Whittaker-Henderson-Lowrie Graduation

# Whittaker-Henderson Graduation

## Overview

Whittaker-Henderson graduation determines a set of values which minimize a difference equation. The method allows for an explicit balancing between goodness of fit and smoothness.

For clarity and brevity, this description will speak of graduating mortality rates. This is merely a specific and common case. The method is equally applicable to rates of many other types.

## Definitions

“Raw” means the mortality rates determined from an experience study, actual deaths divided by actual exposure.

“Wt” means the weights corresponding to each raw rate. Typically the exposure is used as the weights. The weight with the best theoretical support is likely the reciprocal of the variance, but the variance is often not available. In my work I always normalize the weights; more on that later. There is always one weight for each raw rate. It is possible to use a zero weight and an artificial raw rate for ages for which there is no data.

“Grad” means the graduated mortality rates.

“n” means the order of difference equation being used to express smoothness.

“h” means the balancing factor between fit and smoothness.

## Formula

This formula is minimized by the Whittaker-Henderson method. The first term is an expression for the goodness of fit of the graduated values to the raw; the second term is an expression for the degree of smoothness of the graduated values. For a solution to this equation, check a standard text on Graduation.

## Implications

The factor h is determined arbitrarily. A low value of h puts more emphasis on fit than smoothness. An extremely low value of h returns the raw rates; there is essentially no graduation. A high value of h puts more emphasis on smoothness than fit. An extremely high value of h yields a least squares fit to a polynomial of degree (n - 1).

The method calculates directly a complete set of graduated values which achieve the desired balance between fit and smoothness. Also the total number of deaths and the average age at death, when the exposures are used as weights, will be the same for the graduated rates as for the raw rates.

# Lowrie’s variation

Walter B. Lowrie’s paper “An Extension of the Whittaker-Henderson Method of Graduation” (see 34 TSA 329) significantly improves Whittaker-Henderson for actuarial use. Lowrie observed that it was possible, by a relatively minor variation in the formula to define smoothness as an exponential plus a polynomial rather than as a polynomial alone. This is significant since mortality rates are generally believed to be closer to exponential than a polynomial over a fairly wide range of ages.

By the way, this is not the same as an exponential with a polynomial exponent. That approach was already available by graduating the logarithm of the mortality rates and exponentiating the graduated values

The only drawback is that the base of the exponential is a parameter than must be determined prior to the graduation. The method does not yield the base. This is not normally a difficulty. I have found good results by using the weighted average rate of increase in the mortality rate from age to age.

Lowrie’s paper contains a second extension which can be used with the normal definition of smoothness or Lowrie’s extension to exponential smoothness. He added a third term to the difference equation which involves the weighted fit of the graduated rates to those of a standard table. The idea is to emphasize fit to the raw rates where there is lots of data and fit to a standard table where there is little. To achieve this result the weights for the standard table must be large where there is little data and small where there is much. One simple approach is to use the average exposure less the actual exposure at each age with a zero floor.

The formula for Lowrie’s variation is

where

*l* is the factor for fit to the standard table, in the range [0,1]

*r* is the assumed growth rate; i.e., (1+*r*) is the base of the exponential

*Ws* is the weights for fit to the standard table

*Std* is the standard table.

# Two-dimensional graduation

When faced with the need to graduate in two-dimensions, most have graduated first along columns and then along rows, or vice versa. That approach is not ideal because it puts one graduation in conflict with another.

Fortunately there is a fairly straight forward extension of Whittaker-Henderson to two dimensions. Instead of one smoothness term, there are two, one for horizontal smoothness with differences taken across rows, and one for vertical smoothness with differences taken down columns. There are also two balance factors to give relative emphasis to fit, horizontal smoothness and vertical smoothness. Note that fit is still defined globally, not separately for horizontal and vertical.

In the following formula, the subscripts h and v refer to horizontal and vertical differences. The formula illustrates order-m differences in the horizontal direction for smoothness and order-n differences in the vertical direction. The double sum indicates summing across both rows and columns, in either order.

It is fairly simple to add Lowrie’s variation to two dimensions. There can be horizontal and/or vertical growth rates. There can be a term for fit to a standard table of the same dimensions.

# Using WHGrad.dll

I have created the dynamic link library, WHGrad.dll, in C, under Windows 2000 and later for 32-bit applications, to perform all of the above variations of Whittaker-Henderson-Lowrie graduation. I am making it available without charge with the hope that it will encourage actuaries and others to make more effective use of experience data that is available to them.

The dll is my own work, but I used ideas from all of the sources in the references given below. Since I developed the dll on a computer owned by Sun Life Financial (SLF), I needed approval from SLF to distribute the dll, and SLF has generously given that approval.

The dll can be used with most computer programming languages under various versions of Windows. Personally I have used it with IBM’s APL2 for workstations and with Microsoft Excel under Windows 2000, Windows XP, and Windows 7, 8 and 10. If you need the dll for a different platform and you are able to compile C code in that platform, I would be happy to send you the source code, but you will be on your own to get it working.

