The Stata Journal (*yyyy*) *vv*, Number *ii*, pp. 1–43

Updates to the ipfraking ecosystem

Stanislav Kolenikov
Abt Associates
stas_kolenikovs@abtassoc.com

Abstract. Kolenikov (2014) introduced package `ipfraking` for weight calibration procedures known as iterative proportional fitting, or raking, of complex survey weights. This article briefly describes the original package, and adds updates to the core program, as well as a host of additional programs that are used to support the process of creating survey weights in the authors' production code.

Keywords: st0001, survey, calibration, weights, raking

1 Introduction and background

Large scale social, behavioral and health data are often collected via complex survey designs that may involve some or all of stratification, multiple stages of selection and unequal probabilities of selection (Korn and Graubard 1995, 1999). In an ideal setting, varying probabilities of selection are accounted for by using the Horvitz-Thompson estimator of the totals (Horvitz and Thompson 1952; Thompson 1997), and the remaining sampling fluctuations can be further ironed out by post-stratification (Holt and Smith 1979). However, on top of the planned differences in probabilities of obtaining a response from a sampled unit, non-response is a practical problem that has been growing more acute over the recent years (Groves et al. 2001; Pew Research Center 2012). The analysis weights that are provided along with the public use microdata by data collecting agencies are designed to account for unequal probabilities of selection, non-response, and other factors affecting imbalance between the population and the sample, thus making the analyses conducted on such microdata generalizable to the target population.

Earlier, I introduced (Kolenikov 2014) a Stata package called `ipfraking` that implements calibration of survey weights to known control totals to ensure that the resulting weighted data are representative of the population of interest. The process of calibration is aimed at aligning the sample totals of the key variables with those known for the population as a whole.

For a given finite population $\mathcal{U}$ of units indexed $i = 1, \dots, N$, the interests of survey statisticians often lie in estimating the population total of a variable Y

$$T[Y] = \sum_{i \in \mathcal{U}} Y_i \tag{1}$$

A sample $\mathcal{S}$ of n units indexed by $j = 1, \dots, n$ is taken from $\mathcal{U}$. If the probability to select the i-th unit is known to be π_i, then the *probability weights*, or *design weights*, are given by the inverse probability of selection:

$$w_{1i} = \pi_i^{-1} \tag{2}$$

With these weights, an unbiased (design-based, non-parametric) estimator of the total (1) is (Horvitz and Thompson 1952)

$$t_1[y] = \sum_{j \in \mathcal{S}} \frac{y_j}{\pi_j} \equiv \sum_{j \in \mathcal{S}} w_{1j} y_j, \tag{3}$$

The subindex 1 indicates that the weights w_{1i} were used in obtaining this estimator. Probability weights protect the end user from potentially informative sampling designs, in which the probabilities of selection are correlated with outcomes, and the design-based methods generally ensure that inference can be generalized to the finite population even when the statistical models used by analysts and researchers are not specified correctly (Pfeffermann 1993; Binder and Roberts 2003).

Often, survey statisticians have auxiliary information on the units in the frame, and such information can be included it at the sampling stage to create more efficient designs. Unequal probabilities of selection are then controlled with probability weights, implemented as `[pw=`*exp*`]` in Stata (and can be permanently affixed to the data set with `svyset` command).

In many situations, however, usable information is not available beforehand, and may only appear in the collected data. The census totals of the age and gender distribution of the population may exist, but age and gender of the sampled units is unknown until the survey measurement is taken on them. It is still possible to capitalize on this additional data by adjusting the weights in such a way that the reweighted data conforms to these known figures. The procedures to perform these reweighting steps are generally known as *weight calibration* (Deville and Särndal 1992; Deville et al. 1993; Kott 2006, 2009; Särndal 2007).

Suppose there are several (categorical) variables, referred to as *control variables*, that are available for both the population and the sample (age groups, race, gender, educational attainment, etc.). Weight calibration aims at adjusting the margins, or low level interactions, via an iterative optimization aimed at satisfying the *control totals* for the control variables $\mathbf{x} = (x_1, \ldots, x_p)$:

$$\sum_{j \in \mathcal{S}} w_{3j} \mathbf{x}_j = T[\mathbf{X}_j] \tag{4}$$

where the right hand side is assumed to be known from a census or a higher quality survey. Deville and Särndal (1992) framed the problem of finding a suitable set of weights as that of constrained optimization with the control equations (4) serving as constraints, and optimization targeted at making the discrepancy between the design weights w_{1j} and calibrated weights w_{3j} as close as possible, in a suitable sense.

In package `ipfraking` (Kolenikov 2014), I implemented a popular calibration algorithm, known as *iterated proportional fitting*, or as *raking*, which consists of iterative updating (post-stratification) of each of the margins. (For an in-depth discussion of distinctions between raking and post-stratification, see Kolenikov (2016).) Since 2014, the continuing code development resulted in additional features that this update documents.

2 Package description

Below, I provide full syntax, and list the new features in a dedicated section.

2.1 Syntax of `ipfraking`

`ipfraking` [*if*] [*in*] [*weight*] `, ctotal(`*matname* [*matname ...*]`)` [`generate(`*newvarname*`) replace double iterate(`#`) tolerance(`#`) ctrltolerance(`#`) trace nodivergence trimhiabs(`#`) trimhirel(`#`) trimloabs(`#`) trimlorel(`#`) trimfrequency(once|sometimes|often) double meta nograph`]

Note that the weight statement `[pw=`*varname*`]` is required, and must contain the initial weights.

Required options

`ctotal(`*matname* [*matname ...*]`)` supplies the names of the matrices that contain the control totals, as well as meta-data about the variables to be used in calibration.

❏ Technical note

The row and column names of the control total matrices (see [P] **matrix rownames**) should be formatted as follows.

- `rownames`: the name of the control variable
- `colnames`: the values the control variables takes
- `coleq`: the name of the variable for which total is computed; typically it is identically equal to 1.

See examples in Section 3.

❏

`generate(`*newvarname*`)` contains the name of the new variable to contain the raked weights.

`replace` indicates that the weight variable supplied in the `[pw=`*varname*`]` expression should be overwritten with the new weights.

One and only one of `generate()` or `replace` must be specified.

Linear calibration

`linear` requests linear calibration of weights.

Options to control convergence

`tolerance(#)` defines convergence criteria (the change of weights from one iteration to next). The default is 10^{-6}.

`iterate(#)` specifies the maximum number of iterations. The default is 2000.

`nodivergence` overrides the check that the change in weights is greater at the current iteration than in the previous one, i.e., ignores this termination condition. It is generally recommended, especially in calibration with simultaneous trimming.

`ctrltolerance(#)` defines the criterion to assess the accuracy of the control totals. It does not impact iterations or convergence criteria, but rather only triggers alerts in the output. The default value is 10^{-6}.

`trace` requests a trace plot to be added.

Trimming options

`trimhiabs(#)` specifies the upper bound U on the greatest value of the raked weights. The weights that exceed this value will be trimmed down, so that $w_{3j} \leq U$ for every $j \in \mathcal{S}$.

`trimhirel(#)` specifies the upper bound u on the adjustment factor over the baseline weight. The weights that exceed the baseline times this value will be trimmed down, so that $w_{3j} \leq uw_{1j}$ for every $j \in \mathcal{S}$.

`trimloabs(#)` specifies the lower bound L on the smallest value of the raked weights. The weights that are smaller than this value will be increased, so that $w_{3j} \geq L$ for every $j \in \mathcal{S}$.

`trimlorel(#)` specifies the lower bound l on the adjustment factor over the baseline weight. The weights that are smaller than the baseline times this value will be increased, so that $w_{3j} \geq lw_{1j}$ for every $j \in \mathcal{S}$.

`trimfreqency(`*keyword*`)` specifies when the trimming operations are to be performed. The following keywords are recognized:

`often` means that trimming will be performed after each marginal adjustment.

`sometimes` means that trimming will be performed after a full set of variables has been used for post-stratification. This is the default behavior if any of the numeric trimming options above are specified.

`once` means that trimming will be performed after the raking process is declared to have converged.

The numeric trimming options `trimhiabs(#)`, `trimhirel(#)`, `trimloabs(#)`, `trimlorel(#)` can be specified in any combination, or entirely omitted to produce untrimmed weights. By default, there is no trimming.

Miscellaneous options

`double` specifies that the new variable named in `generate()` option should be generated as double type. See [D] **data types**.

`meta` puts information taken by `ipfraking` as inputs and produced throughout the process into characteristics stored with the variable specified in `generate()` option. See Section 3.5.

`nograph` omits the histogram of the calibrated weights, which can be used to speed up `ipfraking` (e.g., in replicate weight production).

2.2 New features of `ipfraking`

Since the first publication, the following features and options were added.

Reporting of results and errors by `ipfraking` was improved in several directions.

1. The discrepancy for the worst fitting category is now being reported.
2. The number of trimmed observations is reported.
3. If `ipfraking` determines that the categories do not match in the control totals received from `ctotals()` and those found in the data, a full listing of categories is provided, and the categories not found in one or the other are explicitly shown.

Linear calibration (Case 1 of Deville and Särndal (1992)) is provided with `linear` option. The weights are calculated analytically:

$$w_{j,\text{lin}} = w_{1j}(1 + \mathbf{x}_j'\lambda). \quad \lambda = \Bigl(\sum_{j\in\mathcal{S}} w_{1j}\mathbf{x}_j\mathbf{x}_j'\Bigr)^{-1}(T[\mathbf{X}_j] - t_1[y]) \tag{5}$$

This works very fast, but has an undesirable artefact of producing negative weights, as the range of weights is not controlled. (As raking works by multiplying the currents weights by positive factors, if the input weights are all positive, the output weights will be positive as well.) Negative weights are not allowed by the official `svy` commands or commands that work with `[pweights]`. In author's experience, running linear weights first, pulling up the negative and small positive weights (`replace weight = 1 if weight <= 1`) and re-raking using the "proper" iterative proportional fitting runs faster than raking from scratch. An example of linearly calibrated weights is given below in Section **??**.

Option `meta` saves more information in characteristics of the calibrated weight variables.

(*Continued on next page*)

```
. capture drop rakedwgt3

. ipfraking [pw=finalwgt], gen( rakedwgt3 ) ///
>     ctotal( ACS2011_sex_age Census2011_region Census2011_race ) ///
>     trimhiabs(200000) trimloabs(2000) meta
 Iteration 1, max rel difference of raked weights = 14.95826
 Iteration 2, max rel difference of raked weights = .21474256
 Iteration 3, max rel difference of raked weights = .02754514
 Iteration 4, max rel difference of raked weights = .00511347
 Iteration 5, max rel difference of raked weights = .00095888
 Iteration 6, max rel difference of raked weights = .00018036
 Iteration 7, max rel difference of raked weights = .00003391
 Iteration 8, max rel difference of raked weights = 6.377e-06
 Iteration 9, max rel difference of raked weights = 1.199e-06
 Iteration 10, max rel difference of raked weights = 2.254e-07
The worst relative discrepancy of  3.0e-08 is observed for race == 3
Target value =   20053682; achieved value =   20053682
Trimmed due to the upper absolute limit: 5 weights.

   Summary of the weight changes

              |    Mean    Std. dev.    Min        Max       CV
--------------+---------------------------------------------------
Orig weights  |    11318      7304      2000      79634   .6453
Raked weights |    22055     18908      4033     200000   .8573
Adjust factor |   2.1486              0.9220    18.9828

. char li rakedwgt3[]
  rakedwgt3[source]:          finalwgt
  rakedwgt3[objfcn]:          2.25435521346e-07
  rakedwgt3[maxctrl]:         3.00266822363e-08
  rakedwgt3[converged]:       1
  rakedwgt3[worstcat]:        3
  rakedwgt3[worstvar]:        race
  rakedwgt3[command]:         [pw=finalwgt], gen( rakedwgt3 ) ctotal( ACS2011_sex_age Census2011_region ..
  rakedwgt3[trimloabs]:       trimloabs(2000)
  rakedwgt3[trimhiabs]:       trimhiabs(200000)
  rakedwgt3[trimfrequency]:   sometimes
  rakedwgt3[hash1]:           2347674164
  rakedwgt3[mat3]:            Census2011_race
  rakedwgt3[over3]:           race
  rakedwgt3[totalof3]:        _one
  rakedwgt3[Census2011_race]: 7.48567503861e-09
  rakedwgt3[mat2]:            Census2011_region
  rakedwgt3[over2]:           region
  rakedwgt3[totalof2]:        _one
  rakedwgt3[Census2011_region]:
                              3.00266822363e-08
  rakedwgt3[mat1]:            ACS2011_sex_age
  rakedwgt3[over1]:           sex_age
  rakedwgt3[totalof1]:        _one
  rakedwgt3[ACS2011_sex_age]: 4.13778410340e-09
  rakedwgt3[note1]:           Raking controls used: ACS2011_sex_age Census2011_region Census2011_race
  rakedwgt3[note0]:           1
```

The following characteristics are stored with the newly created weight variable (see [P] **char**).

`command`	The full command as typed by the user
matrix name	The relative matrix difference from the corresponding control total, see [D] **functions**
`trimhiabs`, `trimloabs`, `trimhirel`, `trimlorel`, `trimfrequency`	Corresponding trimming options, if specified
`maxctrl`	the greatest `mreldif` between the targets and the achieved weighted totals
`objfcn`	the value of the relative weight change at exit
`converged`	whether `ipfraking` exited due to convergence (1) vs. due to an increase in the objective function or reaching the limit on the number of iterations (0)
`source`	weight variable specified as the `[pw=]` input
`worstvar`	the variable in which the greatest discrepancy between the targets and the achieved weighted totals (`maxctrl`) was observed
`worstcat`	the category of the `worstvar` variable in which the greatest discrepancy was observed

For the control total matrices $\# = 1, 2, \ldots$, the following meta-information is stored.

`mat`#	the name of the control total matrix
`totalof`#	the multiplier variable (matrix' `coleq`
`over`#	the margin associated with the matrix (i.e., the categories represented by the columns)

Also, `ipfraking` stores the notes regarding the control matrices used, and which of the margins did not match the control totals, if any. See [D] **notes**.

2.3 Excel reports on raked weights: `ipfraking_report`

`ipfraking_report using` *filename* `, raked_weight(`*varname*`)` [
`matrices(`*namelist*`) by(`*varlist*`) xls replace force`]

The utility command `ipfraking_report` produces a detailed report describing the raked weights, and places it into *filename*`.dta` file (or, if `xls` option is specified, both *filename*`.dta` and *filename*`.xls` files).

Along the way, `ipfraking_report` runs a regression of the log raking ratio w_{3j}/w_{1j} on the calibration variables. This regression is expected to have R^2 very close to 1, and the regression coefficients provide insights regarding which categories received greater vs. smaller adjustments.

(*Continued on next page*)

```
. ipfraking_report using rakedwgt3-report, raked_weight(rakedwgt3) replace by(_one)
Margin variable sex_age (total variable: _one; categories: 11 12 13 21 22 23).
Margin variable region (total variable: _one; categories: 1 2 3 4).
Margin variable race (total variable: _one; categories: 1 2 3).
Auxiliary variable _one (categories: 1).
file rakedwgt3-report.dta saved
      Source |       SS           df       MS      Number of obs   =    10,351
-------------+----------------------------------   F(10, 10340)    >  99999.00
       Model |  2086.13859        10  208.613859   Prob > F        =    0.0000
    Residual |   .78315703    10,340  .000075741   R-squared       =    0.9996
-------------+----------------------------------   Adj R-squared   =    0.9996
       Total |  2086.92175    10,350  .201634952   Root MSE        =     .0087

------------------------------------------------------------------------------
     __000003 |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
      sex_age |
          11  |   .0644365   .0002775   232.21   0.000     .0638925    .0649804
          12  |   .4545577   .0003154  1441.25   0.000     .4539395     .455176
          13  |   .6782466   .0002804  2418.71   0.000     .6776969    .6787963
          22  |   .3966406   .0003049  1300.84   0.000     .3960429    .3972383
          23  |   .7304392   .0002726  2679.97   0.000     .7299049    .7309734
             |
      region |
          NE  |  -.4455127   .0002536 -1756.49   0.000    -.4460099   -.4450155
          MW  |  -.4428144   .0002335 -1896.53   0.000    -.4432721   -.4423567
           W  |  -.6672675   .0002407 -2772.21   0.000    -.6677393   -.6667957
             |
        race |
       Black  |   .3360321   .0002848  1180.08   0.000     .3354739    .3365902
       Other  |   1.613276   .0006303  2559.34   0.000     1.612041    1.614512
             |
       _cons |   .5864801   .0002455  2388.48   0.000     .5859988    .5869614
------------------------------------------------------------------------------

Raking adjustments for sex_age variable:
  the smallest was       1.798 for category 21 (21)
  the greatest was       3.732 for category 23 (23)
Raking adjustments for region variable (1=NE, 2=MW, 3=S, 4=W):
  the smallest was       0.922 for category 4 (W)
  the greatest was       1.798 for category 3 (S)
Raking adjustments for race variable (1=white, 2=black, 3=other):
  the smallest was       1.798 for category 1 (White)
  the greatest was       9.023 for category 3 (Other)
```

Options of `ipfraking_report`

`raked_weight(`*varname*`)` specifies the name of the raked weight variable to create the report for. This is a required option.

`matrices(`*namelist*`)` specifies a list of matrices (formatted as the matrices supplied to `ctotal()` option of `ipfraking`) to produce weighting reports for. In particular, the variables and their categories are picked up from these matrices; and the control totals/proportions are compared to those defined by the weight being reported on.

`by(`*varlist*`)` specifies a list of additional variables for which the weights are to be tabulated in the raking weights report. The difference with the `matrices()` option is that the control totals for these variables may not be known (or may not be relevant). In particular, `by(_one)`, where `_one` is identically one, will produce the overall report.

`xls` requests exporting the report to an Excel file.

`replace` specifies that the files produced by `ipfraking_report` (i.e., the `.dta` and the .xls file if `xls` option is specified) should be overwritten.

`force` requires that a variable that may be found repeatedly (between the calibration variables supplied originally to `ipfraking`, the variables found in the independent total `matrices()`, and the variables without the control totals provided in `by()` option) is processed every time it is encountered. (Otherwise, it is only processed once.)

Variables in the raking report

The following variables are saved in the raking report.

(*Continued on next page*)

Variable name	Definition
`Weight_Variable`	The name of the weight variable, `generate()`
`C_Total_Margin_Variable_Name`	The name of the control margin, `rowname` of the corresponding `ctotal()` matrix
`C_Total_Margin_Variable_Label`	The label of the control margin variable
`Variable_Class`	The role of the variable in the report: Raking margin: a variable used as a calibration margin (picked up automatically from the `ctotal()` matrix, provided `meta` option was specified) Other known target: supplied with `matrices()` option of `ipfraking_report` Auxiliary variable: additional variable supplied with `by()` option of `ipfraking_report`
`C_Total_Arg_Variable_Name`	The name of the multiplier variable
`C_Total_Arg_Variable_Label`	The label of the multiplier variable
`C_Total_Margin_Category_Number`	Numeric value of the control total category
`C_Total_Margin_Category_Label`	Label of the control total category
`Category_Total_Target`	The control total to be calibrated to (the specific entry in the `ctotal()` matrix)
`Category_Total_Prop`	Control total proportion (the ratio of the specific entry in the `ctotal()` matrix to the matrix total)
`Unweighted_Count`	Number of sample observations in the category
`Unweighted_Prop`	Unweighted proportion
`Unweighted_Prop_Discrep`	Difference `Unweighted_Prop - Category_Total_Prop`
`Category_Total_SRCWGT`	Weighted category total, with source weight
`Category_Prop_SRCWGT`	Weighted category proportion, with source weight
`Category_Total_Discrep_SRCWGT`	Difference `Category_Total_SRCWGT -` `- Category_Total_Target`
`Category_Prop_Discrep_SRCWGT`	Difference `Category_Prop_SRCWGT -` `- Category_Total_Prop`
`Category_RelDiff_SRCWGT`	`reldif(Category_Total_SRCWGT, Category_Total_Target)`
`Overall_Total_SRCWGT`	Sum of source weights
`Source`	The name of the matrix from which the totals were obtained
`Comment`	Placeholder for comments, to be entered during manual review

For each of the input weights (`SRCWGT` suffix), raked weights (`RKDWGT` suffix) and raking ratio (the ratio of raked and input weights, `RKDRATIO` suffix), the following summaries are provided.

Variable name	Definition
`Min_`*`WEIGHT`*	Min of source weights
`P25_`*`WEIGHT`*	25th percentile of source weights
`P50_`*`WEIGHT`*	Median of source weights
`P75_`*`WEIGHT`*	75th percentile of source weights
`Max_`*`WEIGHT`*	Max of source weights
`Mean_`*`WEIGHT`*	Mean of source weights
`SD_`*`WEIGHT`*	Standard deviation of source weights
`DEFF_`*`WEIGHT`*	Apparent UWE DEFF of source weights

Example

```
. use rakedwgt3-report, clear
(Weighting report on rakedwgt3)
. list C_Total_Margin_Variable_Name C_Total_Margin_Category_Label ///
>        Category_Total_Target Category_Total_RKDWGT DEFF_SRCWGT DEFF_RKDWGT , ///
>        sepby( C_Total_Margin_Variable_Name )

     +-------------------------------------------------------------------------+
     | C_Tota..   ~y_Label   Categor~t   Categor..   DEFF_SR~T   DEFF_RK~T |
     |-------------------------------------------------------------------------|
  1. |  sex_age         11    41995394    41995394   1.2148059   1.6259899 |
  2. |  sex_age         12    42148662    42148662   1.2462168   1.5716613 |
  3. |  sex_age         13    26515340    26515340   1.2241095   1.5460785 |
  4. |  sex_age         21    41164255    41164255   1.2325105   1.5639529 |
  5. |  sex_age         22    43697440    43697440   1.1937826   1.5175312 |
  6. |  sex_age         23    32773080    32773080    1.233902    1.664307 |
     |-------------------------------------------------------------------------|
  7. |   region         NE    40679030    40679030   1.3056639   1.3657837 |
  8. |   region         MW    49205289    49205289   1.3475551   1.4909581 |
  9. |   region          S    85024007    85024006   1.4950056   1.4912995 |
 10. |   region          W    53385843    53385844    1.459859   2.3772667 |
     |-------------------------------------------------------------------------|
 11. |     race      White   1.784e+08   1.784e+08   1.4059259   1.4337901 |
 12. |     race      Black    29856865    29856865   1.5173846   1.5092533 |
 13. |     race      Other    20053682    20053682   1.3179136   1.2264706 |
     |-------------------------------------------------------------------------|
 14. |     _one          1           .   2.283e+08   1.4164382   1.7349278 |
     +-------------------------------------------------------------------------+
```

Functionality of `ipfraking_report` is aimed at the manual review of its reporting of the categories that differ the most in the output, and the resulting report file in Excel, although for some aspects of automated quality control, it will be useful, as well.

2.4 Collapsing weighting cells: `wgtcellcollapse`

An additional new component of `ipfraking` package is a tool to semi-automatically collapse weighting cells, in order to achieve some minimal sample size.

`wgtcellcollapse` *`task`* [*if*] [*in*] , [`task_options`]

where *task* is one of:

`define` to define collapsing rules explicitly

`sequence` to create collapsing rules for a sequence of categories

`report` to list the currently defined collapsing rules

`candidate` to find rules applicable to a given category

`collapse` to perform cell collapsing

`label` to label collapsed cells using the original labels after `wgtcellcollapse collapse`

2.5 Syntax of `wgtcellcollapse report`

`wgtcellcollapse report , variables(`*varlist*`)` [`break`]

`variables(`*varlist*`)` is the list of variables for which the collapsing rule are to be reported

`break` requires `wgtcellcollapse report` to exit with error when technical inconsistencies are encountered

2.6 Syntax of `wgtcellcollapse define`

`wgtcellcollapse define , variables(`*varlist*`)` [`from(`*numlist*`) to(#)` `label(`*string*`) max(#) clear`]

`variables(`*varlist*`)` is the list of variables for which the collapsing rule can be used

`from(`*numlist*`)` is the list of categories that can be collapsed according to this rule

`to(#)` is the numeric value of the new, collapsed category

`label(`*string*`)` is the value label to be attached to the new, collapsed category

`max(#)` overrides the automatically determined max value of the collapsed variable

`clear` clears all the rules currently defined

Individual collapsing rules can be defined as follows.

```
.
. clear
.
. set obs 4
number of observations (_N) was 0, now 4
.
. gen byte x = _n
.
. label define x_lbl 1 "One" 2 "Two" 3 "Three" 4 "Four"
.
. label values x x_lbl
```

```
.
. wgtcellcollapse define, var(x) from(1 2 3) to(123)
.
. wgtcellcollapse report, var(x)
Rule (1): collapse together
  x == 1 (One)
  x == 2 (Two)
  x == 3 (Three)
  into x == 123 (123)
  WARNING: unlabeled value x == 123

.
```

Note how `break` option of `wgtcellcollapse` can be used to abort the execution when technical deficiencies in the rules or in the data are encountered. In this case, the label of the new category 123 was not defined, and this is considered a serious enough deficiency to stop.

```
.
. wgtcellcollapse report, var(x) break
Rule (1): collapse together
  x == 1 (One)
  x == 2 (Two)
  x == 3 (Three)
  into x == 123 (123)
  ERROR: unlabeled value x == 123
assertion is false
r(9);
.
. wgtcellcollapse define, var(x) clear
.
. wgtcellcollapse define, var(x) from(1 2 3) to(123) label("One through three")
.
. wgtcellcollapse report, var(x) break
Rule (1): collapse together
  x == 1 (One)
  x == 2 (Two)
  x == 3 (Three)
  into x == 123 (One through three)

.
```

2.7 Syntax of `wgtcellcollapse sequence`

`wgtcellcollapse sequence , variables(`*varlist*`) from(`*numlist*`) depth(`#`)`

`variables(`*varlist*`)` is the list of variables for which the collapsing rule can be used

`from(`*numlist*`)` is the sequence of values from which the plausible subsequences can be constructed

`depth(`#`)` is the maximum number of the original categories that can be collapsed

Moderate length sequences of collapsing categories can be defined as follows.

```
.
. clear
.
. set obs 4
number of observations (_N) was 0, now 4
.
. gen byte x = _n
.
. label define x_lbl 1 "One" 2 "Two" 3 "Three" 4 "Four"
.
. label values x x_lbl
.
. wgtcellcollapse sequence, var(x) from(1 2 3 4) depth(3)
.
. wgtcellcollapse report, var(x)
Rule (1): collapse together
  x == 1 (One)
  x == 2 (Two)
  into x == 212 (One to Two)
Rule (2): collapse together
  x == 2 (Two)
  x == 3 (Three)
  into x == 223 (Two to Three)
Rule (3): collapse together
  x == 3 (Three)
  x == 4 (Four)
  into x == 234 (Three to Four)
Rule (4): collapse together
  x == 1 (One)
  x == 2 (Two)
  x == 3 (Three)
  into x == 313 (One to Three)
Rule (5): collapse together
  x == 1 (One)
  x == 223 (Two to Three)
  into x == 313 (One to Three)
Rule (6): collapse together
  x == 3 (Three)
  x == 212 (One to Two)
  into x == 313 (One to Three)
Rule (7): collapse together
  x == 2 (Two)
  x == 3 (Three)
  x == 4 (Four)
  into x == 324 (Two to Four)
Rule (8): collapse together
  x == 2 (Two)
  x == 234 (Three to Four)
  into x == 324 (Two to Four)
Rule (9): collapse together
  x == 4 (Four)
  x == 223 (Two to Three)
```

```
    into x == 324 (Two to Four)
.
```

When creating sequential collapses, `wgtcellcollapse sequence` uses the following mnemonics in creating the new labels:

- First comes the length of the collapsed subsequence (up to `depth(`#`)`).
- Then comes the starting value of the category in the subsequence (padded by zeroes as needed).
- Then comes the ending value of the category in the subsequence (padded by zeroes as needed).

In the example above, rules 7 through 9 lead to collapsing into the new category 324. This should be interpreted as "the subsequence of length 3 that starts with category 2 and ends with category 4". A numeric value of the collapsed category that reads like 50412 means "the subsequence of length 5 that starts with category 4 and ends with category 12".

Note that `wgtcellcollapse sequence` respects the order in which the categories are supplied in the `from()` option, and does not sort them.

2.8 Syntax of `wgtcellcollapse candidate`

`wgtcellcollapse candidate , variable(`*varname*`) category(`#`)` [max #]

`variable(`*varname*`)` is the variable whose collapsing rules are to be searched

`category(`#`)` is the category for which the candidate rules are to be identified

`max(`#`)` is the maximum value of the categories in the candidate rules to be returned

The rules found are quietly returned through the mechanism of `sreturn`, see [P] **return**, as they are intended to be stay in memory sufficiently long for `wgtcellcollapse collapse` to evaluate each rule.

```
.
. wgtcellcollapse candidate, var(x) cat(2)
.
. sreturn list
macros:
           s(goodrule) : "1 2 4 7 8"
              s(rule8) : "2:234=324"
              s(rule7) : "2:3:4=324"
              s(rule4) : "1:2:3=313"
              s(rule2) : "2:3=223"
              s(rule1) : "1:2=212"
                s(cat) : "2"
                  s(x) : "x"
```

```
.
. wgtcellcollapse candidate, var(x) cat(2) max(9)
.
. sreturn list
macros:
          s(goodrule) : "1 2 4 7"
             s(rule7) : "2:3:4=324"
             s(rule4) : "1:2:3=313"
             s(rule2) : "2:3=223"
             s(rule1) : "1:2=212"
               s(cat) : "2"
                 s(x) : "x"
.
. wgtcellcollapse candidate, var(x) cat(212)
.
. sreturn list
macros:
          s(goodrule) : "6"
             s(rule6) : "3:212=313"
               s(cat) : "212"
                 s(x) : "x"
.
. wgtcellcollapse candidate, var(x) cat(55)
.
. sreturn list
macros:
               s(cat) : "55"
                 s(x) : "x"
.
```

In the second call to the option `max(9)` was used to restrict the returned rules to the rules that deal with the original categories only. In the third call, a list of rules that involve a collapsed category `cat(212)` was requested. Requests for nonexisting categories are not considered errors, but simply produce empty lists of “good rules”

2.9 Syntax of `wgtcellcollapse collapse`

`wgtcellcollapse collapse` [*if*] [*in*] `, variables(`*varlist*`) mincellsize(`#`)`
`saving(`*dofile_name*`)` [`generate(`*newvarname*`) replace append`
`feed(`*varname*`) strict sort(`*varlist*`) run maxpass(`#`) maxcategory(`#`)`
`zeroes(`*numlist*`) greedy`]

`variables(`*varlist*`)` provides the list of variables whose cells are to be collapsed. When more than one variable is specified, `wgtcellcollapse collapse` proceeds from right to left, i.e., first attempts to collapse the rightmost variable.

`mincellsize(`#`)` specifies the minimum cell size for the collapsed cells. For most weighting purposes, values of 30 to 50 can be recommended.

`generate(`*newvarname*`)` specifies the name of the collapsed variable to be created.

`feed(`*varname*`)` provides the name of an already existing collapsed variable.

`strict` modifies the behavior of `wgtcellcollapse collapse` so that only collapsing rules for which all participating categories have nonzero counts are utilized.

`sort(`*varlist*`)` sorts the data set before proceeding to collapse the cell. The default sort order is in terms of the values of the collapsed variable. A different sort order may produce a different set of collapsed cell when cells are tied on size.

`maxpass(`*#*`)` specifies the maximum number of passes through the data set. The default value is 10000.

`maxcategory(`*#*`)` is the maximum category value of the variable being collapsed. It is passed to the internal calls to `wgtcellcollapse candidate`, see above.

`zeroes(`*numlist*`)` provides a list of the categories of the collapsed variable that may have zero counts in the data.

`greedy` modifies the behavior `wgtcellcollapse collapse` to prefer the rules that collapse the maximum number of categories.

Options to deal with the do-file to write the collapsing code to:

`saving(`*dofile_name*`)` specifies the name of the do-file that will contain the cell collapsing code.

`replace` overwrites the do-file if one exists.

`append` appends the code to the existing do-file.

`run` specifies that the do-file created is run upon completion. This option is typically specified with most runs.

