TRANSACTIONS OF SOCIETY OF ACTUARIES
1984 VOL. 36

MULTIDIMENSIONAL WHITTAKER-HENDERSON GRADUATION

FRANK E. KNORR

ABSTRACT

With the increasing ability to maintain better records and larger volumes of data, reports generally display tables of data broken down by many variables. This paper presents a variation of the Whittaker-Henderson graduation method that may be used to graduate such multidimensional data. The main purpose of this paper is to present the underlying theory. No rigorous proofs are presented here since they are easily developed from other proofs available in the literature. Also presented in this paper is another modification of the Whittaker-Henderson method, designed to help the actuary find an acceptable smoothness constant more quickly. Examples are presented to illustrate these two enhancements. A general program that performs the graduation is described in the appendix. Other possible enhancements to the Whittaker-Henderson method are mentioned in an effort to develop a method that is as general as possible.

BACKGROUND

Multidimensional data should be familiar to every actuary. Premiums are determined by factors such as age, plan of insurance, smoker or nonsmoker; financial reports are prepared by line of business, state, accounting period, and other factors; statistical studies show data broken down and summarized by many variables such as age, sex, policy duration, and underwriting classification. Each variable defines a dimension. A multidimensional array of data is merely data organized in a collection of detailed cells. Each cell represents a unique combination of values of those variables. For example, consider the three-dimensional array with the dimensions defined by (1) decennial age groups, (2) sex, and (3) state. This array is merely data organized into 1,000 cells (10 age groups by 2 sexes by 50 states), where all the data in a given cell represent the same decennial age group, same sex, and same state. Data for any other combination of age, sex, and state would be found in another cell.

In many cases the pattern of data across certain dimensions is expected to be smooth. These are the only pertinent dimensions as far as graduation is concerned. These dimensions must be defined by variables whose values have a definite order, such as policy duration (where duration 1 is followed by duration 2, and so forth), or premium rate-up due to physical impairment (where 100–150 percent is between 50–100 percent and 150–200 percent) or age. Data are almost always expected to be smooth across age. Some dimensions defined by variables such as marital status or national origin are difficult to order and, therefore, smoothness across these dimensions is not expected. There are other dimensions where smoothness is not expected even when their values are in a definite order. For example, if one of the variables were state of residence, one would not expect that mortality rates would be smooth across that dimension when the states were arranged in alphabetical order. In other cases, there are variables, such as sex, smoker versus non-smoker, or face amounts less than \$250,000 versus not less than \$250,000, with only two possible values. Even if the two values could be ordered easily, it does not make sense to smooth two bits of data.

If the underlying curve of data across a given dimension is assumed to be smoother than some raw data, and its shape is assumed to be close to a polynomial, then the Whittaker-Henderson graduation method may be used to remove fluctuations due to random errors. The entire graduation process can be summarized in one statement: minimize the value of $F + kS$, where

$F \quad = \sum_{x=1}^{n} W_x (u_x - u_x'')^2$, a measure of fit;

$S \quad = \sum_{x=1}^{n-z} (\Delta^z u_x)^2$, a measure of smoothness;

u_x = The graduated data;

u_x'' = The ungraduated (that is, raw) data;

W_x = A nonnegative number representing the weight to be given to the difference between the graduated and ungraduated values associated with a particular x value;

k = A nonnegative number representing the smoothness constant.

In other words, the graduation process dictates the best possible fit (smallest possible value of F) and the best possible smoothness (smallest S) with the proper balance defined by k. The shape of the underlying data is assumed to be close to a polynomial of degree $z-1$. (For example, if the third-order differences of the graduated data are equal to zero, then the data lie on a quadratic curve.) The magnitude of k determines how close the graduated data will be to the polynomial. The greater the magnitude of k, the more the graduated data resemble the polynomial; the smaller the magnitude of k, the more the graduated data resemble the original data.