The following documents the use of WHGrad.dll. If you use it only under Excel, you can skip the rest of this section because my sample Excel workbook referred to in the next section contains a module that handles all the references to WHGrad.dll.

I present below the two main functions that are available using C notation. If the notation doesn’t make sense, ask one of your IT specialists. The entry names for 1-dimensional and 2-dimensional graduations are \_WHL1G@60 and \_WHL2G@84, respectively.

int \_\_declspec(dllexport) \_System WHL1G(long n, long order, double h,

double rate, double l,

double \*raw, double \*weight, double \*grad, double \*stat,

double \*wt\_std, double \*std, char \*msg) {

// dll access to function to do 1-dimensional Whittaker-Henderson-Lowrie graduation

// returns 0 if successful, otherwise an error code (see msg)

// n: the number of elements to graduate

// order: order of differences to minimize (1,2,3 or 4)

// h: balance factor for fit vs smoothness

// l: balance between fit to raw data and fit to the standard table

// rate: the rate of growth of an exponential as the standard of smoothness

// raw: ungraduated values

// weight: the weights for each raw value

// grad: graduated values

// stat: F\*(1-l) + T\*l + h\*S, F, S, 0, T

// The expression minimized is F\*(1-l) + T\*l + h\*S

// where F is the weighted sum of the squared error between the graduated and raw

// data, T is the weighted sum of squared error between graduated and standard table,

// and S is the sum of the squared order-differences

// wt\_std: the weights for the standard table

// std: the rates for the standard table

// msg: error message if return code is non-zero

int \_\_declspec(dllexport) \_System WHL2G(long row, long col, long horder, long vorder,

double h, double v, double hrate, double vrate, double l,

double \*raw, double \*weight, double \*grad, double \*stat,

double \*wt\_std, double \*std, char \*msg) {

// dll access to function to do 2-dimensional Whittaker-Henderson-Lowrie graduation

// result and arguments are as above except as follows

// references to matrices must have data in row-major order

// row, col: number of rows and columns in matrices

// horder, vorder: order of differences for horizontal and vertical smoothness

// h, v: balance factors for fit vs horizontal and vertical smoothness

// hrate, vrate: the rate of growth for horizontal and vertical exponentials

// stat: F\*(1-l) + T\*l + h\*HS + v\*VS, F, HS, VS, T

// The expression minimized is F\*(1-l) + T\*l + h\*HS + v\*VS

// where HS and VS are sums of squared horizontal horder-differences and vertical

// vorder-differences. Other elements are as for 1-dimensional.

# Sample in Excel

See the workbook, GradSample.xls.

For the WH functions to work, you will need to put the file, WHGrad.dll, into a directory in your path. However, some are not allowed by IT rules to put WHGrad.dll in a suitable directory, and some may simply prefer to put it elsewhere. In that case, both Declare statements in GradSample.xls must be modified. The value following the Lib keyword is “WHGrad.dll”; that should be changed to specify the fully qualified name (including directories). For example, you might specify “C:\Grad\WH\WHGrad.dll”. Substitute the actual directory structure you used.

# 64-bit applications

If you work with a 64-bit application, like 64-bit Excel, then all of the above two sections apply to WHGrad64.dll and GradSample64.xlsm instead.

Note that it is the application that counts. While I used 32-bit Excel on a 64-bit machine and operating system, I used the 32-bit version. Now I have the 64-bit version of Excel and I use WHGrad64.dll.

WHGrad64.dll has two other functions that may be useful to you. MortFn fits a mortality function to raw mortality rates. SolvEq solves systems of linear equations. See the webpage where you found this document, <http://www.howardfamily.ca/graduation>.

# Validation of the dll

Good actuarial practice requires that code developed by others be validated. The workbook ValidateWHgrad.xlsm (or ValidateWHgrad64.xlsm) is provided for that purpose. It contains a simple case of a parabola which the graduation should reproduce is the order of difference is 3 or higher. It also contains a sample 2-dimensional graduation both by the dll and by VBA code. Anyone who is familiar with WH graduation and with VBA can verify that the code is correctly implementing the WH method. See the worksheet “Describe” and the VBA code for more information.

# VBA Utilities for graduation

This section gives a brief description of the procedures in the module called “Grad”, which you can find in the sample workbook. That module has a detailed definition of each function, and several of them are used in the sample workbook.

To make the following procedures available to your workspace, drag the module, Grad, from the sample workbook to your workbook while in the VBA window. To do any Whittaker-Henderson-Lowrie graduation, you must first copy the file, WHGrad.dll, into a directory in your path, such as c:\Windows.

Many of the utility functions work as Excel array functions. If you are not familiar with array functions, check Excel’s help.

