Improving Experiments by Optimal Blocking: Minimizing the Maximum Within-block Distance

Michael J. Higgins
Jasjeet Sekhon

University of California at Berkeley

July 18, 2013

Goal

We provide the first polynomial time algorithm for blocking with multiple treatment categories

- ensures good covariate balance by design
- is efficient enough to be used in large experiments
- better balance than greedy methods
- can estimate conditional variances

Covariate imbalance in randomized experiments

- PROBLEM: In finite samples, there is a probability of bad covariate balance between treatment groups
- Bad imbalance on important covariates $\rightarrow$ Imprecise estimates of treatment effects
- Even in large samples, covariate adjustment may improve precision of estimates

Adjustment and covariate imbalance

- Regression adjustment [Freedman, 2008, Lin, 2012]
- Post-stratification [Miratrix, Sekhon, and Yu, 2013]:
  - Group similar units together after *after* randomization
  - SATE/PATE results good; *ex post* problems arise
  - Data mining concerns
- Re-randomization [Lock Morgan and Rubin, 2012]:
  - Repeat randomly assigning treatments until covariate balance is "acceptable"
- LESSON: design the randomization to build in adjustment

Current blocking approaches for two treatments

- Matched-pairs blocking: Pair "most-similar" units together. For each pair, randomly assign one unit to treatment, one to control [Imai, 2008]
- Optimal Multivariate Matching Before Randomization [Greevy, Lu, Silber, and Rosenbaum, 2004]
- Optimal-greedy blocking [Moore, 2012]

Matched-Pairs

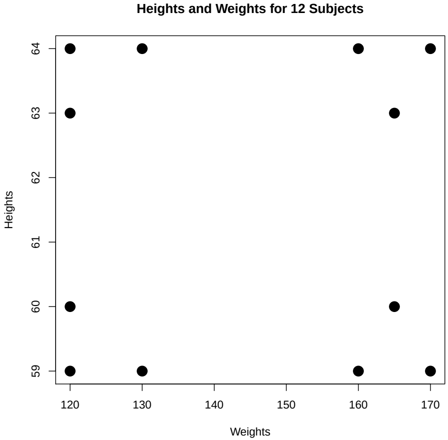
- No efficient way to extend approach to more than two treatment categories
- Fixed block sizes (2 units): design may pair units from different clusters
- Cannot estimate conditional variances [Imbens, 2011]
- Difficulty with treatment effect heterogeneity
- Worse problems with some tests—e.g., rank sum

Blocking by minimizing the Maximum Within-Block Distance (MWBD)

- Experiment with n units and r treatment categories
- 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 distance between any two units within a block—the MWBD—is minimized
- Threshold t^*: Allows designs with multiple treatment categories, multiple replications of treatments within a block

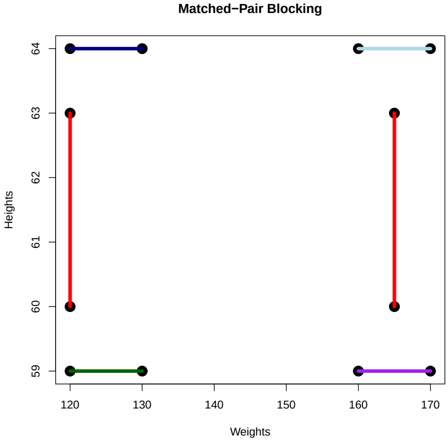
A simple example:

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



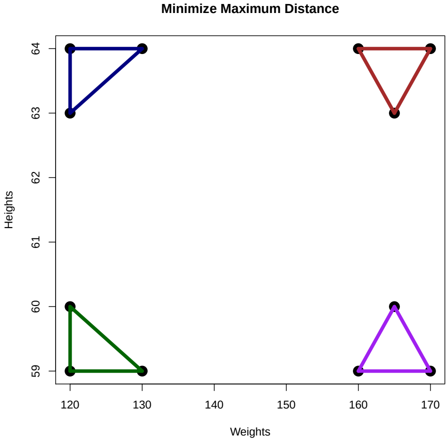
A simple example:

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



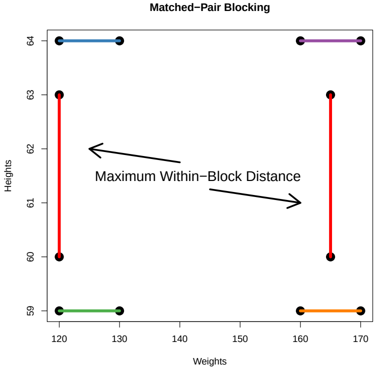
A simple example:

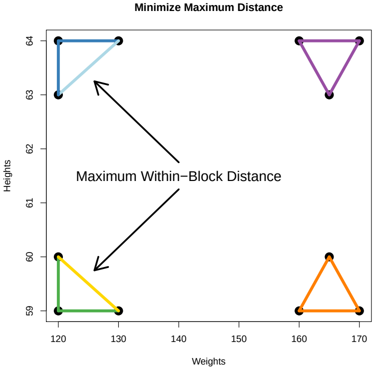
Threshold $t^* = 2$. Distance = Mahalanobis distance.



A simple example:

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





Optimal blocking and approximately optimal blocking

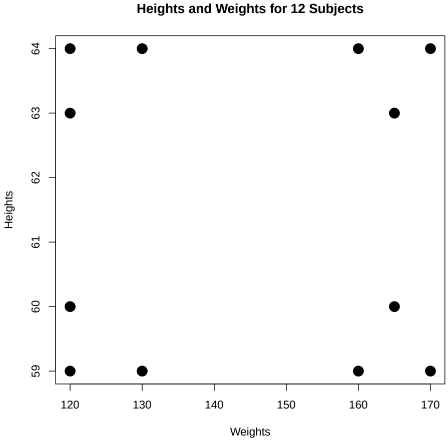
- *For all blockings that contain at least t^* units*:
- Let λ denote the smallest MWBD achievable by such a blocking—any blocking that meets this bound is called an *optimal blocking*
- Finding an optimal blocking is an NP-hard problem—feasible to find in small experiments, may not be feasible in large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

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 placed in the same block
- Edge-weights are some measure of distance between pretreatment covariates (e.g. Mahalanobis distance)

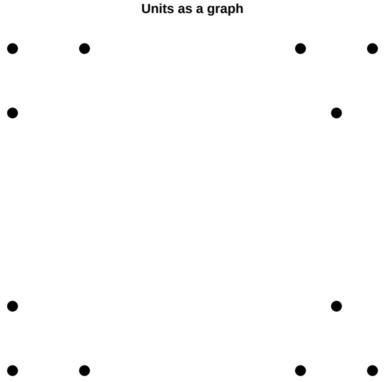
Viewing experimental units as a graph: In pictures

Distance = Mahalanobis distance.



Viewing experimental units as a graph: In pictures

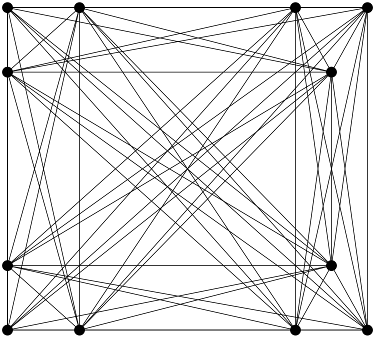
Distance = Mahalanobis distance.



Viewing experimental units as a graph: In pictures

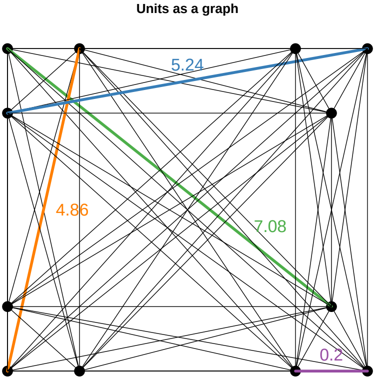
Distance = Mahalanobis distance.