SMOOTHNESS CONSTANTS

The selection of proper smoothness constants has long been a concern in the use of the Whittaker-Henderson method ([7], p. 65 and [6], p. 433). There appears to be no relationship between the proper constants for two sets of data unless the two sets of data are very similar. To illustrate this, consider the graduation of mortality rates where the exposures (denoted by E_i) were used as weights. Assume that the proper smoothness constant was found to be 25,000. If the relative amount of exposure were used instead (that is, use $E_i \div \sum_{x=1}^{n} E_x$ instead of E_i as the ith weight), and if the total amount of exposure is 1,000,000, then the proper k would be 0.025. In fact, the two sets of graduated rates would be identical, since minimizing

$$\sum_{x=1}^{n} W_x (u_x - u''_x)^2 + kS \text{ has the same result as minimizing}$$

$$\sum_{x=1}^{n} \frac{W_x}{c} (u_x - u''_x)^2 + \frac{k}{c} S, \text{ where } c \text{ is any positive constant.}$$

The usual approach seeks to minimize the expression $F + kS$ with k being a nonnegative number. If $k = 0$, there is complete emphasis on fit. As k becomes larger, there is increasingly more emphasis on smoothness. Thus, the two extremes of the graduated values are given by k values of 0 and ∞. Another approach is to minimize the expression $(1 - k') F + k'S$ with k' having a possible range of values between 0 and 1. When $k' = 1$, there is complete emphasis on smoothness; when $k' = 0$, there is complete emphasis on fit. Hence, the two extremes of graduation are given by k' values of 0 and 1. This alteration of the function to be minimized in the graduation process is analogous to finding the proper balance by taking some weight off the left side of a scale and adding it to the right side, instead of only adding weight to the right side. The approach taken by this author is to not only limit the range of the smoothness constants to the interval from 0 to 1 but also limit the range of the entire expression to be minimized to the interval from 0 to 1.

The expression to be minimized that this author has adopted is $\frac{1-k}{F_T}F + \frac{k}{S_T}S$, where $0 < k < 1$. The constant F_T is equal to the value of F when the graduated data is totally smooth (that is, all zth differences are equal to zero). F_T is the maximum value of F. This value is determined by assuming that the graduated values lie on the least square polynomial of degree $z - 1$. The constant S_T, which is the maximum value of S, is equal to the value of S when there is complete fit. Since this occurs when the graduated values equal the ungraduated values, the constant S_T is equal to $\sum_{x=1}^{n-z} (\Delta^z u''_x)^2$. The introduction of these two new terms, F_T and S_T, has the effect of standardizing fit and smoothness, since $\frac{F}{F_t}$ ranges from a value of 0 (when no smoothing has taken place) to 1 (when the graduated data is completely smooth) and $\frac{S}{S_T}$ ranges from 1 (with no smoothing) to 0 (with total smoothness). The minimized expression is equal to 1 when there is no smoothness and also when there is total smoothness; in other instances the expression is less than 1. Fit and smoothness are standardized so that k could determine the degree of smoothness relatively consistently for different sets of data.

The modification of the Whittaker-Henderson smoothness constant is illustrated in the following example. Table 1 shows the ungraduated values, which were taken directly from the Part 5 Study Note ([3], p. 59), and the graduated values. Smoothness is measured using second differences and a k value of 0.95. Figure 1 graphically shows the graduated and ungraduated values. Figure 2 shows how the values of F/F_T and S/S_T change as k goes from 0 to 1. As can be seen, the use of a $k = 0.95$ achieves better than 99 percent of the total smoothness possible. The same graduation can be achieved without the modification presented in this section, but the required constant would be 26.25 (that is, $0.95\, F_T - 0.05\, S_T$). The graduated data that appear in the study note can be achieved by the modified formula if the smoothness constant is set equal to 0.99542.

MULTIDIMENSIONAL GRADUATION

When multidimensional graduation is discussed, one application that immediately comes to mind is the graduation of select mortality rates. The two

TABLE 1

SAMPLE DATA, UNGRADUATED AND GRADUATED

x	W_x	u''_x	u_x	$u_x - u''_x$	$W_x(u_x - u''_x)^2$	Δu_x	$\Delta^2 u_x$	$(\Delta^2 u_x)^2$
1.......	3	34	27.16	−6.84	140.357	1.79	.77	.593
2.......	5	24	28.95	4.95	122.512	2.56	.62	.384
3.......	8	31	31.51	.51	2.081	3.18	.31	.096
4.......	10	40	34.69	−5.31	281.961	3.49	2.01	4.040
5.......	15	30	38.18	8.18	1,003.686	5.50	−.96	.922
6.......	20	49	43.68	−5.32	566.048	4.54	.12	.014
7.......	23	48	48.22	.22	1.113	4.66	1.02	1.040
8.......	20	48	52.88	4.88	476.288	5.68	−1.80	3.240
9.......	15	67	58.56	−8.44	1,068.504	3.88	.20	.040
10.......	13	58	62.44	4.44	256.277	4.08	.00	.000
11.......	11	67	66.52	− .48	2.534	.00	.00	.000
Totals . .	. . .	. . . .	. . . .	. . . .	3,921.361	. . . .	. . . .	10.370

NOTE.—$z = 2$, $k = .95$, $F_T = 4{,}649$, $\frac{F}{F_T} = .8433$, $S_T = 3{,}365$, $\frac{S}{S_T} = .0031$.

