Supplement: Generalized full matching

Fredrik Sävje[1], Michael J. Higgins[2], and Jasjeet S. Sekhon[3]

[1]*Department of Political Science & Department of Statistics and Data Science, Yale University.*
[2]*Department of Statistics, Kansas State University.*
[3]*Travers Department of Political Science and Department of Statistics, UC Berkeley.*

S1 Brief overview of graph theory

Graph A graph $G = (\mathbf{V}, \mathbf{E})$ consists of a set of indices $\mathbf{V} = \{a, b, \cdots\}$, called vertices, and a set of 2-element subsets of $\mathbf{V}$, called edges. If an edge $\{i, j\} \in \mathbf{E}$, we say that i and j are connected in the graph. In a directed graph (or digraph), the edges (which are called arcs in a digraph) are ordered sets (i, j). In other words, i can be connected to j without the reverse being true in a digraph.

Weighted graph A weighted graph assigns a weight or cost to each edge or arc. In our case, the weights are exclusively given by the distance of the connected vertices according to the metric used in the matching problem.

Adjacent Vertices i and j are adjacent in G if an edge (or an arc) connecting i and j exists in $\mathbf{E}$.

Geodesic distance The geodesic distance between i and j in G is the number of edges or arcs (in our case, of any directionality) in the shortest path connecting i and j in G.

Subgraph $G_1 = (\mathbf{V}_1, \mathbf{E}_1)$ is a subgraph of $G_2 = (\mathbf{V}_2, \mathbf{E}_2)$ if $\mathbf{V}_1 \subseteq \mathbf{V}_2$ and $\mathbf{E}_1 \subseteq \mathbf{E}_2$. In that case, we also say that G_2 is a supergraph of G_1. G_1 is a spanning subgraph of G_2 if $\mathbf{V}_1 = \mathbf{V}_2$.

Complete graph G is complete if $\{i, j\} \in \mathbf{E}$ for any two vertices $i, j \in \mathbf{V}$. If G is directed, both (i, j) and (j, i) must be in $\mathbf{E}$.

Union The union of $G_1 = (\mathbf{V}_1, \mathbf{E}_1)$ and $G_2 = (\mathbf{V}_2, \mathbf{E}_2)$ is $G_1 \cup G_2 = (\mathbf{V}_1 \cup \mathbf{V}_2, \mathbf{E}_1 \cup \mathbf{E}_2)$.

Graph difference The difference between two graphs $G_1 = (\mathbf{V}, \mathbf{E}_1)$ and $G_2 = (\mathbf{V}, \mathbf{E}_2)$ spanning the same set of vertices is $G_1 - G_2 = (\mathbf{V}, \mathbf{E}_1 \setminus \mathbf{E}_2)$.

Independent set A set of vertices $\mathbf{I} \subseteq \mathbf{V}$ is independent in $G = (\mathbf{V}, \mathbf{E})$ if no two vertices in the set are adjacent:

$$\forall\, i, j \in \mathbf{I},\ \{i, j\} \notin \mathbf{E}.$$

Maximal independent set An independent set of vertices $\mathbf{I}$ in $G = (\mathbf{V}, \mathbf{E})$is maximal if for any additional vertex $i \in \mathbf{V}$ the set $\{i\} \cup \mathbf{I}$ is not independent:

$$\forall\, i \in \mathbf{V} \setminus \mathbf{I},\ \exists j \in \mathbf{I},\ \{i, j\} \in \mathbf{E}.$$

Cluster graph The (directed) cluster graph induced by some partition of $\mathbf{V}$ is the graph where arcs exists between any pair of units in the same component of the partition and no other arcs exist.

Adjancency matrix The adjacency matrix $\mathbf{A}$ of a graph $G = (\mathbf{V}, \mathbf{E})$ with n vertices is a n-by-n binary matrix where the entry i, j is one if the edge $\{i, j\}$ or arc (i, j) is in $\mathbf{E}$, and otherwise zero.

S2 Proofs

We here provide proofs for the propositions in the main paper. The relevant propositions from the paper are restated with their original numbering for reference.

S2.1 Optimality

Recall the two objective functions:

$$\begin{aligned} L_{\text{BN}}(\mathbf{M}) &= \max_{\mathbf{m}\in\mathbf{M}} \max\{d(i,j) : i,j \in \mathbf{m}\}, \\ L_{\text{WBN}}(\mathbf{M}) &= \max_{\mathbf{m}\in\mathbf{M}} \max\{d(i,j) : i,j \in \mathbf{m} \wedge W_i \neq W_j\}. \end{aligned}$$

Lemma 8. *The closed neighborhood of each vertex in the C-compatible nearest neighbor digraph $G_C = (\mathbf{V}, \mathbf{E}_C)$ satisfies the matching constraints $C = (c_1, \cdots, c_k, t)$:*

$$\forall i \in \mathbf{V},\ \forall x \in \{1, \cdots, k\},\ |\,\mathrm{N}[i] \cap \mathbf{w}_x| \geq c_x \qquad \textit{and} \qquad \forall i \in \mathbf{V},\ |\,\mathrm{N}[i]| \geq t.$$