Units as a graph



Viewing experimental units as a graph: In pictures

Distance = 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 $\rightarrow$ 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}

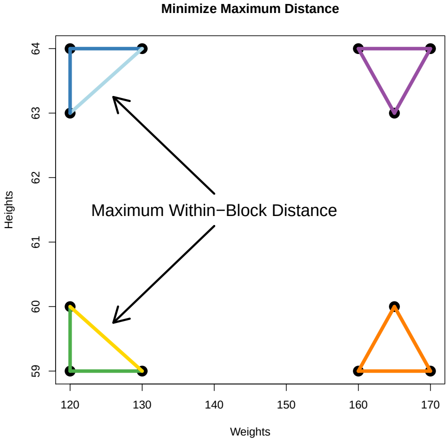
Note about edge weights

- If our concern is covariate balance, natural choices for edge weights measure distance between block covariates—e.g., Mahalanobis, L^1, L^2, distances
- Our method only requires weights to satisfy the triangle inequality: for any distinct vertices $\{i\}, \{j\}, \{k\}$,

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

A simple example:

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



Optimal blocking as a graph partitioning problem

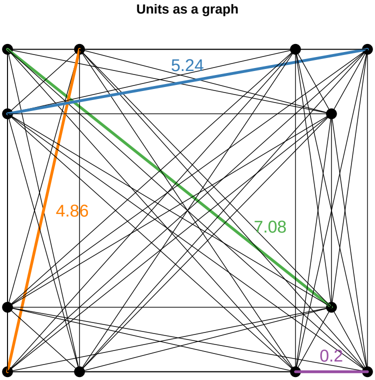
- A *partition* of V is a division of V into disjoint *blocks* of vertices $(V_1, V_2, \ldots, V_\ell)$
- Blocking of units $\leftrightarrow$ Partition of a graph:
  Two units are in the same block of the blocking if and only if their corresponding units are in the same block of the partition
- Optimal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

Bottleneck subgraphs

- Bottleneck subgraphs helpful for solving partitioning problems
- Define the *bottleneck subgraph for mamimum weight of* 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

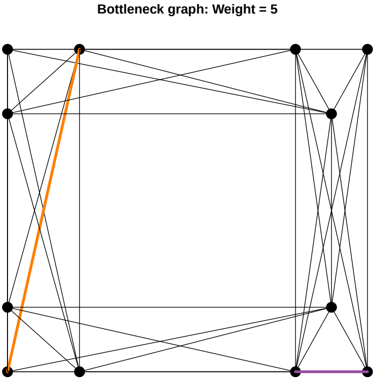
Bottleneck subgraph: In pictures

Complete graph



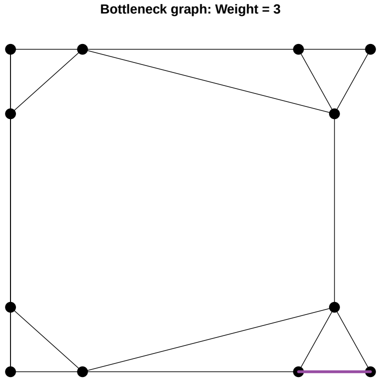
Bottleneck subgraph: In pictures

Bottleneck subgraph of weight 5



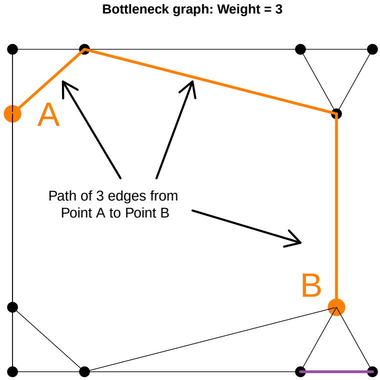
Bottleneck subgraph: In pictures

Bottleneck subgraph of weight 3



Bottleneck subgraph: In pictures

Bottleneck subgraph of weight 3



Approximate algorithm outline:

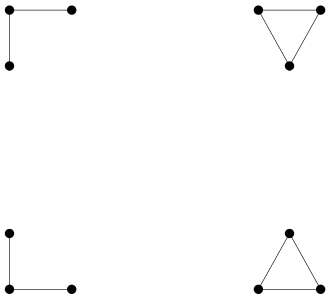
- Find the bottleneck subgraph of “appropriate” weight
  - can use k-nearest neighbor graph
- Select block centers that are “just far enough apart”
- Grow from these block centers to obtain an approximately optimal partition—and thus, an approximately optimal blocking
- Approach closely follows Hochbaum and Shmoys [1986]

Algorithm step-by-step: Find bottleneck graph

- Find smallest weight threshold λ^- such that each vertex in the corresponding bottleneck subgraph is connected to at least $t^* - 1$ edges.
- Can show that $\lambda^- \leq \lambda$, where λ is the smallest MWBD possible.
- Bottleneck subgraph can be constructed in polynomial time.

