Optimal Blocking by Minimizing the Maximum Intra-block Dissimilarity

Michael J. Higgins
Jasjeet Sekhon

University of California at Berkeley

May 7, 2013

Goal

Our goal: Devise a method for blocking for experiments with multiple treatment categories (and possibly multiple replications of each treatment within each block) that ensures good covariate balance between treatment groups in small experiments and is efficient enough to be used in large experiments.

Covariate imbalance in randomized experiments

- Randomized controlled experiment with n units and r treatment categories.
- PROBLEM: Although randomized experiments are a good idea, there may be a non-negligible probability of bad covariate balance between treatment groups—a treatment group has too many Republicans, tall people, terminally ill people, etc.
- Bad imbalance on important covariates $\longrightarrow$ Inaccurate estimates of treatment effects.
- Problem is most likely to happen in small experiments; critical to avoid this problem if cost of additional units is large (e.g. a medical trial).

Post-randomization solutions to covariate imbalance

- Post-stratification [Miratrix et al., 2012]:
  - Group similar units together when performing analysis *after* running the experiment to increase precision of treatment effect estimates.
  - Data mining concerns.
  - May not be most efficient method for small experiments.
- Re-randomization [Lock Morgan and Rubin, 2012]:
  - Analyze covariate balance after randomization. Repeat randomly assigning treatments until covariate balance is "acceptable."
  - Standard errors calculated across only "valid" randomizations. These calculations can be complex—even when covariate balance is good!
- Moral of the story: Preferable to design the randomization scheme to avoid covariate imbalance.

Blocking

- Blocking can be used to prevent covariate imbalance from randomization.
- *Blocking*—grouping units according to common traits (e.g. same political party, similar soil fertility) before treatment is assigned.
- *Completely randomize* treatment independently across blocks—every possible "balanced" treatment assignment within a block is equally likely.
- Traits used to form blocks are called *block covariates*.

Blocking: Cont'd

- Blocking may still be a good idea when covariate balance not the main concern (for example, in large experiments).
- When block covariates are good predictors of response, blocking can reduce variance of average treatment effect estimates.
- GOAL: Derive a good blocking technique for experiments with two *or more* treatment categories that can ensure some level of covariate balance in small experiments, and can be applied to large experiments.

Matched-pairs: Current solution

- Matched-pairs blocking: Pair "most-similar" units together. For each pair, randomly assign one unit to treatment, one to control.
- Shortcomings:
- No efficient way to extend approach to cases where more than two treatment categories.
- Fixed block sizes (2 units): If units form clusters with more than two units, matched-pairs design may pair units from different clusters.
- Even with two treatments, having blocks of more than two units may be helpful in estimating standard errors of estimates under some models [Imbens, 2011].

Blocking by minimizing the MIBD

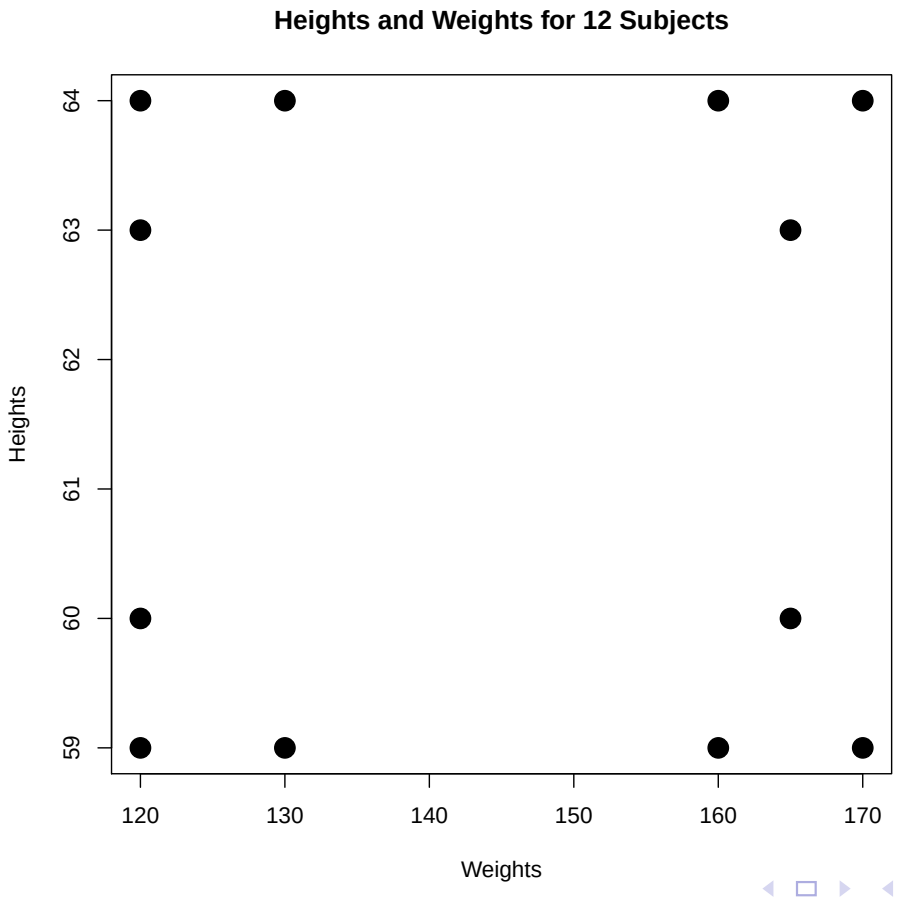
- Select a threshold $t^* \geq r$ for a minimum number of units to be contained in a block.
- Block units so that each block contains at least t^* units, and so that the maximum "dissimilarity" (e.g. Mahalanobis distance between block covariates) between any two units within a block—the maximum intra-block dissimilarity (MIBD)—is minimized.

Blocking by minimizing the MIBD

- Select a threshold $t^* \geq r$ for a minimum number of units to be contained in a block.
- Block units so that each block contains at least t^* units, and so that the maximum "dissimilarity" (e.g. Mahalanobis distance between block covariates) between any two units within a block—the maximum intra-block dissimilarity (MIBD)—is minimized.
- Minimizing the MIBD: Can ensure covariate balance in randomization.
- Threshold t^*: Allows designs with multiple treatment categories, multiple replications of treatments within a block.
- We find a "good" blocking in polynomial time: applicable to large experiments.

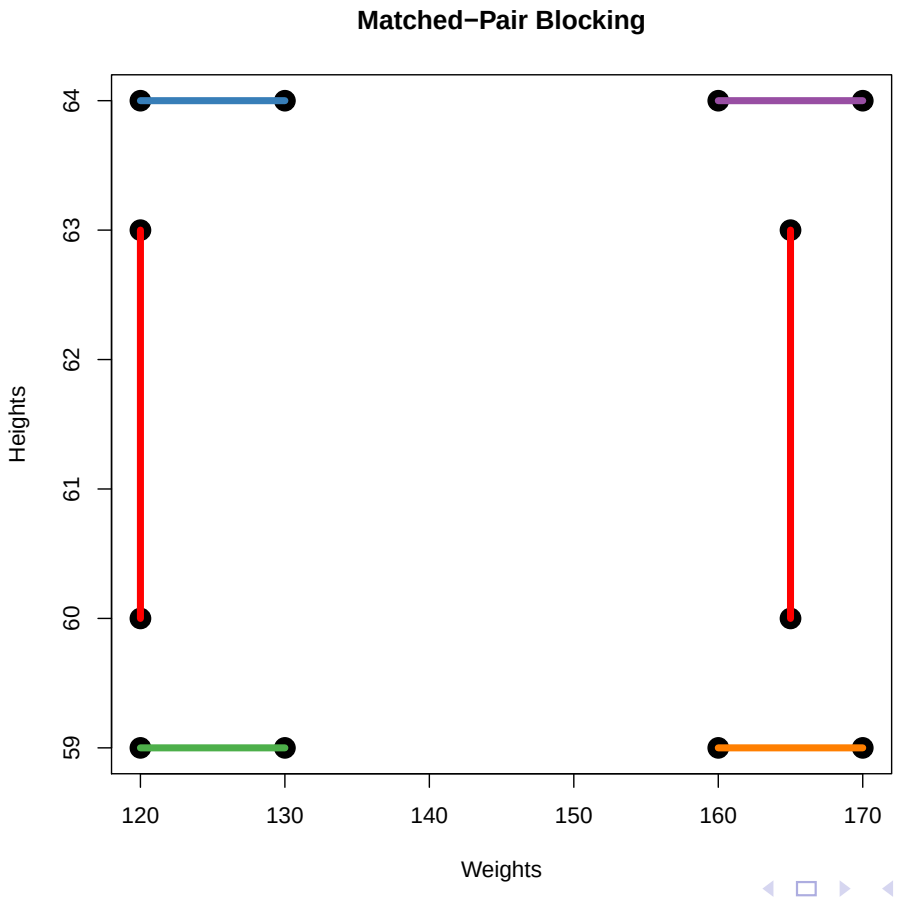
A simple example:

Threshold $t^* = 2$. Dissimilarity = Mahalanobis distance.



