Almost-Matching-Exactly for Treatment Effect Estimation under Network Interference

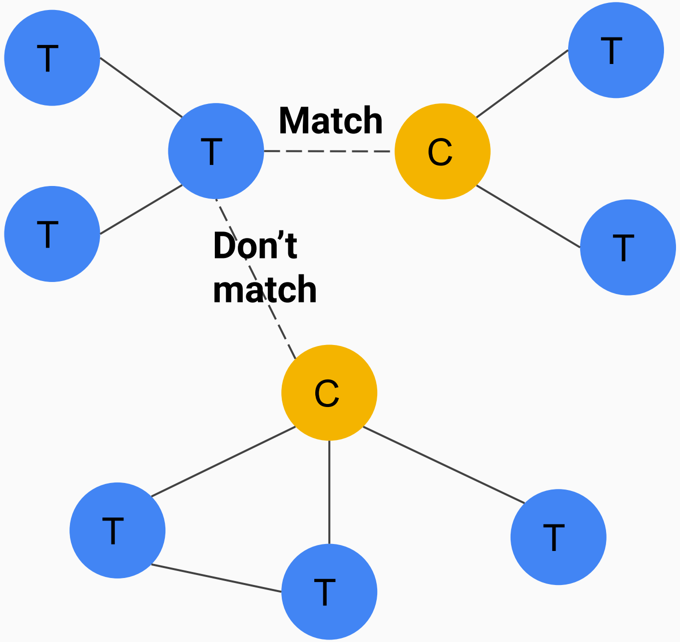
M. Usaid Awan, **Marco Morucci**, Vittorio Orlandi, Sudeepa Roy, Cynthia Rudin, Alexander Volfovsky

Duke University

FLAME-Networks

Results from randomized experiments on networks are biased.

We correct this bias by comparing units with similar connections.



Causal Inference

Goal: Want to know the effect of a **treatment** on an **outcome** for a set of **units.**

Causal Inference

Goal: Want to know the effect of a **treatment** on an **outcome** for a set of **units.**

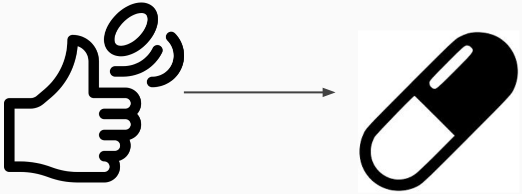
~2 Conditions

Causal Inference

Goal: Want to know the effect of a **treatment** on an **outcome** for a set of **units.**

~2 Conditions

Randomized treatment

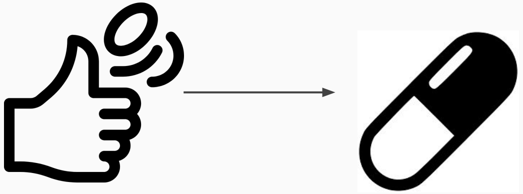


Causal Inference

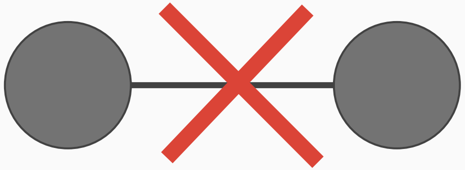
Goal: Want to know the effect of a **treatment** on an **outcome** for a set of **units.**

~2 Conditions

Randomized treatment



No Interference



Causal Inference

Does assigning students to an additional class raise their test scores?

Causal Inference

Does assigning students to an additional class raise their test scores?

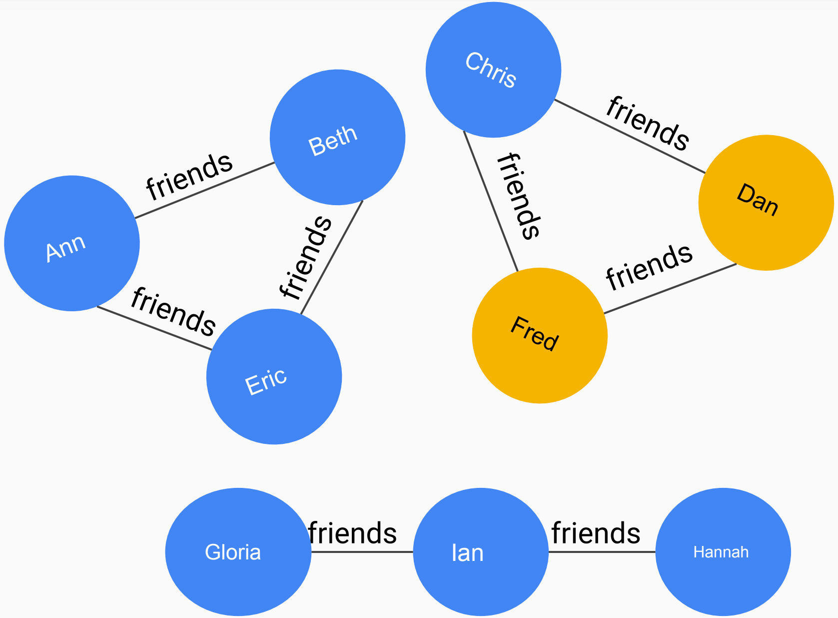
Perform an experiment: Randomly assign some students to take the class, then administer tests

Causal Inference

Does assigning students to an additional class raise their test scores?

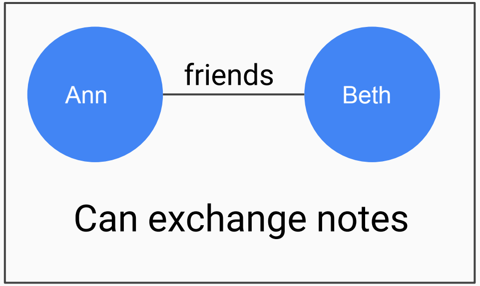
Perform an experiment: Randomly assign some students to take the class, then administer tests

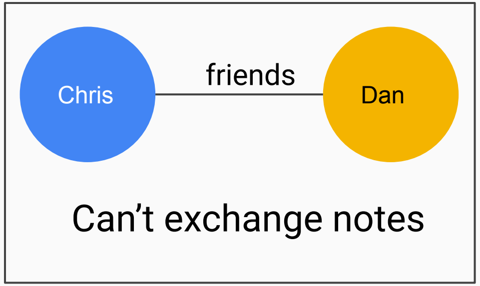
But: Students in our settings are friends



Networks Matter!

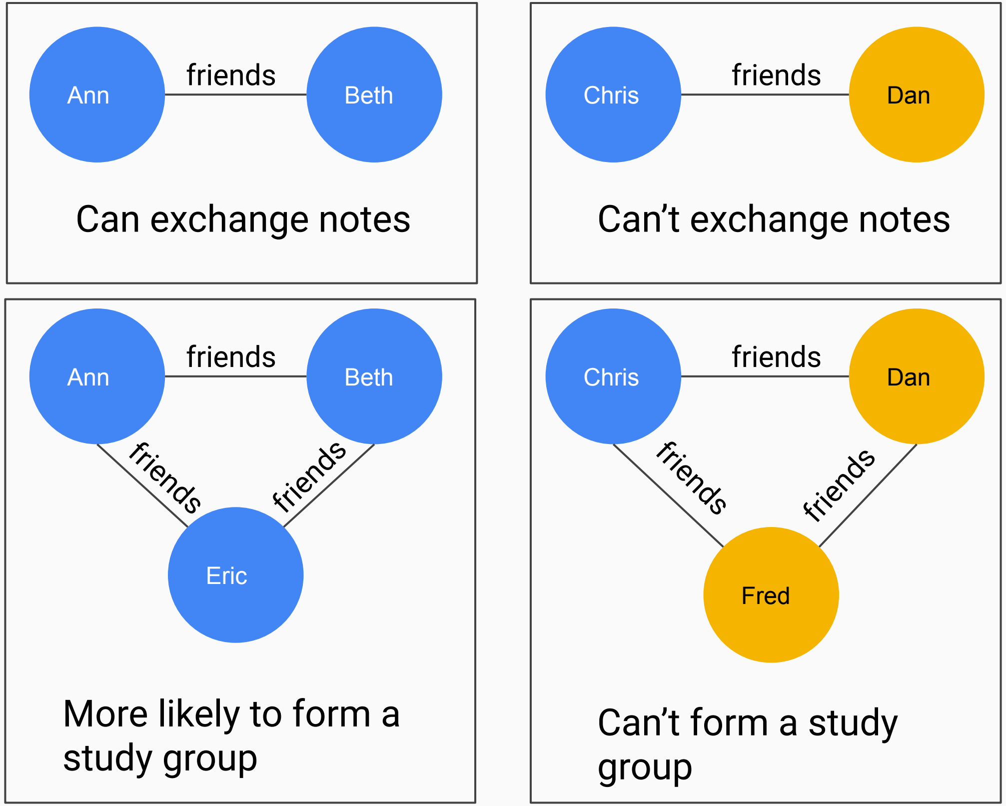
Treated students with treated friends and treated students without treated friends have different experiences.