The primary intent of `wgtcellcollapse collapse` is to create the code that can be utilized for both the survey data file and the population targets data file that are assumed to have identically named variables. Thus it does not only manipulate the data in the memory and collapses the cells, but also produces the do-file code that can be recycled. To that effect, when a do-file is created with the `replace` and `saving()` options, the user needs to specify `generate()` option to provide the name of the collapsed variable; and when the said do-file is appended with the the `replace` and `saving()` options, the name of that variable is provided with the `feed()` option.

The algorithm `wgtcellcollapse collapse` uses to identify the cells to be collapsed is a variation of greedy search. It first identifies the cells with the lowest (positive) counts; finds the candidate rules for the variable(s) to be collapsed; and uses the rule that has produces the smallest size of the collapsed cell across all applicable rules. So when it finds several rules that are applicable to the cell being currently processed that has a size of 5, and the candidate rules produces cells of sizes 7, 10 and 15, `wgtcellcollapse collapse` will use the rule that produces the cell of size 7. The algorithm runs until all cells have sizes of at least `mincellsize(`*#*`)` or until `maxpass(`*#*`)` passes through the data are executed. It is a pretty dumb algorithm, actually, and it fails quite often. For that reason, a number of hooks are provided to modify its behavior.

Hint 1. Since `wgtcellcollapse collapse` works with the sample data, it will not be able to identify categories that are not observed in the sample (e.g., rare categories), but may be present in the population. This will lead to errors at the raking stage, when the control total matrices have more categories than the data, forcing `ipfraking` to stop. To help with that, the option `zeroes()` allows the user to pass the categories of the variables that are known to exist in the population but not in the sample.

Hint 2. The behavior of `wgtcellcollapse collapse, zeroes()` may still not be satisfactory. As it evaluates the sample sizes of the collapsed cells across a number of candidate rules that involve zero cells, it will probably pick up the rule with lowest number, and that rule may as well leave some other candidate rules with zero cells untouched. This may create problems when `wgtcellcollapse collapse` returns to those untouched cells, and looks for the existing cells to collapse them with, creating collapsing rules with breaks in the sequences. To improve upon that behavior, option `greedy` makes `wgtcellcollapse collapse` look for a rule that has many categories as possible, thus collapsing as many categories with zero counts in one swipe as it can.

Hint 3. Other than for dealing with zero cells, the option `strict` should be specified most of the times. It effectively makes sure that the candidate rules correspond to the actual data.

Hint 4. Sometimes, you see some combinations in the data that seem like a nobrainer to collapse. Well, they are nobrainers to you, but `wgtcellcollapse collapse` is not that smart. If you want to guarantee some specific combination of cells to be collapsed by `wgtcellcollapse collapse`, your best bet may be to explicitly identify them with the *if* condition, and specify some ridiculously large cell size like `mincellsize(10000)` so that `wgtcellcollapse collapse` makes every possible effort to collapse those cells. It will exit with a complaint that this size could not be achieved, but hopefully the cells will be collapsed as needed.

2.10 Syntax of `wgtcellcollapse label`

`wgtcellcollapse label , variable(`*varname*`) category(`#`)` [`verbose force`]

`variable(`*varname*`)` is the collapsed variable to be labeled.

`verbose` outputs the labeling results. There may be a lot of output.

`force` instructs `wgtcellcollapse label` to only use categories present in the data.

Motivating example

Development of `wgtcellcollapse` was to address the need to collapse cells of the margin variables so that each cell has a minimum sample size; and to do so in a way that can be easily made consistent between the sample data and the population targets data. The problem arises when some of the target variables have dozens of categories, most of which have small counts. While the primary motivation comes from transportation surveys, the ideas are also applicable to other domains, e.g., continuous age variables or highly detailed race/ethnicity or region of origin categories in health or economic surveys.

The workflow of `wgtcellcollapse` is demonstrated with the following simulated data set of trips along a metro line composed of 21 stations:

```
. use stations, clear
. list station_id, sep(0)

     +--------------------+
     |         station_id |
     |--------------------|
  1. |         1. Alewife |
  2. |       2. Brookline |
  3. |       8. Carmenton |
  4. |       11. Dogville |
  5. |       18. East End |
  6. |     24. Framington |
  7. | 26. Grand Junction |
  8. |     30. High Point |
  9. |     36. Irvingtown |
 10. |    39. Johnsville |
 11. |    40. King Street |
 12. |       44. Limerick |
 13. |    47. Moscow City |
 14. |   49. Ninth Street |
 15. |   50. Ontario Lake |
 16. |53. Picadilly Square|
 17. |     55. Queens Zoo |
 18. | 60. Redline Circle |
 19. |  62. Silver Spring |
 20. |    68. Toledo Town |
 21. |  69. Union Station |
     +--------------------+
```

Turnstile counts were collected at entrances and exits of the stations, producing the following population figures.

```
. use trip_population, clear
. table board_id daypart , c(sum num_pass) cellwidth(10)

---------------------------------------------------------------------------------
                   |                             daypart
          board_id |    AM Peak      Midday  PM Reverse       Night     Weekend
-------------------+-------------------------------------------------------------
        1. Alewife |       1423          34         219         113          44
      2. Brookline |       7198         298         773         169         144
      8. Carmenton |      19254         181        3739         872         422
       11. Dogville |      12626         872        3476         769        1270
       18. East End |       2470         143        1263         145         114
     24. Framington |        634          50        1296         133          60
 26. Grand Junction |       2208         233         439          88         166
     30. High Point |       4319         424        3740         482         115
```

```
      36. Irvingtown |       1221          34         444          30         167
      39. Johnsville |         93           4          64           2           6
     40. King Street |        398          46          76          11          13
        44. Limerick |       1021          19         129          53          34
     47. Moscow City |       3300         776         984         140         301
    49. Ninth Street |         38          22         191           5           5
    50. Ontario Lake |        606          22          80          18          23
53. Picadilly Square |        642          71         622         153          69
     55. Queens Zoo  |        331          23         174          15          19
  60. Redline Circle |        270           4          63          13           3
   62. Silver Spring |       3402         240         950         206         445
     68. Toledo Town |       5085          61         744         272         112
-----------------------------------------------------------------------------------

. table alight_id daypart , c(sum num_pass) cellwidth(10)

-----------------------------------------------------------------------------------
                     |                             daypart
           alight_id |    AM Peak      Midday  PM Reverse       Night     Weekend
---------------------+-------------------------------------------------------------
        2. Brookline |         19                       3           2
        8. Carmenton |        492          18          56          23          15
        11. Dogville |       2475          42         423         153          80
        18. East End |        929          31         193          67          68
      24. Framington |        404          13          91          28          27
  26. Grand Junction |        576          20         147          42          41
      30. High Point |       2189          89         560         165         167
      36. Irvingtown |        288          10          91          21          18
      39. Johnsville |         41                      11           2           1
     40. King Street |        131           3          38           8           6
        44. Limerick |        277           9          87          20          18
     47. Moscow City |       1746          78         556         142         128
    49. Ninth Street |         88           2          25           3           4
    50. Ontario Lake |        232          11          70          14          14
53. Picadilly Square |        633          33         198          47          47
      55. Queens Zoo |        230          10          71          13          14
  60. Redline Circle |         90           2          26           3           4
   62. Silver Spring |       1134          67         369          91          85
     68. Toledo Town |       1372          81         444         112         118
   69. Union Station |      53193        3038       16007        2733        2677
-----------------------------------------------------------------------------------
```

A survey was administered to a sample of the metro line users, with the following counts of cases collected.

```
. use trip_sample, clear

. tab board_id daypart

                     |                         daypart
            board_id |   AM Peak     Midday  PM Revers      Night    Weekend |     Total
---------------------+-------------------------------------------------------+----------
          1. Alewife |        46          4         11          7          3 |        71
        2. Brookline |       236          4         35          6          7 |       288
        8. Carmenton |       653          4        184         47         24 |       912
        11. Dogville |       410         41        166         35         56 |       708
        18. East End |        85          5         64          4          4 |       162
      24. Framington |        30          3         74          3          1 |       111
  26. Grand Junction |        72         13         23          5          6 |       119
      30. High Point |       158         20        187         25         12 |       402
      36. Irvingtown |        34          2         25          1         15 |        77
      39. Johnsville |         5          1          1          0          0 |         7
```

```
        40. King Street |        17          1          2          0          1 |        21 
           44. Limerick |        28          0          9          1          3 |        41 
        47. Moscow City |        94         31         49          7         13 |       194 
       49. Ninth Street |         0          0          9          0          0 |         9 
       50. Ontario Lake |        13          1          4          1          1 |        20 
   53. Picadilly Square |        23          4         35          7          5 |        74 
         55. Queens Zoo |        10          1         14          0          2 |        27 
     60. Redline Circle |        13          0          5          0          0 |        18 
      62. Silver Spring |       106         18         38         12         17 |       191 
        68. Toledo Town |       149          6         33         11          3 |       202 
------------------------+-------------------------------------------------------+----------
                  Total |     2,182        159        968        172        173 |     3,654 

. tab alight_id daypart

                        |                        daypart
              alight_id |   AM Peak     Midday  PM Revers      Night    Weekend |     Total
------------------------+-------------------------------------------------------+----------
           2. Brookline |         1          0          0          0          0 |         1 
           8. Carmenton |        11          1          1          0          1 |        14 
           11. Dogville |        85          1         14          6          5 |       111 
           18. East End |        36          1         18          1          4 |        60 
         24. Framington |        15          1          2          2          2 |        22 
     26. Grand Junction |        15          2          8          1          1 |        27 
         30. High Point |        73          4         22         11          8 |       118 
         36. Irvingtown |         9          0          4          2          2 |        17 
        39. Johnsville |         3          0          1          0          0 |         4 
        40. King Street |         0          0          3          0          0 |         3 
           44. Limerick |        13          0          2          0          2 |        17 
        47. Moscow City |        81          6         22          6          6 |       121 
       49. Ninth Street |         3          1          1          0          0 |         5 
       50. Ontario Lake |         2          0          1          2          1 |         6 
   53. Picadilly Square |        23          1          8          3          2 |        37 
         55. Queens Zoo |         6          0          5          1          0 |        12 
     60. Redline Circle |         5          0          0          0          0 |         5 
      62. Silver Spring |        49          0         19          3          9 |        80 
        68. Toledo Town |        43          3         24          6          7 |        83 
      69. Union Station |     1,709        138        813        128        123 |     2,911 
------------------------+-------------------------------------------------------+----------
                  Total |     2,182        159        968        172        173 |     3,654 
```

As only 3654 surveys were collected from a total of 96783 riders, we would reasonably expect that things do not align quite well. We expect weighting to correct for at least a portion of that nonresponse. The data available for calibration includes the population turnstile counts listed above, and we will produce interactions of daypart and station that will serve as two weighting margins (one for the stations where the metro users boarded, and one for the stations where they got off).

First, we need to define the weighting rules. In this case, the stations are numbered sequentially, with the northernmost, say, station Alewife being number 3, and the southernmost station, Union Station, where everybody gets off to rush to their city jobs or attractions, being number 73. Below, we create a list of stations and provide it to `wgtcellcollapse sequence`. We would be collapsing stations along the line, with the expectation that travelers boarding or leaving at adjacent stations within the same day part are more similar to one another than the travelers boarding or leaving a particular station at different times of the day. Still, some collapsing rules can be defined for the daypart variable as well — mostly because wgtcellcollapse collapse expects all variables to have collapsing rules defined.

```
. use trip_sample, clear
. wgtcellcollapse sequence , var(daypart) from(2 3 4) depth(3)
. levelsof board_id, local(stations_on)
1 2 8 11 18 24 26 30 36 39 40 44 47 49 50 53 55 60 62 68
. levelsof alight_id, local(stations_off)
2 8 11 18 24 26 30 36 39 40 44 47 49 50 53 55 60 62 68 69
. local all_stations : list stations_on | stations_off
. * relies on stations being in sequential order!!!
. wgtcellcollapse sequence , var(board_id alight_id) from(`all_stations´) depth(20)
. save trip_sample_rules, replace
file trip_sample_rules.dta saved
```

The number of collapsing rules for variables board_id and alight_id is 2961 each.

The first pass of weight collapse and raking

Let us say that we want to define weighting cells with at least 20 cases in each. We will thus start with weighting cells defined as station-by-daypart interaction, and collapsing stations within daypart to achieve the cell sizes of at least 20 cases. Here is what a simple run of wgtcellcollapse collapse might look like.

```
. use trip_sample_rules, clear
. wgtcellcollapse collapse, variables(daypart board_id) mincellsize(20) ///
>         generate(dpston1) saving(dpston1.do) replace run
(note: file dpston1.do not found)
Pass 0 through the data...
  smallest count = 1 in the cell      2000039
  Invoking rule 39:40=23940
  replace dpston1 = 2023940 if inlist(dpston1, 2000039, 2000040)
Pass 1 through the data...
  smallest count = 1 in the cell      2000050
  Invoking rule 50:53=25053
  replace dpston1 = 2025053 if inlist(dpston1, 2000050, 2000053)
Pass 2 through the data...
  smallest count = 1 in the cell      2000055
  Invoking rule 55:25053=35055
  replace dpston1 = 2035055 if inlist(dpston1, 2000055, 2025053)
Pass 3 through the data...
  smallest count = 1 in the cell      3000039
  Invoking rule 39:40=23940
  replace dpston1 = 3023940 if inlist(dpston1, 3000039, 3000040)
Pass 4 through the data...
  smallest count = 1 in the cell      4000036
  Invoking rule 36:39:40:44=43644
  replace dpston1 = 4043644 if inlist(dpston1, 4000036, 4000039, 4000040, 4000044)
Pass 5 through the data...
  smallest count = 1 in the cell      4000050
  Invoking rule 43644:24749:50=73650
  replace dpston1 = 4073650 if inlist(dpston1, 4043644, 4024749, 4000050)
Pass 6 through the data...
  smallest count = 1 in the cell      5000024
```

```
    Invoking rule 18:24=21824
    replace dpston1 = 5021824 if inlist(dpston1, 5000018, 5000024)
  Pass 7 through the data...
    smallest count = 1 in the cell      5000040
    Invoking rule 40:44=24044
    replace dpston1 = 5024044 if inlist(dpston1, 5000040, 5000044)
  Pass 8 through the data...
    smallest count = 1 in the cell      5000050
    Invoking rule 24044:24749:50=54050
    replace dpston1 = 5054050 if inlist(dpston1, 5024044, 5024749, 5000050)
  Pass 9 through the data...
    smallest count = 2 in the cell      2000036
    Invoking rule 36:23940=33640
    replace dpston1 = 2033640 if inlist(dpston1, 2000036, 2023940)
  Pass 10 through the data...
    smallest count = 2 in the cell      5000055
    Invoking rule 53:55=25355
    replace dpston1 = 5025355 if inlist(dpston1, 5000053, 5000055)
  Pass 11 through the data...
    smallest count = 3 in the cell      2000024
    Invoking rule 24:22630:33640=62440
    replace dpston1 = 2062440 if inlist(dpston1, 2000024, 2022630, 2033640)
  Pass 12 through the data...
    smallest count = 3 in the cell      3023940
    Invoking rule 44:23940=33944
    replace dpston1 = 3033944 if inlist(dpston1, 3000044, 3023940)
  Pass 13 through the data...
    smallest count = 3 in the cell      4000024
    Invoking rule 24:22630:73650=102450
    replace dpston1 = 4102450 if inlist(dpston1, 4000024, 4022630, 4073650)
  Pass 14 through the data...
    smallest count = 3 in the cell      5000001
    Invoking rule 1:2=20102
    replace dpston1 = 5020102 if inlist(dpston1, 5000001, 5000002)
  Pass 15 through the data...
    smallest count = 3 in the cell      5000068
    Invoking rule 25355:26062:68=55368
    replace dpston1 = 5055368 if inlist(dpston1, 5025355, 5026062, 5000068)
  Pass 16 through the data...
    smallest count = 4 in the cell      2000001
    Invoking rule 1:2=20102
    replace dpston1 = 2020102 if inlist(dpston1, 2000001, 2000002)
  Pass 17 through the data...
    smallest count = 4 in the cell      2000008
    Invoking rule 8:21118:62440=90840
    replace dpston1 = 2090840 if inlist(dpston1, 2000008, 2021118, 2062440)
  Pass 18 through the data...
    smallest count = 4 in the cell      3000050
    Invoking rule 49:50=24950
    replace dpston1 = 3024950 if inlist(dpston1, 3000049, 3000050)
  Pass 19 through the data...
    smallest count = 4 in the cell      4000018
    Invoking rule 18:24:26=31826
    replace dpston1 = 4031826 if inlist(dpston1, 4000018, 4000024, 4000026)
  Pass 20 through the data...
    smallest count = 5 in the cell      1000039
    Invoking rule 39:40=23940
    replace dpston1 = 1023940 if inlist(dpston1, 1000039, 1000040)
  Pass 21 through the data...
    smallest count = 5 in the cell      2000018
    Invoking rule 20102:20811:18=50118
    replace dpston1 = 2050118 if inlist(dpston1, 2020102, 2020811, 2000018)
  Pass 22 through the data...
    smallest count = 5 in the cell      3000060
    Invoking rule 24950:25355:60=54960
    replace dpston1 = 3054960 if inlist(dpston1, 3024950, 3025355, 3000060)
  Pass 23 through the data...
    smallest count = 5 in the cell      5021824
    Invoking rule 26:21824=31826
    replace dpston1 = 5031826 if inlist(dpston1, 5000026, 5021824)
  Pass 24 through the data...
    smallest count = 5 in the cell      5054050
    Invoking rule 54050:55368=104068
    replace dpston1 = 5104068 if inlist(dpston1, 5054050, 5055368)
  Pass 25 through the data...
    smallest count = 6 in the cell      2000068
    Invoking rule 35055:26062:68=65068
    replace dpston1 = 2065068 if inlist(dpston1, 2035055, 2026062, 2000068)
  Pass 26 through the data...
    smallest count = 6 in the cell      4000002
    Invoking rule 1:2=20102
    replace dpston1 = 4020102 if inlist(dpston1, 4000001, 4000002)
  Pass 27 through the data...
    smallest count = 6 in the cell      4102450
    Invoking rule 53:102450=112453
    replace dpston1 = 4112453 if inlist(dpston1, 4000053, 4102450)
  Pass 28 through the data...
    smallest count = 7 in the cell      4000047
    Invoking rule 47:49:50:53:55:60:62=74762
    replace dpston1 = 4074762 if inlist(dpston1, 4000047, 4000049, 4000050, 4000053, 4000055, 4000060, 400006
  > 2)
  Pass 29 through the data...
    smallest count = 9 in the cell      4031826
    Invoking rule 30:31826=41830
    replace dpston1 = 4041830 if inlist(dpston1, 4000030, 4031826)
  Pass 30 through the data...
    smallest count = 10 in the cell      1000055
    Invoking rule 55:60=25560
    replace dpston1 = 1025560 if inlist(dpston1, 1000055, 1000060)
  Pass 31 through the data...
    smallest count = 10 in the cell      5020102
    Invoking rule 8:20102=30108
    replace dpston1 = 5030108 if inlist(dpston1, 5000008, 5020102)
  Pass 32 through the data...
    smallest count = 11 in the cell      2090840
    Invoking rule 90840:44:47=110847
    replace dpston1 = 2110847 if inlist(dpston1, 2090840, 2000044, 2000047)
  Pass 33 through the data...
    smallest count = 11 in the cell      3000001
    Invoking rule 1:2=20102
    replace dpston1 = 3020102 if inlist(dpston1, 3000001, 3000002)
  Pass 34 through the data...
    smallest count = 11 in the cell      4000068
    Invoking rule 68:74762=84768
    replace dpston1 = 4084768 if inlist(dpston1, 4000068, 4074762)
  Pass 35 through the data...
    smallest count = 11 in the cell      5031826
    Invoking rule 30:31826=41830
    replace dpston1 = 5041830 if inlist(dpston1, 5000030, 5031826)
  Pass 36 through the data...
```

```
    smallest count = 12 in the cell      2065068
    WARNING: could not find any rules to collapse dpston1 == 2065068
  Pass 37 through the data...
    smallest count = 12 in the cell      3033944
    Invoking rule 26:23036:33944=62644
    replace dpston1 = 3062644 if inlist(dpston1, 3000026, 3023036, 3033944)
  Pass 38 through the data...
    smallest count = 13 in the cell      1000050
    Invoking rule 50:53=25053
    replace dpston1 = 1025053 if inlist(dpston1, 1000050, 1000053)
  Pass 39 through the data...
    smallest count = 13 in the cell      2000026
    Invoking rule 50118:24:26=70126
    replace dpston1 = 2070126 if inlist(dpston1, 2050118, 2000024, 2000026)
  Pass 40 through the data...
    smallest count = 13 in the cell      4020102
    Invoking rule 8:20102=30108
    replace dpston1 = 4030108 if inlist(dpston1, 4000008, 4020102)
  Pass 41 through the data...
    smallest count = 13 in the cell      4112453
    Invoking rule 11:18:112453=131153
    replace dpston1 = 4131153 if inlist(dpston1, 4000011, 4000018, 4112453)
  Pass 42 through the data...
    smallest count = 13 in the cell      5000047
    Invoking rule 36:39:40:44:47=53647
    replace dpston1 = 5053647 if inlist(dpston1, 5000036, 5000039, 5000040, 5000044, 5000047)
  Pass 43 through the data...
    smallest count = 14 in the cell      3000055
    Invoking rule 53:55=25355
    replace dpston1 = 3025355 if inlist(dpston1, 3000053, 3000055)
  Pass 44 through the data...
    smallest count = 15 in the cell      5104068
    WARNING: could not find any rules to collapse dpston1 == 5104068
  Pass 45 through the data...
    smallest count = 17 in the cell      5000062
    Invoking rule 11:18:24:26:30:36:39:40:44:47:49:50:53:55:60:62=161162
    replace dpston1 = 5161162 if inlist(dpston1, 5000011, 5000018, 5000024, 5000026, 5000030, 5000036, 500003
  > 9, 5000040, 5000044, 5000047, 5000049, 5000050, 5000053, 5000055, 5000060, 5000062)
  Pass 46 through the data...
    smallest count = 18 in the cell      2000062
    Invoking rule 30:36:39:40:44:47:49:50:53:55:60:62=123062
    replace dpston1 = 2123062 if inlist(dpston1, 2000030, 2000036, 2000039, 2000040, 2000044, 2000047, 200004
  > 9, 2000050, 2000053, 2000055, 2000060, 2000062)
  Pass 47 through the data...
    smallest count = 18 in the cell      3054960
    Invoking rule 62:54960=64962
    replace dpston1 = 3064962 if inlist(dpston1, 3000062, 3054960)
  Pass 48 through the data...
    smallest count = 22 in the cell      1023940
    Done collapsing! Exiting...

  . return list

  scalars:
           r(arg_min_id) =  1023940
                  r(min) =  22

  macros:
              r(cfailed) : "2065068,5104068"
               r(failed) : "2065068 5104068"

  . wgtcellcollapse collapse, variables(daypart alight_id) mincellsize(20) ///
  >         generate(dpstoff1) saving(dpstoff1.do) replace run
```

```
(note: file dpstoff1.do not found)
Pass 0 through the data...
  smallest count = 1 in the cell      1000002
  Invoking rule 2:8=20208
  replace dpstoff1 = 1020208 if inlist(dpstoff1, 1000002, 1000008)
Pass 1 through the data...
  smallest count = 1 in the cell      2000008
  Invoking rule 8:11=20811
  replace dpstoff1 = 2020811 if inlist(dpstoff1, 2000008, 2000011)
Pass 2 through the data...
  smallest count = 1 in the cell      2000018
  Invoking rule 18:24=21824
  replace dpstoff1 = 2021824 if inlist(dpstoff1, 2000018, 2000024)
Pass 3 through the data...
  smallest count = 1 in the cell      2000049
  Invoking rule 49:50:53=34953
  replace dpstoff1 = 2034953 if inlist(dpstoff1, 2000049, 2000050, 2000053)
Pass 4 through the data...
  smallest count = 1 in the cell      3000008
  Invoking rule 8:11=20811
  replace dpstoff1 = 3020811 if inlist(dpstoff1, 3000008, 3000011)
Pass 5 through the data...
  smallest count = 1 in the cell      3000039
  Invoking rule 39:40=23940
  replace dpstoff1 = 3023940 if inlist(dpstoff1, 3000039, 3000040)
Pass 6 through the data...
  smallest count = 1 in the cell      3000049
  Invoking rule 49:50=24950
  replace dpstoff1 = 3024950 if inlist(dpstoff1, 3000049, 3000050)
Pass 7 through the data...
  smallest count = 1 in the cell      4000018
  Invoking rule 18:24=21824
  replace dpstoff1 = 4021824 if inlist(dpstoff1, 4000018, 4000024)
Pass 8 through the data...
  smallest count = 1 in the cell      4000026
  Invoking rule 26:21824=31826
  replace dpstoff1 = 4031826 if inlist(dpstoff1, 4000026, 4021824)
Pass 9 through the data...
  smallest count = 1 in the cell      4000055
  Invoking rule 53:55=25355
  replace dpstoff1 = 4025355 if inlist(dpstoff1, 4000053, 4000055)
Pass 10 through the data...
  smallest count = 1 in the cell      5000008
  Invoking rule 8:11=20811
  replace dpstoff1 = 5020811 if inlist(dpstoff1, 5000008, 5000011)
Pass 11 through the data...
  smallest count = 1 in the cell      5000026
  Invoking rule 24:26=22426
  replace dpstoff1 = 5022426 if inlist(dpstoff1, 5000024, 5000026)
Pass 12 through the data...
  smallest count = 1 in the cell      5000050
  Invoking rule 50:53=25053
  replace dpstoff1 = 5025053 if inlist(dpstoff1, 5000050, 5000053)
Pass 13 through the data...
  smallest count = 2 in the cell      1000050
  Invoking rule 49:50=24950
  replace dpstoff1 = 1024950 if inlist(dpstoff1, 1000049, 1000050)
Pass 14 through the data...
  smallest count = 2 in the cell      2000026
  Invoking rule 26:21824=31826
```

```
    replace dpstoff1 = 2031826 if inlist(dpstoff1, 2000026, 2021824)
  Pass 15 through the data...
    smallest count = 2 in the cell      2020811
    Invoking rule 20811:31826=50826
    replace dpstoff1 = 2050826 if inlist(dpstoff1, 2020811, 2031826)
  Pass 16 through the data...
    smallest count = 2 in the cell      2034953
    Invoking rule 47:34953=44753
    replace dpstoff1 = 2044753 if inlist(dpstoff1, 2000047, 2034953)
  Pass 17 through the data...
    smallest count = 2 in the cell      3000024
    Invoking rule 24:26=22426
    replace dpstoff1 = 3022426 if inlist(dpstoff1, 3000024, 3000026)
  Pass 18 through the data...
    smallest count = 2 in the cell      3000044
    Invoking rule 44:23940=33944
    replace dpstoff1 = 3033944 if inlist(dpstoff1, 3000044, 3023940)
  Pass 19 through the data...
    smallest count = 2 in the cell      3024950
    Invoking rule 53:24950=34953
    replace dpstoff1 = 3034953 if inlist(dpstoff1, 3000053, 3024950)
  Pass 20 through the data...
    smallest count = 2 in the cell      4000036
    Invoking rule 36:39:40:44:47=53647
    replace dpstoff1 = 4053647 if inlist(dpstoff1, 4000036, 4000039, 4000040, 4000044, 4000047)
  Pass 21 through the data...
    smallest count = 2 in the cell      4000050
    Invoking rule 50:53:55:60:62=55062
    replace dpstoff1 = 4055062 if inlist(dpstoff1, 4000050, 4000053, 4000055, 4000060, 4000062)
  Pass 22 through the data...
    smallest count = 2 in the cell      5000036
    Invoking rule 36:39:40:44=43644
    replace dpstoff1 = 5043644 if inlist(dpstoff1, 5000036, 5000039, 5000040, 5000044)
  Pass 23 through the data...
    smallest count = 3 in the cell      1000039
    Invoking rule 36:39=23639
    replace dpstoff1 = 1023639 if inlist(dpstoff1, 1000036, 1000039)
  Pass 24 through the data...
    smallest count = 3 in the cell      2000068
    Invoking rule 30:36:39:40:44:47:49:50:53:55:60:62:68=133068
    replace dpstoff1 = 2133068 if inlist(dpstoff1, 2000030, 2000036, 2000039, 2000040, 2000044, 2000047, 2000
  > 049, 2000050, 2000053, 2000055, 2000060, 2000062, 2000068)
  Pass 25 through the data...
    smallest count = 3 in the cell      5022426
    Invoking rule 18:22426=31826
    replace dpstoff1 = 5031826 if inlist(dpstoff1, 5000018, 5022426)
  Pass 26 through the data...
    smallest count = 3 in the cell      5025053
    Invoking rule 47:49:25053=44753
    replace dpstoff1 = 5044753 if inlist(dpstoff1, 5000047, 5000049, 5025053)
  Pass 27 through the data...
    smallest count = 4 in the cell      3000036
    Invoking rule 36:33944=43644
    replace dpstoff1 = 3043644 if inlist(dpstoff1, 3000036, 3033944)
  Pass 28 through the data...
    smallest count = 4 in the cell      4025355
    Invoking rule 25355:60:62:68=55368
    replace dpstoff1 = 4055368 if inlist(dpstoff1, 4025355, 4000060, 4000062, 4000068)
  Pass 29 through the data...
    smallest count = 4 in the cell      4031826
    Invoking rule 11:31826=41126
    replace dpstoff1 = 4041126 if inlist(dpstoff1, 4000011, 4031826)
  Pass 30 through the data...
    smallest count = 4 in the cell      5043644
    Invoking rule 30:43644=53044
    replace dpstoff1 = 5053044 if inlist(dpstoff1, 5000030, 5043644)
  Pass 31 through the data...
    smallest count = 5 in the cell      1000060
    Invoking rule 24950:25355:60=54960
    replace dpstoff1 = 1054960 if inlist(dpstoff1, 1024950, 1025355, 1000060)
  Pass 32 through the data...
    smallest count = 5 in the cell      3000055
    Invoking rule 55:34953=44955
    replace dpstoff1 = 3044955 if inlist(dpstoff1, 3000055, 3034953)
  Pass 33 through the data...
    smallest count = 5 in the cell      4055062
    Invoking rule 55062:68:69=75069
    replace dpstoff1 = 4075069 if inlist(dpstoff1, 4055062, 4000068, 4000069)
  Pass 34 through the data...
    smallest count = 6 in the cell      1000055
    Invoking rule 53:55=25355
    replace dpstoff1 = 1025355 if inlist(dpstoff1, 1000053, 1000055)
  Pass 35 through the data...
    smallest count = 6 in the cell      2050826
    Invoking rule 50826:133068=180868
    replace dpstoff1 = 2180868 if inlist(dpstoff1, 2050826, 2133068)
  Pass 36 through the data...
    smallest count = 6 in the cell      5020811
    Invoking rule 20811:31826=50826
    replace dpstoff1 = 5050826 if inlist(dpstoff1, 5020811, 5031826)
  Pass 37 through the data...
    smallest count = 7 in the cell      5000068
    Invoking rule 62:68=26268
    replace dpstoff1 = 5026268 if inlist(dpstoff1, 5000062, 5000068)
  Pass 38 through the data...
    smallest count = 8 in the cell      2044753
    WARNING: could not find any rules to collapse dpstoff1 == 2044753
  Pass 39 through the data...
    smallest count = 8 in the cell      4053647
    Invoking rule 30:53647=63047
    replace dpstoff1 = 4063047 if inlist(dpstoff1, 4000030, 4053647)
  Pass 40 through the data...
    smallest count = 9 in the cell      5044753
    Invoking rule 53044:44753=93053
    replace dpstoff1 = 5093053 if inlist(dpstoff1, 5053044, 5044753)
  Pass 41 through the data...
    smallest count = 10 in the cell      1054960
    Invoking rule 62:54960=64962
    replace dpstoff1 = 1064962 if inlist(dpstoff1, 1000062, 1054960)
  Pass 42 through the data...
    smallest count = 10 in the cell      3022426
    Invoking rule 18:22426=31826
    replace dpstoff1 = 3031826 if inlist(dpstoff1, 3000018, 3022426)
  Pass 43 through the data...
    smallest count = 10 in the cell      3043644
    Invoking rule 30:43644=53044
    replace dpstoff1 = 3053044 if inlist(dpstoff1, 3000030, 3043644)
  Pass 44 through the data...
    smallest count = 10 in the cell      4041126
    Invoking rule 41126:63047=101147
```

```
    replace dpstoff1 = 4101147 if inlist(dpstoff1, 4041126, 4063047)
  Pass 45 through the data...
    smallest count = 10 in the cell      4055368
    WARNING: could not find any rules to collapse dpstoff1 == 4055368
  Pass 46 through the data...
    smallest count = 12 in the cell      1020208
    Invoking rule 20208:21118:24=50224
    replace dpstoff1 = 1050224 if inlist(dpstoff1, 1020208, 1021118, 1000024)
  Pass 47 through the data...
    smallest count = 12 in the cell      1023639
    Invoking rule 23639:40:44=43644
    replace dpstoff1 = 1043644 if inlist(dpstoff1, 1023639, 1000040, 1000044)
  Pass 48 through the data...
    smallest count = 13 in the cell      2180868
    Invoking rule 69:180868=190869
    replace dpstoff1 = 2190869 if inlist(dpstoff1, 2000069, 2180868)
  Pass 49 through the data...
    smallest count = 13 in the cell      5050826
    Invoking rule 50826:93053=140853
    replace dpstoff1 = 5140853 if inlist(dpstoff1, 5050826, 5093053)
  Pass 50 through the data...
    smallest count = 15 in the cell      1000026
    Invoking rule 26:50224=60226
    replace dpstoff1 = 1060226 if inlist(dpstoff1, 1000026, 1050224)
  Pass 51 through the data...
    smallest count = 15 in the cell      3020811
    Invoking rule 20811:31826=50826
    replace dpstoff1 = 3050826 if inlist(dpstoff1, 3020811, 3031826)
  Pass 52 through the data...
    smallest count = 15 in the cell      3044955
    Invoking rule 44955:60:62=64962
    replace dpstoff1 = 3064962 if inlist(dpstoff1, 3044955, 3000060, 3000062)
  Pass 53 through the data...
    smallest count = 16 in the cell      5026268
    Invoking rule 69:26268=36269
    replace dpstoff1 = 5036269 if inlist(dpstoff1, 5000069, 5026268)
  Pass 54 through the data...
    smallest count = 22 in the cell      3000047
    Done collapsing! Exiting...
. return list

scalars:
         r(arg_min_id) =  3000047
                r(min) =  22

macros:
            r(cfailed) : "2044753,4055368"
             r(failed) : "2044753 4055368"
```

The collapsed values of the variables `dpston` (DayPart-STation-ON) and `dpstoff` (DayPart-STation-OFF) combine the values of the parent variables. The value of `dpston==1000003` indicates `daypart==1` and station ID 3. The value of `dpston==2065270` indicates `daypart==2` and sequence of six stations from 52 to 70.