1. Graduate. This is the main function for Whittaker-Henderson-Lowrie graduation. However, it is set up to be called by a VBA procedure, and not directly from Excel.
2. GradRange. This function calls Graduate. It is set up so that it can easily be used in Excel as an array function. Most of the arguments will normally be ranges in Excel.
3. Stats. This is a 5-element variable. It holds the graduation statistics from the last run. The 5-elements are always the value of the function being minimized, the term for fit, the term for horizontal smoothness, the term for vertical smoothness (if 2-dimensional), and the term for fit to the standard table (if *l* is not zero).
4. GradStat. This function returns only the 5-element statistics vector and discards the graduated values. The arguments are the same as for GradRange.
5. PlotGrad. This subroutine plots raw and graduated values vs age.
6. PlotStat. This subroutine plots the graduations statistics versus h.
7. Lagrange. This function interpolates (or extrapolates) using a Lagrange polynomial passing through the pivotal points provided.
8. LagrRange. This function calls Lagrange. It uses arguments that are Excel ranges, and therefore, works well as an Excel array function.
9. LSEst. This function finds a weighted Least Squares polynomial, and calculates the functional values with the specified arguments.
10. LSRange. This function calls LSEst. It uses range arguments, and gives a range result.

# Tips on Graduation

1. It is helpful to normalize the weights. That it, multiply all the weights by a factor so that they sum to the number of items being graduated. Then if all the squared errors and all the squared differences were about the same size, the fit and smoothness factors would be about the same size. With normalized weights you can think of values of h less than 1 giving greater emphasis on fit, and values more than 1 giving greater emphasis on smoothness. (The VBA utility, Graduate, normalizes weights by default, but the dll does not.)
2. Since the goal in graduation is usually to discover the smooth underlying relationship that is hidden by the fluctuation in the raw data, you will usually emphasize smoothness over fit. I usually find a satisfactory result with h in the range of 100 to 2000. If fewer deaths, h will need to be larger.
3. However, you will rarely need a smoothness sum (i.e. the sum of the squared differences) less than 1E-8 for 1-dimensional graduation, and certainly not less than 1E-7 for 2-dimensional. I usually try to get the average squared difference in the range of 1E-10 to 1E-9.
4. I know of no objective way to determine the “best” value of h for a particular graduation. I suggest you try several values and look at a graph of the results.
5. When graphing a graduated mortality table, I suggest looking at the ratio of graduated to a standard table. This keeps the range of values narrow enough to see clearly over many ages. If you want to look at the rates themselves, use a semi-log plot.
6. Be cautious about extending your graduation into ages where you have very little data. Since smoothness counts equally at all ages, but fit largely ignores areas with very little data, your graduation may give you rates that are unreasonable at those ages. In particular, if you use 3rd order WH, you will tend to get an upswing in rates at both the low and high ages. You may want that for an insurance table, but definitely not for an annuity table.
7. The above problem can be minimized (for annuity tables) by using a growth rate in Lowrie’s variation.
8. When graduating lapse rates by age and duration, expect much less smoothness by duration than by age. Set the balance factor for duration much lower than for age.
9. A good way to study rates of mortality improvement by age is to start with a 2-dimensional graduation of data by age and year with fairly strong smoothness by year.
10. If constructing a table over a wide range of ages, it may be necessary to do the construction in parts. For example, there is appreciable annuity experience only for ages from the 60’s to the 90’s. The oldest ages are usually either completely theoretical or generally reflect population data. The youngest ages may be constructed from insurance studies. If the parts overlap, you need to blend from one to the other smoothly. If the parts are disjoint, an interpolation method such as Lagrange can be useful because it moves smoothly for the last few ages of one part to the first few of the next.
11. Once you become familiar with these tools, you will find that graduating is easy. You should be able to construct a useful table in less than one hour. Why settle for the usual practice of multiplying a standard table by a single factor?

# References

London, Dick. *Graduation: the revision of estimates.* Abington: ACTEX Publications, 1985.

Lowrie, Walter B. “An Extension of the Whittaker-Henderson Method of Graduation,” *TSA,* XXXIV (1982), 329.

Press, William H., et al. *Numerical Recipes in C: The Art of Scientific Computing*. 1992. Retrieved 22 January 2007 from http://www.nrbook.com/a/bookcpdf.php.

# Finite Differences

Since finite differences are no longer on the actuarial syllabus, the references to finite differences may be obtuse for some. The following is a very brief overview of finite differences, just enough to understand how smoothness is defined for Whittaker-Henderson.

Finite differences are useful for discrete functions with arguments at integral values, such as a mortality table, which I will use for examples. The first order difference is technically the slope of the secant from the current value to the next. Thus,



We define higher order differences as the difference of next lower order differences. For example, second differences are



In general,



Finite differences are analogous to derivatives. Just as the derivative of order n+1 is zero for an n-degree polynomial, the finite difference of order n+1 is zero for an n-degree polynomial. For example, if fourth differences are all zero, the values could all be fit to a cubic equation. Thus, when minimizing the 4th differences, the curve with ideal smoothness is a cubic.

R.C.W. (Bob) Howard, FSA, FCIA

[bob@howardfamily.ca](mailto:bob@howardfamily.ca)

2007 01 22, revised 2016 04 17, 2019 05 18