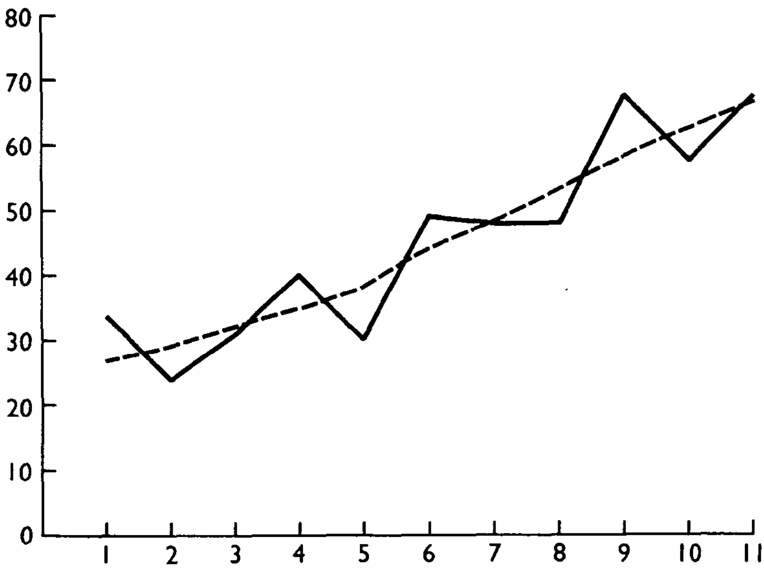
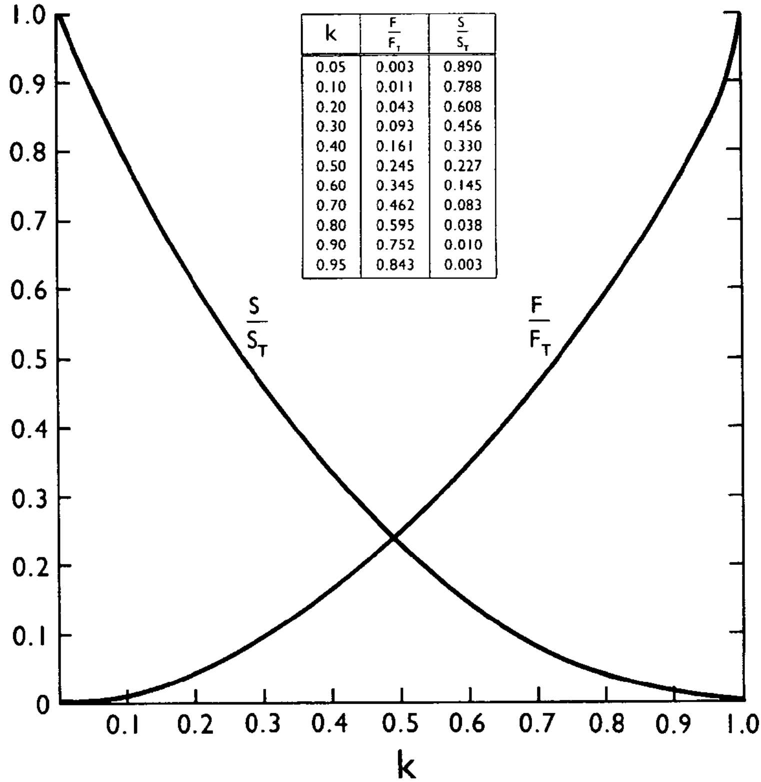


FIG. 1.—Sample data, ungraduated (———) and graduated (– – –).

dimensions in this case are defined by the variables issue age and policy duration. In another application, annuity tables may have mortality rates varying by age and calendar year. The application that prompted the development of the general multidimensional graduation program presented in the



k	$\frac{F}{F_T}$	$\frac{S}{S_T}$
0.05	0.003	0.890
0.10	0.011	0.788
0.20	0.043	0.608
0.30	0.093	0.456
0.40	0.161	0.330
0.50	0.245	0.227
0.60	0.345	0.145
0.70	0.462	0.083
0.80	0.595	0.038
0.90	0.752	0.010
0.95	0.843	0.003

FIG. 2.—Values of F/F_T and S/S_T as k goes from 0 to 1.

appendix is the construction of a disability table for valuation purposes. Disability termination rates vary by age at disablement, duration since disablement, elimination period, and occupational classification. Disability incidence rates vary by age (probably issue age and policy duration, although attained age is generally used), elimination period, and occupational class. These applications will be used to explain and demonstrate the underlying theory.

