Data and Computational Appendix to "Leave-out estimation of variance components"

Patrick Kline, Raffaele Saggio, Mikkel Sølvsten

March 14, 2020

1 Data

This Appendix describes construction of the data used in the application of Section 9.

1.1 Veneto Workers History

Our data come from the Veneto Workers History (VWH) file, which provides social security based earnings records on annual job spells for all workers employed in the Italian region of Veneto at any point between the years 1975 and 2001. Each job-year spell in the VWH lists a start date, an end date, the number of days worked that year, and the total wage compensation received by the employee in that year. The earnings records are not top-coded. We also observe the gender of each worker and several geographic variables indicating the location of each employer. See Card, Devicienti, and Maida (2014) and Serafinelli (2019) for additional discussion and analysis of the VWH.

We consider data from the years 1984–2001 as prior to that information on days worked tend to be of low quality. To construct the person-year panel used in our analysis, we follow the sample selection procedures described in Card, Heining, and Kline (2013). First, we drop employment spells in which the worker's age lies outside the range 18–64. The average worker in this sample has 1.21 jobs per year. To generate unique worker-firm assignments in each year, we restrict attention to spells associated with "dominant jobs" where the worker earned the most in each corresponding year. From this person-year file, we then exclude workers that (i) report a daily wage less than 5 real euros or have zero days worked (1.5% of remaining person-year observations) (ii) report a log daily wage change one year to the next that is greater than 1 in absolute value (6%) (iii) are employed in the public sector (10%) or (iv) have more than 10 jobs in any year or that have gender missing (0.1%).

2 Computation

This Appendix describes the key computational aspects of the leave-out estimator $\hat{\theta}$, with an emphasis on the application to two-way fixed effects models with two time periods discussed in Example 3 and Section 9.

2.1 Leave-One-Out Connected Set

Existence of $\hat{\theta}$ requires $P_{ii} < 1$ (see Lemma 1) and the following describes an algorithm which prunes the data to ensure that $P_{ii} < 1$. In the two-way fixed effects model of Section 9, this condition requires that the bipartite network formed by worker-firm links remains connected when any one worker is removed. This boils down to finding workers that constitute cut vertices or *articulation points* in the corresponding bipartite network.

The algorithm below takes as input a connected bipartite network $\mathcal{G}$ where workers and firms are vertices. Edges between two vertices correspond to the realization of a match between a worker and a firm (see Jochmans and Weidner, 2019; Bonhomme, 2017, for discussion). In practice, one typically starts with a $\mathcal{G}$ corresponding to the *largest connected component* of a given bipartite network (see, e.g., Card et al., 2013). The output of the algorithm is a subset of $\mathcal{G}$ where removal of any given worker does not break the connectivity of the associated graph.

The algorithm relies on existing functions that efficiently finds articulation points and largest connected components. In MATLAB such functions are available in the *Boost Graph Library* and in R they are available in the *igraph* package.

Algorithm 1 Leave-One-Out Connected Set

```
1: function PruningNetwork(𝒢)        ▷ 𝒢 ≡ Connected bipartite network of firms and workers
2:     Construct 𝒢1 from 𝒢 by deleting all workers that are articulation points in 𝒢
3:     Let 𝒢 be the largest connected component of 𝒢1
4:     Return 𝒢
5: end function
```

The algorithm typically completes in less than a minute for datasets of the size considered in our application. Furthermore, the vast majority of firms removed using this algorithm are only associated with one mover.

2.2 Leave-Two-Out Connected Set

We also introduced a leave-two-out connected set, which is a subset of the original data such that removal of any *two* workers does not break the connectedness of the bipartite network formed by worker-firm links. The following algorithm proceeds by applying the idea in Algorithm 1 to each of the networks constructed by dropping one worker. A crucial difference from Algorithm 1 is that *two* workers who do not break connectedness in the input network may break connectedness when other workers have been removed. For this reason, the algorithm runs in an iterative fashion until it fails to remove any additional workers.