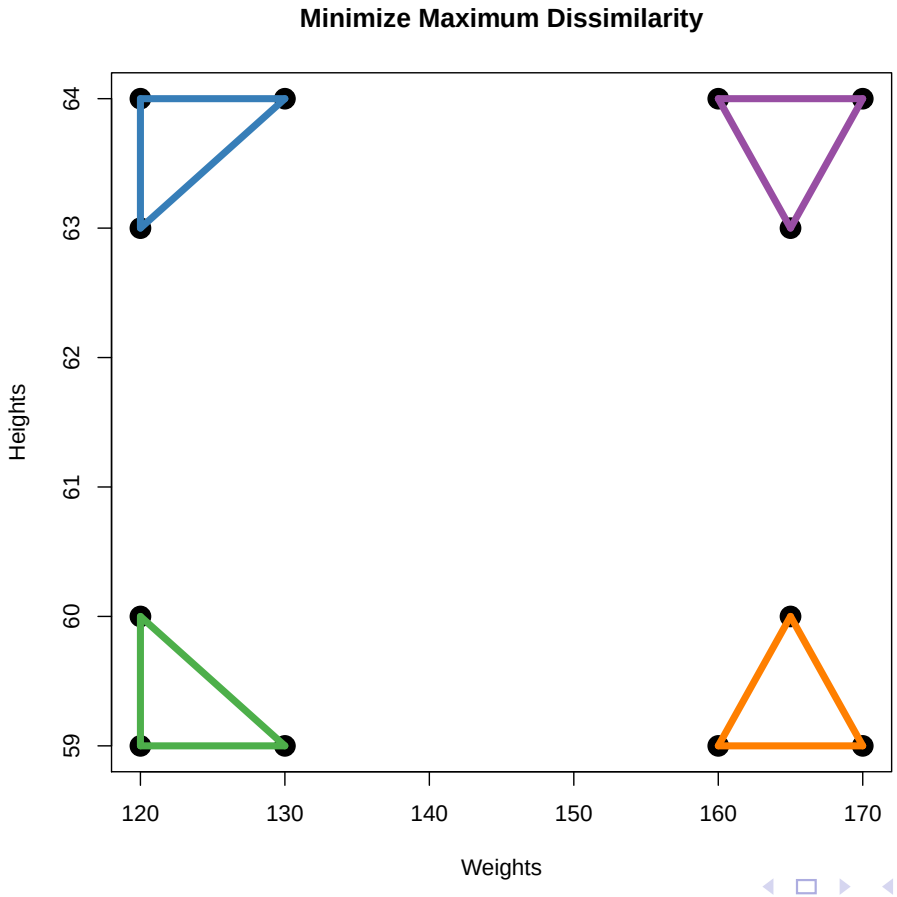
A simple example:

Threshold $t^* = 2$. Dissimilarity = Mahalanobis distance.



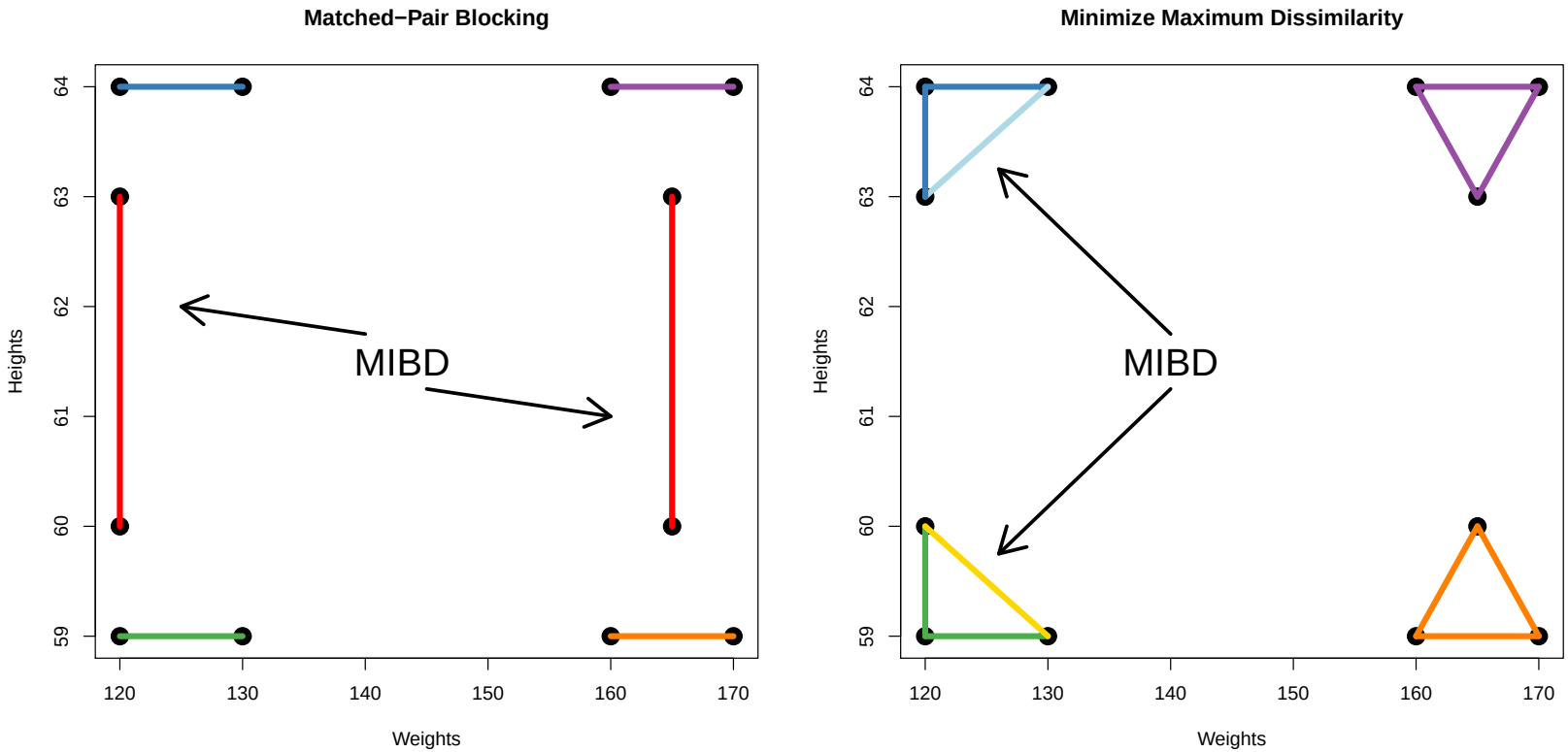
A simple example:

Threshold $t^* = 2$. Dissimilarity = Mahalanobis distance.



A simple example:

Threshold $t^* = 2$. Dissimilarity = Mahalanobis distance.



