A Feasible Estimator for Linear Models with Multi-Way Fixed Effects*

Sergio Correia
Duke University

March 2016

Preliminary Version

Abstract

I propose a feasible and computationally efficient estimator of linear models with multiple levels of fixed effects. First, I show that solving the two–way fixed effects model is equivalent to solving a linear system on a weighted graph, and apply recent advances in spectral graph theory to obtain a nearly–linear time estimator (Kelner et al, 2013). Second, I embed this estimator into an improved version of the generalized within–estimator of Guimarães and Portugal (2010) and Gaure (2013), replacing their projections with symmetric ones amenable to conjugate gradient acceleration, guaranteeing monotonic convergence. The proposed estimator has the fastest known asymptotic running time, and performs particularly well with large datasets and high–dimensional fixed effects. Morever, by combining insights from graph theory, it opens the door to further improvements on the estimation and inference of models with multi–way fixed effects.

Keywords: fixed effects, panel data, high-dimensional fixed effects, alternating projections, graph laplacian

JEL Classification: C01, C13, C23, C55, C81

*I wish to thank Paulo Guimarães for his immense feedback, as well as Manuel Adelino, Sharon Belenzon, Kevin Deweese, Matthieu Gomez, Amine Ouazad, Marcos Raydan and Mark E. Schaffer for their helpful suggestions. All errors are my own.

1 Introduction

The emergence of large–scale administrative and private sector datasets has made linear models with large sets of fixed effects commonplace in applied economic research (Einav and Levin 2014). The reason is twofold. First, the scale of these datasets allows for flexible model parameterizations, where—for instance—age covariates are replaced with age fixed effects. Second, these datasets are often structured as panels where observations correspond to multiple economic units simultaneously. Examples include employers and employees (Abowd, Kramarz, and Margolis 1999), employers, employees and job titles (Carneiro, Guimarães, and Portugal 2012), students and teachers (Rockoff 2004), schools, grades and subjects (Chetty, Friedman, and Rockoff 2014) CEOs and firms (Bertrand and Schoar 2003), exporters and importers (Head and Mayer 2014), and so on. In these settings, including multiple levels of fixed effects allows researchers to control for unobserved heterogeneity specific to each individual or group, which could otherwise preclude causal inference due to omitted variable biases (Gormley and Matsa 2014).

The traditional approach to estimate these models—apply the within transformation with respect to the fixed effect with more categories and to add one dummy variable for each category of all subsequent fixed effects (Wooldridge 2010)—is unfeasible with large datasets or if there is more than one set of fixed effects with many categories. For instance, the dataset employed by Carneiro, Guimarães, and Portugal (2012) comprises a total of 31.6 million observations, with 6.4 million individuals, 624 thousand firms, and 115 thousand occupations. Just storing the required indicator matrices would require 23.4 terabytes of memory, 91 times the total memory of the largest NBER computer server available (as of 2015). Alternative approaches either only work in very specific setups, such as strongly balanced panels (Baltagi 2008), or—as described by Gormley and Matsa (2014)—are inconsistent.

To address this limitation, there have been multiple efforts in recent years to provide a feasible and computationally efficient estimator that allows for multiple levels of fixed effects. The first such estimator, Abowd, Creecy, and Kramarz (2002), used the conjugate gradient method with a diagonal preconditioner to construct a practical solver for two levels of fixed effects. However, its convergence can be quite poor (Koutis, Miller, and Peng 2012) if the graph that underlies the fixed effects is poorly connected.[1] Subsequently, Guimarães and Portugal (2010) and Gaure (2013a) constructed an elegant estimator by combining the method of alternating projections with the Frisch–Waugh–Lovell theorem. This estimator allows for more than two levels of fixed effects, but suffers from slow convergence rates (Gaure 2015). In particular, convergence rates can be shown to be *arbitrarily slow* (Bauschke et al. 2003), meaning that numer-

[1] In particular, conjugate Gradient has an asymptotic runtime of $\mathcal{O}(\kappa \log \epsilon^{-1})$ where κ is the relative condition number (the ratio of the largest to the smallest eigenvalue) of the full-rank version of the matrix. See Golub and Van Loan (2013) and Spielman (2010) for a more detailed discussion of the topic.

ical convergence can take an arbitrarily large number of iterations, a particularly troublesome fact given the large datasets that are more likely to require these methods. As a solution, these methods apply acceleration techniques that often speed up the results significantly, but that depending on the problem can make convergence even slower (Hernández-Ramos, Escalante, and Raydan 2011). Other estimators are also of limited applicability: they are either limited to only one high–dimensional fixed effect (Cornelissen 2008; Somaini and Wolak 2015; Mittag 2015), or arrive at consistent but potentially very inefficient estimators (the spell fixed effects discussed Andrews, Schank, and Upward 2006).[2]

This paper has two central contributions. First, it shows how the solution of the two–way fixed effects model is equivalent to that of a linear system on a graph Laplacian matrix. By doing so, it allows the application of a new class of combinatorial algorithms that have an unprecedented nearly–linear running time. Moreover, it leverages the link with graph theory to apply additional techniques such as simplifications onto a 3–core graph. Second, this paper solves the main shortcomings of the multi–way fixed effects estimator of Guimarães and Portugal (2010) and Gaure (2013a), by replacing their projections with symmetric ones, which are then combined with a conjugate gradient acceleration.

Achieving a computationally efficient estimator is important for reasons well beyond OLS:

1. Thanks to the Frisch–Waugh–Lovell theorem, this estimator can be trivially extended to other linear models such as two–stage least squares, limited-information maximum likelihood, and two–step linear GMM.
2. It can be used as a building block for nonlinear models. Existing implementations include the two–way fixed effects Poisson regression of Guimaraes (2014), interactive fixed effects (Bai 2009), iterated and continuously updated GMM, and spillover models (Arcidiacono et al. 2012).
3. Inference with bootstrapping and jacknife resampling can be vastly speed up, making these approaches feasible with many fixed effects (see however the warning of Cattaneo, Jansson, and Newey 2015).

Therefore, the advances proposed in this paper have a wide range of benefits across many fields and methods.

[2] On the other hand, two promising alternatives are multi–grid methods (see Golub and Van Loan 2013 for an introduction) and the LSMR solver of Fong and Saunders (2011), applied to the fixed effects problem by Gomez (2016). Although more empirical studies are required to assess their performance, both can be improved by combining them with techniques discussed in this paper, such as focusing on the two–core problem or applying the RCM algorithm.

