Estimating Multi-Way Fixed Effect Models with `reghdfe`

Sergio Correia, Duke University

2016 Stata Conference, Chicago Illinois

sergio.correia@gmail.com
http://scorreia.com/software/reghdfe/
https://github.com/sergiocorreia/reghdfe
Introduction

`reghdfe` implements the estimator from:


Borrows heavily from previous contributions, many from the Stata camp (`reg2hdfe`, `a2reg`, `gpreg`)

Use it to control for unobservables that stay constant within an economic unit (workers, firms, exporters, importers, etc.)

Applications in many fields: accounting (DeHaan et al 2015), finance (Gormley et al 2015), labor (Guimarães et al 2015), trade (Mayer 2016), etc.
Estimator
We want to compute the least squares estimates $\hat{\beta}$ of

$$y = X\beta + D\alpha + \varepsilon$$

- $D = [D_1 \ D_2 \ \cdots \ D_F]$ consists of $F$ indicator matrices
- If $F = 1$, this collapses to a standard fixed effect regression ($xtreg$, $areg$)
- Can’t use dummies because $[D_2 \ \cdots \ D_F]$ is too large
Steps:

1. Compute the residuals of $\mathbf{y}$ and $\mathbf{X}$ against $\mathbf{D}$:

   $\tilde{\mathbf{y}} = \mathbf{M}_D \mathbf{y}$

   $\tilde{\mathbf{X}} = \mathbf{M}_D \mathbf{X}$

2. Apply the Frisch–Waugh–Lovell Theorem:

   $\hat{\mathbf{\beta}} = (\tilde{\mathbf{X}}' \tilde{\mathbf{X}})^{-1} \tilde{\mathbf{X}}' \tilde{\mathbf{y}}$

Thus, we can just focus on one variable at a time: $\tilde{\mathbf{y}}$
To obtain $\hat{y} = M_Dy$, find an $\hat{\alpha}$ that satisfies the normal equations

$$D'e = 0 \quad , \quad e \overset{\text{def}}{=} y - D\hat{\alpha}$$

In plain English:

For every level $g$ of every fixed effect $f$ the mean of the residuals must be zero:

$$\bar{e}_i = 0 \quad , \quad i \in I(f, g)$$

Note: We don’t care if $\hat{\alpha}$ is unique
Outline of the Algorithm

1. Divide and conquer: apply FWL to work on one variable at a time
2. Apply Method of Alternating Projections (MAP)
3. Accelerate MAP with conjugate gradient
4. Insights from graph theory: exactly the same problem as solving a Graph Laplacian
$$\lim_{n \to \infty} \left\| \left( M_1 \cdot M_2 \ldots M_F \right)^n y - M_{12\ldots F} y \right\| = 0$$

Suggests iteration:

$$y_{k+1} = \left( M_1 \cdot M_2 \ldots M_F \right) y_k$$
sysuse auto, clear

// Benchmark
areg price gear length i.trunk, absorb(turn)
foreach var in price gear length { // FWL Step
    forval i = 1/10 { // MAP Step
        foreach fe in turn trunk {
            qui areg `var', absorb(`fe')
            predict double resid, resid
            drop `var'
            rename resid `var'
        }
    }
}

regress price gear length, dof(38) nocons
Bauschke et al (2003):

[...] The main practical drawback of the MAP appears to be that it is often slowly convergent [...] Franchetti and Light and Bauschke, Borwein, and Lewis have given examples showing that the convergence [...] can be arbitrarily slow!

It can be very, very slow!

(In particular when the underlying fixed effects are poorly connected)
<table>
<thead>
<tr>
<th></th>
<th>y</th>
<th>id1</th>
<th>id2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>8</td>
<td>0</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>9</td>
<td>0</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>10</td>
<td>0</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

**Figure 1:** This dataset will turn your PC into a heater in the winter
Guimarães & Portugal (2010) and Gaure (2013) apply accelerations that are related to steepest descent

\[ y_{k+1} = t \left( M_1 \cdot M_2 \ldots M_F \right) y_k + (1 - t) y_k \]

Often improve speeds significantly, but ...
Bauschke et al (2003):

[...] perhaps surprisingly, we show that the acceleration scheme may actually be slower than the MAP [...]!


[...] the steepest descent method is known for its slowness in the presence of ill-conditioned problems [...]
• Why apply steepest descent and not conjugate gradient?
• Because CG requires a symmetric transform and
  \( T \overset{\text{def}}{=} M_1 \cdot M_2 \ldots M_F \) is not symmetric
• Solution: follow Hernández-Ramos et al (2011) and make it symmetric:

  \[
  T^{\text{Sym}} \overset{\text{def}}{=} M_1 \cdot M_2 \ldots M_F \ldots M_2 \cdot M_1
  \]

  \[
  T^{\text{Cim}} \overset{\text{def}}{=} (M_1 \cdot M_2 \ldots M_F) / F
  \]

• Theoretical advantages (monotonic convergence) and practical ones (as fast as other methods for easy problems, significantly faster for ill-defined ones)
Not fast enough for some applications, can we speed it even more? Yes!
Let’s rewrite the two–way fixed effect model as a graph:

If CEO $j$ has only worked at firm $k$:

$$\sum_{i \in j} y_i - n_j \hat{\alpha}_j - n_j \hat{\gamma}_k = 0$$

**Figure 2:** Graph of CEO–Firm Connections
Link with Graph Theory

- Solving a two-way fixed effects problem is exactly the same problem as solving $\mathbf{Lx} = \mathbf{b}$ where $\mathbf{L}$ is a Laplacian matrix.
  - Laplacian systems can now be solved in nearly-linear time, instead of in $O(n^{2.36})$!
  - This is a fundamental breakthrough in graph theory and numerical optimization, and we can apply it to solve our model.
- Can also apply other insights from graph theory (e.g. graph condition number).
However:

- Solver has a very complex implementation
- Suffers from cache locality problems (Hoske et al 2015, Boman et al 2016)
- What’s the point of an $O(n)$ solver if Stata requires multiple sorts? $O(n \log n)$
- Solution: use a better sorting algorithm (see ftools package)
Implementation
sysuse auto
ssc install reghdfe
reghdfe price weight, absorb(turn trunk foreign)
. reghdfe price weight, absorb(turn trunk foreign)
(dropped 9 singleton observations)
(converged in 13 iterations)

HDFE Linear regression
Absorbing 3 HDFE groups

|        | Coef. | Std. Err. | t     | P>|t| | [95% Conf. Interval] |
|--------|-------|-----------|-------|-----|---------------------|
| price  |       |           |       |     |                     |
| weight | 4.97842 | 0.9872606 | 5.04  | 0.000 | 2.980037 - 6.977246 |

Absorbed

Absorbed degrees of freedom:

<table>
<thead>
<tr>
<th>Absorbed FE</th>
<th>Num. Coefs.</th>
<th>Categories</th>
<th>Redundant</th>
</tr>
</thead>
<tbody>
<tr>
<td>turn</td>
<td>13</td>
<td>13</td>
<td>0</td>
</tr>
<tr>
<td>trunk</td>
<td>12</td>
<td>13</td>
<td>1</td>
</tr>
<tr>
<td>foreign</td>
<td>1</td>
<td>2</td>
<td>1 ?</td>
</tr>
</tbody>
</table>

? = number of redundant parameters may be higher

Figure 3: reghdfe screenshot
Design Principles: Simplicity

a2reg price gear, individual(turn) unit(foreign) indeffect(FE1) uniteffect(FE2)

reg2hdfe price gear, id1(turn) id2(trunk) fe1(FE1) fe2(FE2) uniteffect(FE2)

gpreg price gear, ivar(turn) jvar(trunk) ife(FE1) jfe(FE2)

felsdvregdm price gear, ivar(turn) jvar(trunk) peff(FE1) feff(FE2)

These are wonderful packages, but can we do better? (See The Zen of Python, Python for Humans, etc.)
Design Principles: Simplicity

reghdfe price gear, a(turn trunk, save)
IV Regressions:

\[
\text{reghdfe price (gear=length), a(turn trunk)}
\]

Multi–way clustering:

\[
\text{reghdfe price gear, a(turn trunk)} \\
\text{vce(cluster turn foreign)}
\]

Additional VCE methods:

\[
\text{reghdfe price gear, a(turn t) vce(cluster turn t, bw(2) kernel(parzen))}
\]
Design Principles: Powerful Under the Hood

Supports most standard Stata features:

```
reghdfe L.price i.foreign [aw=length],
a(turn trunk)
```

Heterogeneous slopes:

```
reghdfe price weight, a(turn##c.gear)
reghdfe price weight, a(turn##c.(gear
length) trunk)
```
Design Principles: Powerful Under the Hood

Save users’ time:

```
reghdfe price gear, absorb(turn#trunk)
cluster(turn#foreign)
```

Also: implemented in heavily optimized Mata code (**reghdfe** is **faster** than **areg** and **xtreg** even for one set of fixed effects!)
Design Principles: Don’t Reinvent the Wheel

Most features come from the Stata community: see `reghdfe, version`

- `ivreg2` or `ivregress` for IV/GMM models
- `avar` for VCE estimation
- `tuples` for MWC
- `group3hdfe` to compute degrees–of–freedom
- Learned a lot from `reg2hdfe`, `a2reg`, etc.
- Supports `esttab`: `viewsource estfe.ado`
Same principle behind use ..., clear

Warn about several gotchas:

• Drop singleton groups, which might affect VCE estimates
• Compute conservative degrees-of-freedom
• Present alternatives to overall R2, which might be misleading
• Fixed effects are not identified; researchers are using it incorrectly; alternatives?
• Can we provide better VCE estimates? (e.g. Cattaneo et al 2016)
• What if every obs. has a varying number of fixed effects? (board of directors)
Improvements and Extensions (2)

- **lsmr** estimator from Matthieu Gomez
- **ftools** allows significant speedups in Stata with large datasets (based on optimizations by Python’s Pandas)
- Publicize collected benchmark datasets
Also see

- Detailed manual
- Github bug tracker
Thank you!