Optimal blocking and approximately optimal blocking

- *For all blockings that contain at least t^* units with each block*:
- Let λ denote the smallest MIBD achievable by such a blocking—any blocking that meets this bound is called an *optimal blocking*.
- Finding an optimal blocking is an NP-hard problem—feasible to find in small experiments, may not be feasible in large experiments [Hochbaum and Shmoys, 1986].

Optimal blocking and approximately optimal blocking

- *For all blockings that contain at least t^* units with each block*:
- Let λ denote the smallest MIBD achievable by such a blocking—any blocking that meets this bound is called an *optimal blocking*.
- Finding an optimal blocking is an NP-hard problem—feasible to find in small experiments, may not be feasible in large experiments [Hochbaum and Shmoys, 1986].
- We now show that finding a blocking with MIBD $\leq 4\lambda$ is possible in polynomial time—finds a "good" blocking when the number of units is small *or large*.
- Denote any such blocking as an *approximately optimal blocking*.

Viewing experimental units as a graph

- Use an idea from Paul Rosenbaum [1989]: Matching problems can be viewed as graph theory partitioning problems.
- Experimental units are vertices in a graph.
- An edge is drawn between two units if they can be matched together.

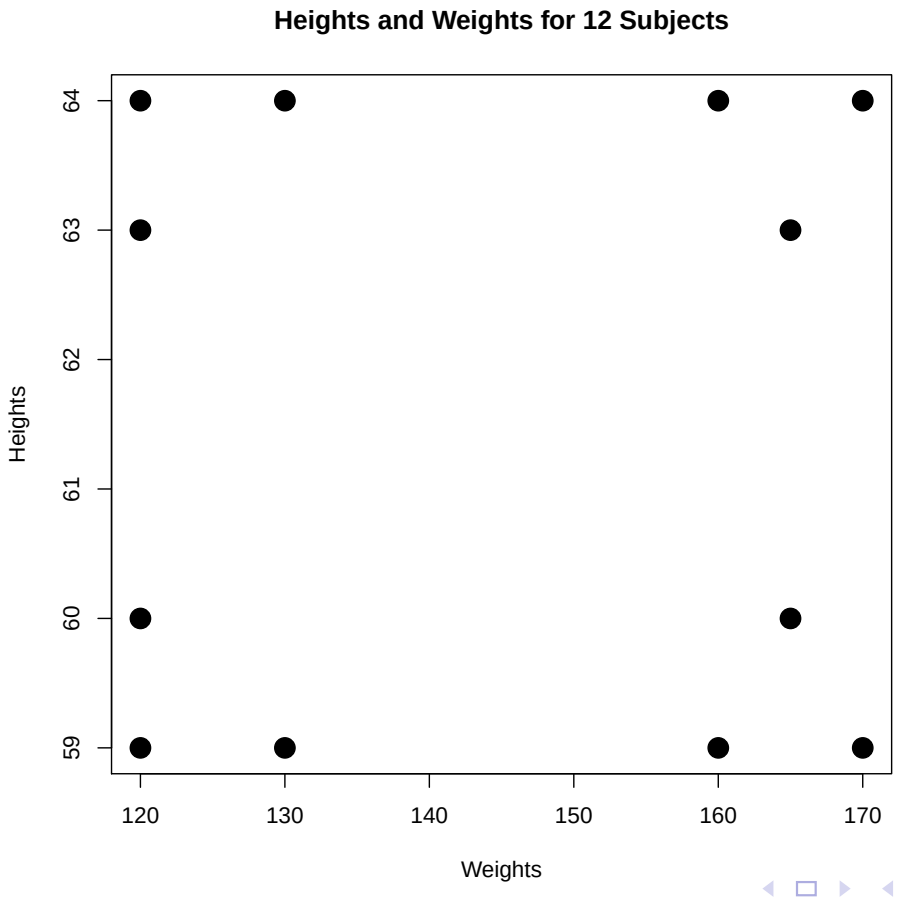
- Edge-weights are some measure of dissimilarity between pretreatment covariates (e.g. Mahalanobis distance).
- Only for matched-pair designs.

Viewing experimental units as a graph: Our application

- Use an idea from Paul Rosenbaum [1989]: Matching problems can be viewed as graph theory partitioning problems.
- Experimental units are vertices in a graph.
- An edge is drawn between two units if
  ~~they can be matched together.~~
  they can be placed in the same block.
- Edge-weights are some measure of dissimilarity between pretreatment covariates (e.g. Mahalanobis distance).
- ~~Only for matched-pair designs.~~
  Arbitrarily large block-sizes.

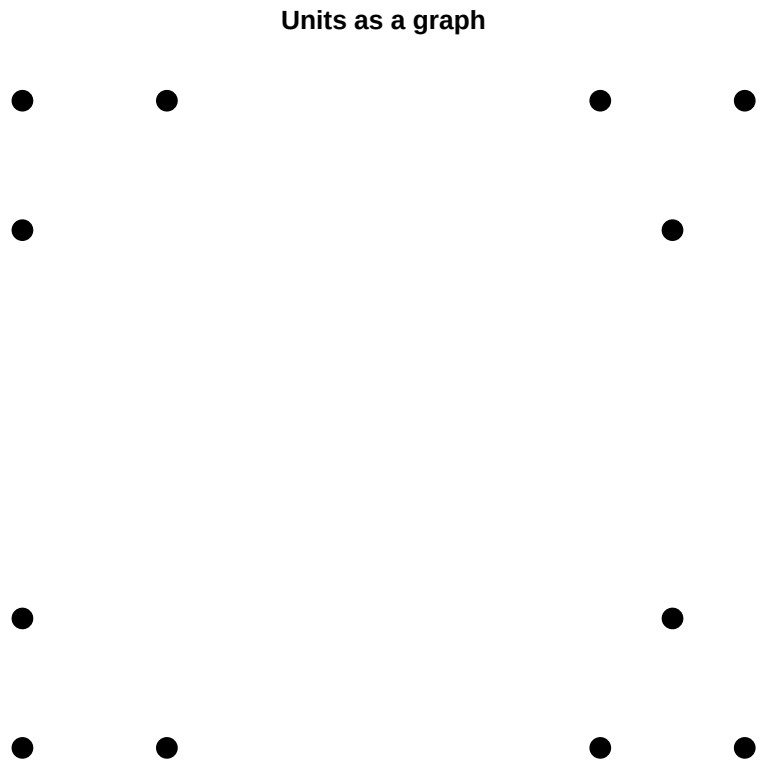
Viewing experimental units as a graph: In pictures

Dissimilarity = Mahalanobis distance.



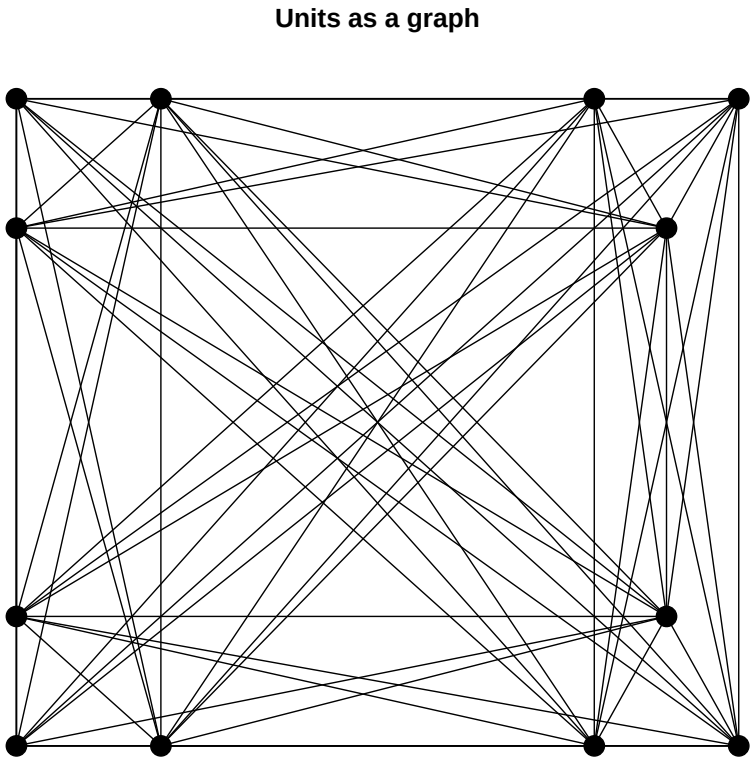
Viewing experimental units as a graph: In pictures

Dissimilarity = Mahalanobis distance.