2 Setup

We are interested in $\hat{\boldsymbol{\beta}}$, the OLS estimator of $\boldsymbol{\beta}$ in a model of the form

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \mathbf{D}\boldsymbol{\alpha} + \boldsymbol{\varepsilon}$$

where $\mathbf{y}$ is an outcome vector of length n, $\mathbf{X}$ is an $n \times k$ matrix of covariates, $\mathbf{D}$ is an $n \times g$ matrix of dummy variables, and $\boldsymbol{\varepsilon}$ is an unobserved error term. The dummy matrix $\mathbf{D}$ represents fixed effects across F dimensions, so it has a block representation $\mathbf{D} = \begin{bmatrix} \mathbf{D}_1 & \mathbf{D}_2 & \cdots & \mathbf{D}_F \end{bmatrix}$. The number of levels (categories, groups) for the f-th dimension is g_f, so $g = \sum_{f=1}^{F} g_f$. If a fixed effects has many categories that increase with the sample size ($g_f \propto n$), it is a "highly–dimensional fixed effect" (Guimarães and Portugal 2010).

For instance, in a matched employer–employee dataset, a model could include individual, firm, and time fixed effects ($f = 3$). If there are 100,000 individuals, 50,000 firms and 10 years, then the total number of categories g would be 150,010, with the individual and time fixed effects being highly dimensional.

To obtain $\hat{\boldsymbol{\beta}}$, I exploit the insight of Guimarães and Portugal (2010) and apply the Frisch–Waugh–Lovell theorem (Frisch and Waugh 1933; Lovell 1963). This theorem implies that the least squares estimates $\hat{\boldsymbol{\beta}}$ can be recovered by first regressing each variable against all the fixed effects, and then regressing the residuals of these variables. It thus allows us to *divide and conquer* the larger problem by focusing on smaller systems.

More formally, let $\mathbf{P_D} = \mathbf{D}(\mathbf{D}'\mathbf{D})^{-1}\mathbf{D}'$ be the projection matrix with respect to $\mathbf{D}$ and $\mathbf{M_D} = \mathbf{I} - \mathbf{P_D}$ the corresponding annihilator or residual–maker matrix. The partialled–out vectors $\tilde{\mathbf{y}} = \mathbf{M_D}\mathbf{y}$ and $\tilde{\mathbf{X}} = \mathbf{M_D}\mathbf{X}$ are the residuals of $\mathbf{y}$ and $\mathbf{X}$ with respect to the fixed effects. Then the two–part theorem states that[3]

1. $\hat{\boldsymbol{\beta}} = (\tilde{\mathbf{X}}'\tilde{\mathbf{X}})^{-1}\tilde{\mathbf{X}}'\tilde{\mathbf{y}}$
2. $\hat{\boldsymbol{\varepsilon}} \equiv \mathbf{y} - \mathbf{X}\hat{\boldsymbol{\beta}} + \mathbf{D}\hat{\boldsymbol{\alpha}} = \tilde{\mathbf{y}} - \tilde{\mathbf{X}}\hat{\boldsymbol{\beta}}$

Note that with $f = 1$ this reduces to the textbook within–transformation, and the operator $\mathbf{M_D}$ will just subtract group means. Also notice that this approach can be extended beyond OLS, to any setup where versions of FWL exist, such as instrumental variables and linear GMM.

Now, the remaining step is to obtain the OLS residuals of a model of the form $\mathbf{y} = \mathbf{D}\boldsymbol{\alpha} + \boldsymbol{\varepsilon}$ for $k + 1$ variables. In terms of normal equations, it is equivalent[4] to solving the linear system

[3] **Proof:** First, note that $\mathbf{M_D}$ is idempotent, $\mathbf{M_D}\mathbf{D} = \mathbf{0}$, $\mathbf{M_D}\hat{\boldsymbol{\varepsilon}} = \hat{\boldsymbol{\varepsilon}}$, and from the normal equations, $\mathbf{X}'\hat{\boldsymbol{\varepsilon}} = \mathbf{0}$. Premultiply $\mathbf{y} = \mathbf{X}\hat{\boldsymbol{\beta}} + \mathbf{D}\hat{\boldsymbol{\alpha}} + \hat{\boldsymbol{\varepsilon}}$ by $\mathbf{X}'\mathbf{M_D}$. Then, replace and invert ■. For the second part, premultiply $\hat{\boldsymbol{\varepsilon}}$ by $\mathbf{M_D}$ and replace ■.

[4] For $f > 1$, $\mathbf{D}'\mathbf{D}$ is not full rank, so the equivalence actually requires some convention on how full rank is achieved. For instance, the last category of each set of fixed effects can be dropped, or the mean of each set of fixed effects can be normalized to zero. Nevertheless, $\hat{\boldsymbol{\alpha}}$ always leads to the same residuals independently on the normalization chosen.

$$(\mathbf{D}'\mathbf{D})\hat{\boldsymbol{\alpha}} = (\mathbf{D}'\mathbf{y}) \tag{1}$$

3 Two-Way Fixed Effects and Graph Laplacians

With two levels of fixed effects, the matrix $D'D = [w_{ij}]$ studied in equation (1) then belongs to the class of symmetric diagonally dominant matrices (SDD), as it is both symmetric and its diagonal elements dominate the sum of its off–diagonal ones ($w_{ii} \geq \sum_{j \neq i} |w_{ij}| \ \forall i$)[5].

This class of matrices is interesting because it can be reduced to Laplacian matrices of the form $\mathbf{Lx} = \mathbf{b}$ through standard reduction techniques.[6] In turn, Laplacians systems have recently acquired prominence as they can be solved in nearly–linear time, in contrast with previous algorithms, such as direct solvers that only run in $\mathcal{O}(n^2.3727)$ time. Thanks to this breakthrough, researchers have been able to improve multiple fundamental algorithms in graph theory and numerical optimizations (see Spielman 2010; Teng 2010 for a review of its applications).