Note that `wgtcellcollapse` returns a list of the cells that it could not collapse in `r(failed)` macro (and a comma delimited list, in `f(cfailed)`). These returned values should be used in production code by making an `assert` (Gould 2003) that these macros are empty. While we know that some cell counts are less than 20, we will ignore the issue for the moment, as there are bigger concerns with the collapsed cells at the moment, as will become clear once we follow through with the workflow and attempt raking.

From the above run, `wgtcellcollapse` produced two files, one for each weighting margin, called `dpston.do` and `dpstoff.do`. An interested reader is welcome to `list` them; they contain long sequences of `replace` commands to perform the cell collapsing. The point of creating these is that they can be run on the population data to create identical categories:

```
. use trip_population, clear
. run dpston1.do
. total num_pass , over(dpston1)
Total estimation                  Number of obs   =        719
      1000001: dpston1 = 1000001
      1000002: dpston1 = 1000002
      1000008: dpston1 = 1000008
      1000011: dpston1 = 1000011
      1000018: dpston1 = 1000018
      1000024: dpston1 = 1000024
      1000026: dpston1 = 1000026
      1000030: dpston1 = 1000030
      1000036: dpston1 = 1000036
      1000044: dpston1 = 1000044
      1000047: dpston1 = 1000047
      1000049: dpston1 = 1000049
      1000062: dpston1 = 1000062
      1000068: dpston1 = 1000068
      1023940: dpston1 = 1023940
      1025053: dpston1 = 1025053
      1025560: dpston1 = 1025560
      2000011: dpston1 = 2000011
      2065068: dpston1 = 2065068
      2070126: dpston1 = 2070126
      2110847: dpston1 = 2110847
      2123062: dpston1 = 2123062
      3000008: dpston1 = 3000008
      3000011: dpston1 = 3000011
      3000018: dpston1 = 3000018
      3000024: dpston1 = 3000024
      3000030: dpston1 = 3000030
      3000036: dpston1 = 3000036
      3000047: dpston1 = 3000047
      3000068: dpston1 = 3000068
      3020102: dpston1 = 3020102
      3025355: dpston1 = 3025355
      3062644: dpston1 = 3062644
      3064962: dpston1 = 3064962
      4030108: dpston1 = 4030108
      4041830: dpston1 = 4041830
      4084768: dpston1 = 4084768
      4131153: dpston1 = 4131153
      5030108: dpston1 = 5030108
      5041830: dpston1 = 5041830
      5053647: dpston1 = 5053647
      5104068: dpston1 = 5104068
      5161162: dpston1 = 5161162
```

```
--------------------------------------------------------------
        Over |      Total   Std. Err.     [95% Conf. Interval]
-------------+------------------------------------------------
num_pass     |
     1000001 |       1423   967.7508     -476.9595    3322.959
     1000002 |       7198    4895.91     -2414.011    16810.01
     1000008 |      19254   13675.81     -7595.347    46103.35
     1000011 |      12626   9682.022     -6382.456    31634.46
     1000018 |       2470   1943.224     -1345.081    6285.081
     1000024 |        634   509.3549     -366.0031    1634.003
     1000026 |       2208   1774.996     -1276.802    5692.802
     1000030 |       4319   3665.427     -2877.235    11515.24
     1000036 |       1221   1046.817     -834.1873    3276.187
     1000044 |       1021    881.426     -709.4802     2751.48
     1000047 |       3300   2970.321     -2531.552    9131.552
     1000049 |         38         35     -30.71457    106.7146
     1000062 |       3402       3176     -2833.357    9637.357
     1000068 |       5085          .             .           .
     1023940 |        491    348.709     -193.6112    1175.611
     1025053 |       1248   765.1955     -254.2881    2750.288
     1025560 |        601     350.65     -87.42178    1289.422
     2000011 |        872    675.393     -453.9812    2197.981
     2065068 |        177   69.36426        40.819     313.181
     2070126 |        708    299.066      120.8517    1295.148
     2110847 |       1110   711.7168     -287.2948    2507.295
     2123062 |        690   412.3311     -119.5187    1499.519
     3000008 |       3739   2665.175     -1493.467    8971.467
     3000011 |       3476   2669.777     -1765.503    8717.503
     3000018 |       1263    997.019     -694.4209    3220.421
     3000024 |       1296   1032.175     -730.4418    3322.442
     3000030 |       3740   3175.677     -2494.723    9974.723
     3000036 |        444   382.5382     -307.0272    1195.027
     3000047 |        984   888.5095     -760.3871    2728.387
     3000068 |        744          .             .           .
     3020102 |        992   553.0017     -93.69354    2077.694
     3025355 |        796   573.1597     -329.2692    1921.269
     3062644 |        708   375.9286      -30.0507    1446.051
     3064962 |       1284    911.761     -506.0362    3074.036
     4030108 |       1154   529.0201      115.3888    2192.611
     4041830 |        715   393.6075     -57.75914    1487.759
     4084768 |        651   318.6053      25.49059    1276.509
     4131153 |       1169   534.4403      119.7475    2218.253
     5030108 |        610   263.2061      93.25444    1126.746
     5041830 |        455   172.8013      115.7439    794.2561
     5053647 |        474   283.9144     -83.40157    1031.402
     5104068 |        270   116.7702      40.74822    499.2518
     5161162 |       1723   909.6551     -62.90172    3508.902
--------------------------------------------------------------

. matrix dpston1 = e(b)

. matrix coleq dpston1 = _one

. matrix rownames dpston1 = dpston1

. run dpstoff1.do

. total num_pass , over(dpstoff1)

Total estimation                  Number of obs    =       719

      1000011: dpstoff1 = 1000011
      1000018: dpstoff1 = 1000018
      1000030: dpstoff1 = 1000030
      1000047: dpstoff1 = 1000047
      1000068: dpstoff1 = 1000068
      1000069: dpstoff1 = 1000069
      1025355: dpstoff1 = 1025355
      1043644: dpstoff1 = 1043644
      1060226: dpstoff1 = 1060226
      1064962: dpstoff1 = 1064962
      2044753: dpstoff1 = 2044753
      2190869: dpstoff1 = 2190869
      3000002: dpstoff1 = 3000002
      3000047: dpstoff1 = 3000047
      3000068: dpstoff1 = 3000068
      3000069: dpstoff1 = 3000069
      3050826: dpstoff1 = 3050826
      3053044: dpstoff1 = 3053044
      3064962: dpstoff1 = 3064962
      4000002: dpstoff1 = 4000002
      4000008: dpstoff1 = 4000008
      4000049: dpstoff1 = 4000049
      4055368: dpstoff1 = 4055368
      4075069: dpstoff1 = 4075069
      4101147: dpstoff1 = 4101147
      5000055: dpstoff1 = 5000055
      5000060: dpstoff1 = 5000060
      5036269: dpstoff1 = 5036269
      5140853: dpstoff1 = 5140853

--------------------------------------------------------------
        Over |      Total   Std. Err.     [95% Conf. Interval]
-------------+------------------------------------------------
num_pass     |
     1000011 |       2475   1468.807     -408.6691    5358.669
     1000018 |        929   360.7303      220.7878    1637.212
     1000030 |       2189   868.0319      484.8161    3893.184
     1000047 |       1746   630.7528      507.6598     2984.34
     1000068 |       1372   426.3969      534.8662    2209.134
     1000069 |      53193   15995.88      21788.72    84597.28
     1025355 |        863   233.1424      405.2777    1320.722
     1043644 |        737   159.5597      423.7407    1050.259
     1060226 |       1491   432.5204      641.8441    2340.156
     1064962 |       1544   426.4228      706.8155    2381.185
     2044753 |        124   33.15528      58.90711    189.0929
     2190869 |       3433   1082.925      1306.921    5559.079
     3000002 |          3          .             .           .
     3000047 |        556   187.4945      187.8971    924.1029
     3000068 |        444   126.0503      196.5289    691.4711
     3000069 |      16007   4295.998      7572.781    24441.22
     3050826 |        910   333.6639      254.9266    1565.073
     3053044 |        787    249.935      296.3092    1277.691
     3064962 |        759   141.1029      481.9765    1036.023
     4000002 |          2          .             .           .
     4000008 |         23          5      13.18363    32.81637
     4000049 |          3          0             .           .
     4055368 |        172   35.93822      101.4434    242.5566
     4075069 |       2841   833.6481      1204.321    4477.679
     4101147 |        648    147.123      359.1573    936.8427
     5000055 |         14   6.595453      1.051322    26.94868
     5000060 |          4          2      .0734531    7.926547
     5036269 |       2880   980.8909      954.2428    4805.757
     5140853 |        634   139.2172      360.6787    907.3213
--------------------------------------------------------------
```

```
. matrix dpstoff1 = e(b)
. matrix coleq dpstoff1 = _one
. matrix rownames dpstoff1 = dpstoff1
```

Once that is done, we can go back to the sample data and try to create raking weights:

```
. use trip_sample, clear
. run dpston1
. run dpstoff1
. gen byte _one = 1
. ipfraking [pw=_one], ctotal(dpston1 dpstoff1) gen(raked_weight1)
categories of dpston1 do not match in the control dpston1 and in the data (nolab option)
This is what dpston1 gives:
  _one:1000001 _one:1000002 _one:1000008 _one:1000011 _one:1000018 _one:1000024 _one:1000026 _one:1000030 _
> one:1000036 _one:1000044 _one:1000047 _one:1000049 _one:1000062 _one:1000068 _one:1023940 _one:1025053 _o
> ne:1025560 _one:2000011 _one:2065068 _one:2070126 _one:2110847 _one:2123062 _one:3000008 _one:3000011 _on
> e:3000018 _one:3000024 _one:3000030 _one:3000036 _one:3000047 _one:3000068 _one:3020102 _one:3025355 _one
> :3062644 _one:3064962 _one:4030108 _one:4041830 _one:4084768 _one:4131153 _one:5030108 _one:5041830 _one:
> 5053647 _one:5104068 _one:5161162
This is what I found in data:
  _one:1000001 _one:1000002 _one:1000008 _one:1000011 _one:1000018 _one:1000024 _one:1000026 _one:1000030 _
> one:1000036 _one:1000044 _one:1000047 _one:1000062 _one:1000068 _one:1023940 _one:1025053 _one:1025560 _o
> ne:2000011 _one:2065068 _one:2070126 _one:2110847 _one:2123062 _one:3000008 _one:3000011 _one:3000018 _on
> e:3000024 _one:3000030 _one:3000036 _one:3000047 _one:3000068 _one:3020102 _one:3025355 _one:3062644 _one
> :3064962 _one:4030108 _one:4041830 _one:4084768 _one:4131153 _one:5030108 _one:5041830 _one:5053647 _one:
> 5104068 _one:5161162
This is what dpston1 has that data don´t:
  _one:1000049
This is what data have that dpston1 doesn´t:

r(111);
end of do-file
r(111);
. ipfraking [pw=_one], ctotal(dpstoff1 dpston1) gen(raked_weight1)
categories of dpstoff1 do not match in the control dpstoff1 and in the data (nolab option)
This is what dpstoff1 gives:
  _one:1000011 _one:1000018 _one:1000030 _one:1000047 _one:1000068 _one:1000069 _one:1025355 _one:1043644 _
> one:1060226 _one:1064962 _one:2044753 _one:2190869 _one:3000002 _one:3000047 _one:3000068 _one:3000069 _o
> ne:3050826 _one:3053044 _one:3064962 _one:4000002 _one:4000008 _one:4000049 _one:4055368 _one:4075069 _on
> e:4101147 _one:5000055 _one:5000060 _one:5036269 _one:5140853
This is what I found in data:
  _one:1000011 _one:1000018 _one:1000030 _one:1000047 _one:1000068 _one:1000069 _one:1025355 _one:1043644 _
> one:1060226 _one:1064962 _one:2044753 _one:2190869 _one:3000047 _one:3000068 _one:3000069 _one:3050826 _o
> ne:3053044 _one:3064962 _one:4055368 _one:4075069 _one:4101147 _one:5036269 _one:5140853
This is what dpstoff1 has that data don´t:
  _one:3000002 _one:4000002 _one:4000008 _one:4000049 _one:5000055 _one:5000060
This is what data have that dpstoff1 doesn´t:

r(111);
.
```

We see that raking failed, because survey nonresponse wiped out some of the smaller stations from the sample. (Note also the informative error message with diagnostics of missing categories produced by `ipfraking`. This is a functionality added since the first 2010 publication in *The Stata Journal*. The message lists the categories found in the data, in the control totals, and in the mismatch.)

The second pass of weight collapse and raking: `zeroes()` option

Having identified the issue, we can overcome it with `zeroes()` option of `wgtcellcollapse collapse` which was developed specifically to address this issue. This option provides the list of stations that may have zero sample counts in a given daypart. For instance, notice that the sample registers only one alighting at Brookline `station==6`) in AM Peak daypart, even though there are passengers exiting in other dayparts. All in all, `wgtcellcollapse` needs to be made aware of the zero sample boardings at Johnsville (39), King Street (40), Limerick (44), Ninth Street (49), Queens Zoo (55) and Redline Circle (60); as well as zero alightings at Brookline (2), Carmenton (8), Irvingtown (36), Johnsville (39), King Street (40), Limerick (44), Moscow City (47), Ninth Street (49), Ontario Lake (50), Queens Zoo (55), Redline Circle (60), and Silver Spring (62).

```
. use trip_sample_rules, clear
. wgtcellcollapse collapse, variables(daypart board_id) mincellsize(20) ///
>         zeroes(39 40 44 49 55 60) ///
>         generate(dpston2) saving(dpston2.do) replace run
Pass 0 through the data...
  smallest count = 1 in the cell      2000039
Processing zero cells...
  Invoking rule 49:50=24950 to collapse zero cells
  replace dpston2 = 1024950 if inlist(dpston2, 1000049, 1000050)
Pass 0 through the data...
  smallest count = 1 in the cell      2000039
  Invoking rule 40:44=24044 to collapse zero cells
  replace dpston2 = 2024044 if inlist(dpston2, 2000040, 2000044)
Pass 0 through the data...
  smallest count = 1 in the cell      2000039
  Invoking rule 49:50=24950 to collapse zero cells
  replace dpston2 = 2024950 if inlist(dpston2, 2000049, 2000050)
Pass 0 through the data...
  smallest count = 1 in the cell      2000039
  Invoking rule 55:60=25560 to collapse zero cells
  replace dpston2 = 2025560 if inlist(dpston2, 2000055, 2000060)
Pass 0 through the data...
  smallest count = 1 in the cell      2000039
  Invoking rule 36:39=23639 to collapse zero cells
  replace dpston2 = 4023639 if inlist(dpston2, 4000036, 4000039)
Pass 0 through the data...
  smallest count = 1 in the cell      2000039
  Invoking rule 40:44=24044 to collapse zero cells
  replace dpston2 = 4024044 if inlist(dpston2, 4000040, 4000044)
Pass 0 through the data...
  smallest count = 1 in the cell      2000039
  Invoking rule 49:50=24950 to collapse zero cells
  replace dpston2 = 4024950 if inlist(dpston2, 4000049, 4000050)
Pass 0 through the data...
  smallest count = 1 in the cell      2000039
  Invoking rule 53:55=25355 to collapse zero cells
```

```
    replace dpston2 = 4025355 if inlist(dpston2, 4000053, 4000055)
  Pass 0 through the data...
    smallest count = 1 in the cell      2000039
    Invoking rule 24950:53:55:60=54960 to collapse zero cells
    replace dpston2 = 4054960 if inlist(dpston2, 4024950, 4000053, 4000055, 4000060)
  Pass 0 through the data...
    smallest count = 1 in the cell      2000039
    Invoking rule 39:40=23940 to collapse zero cells
    replace dpston2 = 5023940 if inlist(dpston2, 5000039, 5000040)
  Pass 0 through the data...
    smallest count = 1 in the cell      2000039
    Invoking rule 49:50=24950 to collapse zero cells
    replace dpston2 = 5024950 if inlist(dpston2, 5000049, 5000050)
  Pass 0 through the data...
    smallest count = 1 in the cell      2000039
    Invoking rule 24950:25355:60=54960 to collapse zero cells
    replace dpston2 = 5054960 if inlist(dpston2, 5024950, 5025355, 5000060)
  Pass 0 through the data...
    smallest count = 1 in the cell      2000039
  Pass 12 through the data...
    smallest count = 1 in the cell      2000039
    Invoking rule 39:24044=33944
    replace dpston2 = 2033944 if inlist(dpston2, 2000039, 2024044)
  Pass 13 through the data...
    smallest count = 1 in the cell      2024950
    Invoking rule 53:24950=34953
    replace dpston2 = 2034953 if inlist(dpston2, 2000053, 2024950)
  Pass 14 through the data...
    smallest count = 1 in the cell      2025560
    Invoking rule 34953:25560=54960
    replace dpston2 = 2054960 if inlist(dpston2, 2034953, 2025560)
  Pass 15 through the data...
    smallest count = 1 in the cell      3000039
    Invoking rule 39:40=23940
    replace dpston2 = 3023940 if inlist(dpston2, 3000039, 3000040)
  Pass 16 through the data...
    smallest count = 1 in the cell      4023639
    Invoking rule 23639:24044=43644
    replace dpston2 = 4043644 if inlist(dpston2, 4023639, 4024044)
  Pass 17 through the data...
    smallest count = 1 in the cell      4054960
    Invoking rule 47:54960=64760
    replace dpston2 = 4064760 if inlist(dpston2, 4000047, 4054960)
  Pass 18 through the data...
    smallest count = 1 in the cell      5000024
    Invoking rule 18:24=21824
    replace dpston2 = 5021824 if inlist(dpston2, 5000018, 5000024)
  Pass 19 through the data...
    smallest count = 1 in the cell      5023940
    Invoking rule 44:23940=33944
    replace dpston2 = 5033944 if inlist(dpston2, 5000044, 5023940)
  Pass 20 through the data...
    smallest count = 1 in the cell      5054960
    Invoking rule 47:54960=64760
    replace dpston2 = 5064760 if inlist(dpston2, 5000047, 5054960)
  Pass 21 through the data...
    smallest count = 2 in the cell      2000036
    Invoking rule 36:33944=43644
    replace dpston2 = 2043644 if inlist(dpston2, 2000036, 2033944)
  Pass 22 through the data...
    smallest count = 2 in the cell      4043644
    Invoking rule 24:22630:43644=72444
    replace dpston2 = 4072444 if inlist(dpston2, 4000024, 4022630, 4043644)
  Pass 23 through the data...
    smallest count = 2 in the cell      5000055
    Invoking rule 53:55=25355
    replace dpston2 = 5025355 if inlist(dpston2, 5000053, 5000055)
  Pass 24 through the data...
    smallest count = 3 in the cell      2000024
    Invoking rule 24:22630:43644=72444
    replace dpston2 = 2072444 if inlist(dpston2, 2000024, 2022630, 2043644)
  Pass 25 through the data...
    smallest count = 3 in the cell      3023940
    Invoking rule 44:23940=33944
    replace dpston2 = 3033944 if inlist(dpston2, 3000044, 3023940)
  Pass 26 through the data...
    smallest count = 3 in the cell      5000001
    Invoking rule 1:2=20102
    replace dpston2 = 5020102 if inlist(dpston2, 5000001, 5000002)
  Pass 27 through the data...
    smallest count = 3 in the cell      5000068
    Invoking rule 25355:26062:68=55368
    replace dpston2 = 5055368 if inlist(dpston2, 5025355, 5026062, 5000068)
  Pass 28 through the data...
    smallest count = 4 in the cell      2000001
    Invoking rule 1:2=20102
    replace dpston2 = 2020102 if inlist(dpston2, 2000001, 2000002)
  Pass 29 through the data...
    smallest count = 4 in the cell      2000008
    Invoking rule 8:21118:72444=100844
    replace dpston2 = 2100844 if inlist(dpston2, 2000008, 2021118, 2072444)
  Pass 30 through the data...
    smallest count = 4 in the cell      3000050
    Invoking rule 49:50=24950
    replace dpston2 = 3024950 if inlist(dpston2, 3000049, 3000050)
  Pass 31 through the data...
    smallest count = 4 in the cell      4000018
    Invoking rule 18:24:26=31826
    replace dpston2 = 4031826 if inlist(dpston2, 4000018, 4000024, 4000026)
  Pass 32 through the data...
    smallest count = 4 in the cell      5033944
    Invoking rule 26:23036:33944=62644
    replace dpston2 = 5062644 if inlist(dpston2, 5000026, 5023036, 5033944)
  Pass 33 through the data...
    smallest count = 5 in the cell      1000039
    Invoking rule 39:40=23940
    replace dpston2 = 1023940 if inlist(dpston2, 1000039, 1000040)
  Pass 34 through the data...
    smallest count = 5 in the cell      2000018
    Invoking rule 20102:20811:18=50118
    replace dpston2 = 2050118 if inlist(dpston2, 2020102, 2020811, 2000018)
  Pass 35 through the data...
    smallest count = 5 in the cell      3000060
    Invoking rule 24950:25355:60=54960
    replace dpston2 = 3054960 if inlist(dpston2, 3024950, 3025355, 3000060)
  Pass 36 through the data...
    smallest count = 5 in the cell      4072444
    Invoking rule 72444:64760=132460
    replace dpston2 = 4132460 if inlist(dpston2, 4072444, 4064760)
  Pass 37 through the data...
```

```
  smallest count = 5 in the cell      5021824
  Invoking rule 21824:62644=81844
  replace dpston2 = 5081844 if inlist(dpston2, 5021824, 5062644)
Pass 38 through the data...
  smallest count = 6 in the cell      2000068
  Invoking rule 62:68=26268
  replace dpston2 = 2026268 if inlist(dpston2, 2000062, 2000068)
Pass 39 through the data...
  smallest count = 6 in the cell      2054960
  Invoking rule 54960:26268=74968
  replace dpston2 = 2074968 if inlist(dpston2, 2054960, 2026268)
Pass 40 through the data...
  smallest count = 6 in the cell      4000002
  Invoking rule 1:2=20102
  replace dpston2 = 4020102 if inlist(dpston2, 4000001, 4000002)
Pass 41 through the data...
  smallest count = 7 in the cell      4025355
  Invoking rule 25355:26062:68=55368
  replace dpston2 = 4055368 if inlist(dpston2, 4025355, 4026062, 4000068)
Pass 42 through the data...
  smallest count = 9 in the cell      4031826
  Invoking rule 30:31826=41830
  replace dpston2 = 4041830 if inlist(dpston2, 4000030, 4031826)
Pass 43 through the data...
  smallest count = 10 in the cell      1000055
  Invoking rule 55:60=25560
  replace dpston2 = 1025560 if inlist(dpston2, 1000055, 1000060)
Pass 44 through the data...
  smallest count = 10 in the cell      5020102
  Invoking rule 8:20102=30108
  replace dpston2 = 5030108 if inlist(dpston2, 5000008, 5020102)
Pass 45 through the data...
  smallest count = 10 in the cell      5055368
  WARNING: could not find any rules to collapse dpston2 == 5055368
Pass 46 through the data...
  smallest count = 11 in the cell      2100844
  Invoking rule 47:100844=110847
  replace dpston2 = 2110847 if inlist(dpston2, 2000047, 2100844)
Pass 47 through the data...
  smallest count = 11 in the cell      3000001
  Invoking rule 1:2=20102
  replace dpston2 = 3020102 if inlist(dpston2, 3000001, 3000002)
Pass 48 through the data...
  smallest count = 12 in the cell      3033944
  Invoking rule 26:23036:33944=62644
  replace dpston2 = 3062644 if inlist(dpston2, 3000026, 3023036, 3033944)
Pass 49 through the data...
  smallest count = 12 in the cell      4000062
  Invoking rule 62:132460=142462
  replace dpston2 = 4142462 if inlist(dpston2, 4000062, 4132460)
Pass 50 through the data...
  smallest count = 12 in the cell      5000030
  Invoking rule 30:36=23036
  replace dpston2 = 5023036 if inlist(dpston2, 5000030, 5000036)
Pass 51 through the data...
  smallest count = 13 in the cell      1024950
  Invoking rule 53:24950=34953
  replace dpston2 = 1034953 if inlist(dpston2, 1000053, 1024950)
Pass 52 through the data...
  smallest count = 13 in the cell      2000026
  Invoking rule 50118:24:26=70126
  replace dpston2 = 2070126 if inlist(dpston2, 2050118, 2000024, 2000026)
Pass 53 through the data...
  smallest count = 13 in the cell      4020102
  Invoking rule 8:20102=30108
  replace dpston2 = 4030108 if inlist(dpston2, 4000008, 4020102)
Pass 54 through the data...
  smallest count = 14 in the cell      3000055
  Invoking rule 53:55=25355
  replace dpston2 = 3025355 if inlist(dpston2, 3000053, 3000055)
Pass 55 through the data...
  smallest count = 14 in the cell      5064760
  Invoking rule 81844:64760=141860
  replace dpston2 = 5141860 if inlist(dpston2, 5081844, 5064760)
Pass 56 through the data...
  smallest count = 17 in the cell      5000062
  Invoking rule 62:141860=151862
  replace dpston2 = 5151862 if inlist(dpston2, 5000062, 5141860)
Pass 57 through the data...
  smallest count = 18 in the cell      3054960
  Invoking rule 62:54960=64962
  replace dpston2 = 3064962 if inlist(dpston2, 3000062, 3054960)
Pass 58 through the data...
  smallest count = 18 in the cell      4055368
  WARNING: could not find any rules to collapse dpston2 == 4055368
Pass 59 through the data...
  smallest count = 20 in the cell      2000030
  Done collapsing! Exiting...
. return list
scalars:
         r(arg_min_id) =  2000030
                r(min) =  20
macros:
            r(cfailed) : "5055368,4055368"
             r(failed) : "5055368 4055368"
. wgtcellcollapse collapse, variables(daypart alight_id) mincellsize(20) ///
>         zeroes(2 8 36 39 40 44 47 49 50 55 60 62) ///
>         generate(dpstoff2) saving(dpstoff2.do) replace run
Pass 0 through the data...
  smallest count = 1 in the cell      1000002
Processing zero cells...
  Invoking rule 39:40=23940 to collapse zero cells
  replace dpstoff2 = 1023940 if inlist(dpstoff2, 1000039, 1000040)
Pass 0 through the data...
  smallest count = 1 in the cell      1000002
  Invoking rule 2:8=20208 to collapse zero cells
  replace dpstoff2 = 2020208 if inlist(dpstoff2, 2000002, 2000008)
Pass 0 through the data...
  smallest count = 1 in the cell      1000002
  Invoking rule 30:36=23036 to collapse zero cells
  replace dpstoff2 = 2023036 if inlist(dpstoff2, 2000030, 2000036)
Pass 0 through the data...
  smallest count = 1 in the cell      1000002
  Invoking rule 26:30:36:39=42639 to collapse zero cells
  replace dpstoff2 = 2042639 if inlist(dpstoff2, 2000026, 2000030, 2000036, 2000039)
Pass 0 through the data...
  smallest count = 1 in the cell      1000002
  Invoking rule 24:26:30:36:39:40=62440 to collapse zero cells
```

```
    replace dpstoff2 = 2062440 if inlist(dpstoff2, 2000024, 2000026, 2000030, 2000036, 2000039, 2000040)
  Pass 0 through the data...
    smallest count = 1 in the cell      1000002
    Invoking rule 44:62440=72444 to collapse zero cells
    replace dpstoff2 = 2072444 if inlist(dpstoff2, 2000044, 2062440)
  Pass 0 through the data...
    smallest count = 1 in the cell      1000002
    Invoking rule 49:50=24950 to collapse zero cells
    replace dpstoff2 = 2024950 if inlist(dpstoff2, 2000049, 2000050)
  Pass 0 through the data...
    smallest count = 1 in the cell      1000002
    Invoking rule 53:55=25355 to collapse zero cells
    replace dpstoff2 = 2025355 if inlist(dpstoff2, 2000053, 2000055)
  Pass 0 through the data...
    smallest count = 1 in the cell      1000002
    Invoking rule 60:25355=35360 to collapse zero cells
    replace dpstoff2 = 2035360 if inlist(dpstoff2, 2000060, 2025355)
  Pass 0 through the data...
    smallest count = 1 in the cell      1000002
    Invoking rule 62:35360=45362 to collapse zero cells
    replace dpstoff2 = 2045362 if inlist(dpstoff2, 2000062, 2035360)
  Pass 0 through the data...
    smallest count = 1 in the cell      1000002
    Invoking rule 2:8=20208 to collapse zero cells
    replace dpstoff2 = 3020208 if inlist(dpstoff2, 3000002, 3000008)
  Pass 0 through the data...
    smallest count = 1 in the cell      1000002
    Invoking rule 55:60=25560 to collapse zero cells
    replace dpstoff2 = 3025560 if inlist(dpstoff2, 3000055, 3000060)
  Pass 0 through the data...
    smallest count = 1 in the cell      1000002
    Invoking rule 2:8:11=30211 to collapse zero cells
    replace dpstoff2 = 4030211 if inlist(dpstoff2, 4000002, 4000008, 4000011)
  Pass 0 through the data...
    smallest count = 1 in the cell      1000002
    Invoking rule 36:39=23639 to collapse zero cells
    replace dpstoff2 = 4023639 if inlist(dpstoff2, 4000036, 4000039)
  Pass 0 through the data...
    smallest count = 1 in the cell      1000002
    Invoking rule 40:23639=33640 to collapse zero cells
    replace dpstoff2 = 4033640 if inlist(dpstoff2, 4000040, 4023639)
  Pass 0 through the data...
    smallest count = 1 in the cell      1000002
    Invoking rule 44:33640=43644 to collapse zero cells
    replace dpstoff2 = 4043644 if inlist(dpstoff2, 4000044, 4033640)
  Pass 0 through the data...
    smallest count = 1 in the cell      1000002
    Invoking rule 49:50=24950 to collapse zero cells
    replace dpstoff2 = 4024950 if inlist(dpstoff2, 4000049, 4000050)
  Pass 0 through the data...
    smallest count = 1 in the cell      1000002
    Invoking rule 55:60=25560 to collapse zero cells
    replace dpstoff2 = 4025560 if inlist(dpstoff2, 4000055, 4000060)
  Pass 0 through the data...
    smallest count = 1 in the cell      1000002
    Invoking rule 2:8=20208 to collapse zero cells
    replace dpstoff2 = 5020208 if inlist(dpstoff2, 5000002, 5000008)
  Pass 0 through the data...
    smallest count = 1 in the cell      1000002
    Invoking rule 36:39=23639 to collapse zero cells
    replace dpstoff2 = 5023639 if inlist(dpstoff2, 5000036, 5000039)
  Pass 0 through the data...
    smallest count = 1 in the cell      1000002
    Invoking rule 40:44=24044 to collapse zero cells
    replace dpstoff2 = 5024044 if inlist(dpstoff2, 5000040, 5000044)
  Pass 0 through the data...
    smallest count = 1 in the cell      1000002
    Invoking rule 49:50=24950 to collapse zero cells
    replace dpstoff2 = 5024950 if inlist(dpstoff2, 5000049, 5000050)
  Pass 0 through the data...
    smallest count = 1 in the cell      1000002
    Invoking rule 53:55=25355 to collapse zero cells
    replace dpstoff2 = 5025355 if inlist(dpstoff2, 5000053, 5000055)
  Pass 0 through the data...
    smallest count = 1 in the cell      1000002
    Invoking rule 24950:53:55:60=54960 to collapse zero cells
    replace dpstoff2 = 5054960 if inlist(dpstoff2, 5024950, 5000053, 5000055, 5000060)
  Pass 0 through the data...
    smallest count = 1 in the cell      1000002
  Pass 24 through the data...
    smallest count = 1 in the cell      1000002
    Invoking rule 2:8=20208
    replace dpstoff2 = 1020208 if inlist(dpstoff2, 1000002, 1000008)
  Pass 25 through the data...
    smallest count = 1 in the cell      2000011
    Invoking rule 11:18=21118
    replace dpstoff2 = 2021118 if inlist(dpstoff2, 2000011, 2000018)
  Pass 26 through the data...
    smallest count = 1 in the cell      2020208
    Invoking rule 20208:21118=40218
    replace dpstoff2 = 2040218 if inlist(dpstoff2, 2020208, 2021118)
  Pass 27 through the data...
    smallest count = 1 in the cell      2024950
    Invoking rule 24950:45362=64962
    replace dpstoff2 = 2064962 if inlist(dpstoff2, 2024950, 2045362)
  Pass 28 through the data...
    smallest count = 1 in the cell      2072444
    Invoking rule 40218:72444=110244
    replace dpstoff2 = 2110244 if inlist(dpstoff2, 2040218, 2072444)
  Pass 29 through the data...
    smallest count = 1 in the cell      3000039
    Invoking rule 39:40=23940
    replace dpstoff2 = 3023940 if inlist(dpstoff2, 3000039, 3000040)
  Pass 30 through the data...
    smallest count = 1 in the cell      3000049
    Invoking rule 49:50=24950
    replace dpstoff2 = 3024950 if inlist(dpstoff2, 3000049, 3000050)
  Pass 31 through the data...
    smallest count = 1 in the cell      3020208
    Invoking rule 20208:21118:24=50224
    replace dpstoff2 = 3050224 if inlist(dpstoff2, 3020208, 3021118, 3000024)
  Pass 32 through the data...
    smallest count = 1 in the cell      4000018
    Invoking rule 18:24=21824
    replace dpstoff2 = 4021824 if inlist(dpstoff2, 4000018, 4000024)
  Pass 33 through the data...
    smallest count = 1 in the cell      4000026
    Invoking rule 26:21824=31826
    replace dpstoff2 = 4031826 if inlist(dpstoff2, 4000026, 4021824)
  Pass 34 through the data...
```

```
    smallest count = 1 in the cell      4025560
    Invoking rule 53:25560=35360
    replace dpstoff2 = 4035360 if inlist(dpstoff2, 4000053, 4025560)
  Pass 35 through the data...
    smallest count = 1 in the cell      5000026
    Invoking rule 24:26=22426
    replace dpstoff2 = 5022426 if inlist(dpstoff2, 5000024, 5000026)
  Pass 36 through the data...
    smallest count = 1 in the cell      5020208
    Invoking rule 11:20208=30211
    replace dpstoff2 = 5030211 if inlist(dpstoff2, 5000011, 5020208)
  Pass 37 through the data...
    smallest count = 1 in the cell      5054960
    Invoking rule 47:54960=64760
    replace dpstoff2 = 5064760 if inlist(dpstoff2, 5000047, 5054960)
  Pass 38 through the data...
    smallest count = 2 in the cell      1000050
    Invoking rule 49:50=24950
    replace dpstoff2 = 1024950 if inlist(dpstoff2, 1000049, 1000050)
  Pass 39 through the data...
    smallest count = 2 in the cell      2042639
    Invoking rule 42639:40:44:47=72647
    replace dpstoff2 = 2072647 if inlist(dpstoff2, 2042639, 2000040, 2000044, 2000047)
  Pass 40 through the data...
    smallest count = 2 in the cell      2064962
    Invoking rule 68:64962=74968
    replace dpstoff2 = 2074968 if inlist(dpstoff2, 2000068, 2064962)
  Pass 41 through the data...
    smallest count = 2 in the cell      3000044
    Invoking rule 44:23940=33944
    replace dpstoff2 = 3033944 if inlist(dpstoff2, 3000044, 3023940)
  Pass 42 through the data...
    smallest count = 2 in the cell      3024950
    Invoking rule 53:24950=34953
    replace dpstoff2 = 3034953 if inlist(dpstoff2, 3000053, 3024950)
  Pass 43 through the data...
    smallest count = 2 in the cell      4024950
    Invoking rule 24950:35360=54960
    replace dpstoff2 = 4054960 if inlist(dpstoff2, 4024950, 4035360)
  Pass 44 through the data...
    smallest count = 2 in the cell      4043644
    Invoking rule 47:43644=53647
    replace dpstoff2 = 4053647 if inlist(dpstoff2, 4000047, 4043644)
  Pass 45 through the data...
    smallest count = 2 in the cell      5023639
    Invoking rule 23639:24044=43644
    replace dpstoff2 = 5043644 if inlist(dpstoff2, 5023639, 5024044)
  Pass 46 through the data...
    smallest count = 2 in the cell      5025355
    Invoking rule 25355:26062:68=55368
    replace dpstoff2 = 5055368 if inlist(dpstoff2, 5025355, 5026062, 5000068)
  Pass 47 through the data...
    smallest count = 3 in the cell      1023940
    Invoking rule 36:23940=33640
    replace dpstoff2 = 1033640 if inlist(dpstoff2, 1000036, 1023940)
  Pass 48 through the data...
    smallest count = 3 in the cell      3050224
    Invoking rule 50224:22630:36=80236
    replace dpstoff2 = 3080236 if inlist(dpstoff2, 3050224, 3022630, 3000036)
  Pass 49 through the data...
    smallest count = 3 in the cell      4000062
    Invoking rule 62:68=26268
    replace dpstoff2 = 4026268 if inlist(dpstoff2, 4000062, 4000068)
  Pass 50 through the data...
    smallest count = 3 in the cell      5022426
    Invoking rule 18:22426=31826
    replace dpstoff2 = 5031826 if inlist(dpstoff2, 5000018, 5022426)
  Pass 51 through the data...
    smallest count = 4 in the cell      2023036
    WARNING: could not find any rules to collapse dpstoff2 == 2023036
  Pass 52 through the data...
    smallest count = 4 in the cell      2110244
    WARNING: could not find any rules to collapse dpstoff2 == 2110244
  Pass 53 through the data...
    smallest count = 4 in the cell      4031826
    Invoking rule 30211:31826=60226
    replace dpstoff2 = 4060226 if inlist(dpstoff2, 4030211, 4031826)
  Pass 54 through the data...
    smallest count = 4 in the cell      5043644
    Invoking rule 43644:64760=103660
    replace dpstoff2 = 5103660 if inlist(dpstoff2, 5043644, 5064760)
  Pass 55 through the data...
    smallest count = 5 in the cell      1000060
    Invoking rule 24950:25355:60=54960
    replace dpstoff2 = 1054960 if inlist(dpstoff2, 1024950, 1025355, 1000060)
  Pass 56 through the data...
    smallest count = 5 in the cell      2074968
    Invoking rule 72647:74968=142668
    replace dpstoff2 = 2142668 if inlist(dpstoff2, 2072647, 2074968)
  Pass 57 through the data...
    smallest count = 5 in the cell      3025560
    Invoking rule 34953:25560=54960
    replace dpstoff2 = 3054960 if inlist(dpstoff2, 3034953, 3025560)
  Pass 58 through the data...
    smallest count = 6 in the cell      1000055
    Invoking rule 53:55=25355
    replace dpstoff2 = 1025355 if inlist(dpstoff2, 1000053, 1000055)
  Pass 59 through the data...
    smallest count = 6 in the cell      3033944
    Invoking rule 80236:33944=110244
    replace dpstoff2 = 3110244 if inlist(dpstoff2, 3080236, 3033944)
  Pass 60 through the data...
    smallest count = 6 in the cell      4054960
    Invoking rule 53647:54960=103660
    replace dpstoff2 = 4103660 if inlist(dpstoff2, 4053647, 4054960)
  Pass 61 through the data...
    smallest count = 6 in the cell      5030211
    Invoking rule 30211:31826=60226
    replace dpstoff2 = 5060226 if inlist(dpstoff2, 5030211, 5031826)
  Pass 62 through the data...
    smallest count = 8 in the cell      3000026
    Invoking rule 18:24:26=31826
    replace dpstoff2 = 3031826 if inlist(dpstoff2, 3000018, 3000024, 3000026)
  Pass 63 through the data...
    smallest count = 8 in the cell      5000030
    Invoking rule 30:36:39:40:44:47:49:50:53:55:60:62=123062
    replace dpstoff2 = 5123062 if inlist(dpstoff2, 5000030, 5000036, 5000039, 5000040, 5000044, 5000047, 5000
  > 049, 5000050, 5000053, 5000055, 5000060, 5000062)
  Pass 64 through the data...
    smallest count = 9 in the cell      4026268
    Invoking rule 103660:26268=123668
    replace dpstoff2 = 4123668 if inlist(dpstoff2, 4103660, 4026268)
  Pass 65 through the data...
    smallest count = 9 in the cell      5055368
    Invoking rule 69:55368=65369
    replace dpstoff2 = 5065369 if inlist(dpstoff2, 5000069, 5055368)
  Pass 66 through the data...
    smallest count = 10 in the cell      1054960
    Invoking rule 62:54960=64962
    replace dpstoff2 = 1064962 if inlist(dpstoff2, 1000062, 1054960)
  Pass 67 through the data...
    smallest count = 10 in the cell      4060226
    Invoking rule 30:60226=70230
    replace dpstoff2 = 4070230 if inlist(dpstoff2, 4000030, 4060226)
  Pass 68 through the data...
    smallest count = 11 in the cell      5103660
    WARNING: could not find any rules to collapse dpstoff2 == 5103660
  Pass 69 through the data...
    smallest count = 12 in the cell      1020208
    Invoking rule 20208:21118:24=50224
    replace dpstoff2 = 1050224 if inlist(dpstoff2, 1020208, 1021118, 1000024)
  Pass 70 through the data...
    smallest count = 12 in the cell      1033640
    Invoking rule 44:33640=43644
    replace dpstoff2 = 1043644 if inlist(dpstoff2, 1000044, 1033640)
  Pass 71 through the data...
    smallest count = 13 in the cell      2142668
    Invoking rule 69:142668=152669
    replace dpstoff2 = 2152669 if inlist(dpstoff2, 2000069, 2142668)
  Pass 72 through the data...
    smallest count = 13 in the cell      3110244
    Invoking rule 47:110244=120247
    replace dpstoff2 = 3120247 if inlist(dpstoff2, 3000047, 3110244)
  Pass 73 through the data...
    smallest count = 13 in the cell      5060226
    Invoking rule 60226:123062=180262
    replace dpstoff2 = 5180262 if inlist(dpstoff2, 5060226, 5123062)
  Pass 74 through the data...
    smallest count = 14 in the cell      3000011
    Invoking rule 11:18:24:26:30=51130
    replace dpstoff2 = 3051130 if inlist(dpstoff2, 3000011, 3000018, 3000024, 3000026, 3000030)
  Pass 75 through the data...
    smallest count = 15 in the cell      1000026
    Invoking rule 26:50224=60226
    replace dpstoff2 = 1060226 if inlist(dpstoff2, 1000026, 1050224)
  Pass 76 through the data...
    smallest count = 15 in the cell      3054960
    Invoking rule 62:54960=64962
    replace dpstoff2 = 3064962 if inlist(dpstoff2, 3000062, 3054960)
  Pass 77 through the data...
    smallest count = 21 in the cell      4070230
    Done collapsing! Exiting...
  . return list
  scalars:
           r(arg_min_id) =  4070230
                  r(min) =  21
  macros:
              r(cfailed) : "2023036,2110244,5103660"
               r(failed) : "2023036 2110244 5103660"
```