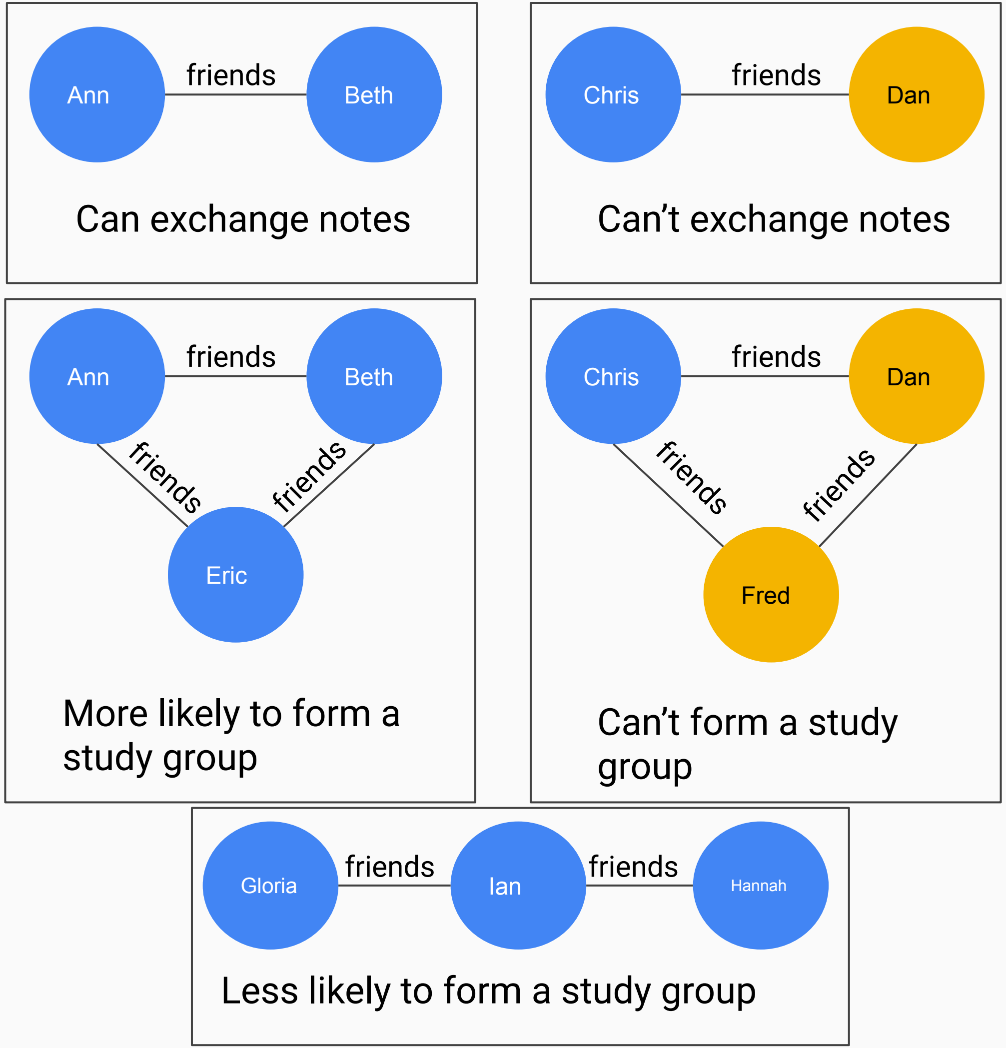
Networks Matter!

Treated students with treated friends and treated students without treated friends have different experiences.



Networks Matter!

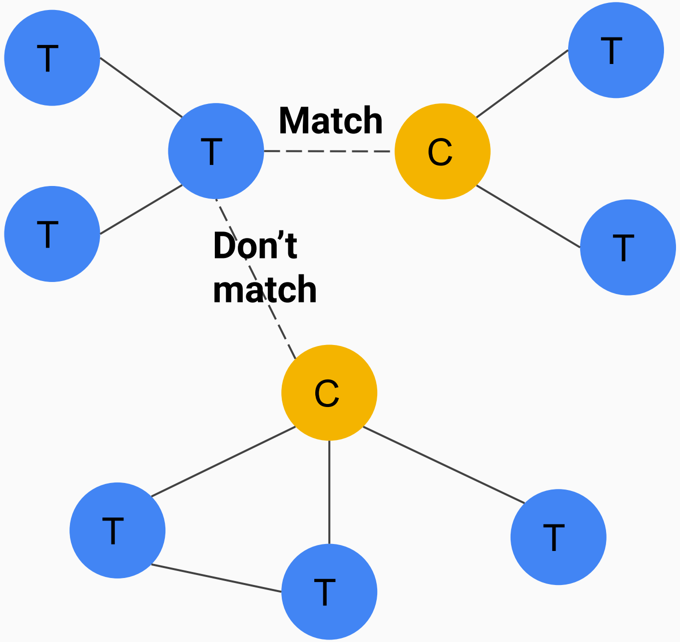
Treated students with treated friends and treated students without treated friends have different experiences.



FLAME-Networks

Traditional Causal inference can’t handle interference.

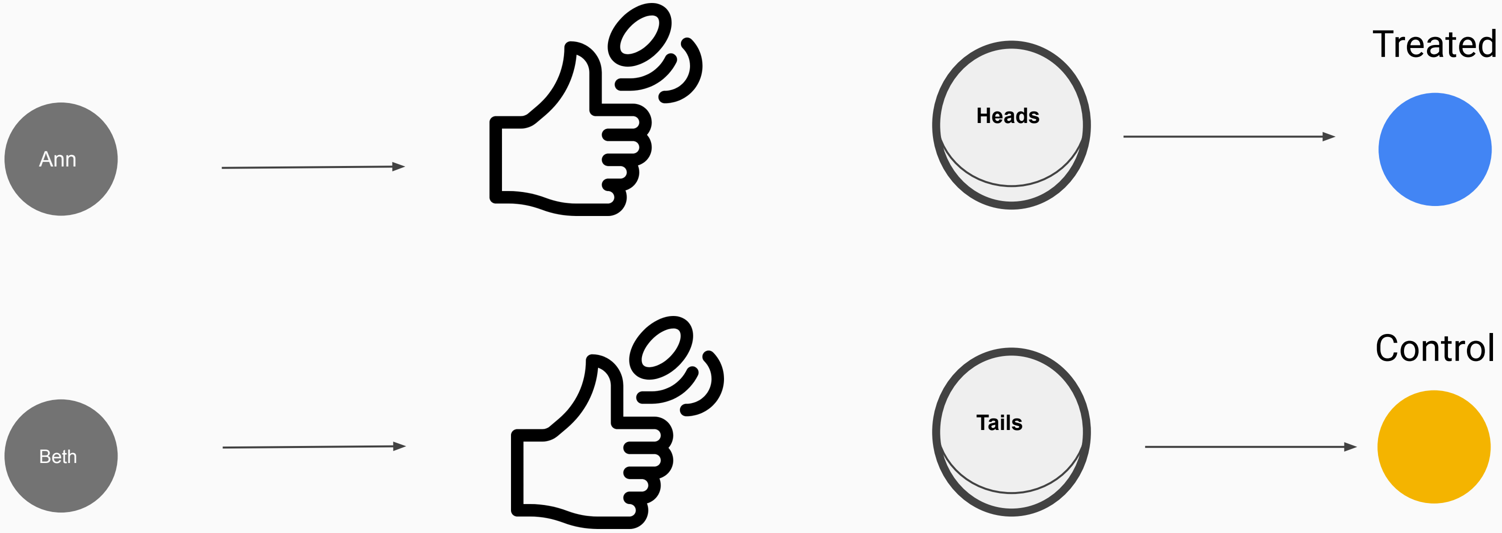
FLAME-Networks uses matching to address this issue.



FLAME-Networks: Causal Inference with Interference

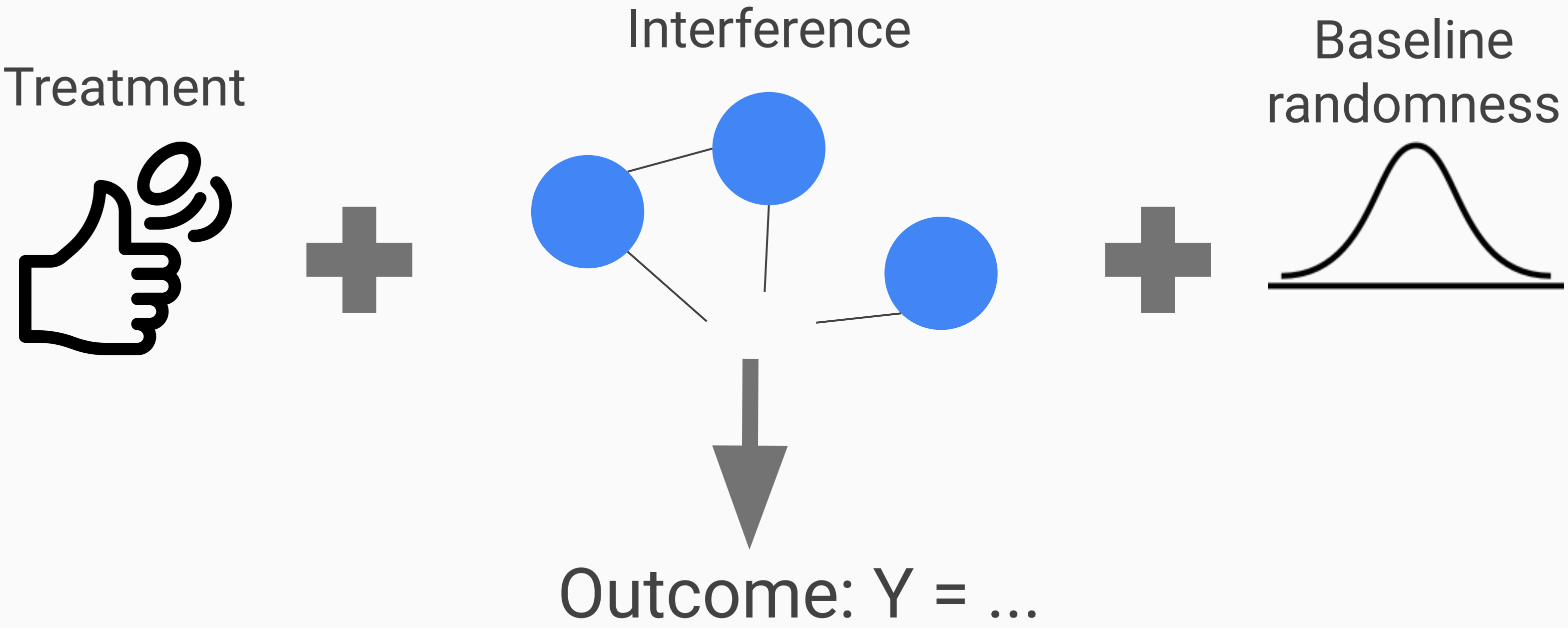
The AME-Networks Assumptions

A0) Ignorability (randomized experiment): Treatment is assigned independently of potential outcomes.