Nearly–linear time Laplacian solvers were first proposed by Spielman and Teng (2004), who built upon the insight of Vaidya (1991) that suggesting that good preconditioners of Laplacian systems could be the certain Laplacians corresponding to subgraphs of the original system. The importance of the Spielman and Teng work cannot be understated, as in the words of Kelner et al. (2013a), it was a "technical tour–de–force that required multiple innovations in spectral and combinatorial graph theory, graph algorithms, and computational linear algebra" and included "the invention of spectral sparsification and ultra–sparsifiers, better and faster constructions of low–stretch spanning trees, and efficient local clustering algorithms". This work was later divided into three three papers (Spielman and Teng 2011; Spielman and Teng 2013; Spielman and Teng 2014), each of which prompted new extensive areas of research. Subsequently, several authors have simplified and improved their solver (most notably Koutis, Miller, and Peng 2010; Michael B. Cohen et al. 2014; Michael B Cohen, Kyng, et al. 2014; Michael B Cohen, Miller, et al. 2014; Kelner et al. 2013a; Kelner et al. 2013b; Lee and Sidford 2013; Castelli Aleardi, Nolin, and Ovsjanikov 2015). The current fastest Laplacian solver is the one of (Michael B. Cohen et al. 2014), who achieve a solver that achieves a solution with relative error ϵ in time $\mathcal{O}(m \log^{1/2} n \log\log^{c} n log(1/\epsilon))$, where m denotes the number of non–zero entries in the Laplacian matrix $\mathbf{L}$ and n the size of the matrix. Ignoring the polylogarithmic terms, this running time is then nearly $\tilde{\mathcal{O}}(m \log^{1/2} n)$ (following convention, I use $\tilde{\mathcal{O}}$

[5] **Proof:** w_{ii} denotes the number of observations where the fixed effect i appeared. w_{ij} denotes the number of observations where the fixed effects i and j appeared. Thus, by construction, $w_{ii} = w_{ij}$ and the condition holds.

[6] See Appendix A of Kelner et al. (2013a) for details on how to apply the reduction on an SDD matrix.

when excluding polylogarithm and error terms).

That said, the solver discussed in this paper is mostly based on Kelner et al. (2013b), who propose a very simple and numerically stable solver that runs in $\mathcal{O}(m \log^2 g \log\log n log(1/\epsilon))$ time[7], or $\tilde{\mathcal{O}}(m \log^2 n)$ time. This solver can be accelerated to $\tilde{\mathcal{O}}(m \log^{3/2} n)$ thanks to the variant of accelerated coordinate descent proposed by Lee and Sidford (2013). Further, the tree constructions of Castelli Aleardi, Nolin, and Ovsjanikov (2015) and Michael B Cohen, Miller, et al. (2014) can be used to further improve its performance, although their combined time complexity has not been analyzed.

3.1 Definitions

Let[8] $\mathbf{Lv} = \boldsymbol{\chi}$ be a graph Laplacian system where $\mathbf{L}$ is the $g \times g$ matrix that represents a weighted undirected graph $G = (V, E, w)$ where V represents the set of vertices or nodes, E represents the set of edges or lines, and $w(e) > 0$ assigns a positive weight to each edge. Let the number of vertices be $g = |V|$ and the number of edges $m = |E|$. Similarly, denote the edge weights $w(e)$ as the conductance of an edge ($e = (a, b)$) and its reciprocal $r_e := 1/w_e$ is the resistance of such edge. Further, we will fix the edges so for two connected vertices (u, v), either $(u, v) \in E$ or $(v, u) \in E$. Finally, the vertex weights will be $w(a) = \sum_{(a,u)\in E} w(a, u)$.

Define the incidence matrix $\mathbf{B} \in \mathbb{R}^{E\times V}$:

$$\mathbf{B}_{(a,b),c} = \begin{cases} 1 & a = c \\ -1 & b = c \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

The resistance matrix $\mathbf{R} \in \mathbb{R}^{E\times E}$:

$$\mathbf{R}_{e_1,e_2} = \begin{cases} r(e) & e = e_1 = e_2 \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

The Laplacian matrix $\mathbf{L} \in \mathbb{R}^{V\times V}$:

$$\mathbf{L}_{a,b} = \begin{cases} w(a) & a = b \\ -w(a, b) & (a, b) \in E \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

As we can see in figure 1, the Laplacian is a full representation of both the structure of the fixed effects problem, as well as of the underlying graph.

[7] This paper estimates the running time in the unit–cost RAM model, which uses harsher assumptions on the speed of computations that most closely resemble that of modern computers

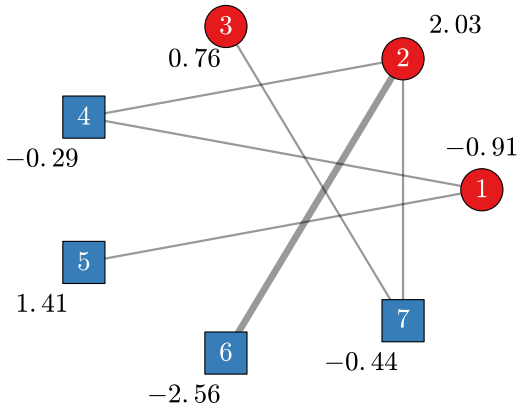
[8] This section closely follows the notation and definitions of Kelner et al. (2013b).

Indiv.	Firm	y
1	4	0.49
1	5	−1.41
2	4	−0.20
2	6	2.11
2	6	0.45
2	7	−0.32
3	7	0.76

(a) Dataset

$$\left(\begin{array}{ccc:cccc} 2 & & & -1 & -1 & & \\ & 4 & & -1 & & -2 & -1 \\ & & 1 & & & & -1 \\ \hdashline -1 & -1 & & 2 & & & \\ -1 & & & & 1 & & \\ & -2 & & & & 2 & \\ & -1 & -1 & & & & 2 \end{array}\right) \times \left(\begin{array}{c} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \hdashline -\alpha_4 \\ -\alpha_5 \\ -\alpha_6 \\ -\alpha_7 \end{array}\right) = \left(\begin{array}{c} -0.91 \\ 2.03 \\ 0.76 \\ \hdashline -0.29 \\ 1.41 \\ -2.56 \\ -0.44 \end{array}\right)$$

(b) Laplacian System



(c) Bipartite Graph (individuals: red circles, firms: blue squares)

Figure 1: Alternative Representations of the Fixed Effects

These figures illustrate how the fixed effects structure of a dataset can be represented as a Laplacian matrix, and therefore, as a weighted graph.