Proof. This follows directly from the construction of G_C. For each treatment-specific constraint, $c_1, \cdots, c_k$, the first step of the algorithm ensures that each vertex has that many arcs pointing to units assigned to the corresponding treatment condition. Similarly, if $t > c_1 + c_2 + \cdots + c_k$, the second step includes additional arcs so that each vertex has t outward-pointing arcs in total. □

Lemma S1. *By the completion of Step 4 of the generalized full matching algorithm, each vertex has at least one labeled vertex in its neighborhood in G_C.*

Proof. By definition, all vertices in a closed neighborhood of a seed are labeled. That is, ℓ is a labeled vertex if and only if $\exists i \in \mathbf{S}, \ell \in \mathrm{N}[i]$. Suppose the lemma does not hold, i.e., that some vertex i does not have a labeled vertex in its neighborhood:

$$\exists i, \forall \ell \in \mathrm{N}[i], \nexists j \in \mathbf{S}, \ell \in \mathrm{N}[j]. \tag{1}$$

It follows directly that i cannot be a seed as all vertices in its neighborhood would otherwise be labeled by definition. Note that (1) entails that i's neighborhood does not have any overlap with any seed's neighborhood:

$$\forall j \in \mathbf{S}, \mathrm{N}[i] \cap \mathrm{N}[j] = \varnothing,$$

However, this violates the maximality condition in the definition of a valid set of seeds (see the third step of the algorithm) and, subsequently, a vertex such as i is not possible. □

Lemma S2. $\mathbf{M}_{\text{ALG}}$ *is an admissible generalized full matching with respect to constraint $C = (c_1, \cdots, c_k, t)$:*

$$\mathbf{M}_{\text{ALG}} \in \mathcal{M}_C.$$

Proof. We must show that $\mathbf{M}_{\text{ALG}}$ satisfies the four conditions of an admissible generalized full matching in Definition 2. Step 5 of the algorithm ensures that $\mathbf{M}_{\text{ALG}}$ is spanning. At this step, any vertex that lacks a label will be assigned the same label as one of the labeled vertices in its neighborhood. Lemma S1 ensures that at least one labeled vertex exists in the neighborhoods of the unassigned vertices.

No vertex is assigned more than one label; that is, $\mathbf{M}_{\text{ALG}}$ is disjoint. To see this, observe that vertices are only assigned labels in either Step 4 or 5, but never in both. Step 3 ensures that the neighborhoods of the seeds are non-overlapping. Thus vertices will be assigned at most one label in Step 4. In Step 5, vertices are explicitly assigned only one label even if several labels could be represented in a vertex's neighborhood.

The remaining two conditions in Definition 2 are ensured by Lemma 8. Step 4 of the algorithm ensures that each matched group is a superset of a seed's neighborhood. From Lemma 8, we have that this neighborhood will satisfy the matching constraints. □

Lemma S3. *If the arc weights in the C-compatible nearest neighbor digraph $G_C = (\mathbf{V}, \mathbf{E}_C)$ are bounded by some λ, the maximum within-group distance in $\mathbf{M}_{\text{ALG}}$ is bounded by 4λ:*

$$\forall (i,j) \in \mathbf{E}_C,\ d(i,j) \leq \lambda \implies \max_{\mathbf{m} \in \mathbf{M}_{\text{ALG}}} \max\{d(i,j) : i,j \in \mathbf{m}\} \leq 4\lambda.$$

Proof. First, consider the distance from any vertex i to the seed in its matched group, denoted j. If i is a seed, we have $i = j$, as each matched group contains exactly one seed by construction. By the self-similarity property of distance metrics, the distance is zero: $d(i,j) = 0$. If i is a labeled vertex (i.e., assigned a label in Step 4 of the algorithm), we have $(j,i) \in \mathbf{E}_C$ by definition of labeled vertices. By the premise of the lemma, $d(j,i)$ is bounded by λ. Due to the symmetry property of distance metrics, this also bounds $d(i,j)$. If i is not a labeled vertex (i.e., assigned a label in Step 5), Lemma S1 tells us that it will be adjacent in G_C to a labeled vertex ℓ in its matched group. We have $(i,\ell) \in \mathbf{E}_C$ so the distance between i and ℓ is bounded by λ. As ℓ is labeled, we have $(j,\ell) \in \mathbf{E}_C$ which implies that $d(j,\ell) \leq \lambda$. From the triangle inequality property of metrics, we have that the distance between unit i and the seed j is at most 2λ.

Now consider any two vertices assigned to the same matched group. We have shown that the distance from each of these vertices to their (common) seed is at most 2λ. By applying the triangle inequality once more, we bound the distance between the two non-seed vertices by 4λ. □

Lemma S4. *The C-compatible nearest neighbor digraph has the smallest maximum arc weight among all digraphs compatible with C, i.e., all graphs for which each vertex's closed neighborhood contains $c_1, c_2, \cdots, c_k$ vertices of each treatment condition and t vertices in total.*

Proof. The definition of the directed neighborhoods is asymmetric in the sense that j is not necessarily in i's neighborhood even if i is in j's neighborhood. Thus, whether a vertex's neighborhood satisfies the constraints is independent of whether other vertices' neighborhoods do so. As a consequence, to minimize the maximum arc weight, we can simply minimize the maximum arc weight in each neighborhood separately. To minimize the arc weights in a single neighborhood, we draw arcs to the vertices closest to the vertex so that the matching constraints are fulfilled. This is exactly how the algorithm constructs the C-compatible nearest neighbor digraph. □