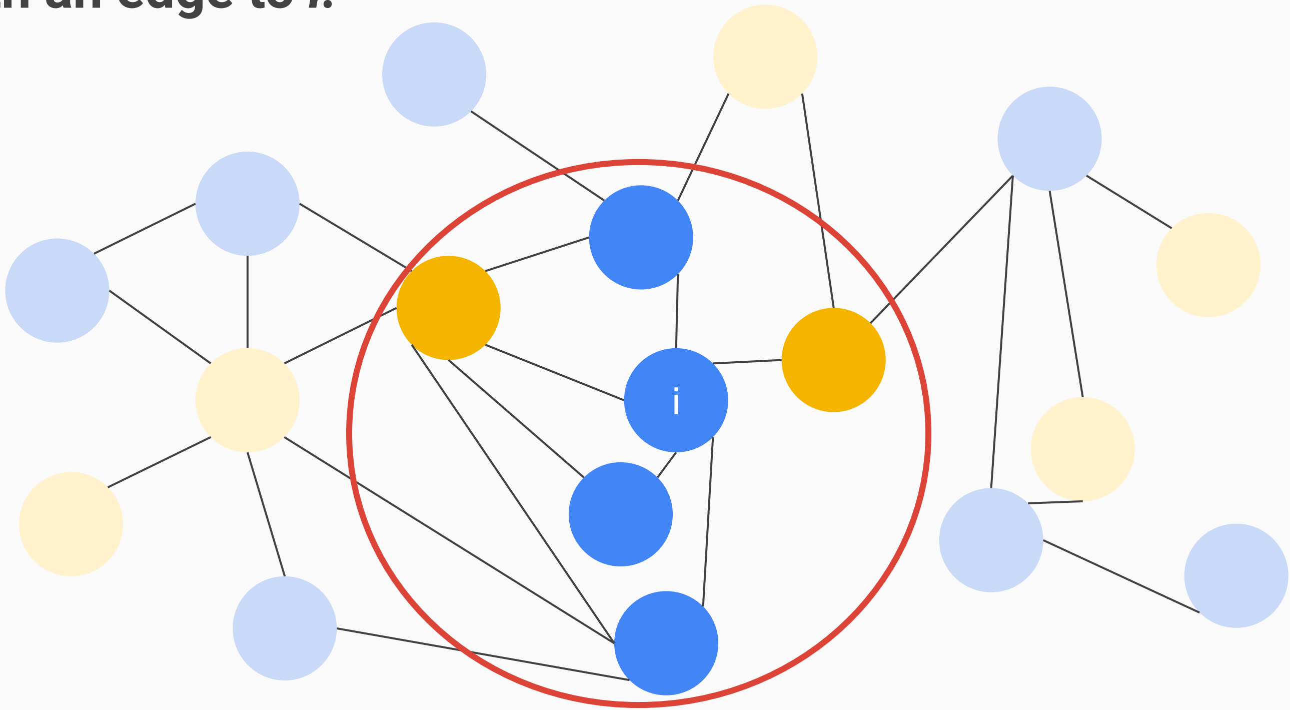
The AME-Networks Assumptions: 2

A1) Additivity of main effects: Potential outcomes are an additive function of treatment effects and interference.



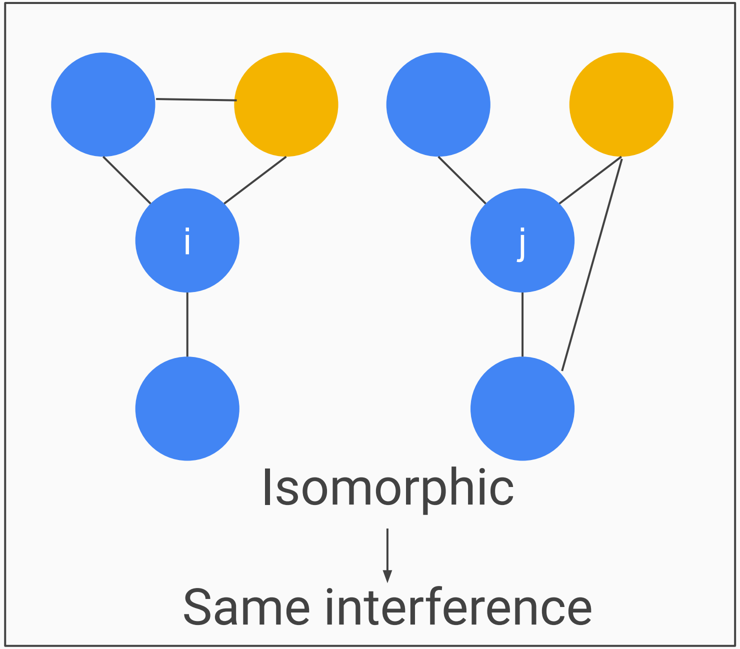
The AME-Networks Assumptions: 3

A2) Neighborhood interference: A unit's interference function depends only on the treatment indicators of the units in its neighborhood, i.e., **units with an edge to *i*.**



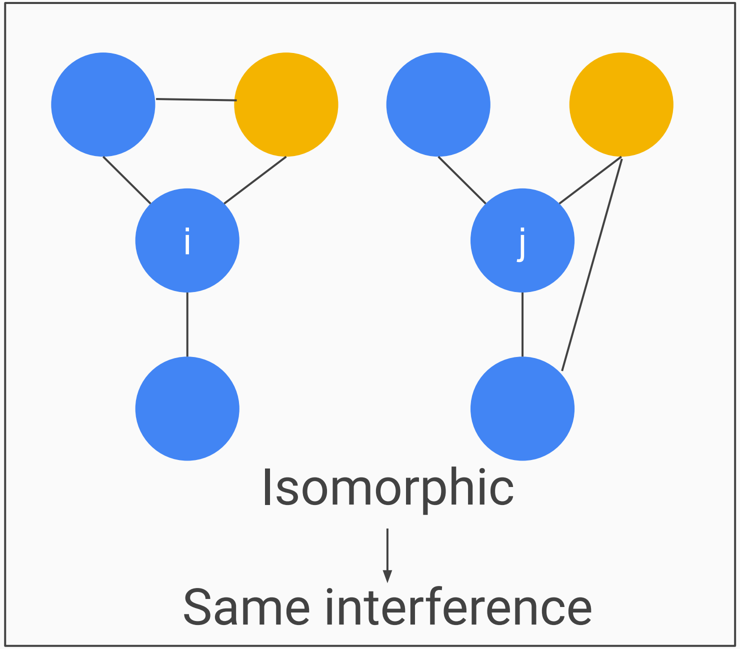
The AME-Networks Assumptions: 4

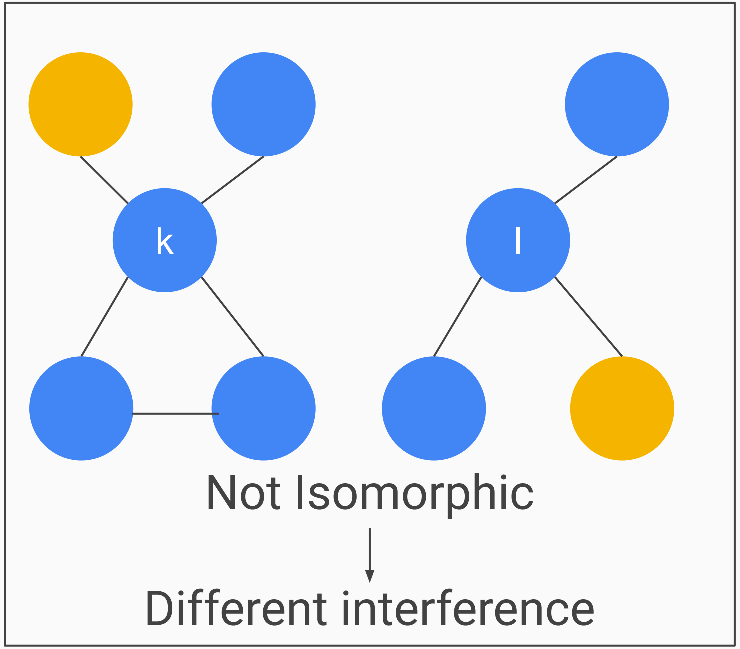
A3) Isomorphic graph interference: If two units, *i* and *j*, have isomorphic neighborhood graphs, then they receive the same amount of interference.



The AME-Networks Assumptions: 4

A3) Isomorphic graph interference: If two units, *i* and *j*, have isomorphic neighborhood graphs, then they receive the same amount of interference.





Identification of the Average Direct Effect

Proposition 1. *Under assumptions A0-A3, potential outcomes in (2) for all units i can be written as:*

$$Y_i(t, \mathbf{t}_{-i}) = t\tau_i + f(G^{\mathbf{t}}_{\mathcal{N}_i}) + \epsilon_i, \tag{3}$$

where τ_i is the direct treatment effect on unit i, and ϵ_i is some baseline response.

