Explanation of the HDFE iteration with 3 FEs

Sergio Correia

July 30, 2014

1 Background

Our objective is to obtain the OLS estimates for β in the regression $y = X\beta + D_1\alpha_1 + D_2\alpha_2 + D_3\alpha_3 + \epsilon$, where D_i represents a set of indicator or design variables, and α_i is the set of associated fixed effects.

Let M, P be the usual annihilator and projection matrices in an OLS regression. When the regressors are dummy variables, P and M are just averages and demeaned values within each group respectively. Also, from the normal equation we know the projection matrix is orthogonal to the residuals: $Pe = 0$.

Recall the two-part FWL theorem. Its first part states that we can regress $y = X\beta + D_1\alpha_1 + D_2\alpha_2 + D_3\alpha_3 + \epsilon$ by first regressing y and each X against the set D_1, D_2, D_3, and then just regressing the residuals of y against the residuals of X. The second part states that the residuals of both regressions are also identical.

Thus, the strategy will be to apply FWL to obtain residuals of (y, X) with respect to all the indicator variables, and then regress the residuals to obtain estimates for β.

Notice that estimates for each α can still be recovered, but they may not be identified.

2 Fixed Point Iteration

To regress against dummies there are two usual approaches: (i) add the dummies explictly or (ii) demean all the variables. Option (ii) is much faster but can only be done *once*, so if we have two or more sets of DVs the usual approach is to apply (ii) to the fixed effect with more dimensions (i.e. categories) and (i) to all the others. However, if there are two or more sets of high-dimensional DVs, we would have to create a lot of dummies which is impractical (b/c of out of memory errors, etc.).

Instead, we'll apply a fixed-point iteration over the normal equations, as suggested by Guimaraes & Portugal (2010) (i.e., guess a value, and iterate our estimates until the normal equations hold). Note that instead of iterating the estimates $\hat{\alpha}_1, \hat{\alpha}_2, \hat{\alpha}_3$, we will iterate $Z_2 := D_2\hat{\alpha}_2$ and $Z_3 := D_3\hat{\alpha}_3$. Also, note that we are not employing Guimaraes et al (2013) version for three sets of FEs, but an extension of their original work, which works significatively faster.

3 Algorithm

Wlog, start with y,

1. Compute $P_1 y$ and $\tilde{y} := M_1 y$

2. Initialize $Z_2^{(0)} = Z_3^{(0)} = 0$

3. Until Z_2 and Z_3 converge, set

   (a) $Z_2^{(n)} = P_2 \left[\tilde{y} + P_1 \left(Z_2^{(n-1)} + Z_3^{(n-1)} \right) - Z_3^{(n-1)} \right]$

   (b) $Z_3^{(n)} = P_3 \left[\tilde{y} + P_1 \left(Z_2^{(n)} + Z_3^{(n-1)} \right) - Z_2^{(n)} \right]$

4. Then, compute $Z_1 = P_1(y - Z_2 - Z_3)$ and with that compute $y^* = y - Z_1 - Z_2 - Z_3$, which is what we desire.

After repeating this for every variable, regress the transformed variables $(y^* = X^* \hat{\beta} + e)$ to obtain the correct estimates $\hat{\beta}$. Notice that the residuals of this FWL regression are the same as those of the full regression.

If you want to obtain the fixed effects for the regression, notice that $y = X\hat{\beta} + Z_1 + Z_2 + Z_3 + e$ where both $\hat{\beta}$ and e are the same as in the transformed regression. Thus, compute $\tilde{e} := y - X\hat{\beta} = e + Z_1 + Z_2 + Z_3$, and then apply the previous step to obtain the Z_1, Z_2, Z_3 fixed effects for that variable.

Notice that if we remove the third FE (i.e. set $Z_3 = 0$), the formula collapses to the usual two-FE version.

4 Deriving the fixed point iteration with three fixed-effects

1. Start with the identity $y = D_1 \hat{\alpha}_1 + D_2 \hat{\alpha}_2 + D_3 \hat{\alpha}_3 + e = Z_1 + Z_2 + Z_3 + e$, where $Z_i := D_i \hat{\alpha}_i$

2. Premultiply by M_1 to obtain $\tilde{y} = M_1 Z_2 + M_1 Z_3 + e$, since $M_1 D_1 = D_1 - P_1 D_1 = D_1 - D_1 = 0$ and $M_1 e = e - P_1 e = 0$ (from the normal equation).

