Using Wcalc and the Dyad Ratios Algorithm

James A. Stimson

Updated 7-1-2008

Wcalc is a program which implements an algorithm to build dimensional "factor scores" from dated items with only partially overlapping cases. Calc requires a PC-compatible machine running on a Windows system, Windows/95 or newer.

Wcalc expects to read ascii data files with file extension ".txt", although other extensions may also be used. It creates output files with the same root file names as the input file selected, although that may be altered at run time. Output files carry the extensions ".rpt" for a report file and ".prn" for the calculated series.

1 Dates and Aggregation

Wcalc expects to see input with a date for the field (or release) dates of surveys. It decodes those dates into aggregation periods, from 1 to T, at run time and produces an aggregated version of the file for input to the dimensional algorithm. This happens anew each time it is used and so the aggregated data are not retained.

The user's responsibility is simply to provide dates for the survey responses and then inform the program of the parameters needed for aggregation, (1) period (daily, monthly, quarterly, annual, or multiple years), (2) first period and (3) last period. The program then puts the responses into appropriate periods and performs (weighted) averaging when more than one survey result is available per period. This permits keeping archived data in a form that retains all the original information and making the choice of aggregation period and beginning and ending dates variable.

2 Input

The input data file is a comma- or tab-delimited ASCII file with one line for each set of survey marginals. That line contains, in order, VARNAME, DATE, VALUE, N.

Example line:

AIDMIN, 5-01-1968, 37,1479

Note that blanks are allowed, but not required. Excel tab-delimited txt files will be correctly formatted, but aren't the only means of producing the file.

VARNAME is an alphanumeric string identifying a series. If it complies with the normal rules for computer variable naming, it requires no punctuation except a following comma or tab to delimit it from DATE. If it includes embedded blanks, then it must be enclosed in quotation marks, as in "VAR NAME", ...

DATE is a string in one of the common forms: mm/dd/yy, mm/dd/yyyy, mm-dd-yy or the European equivalents, dd/mm/yy, dd/mm/yyyy, or dd-mm-yy. These have in common that they always contain two delimiters, either "/" or "-". Leading zeros on month and day are allowable, but not required: 05/ and 5/ are both 5. Two digit year fields are presumed to be 20th Century dates, and 1900 is added to them.

Full dates must always be present, even if they are more specific than needed (e.g. in annual data the user doesn't care about month or day). Where specifics are unknown (and insignificant) any legitimate date will do. Dates must be legitimate dates, i.e.,those that actually exist. The Microsoft date converter will not tolerate June 31 or even February 29 for a year which is not leap year. If they don't decode to an actual date, they cause a runtime error.

VALUE is a single indicator of the concept to be measured. It could be one of the response categories from a survey item, e.g. percent approve, or an index of multiple responses. In any case, the value must NOT include zero or negative values, as these violate the ratio-based structure of the algorithm. Near zero values ought also to be avoided. Value can be represented in any standard numeric format.

As in principal components analysis, the polarity of the resulting dimension measure is determined by the dominant direction of coding of the raw items. Coding all in the same direction, e.g. liberal, approving, or whatever, will insure against a solution opposite of what is intended. Negative covariance between item and scale, when observed, is treated as meaningful (a revision from earlier versions of the algorithm.)

N is the number of cases in the survey. It is used by the program only for the limited purpose of weighting in the case of multiple instances of a variable within an aggregation

interval. When either (a) N is unknown, or (b) the user prefers unweighted averages, enter 0. A zero is replaced by 1000 for the averaging process, a fact which matters only if N is known for some instances of a variable and unknown for others within an aggregation interval. Even if N is unknown or weighting is not desired, some value must be present or the file will be misread.

Case Order: The input file must be sorted by (1) VARNAME and (within that) DATE (ascending). The program searches for break points, both between variables and between aggregation intervals within variables, as it reads. Any mis-sorting thus leads to undesirable consequences. VARNAME is case specific, e.g., "Clinton" and "clinton" are assumed to be different series. The order of VARNAME's within the file has no consequence.

3 Program Interaction

Double-clicking on Wcalc.exe results in the standard Windows dialogue box for selecting an input file. It employs the default file extension filter ".txt" but may be modified in the usual way.

The program requests an output file root name, defaulting to the input file root name.

On execution the program first makes a complete pass through the input file to determine the range of dates represented in the file. The range will be the maximum possible span if the series associated with earliest and latest items in the file are actually used. [Examples of unused items are one-time only series or multiple series which nonetheless fall in only a single aggregation interval.]