We will continue to disregard the cell counts of insufficient size for the time being. Running the resulting do-files `dpston.do` and `dpstoff.do` on the population data to create control totals, and providing these control totals to `ipfraking` program produces an apparently successful raking result:

```
. use trip_sample, clear
. run dpston2
. run dpstoff2
. gen byte _one = 1
. ipfraking [pw=_one], ctotal(dpston2 dpstoff2) gen(raked_weight2)
 Iteration 1, max rel difference of raked weights = 36.208881
 Iteration 2, max rel difference of raked weights = .05484732
 Iteration 3, max rel difference of raked weights = .0055794
 Iteration 4, max rel difference of raked weights = .00053851
 Iteration 5, max rel difference of raked weights = .00005171
 Iteration 6, max rel difference of raked weights = 4.962e-06
 Iteration 7, max rel difference of raked weights = 4.762e-07
The worst relative discrepancy of  3.9e-08 is observed for dpstoff2 == 5180262
Target value =       483; achieved value =       483
   Summary of the weight changes
              |    Mean    Std. dev.    Min        Max       CV
--------------+------------------------------------------------
Orig weights  |       1          0        1          1        0
Raked weights |  26.487     5.9013   8.1096     37.001    .2228
Adjust factor | 26.4869              8.1096    37.0014
. whatsdeff raked_weight2
    Group     |     Min     |     Mean    |     Max     |     CV    |   DEFF    |    N    |  N eff
--------------+-------------+-------------+-------------+-----------+-----------+---------+--------
      Overall |        8.11 |       26.49 |       37.00 |    0.2228 |    1.0496 |    3654 | 3481.24
```

Note the use of utility program `whatsdeff` to compute the design effect due to unequal weighting; see section **??**. The problem of zero cells appeared to have been solved: each and every population combination of daypart and station is properly reflected in control total categories, and there are

The weighting cells, however, are still not without problems. Consider this cross-tab of original and collapsed stations (the first part of the *if* expression identifies the daypart, AM Peak; the second part identifies collapsed stations, given the nomenclature of `dpstoff` variable described on page **??** as the concatenation of the first variable of the interaction, `daypart`; the length of the collapsed sequence, and its starting and end points).

```
. tab alight_id dpstoff2 if daypart == 1 & mod(dpstoff2,100*100)>100
                     |          Long ID of the interaction
           alight_id |   1025355    1043644    1060226    1064962 |     Total
---------------------+--------------------------------------------+----------
        2. Brookline |         0          0          1          0 |         1
        8. Carmenton |         0          0         11          0 |        11
      24. Framington |         0          0         15          0 |        15
  26. Grand Junction |         0          0         15          0 |        15
      36. Irvingtown |         0          9          0          0 |         9
      39. Johnsville |         0          3          0          0 |         3
```

```
          44. Limerick |         0         13          0          0 |        13 
      49. Ninth Street |         0          0          0          3 |         3 
      50. Ontario Lake |         0          0          0          2 |         2 
  53. Picadilly Square |        23          0          0          0 |        23 
        55. Queens Zoo |         6          0          0          0 |         6 
    60. Redline Circle |         0          0          0          5 |         5 
     62. Silver Spring |         0          0          0         49 |        49 
-----------------------+--------------------------------------------+----------
                 Total |        29         25         42         59 |       155 
```

To the human eye, it is obvious that Picadilly Square (53) and Queens Zoo (55) should have been a part of the six-station sequence 1064962 spanning from Ninth Street (49) to Silver Spring (62). Instead, `wgtcellcollapse` decided to separate these two stations out into their own cell. How did that happen? The logic of `wgtcellcollapse` is to collapse categories in such a way as to produce the result with the smallest possible count. Thus, within AM Peak daypart, the sequence of collapsing steps was as follows.

The zero cells were collapsed first: Johnsville (39) and King Street (40) resulting in an intermediate cell of size 3.

The smallest cell of size 1 (Brookline (2)) was collapsed with its neighbor (Carmenton (8)) resulting in an intermediate cell of size 12.

The smallest cell of size 2 (Ontario Lake (50)) was collapsed with its neighbor (Ninth Street (49)) resulting in an intermediate cell of size 5.

The smallest cell of size 3, collapsed Johnsville (39) and King Street (40), was further collapsed with its neighbor Irvingtown (36) resulting in an intermediate cell of size .

The smallest cell of size 5, Redline Circle (60), was collapsed by a three-way rule with a duo Picadilly Square (53) + Queens Zoo (55), which actually was empty, and a small cell Ontario Lake (50) + Ninth Street (49), resulting in an intermediate cell of size 10.

Let us look at that last step in more detail. At this stage, Redline Circle (60) with 5 exiting passengers in the sample could be collapsed with:

1. Silver Spring (62), to form a cell of size 54;
2. Queens Zoo (55), to form a cell of size 11;
3. a sequence of Picadilly Square (53) and Queens Zoo (55), to form a cell of size 34;
4. . . . and many other options

However, at pass 55, `wgtcellcollapse` picked the rule 24950:25355:60=54960 which, at the time it was processed, had a count of 5 in the cell 24950, a count of zero in the cell 25355, and a count of 5 in the original station Redline Circle (60). (Note that the cell 25355 would actually form eventually at pass 58.) The problem lies with the zero count of the ghost of the cell 25355.

To overcome this problem, `wgtcellcollapse` have a `strict` option that only allows the rules that have a non-zero count in every component of the rule (so 24950:25355:60=54960 would not be a legal merge under that option). As is easily seen, this option directly contradicts the `zeroes()` option, and that necessitates separate runs.