Lemma 9. *The distance between any two vertices connected by an arc in the C-compatible nearest neighbor digraph $G_C = (\mathbf{V}, \mathbf{E}_C)$ is less or equal to the maximum within-group distance in an optimal matching:*

$$\forall (i,j) \in \mathbf{E}_C,\ d(i,j) \leq \min_{\mathbf{M} \in \mathcal{M}_C} L_{\text{BN}}(\mathbf{M}).$$

Proof. Let w^* be the maximum within-group distance in an optimal matching and let w_C^+ be the maximum weight of an arc in G_C:

$$w^* = \min_{\mathbf{M} \in \mathcal{M}_C} L_{\text{BN}}(\mathbf{M}) \qquad \text{and} \qquad w_C^+ = \max\{d(i,j) : (i,j) \in E_C\}.$$

Furthermore, let $B_C = (\mathbf{U}, \mathbf{E}_C^b)$ be the digraph that contains arcs between all units at a distance strictly closer than w_C^+:

$$\mathbf{E}_C^b = \{(i,j) : d(i,j) < w_C^+\}.$$

B_C must contain a vertex whose neighborhood does not satisfy the size constraints. If no such vertex exists, a digraph compatible with C with a smaller maximum arc weight than in G_C exists as a subgraph of B_C. This contradicts Lemma S4.

Let $B_{\text{OP}} = (\mathbf{U}, \mathbf{E}_{\text{OP}}^b)$ be the digraph that contains arcs between all units at a distance weakly closer than w^*:

$$\mathbf{E}_{\text{OP}}^b = \{(i,j) : d(i,j) \leq w^*\}.$$

By construction, B_{OP} is a supergraph of the cluster graph induced by the optimal matching. That is, arcs are drawn in B_{OP} between all units assigned to the same matched group in the optimal matching. As the optimal matching

is admissible, each vertex's neighborhood in B_{OP} is compatible with C.

Suppose that the lemma does not hold: $w_C^+ > w^*$. It follows that $\mathbf{E}_{\text{OP}}^b \subset \mathbf{E}_C^b$. Because at least one vertex's neighborhood does not satisfy the size constraint in B_C, that must also be the case in B_{OP}. This, however, implies that the optimal matching is not admissible which, in turn, contradicts optimality. □

Theorem 10. $\mathbf{M}_{\text{ALG}}$ *is a 4-approximate generalized full matching with respect to the matching constraint* $\mathcal{C} = (c_1, \cdots, c_k, t)$ *and matching objective* L_{BN}*:*

$$\mathbf{M}_{\text{ALG}} \in \mathcal{M}_C \qquad \textit{and} \qquad L_{\text{BN}}(\mathbf{M}_{\text{ALG}}) \leq \min_{\mathbf{M} \in \mathcal{M}_C} 4L_{\text{BN}}(\mathbf{M}).$$

Proof. Admissibility follows from Lemma S2. Approximate optimality follows from Lemmas S3 and 9. □

Lemma S5. *When all treatment-specific constraints are less or equal to one and the overall size constraint is the sum of the treatment-specific constraints, the distance between any two vertices connected by an arc in* $G_C = (\mathbf{V}, \mathbf{E}_C)$ *is less or equal to the maximum within-group distance in an optimal matching with* L_{WBN} *as objective:*

$$c_1, c_2, \cdots, c_k \leq 1 \wedge t = \textstyle\sum_{x=1}^k c_x \implies \forall (i,j) \in \mathbf{E}_C, d(i,j) \leq \min_{\mathbf{M} \in \mathcal{M}_C} L_{\text{WBN}}(\mathbf{M}).$$

Proof. Let w_s^+ be the maximum weight of an arc connecting two units with the same treatment conditions in G_C, and let w_d^+ the maximum arc weight between units with different conditions:

$$w_s^+ = \max\{d(i,j) : (i,j) \in \mathbf{E}_C \wedge W_i = W_j\},$$
$$w_d^+ = \max\{d(i,j) : (i,j) \in \mathbf{E}_C \wedge W_i \neq W_j\}.$$

Note that

$$\max\{d(i,j) : (i,j) \in \mathbf{E}_C\} = \max\{w_s^+, w_d^+\}.$$

First, consider w_s^+. Since $c_1, c_2, \cdots, c_k \leq 1$ and $t = \sum_{x=1}^k c_x$, each unit will have at most one arc pointing to a unit with the same treatment condition as its own:

$$\forall i,\ |\{(i,j) : (i,j) \in \mathbf{E}_C \wedge W_i = W_j\}| = c_{W_i} \leq 1.$$