3. Rearrange $Z_2 + Z_3 = \tilde{y} + P_1 (Z_2 + Z_3) - e$

4. Premultiply by P_2 and notice that $P_2 e = 0$, and after rearranging, obtain $Z_2 = P_2 [\tilde{y} + P_1 (Z_2 + Z_3) - Z_3]$

5. Premultiply by P_3 and obtain the equivalent equation for the third fixed effect.

With these formulas, and adding a little more structure (to show that the fixed point is attractive, etc.), we can turn the formulas above into a fixed point iteration that will converge linearly.

5 Implementation details

This fixed point iteration is very slow, but there are (at least) two venues for speeding it up.

1. We can minimize the number of P operations and speed them up. First, notice that sorting the dataset (an $O(n \log(n))$ operation) is not needed to obtain group means. Also, since in this case P just demeans a variable within a group, we can store the output of P_i as a $G_1 \times 1$ vector, instead of a $N \times 1$ vector. Finally, we can try precompute intermediate steps such as $P_2 \tilde{y}$ (which we only compute once) and $P_1 Z_2$ (which we only compute once per iteration, instead of twice).

2. We can try to extrapolate the fixed-point iteration so it converges faster. These methods are called acceleration tecniques, and of those, we use a vector-based version of Aitken's Δ^2 method. In particular, we are employing method 3 as described by Macleod (1986), which has a reasonable performance, and accelerating every three iterations.

All in all, each iteration requires only 6 P operations (2 in the case of just two fixed effects), which are implemented reasonably fast. Together with the acceleration, this method becomes both fast and conservative in terms of memory usage.

6   Extension: Algorithm beyond three sets of fixed effects

The algorithm of part 3 can be extended to an arbitrarily large number of fixed effects. This will lead to the following steps:

Wlog, start with y, and let there be G sets of fixed effects.

1. Compute $P_1 y$ and $\tilde{y} := M_1 y$

2. Precompute $P_g \tilde{y} \quad \forall g > 1$

3. Initialize $Z_g^{(0)} = 0 \quad \forall g > 1$

4. Initialize $\Sigma^{(0)} := \sum_{g=2}^{G} Z_g^{(0)} = 0$

5. Until all Z_g converge for all $g > 1$ (this condition can also be replaced by a condition on Σ), set for $g = 2, 3, \ldots, G$:

    (a) $Z_g^{(n)} = Z_g^{(n-1)} + P_g \left[\tilde{y} + (P_1 - 1)\Sigma^{(n-1,g-1)} \right]$
    (b) $\Sigma^{(n-1,g)} = \Sigma^{(n-1,g)} + Z_g^{(n)} - Z_g^{(n-1)}$

6. Then, compute $Z_1 = P_1(y - \Sigma)$ and with that compute $y^*[= My = M_1(y - \Sigma)] = y - Z_1 - \Sigma$.

Notice that step 5 could be expressed more compactly just in terms of Σ, but that complicates obtaining the individual FEs.

7   Extension: Interactions between FEs and continuous variables

If we interact a variable V elementwise with each column of a design matrix D representing a set of K fixed effects, a few adjustments would be required.

First, note that it's equivalent to estimating a system of bivariate regressions. For instance, if the D matrix had three fixed effects (i.e. it's a $N \times 3$ matrix), then P_D could be written as the partitioned matrix:

$$\begin{pmatrix} P_1 & 0 & 0 \\ 0 & P_2 & 0 \\ 0 & 0 & P_3 \end{pmatrix}$$

And each P_j would be in turn be equal to (for the case of 3 obs. in the group j):

$$\begin{pmatrix} v_{11}^2 & v_1 v_2 & v_1 v_3 \\ v_2 v_1 & v_{22}^2 & v_2 v_3 \\ v_3 v_1 & v_3 v_2 & v_{33}^2 \end{pmatrix}$$

(subindices are relative to each group)

Then, we can show that $U := P_D y$ is such that $u_i = v_i \times q_{g(i)}$, where Q is a $K \times 1$ vector s.t.
$q_j = \frac{\sum_{i \text{ s.t. } D_{ij}=1} v_i y_i}{v_i^2}$.

In terms of code, before we obtained a Q matrix where $v_i = 1$, so we were basically computing group means ($q_j = \frac{\sum_{i \text{ s.t. } D_{ij}=1} y_i}{N_j}$) and now we are i) weighting them and ii) multiplying by v_i afterwards.

Some optimizations can be done, such as precomputing the denominators of Q, and doing scalar products of v and y so we can use the code of group means without many changes.

8 Extension: Instrumental Variables

Just demean *all* variables w.r.t the set of fixed effects. Then, we just run the first stage and second stage regressions taking care of adjusting the standard errors.