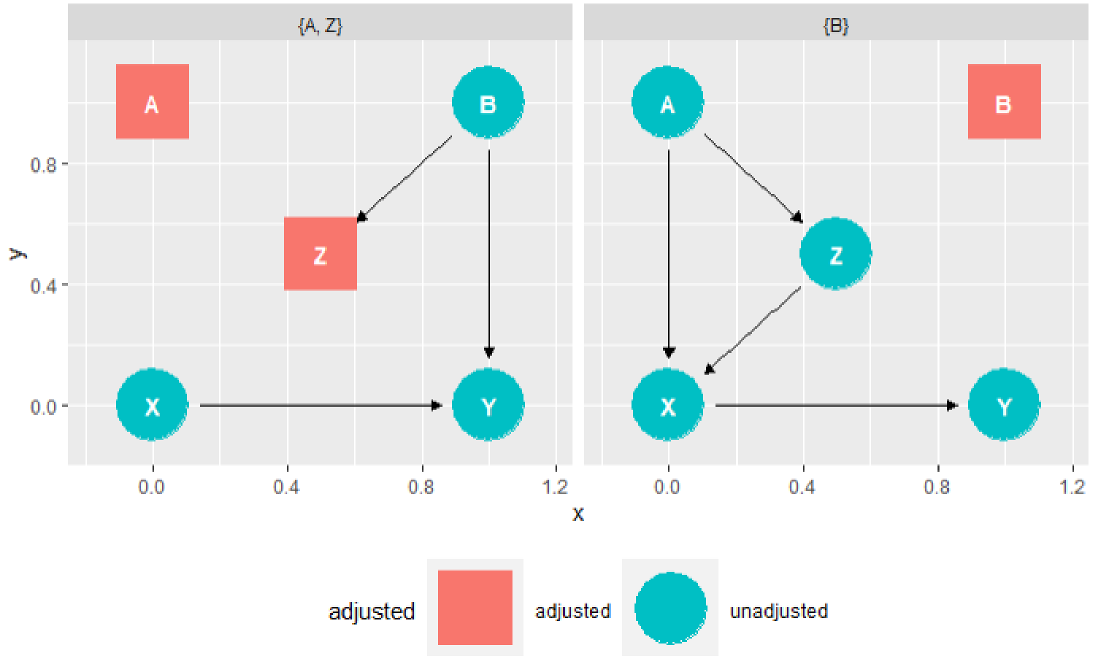



# The Backdoor Adjustment



# The Backdoor Adjustment



# Summary of Part I

Answering causal research questions requires not only data about it, but also **knowledge about the data generation process**

Directed, acyclic graphs make causal **assumptions explicit** and allow us to **systematically analyze** a phenomenon from a causal perspective

The three basic types of associations in causal DAGs are **mediators, forks, and colliders**, and they behave differently when controlled for

Using the backdoor criterion, we can determine **which variables to adjust for** and which to ignore to deconfound the causal relationship of interest

# Frequentist Methods

State of the art for statistical inference in software engineering

# Basics

The basic tool of frequentist methods for data analysis is the **null-hypothesis significance test** (NHST). The basic approach is:

1. Formulate a **null-hypothesis** and alternate hypothesis
2. Select an appropriate **NHST variant**
3. **Stratify** the data by the independent variable
4. Perform the test, i.e., determine if there is a **statistically significant difference** in the distribution of the outcome variable between the strata

The **p-value** represents the probability – under the null-hypothesis – of observing data *at least as extreme* as the ones that were actually observed. If $p < \alpha$ then $h_0$ is an unlikely explanation for the data and it can be rejected.

# Issues

Frequentist methods are under critique for at least the following three reasons.

**Arbitrary significance level**

**Oversimplified summary**

**Unsound extension of the modus tollens**

# Modus tollens in frequentist Analyses

This extension is not sound!

**Modus tollens**

$$\frac{X \to \neg Y \quad Y}{\neg X}$$

If X implies that Y is false, and we observe Y, then X is false.

**Probabilistic extension**

$$\frac{P[Y|X] < \epsilon \quad Y}{P[X] < \epsilon}$$

If X implies that Y is probably false (equivalently: Y is improbably true), and we observe Y, then X is probably false.