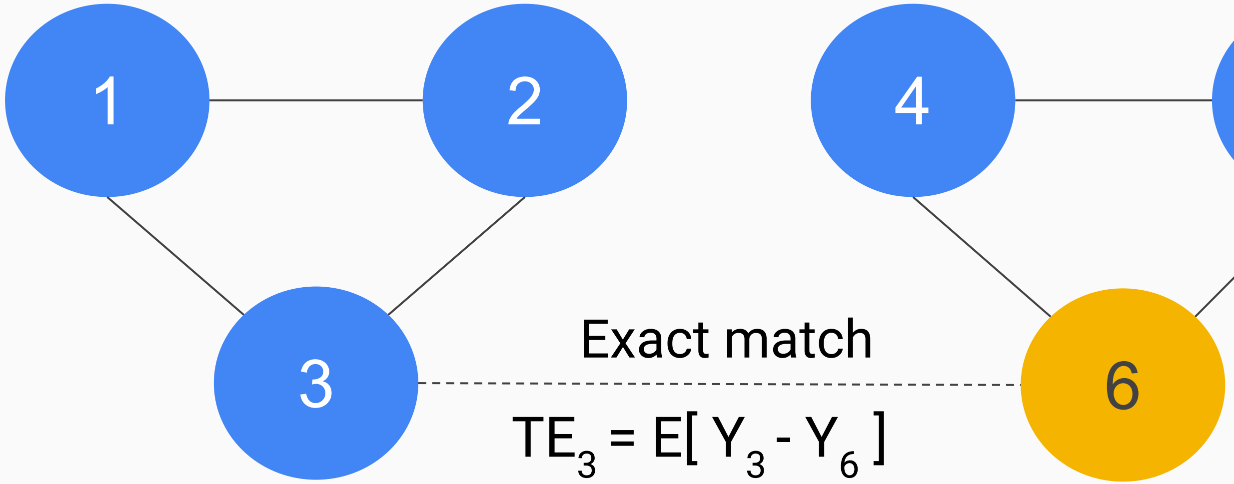
In addition, suppose that baseline responses for all units are equal to each other in expectation, i.e., for all i, $\mathbb{E}[\epsilon_i] = \alpha$. Then under assumptions A0-A3, for neighborhood graph structures g_i of unit i and treatment vectors $\mathbf{t}$, the ADE is identified as:

$$ADE = \frac{1}{n^{(1)}} \sum_{i=1}^{n} \mathbb{E}\big[T_i \times \big(\mathbb{E}[Y_i | G^{\mathbf{T}}_{\mathcal{N}_i} \simeq g^{\mathbf{t}}_i, T_i = 1] - \mathbb{E}[Y_i | G^{\mathbf{T}}_{\mathcal{N}} \simeq g^{\mathbf{t}}_i, T_i = 0]\big)\big],$$

where $G^{\mathbf{T}}_{\mathcal{N}_i}$ is the neighborhood graph of i labelled according to the treatment assignment $\mathbf{T}$.

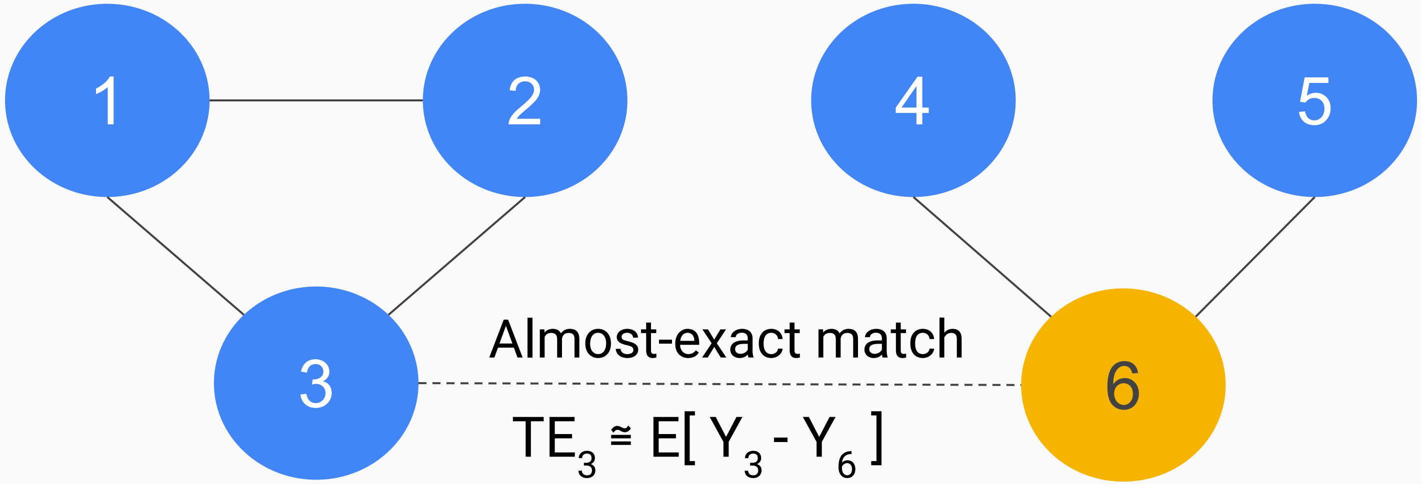
Main Takeaway

We can estimate the treatment effect by **matching** treated units to control units with isomorphic neighborhood graphs



Main Takeaway

We are unlikely to find **exact** matches for each unit, so construct an **almost-exact** match



FLAME-Networks: The Algorithm

2 Steps

1. Construct a measure of similarity between units' neighborhood graphs

2. Use that measure to match together units with similar neighborhood graphs

2 Steps

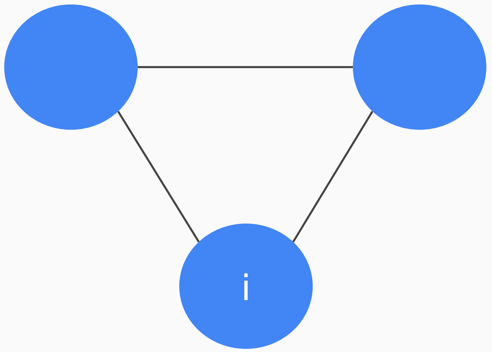
1. Construct a measure of similarity between units' neighborhood graphs