Algorithm 2 Leave-Two-Out Connected Set

function PRUNINGNETWORK2($\mathcal{G}$) $\quad \triangleright\ \mathcal{G} \equiv$ Leave-one-out connected bipartite network of firms and workers
  $a = 1$
  while $a > 0$ **do**
    $\mathcal{G}^{del} = \emptyset$
    for $g = 1, \ldots, N$ **do**
      Construct $\mathcal{G}_1$ from $\mathcal{G}$ by deleting worker g
      Add all workers that are articulation points in $\mathcal{G}_1$ to $\mathcal{G}^{del}$
    end for
    $a = |\mathcal{G}^{del}|$
    if $a > 0$ **then**
      Construct $\mathcal{G}_1$ from $\mathcal{G}$ by deleting all workers in $\mathcal{G}^{del}$
      Let $\mathcal{G}_2$ be the largest connected component of $\mathcal{G}_1$
      Let $\mathcal{G}$ be the output of applying Algorithm 1 to $\mathcal{G}_2$
    end if
  end while
  Return $\mathcal{G}$
end function

2.3 Computing $\hat{\theta}$

Our proposed leave-out estimator is a function of the $2n$ quadratic forms

$$P_{ii} = x_i' S_{xx}^{-1} x_i \qquad B_{ii} = x_i' S_{xx}^{-1} A S_{xx}^{-1} x_i \qquad \text{for } i = 1, ..., n.$$

The estimates reported in Section 9 of the paper rely on exact computation of these quantities. In our application, k is on the order of hundreds of thousands, making it infeasible to compute S_{xx}^{-1} directly. To circumvent this obstacle, we instead compute the k-dimensional vector $z_{i,exact} = S_{xx}^{-1} x_i$ separately for each $i = 1, .., n$. That is, we solve separately for each column of Z_{exact} in the system

$$\underset{k\times k}{S_{xx}} \underset{k\times n}{Z_{exact}} = \underset{k\times n}{X'}.$$

We then form $P_{ii} = x_i' z_{i,exact}$ and $B_{ii} = z_{i,exact}' A z_{i,exact}$. The solution $z_{i,exact}$ is computed via MATLAB's preconditioned conjugate gradient routine *pcg*. In computing this solution, we utilize the preconditioner developed by Koutis et al. (2011), which is optimized for diagonally dominant design matrices S_{xx}. These column-specific calculations are parallelized across different cores using MATLAB's *parfor* command.

2.3.1 Leaving a Cluster Out

Table A.1 applies the leave-cluster-out estimator introduced in Remark 3 to estimate the variance of firm effects with more than two time periods and potential serial correlation. The estimator takes the form $\hat{\theta}_{cluster} = \sum_{i=1}^{n} y_i \tilde{x}_i' \hat{\beta}_{-c(i)}$ where $\hat{\beta}_{-c(i)}$ is the OLS estimator obtained after leaving out all observations in the cluster to which observation i belongs. A representation of $\hat{\theta}_{cluster}$ that is useful for computation takes the observations in the c-th cluster and collect their outcomes in y_c and their regressors in X_c. The leave-cluster-out estimator is then

$$\hat{\theta}_{cluster} = \hat{\beta}' A \hat{\beta} - \sum_{c=1}^{C} y_c' B_c (I - P_c)^{-1} (y_c - X_c \hat{\beta}),$$

where C denotes the total number of clusters, $P_c = X_c S_{xx}^{-1} X_c'$, and $B_c = X_c S_{xx}^{-1} A S_{xx}^{-1} X_c'$. Since the entries of P_c and B_c are of the form $P_{i\ell} = x_i' S_{xx}^{-1} x_\ell$ and $B_{i\ell} = x_i' S_{xx}^{-1} A S_{xx}^{-1} x_\ell$, computation can proceed in a similar fashion as described earlier for the leave-one-out estimator.

When defining the cluster as a worker-firm match, Table A.1 applies $\hat{\theta}_{cluster}$ to the two-way fixed effects model in (5). When defining the cluster as a worker, the individual effects can not be estimated after leaving a cluster out. Table A.1 therefore applies $\hat{\theta}_{cluster}$ after demeaning at the individual level. This transformation removes the individual effects so that the resulting model can be estimated after leaving a cluster out.

2.3.2 Johnson-Lindenstrauss Approximation

When n is on the order of hundreds of millions and k is on the order of tens of millions, the exact algorithm may no longer be tractable. The JLA simplifies computation of P_{ii} considerably by only requiring the solution of p systems of k linear equations. That is, one need only solve for the columns of Z_{JLA} in the system

$$\underset{k\times k}{S_{xx}}\underset{k\times p}{Z_{JLA}} = \underset{k\times p}{(R_P X)'},$$

which reduces computation time dramatically when p is small relative to n.