The concept of graduating a grid of data (a two-dimensional array) was developed by Steven F. McKay and John C. Wilkin and first published as an appendix to an article by Francisco Bayo and John C. Wilkin [1] and is merely expanded here to include higher dimensions and differences other than second differences. The concept of balancing fit and smoothness remains the same as in the one-dimensional case, but fit and smoothness have slightly different definitions for data with D dimensions:

$$F = \sum_{x_1=1}^{n_1} \sum_{x_2=1}^{n_2} \cdots \sum_{x_D=1}^{n_D} W_{x_1,x_2,x_3,\ldots,x_D} (u_{x_1,x_2,\ldots,x_D} - u''_{x_1,x_2,\ldots,x_D})^2,$$

$$\begin{aligned} kS = {} & k_1 \sum_{x_2=1}^{n_2} \sum_{x_3=1}^{n_3} \cdots \sum_{x_D=l}^{n_D} \sum_{x_1=1}^{n_1-z_1} (\underset{1}{\Delta}^{z_1} u_{x_1,x_2,\ldots,x_D})^2 \\ & + k_2 \sum_{x_1=1}^{n_1} \sum_{x_3=1}^{n_3} \cdots \sum_{x_D=1}^{n_D} \sum_{x_2=1}^{n_2-z_2} (\underset{2}{\Delta}^{z_2} u_{x_1,x_2,\ldots,x_D})^2 \\ & + k_3 \sum_{x_1=1}^{n_1} \sum_{x_2=1}^{n_2} \cdots \sum_{x_D=1}^{n_D} \sum_{x_3=1}^{n_3-z_3} (\underset{3}{\Delta}^{z_3} u_{x_1,x_2,\ldots,x_D})^2 \\ & \quad \vdots \\ & + k_D \sum_{x_1=1}^{n_1} \sum_{x_2=1}^{n_2} \cdots \sum_{x_{D-1}=1}^{n_{D-1}} \sum_{x_D=l}^{n_D-z_D} (\underset{D}{\Delta}^{z_D} u_{x_1,x_2,\ldots,x_D})^2, \end{aligned}$$

where $\underset{i}{\Delta}^{z_i}$ is defined by

$$\underset{i}{\Delta}\, u_{x_1,x_2,\ldots,x_i,\ldots,x_D} = u_{x_1,x_2,\ldots,(x_i+1),\ldots,x_D} - u_{x_1,x_2,\ldots,x_i,\ldots,x_D},$$

$$\underset{i}{\Delta}^{z_i} u_{x_1,x_2,\ldots,x_i,\ldots,x_D} = \underset{i}{\Delta}^{z_i-1} u_{x_1,x_2,\ldots,(x_i+1),\ldots,x_D} - \underset{i}{\Delta}^{z_i-1} u_{x_1,x_2,\ldots,x_i,\ldots,x_D}.$$

These formulas may seem overwhelming at first, but they can be explained quite easily. Fit is quantified by taking the difference between the graduated data and the ungraduated data in every cell, squaring each difference, multiplying it by the weight assigned to that cell, and then adding them all up for all the cells in the entire array. For example, if a select mortality table is being graduated with 15 durations and 9 groups of issue ages ($D = 2$, $n_1 = 15$, $n_2 = 9$), then there will be 135 crude rates and 135 graduated rates. The difference between the crude rate and graduated rate for the first policy duration and the first issue age is squared and multiplied by the exposures for that cell. The same thing is done with the data for the first policy duration and the second issue age, and so on for all the cells. The sum of all these is equal to F.

Smoothness is also measured at each cell. However, now there is smoothness in D different directions at each cell. For example, the graduated rate for the fifth policy duration and seventh age group should be smooth relative to the neighboring rates in the fifth policy duration (that is, age groups 6, 8, 5, 9, and so on). It should also be smooth relative to the neighboring graduated rates in the seventh age group (that is, policy durations 4, 6, 3, 7, and so on). Smoothness in each direction may be defined independently. For example, third differences may be minimized across policy duration while second differences are minimized across age. There may also be a different smoothness constant (k_i) for each dimension.