The third pass of weight collapse and raking: `strict` and `feed` options

We will separate the two runs of `wgtcellcollapse` into a run that only deals with zeroes, and another run that deals with everything else. To prevent `wgtcellcollapse` from any further merges, `mincellsize(1)` can be specified in the first run. As the relevant variables will have already been created by the first run, the option to pass the variable name to be further modified is `feed()`. To make sure that the relevant variable exists in the data set, the option `run` instructs `wgtcellcollapse` to run the do-file it just created, thus creating or modifying the collapsed cell variable. Finally, instead of specifying `replace` to overwrite the do-files that `wgtcellcollapse` creates, we need to specify `append` to keep adding to these files.

```
. use trip_sample_rules, clear
. wgtcellcollapse collapse, variables(daypart board_id) mincellsize(1) ///
>         zeroes(39 40 44 49 55 60) ///
>         generate(dpston3) saving(dpston3.do) replace run
Pass 0 through the data...
  smallest count = 1 in the cell      2000039
Processing zero cells...
  Invoking rule 49:50=24950 to collapse zero cells
  replace dpston3 = 1024950 if inlist(dpston3, 1000049, 1000050)
Pass 0 through the data...
  smallest count = 1 in the cell      2000039
  Invoking rule 40:44=24044 to collapse zero cells
  replace dpston3 = 2024044 if inlist(dpston3, 2000040, 2000044)
Pass 0 through the data...
  smallest count = 1 in the cell      2000039
  Invoking rule 49:50=24950 to collapse zero cells
  replace dpston3 = 2024950 if inlist(dpston3, 2000049, 2000050)
Pass 0 through the data...
  smallest count = 1 in the cell      2000039
  Invoking rule 55:60=25560 to collapse zero cells
  replace dpston3 = 2025560 if inlist(dpston3, 2000055, 2000060)
Pass 0 through the data...
  smallest count = 1 in the cell      2000039
  Invoking rule 36:39=23639 to collapse zero cells
  replace dpston3 = 4023639 if inlist(dpston3, 4000036, 4000039)
Pass 0 through the data...
  smallest count = 1 in the cell      2000039
  Invoking rule 40:44=24044 to collapse zero cells
  replace dpston3 = 4024044 if inlist(dpston3, 4000040, 4000044)
Pass 0 through the data...
  smallest count = 1 in the cell      2000039
  Invoking rule 49:50=24950 to collapse zero cells
  replace dpston3 = 4024950 if inlist(dpston3, 4000049, 4000050)
Pass 0 through the data...
  smallest count = 1 in the cell      2000039
  Invoking rule 53:55=25355 to collapse zero cells
  replace dpston3 = 4025355 if inlist(dpston3, 4000053, 4000055)
Pass 0 through the data...
  smallest count = 1 in the cell      2000039
  Invoking rule 24950:53:55:60=54960 to collapse zero cells
  replace dpston3 = 4054960 if inlist(dpston3, 4024950, 4000053, 4000055, 4000060)
Pass 0 through the data...
  smallest count = 1 in the cell      2000039
  Invoking rule 39:40=23940 to collapse zero cells
  replace dpston3 = 5023940 if inlist(dpston3, 5000039, 5000040)
Pass 0 through the data...
  smallest count = 1 in the cell      2000039
  Invoking rule 49:50=24950 to collapse zero cells
  replace dpston3 = 5024950 if inlist(dpston3, 5000049, 5000050)
Pass 0 through the data...
  smallest count = 1 in the cell      2000039
  Invoking rule 24950:25355:60=54960 to collapse zero cells
  replace dpston3 = 5054960 if inlist(dpston3, 5024950, 5025355, 5000060)
Pass 0 through the data...
  smallest count = 1 in the cell      2000039
Pass 12 through the data...
  smallest count = 1 in the cell      2000039
  Done collapsing! Exiting...
. wgtcellcollapse collapse, variables(daypart board_id) mincellsize(20) ///
>         strict feed(dpston3) saving(dpston3.do) append run
Pass 12 through the data...
  smallest count = 1 in the cell      2000039
  Invoking rule 39:24044=33944
  replace dpston3 = 2033944 if inlist(dpston3, 2000039, 2024044)
Pass 13 through the data...
  smallest count = 1 in the cell      2024950
  Invoking rule 53:24950=34953
  replace dpston3 = 2034953 if inlist(dpston3, 2000053, 2024950)
Pass 14 through the data...
  smallest count = 1 in the cell      2025560
  Invoking rule 34953:25560=54960
  replace dpston3 = 2054960 if inlist(dpston3, 2034953, 2025560)
Pass 15 through the data...
  smallest count = 1 in the cell      3000039
  Invoking rule 39:40=23940
  replace dpston3 = 3023940 if inlist(dpston3, 3000039, 3000040)
Pass 16 through the data...
  smallest count = 1 in the cell      4023639
  Invoking rule 23639:24044=43644
  replace dpston3 = 4043644 if inlist(dpston3, 4023639, 4024044)
Pass 17 through the data...
  smallest count = 1 in the cell      4054960
  Invoking rule 47:54960=64760
  replace dpston3 = 4064760 if inlist(dpston3, 4000047, 4054960)
Pass 18 through the data...
  smallest count = 1 in the cell      5000024
  Invoking rule 18:24=21824
  replace dpston3 = 5021824 if inlist(dpston3, 5000018, 5000024)
Pass 19 through the data...
  smallest count = 1 in the cell      5023940
  Invoking rule 44:23940=33944
  replace dpston3 = 5033944 if inlist(dpston3, 5000044, 5023940)
Pass 20 through the data...
  smallest count = 1 in the cell      5054960
  Invoking rule 47:54960=64760
  replace dpston3 = 5064760 if inlist(dpston3, 5000047, 5054960)
```

```
Pass 21 through the data...
  smallest count = 2 in the cell      2000036
  Invoking rule 36:33944=43644
  replace dpston3 = 2043644 if inlist(dpston3, 2000036, 2033944)
Pass 22 through the data...
  smallest count = 2 in the cell      4043644
  Invoking rule 43644:64760=103660
  replace dpston3 = 4103660 if inlist(dpston3, 4043644, 4064760)
Pass 23 through the data...
  smallest count = 2 in the cell      5000055
  Invoking rule 53:55=25355
  replace dpston3 = 5025355 if inlist(dpston3, 5000053, 5000055)
Pass 24 through the data...
  smallest count = 3 in the cell      2000024
  Invoking rule 18:24=21824
  replace dpston3 = 2021824 if inlist(dpston3, 2000018, 2000024)
Pass 25 through the data...
  smallest count = 3 in the cell      3023940
  Invoking rule 44:23940=33944
  replace dpston3 = 3033944 if inlist(dpston3, 3000044, 3023940)
Pass 26 through the data...
  smallest count = 3 in the cell      4000024
  Invoking rule 18:24=21824
  replace dpston3 = 4021824 if inlist(dpston3, 4000018, 4000024)
Pass 27 through the data...
  smallest count = 3 in the cell      5000001
  Invoking rule 1:2=20102
  replace dpston3 = 5020102 if inlist(dpston3, 5000001, 5000002)
Pass 28 through the data...
  smallest count = 3 in the cell      5000068
  Invoking rule 62:68=26268
  replace dpston3 = 5026268 if inlist(dpston3, 5000062, 5000068)
Pass 29 through the data...
  smallest count = 4 in the cell      2000001
  Invoking rule 1:2=20102
  replace dpston3 = 2020102 if inlist(dpston3, 2000001, 2000002)
Pass 30 through the data...
  smallest count = 4 in the cell      2000008
  Invoking rule 8:20102=30108
  replace dpston3 = 2030108 if inlist(dpston3, 2000008, 2020102)
Pass 31 through the data...
  smallest count = 4 in the cell      2043644
  Invoking rule 30:43644=53044
  replace dpston3 = 2053044 if inlist(dpston3, 2000030, 2043644)
Pass 32 through the data...
  smallest count = 4 in the cell      3000050
  Invoking rule 49:50=24950
  replace dpston3 = 3024950 if inlist(dpston3, 3000049, 3000050)
Pass 33 through the data...
  smallest count = 4 in the cell      5033944
  Invoking rule 33944:64760=93960
  replace dpston3 = 5093960 if inlist(dpston3, 5033944, 5064760)
Pass 34 through the data...
  smallest count = 5 in the cell      1000039
  Invoking rule 39:40=23940
  replace dpston3 = 1023940 if inlist(dpston3, 1000039, 1000040)
Pass 35 through the data...
  smallest count = 5 in the cell      3000060
  Invoking rule 55:60=25560
  replace dpston3 = 3025560 if inlist(dpston3, 3000055, 3000060)
```

```
Pass 36 through the data...
  smallest count = 5 in the cell      4000026
  Invoking rule 26:21824=31826
  replace dpston3 = 4031826 if inlist(dpston3, 4000026, 4021824)
Pass 37 through the data...
  smallest count = 5 in the cell      5021824
  Invoking rule 26:21824=31826
  replace dpston3 = 5031826 if inlist(dpston3, 5000026, 5021824)
Pass 38 through the data...
  smallest count = 6 in the cell      2000068
  Invoking rule 62:68=26268
  replace dpston3 = 2026268 if inlist(dpston3, 2000062, 2000068)
Pass 39 through the data...
  smallest count = 6 in the cell      2054960
  Invoking rule 54960:26268=74968
  replace dpston3 = 2074968 if inlist(dpston3, 2054960, 2026268)
Pass 40 through the data...
  smallest count = 6 in the cell      4000002
  Invoking rule 1:2=20102
  replace dpston3 = 4020102 if inlist(dpston3, 4000001, 4000002)
Pass 41 through the data...
  smallest count = 7 in the cell      4025355
  WARNING: could not find any rules to collapse dpston3 == 4025355
Pass 42 through the data...
  smallest count = 7 in the cell      5025355
  WARNING: could not find any rules to collapse dpston3 == 5025355
Pass 43 through the data...
  smallest count = 8 in the cell      2021824
  Invoking rule 26:21824=31826
  replace dpston3 = 2031826 if inlist(dpston3, 2000026, 2021824)
Pass 44 through the data...
  smallest count = 10 in the cell      1000055
  Invoking rule 55:60=25560
  replace dpston3 = 1025560 if inlist(dpston3, 1000055, 1000060)
Pass 45 through the data...
  smallest count = 10 in the cell      4103660
  Invoking rule 62:103660=113662
  replace dpston3 = 4113662 if inlist(dpston3, 4000062, 4103660)
Pass 46 through the data...
  smallest count = 10 in the cell      5020102
  Invoking rule 8:20102=30108
  replace dpston3 = 5030108 if inlist(dpston3, 5000008, 5020102)
Pass 47 through the data...
  smallest count = 11 in the cell      3000001
  Invoking rule 1:2=20102
  replace dpston3 = 3020102 if inlist(dpston3, 3000001, 3000002)
Pass 48 through the data...
  smallest count = 11 in the cell      4000068
  Invoking rule 68:113662=123668
  replace dpston3 = 4123668 if inlist(dpston3, 4000068, 4113662)
Pass 49 through the data...
  smallest count = 11 in the cell      5031826
  Invoking rule 30:31826=41830
  replace dpston3 = 5041830 if inlist(dpston3, 5000030, 5031826)
Pass 50 through the data...
  smallest count = 12 in the cell      2030108
  Invoking rule 11:30108=40111
  replace dpston3 = 2040111 if inlist(dpston3, 2000011, 2030108)
Pass 51 through the data...
  smallest count = 12 in the cell      3033944
  Invoking rule 36:33944=43644
  replace dpston3 = 3043644 if inlist(dpston3, 3000036, 3033944)
Pass 52 through the data...
  smallest count = 12 in the cell      4031826
  Invoking rule 30:31826=41830
  replace dpston3 = 4041830 if inlist(dpston3, 4000030, 4031826)
Pass 53 through the data...
  smallest count = 13 in the cell      1024950
  Invoking rule 53:24950=34953
  replace dpston3 = 1034953 if inlist(dpston3, 1000053, 1024950)
Pass 54 through the data...
  smallest count = 13 in the cell      3024950
  Invoking rule 53:24950=34953
  replace dpston3 = 3034953 if inlist(dpston3, 3000053, 3024950)
Pass 55 through the data...
  smallest count = 13 in the cell      4020102
  Invoking rule 8:20102=30108
  replace dpston3 = 4030108 if inlist(dpston3, 4000008, 4020102)
Pass 56 through the data...
  smallest count = 15 in the cell      5000036
  Invoking rule 36:93960=103660
  replace dpston3 = 5103660 if inlist(dpston3, 5000036, 5093960)
Pass 57 through the data...
  smallest count = 19 in the cell      3025560
  Invoking rule 62:25560=35562
  replace dpston3 = 3035562 if inlist(dpston3, 3000062, 3025560)
Pass 58 through the data...
  smallest count = 20 in the cell      5026268
  Done collapsing! Exiting...
. wgtcellcollapse collapse, variables(daypart alight_id) mincellsize(1) ///
>         zeroes(2 8 36 39 40 44 47 49 50 55 60 62) ///
>         generate(dpstoff3) saving(dpstoff3.do) replace run
Pass 0 through the data...
  smallest count = 1 in the cell      1000002
Processing zero cells...
  Invoking rule 39:40=23940 to collapse zero cells
  replace dpstoff3 = 1023940 if inlist(dpstoff3, 1000039, 1000040)
Pass 0 through the data...
  smallest count = 1 in the cell      1000002
  Invoking rule 2:8=20208 to collapse zero cells
  replace dpstoff3 = 2020208 if inlist(dpstoff3, 2000002, 2000008)
Pass 0 through the data...
  smallest count = 1 in the cell      1000002
  Invoking rule 30:36=23036 to collapse zero cells
  replace dpstoff3 = 2023036 if inlist(dpstoff3, 2000030, 2000036)
Pass 0 through the data...
  smallest count = 1 in the cell      1000002
  Invoking rule 26:30:36:39=42639 to collapse zero cells
  replace dpstoff3 = 2042639 if inlist(dpstoff3, 2000026, 2000030, 2000036, 2000039)
Pass 0 through the data...
  smallest count = 1 in the cell      1000002
  Invoking rule 24:26:30:36:39:40=62440 to collapse zero cells
  replace dpstoff3 = 2062440 if inlist(dpstoff3, 2000024, 2000026, 2000030, 2000036, 2000039, 2000040)
Pass 0 through the data...
  smallest count = 1 in the cell      1000002
  Invoking rule 44:62440=72444 to collapse zero cells
  replace dpstoff3 = 2072444 if inlist(dpstoff3, 2000044, 2062440)
Pass 0 through the data...
  smallest count = 1 in the cell      1000002
```

```
    Invoking rule 49:50=24950 to collapse zero cells
    replace dpstoff3 = 2024950 if inlist(dpstoff3, 2000049, 2000050)
  Pass 0 through the data...
    smallest count = 1 in the cell      1000002
    Invoking rule 53:55=25355 to collapse zero cells
    replace dpstoff3 = 2025355 if inlist(dpstoff3, 2000053, 2000055)
  Pass 0 through the data...
    smallest count = 1 in the cell      1000002
    Invoking rule 60:25355=35360 to collapse zero cells
    replace dpstoff3 = 2035360 if inlist(dpstoff3, 2000060, 2025355)
  Pass 0 through the data...
    smallest count = 1 in the cell      1000002
    Invoking rule 62:35360=45362 to collapse zero cells
    replace dpstoff3 = 2045362 if inlist(dpstoff3, 2000062, 2035360)
  Pass 0 through the data...
    smallest count = 1 in the cell      1000002
    Invoking rule 2:8=20208 to collapse zero cells
    replace dpstoff3 = 3020208 if inlist(dpstoff3, 3000002, 3000008)
  Pass 0 through the data...
    smallest count = 1 in the cell      1000002
    Invoking rule 55:60=25560 to collapse zero cells
    replace dpstoff3 = 3025560 if inlist(dpstoff3, 3000055, 3000060)
  Pass 0 through the data...
    smallest count = 1 in the cell      1000002
    Invoking rule 2:8:11=30211 to collapse zero cells
    replace dpstoff3 = 4030211 if inlist(dpstoff3, 4000002, 4000008, 4000011)
  Pass 0 through the data...
    smallest count = 1 in the cell      1000002
    Invoking rule 36:39=23639 to collapse zero cells
    replace dpstoff3 = 4023639 if inlist(dpstoff3, 4000036, 4000039)
  Pass 0 through the data...
    smallest count = 1 in the cell      1000002
    Invoking rule 40:23639=33640 to collapse zero cells
    replace dpstoff3 = 4033640 if inlist(dpstoff3, 4000040, 4023639)
  Pass 0 through the data...
    smallest count = 1 in the cell      1000002
    Invoking rule 44:33640=43644 to collapse zero cells
    replace dpstoff3 = 4043644 if inlist(dpstoff3, 4000044, 4033640)
  Pass 0 through the data...
    smallest count = 1 in the cell      1000002
    Invoking rule 49:50=24950 to collapse zero cells
    replace dpstoff3 = 4024950 if inlist(dpstoff3, 4000049, 4000050)
  Pass 0 through the data...
    smallest count = 1 in the cell      1000002
    Invoking rule 55:60=25560 to collapse zero cells
    replace dpstoff3 = 4025560 if inlist(dpstoff3, 4000055, 4000060)
  Pass 0 through the data...
    smallest count = 1 in the cell      1000002
    Invoking rule 2:8=20208 to collapse zero cells
    replace dpstoff3 = 5020208 if inlist(dpstoff3, 5000002, 5000008)
  Pass 0 through the data...
    smallest count = 1 in the cell      1000002
    Invoking rule 36:39=23639 to collapse zero cells
    replace dpstoff3 = 5023639 if inlist(dpstoff3, 5000036, 5000039)
  Pass 0 through the data...
    smallest count = 1 in the cell      1000002
    Invoking rule 40:44=24044 to collapse zero cells
    replace dpstoff3 = 5024044 if inlist(dpstoff3, 5000040, 5000044)
  Pass 0 through the data...
    smallest count = 1 in the cell      1000002
    Invoking rule 49:50=24950 to collapse zero cells
    replace dpstoff3 = 5024950 if inlist(dpstoff3, 5000049, 5000050)
  Pass 0 through the data...
    smallest count = 1 in the cell      1000002
    Invoking rule 53:55=25355 to collapse zero cells
    replace dpstoff3 = 5025355 if inlist(dpstoff3, 5000053, 5000055)
  Pass 0 through the data...
    smallest count = 1 in the cell      1000002
    Invoking rule 24950:53:55:60=54960 to collapse zero cells
    replace dpstoff3 = 5054960 if inlist(dpstoff3, 5024950, 5000053, 5000055, 5000060)
  Pass 0 through the data...
    smallest count = 1 in the cell      1000002
  Pass 24 through the data...
    smallest count = 1 in the cell      1000002
    Done collapsing! Exiting...
  . wgtcellcollapse collapse, variables(daypart alight_id) mincellsize(20) ///
  >         strict feed(dpstoff3) saving(dpstoff3.do) append run
  Pass 24 through the data...
    smallest count = 1 in the cell      1000002
    Invoking rule 2:8=20208
    replace dpstoff3 = 1020208 if inlist(dpstoff3, 1000002, 1000008)
  Pass 25 through the data...
    smallest count = 1 in the cell      2000011
    Invoking rule 11:18=21118
    replace dpstoff3 = 2021118 if inlist(dpstoff3, 2000011, 2000018)
  Pass 26 through the data...
    smallest count = 1 in the cell      2020208
    Invoking rule 20208:21118=40218
    replace dpstoff3 = 2040218 if inlist(dpstoff3, 2020208, 2021118)
  Pass 27 through the data...
    smallest count = 1 in the cell      2024950
    Invoking rule 24950:45362=64962
    replace dpstoff3 = 2064962 if inlist(dpstoff3, 2024950, 2045362)
  Pass 28 through the data...
    smallest count = 1 in the cell      2072444
    Invoking rule 40218:72444=110244
    replace dpstoff3 = 2110244 if inlist(dpstoff3, 2040218, 2072444)
  Pass 29 through the data...
    smallest count = 1 in the cell      3000039
    Invoking rule 39:40=23940
    replace dpstoff3 = 3023940 if inlist(dpstoff3, 3000039, 3000040)
  Pass 30 through the data...
    smallest count = 1 in the cell      3000049
    Invoking rule 49:50=24950
    replace dpstoff3 = 3024950 if inlist(dpstoff3, 3000049, 3000050)
  Pass 31 through the data...
    smallest count = 1 in the cell      3020208
    Invoking rule 11:20208=30211
    replace dpstoff3 = 3030211 if inlist(dpstoff3, 3000011, 3020208)
  Pass 32 through the data...
    smallest count = 1 in the cell      4000018
    Invoking rule 18:24=21824
    replace dpstoff3 = 4021824 if inlist(dpstoff3, 4000018, 4000024)
  Pass 33 through the data...
    smallest count = 1 in the cell      4000026
    Invoking rule 26:21824=31826
    replace dpstoff3 = 4031826 if inlist(dpstoff3, 4000026, 4021824)
  Pass 34 through the data...
    smallest count = 1 in the cell      4025560
    Invoking rule 53:25560=35360
    replace dpstoff3 = 4035360 if inlist(dpstoff3, 4000053, 4025560)
  Pass 35 through the data...
    smallest count = 1 in the cell      5000026
    Invoking rule 24:26=22426
    replace dpstoff3 = 5022426 if inlist(dpstoff3, 5000024, 5000026)
  Pass 36 through the data...
    smallest count = 1 in the cell      5020208
    Invoking rule 11:20208=30211
    replace dpstoff3 = 5030211 if inlist(dpstoff3, 5000011, 5020208)
  Pass 37 through the data...
    smallest count = 1 in the cell      5054960
    Invoking rule 47:54960=64760
    replace dpstoff3 = 5064760 if inlist(dpstoff3, 5000047, 5054960)
  Pass 38 through the data...
    smallest count = 2 in the cell      1000050
    Invoking rule 49:50=24950
    replace dpstoff3 = 1024950 if inlist(dpstoff3, 1000049, 1000050)
  Pass 39 through the data...
    smallest count = 2 in the cell      2042639
    WARNING: could not find any rules to collapse dpstoff3 == 2042639
  Pass 40 through the data...
    smallest count = 2 in the cell      2064962
    Invoking rule 68:64962=74968
    replace dpstoff3 = 2074968 if inlist(dpstoff3, 2000068, 2064962)
  Pass 41 through the data...
    smallest count = 2 in the cell      3000024
    Invoking rule 24:26=22426
    replace dpstoff3 = 3022426 if inlist(dpstoff3, 3000024, 3000026)
  Pass 42 through the data...
    smallest count = 2 in the cell      3000044
    Invoking rule 44:23940=33944
    replace dpstoff3 = 3033944 if inlist(dpstoff3, 3000044, 3023940)
  Pass 43 through the data...
    smallest count = 2 in the cell      3024950
    Invoking rule 53:24950=34953
    replace dpstoff3 = 3034953 if inlist(dpstoff3, 3000053, 3024950)
  Pass 44 through the data...
    smallest count = 2 in the cell      4024950
    Invoking rule 24950:35360=54960
    replace dpstoff3 = 4054960 if inlist(dpstoff3, 4024950, 4035360)
  Pass 45 through the data...
    smallest count = 2 in the cell      4043644
    Invoking rule 47:43644=53647
    replace dpstoff3 = 4053647 if inlist(dpstoff3, 4000047, 4043644)
  Pass 46 through the data...
    smallest count = 2 in the cell      5023639
    Invoking rule 23639:24044=43644
    replace dpstoff3 = 5043644 if inlist(dpstoff3, 5023639, 5024044)
  Pass 47 through the data...
    smallest count = 2 in the cell      5025355
    WARNING: could not find any rules to collapse dpstoff3 == 5025355
  Pass 48 through the data...
    smallest count = 3 in the cell      1023940
    Invoking rule 36:23940=33640
    replace dpstoff3 = 1033640 if inlist(dpstoff3, 1000036, 1023940)
  Pass 49 through the data...
    smallest count = 3 in the cell      4000062
    Invoking rule 62:68=26268
    replace dpstoff3 = 4026268 if inlist(dpstoff3, 4000062, 4000068)
  Pass 50 through the data...
```

```
  smallest count = 3 in the cell      5022426
  Invoking rule 18:22426=31826
  replace dpstoff3 = 5031826 if inlist(dpstoff3, 5000018, 5022426)
Pass 51 through the data...
  smallest count = 4 in the cell      2023036
  WARNING: could not find any rules to collapse dpstoff3 == 2023036
Pass 52 through the data...
  smallest count = 4 in the cell      2110244
  Invoking rule 47:110244=120247
  replace dpstoff3 = 2120247 if inlist(dpstoff3, 2000047, 2110244)
Pass 53 through the data...
  smallest count = 4 in the cell      3000036
  Invoking rule 36:33944=43644
  replace dpstoff3 = 3043644 if inlist(dpstoff3, 3000036, 3033944)
Pass 54 through the data...
  smallest count = 4 in the cell      4031826
  Invoking rule 30211:31826=60226
  replace dpstoff3 = 4060226 if inlist(dpstoff3, 4030211, 4031826)
Pass 55 through the data...
  smallest count = 4 in the cell      5043644
  Invoking rule 43644:64760=103660
  replace dpstoff3 = 5103660 if inlist(dpstoff3, 5043644, 5064760)
Pass 56 through the data...
  smallest count = 5 in the cell      1000060
  Invoking rule 55:60=25560
  replace dpstoff3 = 1025560 if inlist(dpstoff3, 1000055, 1000060)
Pass 57 through the data...
  smallest count = 5 in the cell      1024950
  Invoking rule 53:24950=34953
  replace dpstoff3 = 1034953 if inlist(dpstoff3, 1000053, 1024950)
Pass 58 through the data...
  smallest count = 5 in the cell      2074968
  Invoking rule 120247:74968=190268
  replace dpstoff3 = 2190268 if inlist(dpstoff3, 2120247, 2074968)
Pass 59 through the data...
  smallest count = 5 in the cell      3025560
  Invoking rule 34953:25560=54960
  replace dpstoff3 = 3054960 if inlist(dpstoff3, 3034953, 3025560)
Pass 60 through the data...
  smallest count = 6 in the cell      4054960
  Invoking rule 53647:54960=103660
  replace dpstoff3 = 4103660 if inlist(dpstoff3, 4053647, 4054960)
Pass 61 through the data...
  smallest count = 6 in the cell      5030211
  Invoking rule 30211:31826=60226
  replace dpstoff3 = 5060226 if inlist(dpstoff3, 5030211, 5031826)
Pass 62 through the data...
  smallest count = 7 in the cell      5000068
  Invoking rule 62:68=26268
  replace dpstoff3 = 5026268 if inlist(dpstoff3, 5000062, 5000068)
Pass 63 through the data...
  smallest count = 8 in the cell      5000030
  Invoking rule 30:103660=113060
  replace dpstoff3 = 5113060 if inlist(dpstoff3, 5000030, 5103660)
Pass 64 through the data...
  smallest count = 9 in the cell      4026268
  Invoking rule 103660:26268=123668
  replace dpstoff3 = 4123668 if inlist(dpstoff3, 4103660, 4026268)
Pass 65 through the data...
  smallest count = 10 in the cell      3022426
  Invoking rule 18:22426=31826
  replace dpstoff3 = 3031826 if inlist(dpstoff3, 3000018, 3022426)
Pass 66 through the data...
  smallest count = 10 in the cell      3043644
  Invoking rule 30:43644=53044
  replace dpstoff3 = 3053044 if inlist(dpstoff3, 3000030, 3043644)
Pass 67 through the data...
  smallest count = 10 in the cell      4060226
  Invoking rule 30:60226=70230
  replace dpstoff3 = 4070230 if inlist(dpstoff3, 4000030, 4060226)
Pass 68 through the data...
  smallest count = 11 in the cell      1025560
  Invoking rule 34953:25560=54960
  replace dpstoff3 = 1054960 if inlist(dpstoff3, 1034953, 1025560)
Pass 69 through the data...
  smallest count = 12 in the cell      1020208
  Invoking rule 11:20208=30211
  replace dpstoff3 = 1030211 if inlist(dpstoff3, 1000011, 1020208)
Pass 70 through the data...
  smallest count = 12 in the cell      1033640
  Invoking rule 44:33640=43644
  replace dpstoff3 = 1043644 if inlist(dpstoff3, 1000044, 1033640)
Pass 71 through the data...
  smallest count = 13 in the cell      5060226
  Invoking rule 60226:113060=170260
  replace dpstoff3 = 5170260 if inlist(dpstoff3, 5060226, 5113060)
Pass 72 through the data...
  smallest count = 15 in the cell      1000024
  Invoking rule 24:26=22426
  replace dpstoff3 = 1022426 if inlist(dpstoff3, 1000024, 1000026)
Pass 73 through the data...
  smallest count = 15 in the cell      2190268
  Invoking rule 69:190268=200269
  replace dpstoff3 = 2200269 if inlist(dpstoff3, 2000069, 2190268)
Pass 74 through the data...
  smallest count = 15 in the cell      3030211
  Invoking rule 30211:31826=60226
  replace dpstoff3 = 3060226 if inlist(dpstoff3, 3030211, 3031826)
Pass 75 through the data...
  smallest count = 15 in the cell      3054960
  Invoking rule 62:54960=64962
  replace dpstoff3 = 3064962 if inlist(dpstoff3, 3000062, 3054960)
Pass 76 through the data...
  smallest count = 16 in the cell      5026268
  Invoking rule 170260:26268=190268
  replace dpstoff3 = 5190268 if inlist(dpstoff3, 5170260, 5026268)
Pass 77 through the data...
  smallest count = 21 in the cell      4070230
  Done collapsing! Exiting...
```

The result still isn't satisfactory, as some collapsed rules still overlap:

```
. tab alight_id dpstoff3 if daypart == 2 & mod(dpstoff3,100*100)>100

                      |    Long ID of the interaction
            alight_id |   2023036    2042639    2200269 |     Total
----------------------+---------------------------------+----------
         8. Carmenton |         0          0          1 |         1
          11. Dogville |         0          0          1 |         1
          18. East End |         0          0          1 |         1
```

```
       24. Framington |         0          0          1 |         1 
   26. Grand Junction |         0          2          0 |         2 
       30. High Point |         4          0          0 |         4 
      47. Moscow City |         0          0          6 |         6 
     49. Ninth Street |         0          0          1 |         1 
 53. Picadilly Square |         0          0          1 |         1 
      68. Toledo Town |         0          0          3 |         3 
    69. Union Station |         0          0        138 |       138 
----------------------+---------------------------------+----------
                Total |         4          2        153 |       159 
```

This overlap can be traced back to the collapsing of zero cells: first, the cell 2023036 came to being by a reasonable, at its face, collapsing of the zero cell Irvingtown (36) with non-zero cell High Point (30); and then the cell 2042639 came to being by a long overreach for the zero cell Johnsville (39) to be collapsed with a non-zero cell Grand Junction (26).

The fourth pass of weight collapse and raking: `greedy` and `maxcat()` options

The process can be improved with an additional option `greedy` that is applicable mostly to the collapsing of zero cells. It modifies behavior of `wgtcellcollapse` to require that, among the possible candidate rules with the lowest count, the rule with the *greatest* number of components is preferred. That is, the long streaks of zeroes from Irvingtown (36) to Limerick (44) in midday part would be collapsed simultaneously. To support this option, and avoid complex collapses of zero cells with the already defined cells, option `maxcategory()` specifies the greatest value of a component of a rule. By specifying `maxcategory(99)`, we can instruct `wgtcellcollapse` to only use rules that deal with individual stations, and do not use the rules that involve collapsed cells (which would have numbers of at least 20102 for the collapsed cell Alewife (1) and Brookline (2)). In the first run, those collapsed cells will always be empty ghosts, and they should not be used.

Note also that with the `greedy` option, one would want to specify the zeroes somewhere in the middle of the streak, and possibly across multiple categories of the interacting variable. In our example, specifying (zeroes(36) would collapse the midday streak of zero counts, but the need to collapse the zeroes in the night and the weekend dayparts would still remain, necessitating something like `zeroes(40)` — which, in turn, will likely create overlapping artifacts in the midday section. However, specifying `zeroes(40)` without (zeroes(36) would take care of all the streaks observed in Table ??.

```
. use trip_sample_rules, clear
. wgtcellcollapse collapse, variables(daypart board_id) mincellsize(1) ///
>         zeroes(39 44 49 60) greedy maxcategory(99) ///
>         generate(dpston4) saving(dpston4.do) replace run
Pass 0 through the data...
  smallest count = 1 in the cell      2000039
Processing zero cells...
  Invoking rule 49:50=24950 to collapse zero cells
  replace dpston4 = 1024950 if inlist(dpston4, 1000049, 1000050)
Pass 0 through the data...
  smallest count = 1 in the cell      2000039
  Invoking rule 40:44=24044 to collapse zero cells
  replace dpston4 = 2024044 if inlist(dpston4, 2000040, 2000044)
Pass 0 through the data...
  smallest count = 1 in the cell      2000039
  Invoking rule 49:50=24950 to collapse zero cells
  replace dpston4 = 2024950 if inlist(dpston4, 2000049, 2000050)
Pass 0 through the data...
  smallest count = 1 in the cell      2000039
  Invoking rule 55:60=25560 to collapse zero cells
  replace dpston4 = 2025560 if inlist(dpston4, 2000055, 2000060)
Pass 0 through the data...
  smallest count = 1 in the cell      2000039
  Invoking rule 36:39:40=33640 to collapse zero cells
  replace dpston4 = 4033640 if inlist(dpston4, 4000036, 4000039, 4000040)
Pass 0 through the data...
  smallest count = 1 in the cell      2000039
  Invoking rule 49:50=24950 to collapse zero cells
  replace dpston4 = 4024950 if inlist(dpston4, 4000049, 4000050)
Pass 0 through the data...
  smallest count = 1 in the cell      2000039
  Invoking rule 49:50:53:55:60=54960 to collapse zero cells
  replace dpston4 = 4054960 if inlist(dpston4, 4000049, 4000050, 4000053, 4000055, 4000060)
Pass 0 through the data...
  smallest count = 1 in the cell      2000039
  Invoking rule 39:40=23940 to collapse zero cells
  replace dpston4 = 5023940 if inlist(dpston4, 5000039, 5000040)
Pass 0 through the data...
  smallest count = 1 in the cell      2000039
  Invoking rule 49:50=24950 to collapse zero cells
  replace dpston4 = 5024950 if inlist(dpston4, 5000049, 5000050)
Pass 0 through the data...
  smallest count = 1 in the cell      2000039
  Invoking rule 55:60=25560 to collapse zero cells
  replace dpston4 = 5025560 if inlist(dpston4, 5000055, 5000060)
Pass 0 through the data...
  smallest count = 1 in the cell      2000039
Pass 10 through the data...
  smallest count = 1 in the cell      2000039
  Done collapsing! Exiting...
. wgtcellcollapse collapse, variables(daypart board_id) mincellsize(20) ///
>         strict feed(dpston4) saving(dpston4.do) append run
Pass 10 through the data...
  smallest count = 1 in the cell      2000039
  Invoking rule 39:24044=33944
  replace dpston4 = 2033944 if inlist(dpston4, 2000039, 2024044)
Pass 11 through the data...
  smallest count = 1 in the cell      2024950
  Invoking rule 53:24950=34953
  replace dpston4 = 2034953 if inlist(dpston4, 2000053, 2024950)
Pass 12 through the data...
  smallest count = 1 in the cell      2025560
  Invoking rule 34953:25560=54960
  replace dpston4 = 2054960 if inlist(dpston4, 2034953, 2025560)
Pass 13 through the data...
  smallest count = 1 in the cell      3000039
  Invoking rule 39:40=23940
  replace dpston4 = 3023940 if inlist(dpston4, 3000039, 3000040)
```

```
Pass 14 through the data...
  smallest count = 1 in the cell      4000044
  Invoking rule 44:33640=43644
  replace dpston4 = 4043644 if inlist(dpston4, 4000044, 4033640)
Pass 15 through the data...
  smallest count = 1 in the cell      4024950
  Invoking rule 47:24950=34750
  replace dpston4 = 4034750 if inlist(dpston4, 4000047, 4024950)
Pass 16 through the data...
  smallest count = 1 in the cell      5000024
  Invoking rule 18:24=21824
  replace dpston4 = 5021824 if inlist(dpston4, 5000018, 5000024)
Pass 17 through the data...
  smallest count = 1 in the cell      5023940
  Invoking rule 44:23940=33944
  replace dpston4 = 5033944 if inlist(dpston4, 5000044, 5023940)
Pass 18 through the data...
  smallest count = 1 in the cell      5024950
  Invoking rule 53:24950=34953
  replace dpston4 = 5034953 if inlist(dpston4, 5000053, 5024950)
Pass 19 through the data...
  smallest count = 2 in the cell      2000036
  Invoking rule 36:33944=43644
  replace dpston4 = 2043644 if inlist(dpston4, 2000036, 2033944)
Pass 20 through the data...
  smallest count = 2 in the cell      4043644
  Invoking rule 43644:34750=73650
  replace dpston4 = 4073650 if inlist(dpston4, 4043644, 4034750)
Pass 21 through the data...
  smallest count = 2 in the cell      5025560
  Invoking rule 34953:25560=54960
  replace dpston4 = 5054960 if inlist(dpston4, 5034953, 5025560)
Pass 22 through the data...
  smallest count = 3 in the cell      2000024
  Invoking rule 18:24=21824
  replace dpston4 = 2021824 if inlist(dpston4, 2000018, 2000024)
Pass 23 through the data...
  smallest count = 3 in the cell      3023940
  Invoking rule 44:23940=33944
  replace dpston4 = 3033944 if inlist(dpston4, 3000044, 3023940)
Pass 24 through the data...
  smallest count = 3 in the cell      4000024
  Invoking rule 18:24=21824
  replace dpston4 = 4021824 if inlist(dpston4, 4000018, 4000024)
Pass 25 through the data...
  smallest count = 3 in the cell      5000001
  Invoking rule 1:2=20102
  replace dpston4 = 5020102 if inlist(dpston4, 5000001, 5000002)
Pass 26 through the data...
  smallest count = 3 in the cell      5000068
  Invoking rule 62:68=26268
  replace dpston4 = 5026268 if inlist(dpston4, 5000062, 5000068)
Pass 27 through the data...
  smallest count = 4 in the cell      2000001
  Invoking rule 1:2=20102
  replace dpston4 = 2020102 if inlist(dpston4, 2000001, 2000002)
Pass 28 through the data...
  smallest count = 4 in the cell      2000008
  Invoking rule 8:20102=30108
  replace dpston4 = 2030108 if inlist(dpston4, 2000008, 2020102)
```

```
Pass 29 through the data...
  smallest count = 4 in the cell      2043644
  Invoking rule 30:43644=53044
  replace dpston4 = 2053044 if inlist(dpston4, 2000030, 2043644)
Pass 30 through the data...
  smallest count = 4 in the cell      3000050
  Invoking rule 49:50=24950
  replace dpston4 = 3024950 if inlist(dpston4, 3000049, 3000050)
Pass 31 through the data...
  smallest count = 4 in the cell      5033944
  Invoking rule 47:33944=43947
  replace dpston4 = 5043947 if inlist(dpston4, 5000047, 5033944)
Pass 32 through the data...
  smallest count = 5 in the cell      1000039
  Invoking rule 39:40=23940
  replace dpston4 = 1023940 if inlist(dpston4, 1000039, 1000040)
Pass 33 through the data...
  smallest count = 5 in the cell      3000060
  Invoking rule 55:60=25560
  replace dpston4 = 3025560 if inlist(dpston4, 3000055, 3000060)
Pass 34 through the data...
  smallest count = 5 in the cell      4000026
  Invoking rule 26:21824=31826
  replace dpston4 = 4031826 if inlist(dpston4, 4000026, 4021824)
Pass 35 through the data...
  smallest count = 5 in the cell      5021824
  Invoking rule 26:21824=31826
  replace dpston4 = 5031826 if inlist(dpston4, 5000026, 5021824)
Pass 36 through the data...
  smallest count = 6 in the cell      2000068
  Invoking rule 62:68=26268
  replace dpston4 = 2026268 if inlist(dpston4, 2000062, 2000068)
Pass 37 through the data...
  smallest count = 6 in the cell      2054960
  Invoking rule 54960:26268=74968
  replace dpston4 = 2074968 if inlist(dpston4, 2054960, 2026268)
Pass 38 through the data...
  smallest count = 6 in the cell      4000002
  Invoking rule 1:2=20102
  replace dpston4 = 4020102 if inlist(dpston4, 4000001, 4000002)
Pass 39 through the data...
  smallest count = 7 in the cell      4054960
  Invoking rule 62:54960=64962
  replace dpston4 = 4064962 if inlist(dpston4, 4000062, 4054960)
Pass 40 through the data...
  smallest count = 8 in the cell      2021824
  Invoking rule 26:21824=31826
  replace dpston4 = 2031826 if inlist(dpston4, 2000026, 2021824)
Pass 41 through the data...
  smallest count = 8 in the cell      5054960
  Invoking rule 43947:54960=93960
  replace dpston4 = 5093960 if inlist(dpston4, 5043947, 5054960)
Pass 42 through the data...
  smallest count = 10 in the cell      1000055
  Invoking rule 55:60=25560
  replace dpston4 = 1025560 if inlist(dpston4, 1000055, 1000060)
Pass 43 through the data...
  smallest count = 10 in the cell      4073650
  Invoking rule 30:73650=83050
  replace dpston4 = 4083050 if inlist(dpston4, 4000030, 4073650)
```

```
Pass 44 through the data...
  smallest count = 10 in the cell      5020102
  Invoking rule 8:20102=30108
  replace dpston4 = 5030108 if inlist(dpston4, 5000008, 5020102)
Pass 45 through the data...
  smallest count = 11 in the cell      3000001
  Invoking rule 1:2=20102
  replace dpston4 = 3020102 if inlist(dpston4, 3000001, 3000002)
Pass 46 through the data...
  smallest count = 11 in the cell      4000068
  Invoking rule 68:64962=74968
  replace dpston4 = 4074968 if inlist(dpston4, 4000068, 4064962)
Pass 47 through the data...
  smallest count = 11 in the cell      5031826
  Invoking rule 30:31826=41830
  replace dpston4 = 5041830 if inlist(dpston4, 5000030, 5031826)
Pass 48 through the data...
  smallest count = 12 in the cell      2030108
  Invoking rule 11:30108=40111
  replace dpston4 = 2040111 if inlist(dpston4, 2000011, 2030108)
Pass 49 through the data...
  smallest count = 12 in the cell      3033944
  Invoking rule 36:33944=43644
  replace dpston4 = 3043644 if inlist(dpston4, 3000036, 3033944)
Pass 50 through the data...
  smallest count = 12 in the cell      4031826
  Invoking rule 11:31826=41126
  replace dpston4 = 4041126 if inlist(dpston4, 4000011, 4031826)
Pass 51 through the data...
  smallest count = 13 in the cell      1024950
  Invoking rule 53:24950=34953
  replace dpston4 = 1034953 if inlist(dpston4, 1000053, 1024950)
Pass 52 through the data...
  smallest count = 13 in the cell      3024950
  Invoking rule 53:24950=34953
  replace dpston4 = 3034953 if inlist(dpston4, 3000053, 3024950)
Pass 53 through the data...
  smallest count = 13 in the cell      4020102
  Invoking rule 8:20102=30108
  replace dpston4 = 4030108 if inlist(dpston4, 4000008, 4020102)
Pass 54 through the data...
  smallest count = 15 in the cell      5000036
  Invoking rule 36:41830=51836
  replace dpston4 = 5051836 if inlist(dpston4, 5000036, 5041830)
Pass 55 through the data...
  smallest count = 19 in the cell      3025560
  Invoking rule 62:25560=35562
  replace dpston4 = 3035562 if inlist(dpston4, 3000062, 3025560)
Pass 56 through the data...
  smallest count = 20 in the cell      5026268
  Done collapsing! Exiting...

. assert "`r(failed)´" == ""

. wgtcellcollapse collapse, variables(daypart alight_id) mincellsize(1) ///
>         zeroes(2 40 49 50 60) greedy maxcategory(99) ///
>         generate(dpstoff4) saving(dpstoff4.do) replace run
Pass 0 through the data...
  smallest count = 1 in the cell      1000002

Processing zero cells...

  Invoking rule 39:40=23940 to collapse zero cells
  replace dpstoff4 = 1023940 if inlist(dpstoff4, 1000039, 1000040)
Pass 0 through the data...
  smallest count = 1 in the cell      1000002
  Invoking rule 1:2:8=30108 to collapse zero cells
  replace dpstoff4 = 2030108 if inlist(dpstoff4, 2000001, 2000002, 2000008)
Pass 0 through the data...
  smallest count = 1 in the cell      1000002
  Invoking rule 30:36:39:40:44=53044 to collapse zero cells
  replace dpstoff4 = 2053044 if inlist(dpstoff4, 2000030, 2000036, 2000039, 2000040, 2000044)
Pass 0 through the data...
  smallest count = 1 in the cell      1000002
  Invoking rule 50:53:55:60:62=55062 to collapse zero cells
  replace dpstoff4 = 2055062 if inlist(dpstoff4, 2000050, 2000053, 2000055, 2000060, 2000062)
Pass 0 through the data...
  smallest count = 1 in the cell      1000002
  Invoking rule 1:2:8=30108 to collapse zero cells
  replace dpstoff4 = 3030108 if inlist(dpstoff4, 3000001, 3000002, 3000008)
Pass 0 through the data...
  smallest count = 1 in the cell      1000002
  Invoking rule 55:60=25560 to collapse zero cells
  replace dpstoff4 = 3025560 if inlist(dpstoff4, 3000055, 3000060)
Pass 0 through the data...
  smallest count = 1 in the cell      1000002
  Invoking rule 1:2:8:11=40111 to collapse zero cells
  replace dpstoff4 = 4040111 if inlist(dpstoff4, 4000001, 4000002, 4000008, 4000011)
Pass 0 through the data...
  smallest count = 1 in the cell      1000002
  Invoking rule 36:39:40:44=43644 to collapse zero cells
  replace dpstoff4 = 4043644 if inlist(dpstoff4, 4000036, 4000039, 4000040, 4000044)
Pass 0 through the data...
  smallest count = 1 in the cell      1000002
  Invoking rule 49:50=24950 to collapse zero cells
  replace dpstoff4 = 4024950 if inlist(dpstoff4, 4000049, 4000050)
Pass 0 through the data...
  smallest count = 1 in the cell      1000002
  Invoking rule 55:60=25560 to collapse zero cells
  replace dpstoff4 = 4025560 if inlist(dpstoff4, 4000055, 4000060)
Pass 0 through the data...
  smallest count = 1 in the cell      1000002
  Invoking rule 1:2:8=30108 to collapse zero cells
  replace dpstoff4 = 5030108 if inlist(dpstoff4, 5000001, 5000002, 5000008)
Pass 0 through the data...
  smallest count = 1 in the cell      1000002
  Invoking rule 36:39:40=33640 to collapse zero cells
  replace dpstoff4 = 5033640 if inlist(dpstoff4, 5000036, 5000039, 5000040)
Pass 0 through the data...
  smallest count = 1 in the cell      1000002
  Invoking rule 49:50=24950 to collapse zero cells
  replace dpstoff4 = 5024950 if inlist(dpstoff4, 5000049, 5000050)
Pass 0 through the data...
  smallest count = 1 in the cell      1000002
  Invoking rule 49:50:53:55:60=54960 to collapse zero cells
  replace dpstoff4 = 5054960 if inlist(dpstoff4, 5000049, 5000050, 5000053, 5000055, 5000060)
Pass 0 through the data...
  smallest count = 1 in the cell      1000002
Pass 14 through the data...
  smallest count = 1 in the cell      1000002
  Done collapsing! Exiting...
. wgtcellcollapse collapse, variables(daypart alight_id) mincellsize(20) ///
>         strict feed(dpstoff4) saving(dpstoff4.do) append run
```

```
Pass 14 through the data...
  smallest count = 1 in the cell      1000002
  Invoking rule 2:8=20208
  replace dpstoff4 = 1020208 if inlist(dpstoff4, 1000002, 1000008)
Pass 15 through the data...
  smallest count = 1 in the cell      2000011
  Invoking rule 11:18=21118
  replace dpstoff4 = 2021118 if inlist(dpstoff4, 2000011, 2000018)
Pass 16 through the data...
  smallest count = 1 in the cell      2000024
  Invoking rule 24:26=22426
  replace dpstoff4 = 2022426 if inlist(dpstoff4, 2000024, 2000026)
Pass 17 through the data...
  smallest count = 1 in the cell      2000049
  Invoking rule 49:55062=64962
  replace dpstoff4 = 2064962 if inlist(dpstoff4, 2000049, 2055062)
Pass 18 through the data...
  smallest count = 1 in the cell      2030108
  Invoking rule 30108:21118=50118
  replace dpstoff4 = 2050118 if inlist(dpstoff4, 2030108, 2021118)
Pass 19 through the data...
  smallest count = 1 in the cell      3000039
  Invoking rule 39:40=23940
  replace dpstoff4 = 3023940 if inlist(dpstoff4, 3000039, 3000040)
Pass 20 through the data...
  smallest count = 1 in the cell      3000049
  Invoking rule 49:50=24950
  replace dpstoff4 = 3024950 if inlist(dpstoff4, 3000049, 3000050)
Pass 21 through the data...
  smallest count = 1 in the cell      3030108
  Invoking rule 11:30108=40111
  replace dpstoff4 = 3040111 if inlist(dpstoff4, 3000011, 3030108)
Pass 22 through the data...
  smallest count = 1 in the cell      4000018
  Invoking rule 18:24=21824
  replace dpstoff4 = 4021824 if inlist(dpstoff4, 4000018, 4000024)
Pass 23 through the data...
  smallest count = 1 in the cell      4000026
  Invoking rule 26:21824=31826
  replace dpstoff4 = 4031826 if inlist(dpstoff4, 4000026, 4021824)
Pass 24 through the data...
  smallest count = 1 in the cell      4025560
  Invoking rule 53:25560=35360
  replace dpstoff4 = 4035360 if inlist(dpstoff4, 4000053, 4025560)
Pass 25 through the data...
  smallest count = 1 in the cell      5000026
  Invoking rule 24:26=22426
  replace dpstoff4 = 5022426 if inlist(dpstoff4, 5000024, 5000026)
Pass 26 through the data...
  smallest count = 1 in the cell      5024950
  Invoking rule 47:24950=34750
  replace dpstoff4 = 5034750 if inlist(dpstoff4, 5000047, 5024950)
Pass 27 through the data...
  smallest count = 1 in the cell      5030108
  Invoking rule 11:30108=40111
  replace dpstoff4 = 5040111 if inlist(dpstoff4, 5000011, 5030108)
Pass 28 through the data...
  smallest count = 2 in the cell      1000050
  Invoking rule 49:50=24950
  replace dpstoff4 = 1024950 if inlist(dpstoff4, 1000049, 1000050)
```

```
Pass 29 through the data...
  smallest count = 2 in the cell      2064962
  Invoking rule 68:64962=74968
  replace dpstoff4 = 2074968 if inlist(dpstoff4, 2000068, 2064962)
Pass 30 through the data...
  smallest count = 2 in the cell      3000024
  Invoking rule 24:26=22426
  replace dpstoff4 = 3022426 if inlist(dpstoff4, 3000024, 3000026)
Pass 31 through the data...
  smallest count = 2 in the cell      3000044
  Invoking rule 44:23940=33944
  replace dpstoff4 = 3033944 if inlist(dpstoff4, 3000044, 3023940)
Pass 32 through the data...
  smallest count = 2 in the cell      3024950
  Invoking rule 53:24950=34953
  replace dpstoff4 = 3034953 if inlist(dpstoff4, 3000053, 3024950)
Pass 33 through the data...
  smallest count = 2 in the cell      4024950
  Invoking rule 24950:35360=54960
  replace dpstoff4 = 4054960 if inlist(dpstoff4, 4024950, 4035360)
Pass 34 through the data...
  smallest count = 2 in the cell      4043644
  Invoking rule 47:43644=53647
  replace dpstoff4 = 4053647 if inlist(dpstoff4, 4000047, 4043644)
Pass 35 through the data...
  smallest count = 2 in the cell      5000044
  Invoking rule 44:33640=43644
  replace dpstoff4 = 5043644 if inlist(dpstoff4, 5000044, 5033640)
Pass 36 through the data...
  smallest count = 2 in the cell      5054960
  Invoking rule 62:54960=64962
  replace dpstoff4 = 5064962 if inlist(dpstoff4, 5000062, 5054960)
Pass 37 through the data...
  smallest count = 3 in the cell      1023940
  Invoking rule 36:23940=33640
  replace dpstoff4 = 1033640 if inlist(dpstoff4, 1000036, 1023940)
Pass 38 through the data...
  smallest count = 3 in the cell      2022426
  Invoking rule 50118:22426=70126
  replace dpstoff4 = 2070126 if inlist(dpstoff4, 2050118, 2022426)
Pass 39 through the data...
  smallest count = 3 in the cell      4000062
  Invoking rule 62:68=26268
  replace dpstoff4 = 4026268 if inlist(dpstoff4, 4000062, 4000068)
Pass 40 through the data...
  smallest count = 3 in the cell      5022426
  Invoking rule 18:22426=31826
  replace dpstoff4 = 5031826 if inlist(dpstoff4, 5000018, 5022426)
Pass 41 through the data...
  smallest count = 4 in the cell      2053044
  Invoking rule 47:53044=63047
  replace dpstoff4 = 2063047 if inlist(dpstoff4, 2000047, 2053044)
Pass 42 through the data...
  smallest count = 4 in the cell      3000036
  Invoking rule 36:33944=43644
  replace dpstoff4 = 3043644 if inlist(dpstoff4, 3000036, 3033944)
Pass 43 through the data...
  smallest count = 4 in the cell      4031826
  Invoking rule 40111:31826=70126
  replace dpstoff4 = 4070126 if inlist(dpstoff4, 4040111, 4031826)
```

```
Pass 44 through the data...
  smallest count = 4 in the cell      5043644
  Invoking rule 43644:34750=73650
  replace dpstoff4 = 5073650 if inlist(dpstoff4, 5043644, 5034750)
Pass 45 through the data...
  smallest count = 5 in the cell      1000060
  Invoking rule 55:60=25560
  replace dpstoff4 = 1025560 if inlist(dpstoff4, 1000055, 1000060)
Pass 46 through the data...
  smallest count = 5 in the cell      1024950
  Invoking rule 53:24950=34953
  replace dpstoff4 = 1034953 if inlist(dpstoff4, 1000053, 1024950)
Pass 47 through the data...
  smallest count = 5 in the cell      2074968
  Invoking rule 63047:74968=133068
  replace dpstoff4 = 2133068 if inlist(dpstoff4, 2063047, 2074968)
Pass 48 through the data...
  smallest count = 5 in the cell      3025560
  Invoking rule 34953:25560=54960
  replace dpstoff4 = 3054960 if inlist(dpstoff4, 3034953, 3025560)
Pass 49 through the data...
  smallest count = 6 in the cell      2070126
  Invoking rule 70126:133068=200168
  replace dpstoff4 = 2200168 if inlist(dpstoff4, 2070126, 2133068)
Pass 50 through the data...
  smallest count = 6 in the cell      4054960
  Invoking rule 53647:54960=103660
  replace dpstoff4 = 4103660 if inlist(dpstoff4, 4053647, 4054960)
Pass 51 through the data...
  smallest count = 6 in the cell      5040111
  Invoking rule 40111:31826=70126
  replace dpstoff4 = 5070126 if inlist(dpstoff4, 5040111, 5031826)
Pass 52 through the data...
  smallest count = 7 in the cell      5000068
  Invoking rule 68:64962=74968
  replace dpstoff4 = 5074968 if inlist(dpstoff4, 5000068, 5064962)
Pass 53 through the data...
  smallest count = 8 in the cell      5000030
  Invoking rule 30:73650=83050
  replace dpstoff4 = 5083050 if inlist(dpstoff4, 5000030, 5073650)
Pass 54 through the data...
  smallest count = 9 in the cell      4026268
  Invoking rule 103660:26268=123668
  replace dpstoff4 = 4123668 if inlist(dpstoff4, 4103660, 4026268)
Pass 55 through the data...
  smallest count = 10 in the cell      3022426
  Invoking rule 18:22426=31826
  replace dpstoff4 = 3031826 if inlist(dpstoff4, 3000018, 3022426)
Pass 56 through the data...
  smallest count = 10 in the cell      3043644
  Invoking rule 30:43644=53044
  replace dpstoff4 = 3053044 if inlist(dpstoff4, 3000030, 3043644)
Pass 57 through the data...
  smallest count = 10 in the cell      4070126
  Invoking rule 30:70126=80130
  replace dpstoff4 = 4080130 if inlist(dpstoff4, 4000030, 4070126)
Pass 58 through the data...
  smallest count = 11 in the cell      1025560
  Invoking rule 34953:25560=54960
  replace dpstoff4 = 1054960 if inlist(dpstoff4, 1034953, 1025560)
```

```
Pass 59 through the data...
  smallest count = 12 in the cell      1020208
  Invoking rule 11:20208=30211
  replace dpstoff4 = 1030211 if inlist(dpstoff4, 1000011, 1020208)
Pass 60 through the data...
  smallest count = 12 in the cell      1033640
  Invoking rule 44:33640=43644
  replace dpstoff4 = 1043644 if inlist(dpstoff4, 1000044, 1033640)
Pass 61 through the data...
  smallest count = 13 in the cell      5070126
  Invoking rule 70126:83050=150150
  replace dpstoff4 = 5150150 if inlist(dpstoff4, 5070126, 5083050)
Pass 62 through the data...
  smallest count = 15 in the cell      1000024
  Invoking rule 24:26=22426
  replace dpstoff4 = 1022426 if inlist(dpstoff4, 1000024, 1000026)
Pass 63 through the data...
  smallest count = 15 in the cell      3040111
  Invoking rule 40111:31826=70126
  replace dpstoff4 = 3070126 if inlist(dpstoff4, 3040111, 3031826)
Pass 64 through the data...
  smallest count = 15 in the cell      3054960
  Invoking rule 62:54960=64962
  replace dpstoff4 = 3064962 if inlist(dpstoff4, 3000062, 3054960)
Pass 65 through the data...
  smallest count = 18 in the cell      5074968
  Invoking rule 69:74968=84969
  replace dpstoff4 = 5084969 if inlist(dpstoff4, 5000069, 5074968)
Pass 66 through the data...
  smallest count = 21 in the cell      2200168
  Done collapsing! Exiting...
. assert "`r(failed)´" == ""
```