**Example 1:**
- X: a person lives in Switzerland
- Y: a person is the King of Sweden

P[Y|X] is probably false, so if we observe Y then P[X] is probably false.

**Example 2:**
- X: a person lives in London
- Y: a person is the Queen of England

P[Y|X] is probably false, **but if we observe Y then P[X] is actually true**.

Furia, C. A., Feldt, R., & Torkar, R. (2019). Bayesian data analysis in empirical software engineering research. *IEEE Transactions on Software Engineering*, *47*(9), 1786-1810.

# Bayesian Data Analysis

For statistical causal inference

# Bayes Theorem

$$P(H|E) = \frac{P(E|H)P(H)}{P(E)}$$

Where

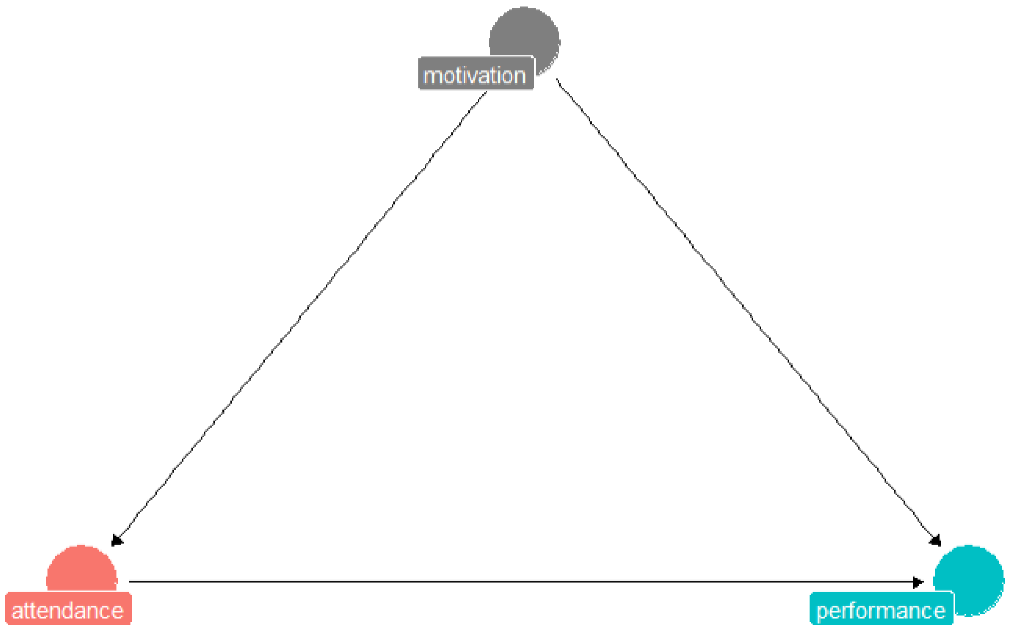
- $P(H|E)$ is the posterior probability, i.e., the probability that hypothesis H is true after observing evidence E,
- $P(E|H)$ is the likelihood, i.e., the probability of observing evidence E given hypothesis H is true,
- $P(H)$ is the prior probability of hypothesis H, and
- $P(E)$ is the marginal likelihood or “model evidence”.

# The Bayesian Data Analysis Approach

1. Define **regression formula**
2. Determine **model distribution**
3. Select **priors** for all included factors
4. Run **prior predictive check**
5. **Fit the model** to the collected data
6. Run **posterior predictive check**
7. Plot **marginal distributions**

# Demonstration of the Bayesian Approach

1. Define **regression formula**




````
```{r formula}
f <- (p ~ a + m)
```
````

# Demonstration of the Bayesian Approach

2. Determine **model distribution**
3. Select **priors** for all included factors

```
```{r prior-types}
get_prior(
  formula = f,
  data = d,
  family = gaussian
)
```
```

```
                 prior     class coef group resp dpar nlpar lb ub       source
                (flat)         b                                      default
                (flat)         b   a1                            (vectorized)
                (flat)         b   m1                            (vectorized)
 student_t(3, 0.2, 2.5) Intercept                                     default
   student_t(3, 0, 2.5)     sigma                         0           default
```

```
```{r priors}
priors <- c(
  prior(normal(0, 1), class = Intercept),
  prior(normal(0, 1), class = b)
)
```
```