Given a vector $\mathbf{f} \in \mathbb{R}$ (a *flow vector* across edges), and an edge $e = (a, b)$, I follow the convention of setting $\mathbf{f}(a, b) = -\mathbf{f}(b, a)$, so f can be interpreted as sending a flow $\mathbf{f}(a, b)$ from a to b, or equivalently, a flow $-\mathbf{f}(a, b)$ from b to a. Then, the following claims follow (see Kelner et al. 2013a, sec. 2 for proofs):

1. $[\mathbf{B}'\mathbf{f}]_a = \sum_{(b,a)\in E} \mathbf{f}(b, a) - \sum_{(a,b)\in E} \mathbf{f}(a, b)$ (net flow in or out of each vertex; if the value is zero the flow is a *circulation*)
2. $\mathbf{L} = \mathbf{B}'\mathbf{R}^{-1}\mathbf{B}$
3. $[\mathbf{B}\mathbf{x}]_{(a,b)} = x(a) - x(b)$
4. $\mathbf{x}'\mathbf{L}\mathbf{x} = 2\sum_{(a,b)\in E} (x(a) - x(b))^2 / r_{(a,b)}$ (the quadratic form of the Laplacian can be interpreted as the potential energy of an electric flow)

3.2 Reduction of the Normal Equations into a Graph Laplacian

In our setting, the vertices of the graphs are the set of fixed effects, the edges are the pairs of fixed effects that share the same observations, and the weights are the sum of observations (or weights, in the econometric sense) that two pairs of fixed effects share.

To reduce the normal equation $(\mathbf{D}'\mathbf{D})\hat{\boldsymbol{\alpha}} = (\mathbf{D}'\mathbf{y})$ into a graph Laplacian, we only need to slightly alter the model definition:

The equation

$$\mathbf{y} = \mathbf{D}\boldsymbol{\alpha} + \boldsymbol{\varepsilon}$$

can be specialized for $f = 2$:

$$\mathbf{y} = \mathbf{D}_1\boldsymbol{\alpha}_1 + \mathbf{D}_2\boldsymbol{\alpha}_2 + \boldsymbol{\varepsilon}$$

Now, change the signs of the second set of fixed effects:

$$\mathbf{y} = \mathbf{D}_1\boldsymbol{\alpha}_1 - \mathbf{D}_2(-\boldsymbol{\alpha}_2) + \boldsymbol{\varepsilon} = \mathbf{D}_1\boldsymbol{\alpha}_1 + \widetilde{\mathbf{D}}_2\tilde{\boldsymbol{\alpha}}_2 + \boldsymbol{\varepsilon}$$

Given this transformation, the matrix $\mathbf{D}'\mathbf{D}$ will then have positive values in its diagonal and negative in its off–diagonal elements, with each row and column adding up to zero ■.

With this simple transformation, we can then proceed to apply the graph–teoretical tools that will compose our solver.

3.3 Outline of the Solver

The Laplacian solver has four steps:

1. **Pruning:** iteratively remove all vertices of degree 1, and then all vertices of degree 2

2. **Reordering:** apply the reverse Cuthill–McKee algorithm (RCM) to reorder the Laplacian and reduce its bandwidth. This also returns the number of disconnected subgraphs in the graph.
3. **Build a tree:** For each disconnected subgraphs, build a low–stretch spanning tree (LSST)
4. **Solve an electric flow problem:** Select a starting flow that is feasible on the LSST. Then, iterate K times; for each iteration randomly augment the LSST with an additional edge, and adjust the flow to make the Kirchoff's Potential Law (KPL) hold. Then, recover the fixed effects from the optimal flows.

Step 1 is the Greedy Elimination algorithm discussed by Koutis, Miller, and Peng (2010). Step 2 follows Pedroche Sánchez et al. (2012a) and addresses the cache locality concerns discussed in the benchmark papers of Hoske et al. (2015) and Boman, Deweese, and Gilbert (2015). Steps 3 to 5 are part of the Dual Randomized Kaczmarz (DRK) solver of Kelner et al. (2013b). These are discussed in more detail below.

3.4 Graph Pruning into a 3–core

For simplicity, suppose we are solving a model with CEO and firm fixed effects, with a graph described as in figure 2. If a firm only had one CEO through the sample, then it's associated vertex has a degree of 1. In this case, we can remove all the observations corresponding to this firm, solve the remaining system of equations, and then subsequently recover the fixed effect of the deleted firm.

To see why, note that in these cases the system has a triangular structure, so the normal equation can be made to hold independently of the value of the fixed effect of the single CEO that worked at that firm in the sample.

After removing the degree–1 vertices, we proceed to greedily remove all the degree-2 vertices. Thus, if vertex a neighbors vertex b and c, it will be removed, together with the associated edges, and a new edge will be created directly between b and c. Note that the specifics of the method are described in Algorithm 3 of Koutis, Miller, and Peng (2010).

Importantly, this method can be applied in other preexisting solvers independently of the other steps. By doing so, it addresses some of the worst–case scenarios of implementations such as Guimarães and Portugal (2010), Gaure (2013b) and Gomez (2016).

3.5 Reordering through the RCM Algorithm

One empirical problem of the Dual Randomized Kaczmarz, which Hoske et al. (2015) and Boman, Deweese, and Gilbert (2015) discuss, is that its implementation requires access to many noncontiguous memory addresses, which leads to a problem known in computer science as *cache misses*, which might dramatically decrease the speed of the solver. As a solution,

