

# EPI66 - Tópicos de Pesquisa I

## Uso de DAGs para a identificação de confundidores na pesquisa em saúde

Ricardo de Souza Kuchenbecker

Rodrigo Citton P. dos Reis - [citton.padilha@ufrgs.br](mailto:citton.padilha@ufrgs.br)

UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL  
PROGRAMA DE PÓS-GRADUAÇÃO EM EPIDEMIOLOGIA

Porto Alegre, 2023

## Relembrando

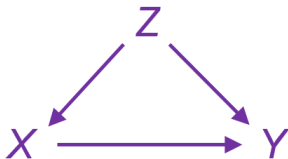
# Como duas variáveis podem estar associadas?



- ▶  $X$  e  $Y$  serão **associadas na população** se:
  - ▶  $X$  causa  $Y$ .
  - ▶  $Y$  causa  $X$ .
  - ▶ existe uma  $Z$  que é causa comum de  $X$  e  $Y$ .
- ▶  $X$  e  $Y$  serão **associadas em subpopulações definadas por  $Z$**  se  $Z$  é um **efeito** de  $X$  e  $Y$ .

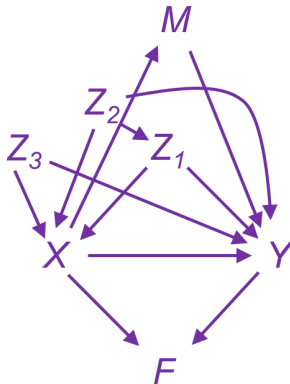
## Como duas variáveis podem estar associadas?

- O que podemos concluir do diagrama abaixo?



## Como duas variáveis podem estar associadas?

- E neste outro caso? O que podemos concluir?



# Grafos acíclicos dirigidos

## grafos acíclicos dirigidos (DAGs)

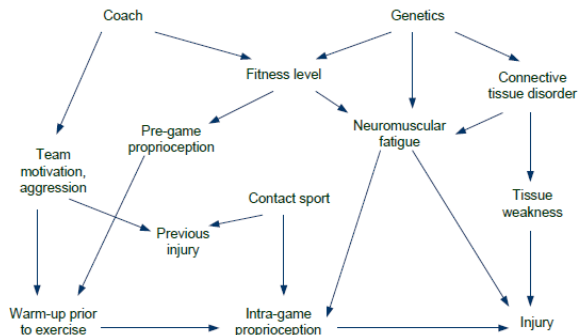
(ou **diagramas causais**), oferecem uma linguagem para especificação de relações (causais) entre as variáveis do quadro conceitual.

## Critério back-door

(1) verifica a existência de **viés de confusão**; (2) em caso afirmativo, verifica a existência de um conjunto de variáveis **suficiente para o controle** (ajuste) do viés de confusão.

# Ferramentas computacionais para a construção de DAGs

## Um exemplo



- A análise de DAGs pode ser tediosa na prática, e se presta bem à automatização por um programa de computador.



# DAGitty

## Welcome to DAGitty!

Launch



[Launch DAGitty  
online in your  
browser](#)

Download



[Download  
DAGitty's source  
for offline use](#)

Learn



[Learn more about  
DAGs and  
DAGitty](#)

Code



The R package  
"dagitty" is  
available on  
[CRAN](#) or [github](#)

# DAGitty

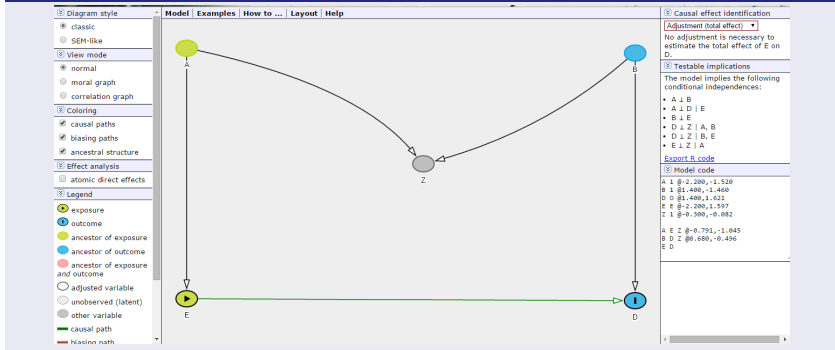
- ▶ O DAGitty é um ambiente baseado em navegador para criar, editar e analisar modelos causais (DAGs).
  - ▶ O foco está no uso de diagramas causais para minimizar o viés em estudos empíricos em epidemiologia e outras disciplinas.