2. Use that measure to match together units with similar neighborhood graphs

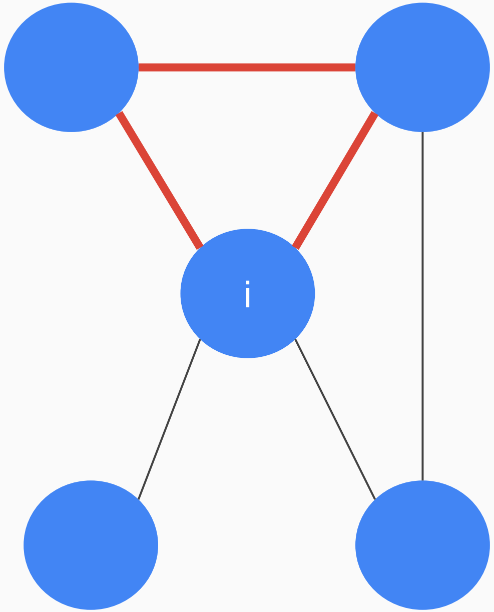
Graph similarity by counting subgraphs

We enumerate and count unique subgraph shapes in neighborhood graphs

Triangle subgraph



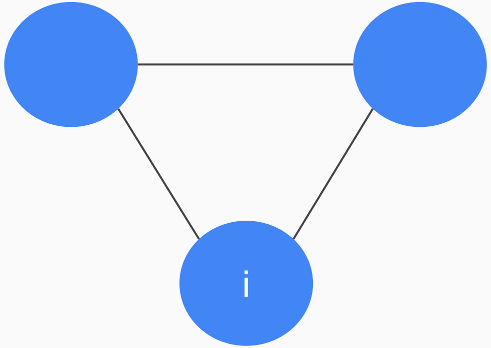
Triangles = 1



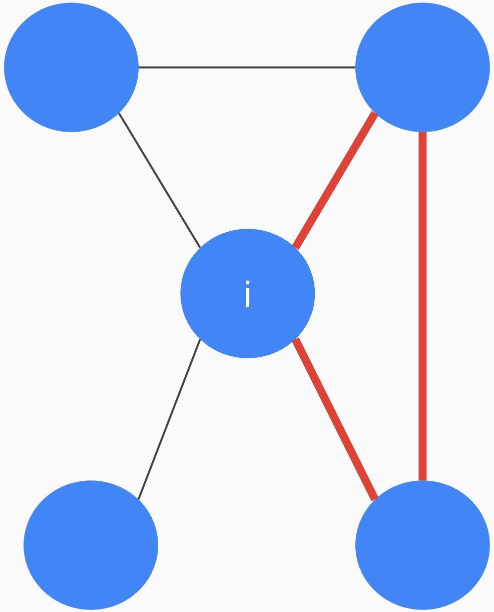
Graph similarity by counting subgraphs

We enumerate and count unique subgraph shapes in neighborhood graphs

Triangle subgraph



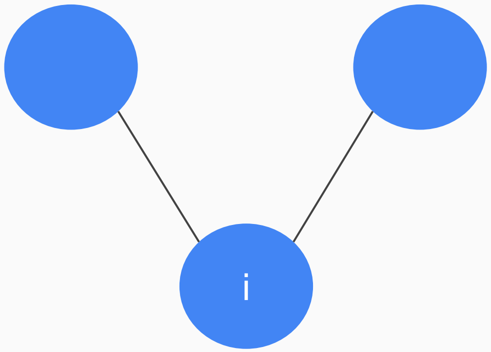
Triangles = 2



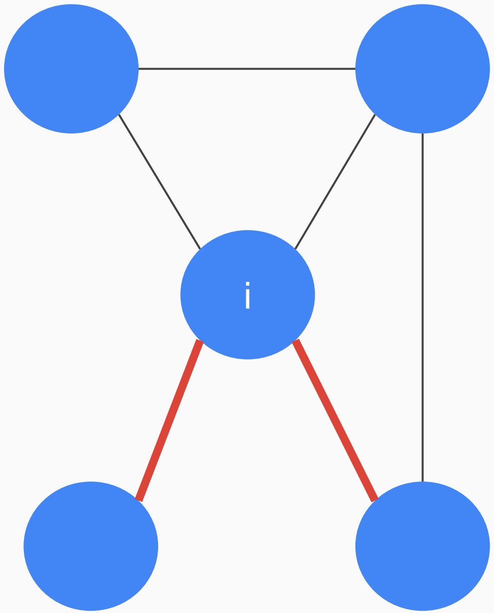
Graph similarity by counting subgraphs

We enumerate and count unique subgraph shapes in neighborhood graphs

V subgraph



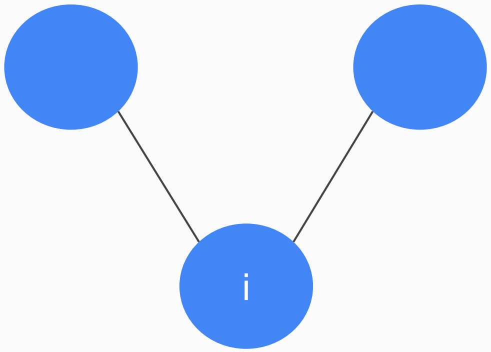
Vs = 1



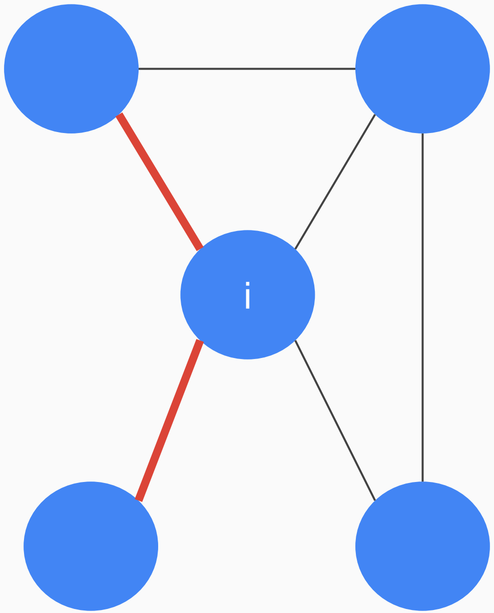
Graph similarity by counting subgraphs

We enumerate and count unique subgraph shapes in neighborhood graphs

V subgraph



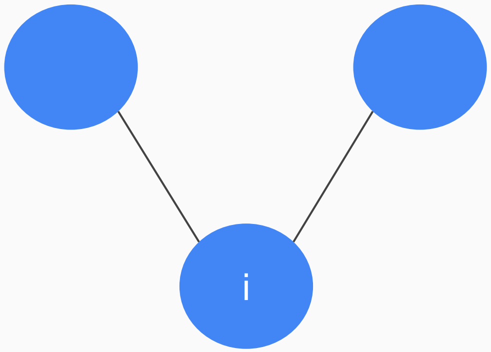
Vs = 2



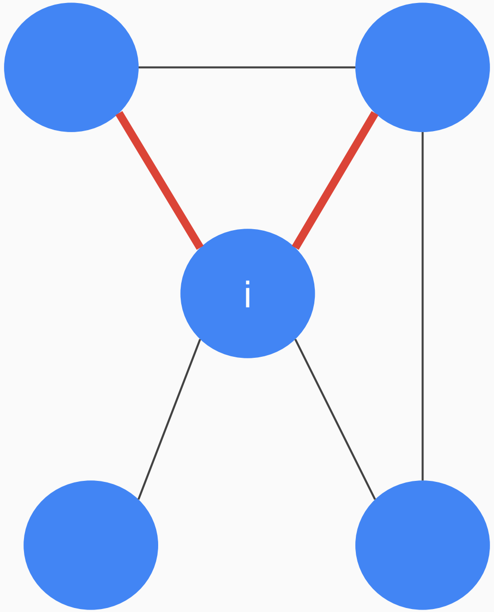
Graph similarity by counting subgraphs

We enumerate and count unique subgraph shapes in neighborhood graphs

V subgraph



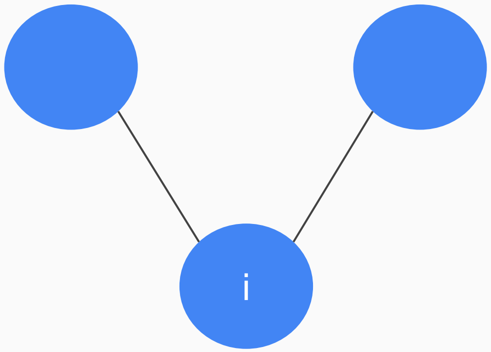
Vs = 3



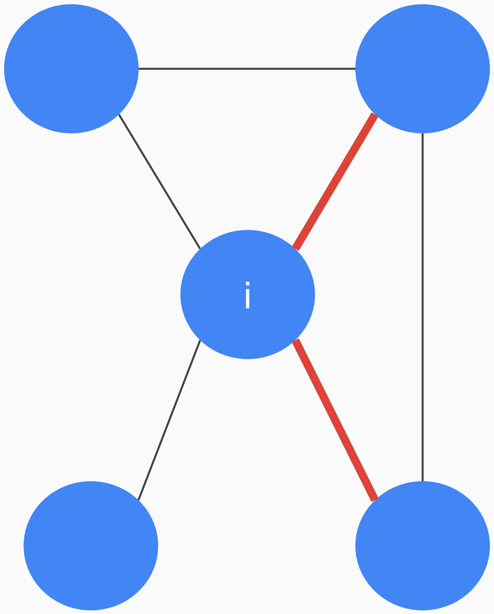
