The Stata Journal (2010)
10, Number 4, pp. 628–649

A simple feasible procedure to fit models with high-dimensional fixed effects

Paulo Guimarães
University of South Carolina
Columbia, South Carolina
and Universidade do Porto
Porto, Portugal
guimaraes@moore.sc.edu

Pedro Portugal
Banco de Portugal and
Universidade Nova de Lisboa
Lisbon, Portugal
pportugal@bportugal.pt

Abstract. In this article, we describe an iterative approach for the estimation of linear regression models with high-dimensional fixed effects. This approach is computationally intensive but imposes minimum memory requirements. We also show that the approach can be extended to nonlinear models and to more than two high-dimensional fixed effects.

Keywords: st0212, fixed effects, panel data

1 Introduction

The increasing availability of large microlevel datasets has spurred interest in methods for estimation of models with high-dimensional fixed effects. Researchers in several fields such as economics, sociology, and political science, among others, find the introduction of fixed effects to be a particularly appealing way of controlling for unobserved heterogeneity that is shared among groups of observations. In this case, it becomes possible to account for all intergroup variability by adding to the set of regressors some dummy variables that absorb group-specific heterogeneity. This approach has the advantage of allowing for the existence of general patterns of correlation between the unobserved effects and the other regressors.

In practice, when fitting a model with a single fixed effect (that is, a factor in the analysis of covariance), one is not required to actually add the group dummy variables to the set of regressors. This is particularly convenient when dealing with high-dimensional fixed effects—that is, in a situation where the number of groups (dummy variables) is very large. For several common procedures such as linear regression, Poisson regression, and logit regression, the fixed effect can be eliminated from the model, thereby making it possible to obtain estimates for the coefficients of the relevant regressors without having to introduce the group dummy variables in the model. For other nonlinear models, it is still possible to avoid the explicit introduction of dummy variables to account for the fixed effect by modifying the iterative algorithm used to solve the maximum likelihood problem (see Greene [2004]).

However, there is no simple solution in situations that have more than one high-dimensional fixed effect. A notable example would be the large employer–employee

panel datasets commonly used in the labor economics literature. When studying relations in the labor market, researchers often want to simultaneously account for two sources of unobserved heterogeneity—the firm and the worker. Explicit introduction of dummy variables is not an option because the number of units (groups) for either firms or workers is too large. Other well-known examples of large datasets with obvious sources of unobserved heterogeneity include these two types of panel datasets: patient claims data, in which the potential sources of heterogeneity are the patient, the doctor, and the hospital; and student performance, in which potential sources of heterogeneity are the student, the teacher, and the school.

Abowd, Kramarz, and Margolis (1999) tackled the problem of accounting for two high-dimensional fixed effects in the linear regression problem. In a widely cited article, the authors proposed several methods that provide approximations to the least-square solution.[1] Later, in an unpublished article, Abowd, Creecy, and Kramarz (2002) presented an iterative algorithm that leads to the exact least-square solution of a linear regression model with two fixed effects. The user-written command `a2reg` is a Stata implementation of this algorithm by Amine Ouazad. In a recent article published in the *Stata Journal*, Cornelissen (2008) presented a new user-written command, `felsdvreg`, which consists of a memory-saving procedure for estimation of a linear regression model with two high-dimensional fixed effects.

At this point, we should make clear that the methods discussed above (as well as those discussed in this article) may not lead to consistent estimation. Unlike the case with most panel-data estimators, consistency is achieved if we are willing to admit that the dimension of the groups is unrelated to sample size. This means that, from an asymptotic point of view, the number of parameters of the model remains unchanged as the sample size tends to infinity. This assumption gets around the incidental parameter problem but more correctly places these estimators as extensions of Stata's `areg` command—that is, as alternative ways to fit models with large sets of dummy variables.

In our own work (Carneiro, Guimarães, and Portugal 2010), we were faced with the problem of fitting a linear regression model with two high-dimensional fixed effects (firm and worker) using a linked employer–employee dataset with over 30 million observations. Implementation of the user-written commands discussed above in a computer with 8 gigabytes of random-access memory (RAM) and running Stata/MP for Windows failed because of memory limitations. However, using an iterative procedure that was simple to implement, we were able to fit the linear regression model with two and even three high-dimensional fixed effects. The approach is computationally intensive, but it has the advantage of imposing minimal memory requirements. In this article, we present a detailed discussion of the method proposed in Carneiro, Guimarães, and Portugal (2010) and show how it can be extended to nonlinear models and to applications with more than two high-dimensional fixed effects.

1. For a discussion on the implementation of these methods in Stata, see Andrews, Schank, and Upward (2006).

2 The linear regression model

2.1 One fixed effect

To start with, consider the conventional linear regression model setup

$$y_i = \beta_1 x_{1i} + \beta_2 x_{2i} + \cdots + \beta_k x_{ki} + \epsilon_i$$

or, more compactly,

$$\mathbf{Y} = \mathbf{X}\beta + \epsilon$$

Application of the least-squares method results in the set of normal equations given below:

$$\left[\begin{array}{l} \frac{\partial SS}{\partial \beta_1} = \sum_i x_{1i}(y_i - \beta_1 x_{1i} - \beta_2 x_{2i} - \cdots - \beta_k x_{ki}) = 0 \\ \frac{\partial SS}{\partial \beta_2} = \sum_i x_{2i}(y_i - \beta_1 x_{1i} - \beta_2 x_{2i} - \cdots - \beta_k x_{ki}) = 0 \\ \qquad\qquad \cdots \\ \frac{\partial SS}{\partial \beta_k} = \sum_i x_{ki}(y_i - \beta_1 x_{1i} - \beta_2 x_{2i} - \cdots - \beta_k x_{ki}) = 0 \end{array}\right] \tag{1}$$

These equations have a closed-form solution, the least-squares estimator, given by the well-known formula

$$\widehat{\beta} = \left(\mathbf{X}'\mathbf{X}\right)^{-1}\mathbf{X}'\mathbf{Y}$$

However, the above formula is one of several alternative ways to find the solution to (1). For instance, we can solve for $\widehat{\beta}$ using a partitioned iterative algorithm. An example of such an algorithm is shown below:

- Initialize $\beta_1^{(0)}, \beta_2^{(0)}, \ldots, \beta_k^{(0)}$
- Solve for $\beta_1^{(1)}$ as the solution to $\frac{\partial SS}{\partial \beta_1} = \sum_i x_{1i}(y_i - \beta_1 x_{1i} - \beta_2^{(0)} x_{2i} - \cdots - \beta_k^{(0)} x_{ki}) = 0$
- Solve for $\beta_2^{(1)}$ as the solution to $\frac{\partial SS}{\partial \beta_2} = \sum_i x_{2i}(y_i - \beta_1^{(1)} x_{1i} - \beta_2 x_{2i} - \cdots - \beta_k^{(0)} x_{ki}) = 0$
- ...
- Repeat until convergence is achieved.