- ▶ O DAGitty é desenvolvido e mantido por **Johannes Textor** (Tumor Immunology Lab and Institute for Computing and Information Sciences, Radboud University Nijmegen).
- ▶ <http://dagitty.net/>

# DAGitty

## Uma visão geral



# DAGitty

## Uma visão geral

The screenshot displays the DAGitty software interface. The central area shows a causal diagram with nodes A (yellow), B (blue), Z (grey), E (yellow with a play button), and D (blue with an exclamation mark). Edges include A → Z, B → Z, A → E, and E → D. The interface is divided into several panels:

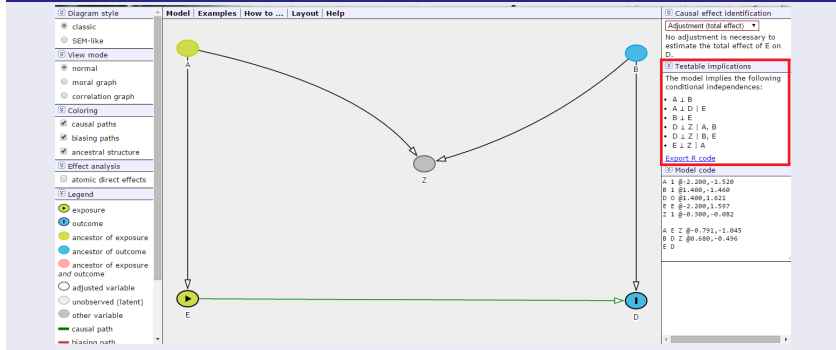
- Left Panel:** Contains settings for Diagram style (classic, SEM-like), View mode (normal, moral graph, correlation graph), Coloring (causal paths, biasing paths, ancestral structure), Effect analysis (atomic direct effects), and Legend (exposure, outcome, ancestor of exposure, ancestor of outcome, ancestor of exposure and outcome, adjusted variable, unobserved (latent), other variable, causal path, biasing path).
- Top Panel:** Includes tabs for Model, Examples, How to ..., Layout, and Help.
- Right Panel:**
  - Causal effect identification:** Shows a dropdown menu set to "Adjustment (total effect)" with a note: "No adjustment is necessary to estimate the total effect of E on D."
  - Testable Implications:** Lists conditional independences implied by the model:
    - $A \perp B$
    - $A \perp D \mid E$
    - $B \perp E$
    - $D \perp Z \mid A, B$
    - $D \perp Z \mid B, E$
    - $E \perp Z \mid A$
  - Export R code:** A button to export the model to R code.
  - Model code:** Displays the model's parameters:
 

```
A 1 @-2.200,-1.520
B 1 @1.400,-1.460
D 0 @1.400,1.621
E 1 @-2.200,1.597
Z 1 @-0.300,-0.082

A E Z @-0.791,-1.045
B D Z @0.680,-0.496
E D
```

# DAGitty

## Uma visão geral



# DAGitty

## Uma visão geral

The screenshot displays the DAGitty web application interface. The central area shows a causal diagram with six nodes: A (red circle), B (red circle), C (red circle), E (yellow circle with a black dot), D (blue circle with a black dot), and F (grey circle). Edges include a red curved arrow from A to C, a red straight arrow from B to C, a red straight arrow from A to E, a red straight arrow from B to D, a green straight arrow from E to D, and black straight arrows from E and D to F. A legend on the left defines node types and path colors. The right sidebar contains analysis results.

**Diagram style**

- classic
- SEM-like

**View mode**

- normal
- moral graph
- correlation graph

**Coloring**

- causal paths
- biasing paths
- ancestral structure

**Effect analysis**

- atomic direct effects

**Legend**

- exposure
- outcome
- ancestor of exposure
- ancestor of outcome
- ancestor of exposure and outcome
- adjusted variable
- unobserved (latent)
- other variable
- causal path
- biasing path

**Model** | Examples | How to ... | Layout | Help

**Causal effect identification**

Adjustment (total effect)

Minimal sufficient adjustment sets for estimating the total effect of E on D:

- A, C
- B, C

**Testable implications**

The model implies the following conditional independences:

- $A \perp B$
- $A \perp D \mid B, C, E$
- $A \perp F \mid D, E$
- $A \perp F \mid B, C, E$
- $B \perp E \mid A, C$
- $B \perp F \mid D, E$
- $C \perp F \mid D, E$

[Export R code](#)

**Model code**

```
A 1 @-2.200,-1.520
B 1 @1.400,-1.460
C 1 @-0.300,-0.082
D 0 @1.400,0.841
E 2 @-2.205,0.841
F 1 @-0.361,1.816

A C @-0.791,-1.045 E
B C @0.680,-0.496 D
C D E
```

## Criando um DAG no DAGitty

- ▶ No menu **Model**, clique em **New model**.
- ▶ O DAGitty irá solicitar o nome da variável de **exposição**, e logo em seguida o nome da variável de **desfecho**.
  - ▶ Estas variáveis serão criadas no grafo com o caminho  $E \rightarrow D$ .
- ▶ Para acrescentar uma nova variável dê um **duplo-clique** na área do grafo e dê um nome para esta variável.
- ▶ Para especificar uma relação entre duas variáveis, dê um duplo-clique na **variável de origem** e um duplo-clique na **variável de destino**.
  - ▶ O mesmo procedimento serve para remover uma relação entre duas variáveis já existente.
- ▶ Para remover uma variável do grafo, clique sobre a variável pressionando a **tecla D**.
- ▶ Para renomear uma variável, clique sobre a variável pressionando a **tecla R**.
- ▶ Na dúvida, consulte os menus **How to ...** e **Help**.