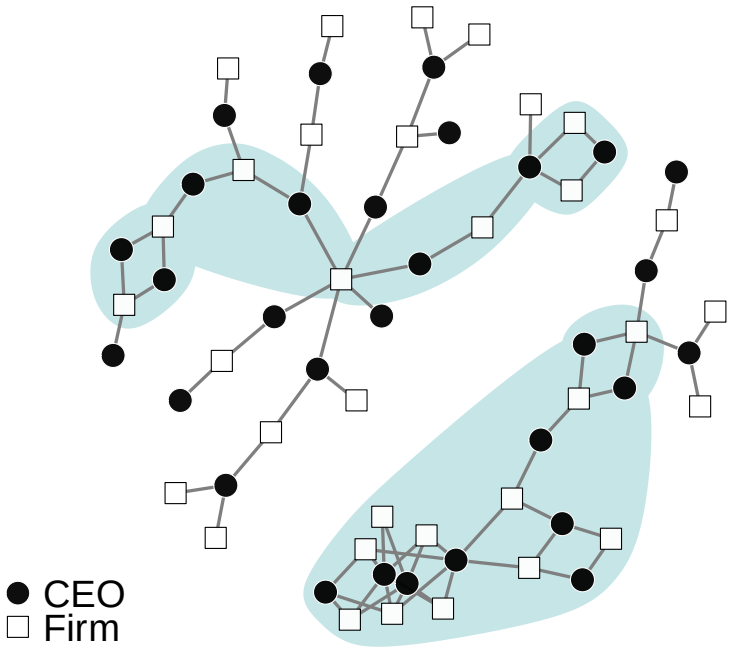


Figure 2: Graph of CEO–Firm Connections

I implement the reverse Cuthill–McKee algorithm (RCM) applied to the Laplacian matrix, as proposed by Pedroche Sánchez et al. (2012a) and illustrated in Pedroche Sánchez et al. (2012b). This algorithm reduces the bandwidth of the matrix, increasing the likelihood that its elements are contiguous in memory. Further, once this algorithm is applied, it is trivial to compute the number of disconnected subgraphs, an input required for the next steps.

3.6 Building a Low–Stretch–Spanning Tree

A spanning tree $T = (V, E_T, w_T)$ is a subgraph of $G = (V, E, w)$ that shares the same vertices, but that has exactly one possible path between any two nodes (if the graph G is disconnected, we work on each subgraph separately). Alternatively, it can be described as a graph that has no cycles (i.e. there is no path leaving a vertex that can arrive back at the same vertex without using an edge more than once).

The reason we are interested in trees with linear systems is solving them takes only $\mathcal{O}(n)$ time, as they can be solved either with the graph pruning method described above (or more generally with Cholesky factorization). Thus, they can in general be used as preconditioners of any system (including conjugate gradient solvers) as they are relatively easy to solve but might have spectral properties similar to the original graph.

Building a spanning tree can be done efficiently with a greedy algorithm such as Kruskal's (Cormen et al. 2009). However, the choice of the tree has a large effect on the speed on the subsequent solver, so Spielman and Teng (2011) propose a special type of tree, known as a low–stretch spanning tree. For each edge $e = (a, b) \in E$, they define the stretch of the edge with respect to the tree T as the relative cost of traversing the tree to reach b from a through the tree with respect to the cost of traversing it directly. Further, they find systems with minimizing the stretch of a tree also reduces the number of iterations required for convergence. However, in contrast to algorithms such as Kruskal's, it is not trivial to find a fast method to build a LSST (see Abraham and Neiman 2014 for a discussion).

The fastest known algorithm to build a LSST is by Abraham and Neiman (2012), building upon Spielman and Teng (2011). However, as discussed by Papp (2014), the benefits of using a LSST might not be worth it on average, specially in comparison with breadth-first-search methods such as Kruskal's. As a solution, Castelli Aleardi, Nolin, and Ovsjanikov (2015) and Michael B Cohen, Miller, et al. (2014) propose modifications to LSSTs that achieve better empirical performance (Hoske et al. 2015 also find that LSST often perform worst that several alternatives)

3.7 Solving the Electrical Flow Problem

Given a spanning tree, the next step of our estimator consists in solving the dual problem of solving the Laplacian system with the dual solver of Kelner et al. (2013b).

A vector flow $\mathbf{f} \in \mathbb{R}^E$ has an energy of $\xi(f) := \mathbf{f}'\mathbf{R}\mathbf{f} = ||\mathbf{f}||^2_{\mathbf{R}}$. Additionally, this flow is *feasible* with respect to an energy demand at each vertex $\boldsymbol{\chi} \in \mathbb{R}^V$ if it meets the demand so $\mathbf{B}'\mathbf{f} = \boldsymbol{\chi}$.

For this to happen, the demand of the system must add up to zero (otherwise the flow does not transmit all the energy it receives or viceversa). The practical implications of this is that we must first normalize our variables so they add up to zero for each disconnected graph. This is equivalent to demeaning the variables, and is also desirable because it abstracts from the problem of dropping a particular fixed effect category, or assigning the constant to one of the two sets of fixed effects.

Following Kelner et al. (2013b), we will focus on the following dual problem to the primal system $\mathbf{L}\mathbf{v} = \boldsymbol{\chi}$:

$$\min_{\mathbf{f} \in \mathbb{R}^E : \mathbf{B}'\mathbf{f}} \xi(f) \tag{5}$$

Denote $\delta(a, b) = \mathbf{v}(a) - \mathbf{v}(b)$ be the voltage protential across the edge (a, b). Also denote a circulation as a cycle across a graph. From the KKT conditions to the problem, we can see that an analogue of Ohm's Law holds: the optimal flow $\mathbf{f}*$ is such that $\mathbf{f}*(e) = \delta(a, b)/r(a, b)$. Further, we can derive the optimality conditions $\mathbf{f}* = \mathbf{R}^{-1}\mathbf{B}\mathbf{v}*$ and restate them in terms of Krichoff's Potential Law (KPL):

A feasible $\mathbf{f} \in \mathbb{R}$ is optimal if and only if $\mathbf{f}'\mathbf{R}\mathbf{c} = 0$ for all circulations $\mathbf{c} \in \mathbb{R}^E$.

Given the above result, we then apply the *SimpleSolver* algorithm by Kelner et al. (2013b) in Section 3.

4 Multi-Way Fixed Effects and Alternating Projections

The Method of Alternating Projections (MAP) of Von Neumann (1949) and Halperin (1962) consists in iteratively applying a transformation across a subspace until convergence is achieved. In our case, Guimarães and Portugal (2010) and Gaure (2013a) implement a block version of MAP that demeans each variable across a fixed effect, obtains the residuals, and then repeats cyclically across all fixed effects until the residuals converge to the partialled–out variables. It has guaranteed convergence, although it can be *arbitrarily slow*, meaning that for any $N > 0$ we can always find a dataset of fixed size that will require at least N iterations. As a solution to this problem, Guimarães and Portugal (2010) and Gaure (2013a) implemented acceleration methods (Aitken's acceleration and a variant of steep descent, respectively) that

on average converge faster but that as described by Hernández-Ramos, Escalante, and Raydan (2011) have no guarantees of monotonic convergence and in fact, can perform even worst than without accelerations if the underlying graph is poorly connected (or for $f > 2$ if the angle between subspaces is low).