Algorithms such as this one are discussed in Smyth (1996). This algorithm is known as the "zigzag" or full Gauss–Seidel algorithm. According to Smyth, this algorithm produces a stable but slow iteration depending on the correlation between the parameter estimators. In this particular case, use of an iterative algorithm to solve the normal equations is highly inefficient. However, this implementation has the advantage of not requiring the explicit calculation of the inverse of the $\mathbf{X}'\mathbf{X}$ matrix.

Consider now what happens if we include a set of dummy variables to account for a fixed effect in the regression. In that case,

$$\mathbf{Y} = \mathbf{Z}\beta + \mathbf{D}\alpha + \epsilon$$

where $\mathbf{Z}$ is the matrix of explanatory variables with $N \times k$ dimension and $\mathbf{D}$ is the $N \times G_1$ matrix of dummy variables. Now we can write the normal equations as

$$\begin{bmatrix} \mathbf{Z}'\mathbf{Z} & \mathbf{Z}'\mathbf{D} \\ \mathbf{D}'\mathbf{Z} & \mathbf{D}'\mathbf{D} \end{bmatrix} \begin{bmatrix} \beta \\ \alpha \end{bmatrix} = \begin{bmatrix} \mathbf{Z}'\mathbf{Y} \\ \mathbf{D}'\mathbf{Y} \end{bmatrix}$$

which can be arranged to show

$$\begin{bmatrix} \mathbf{Z}'\mathbf{Z}\beta + \mathbf{Z}'\mathbf{D}\alpha = \mathbf{Z}'\mathbf{Y} \\ \mathbf{D}'\mathbf{Z}\beta + \mathbf{D}'\mathbf{D}\alpha = \mathbf{D}'\mathbf{Y} \end{bmatrix}$$

Solving each set of equations independently yields

$$\begin{bmatrix} \beta = \left(\mathbf{Z}'\mathbf{Z}\right)^{-1}\mathbf{Z}'\left(\mathbf{Y} - \mathbf{D}\alpha\right) \\ \alpha = \left(\mathbf{D}'\mathbf{D}\right)^{-1}\mathbf{D}'\left(\mathbf{Y} - \mathbf{Z}\beta\right) \end{bmatrix}$$

The above partition of the normal equations suggests a convenient iteration strategy. To obtain the exact least-squares solution, one can simply alternate between estimation of β and estimation of α. It is important to mention that we no longer have to worry about the dimensionality of $\mathbf{D}$. The expression $\left(\mathbf{D}'\mathbf{D}\right)^{-1}\mathbf{D}'$ used on the estimation of α translates into a simple group average of the residuals of the regression of $\mathbf{Y}$ on $\mathbf{Z}$. On the other hand, the expression $\mathbf{D}\alpha$ that shows up on the equation for β is a column vector containing all the elements of α. Estimation of β consists of a simple linear regression of a transformed $\mathbf{Y}$ on $\mathbf{Z}$. In our implementation, instead of transforming $\mathbf{Y}$, we will keep $\mathbf{Y}$ as the dependent variable and add $\mathbf{D}\alpha$ as an additional covariate. When the estimation procedure converges, the coefficient on $\mathbf{D}\alpha$ must equal one and the vector $\mathbf{D}\alpha$ will contain all the estimated coefficients for the group dummy variables. With this approach, we avoided inversion of a potentially large matrix that would be required if we had simply added $\mathbf{D}$ to the set of regressors. As an illustration of this approach, we use `auto.dta` and replicate the coefficient estimates obtained with `areg` as shown in example 1 of [R] **areg**.

```
. sysuse auto
(1978 Automobile Data)
. keep if rep78 < .
(5 observations deleted)
. generate double fe1=0
. local rss1=0
. local dif=1
. local i=0
```

```
. while abs(`dif´)>epsdouble() {
  2. quietly {
  3. regress mpg weight gear_ratio fe1
  4. local rss2=`rss1´
  5. local rss1=e(rss)
  6. local dif=`rss2´-`rss1´
  7. capture drop yhat
  8. predict double yhat
  9. generate double temp1=mpg-yhat+_b[fe1]*fe1
 10. capture drop fe1
 11. egen double fe1=mean(temp1), by(rep78)
 12. capture drop temp1
 13. local i=`i´+1
 14. }
 15. }
. display "Total Number of Iterations --> " `i´
Total Number of Iterations --> 13
. quietly regress mpg weight gear_ratio fe1
. estimates table, b(%10.7f)
```

Variable	active
weight	-0.0051031
gear_ratio	0.9014780
fe1	1.0000000
_cons	34.0588892

As we implied earlier, the estimated coefficients are identical to those obtained using the `areg` command (see [R] **areg**). The final regression includes an additional variable, `fe1`, with a coefficient of one. This variable was created during estimation and contains the estimates of the fixed effect.

2.2 More than one fixed effect

Suppose now that instead of a single high-dimensional fixed effect, we have two high-dimensional fixed effects. That is, we now intend to fit the following model:

$$\mathbf{Y} = \mathbf{Z}\beta + \mathbf{D}_1\alpha + \mathbf{D}_2\gamma + \epsilon$$

where $\mathbf{D}_1$ is $N \times G_1$, $\mathbf{D}_2$ is $N \times G_2$, and both G_1 and G_2 have high dimensionality. As discussed earlier, in this particular case, estimation of the linear regression model is complicated. However, implementation of the partitioned algorithm discussed above is straightforward. Proceeding as we did before, we can solve the normal equations as

$$\left[\begin{array}{c} \beta = \left(\mathbf{Z}'\mathbf{Z}\right)^{-1}\mathbf{Z}'\left(\mathbf{Y} - \mathbf{D}_1\alpha - \mathbf{D}_2\gamma\right) \\ \alpha = \left(\mathbf{D}_1'\mathbf{D}_1\right)^{-1}\mathbf{D}_1'\left(\mathbf{Y} - \mathbf{Z}\beta - \mathbf{D}_2\gamma\right) \\ \gamma = \left(\mathbf{D}_2'\mathbf{D}_2\right)^{-1}\mathbf{D}_2'\left(\mathbf{Y} - \mathbf{Z}\beta - \mathbf{D}_1\alpha\right) \end{array}\right] \tag{2}$$