# Criando um DAG no DAGitty

- ▶ No menu **Model** é possível:
  - ▶ exportar o grafo
  - ▶ publicar o grafo
- ▶ Uma vez publicado, o grafo pode ser:
  - ▶ modificado
  - ▶ apagado



## Atividade 2

## Atividade 2

Atividade em pequenos grupos.

1. Utilize o DAGitty para construir o DAG do “**exemplo das catecolaminas**”.
  - ▶ Verifique se as conclusões obtidas concordam com as que você obteve com o critério back-door “feito a mão”.
2. Utilize o DAGitty para construir um DAG do quadro conceitual relacionado à sua questão de pesquisa no mestrado ou doutorado.

## DAGitty encontra o R

- ▶ Uma versão do software baseado na web 'DAGitty', disponível em <http://dagitty.net>, para análise de modelos causais estruturais (também conhecidos como gráficos acíclicos direcionados ou DAGs).
- ▶ Este pacote calcula conjuntos de ajuste de covariáveis para estimar efeitos causais, enumera variáveis instrumentais, deriva **implicações testáveis** (d-separação), gera modelos equivalentes e inclui um recurso simples para simulação de dados.

# DAGitty encontra o R

```
# Instala o pacote dagitty  
# install.packages("dagitty")  
  
# Carrega o pacote dagitty  
library(dagitty)  
  
# Carrega o DAG a partir do DAGitty  
dag1 <- downloadGraph(x = "dagitty.net/mBYpOXW")
```

# DAGitty encontra o R

```
plot(dag1)
```



## DAGitty encontra o R

```
dag1.bd <- backDoorGraph(dag1)  
plot(dag1.bd)
```



## DAGitty encontra o R

```
print(adjustmentSets(dag1,  
                     effect = "total"))
```

```
## { ESTRESSE, IDADE, TABAGISMO }
```

```
# impliedConditionalIndependencies(dag1,
```

```
#                               max.results = 2)
```

## ggdag: dagitty encontra o ggplot2

- ▶ O `ggdag` estende o pacote `dagitty` para funcionar no contexto do *tidyverse*.
- ▶ Ele usa os algoritmos do `dagitty` para analisar DAGs para produzir resultados organizados, que podem ser usados no `ggplot2` e no `ggraph` e manipulados com outras ferramentas do *tidyverse*, como `dplyr`.

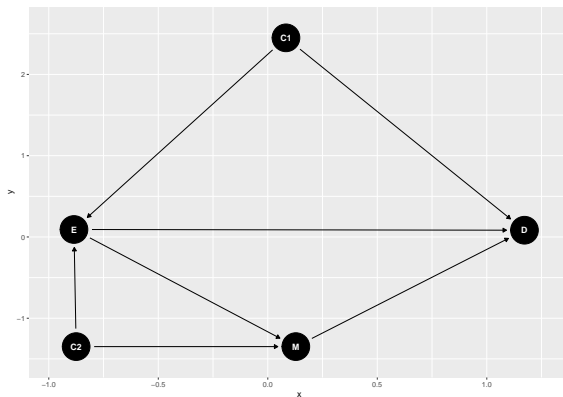


## ggdag: dagitty encontra o ggplot2

```
# Instala o pacote ggdag  
# install.packages("ggdag")  
  
# Carrega o pacote ggdag  
library(ggdag)  
  
# Carrega o DAG a partir do DAGitty  
dag2 <- downloadGraph(x = "dagitty.net/mQLajCg")
```

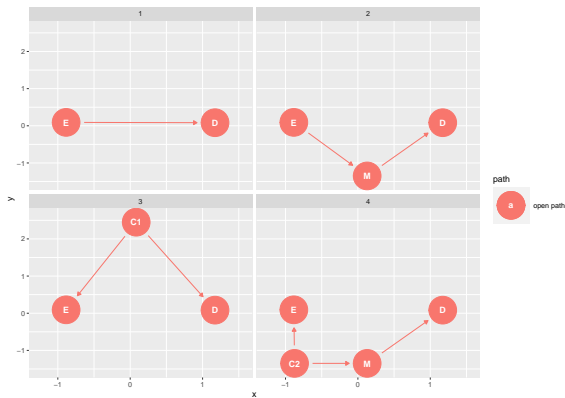
## ggdag: dagitty encontra o ggplot2

```
ggdag(dag2)
```