To compute B_{ii}, it is necessary to solve linear systems involving both A_1 and A_2, leading to $2p$ systems of equations when $A_1 \neq A_2$. However, for variance decompositions like the ones considered in Section 9, the same $2p$ systems can be reused for all three variance components, leading to a total of $3p$ systems of equations for the full variance decomposition. This is so because the three variance components use the matrices $A_\psi = A_f' A_f$, $A_{\alpha,\psi} = \frac{1}{2}(A_d' A_f + A_f' A_d)$, and $A_\alpha = A_d' A_d$ where

$$A_f' = \tfrac{1}{\sqrt{n}}\begin{bmatrix} 0 & 0 & 0 \\ f_1 - \bar{f} & \dots & f_n - \bar{f} \\ 0 & 0 & 0 \end{bmatrix} \quad \text{and} \quad A_d' = \tfrac{1}{\sqrt{n}}\begin{bmatrix} d_1 - \bar{d} & \dots & d_n - \bar{d} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}.$$

Based on these insights, Algorithm 3 below takes as inputs X, A_f, A_d, and p, and returns $\hat{P}_{ii}$ and three different $\hat{B}_{ii}$'s which are ultimately used to construct the corresponding variance component $\hat{\theta}_{JLA}$ as defined in Section 3.

2.3.3 Performance of the JLA

Figure 2.1 evaluates the performance of the Johnson-Lindenstrauss approximation across 4 VWH samples that correspond to different (overlapping) time intervals (2000–2001; 1999–2001; 1998–2001; 1997–2001). The x-axis in Figure 2.1 reports the total number of person and firm effects associated with a particular sample.

Figure 2.1 shows that the computation time for exact computation of (B_{ii}, P_{ii}) increases rapidly as the number of parameters of the underlying AKM model grow; in the largest dataset considered – which involves more than a million worker and firm effects – exact computation takes about 8 hours. Computation of JLA complete in markedly shorter time: in the largest dataset considered computation time is less than 5 minutes when $p = 500$ and slightly over 6 minutes when $p = 2500$. Notably, the JLA delivers estimates of the variance

Algorithm 3 Johnson-Lindenstrauss Approximation for Two-Way Fixed Effects Models

function JLA(X,A_f,A_d,p)
  Generate $R_B, R_P \in \mathbb{R}^{p \times n}$, where (R_B, R_P) are composed of mutually independent Rademacher entries
  Compute $(R_P X)'$, $(R_B A_f)'$, $(R_B A_d)' \in \mathbb{R}^{k \times p}$
  for $\kappa = 1, \ldots, p$ **do**
    Let $r_{\kappa,0}$, $r_{\kappa,1}$, $r_{\kappa,2} \in \mathbb{R}^k$ be the κ-th columns of $(R_P X)'$, $(R_B A_f)'$, $(R_B A_d)'$
    Let $z_{\kappa,\ell} \in \mathbb{R}^k$ be the solution to $S_{xx} z = r_{\kappa,\ell}$ for $\ell = 0, 1, 2$
  end for
  Construct $Z_\ell = (z_{1,\ell}, \ldots, z_{p,\ell}) \in \mathbb{R}^{k \times p}$ for $\ell = 0, 1, 2$
  Construct $\hat{P}_{ii} = \frac{1}{p}\left\|Z_0' x_i\right\|^2$, $\hat{B}_{ii,\psi} = \frac{1}{p}\left\|Z_1' x_i\right\|^2$, $\hat{B}_{ii,\alpha} = \frac{1}{p}\left\|Z_2' x_i\right\|^2$, $\hat{B}_{ii,\alpha\psi} = \frac{1}{p}(Z_1' x_i)'(Z_2' x_i)$ for $i = 1, \ldots, n$
  Return $\{\hat{P}_{ii}, \hat{B}_{ii,\psi}, \hat{B}_{ii,\alpha}, \hat{B}_{ii,\alpha\psi}\}_{i=1}^n$
end function

of firm effects almost identical to those computed via the exact method, with the quality of the approximation increasing for larger p. For instance, in the largest dataset, the exact estimate of variance of firm effects is 0.028883. By comparison, the JLA estimate equals 0.028765 when $p = 500$ and 0.0289022 when $p = 2500$.

In summary: for a sample with more than a million worker and firm effects, the JLA cuts computation time by a factor of 100 while introducing an approximation error of roughly 10^{-4}.