From the self-similarity and non-negativity properties of distance metrics, we have:

$$\forall i,j,\ 0 = d(i,i) \leq d(i,j).$$

By construction of G_C, all arcs in the set will be self-loops and, thus, at distance zero:

$$w_s^+ = \max\{d(i,i) : (i,i) \in \mathbf{E}_C\} = 0.$$

From non-negativity, it follows that:

$$\max\{d(i,j) : (i,j) \in \mathbf{E}_C\} = \max\{0, w_d^+\} = w_d^+.$$

Let w^* be the maximum within-group distance between units assigned to different treatment conditions when L_{WBN} is used as objective:

$$w^* = \min_{\mathbf{M} \in \mathcal{M}_C} L_{\text{WBN}}(\mathbf{M}).$$

Let $B_d = (\mathbf{U}, \mathbf{E}_d^b)$ be the digraph that contains all arcs between units that either are strictly closer than w_d^+ or have the same treatment condition:

$$\mathbf{E}_d^b = \{(i,j) : d(i,j) < w_d^+ \vee W_i = W_j\}.$$

Following the same logic as in the proof of Lemma 9, B_d must contain a vertex whose neighborhood is not compatible with C.

Let $B_{\text{OP}} = (\mathbf{U}, \mathbf{E}^b_{\text{OP}})$ be the digraph that contains all arcs between units that either are weakly closer than w^* or have the same treatment condition:

$$\mathbf{E}^b_{\text{OP}} = \{(i,j) : d(i,j) \leq w^* \vee W_i = W_j\}.$$

By construction, B_{OP} is a supergraph of the cluster graph induced by the optimal matching. That is, arcs are drawn in B_{OP} between all units assigned to the same matched group in the optimal matching. As the optimal matching is admissible, each vertex's neighborhood in B_{OP} is compatible with C.

Assume $w^+_d > w^*$. It follows that $\mathbf{E}^b_{\text{OP}} \subset \mathbf{E}^b_d$. As at least one vertex's neighborhood does not satisfy the size constraint in B_d, that must be the case in B_{OP}. This, however, implies that the optimal matching is not admissible which, in turn, contradicts optimality. We conclude that $w^+_d \leq w^*$. □

Theorem 11. $\mathbf{M}_{\text{ALG}}$ *is a 4-approximate conventional full matching with respect to the matching constraint* $C = (1, \cdots, 1, k)$ *and matching objective* L_{WBN}*:*

$$\mathbf{M}_{\text{ALG}} \in \mathcal{M}_C \qquad \textit{and} \qquad L_{\text{WBN}}(\mathbf{M}_{\text{ALG}}) \leq \min_{\mathbf{M} \in \mathcal{M}_C} 4L_{\text{WBN}}(\mathbf{M}).$$

Proof. Admissibility follows from Lemma S2. Note that all distances considered by L_{WBN} are considered by L_{BN} as well. As a result, the latter acts as a bound for the former:

$$\forall \mathbf{M} \in \mathcal{M}_C, L_{\text{WBN}}(\mathbf{M}) \leq L_{\text{BN}}(\mathbf{M}).$$

Approximate optimality follows from Lemma S3 and S5:

$$L_{\text{WBN}}(\mathbf{M}_{\text{ALG}}) \leq L_{\text{BN}}(\mathbf{M}_{\text{ALG}}) \leq 4 \min_{\mathbf{M} \in \mathcal{M}_C} L_{\text{WBN}}(\mathbf{M}).$$ □

S2.2 Complexity

Lemma S6. *A* C*-compatible nearest neighbor digraph can be constructed in polynomial time using linear memory.*

Proof. In the first step of the algorithm, we construct G_w as the union of $\text{NN}(c_x, G(\mathbf{U} \to \mathbf{w}_x))$ for each treatment condition x. The operands of this union can be constructed using nearest neighbor searches for each treatment condition. With a naive implementation, such searches can be done sequentially for each $i \in \mathbf{U}$ by sorting the set $\{d(i,j) : j \in \mathbf{w}_x\}$ and drawing an arc from i to the first c_x elements in the sorted set. When using standard sorting algorithms, this has a time complexity of $O(n|\mathbf{w}_x| \log |\mathbf{w}_x|)$ and a space complexity of $O(c_x n)$ (Knuth 1998). Note that $|\mathbf{w}_x| \leq n$ for all treatments, so the search requires $O(n^2 \log n)$ time. The union operation can be performed in linear time in the total number of arcs, $O[(c_1 + c_2 + \cdots + c_k)n]$. As each $\text{NN}(c_x, G(\mathbf{U} \to \mathbf{w}_x))$ can be derived sequentially and the size constraints are fixed, the G_w digraph can be constructed in $O(n^2 \log n)$ time.

In the second step, G_r can be constructed in a similar fashion. For each $i \in \mathbf{U}$, sort the set $\{d(i,j) : j \in \mathbf{U} \wedge (i,j) \notin \mathbf{E}_w\}$ and draw an arc from i to the first $r = t - c_1 - \cdots - c_k$ elements in that set. Like above, this has a complexity of $O(n^2 \log n)$. Finally, the union between G_w and G_r can be constructed in linear time in the total number of arcs. As the number of arcs per vertex is fixed at t, the union is completed in $O(n)$ time. The steps are sequential so the total complexity of both Step 1 and 2 is $O(n^2 \log n)$. □