# Demonstration of the Bayesian Approach

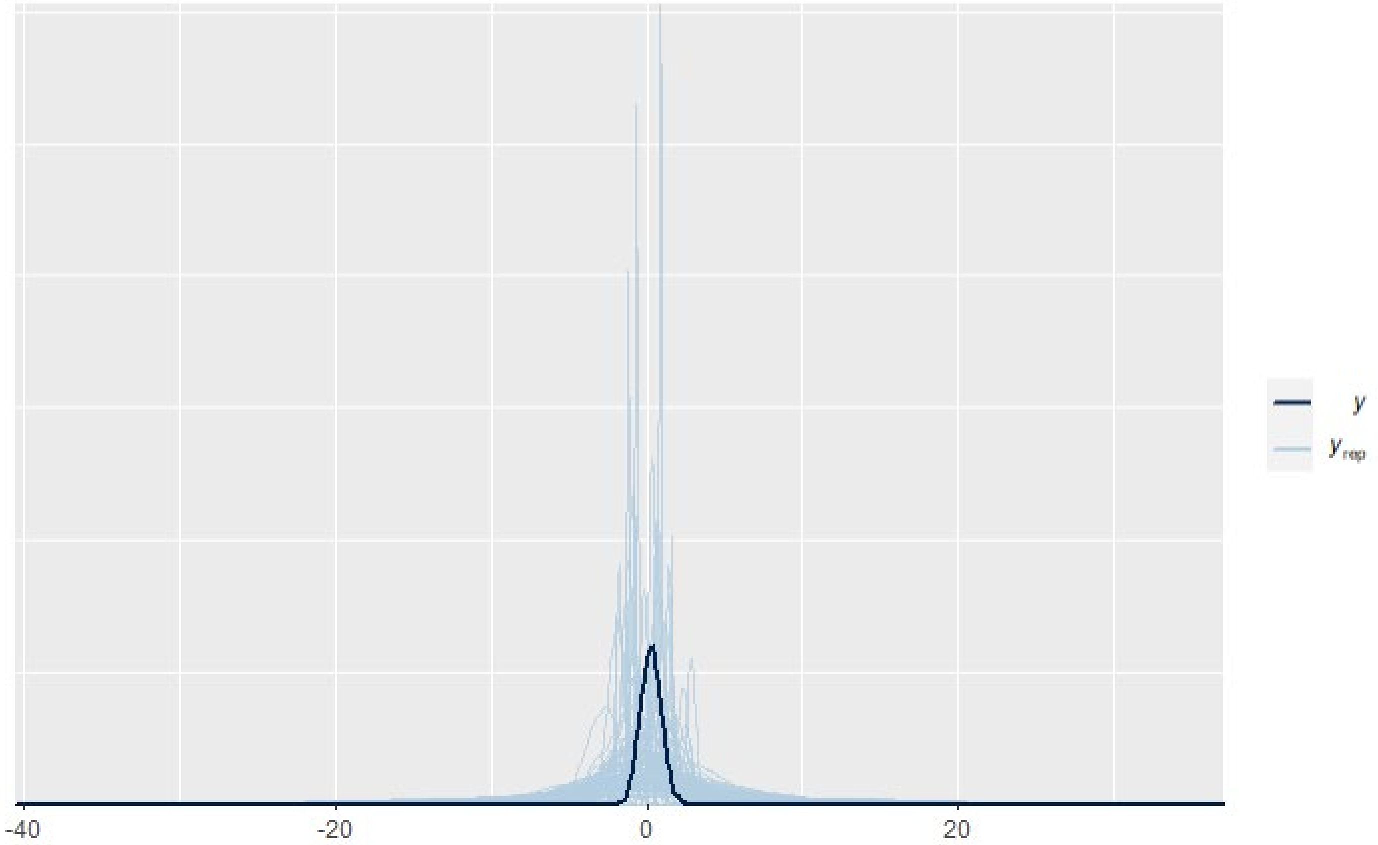
4. Run **prior predictive check**

```
```{r model-prior}
m.prior <-
  brm(
    data = d, # specify the data to train on (despite not necessary for prior predictive checks)
    family = gaussian, # specify the distribution type of the outcome variable
    f, # specify the regression formula
    prior = priors, # specify the priors for each factor in the formula
    iter = 4000, warmup = 1000, chains = 4, cores = 4,
    seed = 4, sample_prior="only",
    file = "fits/m.prior" # specify where to save the pre-compiled model
  )
```
```