Graph similarity by counting subgraphs

We enumerate and count unique subgraph shapes in neighborhood graphs

V subgraph



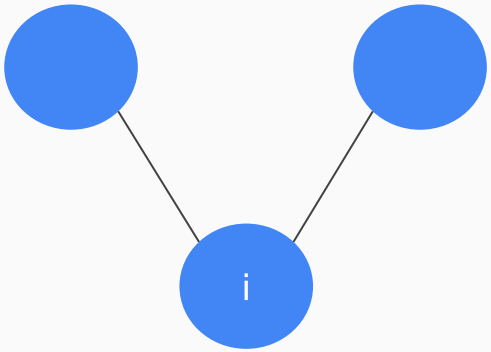
Vs = 4



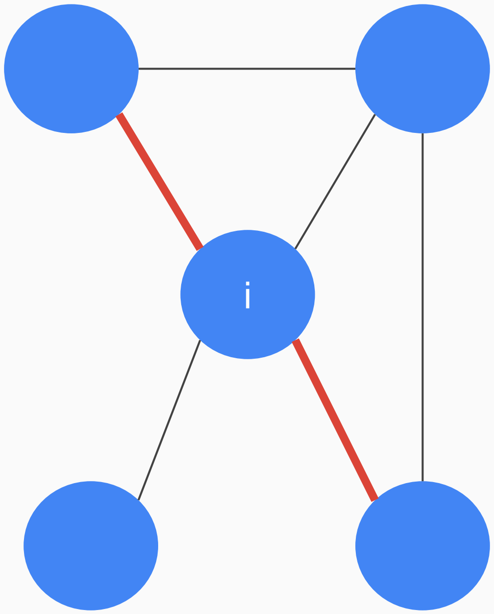
Graph similarity by counting subgraphs

We enumerate and count unique subgraph shapes in neighborhood graphs

V subgraph



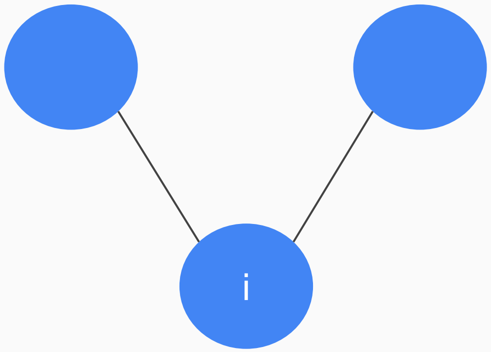
Vs = 5



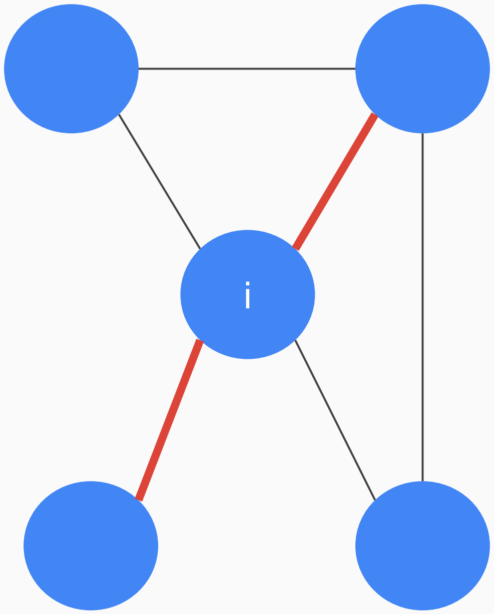
Graph similarity by counting subgraphs

We enumerate and count unique subgraph shapes in neighborhood graphs

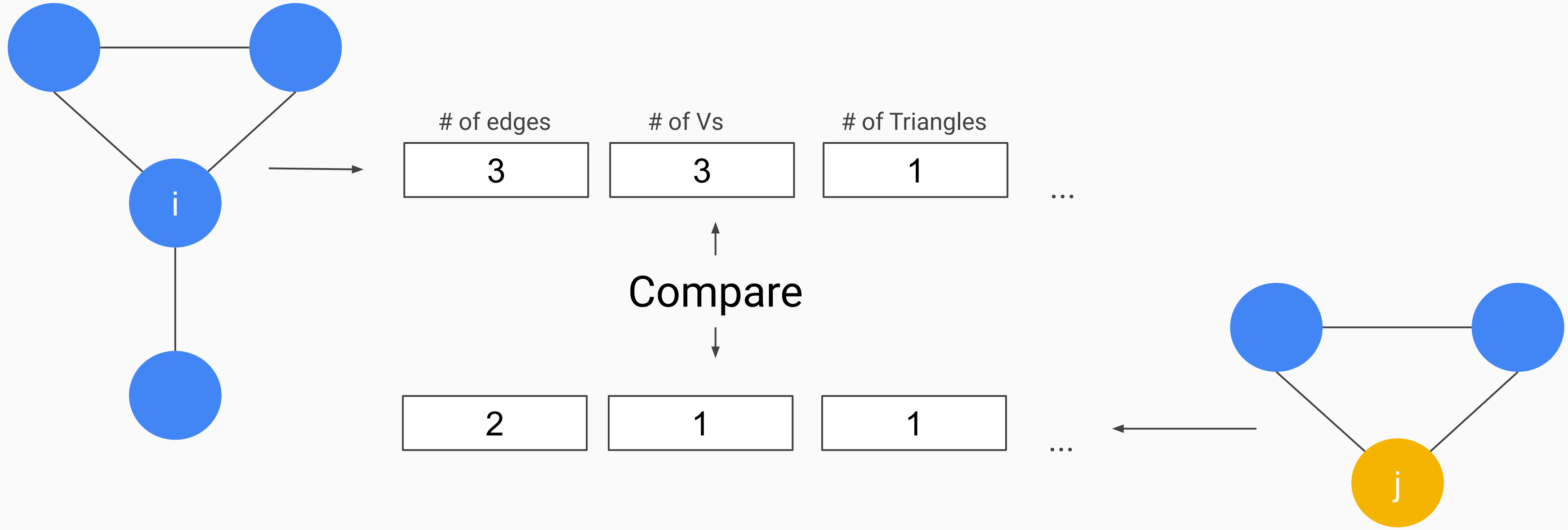
V subgraph



Vs = 6



Graph similarity by counting subgraphs

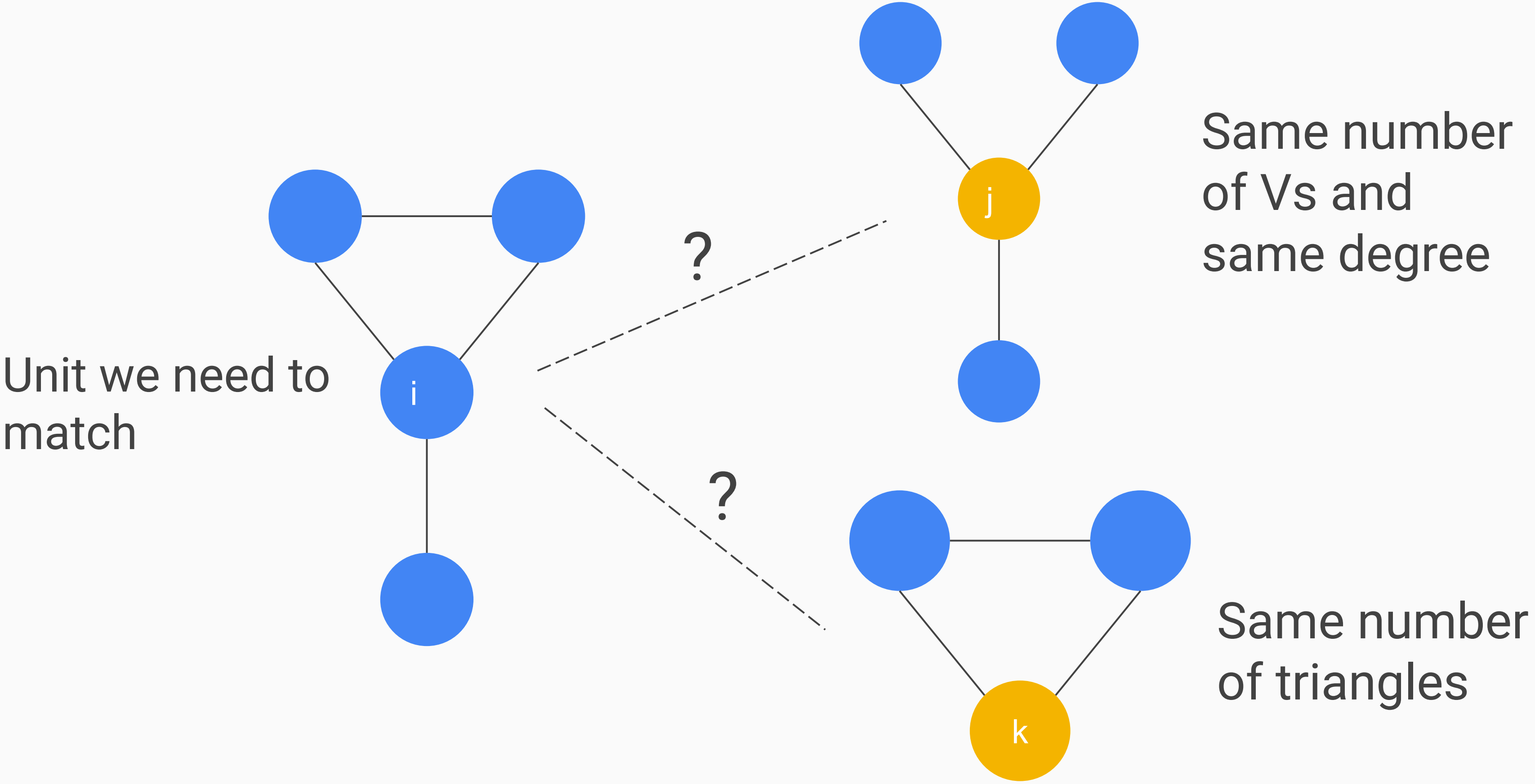


2 Steps

1. Construct a measure of similarity between units' neighborhood graphs

2. Use that measure to match together units with similar neighborhood graphs

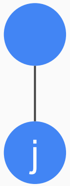
Who should we match *i* to?



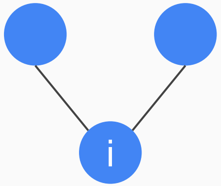
Who should we match *i* to?

We need a measure of **importance** of subgraphs for matching

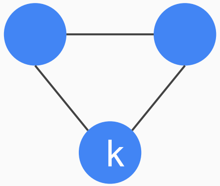
Edge:
Importance = 1



V:
Importance = 2

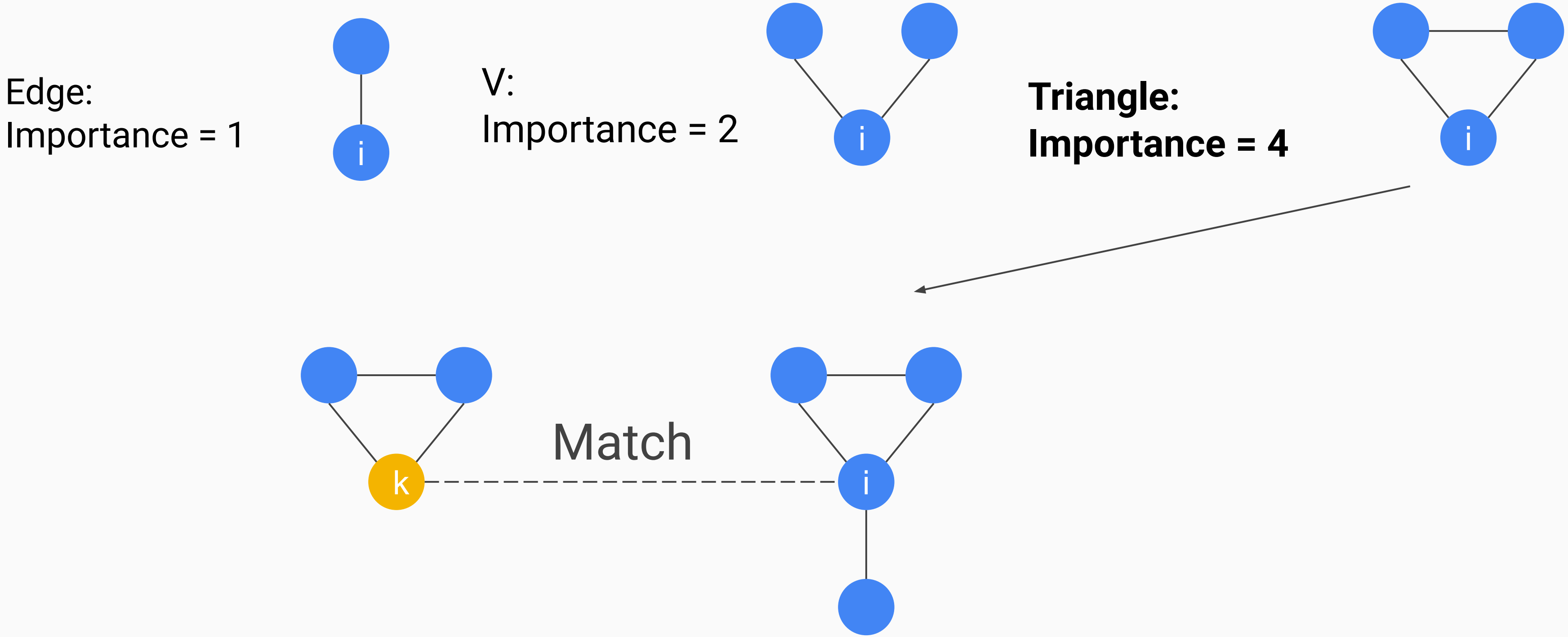


Triangle:
Importance = 4



Who should we match *i* to?