Remark S7. *For most common metrics, standard sorting algorithms are inefficient. Storing the data points in a structure made for the purpose, such as a kd- or bd-tree, typically leads to large improvements (Friedman, Bentley, and Finkel 1977). Each* $\text{NN}(c_x, G(\mathbf{U} \to \mathbf{w}_x))$ *can then be constructed in* $O(n \log n)$ *average time, without changing the memory complexity. However, this approach typically requires a preprocessing step to build the search tree. In*

the proof of Lemma S6, the search set is unique for each vertex when G_r is constructed. We can, therefore, not use these specialized algorithms if we construct G_r in the way suggested there. However, the construction can easily be transformed into a problem with a fixed search set. Note that:

$$\text{NN}(r, G(\mathbf{U} \to \mathbf{U}) - G_w) = \text{NN}(r, \text{NN}(t, G(\mathbf{U} \to \mathbf{U})) - G_w).$$

That is, finding the r nearest neighbors not already connected in G_w is the same as finding the r nearest neighbors not already connected in G_w among the t nearest neighbors in the complete graph. The first nearest neighbor search, $\text{NN}(t, G(\mathbf{U} \to \mathbf{U}))$*, has a fixed search set and can thus be completed in $O(n \log n)$. The second nearest neighbor search involves sorting at most t elements for each vertex, which is done in constant time as t is fixed.*

Theorem 12. *In the worst case, the generalized full matching algorithm terminates in polynomial time using linear memory.*

Proof. The algorithm runs sequentially. The first and second steps can be completed in $O(n^2 \log n)$ worst-case time as shown in Lemma S6, or, in many cases, in $O(n \log n)$ average time as discussed in Remark S7.

Steps 3 and 4 can be done by sequentially labeling seeds and their neighbors as they are selected. Any vertex whose neighborhood does not contain any labeled vertices can be a valid seed, and any vertex that is adjacent to labeled vertices can never become a seed. Thus, traversing the vertices in any order and greedily selecting units as seed will yield a valid set of seeds. As the size of each seed's neighborhood is fixed at t, this step is completed in $O(n)$ time.

Finally, assigning labels to unlabeled vertices in the last step can be done by traversing over their neighborhoods. Thus, Step 5 also requires $O(n)$ time to complete. □

S3 Additional simulation results

The following tables present additional results from the simulation study. Sections S3.1 and S3.2 provide results about aggregated distances for the algorithms and the structure of the matched groups they produce. Section S3.3 gives complete results for the measures presented in the paper. These tables also include the results for 1:2-matching without replacement.

S3.1 Distances

We investigate five different functions aggregating within-group distances:

$$\begin{aligned}
L_{\text{BN}}(\mathbf{M}) &= \max_{\mathbf{m} \in \mathbf{M}} \max\{d(i,j) : i,j \in \mathbf{m}\},\\
L_{\text{WBN}}(\mathbf{M}) &= \max_{\mathbf{m} \in \mathbf{M}} \max\{d(i,j) : i,j \in \mathbf{m} \wedge W_i \neq W_j\},\\
L_{\text{MEAN}}(\mathbf{M}) &= \sum_{\mathbf{m} \in \mathbf{M}} \frac{|\mathbf{w}_1 \cap \mathbf{m}|}{|\mathbf{w}_1|} \text{mean}\{d(i,j) : i,j \in \mathbf{m} \wedge i \neq j\},\\
L_{\text{WMEAN}}(\mathbf{M}) &= \sum_{\mathbf{m} \in \mathbf{M}} \frac{|\mathbf{w}_1 \cap \mathbf{m}|}{|\mathbf{w}_1|} \text{mean}\{d(i,j) : i,j \in \mathbf{m} \wedge W_i \neq W_j\},\\
L_{\text{WSUM}}(\mathbf{M}) &= \sum_{\mathbf{m} \in \mathbf{M}} \sum \{d(i,j) : i,j \in \mathbf{m} \wedge W_i \neq W_j\}.
\end{aligned}$$

L_{BN} is the maximum within-group distance between any two units, and L_{WBN} is the maximum distance between treated and control units. They are the objectives discussed in Section 4.1 in the main paper and are the ones used by the `quickmatch` package. L_{WMEAN} is the average within-group distance between treated and control units weighted by the number treated units in the groups. It is the objective function discussed by Rosenbaum (1991) when he introduced full matching. As Rosenbaum notes, this objective is neutral in the sense that the size of the matched groups matters only insofar as it affects the within-group distances. To contrast with L_{BN}, we include

L_{MEAN}, which is a version of the mean distance objective that also considers within-group distances between units assigned to the same treatment condition.

Finally, L_{WSUM} is the sum of within-group distances between treated and control units. With the terminology of Rosenbaum (1991), this function favors small subclasses and is, thus, not neutral. As a consequence, if we were to use L_{WSUM} as our objective, we would accept matchings with worse balance if the matched groups were sufficiently smaller. When the matching structure is fixed (as with 1:1- and 1:k-matching without replacement), L_{WSUM} is proportional to L_{WMEAN} and, thus, identical for practical purposes. Both the `optmatch` and `Matching` packages use the sum as their objective.