# Demonstration of the Bayesian Approach

4. Run **prior predictive check**

```
```{r prior-predictive-check}
ndraws <- 100
brms::pp_check(m.prior, ndraws=ndraws)
```
```



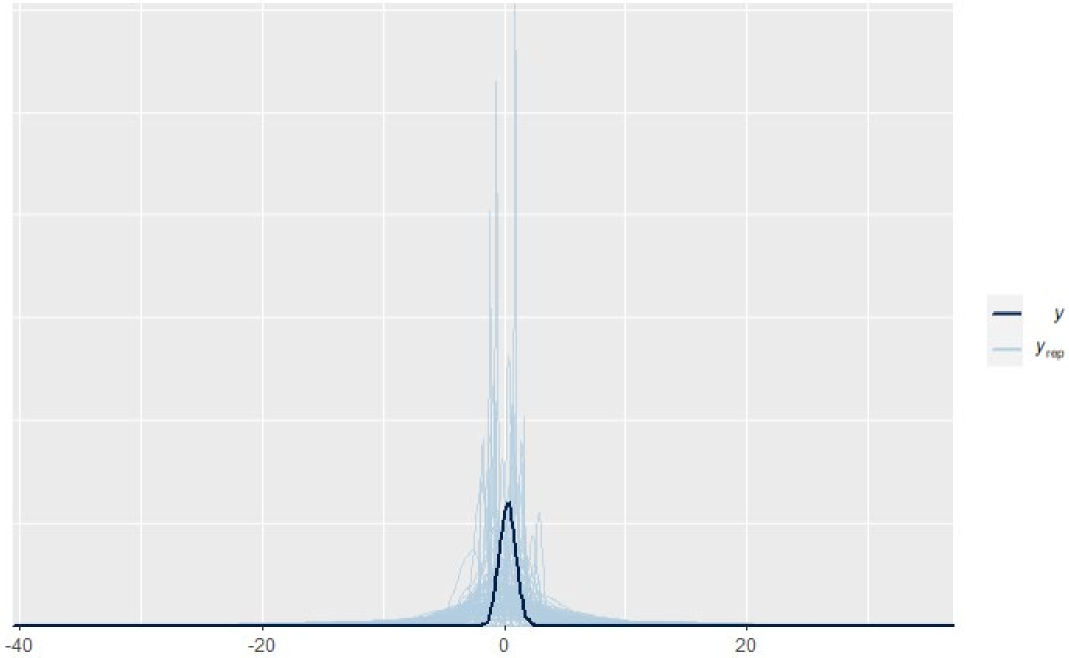
# Demonstration of the Bayesian Approach