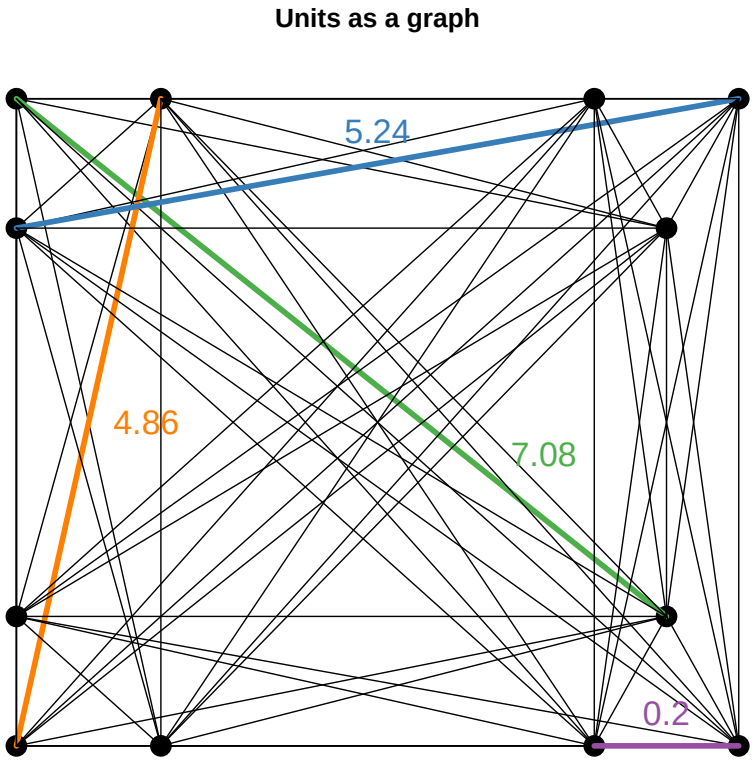
Viewing experimental units as a graph: In pictures

Dissimilarity = Mahalanobis distance.



Viewing experimental units as a graph: In pictures

Dissimilarity = Mahalanobis distance.



Notation:

- A graph G is defined by its vertex set V and its edge set E: $G = (V, E)$.
- Vertices in V denoted by $\{i\}$; n units $\longrightarrow$ n vertices in V.
- Edges in E are denoted by (i, j).
- A *complete* graph has an edge $(i, j) \in E$ between every two distinct vertices $\{i\}, \{j\} \in V$; $\frac{n(n-1)}{2}$ edges overall.
- The weight of edge $(i, j) \in E$ is denoted by w_{ij}: at most $\frac{n(n-1)}{2}$ distinct values of w_{ij}.

Note about edge weights

- If our concern is covariate balance, natural choices for edge weights measure dissimilarity between block covariates (e.g. Mahalanobis distance, weighted Mahalanobis distance).

Note about edge weights

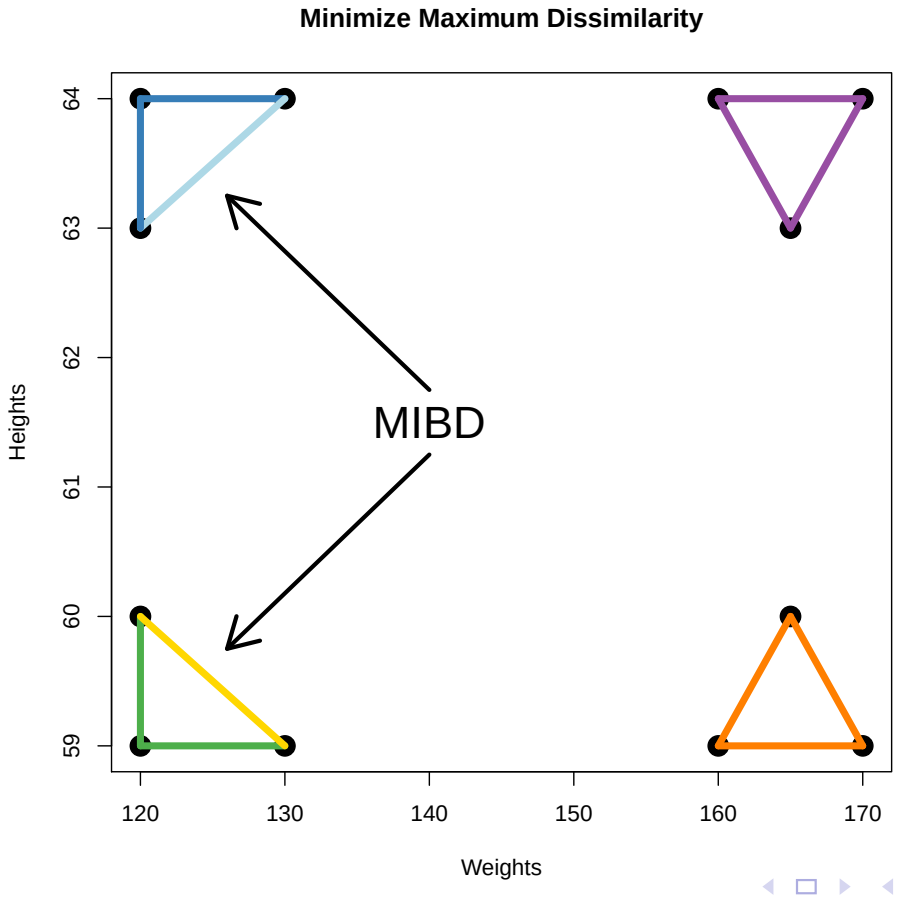
- If our concern is covariate balance, natural choices for edge weights measure dissimilarity between block covariates (e.g. Mahalanobis distance, weighted Mahalanobis distance).
- Our method only requires weights to satisfy the triangle inequality: for any distinct vertices $\{i\}, \{j\}, \{k\}$,

$$w_{ik} + w_{kj} \geq w_{ij}.$$

- This holds for any distance metric.

A simple example:

Threshold $t^* = 2$. Dissimilarity = Mahalanobis distance.



Optimal blocking as a graph partitioning problem

- A *partition* of V is a division of V into disjoint *blocks* of vertices $(V_1, V_2, \ldots, V_\ell)$.
- Blocking of units $\leftrightarrow$ Partition of a graph:
  Two units are in the same block of the blocking if and only if their corresponding units are in the same block of the partition.

Optimal blocking as a graph partitioning problem

- A *partition* of V is a division of V into disjoint *blocks* of vertices $(V_1, V_2, \ldots, V_\ell)$.
- Blocking of units $\leftrightarrow$ Partition of a graph:
  Two units are in the same block of the blocking if and only if their corresponding units are in the same block of the partition.
- Optimal blocking problems are optimal partitioning problems: we want to find a partition $(V_1^*, V_2^*, \ldots, V_{\ell^*}^*)$ with $|V_j^*| \geq t^*$ that minimizes the maximum intra-block edge weight.