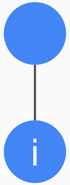
We need a measure of **importance** of subgraphs for matching



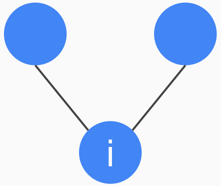
Who should we match *i* to?

We need a measure of **importance** of subgraphs for matching

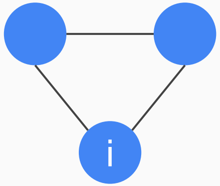
Edge:
Importance = 1

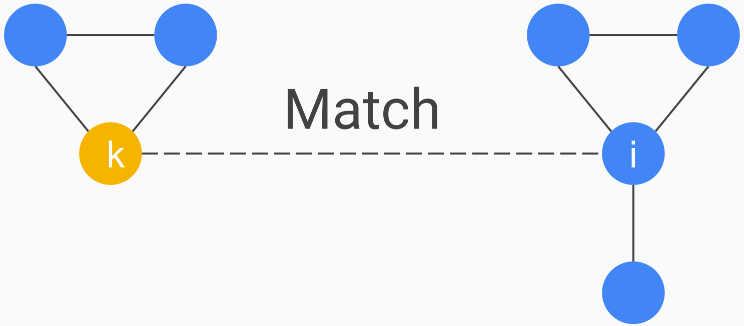


V:
Importance = 2

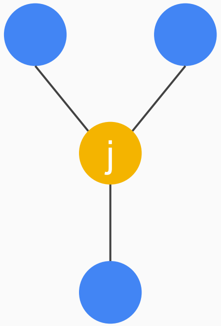


Triangle:
Importance = 4



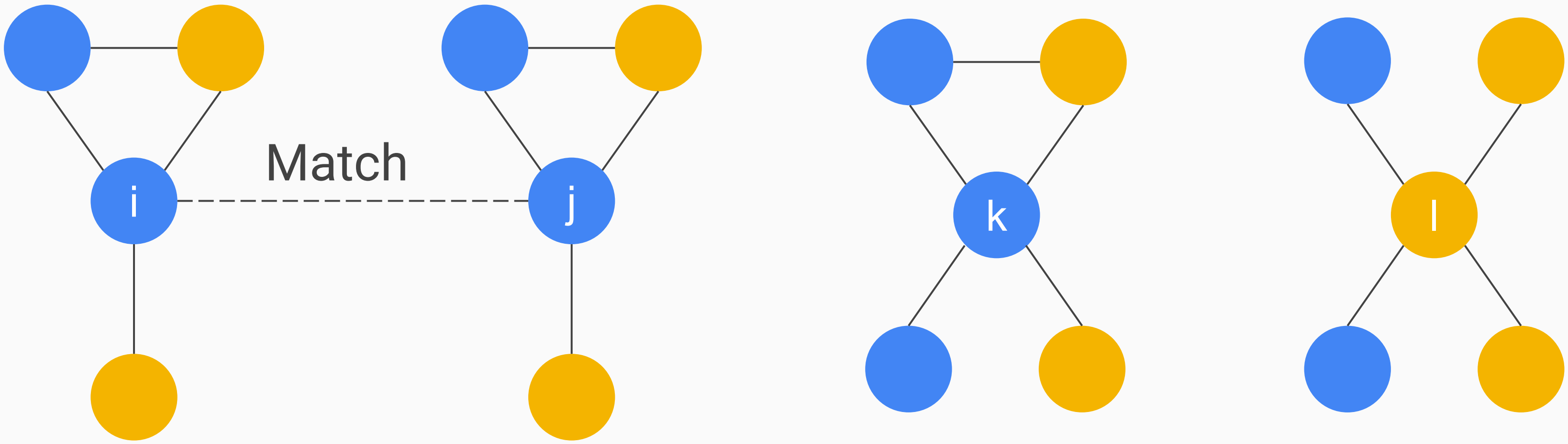


Don't
Match



The Fast, Large scale, Almost Matching Exactly (FLAME) Algorithm (Wang et al.)

1. Match as many units on all subgraphs as we can.



The Fast, Large scale, Almost Matching Exactly (FLAME) Algorithm (Wang et al.)

2. Find which subgraph is **least** predictive of the outcome on a training set

On a training set learn:

Y = f(*n triangles*)

Y = f(*n edges*)

The Fast, Large scale, Almost Matching Exactly (FLAME) Algorithm (Wang et al.)

2. Find which subgraph is **least** predictive of the outcome on a training set

On a training set learn:

Y = f(*n triangles*) Y = f(*n edges*)

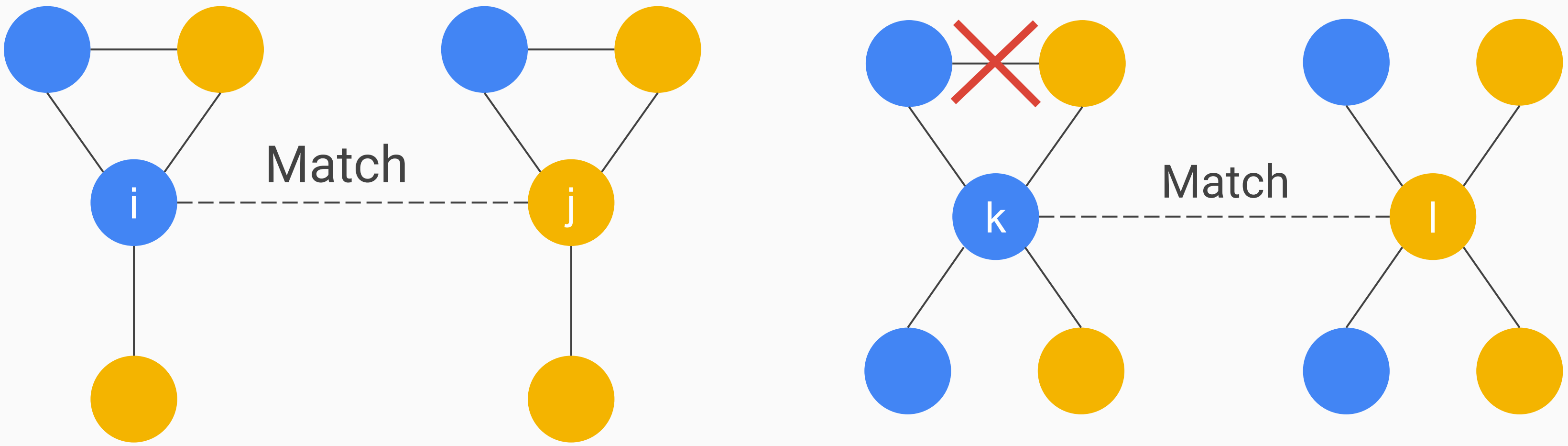
Compare prediction error:

(f(*n triangles*) - *Y*)^2 > (f(*n edges*) - *Y*)^2

Triangles is least predictive

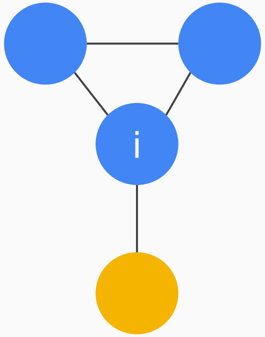
The Fast, Large scale, Almost Matching Exactly (FLAME) Algorithm (Wang et al.)

3. Exclude that subgraph, repeat from 1



The FLAME-Networks Procedure

1. For each unit, *i*, enumerate and count all the unique subgraph in the neighborhood graph of *i*



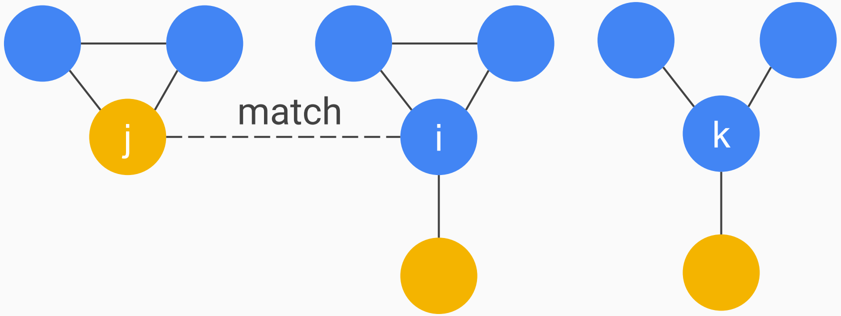
Subgraph counts for G_{Ni}
3 singletons
5 Vs
1 Triangle
1 Triangle + singleton
2 Triangle + V
1 Triangle + 2Vs

2. Construct a 0/1 feature vector for *i* from the subgraph counts

# of singletons = 1	# of singletons = 2	# of singletons = 3	
0	0	1	...

	# of triangles = 1	# of triangles = 2	# of triangles = 3	
...	1	0	0	...

3. Run FLAME with the binarized subgraph counts as inputs



FLAME-Networks: 2 Steps

1. Construct a measure of similarity between units’ neighborhood graphs

2. Use that measure to match together units with similar neighborhood graphs

Experiments

Competing Methods

1. **Naive:** the simple difference between mean treated outcome and mean control outcome
2. **All Eigenvectors:** We take the eigendecomposition of the adjacency matrix of the unit graph, and match together units with the closest eigenvectors in in terms of L2 distance
3. **First eigenvector:** Same as above but we match on the first eigenvector only
4. **Stratified Naive (Sussman and Airoldi 2017):** A difference in means estimator where units are stratified by their treated degree.
5. **SANIA MIV LUE (Sussman and Airoldi 2017):** Minimum integrated variance linear unbiased estimator under the assumptions that interference is symmetrically received.