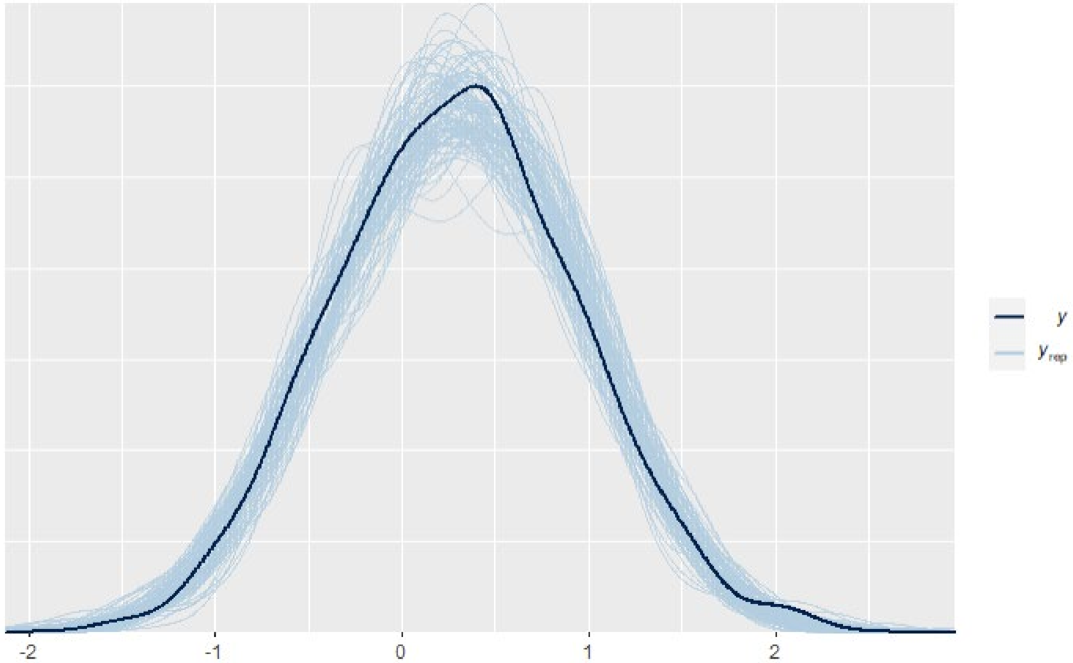
5. **Fit the model** to the collected data

```
```{r model}
m <-
  brm(data = d, family = gaussian, f, prior = priors,
    iter = 4000, warmup = 1000, chains = 4, cores = 4,
    seed = 4,
    file = "fits/m"
  )
```
```