Table S1 presents the distance measures for the different methods. As distances have no natural scale, we normalize the results by the results of conventional full matching in smaller sample. We see that 1:1-matching with replacement greatly outperforms the other methods, especially on L_{WSUM} which is the objective function it uses. The implementations of both conventional and generalized full matching perform largely the same, with a slight advantage to `optmatch` on the L_{WSUM} measure. All versions of matching without replacement performs considerably worse than the other methods, in particular on the measures they do not use as their objective. The optimal implementations produce shorter distances than the greedy versions, but the differences are small.

Comparisons in aggregated distances between methods that impose different matching constraints can be awkward because the methods solve different types of matching problems. For example, 1:2-matching will necessarily lead to larger distances than 1:1-matching, but the former can be preferable if, for example, we are interested in ATT and control units vastly outnumbers treated units. Comparisons between methods using the same matching constraints should, however, be informative.

Table S1. Aggregated distances for matching methods with samples of 1,000 and 10,000 units.

	1,000 units					10,000 units				
	L_{BN}	L_{WBN}	L_{MEAN}	L_{WMEAN}	L_{WSUM}	L_{BN}	L_{WBN}	L_{MEAN}	L_{WMEAN}	L_{WSUM}
Greedy 1:1	1.87	2.67	1.41	1.50	0.43	2.20	3.14	0.89	0.95	2.69
Optimal 1:1	1.29	1.85	1.20	1.27	0.36	1.87	2.68	0.80	0.85	2.41
Replacement 1:1	0.45	0.51	0.65	0.66	0.19	0.19	0.20	0.20	0.21	0.59
Greedy 1:2	3.66	5.23	3.21	4.31	2.46	3.99	5.71	2.51	3.69	20.97
Optimal 1:2	3.27	4.68	3.17	3.79	2.17	3.93	5.62	2.96	3.50	19.87
Full matching	1.00	1.00	1.00	1.00	1.00	0.39	0.38	0.31	0.31	3.10
GFM	1.00	1.00	0.99	0.98	1.05	0.39	0.38	0.31	0.30	3.25
Refined GFM	0.95	1.25	0.98	1.10	1.19	0.37	0.49	0.31	0.34	3.70

Notes: The measures are normalized by the result for conventional full matching in the sample with 1,000 units. Results are based on 10,000 simulation rounds. Simulation errors are negligible.

S3.2 Group structure

Table S2 presents measures of the group structure for the different matching methods. The first measure is the average size of the matched groups. 1:1- and 1:2-matching without replacement have a fixed group size of either two or three units. The group size for matching with replacement depends on the sparseness of the control units. Overlap is reasonably good with the current data generating process, and the average group size increases with only 20% compared to matching without replacement. The full matching methods lead to larger groups since they do not discard units. Given the unconditional propensity score of 26.5%, the expected minimum group size among matchings that do not discard units is 3.77 units, which is close to what the methods produce. The groups are slightly smaller with conventional full matching. This is likely a result of both that implementation's optimality and its use of a non-neutral objective function (i.e., L_{WSUM}). In the second column, we present the standard deviation of the group sizes. We see that the full matching methods have considerably higher variability. This is a consequence of their ability to adapt the matching to the distribution of units in the covariate space.

Next, we investigate the share of the sample that is discarded. For a given level of balance, we want to drop as few units as possible. Predictably, 1:1-matching leads to that a sizable portion of the sample is left unassigned. This is especially the case when we match with replacement. Fewer units are discarded with 1:2-matching, and by construction, no units are discarded with the full matching methods.

The fourth column reports the standard deviation of the weights implicitly used for the adjustment in the estimator. Weight variation is necessary to balance an unbalanced sample. However, for a given level of balance, we want the weights to be as uniform as possible. Since we are estimating ATT, the implied weights for treated units are fixed at $|\mathbf{w}_1|^{-1}$ for all methods. Weights for controls do, however, vary. The implied weight for control unit i assigned to matched group $\mathbf{m}$ is

$$\text{wgh}_i = \frac{|\mathbf{w}_1 \cap \mathbf{m}|}{|\mathbf{w}_1| \times |\mathbf{w}_0 \cap \mathbf{m}|},$$

and zero if not assigned to a group.

Examining the results, we see that the amount of variation is correlated with how well the methods are able to minimize distances. For example, 1:1-matching with replacement produces the shortest distances, but as a result, also the most weight variation. The choice of method depends on how one resolves the trade-off between weight variation and balance, which, in turn, depends on how strongly the covariates are correlated with the outcome and treatment assignment. For this reason, the best choice of matching method will differ depending on the data generating process. It appears, however, that all full matching methods lead to matchings with substantially smaller distances than 1:1-matching without replacement with only slightly higher weight variation (i.e., close to a Pareto improvement). Similarly, but less pronounced, the `optmatch` package dominates the `quickmatch` package; the former produces about the same distances but with less weight variation.

Table S2. Group composition for matching methods with samples of 1,000 and 10,000 units.