Click on the **Aggregation Interval** command button to select one of the possibilities, daily, monthly, quarterly, annual, or multiple years. Selecting the last case produces a further prompt for the multiple.

Next comes an options form. Text windows allow override of the minimum and maximum dates, the form varying with the type of aggregation interval. Overriding would be appropriate either because the analyst desires a shorter span than the available data or because the program estimates of beginning and ending maxima/ minima are not accurate, as noted above. Since the date range is evaluated without knowledge of aggregation interval, the program cannot know which items will be deleted from analysis.

Number of dimensions to extract is selectable as 1 (default) or 2. Estimating a second dimension with the data problems typical of survey marginals is hazardous and largely

untested. It is not recommended in the absence of a strong prior and very clean and dense data.

Eccentric Calendar (default: not selected) allows breaking an annual or multiple year estimate into "years" that end after the 3rd quarter (September 30) and begin with the 4th quarter of the previous year (October 1). If selected, dates through September 30, 2001 would count toward 2001, with 2002 beginning on October 1, 2001. [This option was developed to permit election analyses in which current year public opinion would predict the current election without time-order problems.]

Exponential Smoothing (default: selected) smoothing is optionally performed on the raw data series prior to extraction. The rationale is that one wishes to observe common movements in the evolution of issue series and not tailor a fit to particular zigs and zags that may be random variation around a deterministic process. Smoothing is particularly appropriate with survey marginals, known to contain random fluctuation from sampling.

The exponential smoothing model is:

$y_t = \alpha x_t + (1 - \alpha) x_{t-1}$

where y is the smoothed version of x. The intuition is that if the past, x_{t-1}, provides any useful information for predicting y_t, then some portion of the variation in x_t is a deviation from the smooth path of x. This is seen in zig-zag behavior, where the series tends to return to normal levels after extreme movements away from them.

The α parameter is estimated by minimizing within sample forecast error. Thus it is fully determined by the data. Exponential smoothing has the desirable property that it will not oversmooth. If the data are already smooth, a situation that often occurs with annual aggregation levels, then α converges on 1.0 and $y_t = x_t$. Smoothing occurs in both forward and backward directions in time, the result of which is that the raw data series become exponentially weighted moving averages of past and future values.

Smoothing operates on the forward and backward recursion series during estimation. That means, in effect that the smoothed value is presumed to be a better measure of the true level of the series than is the original, and that it is the ratios of the smoothed values to past and future(smoothed) values of the series that drive the ultimate measure. The impact of smoothing varies in direct proportion to the apparent randomness of the series. Where the original series are highly patterned, the impact of smoothing is rarely discernable. Where, in contrast, they exhibit a good deal of period-to-period zig-zag fluctuation, the effect of smoothing is large.

After the options screen, click on **Extract** to begin execution. During the iterative estimation

process the screen display shows a graphic representation of estimated series and information on the progress of each iteration. That information is a "convergence" score, the maximum change in any item-scale correlation (weighted by number of periods in which the item is available) between one iteration and the next. It is a summary estimate of the effects of the most recent changes in the communality estimates. It should converge on zero as solution is approached. "Criterion" is the numeric solution criterion against which the convergence score is judged. It starts out (and usually stays at) .001, but grows by a factor of two on failure—convergence scores getting larger between iterations. "Items" is the number of items in the analysis, which may be smaller than the number in the input file.

The graphic display gives some indication of the progress of the iterative solution. Particularly for the first iteration or two it exaggerates the differences between different estimations of the series, because series metrics (means and standard deviations) are initially unstable. Periods for which all series are missing are represented as circles on the graph. They are subject to interpolation in the final series.

4 Metric

Conversion of data values to ratios causes the original metric to be lost. Rather than produce output in an arbitrary and meaningless metric the program attempts to simulate the original by using the means and standard deviations of the input items—weighted by their contribution to the scale—to produce a summary metric for the scale. This is usually pretty successful. But there are conditions where it will not be, for example if many input items cover over a limited period where values might be atypical.

Advice There is nothing holy about this method. If you have a better means to determine series metric, use it.

5 Output

Output I: rootname.prn The prn file contains two or three items per line, the date of each period and the estimated series value. Date is YYYY for annual (or multiples of annual) aggregation or YYYY MM, where M is numeric month or quarter, YYYY Q of the estimate. This (tab-delimited) ASCII file can easily be imported into graphics software for display.