Algorithm 1 Method of Alternating Projections (MAP)

Input: $\text{vec}\, y \in \mathbb{R}^n$ (variable); $\text{vec}\, w \in \mathbb{R}^n_+$ (weights); $\epsilon \in \mathbb{R}_+$ (tolerance)
Output: $\widetilde{\text{vec}}\, y \in \mathbb{R}^n$ (residuals)

1 **repeat**
2 $\quad \widetilde{\text{vec}}\, y \leftarrow \text{T}(\text{vec}\, y, \text{vec}\, w)$ $\quad \triangleright$ T is either Halperin, Symmetric Halperin or Cimmino
3 $\quad \delta \leftarrow ||\widetilde{\text{vec}}\, y - \text{vec}\, y||_2 \,/\, ||\widetilde{\text{vec}}\, y||_2$ $\quad \triangleright$ Relative difference
4 $\quad \text{vec}\, y \leftarrow \widetilde{\text{vec}}\, y$
5 **until** $\delta \leq \epsilon$
6 **return** $\widetilde{\text{vec}}\, y$

As a solution, I present two alternative transformations to the Halperin transformation implemented by Guimarães and Portugal (2010) and Gaure (2013a). They have the advantage of being symmetric, which allows us to apply a conjugate gradient acceleration to them, which in turn (Hernández-Ramos, Escalante, and Raydan 2011) has monotonic convergence at a speed much faster than previous methods. In forthcoming benchmarks, the Symmetric MAP variant performs faster on average, with the Cimmino transform performing relatively slow but with more stable convergence times.

Algorithm 2 Halperin Transform

Input: Regression variable $\text{vec}\, y \in \mathbb{R}^n$; weights $\text{vec}\, w \in \mathbb{R}^n_+$

1 **function** HALPERIN($\text{vec}\, y, \text{vec}\, w$)
2 $\quad$ **for** $d \leftarrow 1$ to D **do**
3 $\quad\quad \text{vec}\, y \leftarrow \text{vec}\, y - \texttt{mean}_d(\text{vec}\, y; \text{vec}\, w)$ $\quad \triangleright$ Subtract the average across each category
4 $\quad$ **end for**
5 $\quad$ **return** $\text{vec}\, y$
6 **end function**

4.1 Solver Embedding

We can embed the Laplacian solver of the previous section into the MAP framework, by combining any two sets of projections. For instance, if $f = 3$ and we have (P_1, P_2, P_3) projections, we can group the first pair into a joint projection P_{12} which can be then solved by the Laplacian solver in each iteration of this solver. Note that since the Laplacian solver precomputes many of its results, its performance will be much faster after the first iteration is completed.

Algorithm 3 Symmetric Halperin Transform

Input: Regression variable vec $y \in \mathbb{R}^n$; weights vec $w \in \mathbb{R}^n_+$

function SymmetricHalperin(vec y, vec w)
  for $d \leftarrow 1, 2, \ldots D, D-1, \ldots, 2, 1$ **do**
    vec $y \leftarrow$ vec $y - \texttt{mean}_d(\text{vec } y; \text{vec } w)$ ▷ Subtract the avg. across each category
  end for
  return vec y
end function

Algorithm 4 Cimmino Transform

Input: Regression variable vec $y \in \mathbb{R}^n$; weights vec $w \in \mathbb{R}^n_+$

function Cimmino(vec y, vec w)
  vec $z \leftarrow 0_{n \times 1}$
  for $d \leftarrow 1$ to D **do**
    vec $z \leftarrow$ vec $z + \texttt{mean}_d(\text{vec } y; \text{vec } w)$ ▷ N.B. parallelizable step
  end for
  return vec $y -$ vec z/D
end function

5 Conclusion

In this paper, I have shown a method for solving a linear model with arbitrarily many fixed effects with many dimensions. It addresses many shortcomings of existing estimators, that had slow convergence properties, in particular with large and complex datasets.

It can also be generalized beyond OLS models, and used as a building block for nonlinear and other estimators. Further, by applying graph–theoretical tools, it can leverage existing techniques such as the estimation of relative condition numbers, in order to assess the robustness of the estimation.

Finally, a note of caution remains. This paper has not dealt with the estimation and identification of the parameters associated with the fixed effects, which is still an open problem.

References

Abowd, John M., Robert H. Creecy, and Francis Kramarz. 2002. *Computing Person and Firm Effects Using Linked Longitudinal Employer-Employee Data*. Longitudinal Employer-Household Dynamics Technical Papers 2002-06. Center for Economic Studies, U.S. Census Bureau. http://ideas.repec.org/p/cen/tpaper/2002-06.html.

Abowd, John M., Francis Kramarz, and David N. Margolis. 1999. "High Wage Workers

and High Wage Firms." *Econometrica* 67 (2). The Econometric Society: 251–333. http://www.jstor.org/stable/2999586.

Abraham, Ittai, and Ofer Neiman. 2012. "Using Petal-Decompositions to Build a Low Stretch Spanning Tree." In *Proceedings of the Forty-Fourth Annual ACM Symposium on Theory of Computing, STOC '12*, 395–406. STOC '12. New York, NY, USA: ACM. doi:10.1145/2213977.2214015.

———. 2014. "Encyclopedia of Algorithms." In, edited by Ming-Yang Kao, 1–5. Boston, MA: Springer US. doi:10.1007/978-3-642-27848-8_804-1.

Andrews, Martyn, Thorsten Schank, and Richard Upward. 2006. "Practical Fixed-Effects Estimation Methods for the Three-Way Error-Components Model." *Stata Journal* 6 (4). College Station, TX: Stata Press: 461–48121. http://www.stata-journal.com/article.html?article=st0112.

Arcidiacono, Peter, Gigi Foster, Natalie Goodpaster, and Josh Kinsler. 2012. "Estimating Spillovers Using Panel Data, with an Application to the Classroom." *Quantitative Economics* 3 (3). Blackwell Publishing Ltd: 421–70. doi:10.3982/QE145.

Bai, Jushan. 2009. "Panel Data Models with Interactive Fixed Effects." *Econometrica* 77 (4). Blackwell Publishing Ltd: 1229–79. doi:10.3982/ECTA6135.

Baltagi, Badi. 2008. *Econometric Analysis of Panel Data*. John Wiley & Sons. https://books.google.com/books?id=0Qdx_70XmyoC.