	1,000 units				10,000 units			
	Size	σ(Size)	% drop	σ(wgh)	Size	σ(Size)	% drop	σ(wgh)
Greedy 1:1	2.00	0.00	46.96	1.81	2.00	0.00	47.03	1.81
Optimal 1:1	2.00	0.00	46.96	1.81	2.00	0.00	47.03	1.81
Replacement 1:1	2.41	0.86	54.70	2.85	2.41	0.87	54.73	2.85
Greedy 1:2	3.00	0.00	20.44	0.84	3.00	0.00	20.54	0.85
Optimal 1:2	3.00	0.00	20.44	0.84	3.00	0.00	20.54	0.85
Full matching	4.24	3.50	0.00	1.93	4.24	3.51	0.00	1.97
GFM	4.74	3.54	0.00	2.13	4.73	3.55	0.00	2.15
Refined GFM	4.55	3.26	0.00	2.04	4.54	3.30	0.00	2.07

Notes: The columns report the average group size, the standard deviation of the size, share of units not assigned to a group and the standard deviation in the weights of the control units implied by the matchings. Results are based on 10,000 simulation rounds. Simulation errors are negligible.

S3.3 Complexity and matching quality

.

Table S3. Covariate balance for matching methods with samples of 1,000 and 10,000 units.

	1,000 units					10,000 units				
	X_1	X_2	X_1^2	X_2^2	X_1X_2	X_1	X_2	X_1^2	X_2^2	X_1X_2
Unadjusted	52.48	52.73	10.72	10.91	13.11	52.528	52.735	10.707	10.874	12.588
Greedy 1:1	5.93	5.94	7.21	7.31	13.87	5.270	5.286	6.647	6.756	13.332
Optimal 1:1	5.94	5.95	7.08	7.19	14.09	5.260	5.279	6.594	6.704	13.396
Replacement 1:1	0.44	0.44	0.76	0.76	0.80	0.043	0.043	0.077	0.079	0.077
Greedy 1:2	26.23	26.39	15.48	15.76	35.59	25.662	25.741	15.410	15.667	36.914
Optimal 1:2	26.18	26.34	15.22	15.48	36.11	25.668	25.759	15.495	15.749	36.745
Full matching	1.00	1.00	1.00	1.00	1.00	0.105	0.105	0.106	0.108	0.096
GFM	0.74	0.75	0.77	0.77	0.80	0.074	0.075	0.075	0.077	0.073
Refined GFM	1.04	1.05	0.99	0.99	1.08	0.108	0.108	0.103	0.104	0.105

Notes: The measures are normalized by the result for conventional full matching in the sample with 1,000 units.

Table S4. Estimator performance for matching methods with samples of 1,000 and 10,000 units.

	1,000 units				10,000 units			
	Bias	SE	RMSE	$\frac{\text{Bias}}{\text{RMSE}}$	Bias	SE	RMSE	$\frac{\text{Bias}}{\text{RMSE}}$
Unadjusted	83.34	1.47	12.70	0.993	83.390	0.47	12.64	0.999
Greedy 1:1	4.86	1.04	1.26	0.583	4.118	0.33	0.71	0.884
Optimal 1:1	4.96	1.04	1.27	0.590	4.153	0.33	0.71	0.885
Replacement 1:1	0.11	1.17	1.16	0.015	0.024	0.38	0.37	0.010
Greedy 1:2	33.97	1.61	5.38	0.956	32.980	0.52	5.02	0.995
Optimal 1:2	34.08	1.59	5.39	0.957	32.939	0.51	5.01	0.995
Full matching	1.00	1.00	1.00	0.151	0.077	0.32	0.32	0.037
GFM	0.77	1.03	1.03	0.113	0.043	0.33	0.33	0.020
Refined GFM	1.20	1.02	1.02	0.178	0.091	0.32	0.32	0.043

Notes: The first three measures in each panel are normalized by the result for conventional full matching.

Table S5. Runtime and memory use by sample size for matching methods.

Panel A: Runtime (in minutes)

	100	500	1K	5K	10K	20K	50K	100K	200K	500K	1M	5M	10M	50M	100M
Greedy 1:1	0.01	0.01	0.01	0.01	0.03	0.09	0.62	2.71	12.85						
Optimal 1:1	0.06	0.06	0.08	1.08	5.03	19.40									
Replacement 1:1	0.01	0.01	0.01	0.01	0.03	0.09	0.58	2.60	12.15						
Greedy 1:2	0.01	0.01	0.01	0.01	0.03	0.09	0.63	2.76	13.20						
Optimal 1:2	0.06	0.06	0.10	5.02	26.19										
Full matching	0.06	0.06	0.08	0.30	0.87	2.84									
GFM	0.00	0.00	0.00	0.00	0.01	0.00	0.01	0.01	0.02	0.06	0.11	0.64	1.05	6.49	14.15
Refined GFM	0.00	0.00	0.00	0.01	0.01	0.01	0.01	0.02	0.03	0.08	0.16	0.96	1.52	9.31	20.07

Panel A: Memory use (in gigabytes)

	100	500	1K	5K	10K	20K	50K	100K	200K	500K	1M	5M	10M	50M	100M
Greedy 1:1	0.03	0.03	0.03	0.04	0.04	0.04	0.05	0.06	0.09						
Optimal 1:1	0.12	0.13	0.15	0.74	2.75	10.21									
Replacement 1:1	0.03	0.03	0.03	0.03	0.04	0.04	0.05	0.06	0.10						
Greedy 1:2	0.03	0.03	0.03	0.04	0.04	0.04	0.05	0.07	0.11						
Optimal 1:2	0.12	0.13	0.15	0.74	2.74										
Full matching	0.12	0.13	0.16	0.74	2.74	10.19									
GFM	0.03	0.03	0.03	0.03	0.03	0.03	0.04	0.05	0.06	0.12	0.21	0.88	1.73	8.57	17.11
Refined GFM	0.03	0.03	0.03	0.03	0.03	0.03	0.04	0.05	0.06	0.12	0.21	0.88	1.73	8.57	17.11