Iterating between these sets of equations provides us with the exact least-squares solution. All we have to do is compute several linear regressions with k explanatory variables

and compute group means of residuals. If we add more fixed effects, the logic remains unchanged. To illustrate the approach, we modify the above algorithm and apply it to the ancillary dataset that accompanies the `felsdvreg` command developed by Thomas Cornelissen. As before, we introduce $\mathbf{D}_1\alpha$ and $\mathbf{D}_2\gamma$ as additional regressors instead of subtracting them from the dependent variable.

```
. use felsdvsimul, clear
. generate double temp=0
. generate double fe1=0
. generate double fe2=0
. local rss1=0
. local dif=1
. local i=0
. while abs(`dif´)>epsdouble() {
  2. quietly {
  3. regress y x1 x2 fe1 fe2
  4. local rss2=`rss1´
  5. local rss1=e(rss)
  6. local dif=`rss2´-`rss1´
  7. capture drop res
  8. predict double res, res
  9. replace temp=res+_b[fe1]*fe1, nopromote
 10. capture drop fe1
 11. egen double fe1=mean(temp), by(i)
 12. replace temp=res+_b[fe2]*fe2, nopromote
 13. capture drop fe2
 14. egen double fe2=mean(temp), by(j)
 15. local i=`i´+1
 16. }
 17. }
. display "Total Number of Iterations --> " `i´
Total Number of Iterations --> 695
. quietly regress y x1 x2 fe1 fe2
. estimates table, b(%10.7f)
```

Variable	active
x1	1.0292584
x2	-0.7094820
fe1	1.0000000
fe2	1.0000000
_cons	-1.9951791

As we hinted above, the estimates for the model coefficients are identical to the least-squares results with all dummy variables included, as reported in Cornelissen (2008). The algorithm took 695 iterations to converge, which is one of the drawbacks of this approach. Fortunately, as discussed below, there is substantial room for improvement. One obvious simplification is to sweep out one of the fixed effects by subtracting the group mean from all variables. By doing this, we avoid dealing with one of the fixed effects. This means that with minor modifications, the code shown above can be used to fit a model with three high-dimensional fixed effects. We illustrate the estimation of

a model with three high-dimensional fixed effects by assuming that our single variable of interest is `x1` and that `x2` is a variable indicating an additional fixed effect. The modified code is shown below:

```
. use felsdvsimul, clear
. generate double temp=0
. generate double fe1=0
. generate double fe2=0
. generate double fe1t=0
. generate double fe2t=0
. egen double mean=mean(x1), by(x2)
. generate double x1t=x1-mean
. capture drop mean
. egen double mean=mean(y), by(x2)
. generate double yt=y-mean
. local rss1=0
. local dif=1
. local i=0
. while abs(`dif´)>epsdouble() {
  2. quietly {
  3. capture drop mean
  4. egen double mean=mean(fe1), by(x2)
  5. replace fe1t=fe1-mean, nopromote
  6. capture drop mean
  7. egen double mean=mean(fe2), by(x2)
  8. replace fe2t=fe2-mean, nopromote
  9. regress yt x1t fe1t fe2t
 10. local rss2=`rss1´
 11. local rss1=e(rss)
 12. local dif=`rss2´-`rss1´
 13. replace temp=yt-_b[x1t]*x1t+_b[fe1t]*(fe1-fe1t)-_b[fe2t]*fe2t, nopromote
 14. capture drop fe1
 15. egen double fe1=mean(temp), by(i)
 16. replace temp=yt-_b[x1t]*x1t-_b[fe1t]*fe1t+_b[fe2t]*(fe2-fe2t), nopromote
 17. capture drop fe2
 18. egen double fe2=mean(temp), by(j)
 19. local i=`i´+1
 20. }
 21. }
. display "Total Number of Iterations --> " `i´
Total Number of Iterations --> 477
. quietly areg y x1 fe1 fe2, absorb(x2)
. estimates table, b(%10.7f)
```

Variable	active
x1	0.9634370
fe1	1.0000000
fe2	1.0000000
_cons	-2.2295298

As we can readily see, the estimated coefficient for `x1` is the same as that obtained by the simple regression on `x1` and three sets of dummy variables.

```
. quietly xi: regress y x1 i.x2 i.i i.j
. estimates table, keep(x1) b(%10.7f)
```

Variable	active
x1	0.9634370

In Carneiro, Guimarães, and Portugal (2010), we estimated a conventional Mincerian wage equation with three high-dimensional fixed effects. Our data source is the *Quadros do Pessoal*, a mandatory employment survey collected yearly by the Portuguese Ministry of Labor and Social Security. The dataset comprised more than 30 million observations spanning from 1986 to 2007. In our estimation, we wanted to account for firm, worker, and job heterogeneity and year fixed effects. With around 6.4 million workers, 624,171 firms, and 115,822 jobs, employing dummy variables to account for the fixed effects was not an option. In the following example, though, we show the output result of one specification with 28 covariates and the three high-dimensional fixed effects estimated using the approach outlined above.

(*Continued on next page*)

```
********** Linear Regression with 3 High-Dimensional Fixed Effects **********

                                        Number of obs   =   30906573
                                        F(6703079,24203493)=   30.81
                                        Prob > F        =     0.0000
                                        R-squared       =     0.8951
                                        Adj R-squared   =     0.8660
                                        Root MSE        =     0.2159
```

