Online Appendix B: The Code for Salary Equity Analysis

Regular salary equity studies can be a best practice among employers committed to salary equity and fairly managed compensation. As discussed in an article forthcoming in *Public Personnel Management* entitled “How to do a salary equity study – With an illustrative example from higher education,” by Lori L. Taylor, Joanna N. Lahey, Molly I. Beck, and Jeffrey E. Froyd, salary equity studies typically use regression techniques to model the relationship between a measure of compensation, a demographic of interest (such as sex or race), and an array of other possible explanatory factors, which are referred to as controls. Below, we present sample code that can be used to conduct such analyses using two popular software packages.

The sample code provides instructions for a specification using panel data and random effects. It can be adapted to conduct a salary analysis in a variety of statistical platforms. Comments and advice are italicized to differentiate them from the actual statistical commands. We conducted our analysis using Stata, which is a statistical package available for Windows, Mac, and Linux/Unix computers (www.stata.com). However, we also provide analytical code to conduct the analysis using SAS/Stat, which is a statistical package available in a variety of platforms (www.sas.com).

In all cases, the data should be structured so there is one observation per person per year. Start with an excel file structured as in Table B1. You may add as many other control variables as you like, but don’t forget that you need to have many more employees than you have variables. The file should include data on anyone who worked for your organization at any time during the analysis period, even if they are no longer employed by your organization.

Table B1: An Illustration of the Data File Structure

Employee ID	year	unit	position	Monthly salary	sex	race	experience	education	Year hired	other control variable
QAD579	2017	accounting	Clerk II	3000	M	W	12	MA	2015	A
QAD579	2016	accounting	Clerk II	2700	M	W	11	MA	2015	A
QAD579	2015	accounting	Clerk I	2000	M	W	10	BA	2015	C
AMZ123	2017	accounting	AA	2500	F	H	7	BA	2012	B
AMZ123	2016	accounting	AA	2400	F	H	6	BA	2012	B
WDA669	2015	purchasing	AA	1800	F	W	0	BA	2015	B

STATA Code

**To start, put all your datasets and Stata files in the same folder on your computer. You will also save your modified version of this file as a plain text file with the extension ".do" as in, salary_study_2018_v1.do. Such files can be used in Stata by going to File, then choosing Do.*

```
scalar modelyear=2017                    /*the most recent year for which you have data*/
```

**This line imports an excel spreadsheet into a Stata dataset.*

```
import excel "filename", sheet("Sheet1") firstrow allstring clear
```

**The import command reads all the variables in a strings (i.e. non-numeric variables) to insure no information content is lost. This command tells Stata that the following variables are actually numbers and should be treated as such. The names of your variables may differ.*

```
destring monthlysalary experience yearhired year, replace
```

**If your data codes sex and race using letters, these commands will turn them into binary variables that can be used in summary statistics and regression analysis. If your dataset codes these variables as 1, 2, etc. numbers, you will still need to turn them into binary variables but will not need the quotation marks. For example, replace the second line of code below to read: replace male = 1 if sex ==1.*

```
gen male=0
replace male=1 if sex=="M"
gen white=0
replace white=1 if race=="W"
gen black=0
replace black=1 if race=="B"
gen hispanic=0
replace hispanic=1 if race=="H"
```

**creating variables for years of service and log monthly salary. Replace variables using the names in your dataset.*

```
gen yrsofservice=year-yearhired
gen logmonthly=ln(monthlysalary)
```

**creating a numeric employee id.*

```
egen id=group(Employee_ID)
```

**This command declares the data to be time series data with year as the unit of time and id as the observation that will be repeated across time.*

```
tsset id year
```

**This line is a shorthand way of not having to rewrite the controls in each specification. Replace the variable names with those you will be using in all of the specifications in your study.*

```
global controls "male white black hispanic experience yrsofservice"
```

**The xi command is a short-hand way of turning a categorical variable into a vector of dummy variables. Here this is creating and using unit, position, other, education and year fixed effects.*

```
xi i.unit i.position i.other i.education i.year
```

**xtreg is a method of regression using panel data. These commands provide the baseline regression results describing what predicts salary.*

```
xtreg logmonthly $controls _I*, re cluster(id)
```

**This portion of the code creates predictions for what each individual would be paid were they a white non-Hispanic male. Only predictions for the most recent year are generated.*

**first you need to capture the individual random effects and calculate the average random effect by position*

```
predict random_effect, u
keep if year==modelyear
egen  mean_random_effect=mean(random_effect), by(position)
```

**then you need to recode the data so that everyone is a white, non-Hispanic male*

```
replace male=1
replace white=1
replace black=0
replace hispanic=0
```

**Next predict wages and the standard error of the predicted wage for those hypothetically white Hispanic males*

```
predict pwage, xb
predict stderr, stdp
gen predicted_logsalary=pwage+mean_random_effect
gen predictedmonthlysal=exp(predicted_logsalary)
```