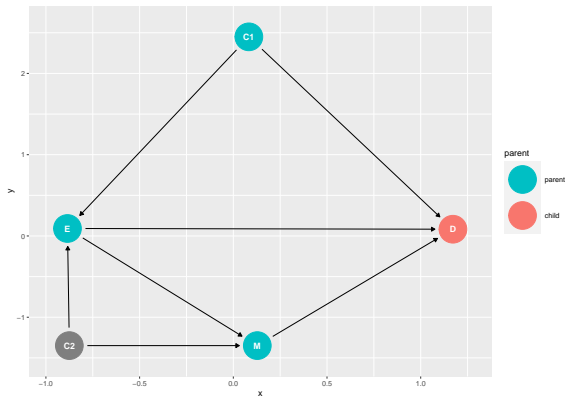
## ggdag: dagitty encontra o ggplot2

```
ggdag_paths(dag2)
```



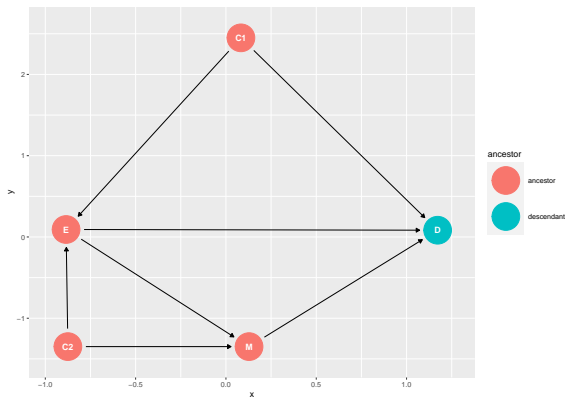
## ggdag: dagitty encontra o ggplot2

```
ggdag_parents(dag2, "D")
```



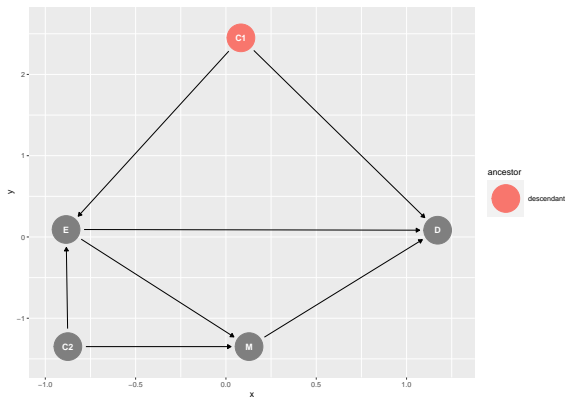
## ggdag: dagitty encontra o ggplot2

```
ggdag_ancestors(dag2, "D")
```



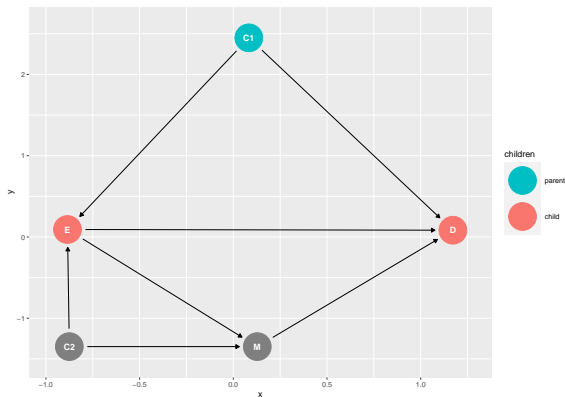
## ggdag: dagitty encontra o ggplot2

```
ggdag_ancestors(dag2, "C1")
```



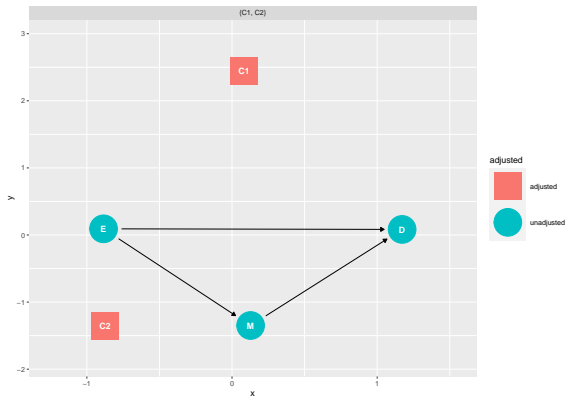
## ggdag: dagitty encontra o ggplot2

```
ggdag_children(dag2, "C1")
```



## ggdag: dagitty encontra o ggplot2

```
ggdag_adjustment_set(dag2)
```





## Bons estudos!