2.3.4 Scaling to Very Large Datasets

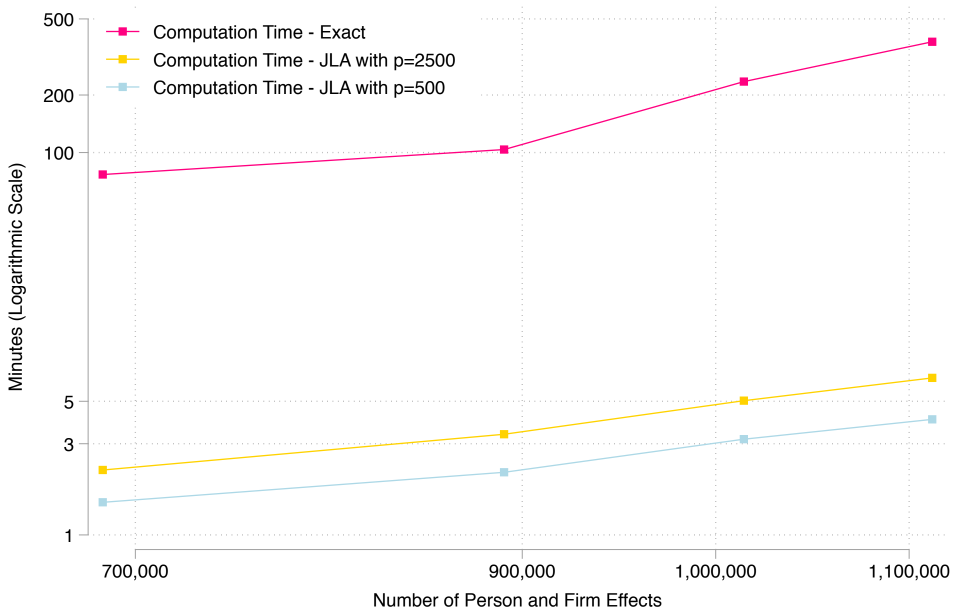
We now study how the JLA scales to much larger datasets of the dimension considered by Card et al. (2013) who fit models involving tens of millions of worker and firm effects to German social security records. To study the computational burden of a model of this scale, we rely on a synthetic dataset constructed from our original leave-one-out sample analyzed in Column 1 of Table 2, i.e., the pooled Veneto sample comprised of wage observations from the years 1999 and 2001. We scale the data by creating replicas of this base sample. To connect the replicas, we draw at random 10% of the movers and randomly exchange their period 1 firm assignments across replicas. By construction, this permutation maintains each (replicated) firm's size while ensuring leave-one-out connectedness of the resulting network.

Wage observations are drawn from a variant of the DGP described in Section 9.5 adapted to the levels formulation of the model. Specifically, each worker's wage is the sum of a rescaled person effect, a rescaled firm effect, and an error drawn independently in each period from a normal with variance $\frac{1}{2}\exp(\hat{a}_0 + \hat{a}_1 B_{gg} + \hat{a}_2 P_{gg} + \hat{a}_3 \ln L_{g2} + \hat{a}_4 \ln L_{g1})$. As highlighted by Figure 2.1, computing the exact estimator in these datasets would be extremely costly. Drawing from a stable DGP allows us to instead benchmark the JLA estimator against the true value of the variance of firm effects.

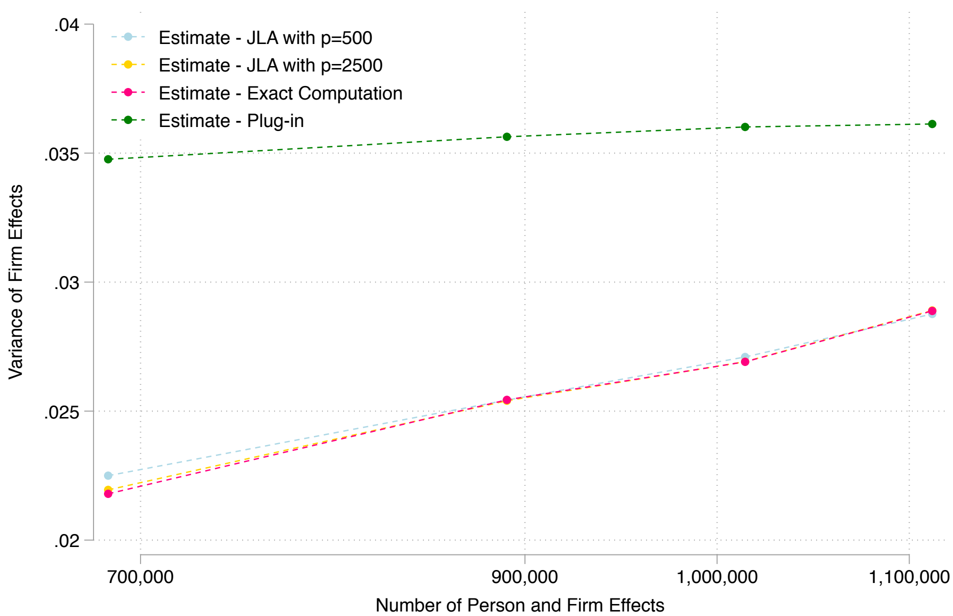
Figure 2.2 displays the results. When setting $p = 250$, the JLA delivers a variance of firm effects remarkably close to the true variance of firm effects defined by our DGP. As expected, the distance between our approximation and the true variance component decreases with the sample size for a fixed p. Remarkably, we are able to compute the AKM variance decomposition in a dataset with approximately 15 million person and year effects in only 35 minutes. Increasing the number of simulated draws in the JLA to $p = 500$ delivers estimates of the variance of firm effects nearly indistinguishable from the true value. This is achieved in approximately one hour in the largest simulated dataset considered. The results of this exercise strongly suggest the leave-out estimator can be scaled to extremely large datasets involving the universe of administrative wage records in large countries such as Germany or the United States.