Bottleneck subgraphs

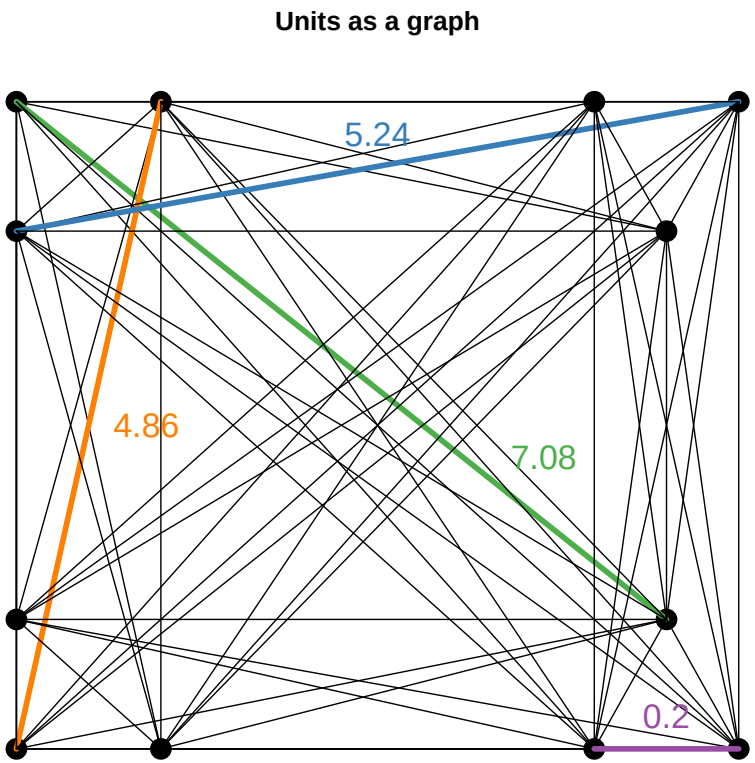
- Bottleneck subgraphs are helpful in solving graph partitioning problems.
- Define the *bottleneck subgraph of G of weight w* $B_w(G) = (V, E_w)$ as the graph that has $(i, j) \in E_w$ if and only if $w_{ij} \leq w$.

Bottleneck subgraphs

- Bottleneck subgraphs are helpful in solving graph partitioning problems.
- Define the *bottleneck subgraph of G of weight w* $B_w(G) = (V, E_w)$ as the graph that has $(i, j) \in E_w$ if and only if $w_{ij} \leq w$.
- At most $\frac{n(n-1)}{2}$ different edge weights $w_{ij} \rightarrow$ At most $\frac{n(n-1)}{2}$ different bottleneck subgraphs.
- All points within a block of our approximately optimal blocking are connected by some path of edges in a bottleneck subgraph; used to show approximate optimality.