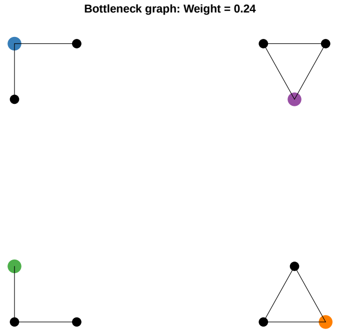
$t^* = 2$

Bottleneck graph: Weight = 0.24



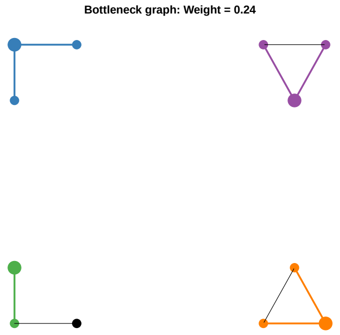
Algorithm step-by-step: Find block centers

- 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.



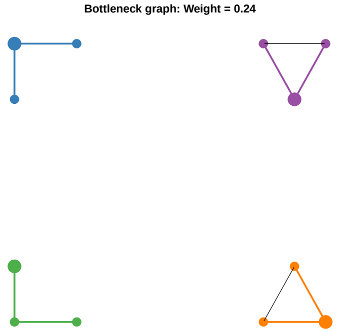
Algorithm step-by-step: Grow from block centers

- 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).



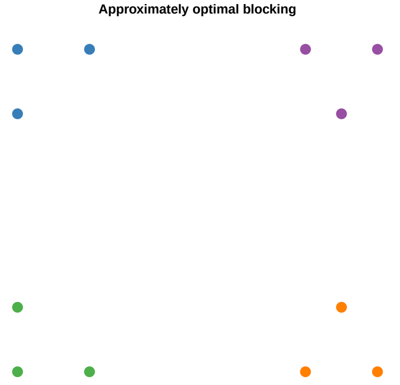
Algorithm step-by-step: Assign all unassigned vertices

- 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.



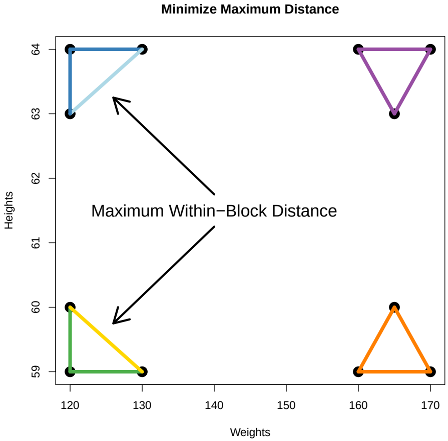
Our blocking

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 MWBD $\leq 4\lambda$, though does much better than that in practice.

Sketch of proof of approximate optimality

- Algorithm is guaranteed to obtain a blocking with MWBD $\leq 4\lambda$, though does much better than that in practice.
- Sketch of proof:
- Each vertex is at most a path of two edges away from a block center $\implies$
  In the worst case: two vertices $\{i\}, \{j\}$ in the same block can be connected by a path of four edges in the bottleneck subgraph (two from vertex $\{i\}$ to the block center, two from the block center to vertex $\{j\}$).