Figure 2.1: Performance of the JLA Algorithm

(a) Computation Time



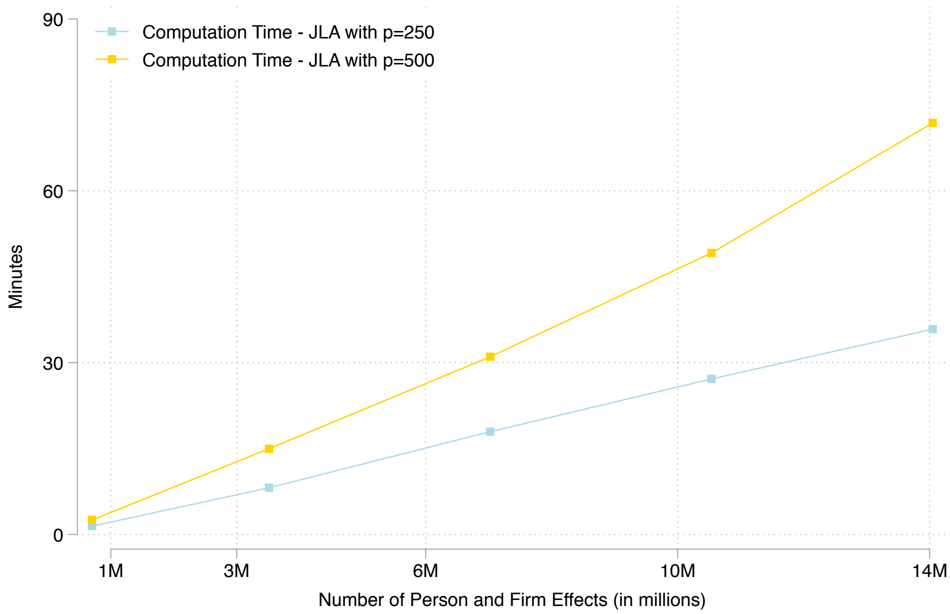
(b) Quality of the Approximation



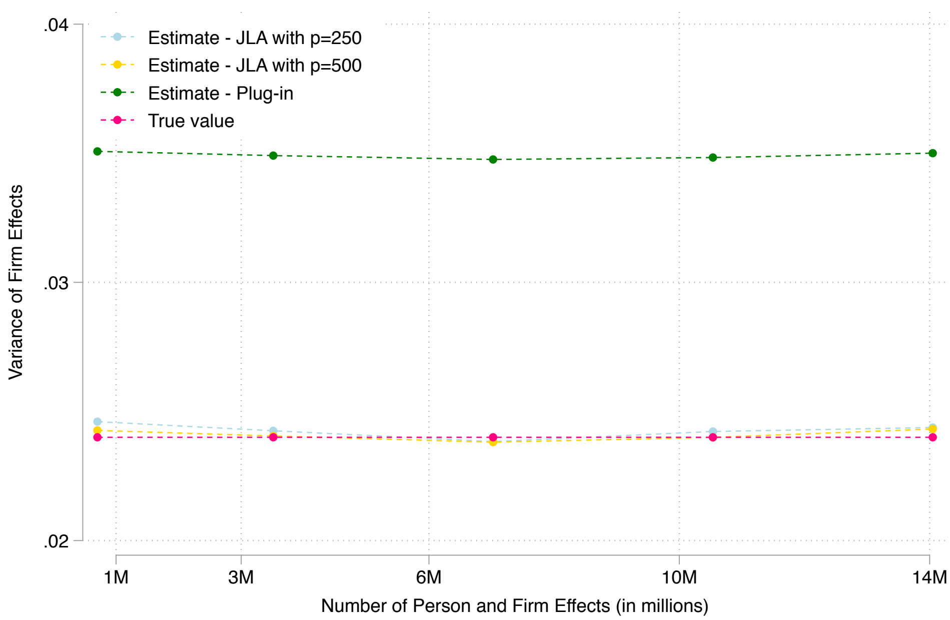
Note: Both panels consider 4 different samples of increasing length. The four samples contain data from the years 2000–2001, 1999–2001, 1998–2001, and 1997–2001, respectively. The x-axis reports the number of person and firm effects in each sample. Panel (a) shows the time to compute the KSS estimate when relying on either exact computation of $\{B_{ii}, P_{ii}\}_{i=1}^{n}$ or the Johnson-Lindenstrauss approximation (JLA) of these numbers using a p of either 500 or 2500. Panel (b) shows the resulting estimates and the plug-in estimate. Computations performed on a 32 core machine with 256 GB of dedicated memory. Source: VWH dataset.