Bauschke, Heinz, Frank Deutsch, Hein Hundal, and Sung-Ho Park. 2003. "Accelerating the Convergence of the Method of Alternating Projections." *Transactions of the American Mathematical Society* 355 (9): 3433–61.

Bertrand, Marianne, and Antoinette Schoar. 2003. "Managing with Style: The Effect of Managers on Firm Policies." *The Quarterly Journal of Economics* 118 (4): 1169–1208. doi:10.1162/003355303322552775.

Boman, Erik G, Kevin Deweese, and John R Gilbert. 2015. "Evaluating the Potential of a Laplacian Linear Solver." *CoRR* abs/1505.00875. http://arxiv.org/abs/1505.00875.

Carneiro, Anabela, Paulo Guimarães, and Pedro Portugal. 2012. "Real Wages and the Business Cycle: Accounting for Worker, Firm, and Job Title Heterogeneity." *American Economic Journal: Macroeconomics* 4 (2): 133–52. doi:10.1257/mac.4.2.133.

Castelli Aleardi, Luca, Alexandre Nolin, and Maks Ovsjanikov. 2015. "Experimental Algorithms: 14th International Symposium, SEA 2015, Paris, France, June 29 – July 1, 2015, Proceedings." In, edited by Evripidis Bampis, 219–31. Cham: Springer International Publishing. doi:10.1007/978-3-319-20086-6_17.

Cattaneo, Matias D, Michael Jansson, and Whitney K Newey. 2015. "Treatment Effects with Many Covariates and Heteroskedasticity." *ArXiv Preprint ArXiv:1507.02493*, July. http:

//arxiv.org/abs/1507.02493v1.

Chetty, Raj, John N. Friedman, and Jonah E. Rockoff. 2014. "Measuring the Impacts of Teachers II: Teacher Value-Added and Student Outcomes in Adulthood." *American Economic Review* 104 (9): 2633–79. doi:10.1257/aer.104.9.2633.

Cohen, Michael B, Rasmus Kyng, Jakub W Pachocki, Richard Peng, and Anup Rao. 2014. "Preconditioning in Expectation." *CoRR* abs/1401.6236. http://arxiv.org/abs/1401.6236.

Cohen, Michael B, Gary L Miller, Jakub W Pachocki, Richard Peng, and Shen Chen Xu. 2014. "Stretching Stretch." *CoRR* abs/1401.2454. http://arxiv.org/abs/1401.2454.

Cohen, Michael B., Rasmus Kyng, Gary L. Miller, Jakub W. Pachocki, Richard Peng, Anup B. Rao, and Shen Chen Xu. 2014. "Solving SDD Linear Systems in Nearly $m \log^{1/2} n$ Time." In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing, STOC '14*, 343–52. STOC '14. New York, NY, USA: ACM. doi:10.1145/2591796.2591833.

Cormen, Thomas H., Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. 2009. *Introduction to Algorithms, Third Edition*. 3rd ed. MIT Press. https://books.google.com/books?id=i-bUBQAAQBAJ.

Cornelissen, T. 2008. "The Stata Command Felsdvreg to Fit a Linear Model with Two High-Dimensional Fixed Effects." *Stata Journal* 8 (2). College Station, TX: Stata Press: 170–18920. http://www.stata-journal.com/article.html?article=sto143.

Einav, Liran, and Jonathan Levin. 2014. "Economics in the Age of Big Data." *Science* 346 (6210). doi:10.1126/science.1243089.

Fong, David Chin-Lung, and Michael Saunders. 2011. "LSMR: An Iterative Algorithm for Sparse Least-Squares Problems." *SIAM Journal on Scientific Computing* 33 (5). SIAM: 2950–71. doi:10.1137/10079687X.

Frisch, Ragnar, and Frederick V. Waugh. 1933. "Partial Time Regressions as Compared with Individual Trends." *Econometrica* 1 (4). The Econometric Society: 387–401. http://www.jstor.org/stable/1907330.

Gaure, Simen. 2013a. "OLS with Multiple High Dimensional Category Variables." *Computational Statistics & Data Analysis* 66 (0): 8–18. doi:10.1016/j.csda.2013.03.024.

———. 2013b. "lfe: Linear Group Fixed Effects." *The R Journal* 5 (2): 104–17. http://journal.r-project.org/archive/2013-2/gaure.pdf.

———. 2015. *Convergence Rate Examples and Theory*. Ragnar Frisch Centre for Economic Research, University of Oslo. https://cran.r-project.org/web/packages/lfe/vignettes/speed.pdf.

Golub, Gene H., and Charles F. Van Loan. 2013. *Matrix Computations*. Matrix Computations. Johns Hopkins University Press. https://books.google.com/books?id=5U-l8U3P-VUC.

Gomez, Matthieu. 2016. "FixedEffectModels.jl." *GitHub Repository*. https://github.com/matthieugomez/FixedEffectModels.jl; GitHub. https://github.com/matthieugomez/

FixedEffectModels.jl.

Gormley, Todd A., and David A. Matsa. 2014. "Common Errors: How to (and Not to) Control for Unobserved Heterogeneity." *Review of Financial Studies* 27 (2): 617–61. doi:10.1093/rfs/hht047.

Guimaraes, Paulo. 2014. "POI2HDFE: Stata module to estimate a Poisson regression with two high-dimensional fixed effects." Statistical Software Components, Boston College Department of Economics. https://ideas.repec.org/c/boc/bocode/s457777.html.

Guimarães, Paulo, and Pedro Portugal. 2010. "A Simple Feasible Procedure to Fit Models with High-Dimensional Fixed Effects." *Stata Journal* 10 (4). College Station, TX: Stata Press: 628–64922. http://www.stata-journal.com/article.html?article=sto212.

Halperin, Israel. 1962. "The Product of Projection Operators." *Acta Sci. Math.(Szeged)* 23: 96–99.

Head, Keith, and Thierry Mayer. 2014. "Chapter 3 - Gravity Equations: Workhorse,Toolkit, and Cookbook." In *Handbook of International Economics*, edited by Elhanan Helpman Gita Gopinath and Kenneth Rogoff, 4:131–95. Handbook of International Economics. Elsevier. doi:10.1016/B978-0-444-54314-1.00003-3.