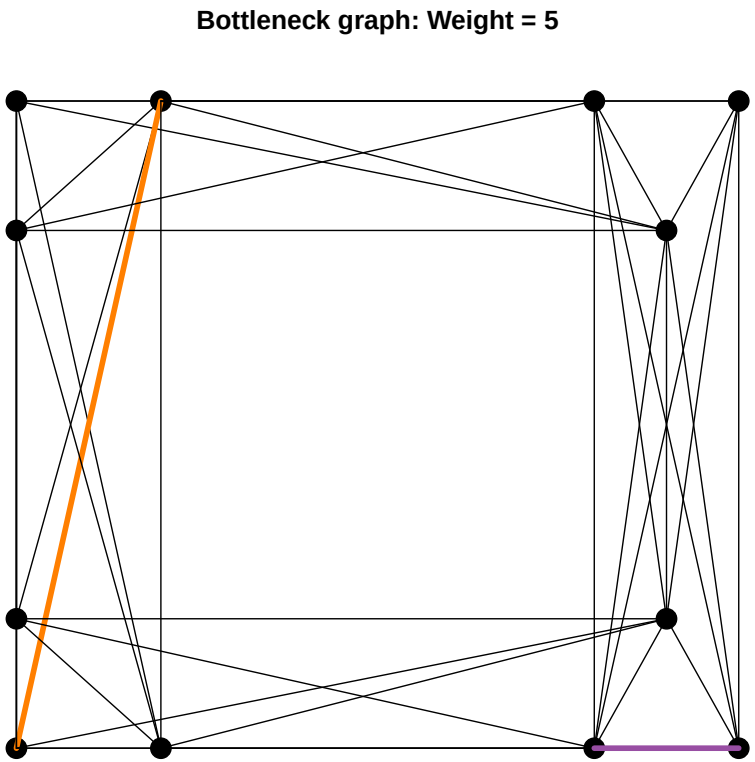
Bottleneck subgraph: In pictures

Complete graph



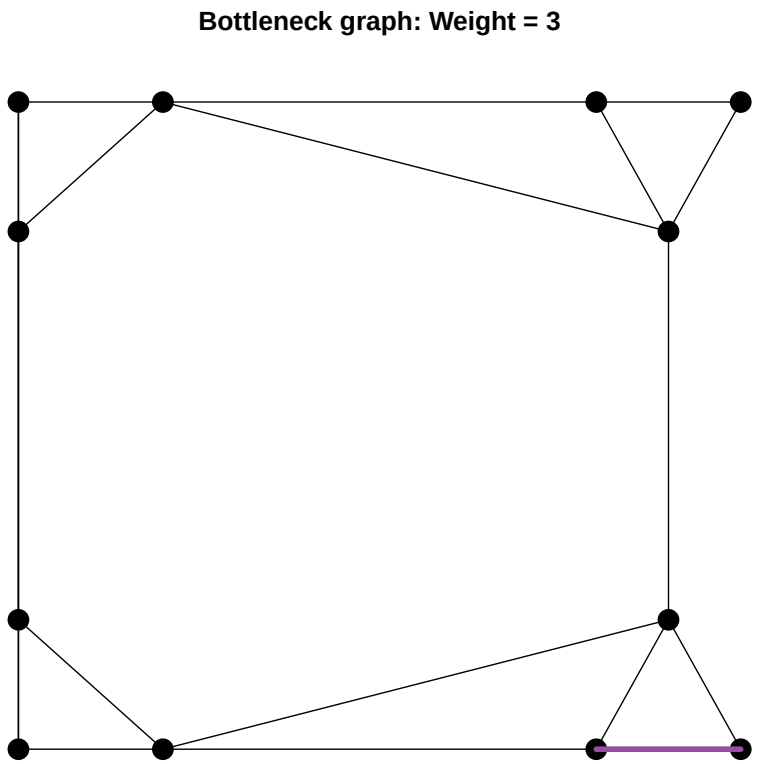
Bottleneck subgraph: In pictures

$B_5(G)$: Bottleneck subgraph of weight 5



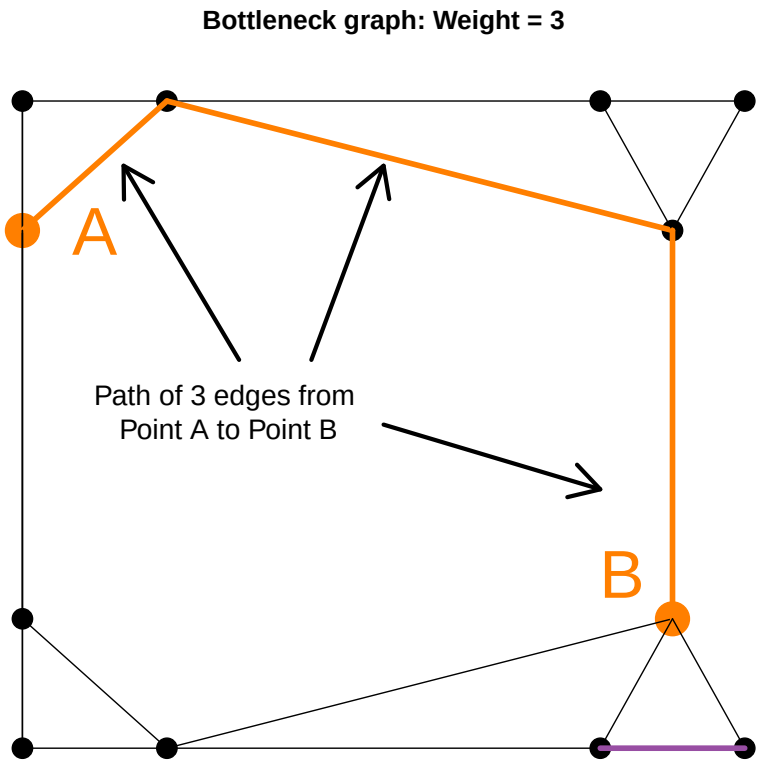
Bottleneck subgraph: In pictures

$B_3(G)$: Bottleneck subgraph of weight 3



Bottleneck subgraph: In pictures

$B_3(G)$: Bottleneck subgraph of weight 3



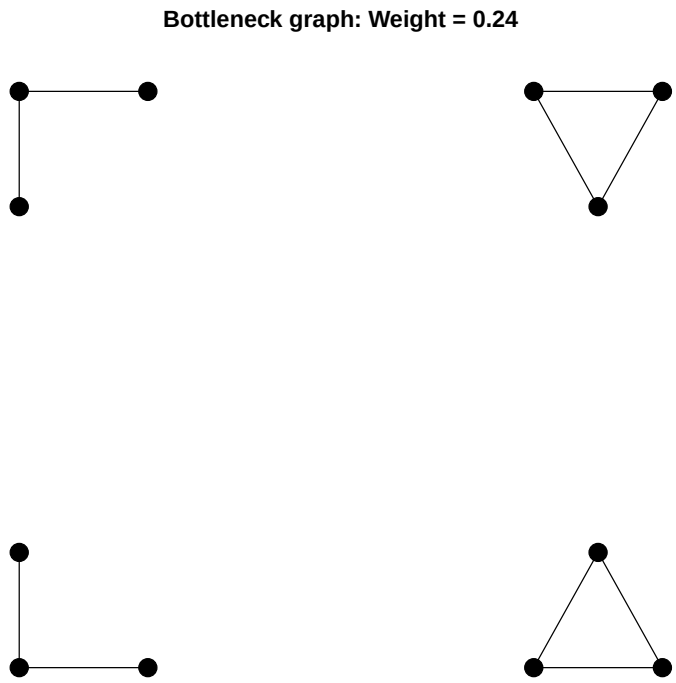
Approximate algorithm outline:

- Find the bottleneck subgraph of "appropriate" weight.
- Select block centers that are "just far enough apart."
- Grow from these block centers to obtain an approximately optimal partition—and thus, an approximately optimal blocking.
- Approach closely follows Hochbaum and Shmoys [1986].

Algorithm step-by-step: Find bottleneck graph

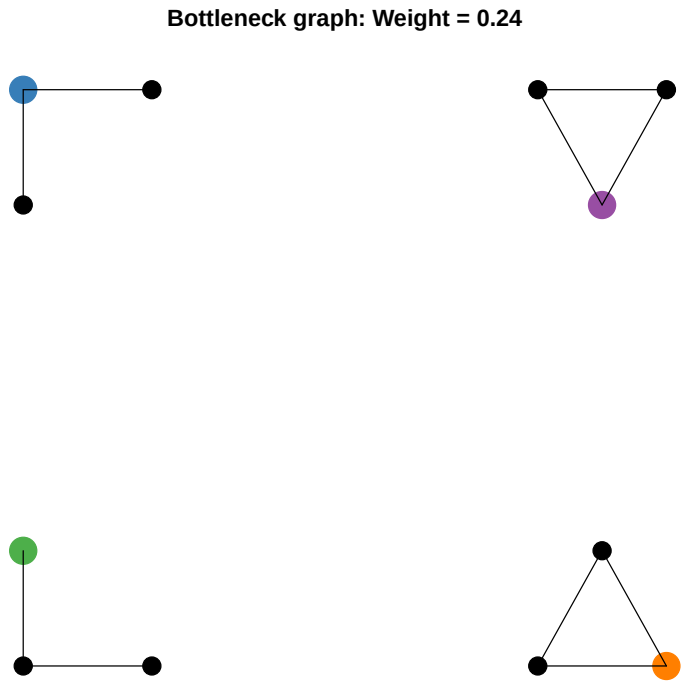
- Find the smallest λ^- such that each vertex in the bottleneck subgraph $B_{\lambda^-}(G)$ is connected to at least $t^* - 1$ edges.
- Can show that $\lambda^- \leq \lambda$, where λ is the minimum MIBD possible.
- At most $\frac{n(n-1)}{2}$ different bottleneck subgraphs—even an exhaustive search takes polynomial time.

$$t^* = 2$$



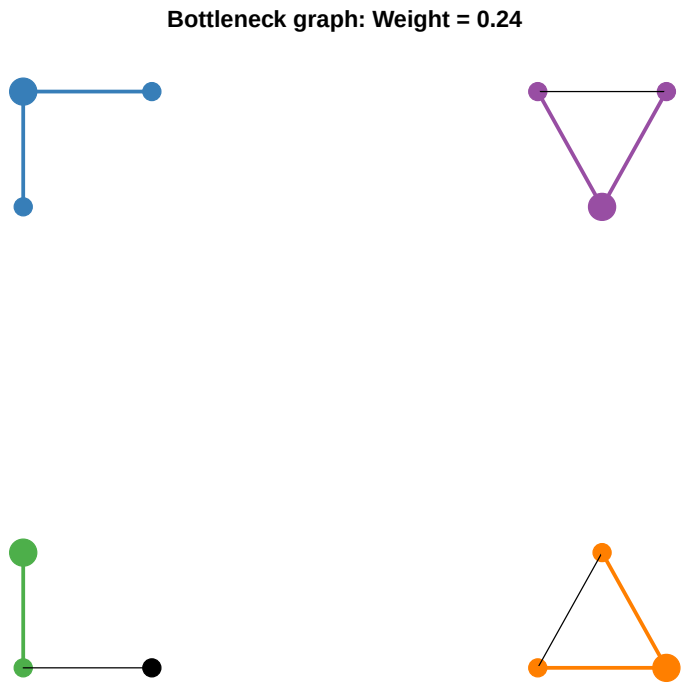
Algorithm step-by-step: Find block centers

- Find a set of vertices—*block centers*—such that:
  - There is no path of two edges or less connecting any of the vertices in the set.
  - For any vertex not in the set, there is a path of two edges or less that connects that vertex to one in the set.
- Any set will do, but some choices of centers are better.



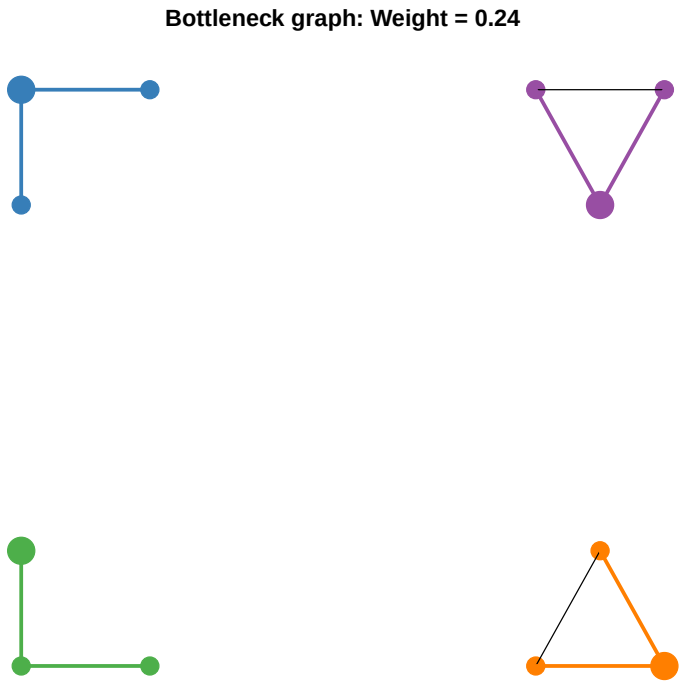
Algorithm step-by-step: Grow from block centers

- Form blocks comprised of a block center plus any vertices connected to that center by a single edge.
- The way our block centers were chosen (no path of two edges connects two block centers), these blocks will not overlap.
- At this point, these blocks contain at least t^* units (by edge connection criterion).