Figure 2.2: Scaling to Very Large Datasets

(a) Computation Time



(b) Quality of the Approximation



Note: Both panels consider synthetic datasets created from the pooled Veneto data in column 1 of Table 2 with $T = 2$. It considers $\{1, 5, 10, 15, 20\}$ replicas of this sample while generating random links across replicas such that firm size and T are kept fixed. Outcomes are generated from a DGP of the sort considered in Table 5. The x-axis reports the number of person and firm effects in each sample. Panel (a) shows the time to compute the Johnson-Lindenstrauss approximation $\hat{\theta}_{JLA}$ using a p of either 250 or 500. Panel (b) shows the resulting estimates, the plug-in estimate, and the true value of the variance of firm effects for the DGP. Computations performed on a 32 core machine with 256 GB of dedicated memory. Source: VWH dataset.

2.4 Split Sample Estimators

Sections 5.2 and 6.2 proposed standard error estimators predicated on being able to construct independent split sample estimators $\widehat{x_i'\beta}_{-i,1}$ and $\widehat{x_i'\beta}_{-i,2}$. This section describes an algorithm for construction of these split sample estimators in the two-way fixed effects model of Example 3. We restrict attention to the case with $T_g = 2$ and consider the model in first differences: $\Delta y_g = \Delta f_g'\psi + \Delta\varepsilon_g$ for $g = 1, \ldots, N$. When worker g moves from firm j to j', we can estimate $\Delta f_g'\psi = \psi_{j'} - \psi_j$ without bias using OLS on any sub-sample where firms j and j' are connected, i.e., on any sample where there exist a path between firm j and j'. To construct two disjoint sub-samples where firms j and j' are connected we therefore use an algorithm to find disjoint paths between these firms and distribute them into two sub-samples which will be denoted $\mathcal{S}_1$ and $\mathcal{S}_2$. Because it can be computationally prohibitive to characterize all possible paths, we use a version of Dijkstra's algorithm to find many short paths.[1]

Our algorithm is based on a network where firms are vertices and two firms are connected by an edge if one or more workers moved between them. This view of the network is the same as the one taken in Section 8, but different from the one used in Sections 2.1 and 2.2 where both firms and workers were viewed as vertices. We use the adjacency matrix $\mathcal{A}$ to characterize the network in this section. To build the sub-samples $\mathcal{S}_1$ and $\mathcal{S}_2$, the algorithm successively drops workers from the network, so $\mathcal{A}_{-\mathcal{S}}$ will denote the adjacency matrix after dropping all workers in the set $\mathcal{S}$.

Given a network characterized by $\mathcal{A}$ and two connected firms j and j' in the network, we let $\dot{P}_{jj'}(\mathcal{A})$ denote the shortest path between them.[2] If j and j' are not connected $\dot{P}_{jj'}(\mathcal{A})$ is empty. Each edge in the path $\dot{P}_{jj'}(\mathcal{A})$ may have more than one worker associated with it. For each edge in $\dot{P}_{jj'}(\mathcal{A})$ the first step of the algorithm picks at random a single worker associated with that edge and places them in $\mathcal{S}_1$, while later steps place all workers associated with the shortest path in one of $\mathcal{S}_1$ and $\mathcal{S}_2$. This special first step ensures that the algorithm finds two independent unbiased estimators of $\Delta f_g'\psi$ whenever the network $\mathcal{A}$ is leave-two-out

[1] The algorithm presented below keeps running until it cannot find any additional paths. In our empirical implementation we stop the algorithm when it fails to find any new paths or as soon as one of the two sub-samples reach a size of at least 100 workers. We found that increasing this cap on the sub-sample size has virtually no effect on the estimated confidence intervals, but tends to increase computation time substantially.