**This code prints the observed wages and predicted wags for the 10% of workers with the largest gap between actual and predicted. If you are using a different cutoff, you would change the 90 (i.e. the 90th percentile) to the cutoff of your choice. If you want output for every employee, simply remove the phrase "if abs(zstat) >=a90"*

```
gen zstat=(logm-predicted_logsalary)/stderr
egen a90=pctile(abs(zstat)), p(90)

list Employee monthly predictedmonthly if abs(zstat) >=a90 & abs(zstat) !=.
```

SAS Code

```
%let modelyear=2017;                              /*the most recent year for which you have data*/

/*This block of code imports an excel spreadsheet into a SAS dataset called “salfile” */

PROC IMPORT OUT= WORK.salfile
        DATAFILE= "filename"
        DBMS=EXCEL REPLACE;
    RANGE="Sheet1$";
    GETNAMES=YES;
    MIXED=NO;
    SCANTEXT=YES;
    USEDATE=YES;
    SCANTIME=YES;
RUN;

/*This block of code constructs indicator variables within salfile. If your data codes sex and race
using letters, these commands will turn them into binary variables that can be used in summary
statistics and regression analysis.  If your dataset codes these variables as 1, 2, etc. numbers,
you will still need to turn them into binary variables but will not need the quotation marks.  For
example replace the second line of code below to read: if sex=1 then male=1......*/

Data salfile; set salfile;
If sex="M" then male=1; else male=0;
If race="W" then white=1; else white=0;
If race="B" then black=1; else black=0;
If race="H" then hispanic=1; else hispanic=0;

/*creating variables for years of service and log monthly salary.  Replace variables using the
names in your dataset*/

yrs_of_service=year-yearhired;
logmonthly=log(monthlysalary);

/*This line is a shorthand way of not having to rewrite the controls in each specification.
Replace the variable names with those you will be using in all of the specifications in your
study.*/

%let controls=%str(male white black hispanic experience yrs_of_service);
```

/**This block of code adds a duplicate observation for each person who appears in the data set in the model year. The variable "predfile" takes on a value of one if the observation is a duplicate, and zero otherwise.*/*

```
data salfile; set salfile salfile(in=duplicate where=(year=&modelyear));
predfile=duplicate;
```

/**By manipulating the characteristics of these duplicate observations, one generates predicted wages for a person with specific characteristics. This portion of the code creates predictions for what each individual would be paid were they a white non-Hispanic male. Only predictions for the most recent year are generated. To ensure that these manipulated observations are not included in the regression estimation, the dependent variable (logmonthly) is set to missing.*/*

```
if predfile=1 then do;
    logmonthly=.;
    male=1;
    white=1;
    black=0;
    hispanic=0;
    end;
```

/**proc mixed implements a method of maximum likelihood regression using panel data. These commands provide the baseline regression results describing what predicts salary. This block of code invokes the mixed procedure and applies it to the data in salfile. The variables in the class statement are declared to be categorical; SAS will construct indicator variables for each of them. This set of commands will generate two output datasets. The first, named "pfile" contains the predicted values for each observation. The second, named "random_effects" contains the estimated random effects for each employee.*/*

```
proc mixed data=salfile method=ML; class unit position education other_control_variable1
Employee_ID year;
model logmonthly=unit position education other_control_variable1 &controls year/solution
outp=pfile;
random Employee_ID /solution;
ods output solutionr= random_effects; run;
run;
```

/**Next capture the predicted wages and the standard errors of the predicted wage for those hypothetically white Hispanic males.*/*

```
data predwagefile; set pfile(where=(predfile=1));
```

```
/*merge on the estimated random effects*/

proc sort; by Employee_ID;
data predwagefile; merge predwagefile(in=uu) random_effects(keep=Employee_ID Estimate);
by Employee_ID;  if uu=1;

/*calculate the mean of the random effects, by position and merge them back onto
predwagefile*/

proc sort; by position;
proc means noprint; var Estimate; by position; output out=avgs mean=avgrandom;
data predwagefile; merge predwagefile avgs(keep=position avgrandom); by position;

/*generate predicted wages, replacing the individual random effects (named "Estimate") with
the average random effects by position (named "avgrandom")*/

predwage=pred-Estimate+ avgrandom;
predictedmonthlysal=exp(predwage);

/*This code prints the observed wages and predicted wags for the 10% of workers with the
largest gap between actual and predicted.  If you are using a different cutoff, you would change
the 90 (i.e. the 90th percentile) to the cutoff of your choice. If you want output for every
employee, simply remove the phrase "where abs(zstat) >=a90"*/

zstat=(log(monthlysalary)-predwage)/StdErrPred;
abszstat=abs(zstat);
proc means noprint; by year; var abszstat; output out=pctl p90=a90;
data predwagefile; merge predwagefile pctl; by year;

proc print; var Employee_ID monthlysalary predictedmonthlysal;  where abszstat >=a90; run;
```