We have finally been able to produce a clean collapse of everything! Note the use of `assert "‘r(failed)’"==""` in the above code snippet to make sure that all cells have the minimal required size of 20.

As a very minor point, we can see some room for improvement in collapsing the cells on the weekend:

```
. tab alight_id dpstoff4 if daypart == 5 & mod(dpstoff4,100*100)>100
```

	Long ID of the interaction		
alight_id	5084969	5150150	Total
8. Carmenton	0	1	1
11. Dogville	0	5	5
18. East End	0	4	4
24. Framington	0	2	2
26. Grand Junction	0	1	1
30. High Point	0	8	8
36. Irvingtown	0	2	2
44. Limerick	0	2	2
47. Moscow City	0	6	6
50. Ontario Lake	0	1	1
53. Picadilly Square	2	0	2
62. Silver Spring	9	0	9
68. Toledo Town	7	0	7

```
    69. Union Station |       123          0 |       123
----------------------+----------------------+----------
                Total |       141         32 |       173
```

Instead of two cells with sizes 141 and 32, it seems like we could produce three cells, with Union Station (69) being its own cell, and everything else split somewhere in the middle.

The fifth pass of weight collapse and raking: *if* conditions

We will now code the collapsing cells for that day part "by hand", and we will put those custom coded cells upfront before the main run.

```
. use trip_sample_rules, clear
. wgtcellcollapse collapse, variables(daypart board_id) mincellsize(1) ///
>         zeroes(39 44 49 60) greedy maxcategory(99) ///
>         generate(dpston5) saving(dpston5.do) replace run
Pass 0 through the data...
  smallest count = 1 in the cell      2000039
Processing zero cells...
  Invoking rule 49:50=24950 to collapse zero cells
  replace dpston5 = 1024950 if inlist(dpston5, 1000049, 1000050)
Pass 0 through the data...
  smallest count = 1 in the cell      2000039
  Invoking rule 40:44=24044 to collapse zero cells
  replace dpston5 = 2024044 if inlist(dpston5, 2000040, 2000044)
Pass 0 through the data...
  smallest count = 1 in the cell      2000039
  Invoking rule 49:50=24950 to collapse zero cells
  replace dpston5 = 2024950 if inlist(dpston5, 2000049, 2000050)
Pass 0 through the data...
  smallest count = 1 in the cell      2000039
  Invoking rule 55:60=25560 to collapse zero cells
  replace dpston5 = 2025560 if inlist(dpston5, 2000055, 2000060)
Pass 0 through the data...
  smallest count = 1 in the cell      2000039
  Invoking rule 36:39:40=33640 to collapse zero cells
  replace dpston5 = 4033640 if inlist(dpston5, 4000036, 4000039, 4000040)
Pass 0 through the data...
  smallest count = 1 in the cell      2000039
  Invoking rule 49:50=24950 to collapse zero cells
  replace dpston5 = 4024950 if inlist(dpston5, 4000049, 4000050)
Pass 0 through the data...
  smallest count = 1 in the cell      2000039
  Invoking rule 49:50:53:55:60=54960 to collapse zero cells
  replace dpston5 = 4054960 if inlist(dpston5, 4000049, 4000050, 4000053, 4000055, 4000060)
Pass 0 through the data...
  smallest count = 1 in the cell      2000039
  Invoking rule 39:40=23940 to collapse zero cells
  replace dpston5 = 5023940 if inlist(dpston5, 5000039, 5000040)
Pass 0 through the data...
  smallest count = 1 in the cell      2000039
  Invoking rule 49:50=24950 to collapse zero cells
  replace dpston5 = 5024950 if inlist(dpston5, 5000049, 5000050)
Pass 0 through the data...
```

```
    smallest count = 1 in the cell      2000039
    Invoking rule 55:60=25560 to collapse zero cells
    replace dpston5 = 5025560 if inlist(dpston5, 5000055, 5000060)
  Pass 0 through the data...
    smallest count = 1 in the cell      2000039
  Pass 10 through the data...
    smallest count = 1 in the cell      2000039
    Done collapsing! Exiting...
  . wgtcellcollapse collapse, variables(daypart board_id) mincellsize(20) ///
  >         strict feed(dpston5) saving(dpston5.do) append run
  Pass 10 through the data...
    smallest count = 1 in the cell      2000039
    Invoking rule 39:24044=33944
    replace dpston5 = 2033944 if inlist(dpston5, 2000039, 2024044)
  Pass 11 through the data...
    smallest count = 1 in the cell      2024950
    Invoking rule 53:24950=34953
    replace dpston5 = 2034953 if inlist(dpston5, 2000053, 2024950)
  Pass 12 through the data...
    smallest count = 1 in the cell      2025560
    Invoking rule 34953:25560=54960
    replace dpston5 = 2054960 if inlist(dpston5, 2034953, 2025560)
  Pass 13 through the data...
    smallest count = 1 in the cell      3000039
    Invoking rule 39:40=23940
    replace dpston5 = 3023940 if inlist(dpston5, 3000039, 3000040)
  Pass 14 through the data...
    smallest count = 1 in the cell      4000044
    Invoking rule 44:33640=43644
    replace dpston5 = 4043644 if inlist(dpston5, 4000044, 4033640)
  Pass 15 through the data...
    smallest count = 1 in the cell      4024950
    Invoking rule 47:24950=34750
    replace dpston5 = 4034750 if inlist(dpston5, 4000047, 4024950)
  Pass 16 through the data...
    smallest count = 1 in the cell      5000024
    Invoking rule 18:24=21824
    replace dpston5 = 5021824 if inlist(dpston5, 5000018, 5000024)
  Pass 17 through the data...
    smallest count = 1 in the cell      5023940
    Invoking rule 44:23940=33944
    replace dpston5 = 5033944 if inlist(dpston5, 5000044, 5023940)
  Pass 18 through the data...
    smallest count = 1 in the cell      5024950
    Invoking rule 53:24950=34953
    replace dpston5 = 5034953 if inlist(dpston5, 5000053, 5024950)
  Pass 19 through the data...
    smallest count = 2 in the cell      2000036
    Invoking rule 36:33944=43644
    replace dpston5 = 2043644 if inlist(dpston5, 2000036, 2033944)
  Pass 20 through the data...
    smallest count = 2 in the cell      4043644
    Invoking rule 43644:34750=73650
    replace dpston5 = 4073650 if inlist(dpston5, 4043644, 4034750)
  Pass 21 through the data...
    smallest count = 2 in the cell      5025560
    Invoking rule 34953:25560=54960
    replace dpston5 = 5054960 if inlist(dpston5, 5034953, 5025560)
  Pass 22 through the data...
    smallest count = 3 in the cell      2000024
    Invoking rule 18:24=21824
    replace dpston5 = 2021824 if inlist(dpston5, 2000018, 2000024)
  Pass 23 through the data...
    smallest count = 3 in the cell      3023940
    Invoking rule 44:23940=33944
    replace dpston5 = 3033944 if inlist(dpston5, 3000044, 3023940)
  Pass 24 through the data...
    smallest count = 3 in the cell      4000024
    Invoking rule 18:24=21824
    replace dpston5 = 4021824 if inlist(dpston5, 4000018, 4000024)
  Pass 25 through the data...
    smallest count = 3 in the cell      5000001
    Invoking rule 1:2=20102
    replace dpston5 = 5020102 if inlist(dpston5, 5000001, 5000002)
  Pass 26 through the data...
    smallest count = 3 in the cell      5000068
    Invoking rule 62:68=26268
    replace dpston5 = 5026268 if inlist(dpston5, 5000062, 5000068)
  Pass 27 through the data...
    smallest count = 4 in the cell      2000001
    Invoking rule 1:2=20102
    replace dpston5 = 2020102 if inlist(dpston5, 2000001, 2000002)
  Pass 28 through the data...
    smallest count = 4 in the cell      2000008
    Invoking rule 8:20102=30108
    replace dpston5 = 2030108 if inlist(dpston5, 2000008, 2020102)
  Pass 29 through the data...
    smallest count = 4 in the cell      2043644
    Invoking rule 30:43644=53044
    replace dpston5 = 2053044 if inlist(dpston5, 2000030, 2043644)
  Pass 30 through the data...
    smallest count = 4 in the cell      3000050
    Invoking rule 49:50=24950
    replace dpston5 = 3024950 if inlist(dpston5, 3000049, 3000050)
  Pass 31 through the data...
    smallest count = 4 in the cell      5033944
    Invoking rule 47:33944=43947
    replace dpston5 = 5043947 if inlist(dpston5, 5000047, 5033944)
  Pass 32 through the data...
    smallest count = 5 in the cell      1000039
    Invoking rule 39:40=23940
    replace dpston5 = 1023940 if inlist(dpston5, 1000039, 1000040)
  Pass 33 through the data...
    smallest count = 5 in the cell      3000060
    Invoking rule 55:60=25560
    replace dpston5 = 3025560 if inlist(dpston5, 3000055, 3000060)
  Pass 34 through the data...
    smallest count = 5 in the cell      4000026
    Invoking rule 26:21824=31826
    replace dpston5 = 4031826 if inlist(dpston5, 4000026, 4021824)
  Pass 35 through the data...
    smallest count = 5 in the cell      5021824
    Invoking rule 26:21824=31826
    replace dpston5 = 5031826 if inlist(dpston5, 5000026, 5021824)
  Pass 36 through the data...
    smallest count = 6 in the cell      2000068
    Invoking rule 62:68=26268
    replace dpston5 = 2026268 if inlist(dpston5, 2000062, 2000068)
  Pass 37 through the data...
    smallest count = 6 in the cell      2054960
    Invoking rule 54960:26268=74968
    replace dpston5 = 2074968 if inlist(dpston5, 2054960, 2026268)
  Pass 38 through the data...
    smallest count = 6 in the cell      4000002
    Invoking rule 1:2=20102
    replace dpston5 = 4020102 if inlist(dpston5, 4000001, 4000002)
  Pass 39 through the data...
    smallest count = 7 in the cell      4054960
    Invoking rule 62:54960=64962
    replace dpston5 = 4064962 if inlist(dpston5, 4000062, 4054960)
  Pass 40 through the data...
    smallest count = 8 in the cell      2021824
    Invoking rule 26:21824=31826
    replace dpston5 = 2031826 if inlist(dpston5, 2000026, 2021824)
  Pass 41 through the data...
    smallest count = 8 in the cell      5054960
    Invoking rule 43947:54960=93960
    replace dpston5 = 5093960 if inlist(dpston5, 5043947, 5054960)
  Pass 42 through the data...
    smallest count = 10 in the cell      1000055
    Invoking rule 55:60=25560
    replace dpston5 = 1025560 if inlist(dpston5, 1000055, 1000060)
  Pass 43 through the data...
    smallest count = 10 in the cell      4073650
    Invoking rule 30:73650=83050
    replace dpston5 = 4083050 if inlist(dpston5, 4000030, 4073650)
  Pass 44 through the data...
    smallest count = 10 in the cell      5020102
    Invoking rule 8:20102=30108
    replace dpston5 = 5030108 if inlist(dpston5, 5000008, 5020102)
  Pass 45 through the data...
    smallest count = 11 in the cell      3000001
    Invoking rule 1:2=20102
    replace dpston5 = 3020102 if inlist(dpston5, 3000001, 3000002)
  Pass 46 through the data...
    smallest count = 11 in the cell      4000068
    Invoking rule 68:64962=74968
    replace dpston5 = 4074968 if inlist(dpston5, 4000068, 4064962)
  Pass 47 through the data...
    smallest count = 11 in the cell      5031826
    Invoking rule 30:31826=41830
    replace dpston5 = 5041830 if inlist(dpston5, 5000030, 5031826)
  Pass 48 through the data...
    smallest count = 12 in the cell      2030108
    Invoking rule 11:30108=40111
    replace dpston5 = 2040111 if inlist(dpston5, 2000011, 2030108)
  Pass 49 through the data...
    smallest count = 12 in the cell      3033944
    Invoking rule 36:33944=43644
    replace dpston5 = 3043644 if inlist(dpston5, 3000036, 3033944)
  Pass 50 through the data...
    smallest count = 12 in the cell      4031826
    Invoking rule 11:31826=41126
    replace dpston5 = 4041126 if inlist(dpston5, 4000011, 4031826)
  Pass 51 through the data...
    smallest count = 13 in the cell      1024950
    Invoking rule 53:24950=34953
    replace dpston5 = 1034953 if inlist(dpston5, 1000053, 1024950)
  Pass 52 through the data...
    smallest count = 13 in the cell      3024950
    Invoking rule 53:24950=34953
    replace dpston5 = 3034953 if inlist(dpston5, 3000053, 3024950)
  Pass 53 through the data...
    smallest count = 13 in the cell      4020102
    Invoking rule 8:20102=30108
    replace dpston5 = 4030108 if inlist(dpston5, 4000008, 4020102)
  Pass 54 through the data...
    smallest count = 15 in the cell      5000036
    Invoking rule 36:41830=51836
    replace dpston5 = 5051836 if inlist(dpston5, 5000036, 5041830)
  Pass 55 through the data...
    smallest count = 19 in the cell      3025560
    Invoking rule 62:25560=35562
    replace dpston5 = 3035562 if inlist(dpston5, 3000062, 3025560)
  Pass 56 through the data...
    smallest count = 20 in the cell      5026268
    Done collapsing! Exiting...
  . assert "`r(failed)´" == ""
  . wgtcellcollapse collapse, variables(daypart alight_id) mincellsize(1) ///
  >         zeroes(2 40 49 50 60) greedy maxcategory(99) ///
  >         generate(dpstoff5) saving(dpstoff5.do) replace run
  Pass 0 through the data...
    smallest count = 1 in the cell      1000002
  Processing zero cells...
    Invoking rule 39:40=23940 to collapse zero cells
    replace dpstoff5 = 1023940 if inlist(dpstoff5, 1000039, 1000040)
  Pass 0 through the data...
    smallest count = 1 in the cell      1000002
    Invoking rule 1:2:8=30108 to collapse zero cells
    replace dpstoff5 = 2030108 if inlist(dpstoff5, 2000001, 2000002, 2000008)
  Pass 0 through the data...
    smallest count = 1 in the cell      1000002
    Invoking rule 30:36:39:40:44=53044 to collapse zero cells
    replace dpstoff5 = 2053044 if inlist(dpstoff5, 2000030, 2000036, 2000039, 2000040, 2000044)
  Pass 0 through the data...
    smallest count = 1 in the cell      1000002
    Invoking rule 50:53:55:60:62=55062 to collapse zero cells
    replace dpstoff5 = 2055062 if inlist(dpstoff5, 2000050, 2000053, 2000055, 2000060, 2000062)
  Pass 0 through the data...
    smallest count = 1 in the cell      1000002
    Invoking rule 1:2:8=30108 to collapse zero cells
    replace dpstoff5 = 3030108 if inlist(dpstoff5, 3000001, 3000002, 3000008)
  Pass 0 through the data...
    smallest count = 1 in the cell      1000002
    Invoking rule 55:60=25560 to collapse zero cells
    replace dpstoff5 = 3025560 if inlist(dpstoff5, 3000055, 3000060)
  Pass 0 through the data...
    smallest count = 1 in the cell      1000002
    Invoking rule 1:2:8:11=40111 to collapse zero cells
    replace dpstoff5 = 4040111 if inlist(dpstoff5, 4000001, 4000002, 4000008, 4000011)
  Pass 0 through the data...
    smallest count = 1 in the cell      1000002
    Invoking rule 36:39:40:44=43644 to collapse zero cells
    replace dpstoff5 = 4043644 if inlist(dpstoff5, 4000036, 4000039, 4000040, 4000044)
  Pass 0 through the data...
    smallest count = 1 in the cell      1000002
    Invoking rule 49:50=24950 to collapse zero cells
    replace dpstoff5 = 4024950 if inlist(dpstoff5, 4000049, 4000050)
  Pass 0 through the data...
```

```
  smallest count = 1 in the cell      1000002
  Invoking rule 55:60=25560 to collapse zero cells
  replace dpstoff5 = 4025560 if inlist(dpstoff5, 4000055, 4000060)
Pass 0 through the data...
  smallest count = 1 in the cell      1000002
  Invoking rule 1:2:8=30108 to collapse zero cells
  replace dpstoff5 = 5030108 if inlist(dpstoff5, 5000001, 5000002, 5000008)
Pass 0 through the data...
  smallest count = 1 in the cell      1000002
  Invoking rule 36:39:40=33640 to collapse zero cells
  replace dpstoff5 = 5033640 if inlist(dpstoff5, 5000036, 5000039, 5000040)
Pass 0 through the data...
  smallest count = 1 in the cell      1000002
  Invoking rule 49:50=24950 to collapse zero cells
  replace dpstoff5 = 5024950 if inlist(dpstoff5, 5000049, 5000050)
Pass 0 through the data...
  smallest count = 1 in the cell      1000002
  Invoking rule 49:50:53:55:60=54960 to collapse zero cells
  replace dpstoff5 = 5054960 if inlist(dpstoff5, 5000049, 5000050, 5000053, 5000055, 5000060)
Pass 0 through the data...
  smallest count = 1 in the cell      1000002
Pass 14 through the data...
  smallest count = 1 in the cell      1000002
  Done collapsing! Exiting...
. * special cells for weekend
. wgtcellcollapse collapse if daypart==5 & inrange(alight_id,1,30), ///
>         variables(daypart alight_id) mincellsize(20) ///
>         strict feed(dpstoff5) saving(dpstoff5.do) append run
Pass 14 through the data...
  smallest count = 1 in the cell      5000026
  Invoking rule 24:26=22426
  replace dpstoff5 = 5022426 if inlist(dpstoff5, 5000024, 5000026)
Pass 15 through the data...
  smallest count = 1 in the cell      5030108
  Invoking rule 11:30108=40111
  replace dpstoff5 = 5040111 if inlist(dpstoff5, 5000011, 5030108)
Pass 16 through the data...
  smallest count = 3 in the cell      5022426
  Invoking rule 18:22426=31826
  replace dpstoff5 = 5031826 if inlist(dpstoff5, 5000018, 5022426)
Pass 17 through the data...
  smallest count = 6 in the cell      5040111
  Invoking rule 40111:31826=70126
  replace dpstoff5 = 5070126 if inlist(dpstoff5, 5040111, 5031826)
Pass 18 through the data...
  smallest count = 8 in the cell      5000030
  Invoking rule 30:70126=80130
  replace dpstoff5 = 5080130 if inlist(dpstoff5, 5000030, 5070126)
Pass 19 through the data...
  smallest count = 21 in the cell      5080130
  Done collapsing! Exiting...
. wgtcellcollapse collapse if daypart==5 & inrange(alight_id,36,68), ///
>         variables(daypart alight_id) mincellsize(20) ///
>         strict feed(dpstoff5) saving(dpstoff5.do) append run
Pass 19 through the data...
  smallest count = 1 in the cell      5024950
  Invoking rule 47:24950=34750
  replace dpstoff5 = 5034750 if inlist(dpstoff5, 5000047, 5024950)
Pass 20 through the data...
```

```
  smallest count = 2 in the cell      5000044
  Invoking rule 44:33640=43644
  replace dpstoff5 = 5043644 if inlist(dpstoff5, 5000044, 5033640)
Pass 21 through the data...
  smallest count = 2 in the cell      5054960
  Invoking rule 62:54960=64962
  replace dpstoff5 = 5064962 if inlist(dpstoff5, 5000062, 5054960)
Pass 22 through the data...
  smallest count = 4 in the cell      5043644
  Invoking rule 43644:34750=73650
  replace dpstoff5 = 5073650 if inlist(dpstoff5, 5043644, 5034750)
Pass 23 through the data...
  smallest count = 7 in the cell      5000068
  Invoking rule 68:64962=74968
  replace dpstoff5 = 5074968 if inlist(dpstoff5, 5000068, 5064962)
Pass 24 through the data...
  smallest count = 11 in the cell      5073650
  WARNING: could not find any rules to collapse dpstoff5 == 5073650
Pass 25 through the data...
  smallest count = 18 in the cell      5074968
  WARNING: could not find any rules to collapse dpstoff5 == 5074968
Pass 26 through the data...
  smallest count = .i in the cell      1000002
  Done collapsing! Exiting...
. * all other cells
. wgtcellcollapse collapse, variables(daypart alight_id) mincellsize(20) ///
>         strict feed(dpstoff5) saving(dpstoff5.do) append run
Pass 24 through the data...
  smallest count = 1 in the cell      1000002
  Invoking rule 2:8=20208
  replace dpstoff5 = 1020208 if inlist(dpstoff5, 1000002, 1000008)
Pass 25 through the data...
  smallest count = 1 in the cell      2000011
  Invoking rule 11:18=21118
  replace dpstoff5 = 2021118 if inlist(dpstoff5, 2000011, 2000018)
Pass 26 through the data...
  smallest count = 1 in the cell      2000024
  Invoking rule 24:26=22426
  replace dpstoff5 = 2022426 if inlist(dpstoff5, 2000024, 2000026)
Pass 27 through the data...
  smallest count = 1 in the cell      2000049
  Invoking rule 49:55062=64962
  replace dpstoff5 = 2064962 if inlist(dpstoff5, 2000049, 2055062)
Pass 28 through the data...
  smallest count = 1 in the cell      2030108
  Invoking rule 30108:21118=50118
  replace dpstoff5 = 2050118 if inlist(dpstoff5, 2030108, 2021118)
Pass 29 through the data...
  smallest count = 1 in the cell      3000039
  Invoking rule 39:40=23940
  replace dpstoff5 = 3023940 if inlist(dpstoff5, 3000039, 3000040)
Pass 30 through the data...
  smallest count = 1 in the cell      3000049
  Invoking rule 49:50=24950
  replace dpstoff5 = 3024950 if inlist(dpstoff5, 3000049, 3000050)
Pass 31 through the data...
  smallest count = 1 in the cell      3030108
  Invoking rule 11:30108=40111
  replace dpstoff5 = 3040111 if inlist(dpstoff5, 3000011, 3030108)
Pass 32 through the data...
```

```
    smallest count = 1 in the cell      4000018
    Invoking rule 18:24=21824
    replace dpstoff5 = 4021824 if inlist(dpstoff5, 4000018, 4000024)
  Pass 33 through the data...
    smallest count = 1 in the cell      4000026
    Invoking rule 26:21824=31826
    replace dpstoff5 = 4031826 if inlist(dpstoff5, 4000026, 4021824)
  Pass 34 through the data...
    smallest count = 1 in the cell      4025560
    Invoking rule 53:25560=35360
    replace dpstoff5 = 4035360 if inlist(dpstoff5, 4000053, 4025560)
  Pass 35 through the data...
    smallest count = 2 in the cell      1000050
    Invoking rule 49:50=24950
    replace dpstoff5 = 1024950 if inlist(dpstoff5, 1000049, 1000050)
  Pass 36 through the data...
    smallest count = 2 in the cell      2064962
    Invoking rule 68:64962=74968
    replace dpstoff5 = 2074968 if inlist(dpstoff5, 2000068, 2064962)
  Pass 37 through the data...
    smallest count = 2 in the cell      3000024
    Invoking rule 24:26=22426
    replace dpstoff5 = 3022426 if inlist(dpstoff5, 3000024, 3000026)
  Pass 38 through the data...
    smallest count = 2 in the cell      3000044
    Invoking rule 44:23940=33944
    replace dpstoff5 = 3033944 if inlist(dpstoff5, 3000044, 3023940)
  Pass 39 through the data...
    smallest count = 2 in the cell      3024950
    Invoking rule 53:24950=34953
    replace dpstoff5 = 3034953 if inlist(dpstoff5, 3000053, 3024950)
  Pass 40 through the data...
    smallest count = 2 in the cell      4024950
    Invoking rule 24950:35360=54960
    replace dpstoff5 = 4054960 if inlist(dpstoff5, 4024950, 4035360)
  Pass 41 through the data...
    smallest count = 2 in the cell      4043644
    Invoking rule 47:43644=53647
    replace dpstoff5 = 4053647 if inlist(dpstoff5, 4000047, 4043644)
  Pass 42 through the data...
    smallest count = 3 in the cell      1023940
    Invoking rule 36:23940=33640
    replace dpstoff5 = 1033640 if inlist(dpstoff5, 1000036, 1023940)
  Pass 43 through the data...
    smallest count = 3 in the cell      2022426
    Invoking rule 50118:22426=70126
    replace dpstoff5 = 2070126 if inlist(dpstoff5, 2050118, 2022426)
  Pass 44 through the data...
    smallest count = 3 in the cell      4000062
    Invoking rule 62:68=26268
    replace dpstoff5 = 4026268 if inlist(dpstoff5, 4000062, 4000068)
  Pass 45 through the data...
    smallest count = 4 in the cell      2053044
    Invoking rule 47:53044=63047
    replace dpstoff5 = 2063047 if inlist(dpstoff5, 2000047, 2053044)
  Pass 46 through the data...
    smallest count = 4 in the cell      3000036
    Invoking rule 36:33944=43644
    replace dpstoff5 = 3043644 if inlist(dpstoff5, 3000036, 3033944)
  Pass 47 through the data...
    smallest count = 4 in the cell      4031826
    Invoking rule 40111:31826=70126
    replace dpstoff5 = 4070126 if inlist(dpstoff5, 4040111, 4031826)
  Pass 48 through the data...
    smallest count = 5 in the cell      1000060
    Invoking rule 55:60=25560
    replace dpstoff5 = 1025560 if inlist(dpstoff5, 1000055, 1000060)
  Pass 49 through the data...
    smallest count = 5 in the cell      1024950
    Invoking rule 53:24950=34953
    replace dpstoff5 = 1034953 if inlist(dpstoff5, 1000053, 1024950)
  Pass 50 through the data...
    smallest count = 5 in the cell      2074968
    Invoking rule 63047:74968=133068
    replace dpstoff5 = 2133068 if inlist(dpstoff5, 2063047, 2074968)
  Pass 51 through the data...
    smallest count = 5 in the cell      3025560
    Invoking rule 34953:25560=54960
    replace dpstoff5 = 3054960 if inlist(dpstoff5, 3034953, 3025560)
  Pass 52 through the data...
    smallest count = 6 in the cell      2070126
    Invoking rule 70126:133068=200168
    replace dpstoff5 = 2200168 if inlist(dpstoff5, 2070126, 2133068)
  Pass 53 through the data...
    smallest count = 6 in the cell      4054960
    Invoking rule 53647:54960=103660
    replace dpstoff5 = 4103660 if inlist(dpstoff5, 4053647, 4054960)
  Pass 54 through the data...
    smallest count = 9 in the cell      4026268
    Invoking rule 103660:26268=123668
    replace dpstoff5 = 4123668 if inlist(dpstoff5, 4103660, 4026268)
  Pass 55 through the data...
    smallest count = 10 in the cell      3022426
    Invoking rule 18:22426=31826
    replace dpstoff5 = 3031826 if inlist(dpstoff5, 3000018, 3022426)
  Pass 56 through the data...
    smallest count = 10 in the cell      3043644
    Invoking rule 30:43644=53044
    replace dpstoff5 = 3053044 if inlist(dpstoff5, 3000030, 3043644)
  Pass 57 through the data...
    smallest count = 10 in the cell      4070126
    Invoking rule 30:70126=80130
    replace dpstoff5 = 4080130 if inlist(dpstoff5, 4000030, 4070126)
  Pass 58 through the data...
    smallest count = 11 in the cell      1025560
    Invoking rule 34953:25560=54960
    replace dpstoff5 = 1054960 if inlist(dpstoff5, 1034953, 1025560)
  Pass 59 through the data...
    smallest count = 11 in the cell      5073650
    Invoking rule 80130:73650=150150
    replace dpstoff5 = 5150150 if inlist(dpstoff5, 5080130, 5073650)
  Pass 60 through the data...
    smallest count = 12 in the cell      1020208
    Invoking rule 11:20208=30211
    replace dpstoff5 = 1030211 if inlist(dpstoff5, 1000011, 1020208)
  Pass 61 through the data...
    smallest count = 12 in the cell      1033640
    Invoking rule 44:33640=43644
    replace dpstoff5 = 1043644 if inlist(dpstoff5, 1000044, 1033640)
  Pass 62 through the data...
```

```
  smallest count = 15 in the cell      1000024
  Invoking rule 24:26=22426
  replace dpstoff5 = 1022426 if inlist(dpstoff5, 1000024, 1000026)
Pass 63 through the data...
  smallest count = 15 in the cell      3040111
  Invoking rule 40111:31826=70126
  replace dpstoff5 = 3070126 if inlist(dpstoff5, 3040111, 3031826)
Pass 64 through the data...
  smallest count = 15 in the cell      3054960
  Invoking rule 62:54960=64962
  replace dpstoff5 = 3064962 if inlist(dpstoff5, 3000062, 3054960)
Pass 65 through the data...
  smallest count = 18 in the cell      5074968
  Invoking rule 69:74968=84969
  replace dpstoff5 = 5084969 if inlist(dpstoff5, 5000069, 5074968)
Pass 66 through the data...
  smallest count = 21 in the cell      2200168
  Done collapsing! Exiting...
. assert "`r(failed)´" == ""
```