[2] Many statistical software packages provide functions that can find shortest paths. In R they are available in the *igraph* package while in MATLAB a package that builds on the work of Yen (1971) is available at `https://www.mathworks.com/matlabcentral/fileexchange/35397-k-shortest-paths-in-a-graph-represented-by-a-sparse-matrix-yen-s-algorithm?focused=3779015&tab=function`.

connected.

For a given worker g with firm assignments $j = j(g,1), j' = j(g,2)$ and a leave-two-out connected network $\mathcal{A}$ the algorithm returns the $\{P_{g\ell,1}, P_{g\ell,2}\}_{\ell=1}^{N}$ introduced in Section 5.2. Specifically, $\widehat{\Delta f_g'\psi}_{-g,1} = \sum_{\ell=1}^{N} P_{g\ell,1}\Delta y_\ell$ and $\widehat{\Delta f_g'\psi}_{-g,2} = \sum_{\ell=1}^{N} P_{g\ell,2}\Delta y_\ell$ are independent unbiased estimators of $\Delta f_g'\psi$ that are also independent of Δy_g. If $\mathcal{A}$ is only leave-one-out connected then the algorithm may only find one path connecting j and j'. When this happens the algorithm sets $P_{g\ell,2} = 0$ for all ℓ as required in the formulation of the conservative standard errors proposed in Kline et al. (2020).

Algorithm 4 Split Sample Estimator for Inference

1: **function** SPLITSAMPLEESTIMATOR$(g, j, j', \mathcal{A})$ ▷ $\mathcal{A} \equiv$ Leave-one-out connected network
2: Let $\mathcal{S}_1 = \emptyset$ and $\mathcal{S}_2 = \emptyset$
3: For each edge in $\dot{P}_{jj'}(\mathcal{A}_{-g})$, pick at random one worker from $\mathcal{A}_{-g}$ who is associated with that edge and add that worker to $\mathcal{S}_1$
4: Add to $\mathcal{S}_2$ all workers from $\mathcal{A}_{-\{g,\mathcal{S}_1\}}$ who are associated with an edge in $\dot{P}_{jj'}(\mathcal{A}_{-\{g,\mathcal{S}_1\}})$
5: Add to $\mathcal{S}_1$ all workers from $\mathcal{A}_{-\{g,\mathcal{S}_1,\mathcal{S}_2\}}$ who are associated with an edge in $\dot{P}_{jj'}(\mathcal{A}_{-g})$
6: Let *stop* $= 1\{\dot{P}_{jj'}(\mathcal{A}_{-\{g,\mathcal{S}_1,\mathcal{S}_2\}}) = \emptyset\}$ and $s = 1$
7: **while** *stop* < 1 **do**
8: Add to $\mathcal{S}_s$ all workers from $\mathcal{A}_{-\{g,\mathcal{S}_1,\mathcal{S}_2\}}$ who are associated with an edge in $\dot{P}_{jj'}(\mathcal{A}_{-\{g,\mathcal{S}_1,\mathcal{S}_2\}})$
9: Let *stop* $= 1\{\dot{P}_{jj'}(\mathcal{A}_{-\{g,\mathcal{S}_1,\mathcal{S}_2\}}) = \emptyset\}$ and update s to $1 + 1\{s = 1\}$
10: **end while**
11: For $s = 1, 2$ and $\ell = 1, \ldots, N$, let $P_{g\ell,s} = 1\{\ell \in \mathcal{S}_s\}\Delta f_\ell'(\sum_{m\in\mathcal{S}_s} \Delta f_m \Delta f_m')^\dagger \Delta f_g$
12: Return $\{P_{g\ell,1}, P_{g\ell,2}\}_{\ell=1}^{N}$
13: **end function**

In line 5, all workers associated with the shortest path in line 3 are added to $\mathcal{S}_1$ if they were not added to $\mathcal{S}_2$ in line 4. This step ensures that all workers associated with $\dot{P}_{jj'}(\mathcal{A}_{-g})$ are used in the predictions. In line 11, $P_{g\ell,s}$ is constructed as the weight observation ℓ receives in the prediction $\Delta f_g'\hat{\psi}_s$ where $\hat{\psi}_s$ is the OLS estimator of ψ based on the sub-sample $\mathcal{S}_s$.