ln_real_hw	Coef.	Std. Err.	t	P>|t|	[95% Conf.	Interval]
age	.0506563	.0038823	13.05	0.000	.043047	.0582655
agesq	-.0002992	6.36e-07	-470.60	0.000	-.0003005	-.000298
hab_1	-.0228571	.0011448	-19.97	0.000	-.0251008	-.0206134
hab_2	-.0202456	.0010558	-19.18	0.000	-.0223149	-.0181763
hab_3	-.0173233	.0010535	-16.44	0.000	-.0193881	-.0152585
hab_4	-.0068901	.0010517	-6.55	0.000	-.0089515	-.0048288
hab_5	.0047679	.00105	4.54	0.000	.0027099	.006826
hab_67	.0658466	.0011658	56.48	0.000	.0635618	.0681314
hab_8910	.1175194	.0011608	101.24	0.000	.1152443	.1197945
y87	.0139098	.0038957	3.57	0.000	.0062743	.0215453
y88	.0036002	.0077708	0.46	0.643	-.0116303	.0188307
y89	.108814	.0116502	9.34	0.000	.0859801	.131648
y91	.0802597	.0194122	4.13	0.000	.0422125	.1183068
y92	.0748048	.0232936	3.21	0.001	.0291501	.1204595
y93	.042098	.0271755	1.55	0.121	-.0111649	.0953609
y94	.0060495	.0310574	0.19	0.846	-.0548219	.0669209
y95	-.0086197	.034939	-0.25	0.805	-.077099	.0598596
y96	-.0186946	.0388209	-0.48	0.630	-.0947821	.057393
y97	.0024842	.0427027	0.06	0.954	-.0812116	.08618
y98	.0294311	.0465845	0.63	0.528	-.0618729	.1207351
y99	.043531	.0504663	0.86	0.388	-.0553812	.1424433
y03	-.0054844	.0659942	-0.08	0.934	-.1348307	.123862
y00	.0412941	.0543482	0.76	0.447	-.0652264	.1478146
y02	.0171165	.0621123	0.28	0.783	-.1046213	.1388544
y04	.0296664	.0698753	0.42	0.671	-.1072867	.1666195
y05	.0194295	.0737572	0.26	0.792	-.1251318	.1639909
y06	-.0153513	.0776394	-0.20	0.843	-.1675216	.1368191
y07	-.0225893	.0815211	-0.28	0.782	-.1823678	.1371893

2.3 Estimation of the standard errors

To provide the standard errors associated with the estimator of β, we would need to estimate

$$V\left(\widehat{\beta}\right) = \sigma^2 \left(\mathbf{X}'\mathbf{X}\right)^{-1}$$

which again raises the problem of inverting the $\mathbf{X}'\mathbf{X}$ matrix. An alternative solution to estimate the elements of $V(\widehat{\beta})$ is to use the known relation

$$V\left(\widehat{\beta}_j\right) = \frac{\sigma^2}{N s_j^2 \left(1 - R^2_{j.123\ldots}\right)}$$

where s_j^2 is the sample variance associated with the x_j variable and $R^2_{j.123\ldots}$ is the coefficient of determination obtained from a regression of x_j on all other remaining

explanatory variables. Estimation of σ^2 is easy because the final regression that provides the estimates of β has the correct sum of squared residuals (SSR). As we will see, the remaining difficulty is the computation of the number of degrees of freedom associated with SSR. With multiple fixed effects, it may be difficult to compute the actual dimension of $\mathbf{X}$ because some of the coefficients for the fixed effects may not be identifiable. If we were simply estimating the model by adding dummy variables to account for the fixed effects, Stata would numerically identify colinearities and drop variables as needed to identify the coefficients in the model. However, to implement our solution, we need to know beforehand the number of coefficients of the dummy variables that can be identified.[2] Computation of $R^2_{j.123...}$ is not a problem because we know how to estimate a model with high-dimensional effects. However, this approach may be time consuming because it would require estimation of a regression with high-dimensional fixed effects for each of the regressors.

Fortunately, there is an alternative strategy that will produce results faster. The idea is a standard application of the well known Frisch–Waugh–Lovell theorem; simply put, it consists of fitting the model in two steps. In the first step, we expurgate the fixed effects from all variables in the model. This involves running a linear regression of each individual variable on only the high-dimensional effects and storing the residuals. In the second step, we run the regression of interest using the stored residuals of the variables obtained in the first step instead of the original variables. Because we are not dealing with the high-dimensional fixed effects, the regressions in the second step are easy to implement and will have the correct standard errors provided that we adjust the degrees of freedom. One reason why this approach works well is that the calculations in the first step are relatively simple. We can see from (2) that in this case, the algorithm involves only the computation of means. In the next example, we again use the ancillary dataset that accompanies the `felsdvreg` user-written command and show how to obtain the correct standard errors in a regression with two high-dimensional fixed effects. In the Stata code shown below, we speed up the algorithm by demeaning all variables with respect to one of the fixed effects.

```
. use felsdvsimul, clear
. recast double _all
. generate double temp=0
. generate double fe2=0
. generate double lastfe2=0
```

2. In the appendix, we provide a more extensive discussion of this issue.

```
. foreach var of varlist y x1 x2 {
  2. quietly {
  3. local i=0
  4. local dif=1
  5. capture drop mean
  6. egen double mean=mean(`var´), by(i)
  7. replace `var´=`var´-mean, nopromote
  8. while abs(`dif´)>epsdouble() {
  9. replace lastfe2=fe2, nopromote
 10. capture drop mean
 11. egen double mean=mean(fe2), by(i)
 12. capture drop fe2
 13. egen double fe2=mean(`var´+mean), by(j)
 14. replace temp=sum(reldif(fe2,lastfe2)), nopromote
 15. local dif=temp[_N]/_N
 16. display `i´ " " `dif´
 17. local i=`i´+1
 18. }
 19. noisily display "Total Number of Iterations for `var´ --> " `i´
 20. generate double `var´_res=`var´-fe2+mean
 21. }
 22. }
Total Number of Iterations for y --> 40
Total Number of Iterations for x1 --> 41
Total Number of Iterations for x2 --> 41
. regress y_res x1_res x2_res, nocons dof(69)
```

Source	SS	df	MS	Number of obs =	100
				F(2, 69) =	17.17
Model	1033.49122	2	516.745611	Prob > F =	0.0000
Residual	2076.72508	69	30.0974649	R-squared =	0.3323
				Adj R-squared =	0.0323
Total	3110.2163	100	31.102163	Root MSE =	5.4861

y_res	Coef.	Std. Err.	t	P>|t|	[95% Conf.	Interval]
x1_res	1.029258	.2151235	4.78	0.000	.6000987	1.458418
x2_res	-.709482	.2094198	-3.39	0.001	-1.127263	-.2917009

Notice that the iterative procedure was much faster, taking about 40 iterations for each variable. The final regression has the correct estimates for both the coefficients and standard errors. To obtain the correct standard errors, we had to adjust the degrees of freedom of the regression. This was done by using the `dof()` option in the `regress` command. Given that in `felsdvsimul.dta` we have 100 observations, that $G_1 = 15$, that $G_2 = 20$, and that we have six mobility groups and two regressors (x_1 and x_2), the regression has $100 - 15 - 20 + 6 - 2 = 69$ degrees of freedom.[3] We also can easily compute clustered standard errors by using the regression on the transformed variables. In the example below, we cluster the standard errors on the variable `g` and confirm the results using the `regress` command.[4]