This can now be applied to producing control totals, and running raking:

```
. use trip_population, clear
. run dpston5.do
. total num_pass , over(dpston5)
Total estimation                  Number of obs   =        719
      1000001: dpston5 = 1000001
      1000002: dpston5 = 1000002
      1000008: dpston5 = 1000008
      1000011: dpston5 = 1000011
      1000018: dpston5 = 1000018
      1000024: dpston5 = 1000024
      1000026: dpston5 = 1000026
      1000030: dpston5 = 1000030
      1000036: dpston5 = 1000036
      1000044: dpston5 = 1000044
      1000047: dpston5 = 1000047
      1000062: dpston5 = 1000062
      1000068: dpston5 = 1000068
      1023940: dpston5 = 1023940
      1025560: dpston5 = 1025560
      1034953: dpston5 = 1034953
      2000047: dpston5 = 2000047
      2031826: dpston5 = 2031826
      2040111: dpston5 = 2040111
      2053044: dpston5 = 2053044
      2074968: dpston5 = 2074968
      3000008: dpston5 = 3000008
      3000011: dpston5 = 3000011
      3000018: dpston5 = 3000018
      3000024: dpston5 = 3000024
      3000026: dpston5 = 3000026
      3000030: dpston5 = 3000030
      3000047: dpston5 = 3000047
      3000068: dpston5 = 3000068
      3020102: dpston5 = 3020102
      3034953: dpston5 = 3034953
      3035562: dpston5 = 3035562
```

```
     3043644: dpston5 = 3043644
     4030108: dpston5 = 4030108
     4041126: dpston5 = 4041126
     4074968: dpston5 = 4074968
     4083050: dpston5 = 4083050
     5000011: dpston5 = 5000011
     5026268: dpston5 = 5026268
     5030108: dpston5 = 5030108
     5051836: dpston5 = 5051836
     5093960: dpston5 = 5093960

--------------------------------------------------------------
        Over |      Total   Std. Err.     [95% Conf. Interval]
-------------+------------------------------------------------
num_pass     |
     1000001 |       1423   967.7508     -476.9595    3322.959
     1000002 |       7198    4895.91     -2414.011    16810.01
     1000008 |      19254   13675.81     -7595.347    46103.35
     1000011 |      12626   9682.022     -6382.456    31634.46
     1000018 |       2470   1943.224     -1345.081    6285.081
     1000024 |        634   509.3549     -366.0031    1634.003
     1000026 |       2208   1774.996     -1276.802    5692.802
     1000030 |       4319   3665.427     -2877.235    11515.24
     1000036 |       1221   1046.817     -834.1873    3276.187
     1000044 |       1021    881.426     -709.4802     2751.48
     1000047 |       3300   2970.321     -2531.552    9131.552
     1000062 |       3402       3176     -2833.357    9637.357
     1000068 |       5085          .             .           .
     1023940 |        491    348.709     -193.6112    1175.611
     1025560 |        601     350.65     -87.42178    1289.422
     1034953 |       1286   774.5361     -234.6262    2806.626
     2000047 |        776   701.0001     -600.2549    2152.255
     2031826 |        426   223.0489     -11.90601     863.906
     2040111 |       1385   714.1628     -17.09692    2787.097
     2053044 |        527   364.2392     -188.1011    1242.101
     2074968 |        443   221.0935      8.933009     877.067
     3000008 |       3739   2665.175     -1493.467    8971.467
     3000011 |       3476   2669.777     -1765.503    8717.503
     3000018 |       1263    997.019     -694.4209    3220.421
     3000024 |       1296   1032.175     -730.4418    3322.442
     3000026 |        439   357.3421     -262.5603     1140.56
     3000030 |       3740   3175.677     -2494.723    9974.723
     3000047 |        984   888.5095     -760.3871    2728.387
     3000068 |        744          .             .           .
     3020102 |        992   553.0017     -93.69354    2077.694
     3034953 |        893   588.1798     -261.7577    2047.758
     3035562 |       1187   894.6009     -569.3461    2943.346
     3043644 |        713   398.6235     -69.60702    1495.607
     4030108 |       1154   529.0201      115.3888    2192.611
     4041126 |       1135   530.8674      92.76204    2177.238
     4074968 |        659   307.4955      55.30219    1262.698
     4083050 |        741   395.9312     -36.32126    1518.321
     5000011 |       1270    834.301      -367.961    2907.961
     5026268 |        557   364.4324     -158.4805    1272.481
     5030108 |        610   263.2061      93.25444    1126.746
     5051836 |        622   215.5712      198.7749    1045.225
     5093960 |        473   261.8954     -41.17225    987.1723
--------------------------------------------------------------

. matrix dpston5 = e(b)

. matrix coleq dpston5 = _one
. matrix rownames dpston5 = dpston5
. run dpstoff5.do
. total num_pass , over(dpstoff5)
Total estimation                   Number of obs   =        719
      1000018: dpstoff5 = 1000018
      1000030: dpstoff5 = 1000030
      1000047: dpstoff5 = 1000047
      1000062: dpstoff5 = 1000062
      1000068: dpstoff5 = 1000068
      1000069: dpstoff5 = 1000069
      1022426: dpstoff5 = 1022426
      1030211: dpstoff5 = 1030211
      1043644: dpstoff5 = 1043644
      1054960: dpstoff5 = 1054960
      2000069: dpstoff5 = 2000069
      2200168: dpstoff5 = 2200168
      3000047: dpstoff5 = 3000047
      3000068: dpstoff5 = 3000068
      3000069: dpstoff5 = 3000069
      3053044: dpstoff5 = 3053044
      3064962: dpstoff5 = 3064962
      3070126: dpstoff5 = 3070126
      4000069: dpstoff5 = 4000069
      4080130: dpstoff5 = 4080130
      4123668: dpstoff5 = 4123668
      5084969: dpstoff5 = 5084969
      5150150: dpstoff5 = 5150150
--------------------------------------------------------------
        Over |      Total   Std. Err.     [95% Conf. Interval]
-------------+------------------------------------------------
num_pass     |
     1000018 |        929   360.7303      220.7878    1637.212
     1000030 |       2189   868.0319      484.8161    3893.184
     1000047 |       1746   630.7528      507.6598     2984.34
     1000062 |       1134   382.7765       382.505    1885.495
     1000068 |       1372   426.3969      534.8662    2209.134
     1000069 |      53193   15995.88      21788.72    84597.28
     1022426 |        980    273.542      442.9623    1517.038
     1030211 |       2986   1614.166     -183.0484    6155.048
     1043644 |        737   159.5597      423.7407    1050.259
     1054960 |       1273   259.8915      762.7619    1783.238
     2000069 |       3038   938.8099      1194.859    4881.141
     2200168 |        519   71.80393      378.0292    659.9708
     3000047 |        556   187.4945      187.8971    924.1029
     3000068 |        444   126.0503      196.5289    691.4711
     3000069 |      16007   4295.998      7572.781    24441.22
     3053044 |        787    249.935      296.3092    1277.691
     3064962 |        759   141.1029      481.9765    1036.023
     3070126 |        913   335.9457      253.4468    1572.553
     4000069 |       2733   728.6906      1302.381    4163.619
     4080130 |        480   132.6806      219.5117    740.4883
     4123668 |        476   72.94794      332.7832    619.2168
     5084969 |       2945   999.9897      981.7468    4908.253
     5150150 |        587   137.3569      317.3308    856.6692
--------------------------------------------------------------
. matrix dpstoff5 = e(b)
. matrix coleq dpstoff5 = _one
. matrix rownames dpstoff5 = dpstoff5
. use trip_sample, clear
. run dpston5
. run dpstoff5
. gen byte _one = 1
. ipfraking [pw=_one], ctotal(dpston5 dpstoff5) gen(raked_weight5)
 Iteration 1, max rel difference of raked weights = 37.856256
 Iteration 2, max rel difference of raked weights = .0250943
 Iteration 3, max rel difference of raked weights = .00252004
 Iteration 4, max rel difference of raked weights = .00030004
 Iteration 5, max rel difference of raked weights = .00003571
 Iteration 6, max rel difference of raked weights = 4.250e-06
 Iteration 7, max rel difference of raked weights = 5.058e-07
The worst relative discrepancy of  5.6e-08 is observed for dpstoff5 == 5150150
Target value =       587; achieved value =       587
   Summary of the weight changes
              |     Mean    Std. dev.    Min        Max       CV
--------------+-------------------------------------------------
Orig weights  |        1         0         1          1       0
Raked weights |   26.487       5.74    14.593     38.634   .2167
Adjust factor |  26.4869             14.5933    38.6339
. whatsdeff raked_weight5
    Group     |    Min     |    Mean    |    Max     |    CV    |   DEFF   |   N    |  N eff
--------------+------------+------------+------------+----------+----------+--------+--------
      Overall |     14.59  |     26.49  |     38.63  |  0.2167  |  1.0470  |  3654  | 3490.13
```

2.11 Linear calibrated weights

Using the existing example, let me demonstrate the linear calibration option of `ipfraking`.

```
. cap drop raked_weight5*
. set rmsg on
r; t=0.00 12:12:08
. ipfraking [pw=_one], ctotal(dpston5 dpstoff5) nograph gen(raked_weight5)
 Iteration 1, max rel difference of raked weights = 37.856256
 Iteration 2, max rel difference of raked weights = .0250943
 Iteration 3, max rel difference of raked weights = .00252004
 Iteration 4, max rel difference of raked weights = .00030004
 Iteration 5, max rel difference of raked weights = .00003571
 Iteration 6, max rel difference of raked weights = 4.250e-06
 Iteration 7, max rel difference of raked weights = 5.058e-07
The worst relative discrepancy of  5.6e-08 is observed for dpstoff5 == 5150150
Target value =       587; achieved value =       587
   Summary of the weight changes
              |     Mean    Std. dev.    Min        Max       CV
--------------+-------------------------------------------------
Orig weights  |        1         0         1          1       0
Raked weights |   26.487       5.74    14.593     38.634   .2167
Adjust factor |  26.4869             14.5933    38.6339
r; t=1.100 12:12:10
. ipfraking [pw=_one], ctotal(dpston5 dpstoff5) nograph gen(raked_weight5l) linear
Linear calibration
```

```
The worst relative discrepancy of  2.0e-14 is observed for dpstoff5 == 5150150
Target value =        587; achieved value =        587
   Summary of the weight changes
              |    Mean    Std. dev.    Min        Max       CV
--------------+-------------------------------------------------
Orig weights  |       1         0         1          1        0
Raked weights |  26.487     5.7387    12.875     38.204    .2167
Adjust factor | 26.4869               12.8752    38.2040
r; t=0.78 12:12:11
. set rmsg off
. label variable raked_weight5l "Linear calibrated weights"
. compare raked_weight5 raked_weight5l
                                            ---------- difference ----------
                             count       minimum      average      maximum
------------------------------------------------------------------------
raked_w~5<raked_~5l           1871     -1.813144    -.0408154   -5.84e-10
raked_w~5>raked_~5l           1783      2.75e-08     .0428298    2.405758
                        ----------
jointly defined               3654     -1.813144     1.20e-10    2.405758
                        ----------
total                         3654
```

2.12 Utility programs

The original package `ipfraking` provided two additional utility programs, `mat2do` and `xls2row`. An additional utility program was added to compute the design effects and margins of error, common tasks associated with describing survey weights. Specifically, the Transparency Initiative of the American Association for Public Opinion Research (AAPOR 2014) requires that

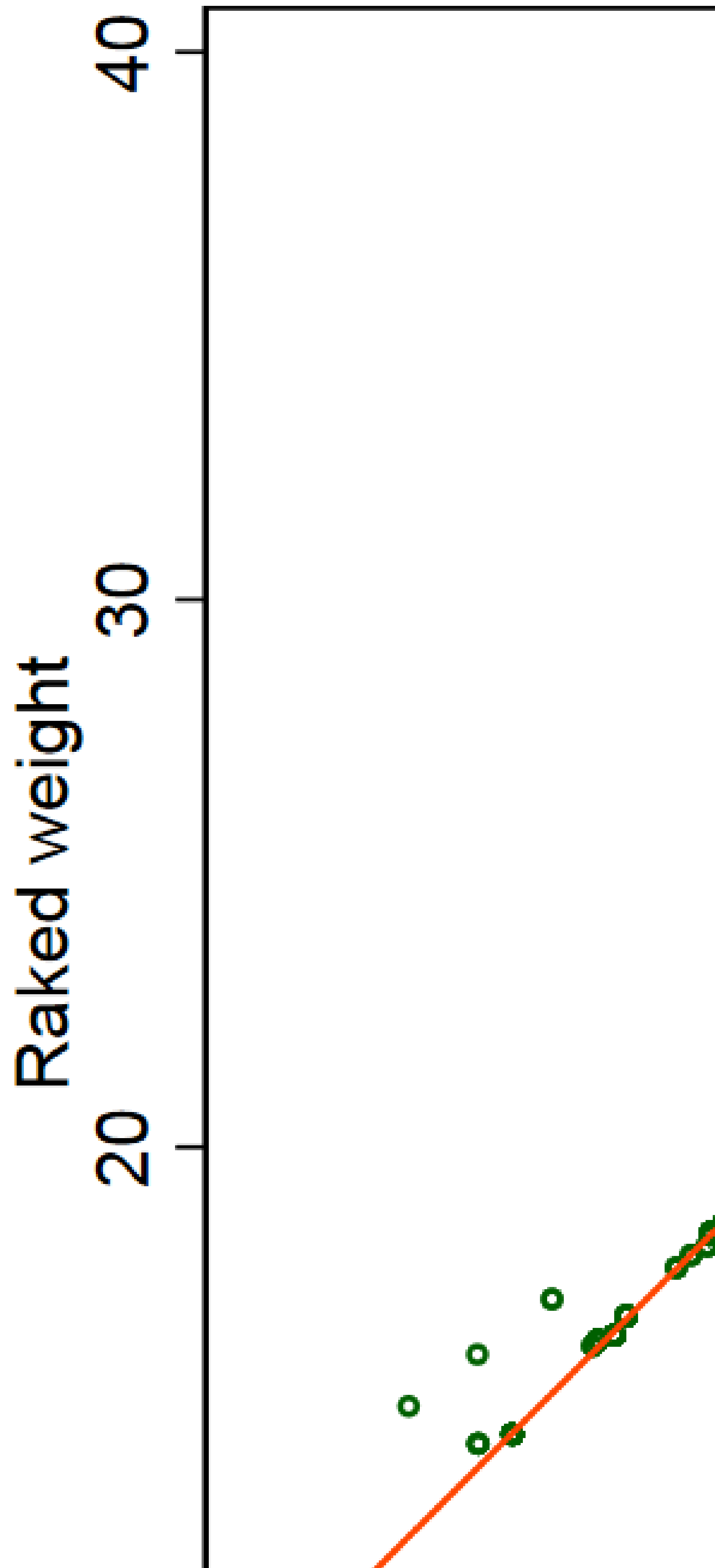
For probability samples, the estimates of sampling error will be reported, and the discussion will state whether or not the reported margins of sampling error or statistical analyses have been adjusted for the design effect due to weighting, clustering, or other factors.

`whatsdeff` *weight_variable* [*if*] [*in*] , [`by(`*varlist*`)`]

The utility program `whatsdeff` calculates the apparent design effect due to unequal weighting, $\mathrm{DEFF}_{\mathrm{UWE}} = 1 + CV_w^2$ = `1 + r(Var)/(r(mean))^2` from `summarize` *weight_variable*. Additionally, it reports the effective sample size, $n/\mathrm{DEFF}_{\mathrm{UWE}}$, and also returns the margins of error for the sample proportions that estimate the population proportions of 10% and 50%.

```
. webuse nhanes2, clear
. whatsdeff finalwgt
    Group    |    Min     |    Mean    |    Max     |    CV    |   DEFF   |   N    |  N eff
-------------------------------------------------------------------------------------------
      Overall |   2000.00  |  11318.47  |  79634.00  |  0.6453  |  1.4164  | 10351  | 7307.97
```

```
. return list
scalars:
                  r(N) =  10351
              r(MOE10) =  .0068792766212984
              r(MOE50) =  .0114654610354974
       r(Neff_Overall) =  7307.97435325364
       r(DEFF_Overall) =  1.416397964696134
. whatsdeff finalwgt, by(sex)
    Group     |     Min     |    Mean     |     Max     |    CV     |   DEFF    |    N    |  N eff
--------------+-------------+-------------+-------------+-----------+-----------+---------+--------
sex           |             |             |             |           |           |         |
         Male |   2000.00   |   11426.14  |   79634.00  |   0.6578  |   1.4326  |   4915  | 3430.94
       Female |   2130.00   |   11221.12  |   61534.00  |   0.6333  |   1.4010  |   5436  | 3880.01
--------------+-------------+-------------+-------------+-----------+-----------+---------+--------
      Overall |   2000.00   |   11318.47  |   79634.00  |   0.6453  |   1.4164  |  10351  | 7307.97
. return list
scalars:
                  r(N) =  10351
              r(MOE10) =  .0068792766212984
              r(MOE50) =  .0114654610354974
       r(Neff_Overall) =  7307.97435325364
       r(DEFF_Overall) =  1.416397964696134
        r(Neff_Female) =  3880.00710397866
        r(DEFF_Female) =  1.40102836266093
          r(Neff_Male) =  3430.938195872213
          r(DEFF_Male) =  1.432552765279559
```


Raked weight
40
30
20

3 Examples

3.1 Basic syntax and input requirements

In this very simple example, I shall demonstrate the basic mechanics of `ipfraking`, its input requirements and output. These examples are intended to only demonstrate the syntax and the output of `ipfraking`, and may or may not provide substantively meaningful results.

▷ Example 1

We shall work with the standard example of `svy` data, an excerpt from the NHANES II data set available from Stata Corp. website. We shall introduce some small changes to the data so that `ipfraking` will have some work to do.

```
. webuse nhanes2, clear
. generate byte _one = 1
.
. svy : total _one , over( sex, nolabel )
(running total on estimation sample)
Survey: Total estimation
Number of strata =      31       Number of obs    =      10351
Number of PSUs   =      62       Population size  =  117157513
                                 Design df        =         31
            1: sex = 1
            2: sex = 2
```

Over	Total	Linearized Std. Err.	[95% Conf.	Interval]
_one				
1	5.62e+07	1377465	5.34e+07	5.90e+07
2	6.10e+07	1396159	5.82e+07	6.38e+07

```
. matrix NHANES2_sex = e(b)
. matrix rownames NHANES2_sex = sex
.
. svy : total _one , over( race, nolabel )
(running total on estimation sample)
Survey: Total estimation
Number of strata =      31       Number of obs    =      10351
Number of PSUs   =      62       Population size  =  117157513
                                 Design df        =         31
            1: race = 1
            2: race = 2
            3: race = 3
```

(*Continued on next page*)

```
------------------------------------------------------------
             |             Linearized
        Over |      Total   Std. Err.     [95% Conf. Interval]
-------------+----------------------------------------------
_one         |
           1 |   1.03e+08    2912042      9.71e+07    1.09e+08
           2 |   1.12e+07    1458814       8213964    1.42e+07
           3 |    2968728    1252160      414930.1     5522526
------------------------------------------------------------
. matrix NHANES2_race = e(b)
. matrix rownames NHANES2_race = race
.
. matrix NHANES2_sex[1,1] = NHANES2_sex[1,1]*1.25
. matrix NHANES2_race[1,1] = NHANES2_race[1,1]*1.4
.
```

Let us now look at the matrices that will serve as an input to the raking procedure.

```
. matrix list NHANES2_sex, f(%12.0g)
NHANES2_sex[1,2]
         _one:     _one:
            1         2
sex  70199350  60998033
. matrix list NHANES2_race, f(%12.0g)
NHANES2_race[1,3]
             _one:        _one:        _one:
                1            2            3
race  144199368.6     11189236      2968728
```

These input matrices are organized as follows. Input matrices always have a single row, just as estimation results `e(b)` do. The column names follow the naming conventions of `e(b)`, namely, the name of the variable for which the total is being computed (here, `_one`) and the numeric categories of the variable that was used in the `over` option (here, `sex`, with values 1 for males and 2 for females; and `race`, with values 1 for whites, 2 for blacks, and 3 for other). These values must be in an increasing order. Since that variable is not stored in the e(b) per se, it needs to be added to this matrix, which is done in the form of the row name. The entries of the matrix are the totals that the weights in the categories of the control variables need to sum up to. In this example, they are scaled to be the population totals. Alternatively, these can be made to sum up to the sample size, as is done sometimes in public opinion research, or to 1, which is what `proportion` estimation command would produce.

The input requirements in terms of control totals are thus made as simple as possible. If a higher quality survey is available, all the survey statistician needs to do is to obtain the totals for the categories of the control variables using `svy: total` ···`, over(` ···`, nolabel )` and save the name of that variable along with the matrix. Note that the `total` is computed with `over( ..., nolabel)` suboption to suppress the otherwise informative labeling of the categories; `ipfraking` expects the numeric values of the categories as column names (see [P] **matrix rownames**). The name of the matrix itself is immaterial, but it is a good programming practice to have informative names

(McConnell 2004). Thus the names of the matrices in the examples generally follow the convention *data_source_variable*.

We are now ready to run `ipfraking` and see what it produces.

```
. ipfraking [pw=finalwgt], ctotal( NHANES2_sex NHANES2_race ) gen( rakedwgt1 )
Warning: the totals of the control matrices are different:
   Target 1 (NHANES2_sex) total             =           131197383
   Target 2 (NHANES2_race) total            =         158357332.6
 Iteration 1, max rel difference of raked weights = .56227988
 Iteration 2, max rel difference of raked weights = .00073288
 Iteration 3, max rel difference of raked weights = 2.356e-07
Warning: the controls NHANES2_sex did not match
   Summary of the weight changes
               |    Mean    Std. dev.     Min        Max       CV
---------------+---------------------------------------------------
Orig weights   |    11318       7304      2000      79634   .6453
Raked weights  |    15299      10274      1914      90831   .6716
Adjust factor  |   1.3490              0.8846     1.5614
```

In this simple case with just two control variables and the control totals that are not very different from the existing sample totals, the procedure converged very quickly in three iterations. A diagnostic message was produced upfront by `ipfraking` informing about apparent differences in total population counts as obtained from the different control total matrices. As a result, the control totals for the variable that was adjusted first (`sex`) could not match the required control totals even after the weights converged in the sense of differing little between iterations. Both of these warnings are only produced when problems are encountered.

The summary table is always produced, and shows some relevant characteristics of the original weights w_{1j}, the raked weights w_{3j}, and the raking ratios w_{3j}/w_{1j}. As expected, the coefficient of variation went up from 0.645 to 0.672.

The graphic output produced by `ipfraking` is shown on Figure 1. Generally, we would want to inspect these graphs to see if there any unexpected patterns, such as highly outlying values, gaps in the distribution (here, there are only six distinct values of the adjustment factor corresponding to the 2×3 combinations of the control variables) or concentration near the limits of the weight range (as is typical for trimmed weights, see below in section 3.3). Also, these graphs may inform later trimming decisions: the trimming limits can be chosen to conform to the breaks in the distributions of the untrimmed raked weights.

Figure 2: Histograms of the raked weights and calibration ratios, Example 1.

◁

Table 1: Control totals for the 2011 US population.

Group	Population
ACS 2011 1-year estimates, Table S0101	
Male, total	153,267,860
Ages 20–39	27.4%
Ages 40–59	27.5%
Ages 60+	17.3%
Female, total	158,324,057
Ages 20–39	26.0%
Ages 40–59	27.6%
Ages 60+	20.7%
US Census Bureau 2011 projections, Table NST-EST2011-01	
Northeast	55,521,598
Midwest	67,158,835
South	116,046,736
West	72,864,748
US Census Bureau 2011 projections, Table NC-EST2011-03	
White	243,470,497
Black	40,750,746
Other	27,370,674
Total	311,591,917

3.2 Preparing control matrices from scratch

In many situations, the control totals will be obtained from outside of Stata, and need to be prepared to work with `ipfraking`.