For daily estimation, the output includes a 5 digit integer date, followed by a text string date, followed by the estimated value for that date. The integer date will convert to a date

variable in spreadsheets and database managers, which will be more convenient for some purposes than the text string. After input (e.g., to Excel) select the date column and format it as a date, and the integer values will be replaced by a date representation of your choice.

Output II: rootname.rpt The report file consists of four sections, a brief header documenting the estimation data and options, an iteration history, with the same information as the on-screen display, and item information.

Iteration history includes the convergence estimate, the cutoff criterion, number of series the analysis. Also "F-B Reliability" is a product moment correlation of the two independent estimations (forward and backward recursion) of the latent series. AlphaF and AlphaB are separately estimated α's for forward and backward recursions.

The item information includes the number of time points for which the item is available (after aggregation into intervals) and the item-scale correlation at solution. This information is interpretable as a factor loading. Its square is interpretable as a communality estimate.

6 Problems and Pitfalls

WCalc always produces correct results when it runs, but sometimes it doesn't run at all. Problems are always related to the input data file. The program is extremely sensitive to data errors. It wants to read "N" cases each exactly of the structure outlined above. When it does not, it crashes and usually produces no useful diagnostic.

Three Common Pitfalls: More than 90% of failures are due to either (1) inclusion of a text header line in the file, (2) a missing item, or (3) a comma in a data item.

1. The fix for #1 is obvious: don't include a header.
2. #2 is most often a result of a missing number of cases. It is OK to enter a zero when N is unknown. It is not OK to leave it blank. Then the program tries to read four items, finds only three, and then totally misreads the rest of the file. This inevitably causes a crash when it will try to assign the alphabetic string in VARNAME to a numeric variable. To diagnose program failure try to load your file into Excel and then observe whether or not it produces exactly four columns for all N cases.
3. The comma is a delimiter for free format files. Thus a number of cases entry of say "1,000" will cause the program to read "1" and then the next item waiting to be read is "000." Because there is now an extra item in the list, the sequence of reading is now

wrong which will result quickly in a "Type Mismatch" error, when the program tries to assign the variable name in the following line to a numeric variable.

A similar issue is European use of the comma to form fractions as in "1/2 = 0,50". Then xenophobic American arithmetic requires a decimal point for fractional numbers.

Mis-sorting About the only other possibility for problems comes from failure to sort dates within variable names. (The CCalc version (below) auto-sorts to eliminate this problem, but the legacy version does not.) For this case the program will generally run to completion, but produce nonsense results. Remember that variable names are case sensitive, so Vname and VNAME are two different variables (or three if arranged in the order Vname, VNAME, Vname.)

When all else fails: **jstimson@email.unc.edu** (with input file as an email attachment).

The Algorithm See Stimson, *Public Opinion in America: Moods, Cycles, and Swings*, 2nd ed., Westview Press: 1999, particularly Appendix 1.

7 Other Software

WCalc: The Current Version (July 1, 2008) This software has two important virtues and numerous vices. The virtues are (1) that it produces correct results, and (2) that it is *known* to do so. The newer versions below also produce correct results, but they have not been widely used, not tested on varied data.

The vices of this version don't show; they are under the hood. This code, to begin, is very old. Most of it was written in 1988 for an old and primitive Basic compiler produced by IBM in the DOS era. It was ported to Windows (and Visual Basic) in 1997, but much of its DOS structure remains. It is a classic example of "spaghetti code," a linear structure filled with logic-defying "goto's." The vice of that legacy programming style is that it is painfully difficult to understand the logic of the code, even for the person who created it. Maintenance is a nightmare. This code has served well and continues to. But it is of an earlier era and deserves to be retired, which is why I have totally rewritten it in C++.

The dyad ratios algorithm is also implemented in an R function (extract.r) and in C++ (ccalc.exe).

extract for R The R version is the bare bones of the dyad ratios algorithm written as a function which can be invoked from the R platform. Its intended use is for inclusion in Zelig,

but it also will stand alone from the R platform. I have not yet made it public because I have not had time to fully document it, but it is available on request.

CCalc CCalc is a Windows GUI version of WCalc written in C++. It is modern code, fully modular and object oriented. It has one (at least) remaining bug that causes it to fail on occasion (but when it produces any results, it produces correct ones.) This is a more user friendly and capable version than the original. It just isn't quite finished.

Its base in Microsoft C++ 6.0. I intend to produce this as a Windows "console" program (i.e., not GUI), abandoning the excesses of Microsoft to produce a program that actually can be used. It will be open source and the C++ code should easily port to Linux or Unix.