# Demonstration of the Bayesian Approach

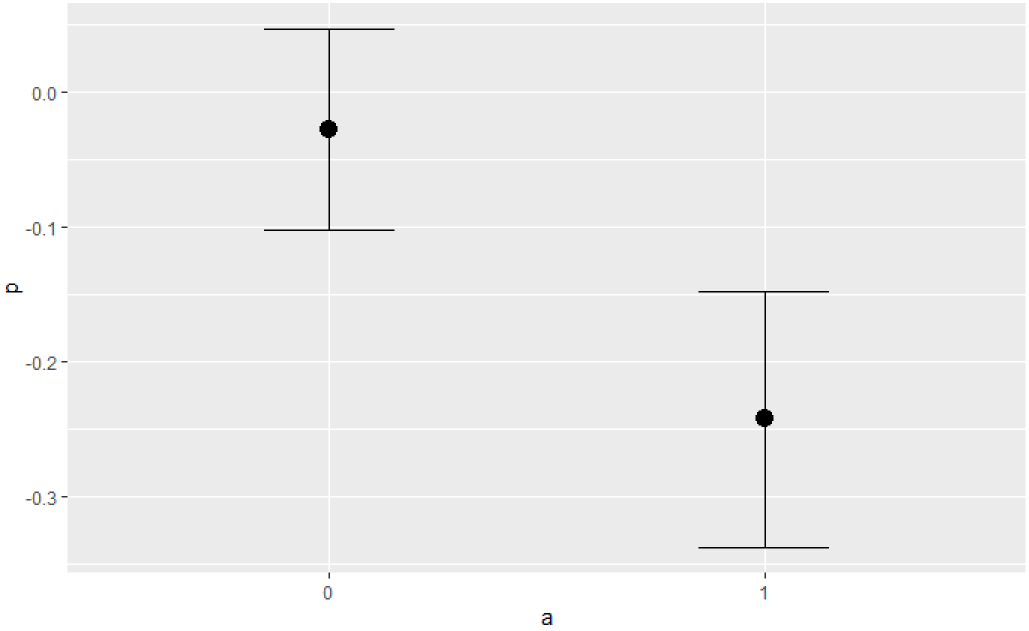
6. Run **posterior predictive check**

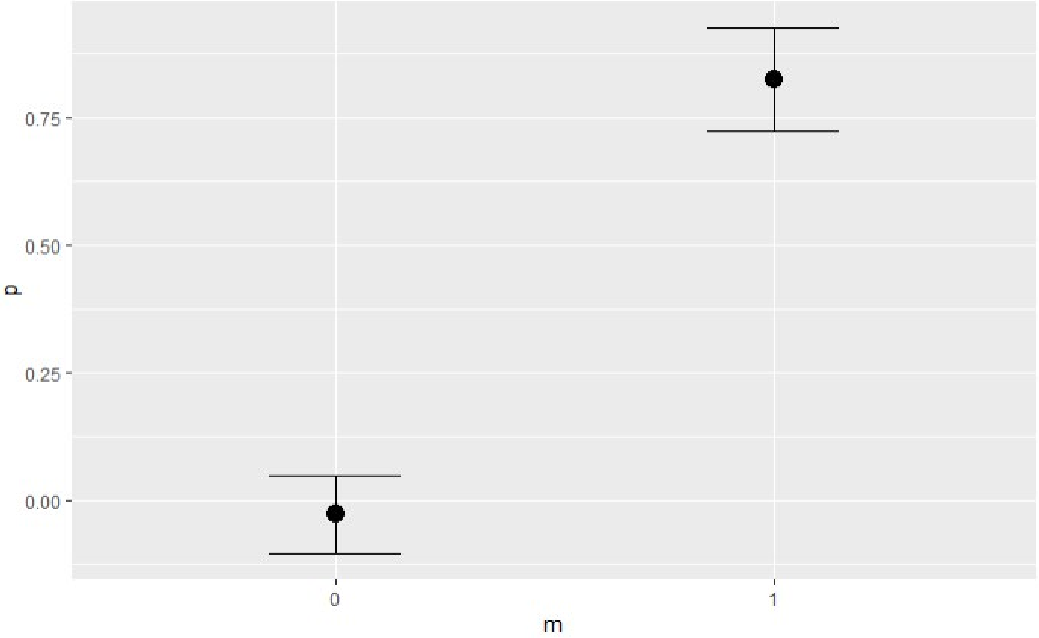


# Demonstration of the Bayesian Approach

7. Plot **marginal distributions**



# Demonstration of the Bayesian Approach

8. Inspect the model summary

```
```{r model-summary}
summary(m)
```

 Family: gaussian
  Links: mu = identity; sigma = identity
Formula: p ~ a + m
   Data: d (Number of observations: 500)
  Draws: 4 chains, each with iter = 4000; warmup = 1000; thin = 1;
         total post-warmup draws = 12000

Population-Level Effects:
          Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
Intercept    -0.03      0.04    -0.10     0.05 1.00    15133     9979
a1           -0.21      0.05    -0.32    -0.11 1.00    11694     9415
m1            0.85      0.05     0.75     0.95 1.00    11202     8346

Family Specific Parameters:
      Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
sigma     0.52      0.02     0.49     0.55 1.00    12008     9140

Draws were sampled using sampling(NUTS). For each parameter, Bulk_ESS
and Tail_ESS are effective sample size measures, and Rhat is the potential
scale reduction factor on split chains (at convergence, Rhat = 1).
```

# Bayesian Data Analysis for Statistical Causal Inference

# General Framework

**Modelling:** Draw a causal DAG around the phenomenon of interest to make your assumptions explicit and discussable.

**Identification:** Apply the backdoor criterion to select, which variables you need to control in order to deconfound the phenomenon of interest.

**Estimation:** If all deconfounders are observable, collect data about the relevant variables and perform a Bayesian data analysis to estimate the causal effect.

Pearl, J. (2009). Causality. Cambridge university press.
Siebert, J. (2023). Applications of statistical causal inference in software engineering. *Information and Software Technology*, *159*, 107198.

# Reading List


**Pearl**, J., & Mackenzie, D. (2018). *The book of why: the new science of cause and effect*. Basic books.

**McElreath**, R. (2018). *Statistical rethinking: A Bayesian course with examples in R and Stan*. Chapman and Hall/CRC.

**Siebert**, J. (2023). Applications of statistical causal inference in software engineering. *Information and Software Technology*, *159*, 107198.

**Furia**, C. A., **Feldt**, R., & **Torkar**, R. (2019). Bayesian data analysis in empirical software engineering research. *IEEE Transactions on Software Engineering*, *47*(9), 1786-1810.

**Furia**, C. A., **Torkar**, R., & **Feldt**, R. (2022). Applying Bayesian analysis guidelines to empirical software engineering data: The case of programming languages and code quality. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, *31*(3), 1-38.