▷ Example 2

Suppose I wanted to calibrate the NHANES II data set to the latest control totals available from the US Census Bureau website. Using the tables S0101 from the 2011 American Community Survey 1-year estimates and NST-EST2011 from the US Census Bureau population projections, the latest available at the time of writing this paper, the figures displayed in Table 1 can be obtained.

Thus, we have information in the two-way age by sex table, as well as two additional margins. We shall need an additional sex-by-age group variable, and we shall try to make its values somewhat informative (e.g., the value `12` of the variable `sex_age` means the first group of sex and the second group of age):

```
. generate byte age_grp = 1 + (age>=40) + (age>=60) if !mi(age)
. generate sex_age = sex*10 + age_grp
```

With that, the matrices will have to be defined explicitly, and their labels need to be hand-coded, too (see [P] **matrix rownames**). Note that the US Census Bureau 2011 projections relate to the total population, while the target population of the study is the population age 20+. Assuming that the age structure is the same across regions and races, the control totals for region and race need to be rescaled to the adult population to avoid the warning messages. (More accurate figures can be obtained from ACS microdata which can be downloaded from the U.S. Census Bureau website.)

```
. matrix ACS2011_sex_age = ( ///
>     153267860*0.274, 153267860*0.275, 153267860*0.173, /// males
>     158324057*0.260, 158324057*0.276, 158324057*0.207  /// females
> )
. matrix colnames ACS2011_sex_age = 11 12 13 21 22 23
. matrix coleq    ACS2011_sex_age = _one
. matrix rownames ACS2011_sex_age = sex_age
. scalar ACS2011_total_pop = 311591917
. matrix ACS2011_adult_pop = ACS2011_sex_age * J(colsof(ACS2011_sex_age),1,1)
. matrix Census2011_region = ///
>     (55521598, 67158835, 116046736, 72864748 )
. matrix Census2011_region = Census2011_region * ACS2011_adult_pop / ACS2011_to
> tal_pop
. matrix colnames Census2011_region = 1 2 3 4
. matrix coleq    Census2011_region = _one
. matrix rownames Census2011_region = region
. matrix Census2011_race = ///
>     (243470497, 40750746, 27370674 )
```

(*Continued on next page*)

```
. matrix Census2011_race = Census2011_race * ACS2011_adult_pop / ACS2011_total_
> pop
. matrix colnames Census2011_race = 1 2 3
. matrix coleq    Census2011_race = _one
. matrix rownames Census2011_race = race
```

Let us check the matrix entries and labels once again before producing the weights. Note that the values of the control variable categories are given in an increasing order.

```
. matrix list ACS2011_sex_age, f(%10.0g)
ACS2011_sex_age[1,6]
             _one:     _one:     _one:     _one:     _one:     _one:
               11        12        13        21        22        23
sex_age  41995394  42148662  26515340  41164255  43697440  32773080
. matrix list Census2011_region, f(%10.0g)
Census2011_region[1,4]
            _one:     _one:     _one:     _one:
              1         2         3         4
region  40679030  49205289  85024007  53385843
. matrix list Census2011_race, f(%11.0g)
Census2011_race[1,3]
            _one:       _one:       _one:
              1           2           3
race   178383622  29856864.7  20053682.2
```

As the labels appear to be in place, let us run `ipfraking`:

```
. ipfraking [pw=finalwgt], gen( rakedwgt2 ) ///
>     ctotal( ACS2011_sex_age Census2011_region Census2011_race )
 Iteration 1, max rel difference of raked weights = 14.95826
 Iteration 2, max rel difference of raked weights = .19495004
 Iteration 3, max rel difference of raked weights = .02204455
 Iteration 4, max rel difference of raked weights = .00315355
 Iteration 5, max rel difference of raked weights = .00043857
 Iteration 6, max rel difference of raked weights = .00006061
 Iteration 7, max rel difference of raked weights = 8.365e-06
 Iteration 8, max rel difference of raked weights = 1.154e-06
 Iteration 9, max rel difference of raked weights = 1.593e-07
   Summary of the weight changes
              |   Mean    Std. dev.    Min        Max       CV
--------------+------------------------------------------------
Orig weights  |   11318      7304      2000      79634   .6453
Raked weights |   22055     19227      4050     338675   .8717
Adjust factor |  2.1464              0.9264    18.3694
```

The diagnostic plots for these weights are given in Figure 2. They do appear to have some outlying cases (which are not very clearly seen on these plots as they are single count observations with outlying weights), and we shall address them in the next section with trimming.

◁

Figure 3: Histograms of the raked weights and calibration ratios, Example 2.

3.3 Trimming options

As discussed in Section ?? above, if variability of the weights becomes excessive, the weights can be trimmed by restricting the extremes. Using `ipfraking` options, upper and/or lower limits can be defined for either the absolute values of the weights or the relative changes from the base weights. The frequency of the trimming operations can also be controlled. Trimming can be applied once to the final data (`trimfreq(once)`) at step ?? of Algorithm 2. Alternatively, trimming can be applied after every full cycle over variables at step ?? of Algorithm 2. Finally, trimming can be applied after each sub-iteration at step ?? of the algorithm.

▷ Example 3

Inspecting the histograms on Figure 2, it appears reasonable to restrict the upper tail of the raked weights. A more detailed investigation of the histogram reveals a somewhat greater concentration of the raked weights around the value of 160,000, and sparse bars beyond 200,000. This latter number will be used as the top cut-off point for trimming, and is provided as an input to `ipfraking` via option `trimhiabs`. Also, I specified the absolute lower bound of 2,000, which is the minimum of the original weights, but, as the output in the previous example suggested, the calibrated weights tend to run above 4,000, so specifying the lower limit as `trimloabs(2000)` may not really affect the calibration procedure.

```
. ipfraking [pw=finalwgt], gen( rakedwgt3 ) ///
>     ctotal( ACS2011_sex_age Census2011_region Census2011_race ) ///
>     trimhiabs( 200000 ) trimloabs( 2000 )
 Iteration 1, max rel difference of raked weights = 14.95826
 Iteration 2, max rel difference of raked weights = .21474256
 Iteration 3, max rel difference of raked weights = .02754514
 Iteration 4, max rel difference of raked weights = .00511347
 Iteration 5, max rel difference of raked weights = .00095888
 Iteration 6, max rel difference of raked weights = .00018036
 Iteration 7, max rel difference of raked weights = .00003391
 Iteration 8, max rel difference of raked weights = 6.377e-06
 Iteration 9, max rel difference of raked weights = 1.199e-06
 Iteration 10, max rel difference of raked weights = 2.254e-07
   Summary of the weight changes
              |     Mean    Std. dev.    Min        Max       CV
--------------------------------------------------------------------
Orig weights  |    11318       7304     2000       79634   .6453
Raked weights |    22055      18908     4033      200000   .8573
Adjust factor |   2.1486              0.9220     18.9828
```

The resulting coefficient of variation of weights, 0.857, is slightly better than that with unrestricted range of weights, 0.872. The summary also shows that the weights were capped at 200,000, as requested.

Setting the absolute limits on the range of the raked weights is often very subjective. A somewhat better plan might be to set limits in terms of the range of the adjustment factors, as shown in the next example. The relative change in the weights can be bounded with `trimlorel()` and `trimhirel()` options. I also demonstrate here how to use the results of `summarize` to feed into `ipfraking`. While ensuring that accurate numbers are being carried over in the context of the code, the approach is fragile for interactive work: simply running the single line with the sole `ipfraking` command that refers to the `r()` return values may break down if `summarize` was not the immediately preceding command.

```
. sum finalwgt
    Variable |       Obs        Mean    Std. Dev.       Min        Max
-------------+--------------------------------------------------------
    finalwgt |     10351    11318.47     7304.04       2000      79634
. ipfraking [pw=finalwgt], gen( rakedwgt4 ) ///
>     ctotal( ACS2011_sex_age Census2011_region Census2011_race ) ///
>     trimhiabs(`=2.5*r(max)´) trimloabs(`=r(min)´) trimhirel(6)
```

(*Continued on next page*)

```
 Iteration 1, max rel difference of raked weights = 5
 Iteration 2, max rel difference of raked weights = .25592859
 Iteration 3, max rel difference of raked weights = .0626759
 Iteration 4, max rel difference of raked weights = .0158786
 Iteration 5, max rel difference of raked weights = .00299304
 Iteration 6, max rel difference of raked weights = .00070812
 Iteration 7, max rel difference of raked weights = .00016401
 Iteration 8, max rel difference of raked weights = .00003734
 Iteration 9, max rel difference of raked weights = 8.434e-06
 Iteration 10, max rel difference of raked weights = 1.898e-06
 Iteration 11, max rel difference of raked weights = 4.265e-07
Warning: the controls ACS2011_sex_age did not match
Warning: the controls Census2011_region did not match
Warning: the controls Census2011_race did not match
   Summary of the weight changes
              |   Mean    Std. dev.    Min        Max       CV
--------------+------------------------------------------------
Orig weights  |   11318      7304      2000      79634   .6453
Raked weights |   21830     18115      4113     199085   .8298
Adjust factor |  2.1323              0.8973     6.0000
```

◁

Setting the trimming options too aggressively may lead to adverse consequences. First, it may bias the estimates, as discussed in Section **??**. Second, as this example demonstrates, it can impede (statistical) convergence: the output contains multiple warnings about targets not being achieved within desired accuracy, while no problems were encountered without trimming.

3.4 Tracking convergence

Let us now look in more detail into the issue of trimming frequency, and demonstrate another diagnostic plot that can be produced by `ipfraking`.

▷ Example 4

We return to the first set of options of Example 3, and re-run the raking procedure.

```
. capture drop rakedwgt3
. ipfraking [pw=finalwgt], gen( rakedwgt3 ) ///
>     ctotal( ACS2011_sex_age Census2011_region Census2011_race ) ///
>     trimhiabs(200000) trimloabs(2000) trimfreq(sometimes) trace
 Iteration 1, max rel difference of raked weights = 14.95826
  (output omitted)
 Iteration 10, max rel difference of raked weights = 2.254e-07
   Summary of the weight changes
              |   Mean    Std. dev.    Min        Max       CV
--------------+------------------------------------------------
Orig weights  |   11318      7304      2000      79634   .6453
Raked weights |   22055     18908      4033     200000   .8573
Adjust factor |  2.1486              0.9220    18.9828
```

The option `trace` requests that trace plots be added to the diagnostic plots, as shown on Figure 3. The trace plots are presented on the absolute scale and on the log scale. The exponentially declining discrepancy appears to be a general phenomenon. In other words, after the first few iterations, discrepancy between the currently weighted totals to the control totals roughly follows the rate of const $\times \alpha^{\mathrm{k}}$ for some $\alpha < 1$, where k is the (outer cycle) iteration number. When convergence is very slow or the sample size is very large, this rule may be helpful in determining the number of iterations necessary to achieve the required accuracy, and hence the expected computing time. Zero cross-cells and collinearity between the control variables may make the convergence factor α close to 1 thus hampering convergence. This happens when the control variables have very similar meaning, such as age and grade of children: it is impossible to have children of age 8 in grade 10. Also, sets of interactions of categorical variables, such as interactions of age group and education along with age group and race, are guaranteed to produce zero cells in the cross-tabulation: it is impossible to have any observations in the cells defined say by (age under 40 interacted with higher education) on one margin against (age above 60 interacted with white race) on the other.

Figure 4: Diagnostic plots for Example 4.

While `trimfreq(sometimes)` is the default in presence of other trimming options, the behavior can be changed with explicit specification of trimming frequency. Note that slightly different weights will be produced that way.

```
. ipfraking [pw=finalwgt], gen( rakedwgt5 ) ///
>     ctotal( ACS2011_sex_age Census2011_region Census2011_race ) ///
>     trimhiabs(200000) trimloabs(2000) trimfreq(often) trace
 Iteration 1, max rel difference of raked weights = 14.95826
 Iteration 2, max rel difference of raked weights = .21613885
 Iteration 3, max rel difference of raked weights = .02673316
 Iteration 4, max rel difference of raked weights = .00480164
 Iteration 5, max rel difference of raked weights = .00086195
 Iteration 6, max rel difference of raked weights = .00015444
 Iteration 7, max rel difference of raked weights = .00002762
 Iteration 8, max rel difference of raked weights = 4.940e-06
 Iteration 9, max rel difference of raked weights = 8.832e-07
   Summary of the weight changes
              |    Mean    Std. dev.    Min        Max       CV
--------------+-------------------------------------------------
Orig weights  |    11318       7304      2000       79634   .6453
Raked weights |    22055      18905      4033      200000   .8572
Adjust factor |   2.1487               0.9220     18.9844
. compare rakedwgt3 rakedwgt5
                                              ---------- difference ----------
                                  count       minimum      average     maximum
------------------------------------------------------------------------------
rakedwgt3<rakedwgt5                3638     -15.27963    -1.226753   -.0128687
rakedwgt3=rakedwgt5                   4
rakedwgt3>rakedwgt5                6709      .0011514     .6652557    2471.578
                             ----------
jointly defined                   10351     -15.27963     .0000264    2471.578
```

```
                              ─────────
     total                        10351
```

In this example, trimming the weights after adjusting each of the margins led to fewer iterations. This may or may not translate to lower overall computing times as more computing is performed within each iteration.

◁

3.5 Metadata

The results of raking operations can be stored with the newly created weight variables for later review and reproduction of the results. Let us reproduce the example in the previous section adding all the metadata available:

▷ **Example 5**

```
. capture drop rakedwgt3
. ipfraking [pw=finalwgt], gen( rakedwgt3 ) ///
>     ctotal( ACS2011_sex_age Census2011_region Census2011_race ) ///
>     trimhiabs(200000) trimloabs(2000) meta
 Iteration 1, max rel difference of raked weights = 14.95826
 Iteration 2, max rel difference of raked weights = .21474256
 Iteration 3, max rel difference of raked weights = .02754514
 Iteration 4, max rel difference of raked weights = .00511347
 Iteration 5, max rel difference of raked weights = .00095888
 Iteration 6, max rel difference of raked weights = .00018036
 Iteration 7, max rel difference of raked weights = .00003391
 Iteration 8, max rel difference of raked weights = 6.377e-06
 Iteration 9, max rel difference of raked weights = 1.199e-06
 Iteration 10, max rel difference of raked weights = 2.254e-07
The worst relative discrepancy of  3.0e-08 is observed for race == 3
Target value =   20053682; achieved value =   20053682
Trimmed due to the upper absolute limit: 5 weights.
   Summary of the weight changes
              │     Mean    Std. dev.    Min        Max       CV
──────────────┼────────────────────────────────────────────────────
Orig weights  │    11318       7304      2000       79634   .6453
Raked weights │    22055      18908      4033      200000   .8573
Adjust factor │   2.1486               0.9220     18.9828
. char li rakedwgt3[]
  rakedwgt3[source]:          finalwgt
  rakedwgt3[objfcn]:          2.25435521346e-07
  rakedwgt3[maxctrl]:         3.00266822363e-08
  rakedwgt3[converged]:       1
  rakedwgt3[worstcat]:        3
  rakedwgt3[worstvar]:        race
  rakedwgt3[command]:         [pw=finalwgt], gen( rakedwgt3 ) ctotal( ACS2011_sex_age Census2011_region ..
  rakedwgt3[trimloabs]:       trimloabs(2000)
  rakedwgt3[trimhiabs]:       trimhiabs(200000)
  rakedwgt3[trimfrequency]:   sometimes
  rakedwgt3[hash1]:           2347674164
  rakedwgt3[mat3]:            Census2011_race
  rakedwgt3[over3]:           race
```

```
    rakedwgt3[totalof3]:        _one
    rakedwgt3[Census2011_race]: 7.48567503861e-09
    rakedwgt3[mat2]:            Census2011_region
    rakedwgt3[over2]:           region
    rakedwgt3[totalof2]:        _one
    rakedwgt3[Census2011_region]:
                                3.00266822363e-08
    rakedwgt3[mat1]:            ACS2011_sex_age
    rakedwgt3[over1]:           sex_age
    rakedwgt3[totalof1]:        _one
    rakedwgt3[ACS2011_sex_age]: 4.13778410340e-09
    rakedwgt3[note1]:           Raking controls used: ACS2011_sex_age Census2011_region Census2011_race
    rakedwgt3[note0]:           1
```

◁

The following characteristics are stored with the newly created weight variable (see [P] **char**).

`command`	The full command as typed by the user
matrix name	The relative matrix difference from the corresponding control total, see [D] **functions**
`trimhiabs`, `trimloabs`, `trimhirel`, `trimlorel`, `trimfrequency`	Corresponding trimming options, if specified
`maxctrl`	the greatest `mreldif` between the targets and the achieved weighted totals
`objfcn`	the value of the relative weight change D_k (**??**) at exit
`converged`	whether `ipfraking` exited due to convergence (1) vs. due to an increase in the objective function or reaching the limit on the number of iterations (0)

Also, `ipfraking` stores the notes regarding the control matrices used, and which of the margins did not match the control totals, if any. See [D] **notes**.

3.6 Replicate weights

As discussed in Section **??**, one of the greater challenges of weight calibration is ensuring that variance estimates take into account the greater precision achieved by adjusting the sample towards the fixed population quantities. As estimating the variances using linearization is cumbersome, replicate variance estimation may be more attractive.

▷ Example 6

The simplest code for calibrated replicate weights is obtained by calling `ipfraking` from within `bsweights` (Kolenikov 2010) which can pass the name of a replicate weight variable to an arbitrary calibration routine. In this example, we shall use the same settings as in Section 3.2 and thus we shall have the calibrated weight `rakedwgt2` which was produced in that example as the main weight for which the bootstrap weights provide the measure of sampling variability.

```
. set seed 2013

. set rmsg on
r; t=0.00 14:50:44

. bsweights bsw , reps(310) n(-1) balanced dots ///
>     calibrate( ipfraking [pw=@], replace nograph meta ///
>     ctotal( ACS2011_sex_age Census2011_region Census2011_race ) )
Balancing within strata:
...............................
Rescaling weights
..................................................    50
..................................................   100
..................................................   150
..................................................   200
..................................................   250
..................................................   300
..........
r; t=178.79 14:53:43

. forvalues k=1/310 {
  2.     _dots `k´ 0
  3.     assert `: char bsw`k´[converged]´ == 1
  4.     assert `: char bsw`k´[maxctrl]´ < 10*c(epsfloat)
  5. }
..................................................    50
..................................................   100
..................................................   150
..................................................   200
..................................................   250
..................................................   300
..........r; t=0.32 14:53:43

. set rmsg off

. svyset [pw=rakedwgt2], vce(bootstrap) bsrw( bsw* ) dof( 31 )

      pweight: rakedwgt2
          VCE: bootstrap
          MSE: off
    bsrweight: bsw1 bsw2 bsw3 bsw4 bsw5 bsw6 bsw7 bsw8 bsw9 bsw10 bsw11 bsw12
  (output omitted)
               bsw301 bsw302 bsw303 bsw304 bsw305 bsw306 bsw307 bsw308 bsw309
               bsw310
    Design df: 31
  Single unit: missing
     Strata 1: <one>
         SU 1: <observations>
        FPC 1: <zero>
```

The options of `bsweights` request 310 replicate weights (a multiple of 31 strata), resample one less PSU than available in a given stratum, and obtain the first-order balance within a stratum. With the 2 PSU/stratum design and these options, `bsweights` produces random half-samples of data. The at-character `@` is a placeholder for the name of the replicate weight variable. For explanations of these and other options of `bsweights`, see Kolenikov (2010). The procedure took about 3 minutes on a laptop computer, which can be considered moderately computationally intensive beyond interactive. A new option of `ipfraking` in the above code is `nograph` that suppresses the histograms. The additional asserts (Gould 2003) following the bootstrap weight generation demonstrate how the minimal quality assurance can be done on the bootstrap weights in the weight production workflow.

A more compact set of weights can be developed based on the existing BRR weights and a slightly more explicit code cycling over the weight variables:

```
. webuse nhanes2brr, clear
. svy : proportion highbp
(running proportion on estimation sample)
BRR replications (32)
────┼─── 1 ───┼─── 2 ───┼─── 3 ───┼─── 4 ───┼─── 5
................................
Survey: Proportion estimation    Number of obs    =      10351
                                 Population size  =  117157513
                                 Replications     =         32
                                 Design df        =         31

──────────────────────────────────────────────────────────────
             │                 BRR
             │ Proportion   Std. Err.     [95% Conf. Interval]
─────────────┼────────────────────────────────────────────────
highbp       │
           0 │   .8941859   .0067023      .8805165    .9078553
           1 │   .1058141   .0067023      .0921447    .1194835
──────────────────────────────────────────────────────────────

. generate byte _one = 1
. generate byte age_grp = 1 + (age>=40) + (age>=60) if !mi(age)
. generate sex_age = sex*10 + age_grp
. ipfraking [pw=finalwgt], gen( rakedwgt2 ) ///
>     ctotal( ACS2011_sex_age Census2011_region Census2011_race )
 Iteration 1, max rel difference of raked weights = 14.95826
 Iteration 2, max rel difference of raked weights = .19495004
 Iteration 3, max rel difference of raked weights = .02204455
 Iteration 4, max rel difference of raked weights = .00315355
 Iteration 5, max rel difference of raked weights = .00043857
 Iteration 6, max rel difference of raked weights = .00006061
 Iteration 7, max rel difference of raked weights = 8.365e-06
 Iteration 8, max rel difference of raked weights = 1.154e-06
 Iteration 9, max rel difference of raked weights = 1.593e-07
```

(*Continued on next page*)

```
   Summary of the weight changes
              |    Mean    Std. dev.    Min        Max       CV
--------------+--------------------------------------------------
Orig weights  |    11318       7304      2000      79634   .6453
Raked weights |    22055      19227      4050     338675   .8717
Adjust factor |   2.1464               0.9264    18.3694
. forvalues k=1/32 {
  2.     quietly ipfraking [pw=brr_`k´], gen( brrc_`k´ ) nograph ///
>         ctotal( ACS2011_sex_age Census2011_region Census2011_race )
  3.     _dots `k´ 0
  4. }
................................
. svyset [pw=rakedwgt2], vce(brr) brrw( brrc* ) dof( 31 )
      pweight: rakedwgt2
          VCE: brr
          MSE: off
    brrweight: brrc_1 brrc_2 brrc_3 brrc_4 brrc_5 brrc_6 brrc_7 brrc_8 brrc_9
               brrc_10 brrc_11 brrc_12 brrc_13 brrc_14 brrc_15 brrc_16
               brrc_17 brrc_18 brrc_19 brrc_20 brrc_21 brrc_22 brrc_23
               brrc_24 brrc_25 brrc_26 brrc_27 brrc_28 brrc_29 brrc_30
               brrc_31 brrc_32
    Design df: 31
  Single unit: missing
     Strata 1: <one>
         SU 1: <observations>
        FPC 1: <zero>
. svy : proportion highbp
(running proportion on estimation sample)
BRR replications (32)
----+--- 1 ---+--- 2 ---+--- 3 ---+--- 4 ---+--- 5
................................
Survey: Proportion estimation    Number of obs    =      10351
                                 Population size  =  228294169
                                 Replications     =         32
                                 Design df        =         31
--------------------------------------------------------------
             |                 BRR
             | Proportion   Std. Err.     [95% Conf. Interval]
-------------+------------------------------------------------
highbp       |
           0 |   .8730544   .0081501      .8564323    .8896766
           1 |   .1269456   .0081501      .1103234    .1435677
--------------------------------------------------------------
```

The data can be analyzed with the standard `svy` prefix, and the standard errors will appropriately capture the efficiency gains from weight calibration. No additional action is required for the analyst or researcher.

◁

CAUTION: the input weights for the replicate weight calibration must be the probability replicate weights. The existing NHANES II weights have been adjusted for non-response and calibrated by the data provider, and are used above for demonstration purposes only.

4 Error messages and troubleshooting

4.1 Critical errors

The following critical errors will stop execution of `ipfraking`.

`pweight is required`

The `[pweight=...]` component of `ipfraking` syntax is required. Probability weights must be specified as inputs to `ipfraking`.

`ctotal() is required`

The `ctotal()` component of `ipfraking` syntax is required. Names of the matrices containing the control totals must be specified.

`one and only one of generate() or replace must be specified`

Either `generate()` option with the name of the new variable must be supplied to `ipfraking`, or `replace` to replace the variable specified in `[pw=...]` statement.

`raking procedure appears diverging`

The maximum relative difference of weights D_k has increased from the previous iteration. This may or may not indicate a problem. Re-run `ipfraking` with `nodivergence` option to override the warning.

`cannot process matrix` *matrix_name*

For whatever reason, `ipfraking` could not process this matrix. The matrix may not have been defined or the variables in this matrix cannot be found.

`variable` *varname* `corresponding to the control matrix` *matrix_name*
`not found`

The variables contained in row or column names of this matrix cannot be found.

varname1 `and` *varname2* `variables are not compatible`

When running `total` *varname1*`, over(`*varname2*`)`, an error was encountered. One of the variables may be a string variable or have missing values resulting in an empty estimation sample.

`categories of` *varname* `do not match in the control` *matrix_name*
`and in the data (nolab option)`

There was a mismatch in the categories of *varname* found in the data and in the control matrix *matrix_name*. This could happen for any of the following reasons: (i) there were more categories in one than in the other; (ii) the entries are in the wrong order in the control matrix; (iii) the labels in the control matrix do not correspond to the category values in the data set; (iv) the control matrix was obtained via `total` *varname2*`, over(`*varname*`)`, but `nolabel` suboption of `over()` was omitted, and the labels of the control matrix may include some unexpected text. Tabulate *varname* without labels, and compare the results to the matrix listing of the *matrix_name*.

`cannot compute controls for` *matrix_name* `over` *varname* `with the current weights`

This is a generic error message that something bad happened while `ipfraking` was computing the totals for the current set of weights. This error message should generally be very rare, but as computing the totals may be the slowest operation of the iterative optimization process, stopping `ipfraking` with a *Ctrl+Break* combination or the *Break* GUI button may produce this error message.

`trimhiabs|trimloabs|trimhirel|trimlorel must be a positive number`

One or more of the trimming options are given as a non-positive number or a non-number.

`trimhiabs must be greater than trimloabs`

`trimhirel must be greater than trimlorel`

The trimming parameters are illogical (the lower bound is greater than the upper bound). Respecify the values of the trimming parameters.

4.2 Other errors and warnings

The following warning messages may be produced by `ipfraking`. The program will continue running, but you must double-check the results for potential problems.

`the totals of the control matrices are different`

The sum of values of the control matrices are different. These sums will be listed for review. Convergence is still possible, but some of the control total checks are likely to fail.

`trimfrequency() option is specified without numeric settings; will be ignored`

The option `trimfrequency()` was specified without any numeric trimming options. There is no way to interpret this, and `ipfraking` will proceed without trimming.

`trimfrequency() option is specified incorrectly, assume default value (sometimes)`

Something other than `often`, `sometimes` or `once` was supplied in `trimfrequency`, and the default value is being used instead.

`raking procedure did not converge`

The maximum number of iterations was reached, but weights never met the convergence criteria (see step **??** of Algorithm 2 in Section **??**). The user may want to increase the number of iterations or relax convergence criteria.

`the controls` *matrix_name* `did not match`

After convergence of weights was declared, `ipfraking` checked again the control totals, and found that the results differed from the target for one or more of the control total matrices. Any of the following can cause this: (i) the sum of entries of this particular matrix differs from the others; (ii) the trimming options are too restrictive, and do not allow the weights to adjust enough; (iii) the problem may not have a solution due to incompatible control totals or a bad sample.

`division by zero weighted total encountered with` *matrix_name* `control`

The weights for a category of the control variable summed to zero. `ipfraking` will skip calibration over this variable and proceed to the next one.

`# missing values of` *varname* `encountered; convergence will be impaired`

A control variable has missing values in the calibration sample. There is little way for `ipfraking` to figure out how to deal with the weights for the observations with missing values. The user would need either to restrict the sample to non-missing values of all control variables, to impute the missing values or to create a separate category for the missing values of a given control variable (which may lead to difficulties in defining valid population control totals for it).

Acknowledgements

The author is grateful to Ben Phillips, Andrew Burkey and Brady West, as well as the editor and an anonymous referee, who suggested additional functionality and provided helpful comments to improve the readability of this article. The opinions stated in this paper are of the author only, and do not represent the position of Abt Associates.

5 References

AAPOR. 2014. *AAPOR Terms and Conditions for Transparency Certification.* The American Association for Public Opinion Research. Available at http://www.aapor.org/AAPOR$_{M}ain/media/MainSiteFiles/TI - Terms - and - Conditions - 10 - 4 - 17.pdf$.

Binder, D. A., and G. R. Roberts. 2003. Design-based and Model-based Methods for Estimating Model Parameters. In *Analysis of Survey Data*, ed. R. L. Chambers and C. J. Skinner, chap. 3. New York: John Wiley & Sons.

Deville, J. C., and C. E. Särndal. 1992. Calibration Estimators in Survey Sampling. *Journal of the American Statistical Association* 87(418): 376–382.

Deville, J. C., C. E. Särndal, and O. Sautory. 1993. Generalized Raking Procedures in Survey Sampling. *Journal of the American Statistical Association* 88(423): 1013–1020.

Gould, W. 2003. Stata tip 3: How to be assertive. *Stata Journal* 3(4).

Groves, R. M., D. A. Dillman, J. L. Eltinge, and R. J. A. Little. 2001. *Survey Nonresponse*. Wiley Series in Survey Methodology, Wiley-Interscience.

Holt, D., and T. M. F. Smith. 1979. Post Stratification. *Journal of the Royal Statistical Society, Series A* 142(1): 33–46.

Horvitz, D. G., and D. J. Thompson. 1952. A Generalization of Sampling Without Replacement From a Finite Universe. *Journal of the American Statistical Association* 47(260): 663–685.

Kolenikov, S. 2010. Resampling inference with complex survey data. *The Stata Journal* 10: 165–199.

———. 2014. Calibrating survey data using iterative proportional fitting. *The Stata Journal* 14(1): 22–59.

———. 2016. Post-stratification or non-response adjustment? *Survey Practice* 9(3). Available at http://www.surveypractice.org/index.php/SurveyPractice/article/view/315.

Korn, E. L., and B. I. Graubard. 1995. Analysis of Large Health Surveys: Accounting for the Sampling Design. *Journal of the Royal Statistical Society, Series A* 158(2): 263–295.

———. 1999. *Analysis of Health Surveys*. John Wiley and Sons.

Kott, P. S. 2006. Using Calibration WeightingtoAdjust for Nonresponse andCoverage Errors. *Survey Methodology* 32(2): 133–142.

———. 2009. Calibration Weighting: Combining Probability Samples and Linear Prediction Models. In *Sample Surveys: Inference and Analysis*, ed. D. Pfeffermann and C. R. Rao, vol. 29B of *Handbook of Statistics*, chap. 25. Oxford, UK: Elsevier.

McConnell, S. 2004. *Code Complete: A Practical Handbook of Software Construction*. 2nd ed. Microsoft Press.

Pew Research Center. 2012. Assessing the Representativeness of Public Opinion Surveys. Technical report, Pew Research Center for People and Press. Available at http://www.people-press.org/files/legacy-pdf/Assessing the Representativeness of Public Opinion Surveys.pdf.

Pfeffermann, D. 1993. The role of sampling weights when modeling survey data. *International Statistical Review* 61: 317–337.

Särndal, C.-E. 2007. The calibration approach in survey theory and practice. *Survey Methodology* 33(2): 99–119.

Thompson, M. E. 1997. *Theory of Sample Surveys*, vol. 74 of *Monographs on Statistics and Applied Probability*. New York: Chapman & Hall/CRC.

About the author

Stanislav (Stas) Kolenikov is a Senior Scientist at Abt Associates. His work involves applications of statistical methods in data collection for public opinion research, public health, transportation, and other disciplines that utilize collection of survey data. Within survey methodology, his expertise includes advanced sampling techniques, survey weighting, calibration, missing data imputation, variance estimation, nonresponse analysis and adjustment, and small area estimation. Besides survey statistics, Stas has extensive experience developing and applying statistical methods in social sciences, with focus on structural equation modeling and microeconometrics. He has been writing Stata programs since 1998 when Stata was version 5.