3. See the appendix for a more detailed explanation.
4. If the clustering variable is different for each observation (each observation is treated as a cluster), then we obtain heteroskedasticity-robust White-corrected standard errors.

```
. quietly regress y_res x1_res x2_res, nocons mse1
. matrix VV=e(V)
. predict double res, residual
. _robust res, variance(VV) minus(31) cluster(g)
. display "Clustered standard error of x1 --> " sqrt(VV[1,1])
Clustered standard error of x1 --> .20274454
. display "Clustered standard error of x2 --> " sqrt(VV[2,2])
Clustered standard error of x2 --> .14387506
. quietly xi: regress y x1 x2 i.i i.j, vce(cluster g)
. display "Clustered standard error of x1 --> " _se[x1]
Clustered standard error of x1 --> .20274454
. display "Clustered standard error of x2 --> " _se[x2]
Clustered standard error of x2 --> .14387506
```

Finally, even though we ran a regression on transformed variables, we are still able to recover the estimates for the coefficients of the fixed effects. We do so by implementing the same iterative procedure discussed above to the residuals obtained when we subtract the effects of x_1 and x_2 from y.

```
. quietly regress y_res x1_res x2_res, nocons
. generate double dum=y-_b[x1_res]*x1-_b[x2_res]*x2
. capture drop mean
. egen double mean=mean(dum), by(i)
. replace dum=dum-mean, nopromote
(29 real changes made)
. local i=0
. local dif=1
. while abs(`dif´)>epsdouble() {
  2. quietly replace lastfe2=fe2, nopromote
  3. capture drop mean
  4. egen double mean=mean(fe2), by(i)
  5. capture drop fe2
  6. egen double fe2=mean(dum+mean), by(j)
  7. quietly replace temp=sum(reldif(fe2,lastfe2)), nopromote
  8. local dif=temp[_N]/_N
  9. local i=`i´+1
 10. }
. display "Total Number of Iterations for fe2 --> " `i´
Total Number of Iterations for fe2 --> 41
. quietly replace dum=dum-fe2, nopromote
. egen double fe1=mean(dum), by(i)
. quietly regress y x1 x2 fe1 fe2, nocons
. estimates table, b(%10.7f)
```

Variable	active
x1	1.0292583
x2	-0.7094820
fe1	1.0000000
fe2	1.0000000

We confirm that the estimated coefficients are correct by adding the variables `fe1` and `fe2` to the linear regression of y on x_1 and x_2. As expected, the estimated β coefficients are the correct ones and the coefficients associated with the variables `fe1` and `fe2` equal one.

Subtracting the influence of the fixed effects from each variable and working with only the residuals has some advantages compared with the process shown earlier that entailed direct estimation of the full regression with all the fixed effects added. First, the simple regressions in step 1 are likely to converge at a faster rate. Second, it is possible to test different specifications of the model using only the transformed variables without the need to deal with the high-dimensional fixed effects. And third, when dealing with very large datasets, we can substantially reduce memory requirements because during step 1 we only need to load into memory the variable being handled and the group identifiers for the fixed effects. In fact, we could do even better because the solution to the algorithm is performed independently across mobility groups, meaning that it would be possible to load each mobility group into memory separately.[5]

3 Extension to nonlinear models

In this section, we show that the iterative approach outlined earlier for the linear regression model can be extended to nonlinear models. With nonlinear models, it is possible to estimate correctly the vector β, but there is no easy solution for estimation of the associated standard errors. While it would be possible to bootstrap the standard errors, this solution is likely to be computationally very expensive. Given that with the iterative approach proposed in this article we obtain the correct values for the log-likelihood function, it may be easier to implement statistical tests for the coefficients based upon likelihood ratios (LRs). To illustrate, let us first consider a typical Poisson regression model with expected value given by

$$E(y_i) = \lambda_i = \exp(\mathbf{x}_i'\beta)$$

We know that the maximum likelihood estimators are obtained as the solution to:

$$\begin{aligned} \frac{\partial \ln L}{\partial \beta} &= \sum_{i=1} y_i\mathbf{x}_i - \mathbf{x}_i \exp(\mathbf{x}_i'\beta) \\ &= \sum_{i=1} \{y_i - \exp(\mathbf{x}_i'\beta)\}\,\mathbf{x}_i = \mathbf{0} \end{aligned}$$

If one of the regressors is a dummy variable, say $\mathbf{d}_j$, then its estimated coefficient, say α_j, has a closed-form solution given by

$$\exp(\alpha_j) = \mathbf{d}_j'\mathbf{y} \times \left[\mathbf{d}_j' \exp\left\{(\mathbf{x}_i'\beta)_{(j)}\right\}\right]^{-1} \tag{3}$$

5. Currently available in the Statistical Software Components archive are two user-written commands that implement the algorithms discussed in this article for estimation of linear regression models with two high-dimensional fixed effects. The first, `gpreg`, is a fast Mata implementation programmed by Johannes Schmieder. The second, `reg2hdfe`, is particularly suited for estimation with large datasets and was programmed by Paulo Guimarães.

where the subscript (j) in the argument of the exponential function shows that $\mathbf{d}_j$ is excluded from the argument. The above expression suggests a simple iterative strategy, much like the one used for the linear regression. We can alternate between estimation of a Poisson regression with k explanatory variables and calculation of the estimates for all the coefficients of the fixed effects using (3). These estimates can be kept in a single column vector. To show this algorithm at work, we replicate the estimates of the Poisson model with fixed effects, which appear in example 1 of [XT] **xtpoisson** as an illustration of that command with the `fe` option. That example shown in the manual includes an exposure variable, `service`, which we incorporated into the algorithm:

```
. webuse ships, clear
. quietly poisson acc op_75_79 co_65_69 co_70_74 co_75_79, exposure(service)
. keep if e(sample)
(6 observations deleted)
. bysort ship: egen sumy=total(acc)
. generate double off=0
. generate double temp=0
. local dif=1
. local ll1=0
. local i=0
. while abs(`dif´)>epsdouble() {
  2. quietly poisson acc op_75_79 co_65_69 co_70_74 co_75_79, offset(off)
  > noconstant
  3. local ll2=`ll1´
  4. local ll1=e(ll)
  5. capture drop xb
  6. predict double xb, xb
  7. quietly replace temp=xb-off+log(service), nopromote
  8. capture drop sumx
  9. bysort ship: egen double sumx=total(exp(temp))
 10. quietly replace off=log(sumy/sumx)+log(service), nopromote
 11. local dif=`ll2´-`ll1´
 12. local i=`i´+1
 13. }
. display "Total Number of Iterations --> " `i´
Total Number of Iterations --> 103
. quietly poisson acc op_75_79 co_65_69 co_70_74 co_75_79, noconstant
> offset(off)
. estimates table, b(%10.7f) eform
```