Hernández-Ramos, Luis M., René Escalante, and Marcos Raydan. 2011. "Unconstrained Optimization Techniques for the Acceleration of Alternating Projection Methods." *Numerical Functional Analysis and Optimization* 32 (10): 1041–66. doi:10.1080/01630563.2011.591954.

Hoske, Daniel, Dimitar Lukarski, Henning Meyerhenke, and Michael Wegner. 2015. "Experimental Algorithms: 14th International Symposium, SEA 2015, Paris, France, June 29 – July 1, 2015, Proceedings." In, edited by Evripidis Bampis, 205–18. Cham: Springer International Publishing. doi:10.1007/978-3-319-20086-6_16.

Kelner, Jonathan A., Lorenzo Orecchia, Aaron Sidford, and Zeyuan Allen Zhu. 2013a. "A Simple, Combinatorial Algorithm for Solving SDD Systems in Nearly-Linear Time." *CoRR* abs/1301.6628. http://arxiv.org/abs/1301.6628.

———. 2013b. "A Simple, Combinatorial Algorithm for Solving SDD Systems in Nearly-Linear Time." In *Proceedings of the Forty-Fifth Annual ACM Symposium on Theory of Computing, STOC '13*, 911–20. STOC '13. New York, NY, USA: ACM. doi:10.1145/2488608.2488724.

Koutis, Ioannis, Gary L Miller, and Richard Peng. 2010. "Approaching Optimality for Solving SDD Linear Systems." In *Foundations of Computer Science (FOCS), 2010 51st Annual IEEE Symposium on*, 235–44. IEEE. doi:10.1137/110845914.

Koutis, Ioannis, Gary L. Miller, and Richard Peng. 2012. "A Fast Solver for a Class of Linear Systems." *Commun. ACM* 55 (10). New York, NY, USA: ACM: 99–107. doi:10.1145/2347736.2347759.

Lee, Yin Tat, and Aaron Sidford. 2013. "Efficient Accelerated Coordinate Descent Methods and Faster Algorithms for Solving Linear Systems." *CoRR* abs/1305.1922. http://arxiv.org/abs/

1305.1922.

Lovell, Michael C. 1963. "Seasonal Adjustment of Economic Time Series and Multiple Regression Analysis." *Journal of the American Statistical Association* 58 (304). [American Statistical Association, Taylor & Francis, Ltd.]: 993–1010. http://www.jstor.org/stable/2283327.

Mittag, Nikolas. 2015. "A Simple Method to Estimate Large Fixed Effects Models Applied to Wage Determinants and Matching." *CERGE-EI Working Paper Series*, no. 532.

Papp, Pál András. 2014. "Low-Stretch Spanning Trees." B.S. Thesis, Eötvös Loránd University. https://www.cs.elte.hu/blobs/diplomamunkak/bsc_alkmat/2014/papp_pal_andras.pdf.

Pedroche Sánchez, Francisco, Miguel Rebollo Pedruelo, Alberto Palomares Chust, and Carlos Carrascosa Casamayor. 2012a. "L-RCM: A Method to Detect Connected Components in Undirected Graphs by Using the Laplacian Matrix and the RCM Algorithm." *CoRR* abs/1206.5726. http://arxiv.org/abs/1206.5726.

———. 2012b. "Some Examples of Detection of Connected Components in Undirected Graphs by Using the Laplacian Matrix and the RCM Algorithm." *International Journal of Complex Systems in Science* 2 (1): 11–15.

Rockoff, Jonah E. 2004. "The Impact of Individual Teachers on Student Achievement: Evidence from Panel Data." *American Economic Review*. JSTOR, 247–52.

Somaini, Paulo, and Frank A Wolak. 2015. "An Algorithm to Estimate the Two-Way Fixed Effects Model." *Journal of Econometric Methods*. De Gruyter, to appear. http://www.degruyter.com/view/j/jem.ahead-of-print/jem-2014-0008/jem-2014-0008.xml.

Spielman, Daniel A. 2010. "Algorithms, Graph Theory, and Linear Equations in Laplacian Matrices." In *Proceedings of the International Congress of Mathematicians: Hyderabad, August 19-27, 2010*, 4:2698–2722. Hindustan Book Agency. https://books.google.com/books?id=P8oxnwEACAAJ.

Spielman, Daniel A, and Shang-Hua Teng. 2014. "Nearly Linear Time Algorithms for Preconditioning and Solving Symmetric, Diagonally Dominant Linear Systems." *SIAM Journal on Matrix Analysis and Applications* 35 (3). SIAM: 835–85. doi:10.1137/090771430.

Spielman, Daniel A., and Shang-Hua Teng. 2004. "Nearly-Linear Time Algorithms for Graph Partitioning, Graph Sparsification, and Solving Linear Systems." In *Proceedings of the Thirty-Sixth Annual ACM Symposium on Theory of Computing, STOC '04*, 81–90. STOC '04. New York, NY, USA: ACM. doi:10.1145/1007352.1007372.

———. 2011. "Spectral Sparsification of Graphs." *SIAM Journal on Computing* 40 (4). Philadelphia, PA, USA: Society for Industrial; Applied Mathematics: 981–1025. doi:10.1137/08074489X.

———. 2013. "A Local Clustering Algorithm for Massive Graphs and Its Application to Nearly Linear Time Graph Partitioning." *SIAM Journal on Computing* 42 (1). Society for In-

dustrial; Applied Mathematics: 1–26. doi:10.1137/080744888.

Teng, Shang-Hua. 2010. "Theory and Applications of Models of Computation: 7th Annual Conference, TAMC 2010, Prague, Czech Republic, June 7-11, 2010. Proceedings." In, edited by Jan Kratochvíl, Angsheng Li, Jiří Fiala, and Petr Kolman, 2–14. Berlin, Heidelberg: Springer Berlin Heidelberg. doi:10.1007/978-3-642-13562-0_2.

Vaidya, Pravin M. 1991. "Solving Linear Equations with Symmetric Diagonally Dominant Matrices by Constructing Good Preconditioners."

Von Neumann, John. 1949. "On Rings of Operators. Reduction Theory." *Annals of Mathematics* 50 (2): 401–85. http://www.jstor.org/stable/1969463.

Wooldridge, J.M. 2010. *Econometric Analysis of Cross Section and Panel Data*. MIT Press. https://books.google.com/books?id=yov6AQAAQBAJ.