2.5 Test of Equal Firm Effects

This section describes computation and interpretation of the test of the hypothesis that firm effects for "younger" workers are equal to firm effects for the "older" workers which applies

Remark 6 of the main text.

The hypothesis of interest corresponds to a restricted and unrestricted model which when written in matrix notation are

$$\Delta y = \Delta F \psi + \Delta \varepsilon \tag{1}$$

$$\Delta y = \Delta F_O \psi^O + \Delta F_Y \psi^Y + \Delta F_3 \psi_3 + \Delta \varepsilon = X\beta + \Delta \varepsilon \tag{2}$$

where Δy and ΔF collects the first differences Δy_g and Δf_g across g. ΔF_O represents ΔF for "doubly connected" firms present in each age group's leave-one-out connected set interacted with a dummy for whether the worker is "old"; ΔF_Y represents ΔF for doubly connected firms interacted with a dummy for young; ΔF_3 represents ΔF for firms that are associated with either younger movers or older movers but not both. Finally, we let $X = (\Delta F_O, \Delta F_Y, \Delta F_3)$, $\beta = (\psi^{O\prime}, \psi^{Y\prime}, \psi_3')'$, and $\psi = (\psi^{O\prime}, \psi_3')'$.

The hypothesis in question is $\psi^O - \psi^Y = 0$ or equivalently $R\beta = 0$ for $R = [I_r, -I_r, 0]$ and $r = |\mathcal{J}| = \dim(\psi^O)$. Thus we can create the numerator of our test statistic by applying Remark 6 to (2) yielding

$$\hat{\theta} = \hat{\beta}' A \hat{\beta} - \sum_{g=1}^{N} B_{gg} \hat{\sigma}_g^2 \tag{3}$$

where $A = \frac{1}{r} R'(R S_{xx}^{-1} R')^{-1} R$; B_{gg} and $\hat{\sigma}_g^2$ are defined as in Section 1.

Two insights help to simplify computation. First, since $\Delta F_O' \Delta F_Y = 0$, $\Delta F_O' \Delta F_3 = 0$ and $\Delta F_Y' \Delta F_3 = 0$, we can estimate equation (2) via two separate regressions, one on the leave-one-out connected set for younger workers and the other on the leave-one-out connected set for older workers. We normalize the firm effects so that the same firm is dropped in both leave-one-out samples.

Second, we note that $\hat{\beta}' A \hat{\beta} = y' B y$ where

$$B = X S_{xx}^{-1} A S_{xx}^{-1} X' = \frac{P_X - P_{\Delta F}}{r}, \tag{4}$$

$P_X = X S_{xx}^{-1} X'$, and $P_{\Delta F} = \Delta F (\Delta F' \Delta F)^{-1} \Delta F'$. Equation (4) therefore implies that B_{ii} in (3) is simply a scaled difference between two statistical leverages: the first one obtained in the unrestricted model (2), say $P_{X,gg}$, and the other on the restricted model of (1), say $P_{\Delta F, gg}$. Section 2.3 describes how to efficiently compute these statistical leverages. To conduct inference on the quadratic form in (3) we apply the routine described in Section 5.2.

References

Bonhomme, S. (2017). Econometric analysis of bipartite networks. *Econometric Analysis of Network data edited by B. Graham and A. De Paula*.

Card, D., F. Devicienti, and A. Maida (2014). Rent-sharing, holdup, and wages: Evidence from matched panel data. *The Review of Economic Studies 81*(1), 84–111.

Card, D., J. Heining, and P. Kline (2013). Workplace heterogeneity and the rise of west german wage inequality. *The Quarterly journal of economics 128*(3), 967–1015.

Jochmans, K. and M. Weidner (2019). Fixed-effect regressions on network data. *Econometrica 87*(5), 1543–1560.

Kline, P., R. Saggio, and M. Sølvsten (2020). Supplement to "Leave-out estimation of variance components". *Econometrica Supplemental Material*.

Koutis, I., G. L. Miller, and D. Tolliver (2011). Combinatorial preconditioners and multilevel solvers for problems in computer vision and image processing. *Computer Vision and Image Understanding 115*(12), 1638–1646.

Serafinelli, M. (2019). "good" firms, worker flows, and local productivity. *Journal of Labor Economics 37*(3), 747–792.

Yen, J. Y. (1971). Finding the k shortest loopless paths in a network. *management Science 17*(11), 712–716.