Variable	active
op_75_79	1.4688312
co_65_69	2.0080025
co_70_74	2.2669302
co_75_79	1.5736955

Extending the algorithm to two fixed effects is straightforward. We use the same dataset as before, but instead of including the dummy variables for year of construction (`co_65_69`, `co_70_74`, and `co_75_79`), we treat the year of construction as a fixed effect; that is, we let the variable `yr_con` identify a second fixed effect. Now the algorithm is

implemented without an exposure variable. For comparability purposes, we first run a Poisson regression that includes dummy variables for both fixed effects.

```
. webuse ships, clear
. local dif=1
. xi: poisson acc op_75_79 i.yr_con i.ship, nolog
i.yr_con          _Iyr_con_1-4        (naturally coded; _Iyr_con_1 omitted)
i.ship            _Iship_1-5          (naturally coded; _Iship_1 omitted)
Poisson regression                                Number of obs   =         34
                                                  LR chi2(8)      =     475.45
                                                  Prob > chi2     =     0.0000
Log likelihood = -118.47588                       Pseudo R2       =     0.6674
```

accident	Coef.	Std. Err.	z	P>|z|	[95% Conf.	Interval]
op_75_79	.2928003	.1127466	2.60	0.009	.071821	.5137796
_Iyr_con_2	.5824489	.1480547	3.93	0.000	.2922671	.8726308
_Iyr_con_3	.4627844	.151248	3.06	0.002	.1663437	.7592251
_Iyr_con_4	-.1951267	.2135749	-0.91	0.361	-.6137258	.2234724
_Iship_2	1.79572	.1666196	10.78	0.000	1.469151	2.122288
_Iship_3	-1.252763	.3273268	-3.83	0.000	-1.894312	-.6112142
_Iship_4	-.9044563	.2874597	-3.15	0.002	-1.467867	-.3410457
_Iship_5	-.1462833	.2351762	-0.62	0.534	-.6072202	.3146537
_cons	1.308451	.1972718	6.63	0.000	.9218049	1.695096

```
. keep if e(sample)
(6 observations deleted)
. bysort ship: egen sumy1=total(acc)
. bysort yr_con: egen sumy2=total(acc)
. generate double off=0
. generate double temp=0
. generate double temp1=0
. generate double temp2=0
. generate double fe1=0
. generate double fe2=0
. local ll1=0
. local i=0
. while abs(`dif´)>epsdouble() {
  2. quietly poisson acc op_75_79, noconstant offset(off)
  3. local ll2=`ll1´
  4. local ll1=e(ll)
  5. capture drop xb
  6. predict double xb, xb
  7. quietly replace temp1=xb-off+fe2, nopromote
  8. capture drop sumx
  9. bysort ship: egen double sumx=total(exp(temp1))
 10. quietly replace fe1=log(sumy1/sumx), nopromote
 11. quietly replace temp2=xb-off+fe1, nopromote
 12. capture drop sumx
 13. bysort yr_con: egen double sumx=total(exp(temp2))
 14. quietly replace fe2=log(sumy2/sumx), nopromote
 15. quietly replace off=fe1+fe2
 16. local dif=`ll2´-`ll1´
 17. local i=`i´+1
 18. }
```

```
. display "Total Number of Iterations --> " `i´
Total Number of Iterations --> 45
. quietly poisson acc op_75_79, noconstant offset(off)
. estimates table, b(%10.7f)
```

Variable	active
op_75_79	0.2928003

To test the statistical significance of the variable `op_75_79` using an LR test, we run the same regression as above but without the `op_75_79` variable and retaining the log-likelihood value. The value for the log likelihood of this restricted regression is −121.88042, which leads to a value of the LR test of LR = 2×(−118.47588+121.88042) = 6.80908. The LR statistic follows a chi-squared distribution with 1 degree of freedom, and its square root should be comparable with the z statistic reported in the Stata output for the Poisson regression. Taking the square root of the LR statistic, we obtain 2.6094213, which is very close to the z statistic for `op_75_79` that is reported by Stata in the Poisson regression that explicitly includes all dummy variables.[6] Application of the algorithm to Poisson regression was straightforward because we could find a closed-form solution for the coefficients associated with the fixed effects. However, in most nonlinear regression models, the fixed effects do not have a closed-form solution. As shown in Guimarães (2004), models from the multinomial logit family such as logit, multinomial logit, and conditional logit all can be fit using Poisson regression, meaning that the above algorithm could be used for these cases. The inexistence of a closed-form solution for the coefficients of the fixed effects does not invalidate use of the zigzag algorithm, but it requires the use of a numerical optimization routine to solve for the coefficients of the fixed effects. This routine may slow down the algorithm considerably. As an example of this approach, we show an application of the zigzag algorithm to fit a negative binomial model with fixed effects.[7]