Experimental Setup

When generating outcomes in our experiments we consider the following subgraphs:

d_i:	The treated degree of unit i
Δ_i:	The number of triangles in $G^{\mathbf{t}}_{\mathcal{N}_i \cup \{i\}}$ with at least one treated unit.
$\bigstar_i^k$:	The number of k-stars in $G^{\mathbf{t}}_{\mathcal{N}_i \cup \{i\}}$ with at least one treated unit.
$\dagger_i^k$:	The number of units in $G^{\mathbf{t}}_{\mathcal{N}_i \cup \{i\}}$ with degree $\geq k$ and at least one treated unit among their neighbors.
B_i:	The vertex betweenness of unit i.
C_i:	The closeness centrality of unit i.

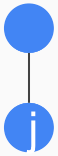
Subgraphs are assigned weight as follows:

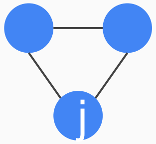
Feature Weight	d_i γ_1	Δ_i γ_2	$\bigstar_i^2$ γ_3	$\bigstar_i^4$ γ_4	$\dagger^3$ γ_5	B_i γ_6	C_i γ_7
Setting 1	0	10	0	0	0	0	0
Setting 2	10	10	0	0	0	0	0
Setting 3	0	10	1	1	1	1	-1
Setting 4	5	1	10	1	1	1	-1

Outcomes are generated from: $Y = \gamma_1 d_i + \gamma_2 \Delta + \gamma_3 \bigstar^2 + \gamma_4 \bigstar^4 + \gamma_5 \dagger^3 + \gamma_6 B + \gamma_7 C + \epsilon_i$

Results: Additive Interference

Outcome generation:

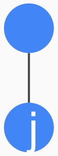
Y = 5 * T + 10 * # 

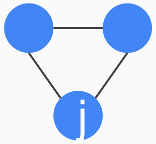
+ 10 * # 

+ eps

Results: Additive Interference

Outcome generation:

Y = 5 * T + 10 * # 

+ 10 * # 

+ eps

FLAME Networks

First Eigenvector

All Eigenvectors

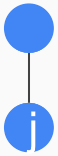
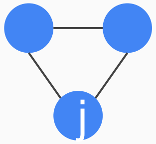
Naive

Stratified

SANIA

Results: Additive Interference

Outcome generation:

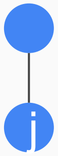
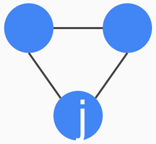
Y = 5 * T + 10 * #  + 10 * #  + eps

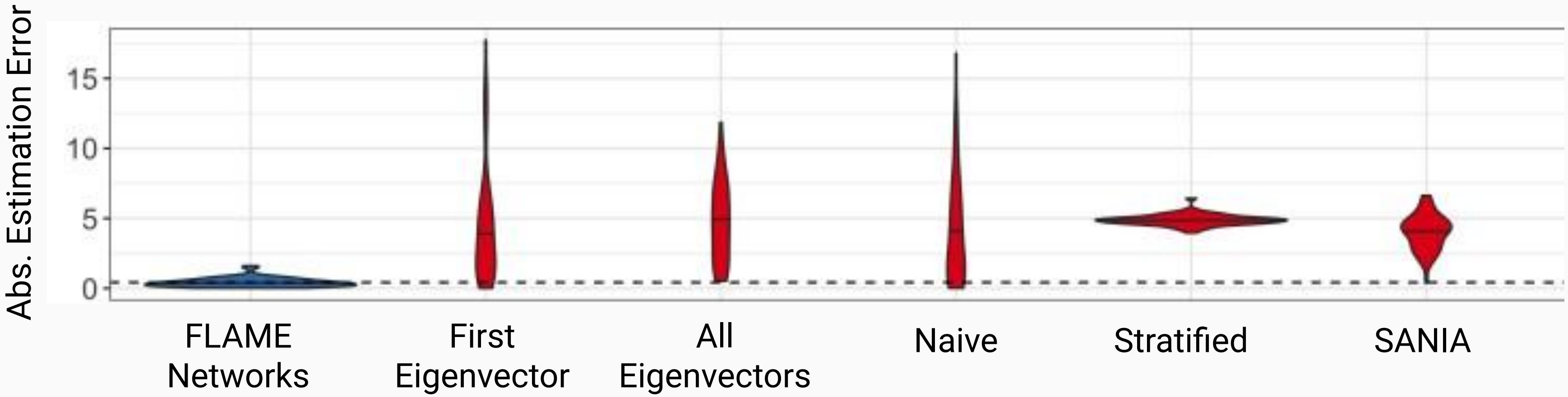
Abs. Estimation Error

FLAME Networks | First Eigenvector | All Eigenvectors | Naive | Stratified | SANIA

Results: Additive Interference

Outcome generation:

Y = 5 * T + 10 * #  + 10 * #  + eps

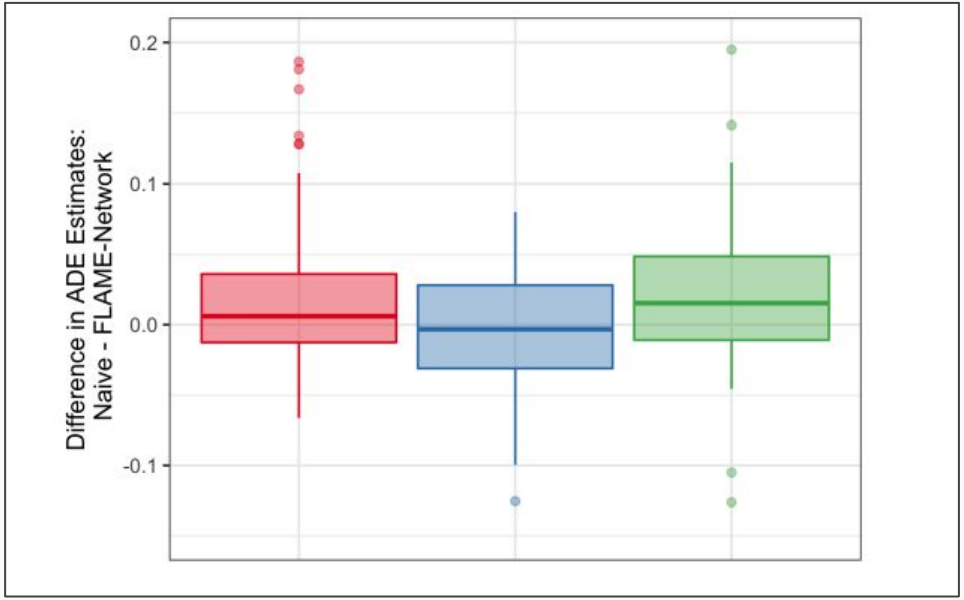


Application: Estimating the Effects of Gender and Education on Self-Help Group Participation in Developing Countries.

Application background

- Data collected by Banerjee et al. (2013) on social networks for 75 villages in Karnataka, India.
- They are a median distance of 46 km from one another other: network interference is experienced solely between individuals from the same village.
- For each village, we study the effect of
  - 1. lack of education on election participation;
  - 2. lack of education on Self-Help-Group (SHG) participation; and
  - 3. being male on SHG participation.
- We compare our estimates – which account for network interference – to naive estimates – which assume no network interference

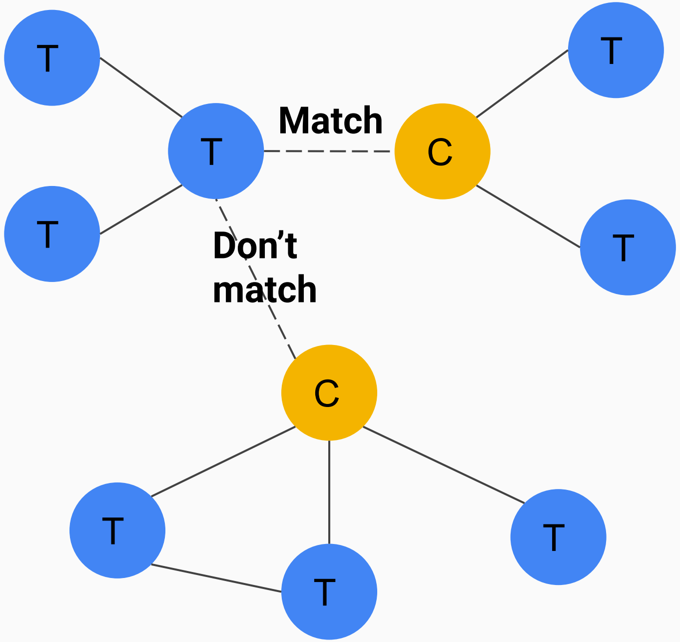
Results: Effects of being male on Self-Help Group participation



Summary

Summary

- Network interference shouldn’t be ignored
- **FLAME-Networks** is an interpretable matching method that takes network interference into account
    - Matches units on their neighborhood subgraphs
    - Excellent performance in practice



Thank you!