Sketch of proof cont'd

- Each vertex is at most a path of two edges away from a block center $\Longrightarrow$
  In the worst case: two vertices $\{i\}, \{j\}$ in the same block can be connected by a path of four edges in the bottleneck subgraph (two from vertex $\{i\}$ to the block center, two from the block center to vertex $\{j\}$).
- In worst case: $(i, k_1), (k_1, k_2), (k_2, k_3), (k_3, j)$ is a path of four edges connecting $\{i\}$ to $\{j\}$.
- Each edge has weight at most $\lambda^- \Longrightarrow$
  The corresponding edge weights satisfy:

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

Sketch of proof cont'd

- Since edge weights satisfy the triangle inequality:

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

it follows that

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

Sketch of proof cont'd

- Since edge weights satisfy the triangle inequality:

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

  it follows that

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

- That is, every edge joining two vertices within the same block has weight $\leq 4\lambda$.
- The maximum within-block distance 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 blocking may improve the blocking. (e.g., divide large blocks into smaller blocks, swap units between blocks)

Neyman-Rubin potential outcomes model

- The Neyman-Rubin potential outcomes framework assumes the following model for response [Splawa-Neyman, Dabrowska, and Speed, 1990, Rubin, 1974]:

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

- Y_{kc}: Observed response of kth unit in block c.
- y_{kct}: Potential outcome of the unit under treatment t.
- T_{kct}: Treatment indicators. $T_{kct} = 1$ if the unit receives treatment t, $T_{kct} = 0$ otherwise.

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.

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

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

- Assume complete randomization of treatment, r divides n_c.

Variance of estimators

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

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

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

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

Variance of estimators

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

- Note: σ_{cs}^2 and σ_{ct}^2 are estimable, γ_{cst} not directly estimable.
- Conservative estimate:

$$\widehat{\mathrm{Var}} = \sum_{c=1}^{b} \frac{n_c^2}{n^2} \left(\frac{2(r-1)}{n_c-1}(\hat{\sigma}_{cs}^2 + \hat{\sigma}_{ct}^2) \right)$$

- Small differences for more general treatment assignments.

When does blocking help?

- Blocking vs. completely randomized treatment assignment (no blocking): which estimates of $SATE_{st}$ have lower variance?
- Blocking helps if and only if:

$$\sum_{c=1}^{b} n_c^2 \left[\left(\frac{(r-1)(\sigma_s^2 + \sigma_t^2) + 2\gamma_{st}}{\sum n_c^2 (n-1)} \right) - \left(\frac{(r-1)(\sigma_{cs}^2 + \sigma_{ct}^2) + 2\gamma_{cst}}{n^2 (n_c - 1)} \right) \right] \geq 0$$

- Intuitive to make $\sigma_{cs}^2, \sigma_{ct}^2$ small w.r.t. σ_s^2, σ_t^2, but other blocking designs may also improve treatment effect estimates.

Can blocking hurt?

- When blocking is completely randomized:

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

$$= \sum_{c=1}^{b} n_c^2 \left(\frac{(r-1)(\sigma_s^2 + \sigma_t^2) + 2\gamma_{st}}{\sum n_c^2(n-1)}\right)$$

Blocked variance = Completely randomized variance

- Any improvement to completely random blocking $\rightarrow$ Reduced variance in treatment effect estimates.

Future Work

- Apply graph partitioning techniques to other statistical problems:
    - Clustering—alternative to k-means.
    - Apply to matching problems.
    - Other problems in nonparametric statistics.
- Improve theoretic results of algorithm.

Bibliography I

David A. Freedman. On regression adjustments in experiments with several treatments. The annals of applied statistics, 2(1): 176–196, 2008.

Robert Greevy, Bo Lu, Jeffrey H. Silber, and Paul Rosenbaum. Optimal multivariate matching before randomization. Biostatistics, 5(4):263–275, 2004.

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.

K. Imai. Variance identification and efficiency analysis in randomized experiments under the matched-pair design. Statistics in medicine, 27(24):4857–4873, 2008.

Bibliography II

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

Winston Lin. Agnostic notes on regression adjustments to experimental data: Reexamining freedman's critique. Annals of Applied Statistics, 2012.

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

Luke W. Miratrix, Jasjeet S. Sekhon, and Bin Yu. Adjusting treatment effect estimates by post-stratification in randomized experiments. Journal of the Royal Statistical Society, Series B, 75(2):369–396, 2013.

Bibliography III

Ryan T Moore. Multivariate continuous blocking to improve political science experiments. Political Analysis, 20(4):460–479, 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.