6. The results are not identical because Stata reports the Wald statistic, while we calculated the LR statistic. Asymptotically, the two statistics are equivalent.
7. The fixed-effects negative binomial model (`xtnbreg` with the `fe` option) is not equivalent to a negative binomial model with dummy variables added for fixed effects (see Guimarães [2008]).

```
. webuse ships, clear
. xi: nbreg acc op_75_79 co_65_69 co_70_74 co_75_79 i.ship, nolog
i.ship            _Iship_1-5          (naturally coded; _Iship_1 omitted)
Negative binomial regression                    Number of obs   =         34
                                                LR chi2(8)      =      41.45
Dispersion     = mean                           Prob > chi2     =     0.0000
Log likelihood = -88.445258                     Pseudo R2       =     0.1898

------------------------------------------------------------------------------
    accident |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
    op_75_79 |   .3324104    .328116     1.01   0.311    -.3106852     .975506
    co_65_69 |   .8380919   .4378077     1.91   0.056    -.0199955    1.696179
    co_70_74 |   1.658684   .4850461     3.42   0.001      .708011    2.609357
    co_75_79 |   .8604224   .5955773     1.44   0.149    -.3068876    2.027732
    _Iship_2 |    2.35359   .4701847     5.01   0.000     1.432045    3.275135
    _Iship_3 |  -1.104561   .5214874    -2.12   0.034    -2.126657    -.082464
    _Iship_4 |  -.9606946   .4905212    -1.96   0.050    -1.922098    .0007092
    _Iship_5 |   -.077889   .4780747    -0.16   0.871    -1.014898    .8591201
       _cons |   .4230202   .5218569     0.81   0.418    -.5998006    1.445841
-------------+----------------------------------------------------------------
    /lnalpha |  -.7372302   .3814595                     -1.484877    .0104166
-------------+----------------------------------------------------------------
       alpha |   .4784372   .1825044                      .2265302    1.010471
------------------------------------------------------------------------------
Likelihood-ratio test of alpha=0:  chibar2(01) =   60.06 Prob>=chibar2 = 0.000
. keep if e(sample)
(6 observations deleted)
. egen id=group(ship)
. quietly sum id
. local maxg=r(max)
. local dif=1
. local ll1=0
. local i=0
. generate double off1=0
. generate double off2=0
. while abs(`dif´)>epsdouble() {
  2. quietly nbreg acc op_75_79 co_65_69 co_70_74 co_75_79, noconstant offset(off1)
  3. local lna=log(e(alpha))
  4. constraint define 1 [lnalpha]_cons=`lna´
  5. local ll2=`ll1´
  6. local ll1=e(ll)
  7. capture drop xb
  8. predict double xb, xb
  9. quietly replace off2=xb-off1, nopromote
 10. forval j=1/`maxg´ {
 11. quietly nbreg acc if id==`j´, offset(off2) constraint(1)
 12. quietly replace off1=_b[_cons] if e(sample), nopromote
 13. }
 14. local dif=`ll2´-`ll1´
 15. local i=`i´+1
 16. }
. display "Total Number of Iterations -->" `i´
Total Number of Iterations -->122
. quietly nbreg acc op_75_79 co_65_69 co_70_74 co_75_79, offset(off1) noconstant
```

```
. estimates table, b(%10.7f)
```

Variable	active
accident	
op_75_79	0.3324104
co_65_69	0.8380920
co_70_74	1.6586841
co_75_79	0.8604225
lnalpha	
_cons	-0.7372302

Following an approach similar to the one used for the negative binomial model, it should be possible to extend the algorithm to other nonlinear models. In general, the algorithm should work well with models that have globally concave log-likelihood functions such as the ones discussed here.

4 Conclusion

In this article, we successfully explored the implementation of the full Gauss–Seidel algorithm to fit regression models with high-dimensional fixed effects. The main advantage of this procedure is the ability to fit linear regression models with two or more high-dimensional fixed effects under minimal memory requirements. Generalizing the procedure to nonlinear regression models is straightforward, particularly in cases having a closed-form solution for the fixed effect.

We do not claim, however, that our procedure is a superior estimation strategy. Quite to the contrary, the zigzag algorithm can be very slow, and researchers should use more efficient estimation techniques whenever available. We know that the linear regression model with two high-dimensional fixed effects is fit much more efficiently with the user-written command `felsdvreg`, the same way that `xtpoisson` is the better approach to fit a Poisson model with a single high-dimensional fixed effect. Nevertheless, the zigzag algorithm may prove useful in some circumstances, namely, when existing approaches do not work because of hardware (memory) limitations or when no other known ways of fitting the model exist. As we mentioned earlier, the estimation strategy outlined in this article is time consuming, but it does have the advantages of imposing minimum memory requirements and of being simple to implement.

There are many ways to improve the speed of the algorithms discussed above, and research is needed to figure out how to improve them. In the examples presented in this article, we used a very strict convergence criterion. In practical applications, though, a more relaxed criterion is likely to substantially lower the number of iterations without meaningful changes to the final results. Other obvious tools that will speed the algorithms include more efficient Stata code (possibly Mata), better starting values, and the use of convergence acceleration techniques in the algorithm. Speed should not be hard to accomplish because the estimates of fixed effects tend to converge monotonically,

making it possible to use the information from the last iterations to adjust the trajectory of the fixed-effect estimates and thus obtain faster convergence.

We would like to make researchers aware of the large-sample properties of these estimators. Given the multiple-dimension panel data, the asymptotic behavior of the estimators can be studied in different ways. The estimators are consistent if we are willing to admit that the dimension of the groups is unrelated to sample size. From the literature on panel data, we know that a critical situation arises when the dimension of one fixed effect (say N) increases without bound while T remains fixed. In this case, the number of individual parameters increases as N increases, raising the incidental parameter problem originally discussed by Neyman and Scott (1948) and recently reviewed by Lancaster (2000). In the linear regression model, it is well known that the least-square dummy variable model (or equivalently, the within estimator) still provides consistent estimates of the slope coefficients, but not of the individual fixed effects. This is because in the linear model, the estimators of the slopes and of the individual effects are asymptotically independent (Hsiao 2003). As for nonlinear models, in general, the estimators of the regression coefficients (the slopes) will be plagued by the incidental parameter problem and, for T fixed, will be inconsistent. With one fixed effect, the incidental problem may be overcome by finding a minimal sufficient statistic for the individual effect as in the conditional logit model of Chamberlain (1980). Lancaster (2000) offers useful reparameterizations for a number of conventional nonlinear regression models. If, however, both N and T increase without bound, the inconsistency generated by the incidental parameter problem is circumvented, leading to consistent estimates of the slope and the individual effects.

We should stress that the article presents a technique for estimation of models with large numbers of dummy variables. From an asymptotic perspective, this approach is not equivalent to the use of panel-data estimators that condition out or difference the fixed effects. This means that consistency and asymptotic normality of our estimators rely on the implicit assumption that the number of groups remains fixed as the sample size tends to infinity.

Finally, we would like to point out that we motivated the introduction of fixed effects in large datasets as a way to control for unobserved heterogeneity. However, there may be other reasons why researchers may want to deal with large numbers of dummy variables. With large datasets, it may not make sense to impose functional relationships in the variables, and we can instead let the data best show those relationships using a dummy variable for each different value of the regressor. With millions of observations, the loss in degrees of freedom is minimal.

5 References

Abowd, J. M., R. H. Creecy, and F. Kramarz. 2002. Computing person and firm effects using linked longitudinal employer–employee data. Technical Paper No. TP-2002-06, Center for Economic Studies, U.S. Census Bureau.
http://lehd.did.census.gov/led/library/techpapers/tp-2002-06.pdf.

Abowd, J. M., F. Kramarz, and D. N. Margolis. 1999. High wage workers and high wage firms. *Econometrica* 67: 251–333.

Andrews, M., T. Schank, and R. Upward. 2006. Practical fixed-effects estimation methods for the three-way error-components model. *Stata Journal* 6: 461–481.

Carneiro, A., P. Guimarães, and P. Portugal. 2010. Real wages and the business cycle: Accounting for worker, firm, and job heterogeneity. Unpublished manuscript.