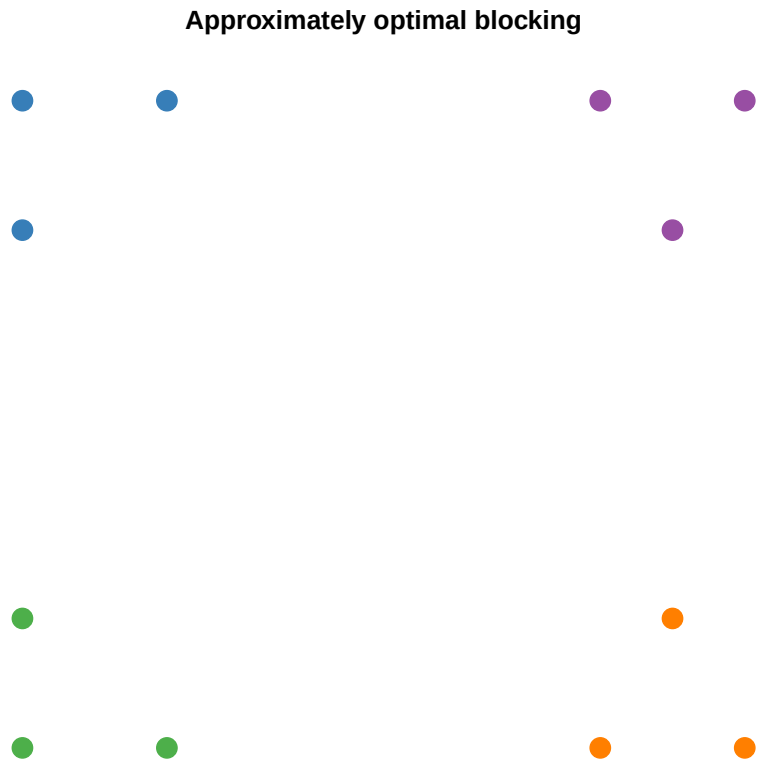
Algorithm step-by-step: Assign all unassigned vertices

- For each unassigned vertex, find the closest block center. Add that vertex to the center's corresponding block.
- The way our block centers were chosen, all unassigned vertices are at most a path of two edges away from a block center.



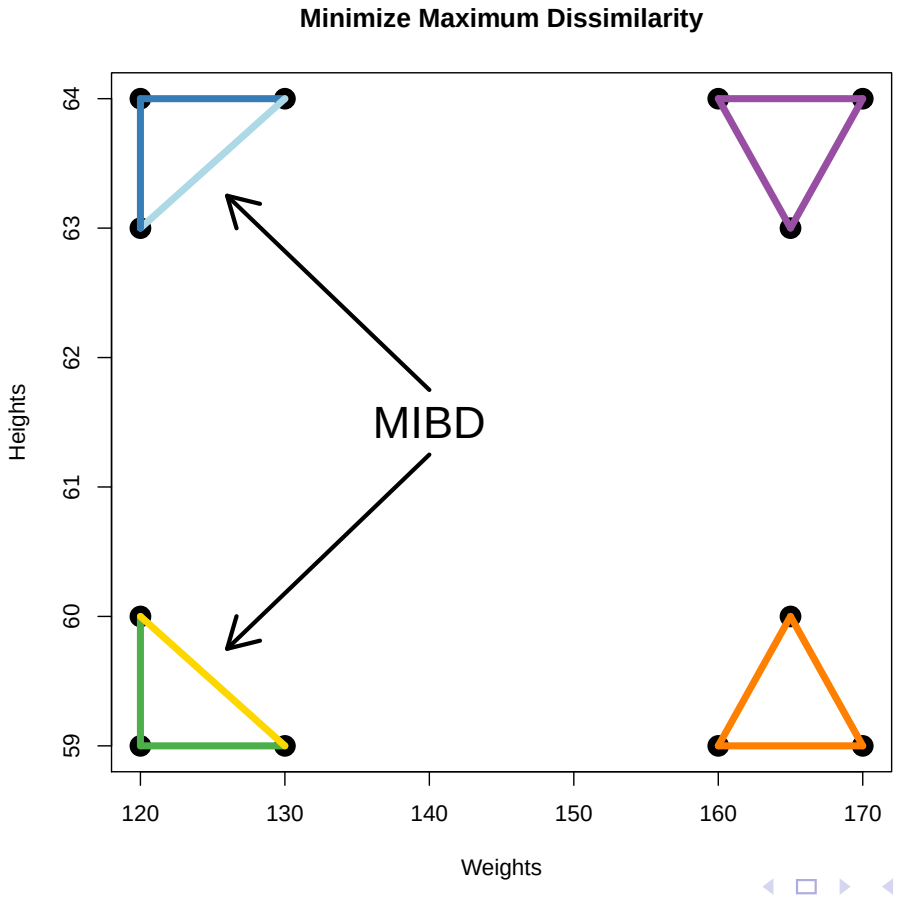
Our blocking

Our approximate algorithm came up with the following blocking:



A simple example:

Threshold $t^* = 2$. Dissimilarity = Mahalanobis distance.



Sketch of proof of approximate optimality

- Algorithm is guaranteed to obtain a blocking with MIBD $\leq 4\lambda$, though does much better than that in practice.

Sketch of proof of approximate optimality

- Algorithm is guaranteed to obtain a blocking with MIBD $\leq 4\lambda$, though does much better than that in practice.
- Sketch of proof:
- In the bottleneck subgraph used in the algorithm $B_{\lambda^-}(G)$, each edge has weight $w_{ij} \leq \lambda^- \leq \lambda$.

Sketch of proof of approximate optimality

- Algorithm is guaranteed to obtain a blocking with MIBD $\leq 4\lambda$, though does much better than that in practice.
- Sketch of proof:
- In the bottleneck subgraph used in the algorithm $B_{\lambda^-}(G)$, each edge has weight $w_{ij} \leq \lambda^- \leq \lambda$.
- Each vertex is no more than a path of two edges away from a block center $\Longrightarrow$
  Any two vertices in the same block can be connected by a path of four edges or fewer in $B_{\lambda^-}(G)$ (two from vertex 1 to the block center, two from the block center to vertex 2).

Sketch of proof cont'd

- For any two vertices $\{i\}, \{j\}$ within the same block, there is, in the worst case, a path of edges $(i, k_1), (k_1, k_2), (k_2, k_3), (k_3, j)$ in $B_{\lambda^-}(G)$ that connects $\{i\}$ to $\{j\}$.
- These edge weights satisfy:

$$w_{ik_1} + w_{k_1k_2} + w_{k_2k_3} + w_{k_3j} \leq 4\lambda^- \leq 4\lambda.$$

Sketch of proof cont'd

- Since edge weights satisfy the triangle inequality:

$$w_{ik} + w_{kj} \geq w_{ij}$$

it follows that

$$w_{ij} \leq w_{ik_1} + w_{k_1 k_2} + w_{k_2 k_3} + w_{k_3 j} \leq 4\lambda^{-} \leq 4\lambda.$$

Sketch of proof cont'd

- Since edge weights satisfy the triangle inequality:

$$w_{ik} + w_{kj} \geq w_{ij}$$

it follows that

$$w_{ij} \leq w_{ik_1} + w_{k_1k_2} + w_{k_2k_3} + w_{k_3j} \leq 4\lambda^{-} \leq 4\lambda.$$

- That is, every edge joining two vertices within the same block has weight $\leq 4\lambda$.
- The MIBD of the approximately optimal blocking is $\leq 4\lambda$.
- QED

Some final remarks about algorithm:

- Algorithm does not contain any inherently random components.
- Quick, local changes to the approximately optimal partition may "improve" the partition.
  For example, finding blocks containing $2t^*$ or more units and dividing them into 2 or more separate blocks.

Estimating treatment effects: Notation

- Recall, there are n units in total.
- There are b blocks, with n_c units within each block $c = 1, \ldots, b$.
- Units within each block are ordered in some way, let (k, c) denote the kth unit in block c.
- Recall, there are r treatment categories, numbered 1 through r. For simplicity, suppose $r \leq n_c$, and that r divides each n_c.
- Recall, treatment is completely randomized within each block, independently across blocks.