Notes: Each cell presents the runtime and memory use for different matching implementations for different sample sizes. The first panel shows runtime in minutes, and the second panel shows memory use in gigabytes. Each column represents a different sample size where "K" denotes thousand and "M" denotes million. The rows indicate matching method. Blank cells indicate that the corresponding matching method did not terminate successfully for the corresponding sample size within reasonable time and memory limits. Each measure is based on 1,000 simulation rounds.

S4 Additional extrapolation results

The following tables provide unadjusted and adjusted covariate averages for all treatment conditions in the full population and in the subpopulation of voters in the 2004 general election.

Table S6. Covariate balance before and after matching adjustment in full QVF population.

Panel A: Covariate balance before matching

	Control	Civic Duty	Hawthorne	Self	Neighbors	Non-experiment
Birth year	1956.19	1956.34	1956.30	1956.21	1956.15	1957.96
Female (%)	49.89	50.02	49.90	49.96	50.00	53.32
Voted Aug 2000 (%)	25.19	25.36	25.04	25.11	25.12	14.65
Voted Aug 2002 (%)	38.94	38.88	39.43	39.19	38.66	22.59
Voted Aug 2004 (%)	40.03	39.94	40.32	40.25	40.67	18.71
Voted Nov 2000 (%)	84.34	84.17	84.44	84.04	84.17	52.49
Voted Nov 2002 (%)	81.09	81.11	81.30	81.15	81.13	41.93
Voted Nov 2004 (%)	100.00	100.00	100.00	100.00	100.00	67.57

Panel B: Covariate balance after matching

	Control	Civic Duty	Hawthorne	Self	Neighbors	Non-experiment
Birth year	1958.16	1958.49	1958.44	1958.57	1958.51	1957.87
Female (%)	53.29	53.28	53.28	53.29	53.28	53.15
Voted Aug 2000 (%)	15.19	15.19	15.19	15.19	15.19	15.19
Voted Aug 2002 (%)	23.42	23.42	23.42	23.42	23.42	23.43
Voted Aug 2004 (%)	19.80	19.80	19.80	19.80	19.80	19.80
Voted Nov 2000 (%)	54.11	54.14	54.13	54.13	54.13	54.11
Voted Nov 2002 (%)	43.94	43.94	43.94	43.94	43.94	43.92
Voted Nov 2004 (%)	100.00	100.00	100.00	100.00	100.00	68.76

Table S7. Covariate balance before and after adjustment among voters in 2004 general election.

Panel A: Covariate balance before matching

	Control	Civic Duty	Hawthorne	Self	Neighbors	Non-experiment
Birth year	1956.19	1956.34	1956.30	1956.21	1956.15	1955.71
Female (%)	49.89	50.02	49.90	49.96	50.00	54.51
Voted Aug 2000 (%)	25.19	25.36	25.04	25.11	25.12	20.49
Voted Aug 2002 (%)	38.94	38.88	39.43	39.19	38.66	31.94
Voted Aug 2004 (%)	40.03	39.94	40.32	40.25	40.67	26.94
Voted Nov 2000 (%)	84.34	84.17	84.44	84.04	84.17	70.16
Voted Nov 2002 (%)	81.09	81.11	81.30	81.15	81.13	58.81
Voted Nov 2004 (%)	100.00	100.00	100.00	100.00	100.00	100.00

Panel B: Covariate balance after matching

	Control	Civic Duty	Hawthorne	Self	Neighbors	Non-experiment
Birth year	1955.90	1956.11	1956.25	1956.26	1956.20	1955.74
Female (%)	54.17	54.16	54.17	54.17	54.16	54.17
Voted Aug 2000 (%)	20.83	20.83	20.81	20.83	20.83	20.83
Voted Aug 2002 (%)	32.46	32.46	32.46	32.46	32.46	32.46
Voted Aug 2004 (%)	27.92	27.92	27.92	27.92	27.92	27.92
Voted Nov 2000 (%)	71.20	71.22	71.21	71.20	71.21	71.20
Voted Nov 2002 (%)	60.44	60.45	60.45	60.45	60.45	60.45
Voted Nov 2004 (%)	100.00	100.00	100.00	100.00	100.00	100.00

References

Friedman, J. H., J. L. Bentley, and R. A. Finkel. 1977. “An algorithm for finding best matches in logarithmic expected time.” *ACM Transactions on Mathematical Software* 3 (3): 209–226.

Knuth, D. E. 1998. *Sorting and searching.* 2th. Vol. 3. The Art of Computer Programming. Redwood City: Addison Wesley Longman.

Rosenbaum, P. R. 1991. “A characterization of optimal designs for observational studies.” *Journal of the Royal Statistical Society. Series B (Methodological)* 53 (3): 597–610.