Chamberlain, G. 1980. Analysis of covariance with qualitative data. *Review of Economic Studies* 47: 225–238.

Cornelissen, T. 2008. The Stata command felsdvreg to fit a linear model with two high-dimensional fixed effects. *Stata Journal* 8: 170–189.

Greene, W. 2004. The behaviour of the maximum likelihood estimator of limited dependent variable models in the presence of fixed effects. *Econometrics Journal* 7: 98–119.

Guimarães, P. 2004. Understanding the multinomial-Poisson transformation. *Stata Journal* 4: 265–273.

———. 2008. The fixed effects negative binomial model revisited. *Economics Letters* 99: 63–66.

Hsiao, C. 2003. *Analysis of Panel Data*. 2nd ed. Cambridge: Cambridge University Press.

Lancaster, T. 2000. The incidental parameter problem since 1948. *Journal of Econometrics* 95: 391–413.

Neyman, J., and E. Scott. 1948. Consistent estimation from partially consistent observations. *Econometrica* 16: 1–32.

Smyth, G. K. 1996. Partitioned algorithms for maximum likelihood and other non-linear estimation. *Statistics and Computing* 6: 201–216.

About the authors

Paulo Guimaraes is a research associate professor at the University of South Carolina and currently is a visiting professor at the University of Porto.

Pedro Portugal is a senior researcher at the Bank of Portugal and a visiting full professor at the Universidade Nova de Lisboa.

Appendix

In this appendix, we try to provide some intuition on the issue of identification of the fixed effects. Consider first a regression model with N observations and a single fixed effect with G_1 levels (a one-way ANOVA model):

$$E(y_{it}) = \mu + \alpha_i$$

If we replace $E(y_{it})$ by the data cell means, we have a system of G_1 equations on $G_1 + 1$ unknowns. To solve this model, we need to impose one restriction (typically, $\mu = 0$ or $\alpha_1 = 0$). With this restriction, we are able to estimate G_1 coefficients of the model. This in turn means that SSR has $N - G_1$ degrees of freedom (or $N - k - G_1$ if there are an additional k noncollinear explanatory variables in the model).

Consider now a regression model with two fixed effects with G_1 and G_2 levels, respectively:

$$E(y_{it}) = \mu + \alpha_i + \eta_j$$

The unique combinations of α_i and η_j available on the data define a set of equations, but the interdependence between these equations does not make obvious how many restrictions are needed to identify the coefficients. Abowd, Creecy, and Kramarz (2002) presented an algorithm that counts the number of restrictions needed to identify the coefficients. To illustrate, consider an example with $G_1 = G_2 = 3$ and the following unique combinations of the levels of the fixed effects:

$$\begin{aligned}
&\mu + \alpha_1 + \eta_1 \\
&\mu + \alpha_1 + \eta_2 \\
&\mu + \alpha_2 + \eta_1 \\
&\mu + \alpha_2 + \eta_3 \\
&\mu + \alpha_3 + \eta_2 \\
&\mu + \alpha_3 + \eta_3
\end{aligned}$$

To solve this system of equations, we can start out by imposing two restrictions, for example, $\mu = 0$ and $\alpha_1 = 0$. With these restrictions in place, we can immediately identify η_1 and η_2. In turn, knowledge of η_1 and η_2 allow the identification of α_2 and α_3, thereby leading to the identification of η_3. This sequence of steps is illustrated below:

$$\begin{array}{c} \boxed{\eta_1} \\ \boxed{\eta_2} \\ \alpha_2 + \boxed{\eta_1} \\ \alpha_2 + \eta_3 \\ \alpha_3 + \boxed{\eta_2} \\ \alpha_3 + \eta_3 \end{array} \quad \rightarrow \quad \begin{array}{c} \boxed{\eta_1} \\ \boxed{\eta_2} \\ \boxed{\alpha_2} + \boxed{\eta_1} \\ \boxed{\alpha_2} + \eta_3 \\ \boxed{\alpha_3} + \boxed{\eta_2} \\ \boxed{\alpha_3} + \eta_3 \end{array} \quad \rightarrow \quad \begin{array}{c} \boxed{\eta_1} \\ \boxed{\eta_2} \\ \boxed{\alpha_2} + \boxed{\eta_1} \\ \boxed{\alpha_2} + \boxed{\eta_3} \\ \boxed{\alpha_3} + \boxed{\eta_2} \\ \boxed{\alpha_3} + \boxed{\eta_3} \end{array}$$

Identification of the coefficients is possible only because the equations are "connected". Consider now an alternative regression model with parameters given by

$$\begin{array}{c} \mu+\alpha_1+\eta_1 \\ \mu+\alpha_1+\eta_2 \\ \mu+\alpha_2+\eta_1 \\ \mu+\alpha_2+\eta_2 \\ \mu+\alpha_3+\eta_3 \\ \mu+\alpha_3+\eta_3 \end{array}$$

If we follow the same strategy as above and set $\mu = 0$ and $\alpha_1 = 0$, we now have the following sequence of steps:

$$\begin{array}{c} \boxed{\eta_1} \\ \boxed{\eta_2} \\ \alpha_2+\boxed{\eta_1} \\ \alpha_2+\boxed{\eta_2} \\ \alpha_3+\eta_3 \\ \alpha_3+\eta_3 \end{array} \quad \rightarrow \quad \begin{array}{c} \boxed{\eta_1} \\ \boxed{\eta_2} \\ \boxed{\alpha_2}+\boxed{\eta_1} \\ \boxed{\alpha_2}+\boxed{\eta_2} \\ \alpha_3+\eta_3 \\ \alpha_3+\eta_3 \end{array}$$

Now, with these two restrictions, we are only able to identify the coefficients in the first four equations. This happens because the first set of equations does not share any coefficients with the remaining equations. In Abowd, Creecy, and Kramarz (2002) terminology, there are now two mobility groups. Only with an additional restriction ($\alpha_3 = 0$ or $\eta_3 = 0$) can we identify the remaining coefficients. It should be obvious that each additional mobility group requires an additional restriction. If we let M designate the number of mobility groups, then we conclude that the number of identified coefficients is G_1+G_2-M, and the degrees of freedom associated with SSR is $N-G_1-G_2+M$ (or $N-k-G_1-G_2+M$ if there are k noncollinear explanatory variables in the model). We can use a similar logic to that outlined above to count the number of identifiable coefficients in a model with more than two fixed effects. However, development of an algorithm for this purpose is no simple task and is well beyond the scope of this article.