Notation cont'd:

- Let T_{kcs} denote a treatment indicator:
  - $T_{kcs} = 1$ if unit (k, c) receives treatment s; otherwise, $T_{kcs} = 0$.
  - Treatment indicators are random: $P(T_{kcs} = 1) = 1/r$.
  - Since one treatment is applied to any given unit, exactly one of $(T_{kc1}, T_{kc2}, \ldots, T_{kcr})$ will be equal to one.
  - Let $\#T_{cs}$ denote the number of units in block c that receive treatment s.
- Let Y_{kc} denote the observed response of (k, c). Since the response of (k, c) depends on its assigned treatment, Y_{kc} is random.

Neyman-Rubin potential outcomes model

- Let y_{kcs} denote the *potential outcome* for unit (k, c) under treatment s—the observed response of (k, c) had that unit received treatment s. Potential outcomes are non-random.
- The Neyman-Rubin potential outcomes framework assumes the following model for response [Splawa-Neyman et al., 1990, Rubin, 1974]:

$$Y_{kc} = y_{kc1} T_{kc1} + y_{kc2} T_{kc2} + \ldots + y_{kcr} T_{kcr}.$$

Neyman-Rubin potential outcomes model

- The Neyman-Rubin potential outcomes framework assumes the following model for response [Splawa-Neyman et al., 1990, Rubin, 1974]:

$$Y_{kc} = y_{kc1} T_{kc1} + y_{kc2} T_{kc2} + \ldots + y_{kcr} T_{kcr}.$$

- Similar to having n boxes; each box contains r tickets; draw one ticket from each box.
  - T_{kcs}: indicates if ticket s is drawn from box (k, c).
  - y_{kcs}: value written on ticket s in box (k, c).
  - Y_{kc}: value of the ticket drawn from box (k, c).
- Complete randomization: draws across boxes are not independent.

Neyman-Rubin potential outcomes model cont'd:

- The Neyman-Rubin potential outcomes framework assumes the following model for response [Splawa-Neyman et al., 1990, Rubin, 1974]:

$$Y_{kc} = y_{kc1} T_{kc1} + y_{kc2} T_{kc2} + \ldots + y_{kcr} T_{kcr}.$$

- Only treatment assignment is random. Everything else is non-random.
- Model makes the *stable-unit treatment value assumption* (SUTVA): the observed Y_{kc} only depends on which treatment is assigned to unit (k, c), and is not affected by the treatment assignment of any other unit (k', c').

Parameters of interest and estimators

- Parameters of interest: Sample average treatment effect of treatment s relative to treatment t (SATE_{st}):

$$\text{SATE}_{st} = \sum_{c=1}^{b} \sum_{k=1}^{n_c} \frac{y_{kcs} - y_{kct}}{n}$$

- Two unbiased estimators of SATE_{st} are the difference-in-means estimator and the the Horvitz-Thompson estimator.

$$\hat{\mu}_{st,\text{diff}} \equiv \sum_{c=1}^{b} \frac{n_c}{n} \sum_{k=1}^{n_c} \left(\frac{y_{kcs} T_{kcs}}{\# T_{cs}} - \frac{y_{kct} T_{kct}}{\# T_{ct}} \right),$$

$$\hat{\mu}_{st,\text{HT}} \equiv \sum_{c=1}^{b} \frac{n_c}{n} \sum_{k=1}^{n_c} \left(\frac{y_{kcs} T_{kcs}}{n_c/r} - \frac{y_{kct} T_{kct}}{n_c/r} \right).$$

- When r divides n_c, these estimators are the same.

Variance of estimators

When r divides n_c:

$$\mathrm{Var}(\hat{\mu}_{st,\mathrm{diff}}) = \mathrm{Var}(\hat{\mu}_{st,\mathrm{HT}})$$

$$= \sum_{c=1}^{b} \frac{n_c^2}{n^2} \left(\frac{r-1}{n_c - 1}(\sigma_{cs}^2 + \sigma_{ct}^2) + 2\frac{\gamma_{cst}}{n_c - 1} \right)$$

$$\mu_{cs} = \frac{1}{n_c} \sum_{k=1}^{n_c} y_{kcs}$$

$$\sigma_{cs}^2 = \frac{1}{n_c} \sum_{k=1}^{n_c} (y_{kcs} - \mu_{cs})^2$$

$$\gamma_{cst} = \frac{1}{n_c} \sum_{k=1}^{n_c} (y_{kcs} - \mu_{cs})(y_{kct} - \mu_{ct})$$

Variance of estimators

When r divides n_c:

$$\begin{aligned} &\mathrm{Var}(\hat{\mu}_{st,\mathrm{diff}}) = \mathrm{Var}(\hat{\mu}_{st,\mathrm{HT}}) \\ = &\sum_{c=1}^{b} \frac{n_c^2}{n^2} \left(\frac{r-1}{n_c - 1}(\sigma_{cs}^2 + \sigma_{ct}^2) + 2\frac{\gamma_{cst}}{n_c - 1} \right) \end{aligned}$$

- Note: σ_{cs}^2 and σ_{ct}^2 are estimable, γ_{cst} not directly estimable (though there are some tricks around this [Abadie and Imbens, 2008, Imbens, 2011]).
- Subtle differences in variances when r does not divide all n_c.

When does blocking help?

- Blocking vs. completely randomized treatment assignment (no blocking): which estimates of $SATE_{st}$ have lower variance?
- Matched-pairs: Blocking is better than not blocking when $\sigma^2_{c(s+t)}$ is, on average, about 1/2 of σ^2_{s+t}.

$$\sigma^2_{c(s+t)} = \frac{1}{n_c}\sum_{k=1}^{n_c}(y_{kcs} + y_{kct} - \mu_{cs} - \mu_{ct})^2$$

$$\sigma^2_{s+t} = \frac{1}{n}\sum_{c=1}^{b}\sum_{k=1}^{n_c}(y_{kcs} + y_{kct} - \mu_s - \mu_t)^2$$

When does blocking help?

- Blocking vs. completely randomized treatment assignment (no blocking): which estimates of $SATE_{st}$ have lower variance?
- Matched-pairs: Blocking is better than not blocking when $\sigma^2_{c(s+t)}$ is, on average, about $1/2$ of σ^2_{s+t}.
- This is satisfied when units are randomly paired.
  Any improvement to completely random pairing $\rightarrow$ matched-pairs better than no blocking.
- Work in progress: Generalize to arbitrary blocking configurations.

Future Work

- Algorithm improvements:
  - Decrease run-time of the algorithm.
  - Add additional structure to improve partition quickly.
  - Can factor of 4 be improved?
- Apply graph partitioning techniques to other statistical problems.
  - Clustering—alternative to k-means.
  - Other problems in nonparametric statistics.

Bibliography I

A. Abadie and G.W. Imbens. Estimation of the conditional variance in paired experiments. Annales d'Economie et de Statistique, No. 91-92:175–187, 2008.

D.S. Hochbaum and D.B. Shmoys. A unified approach to approximation algorithms for bottleneck problems. Journal of the ACM (JACM), 33(3):533–550, 1986.

Guido W. Imbens. Experimental design for unit and cluster randomized trials, 2011.

Kari Lock Morgan and Donald B Rubin. Rerandomization to improve covariate balance in experiments. Annals of Statistics, 40(2):1263–1282, 2012.

Bibliography II

L.W. Miratrix, J.S. Sekhon, and B. Yu. Adjusting treatment effect estimates by post-stratification in randomized experiments. JR Stat. Soc. Ser. B. Stat. Methodol. To appear, 2012.

P.R. Rosenbaum. Optimal matching for observational studies. Journal of the American Statistical Association, 84(408): 1024–1032, 1989.

Donald B Rubin. Estimating causal effects of treatments in randomized and nonrandomized studies. Journal of Educational Psychology; Journal of Educational Psychology, 66(5):688, 1974.

Jerzy Splawa-Neyman, DM Dabrowska, and TP Speed. On the application of probability theory to agricultural experiments. essay on principles. section 9. Statistical Science, 5(4):465–472, 1990.