In dealing with multidimensional data, the ravel function found in APL is very helpful. For example, the calculation of F (which involves the summing of squares) becomes a trivial task. The best way to explain how the ravel function works is through an illustration. Consider the following 2 × 3 × 5 matrix (two planes, three rows, and five columns):

Plane 1:

2	2	2	2	2
13	17	19	23	29
11	15	17	21	27

Plane 2:

1	1	1	1	1
2	3	5	7	11
1	2	4	6	10

The result of raveling these data is the following one-dimensional array (vector) with thirty elements:

2 2 2 2 2 13 17 19 23 29 11 15 17 21 27 1 1 1 1 1 2 3 5 7 11 1 2 4 6 10

Notice the order in which the cells were taken: The ravel function loops through the values of the dimensions beginning with the last dimension (column) and ending with the first dimension (plane). In other words, the plane and row are initially held constant and the column position varies. When all the column entries in a particular row have been used, we move to the next row within the plane. When an entire plane has been used, we move onto the next plane and repeat the process. This process is similar to the way the odometer of a car loops through the values 0 through 9 in each position. If the weights and both the graduated and ungraduated data were raveled, F then could be defined as

$$F = \sum_{y=1}^{N} W_y \, (u_y - u''_y)^2,$$

where $N = n_1 \times n_2 \times n_3 \times \ldots \times n_D$ and is the total number of cells in the multidimensional data.

Determining the graduated data amounts to solving the equation $Au = Wu''$. This is the same equation mentioned in the Part 5 Study Note ([3], p. 53), except that the elements are again defined slightly differently and are expressed using the ravel function.

$$A = W + \sum_{i=1}^{D} k_i \, K_i^T K_i \, ;$$

$u = (u_1, u_2, \ldots, u_N)$, the raveled graduated data;

$u'' = (u''_1, u''_2, \ldots, u''_N)$, the raveled ungraduated data;

W = An $N \times N$ matrix with the weights $(w_1, w_2, \ldots, w_N)$ down the diagonal and zeros everywhere else;

K_i = Another $N \times N$ matrix with binomial coefficients that are needed to determine $\Delta_i^{z_i}$;

K_i^T = The transpose of matrix K_i.

The steps required to prove that the equation $Au = Wu''$ is the result when $F + kS$ is minimized parallel the steps in the Part 5 Study Note. This equation has a unique solution for the same reasons.

Similarly, the property of preserving the total number of deaths or terminations, if exposures are used as weights, also applies to the multidimensional graduation. However, this is only valid for the entire set of data. This property does not hold for the number of deaths in all the policy durations of a single issue age group, but the total number of deaths for all age groups and all durations would be preserved.

If, instead of working with the expression $F + kS$, we use the expression $(1-k)\frac{F}{F_T} + k\frac{S}{S_T}$, then the formulas required to produce a multidimensional graduation are as follows:

$$\frac{1-k}{F_T}F = \frac{1-\sum_{i=1}^{D} k_i}{F_T} \sum_{x_1=1}^{n_1} \sum_{x_2=1}^{n_2} \cdots \sum_{x_D=1}^{n_D} W_{x1,x2,\, x3,\, \ldots,\, xD}\,(u_{x1,x2,x3,\, \ldots,\, xD} - u''_{x1,x2,\, x3,\, \ldots,\, xD})^2,$$

$$\frac{k}{S_T}S = \frac{1}{S_T}[k_1 \sum_{x_2=1}^{n_2} \sum_{x_3=1}^{n_3} \cdots \sum_{x_D=1}^{n_D} \sum_{x_1=1}^{n_1-z_1} (\Delta_1^{z1} u_{x1,x2,\, x3,\, \ldots,\, xD})^2$$

$$+ k_2 \sum_{x_1=1}^{n_1} \sum_{x_3=1}^{n_3} \cdots \sum_{x_D=1}^{n_D} \sum_{x_2=1}^{n_2-z_2} (\Delta_2^{z2} u_{x1,x2,\, x3,\, \ldots,\, xD})^2$$

$$+ k_3 \sum_{x_1=1}^{n_1} \sum_{x_2=1}^{n_2} \cdots \sum_{x_D=1}^{n_D} \sum_{x_3=1}^{n_3-z_3} (\Delta_3^{z3} u_{x1,x2,\, x3,\, \ldots,\, xD})^2$$

.

.

.

$$+ k_D \sum_{x_1=1}^{n_1} \sum_{x_2=1}^{n_2} \cdots \sum_{x_{D-1}=1}^{n_{D-1}} \sum_{x_D=l}^{n_D-z_D} (\Delta_D^{zD}\, u_{x1,x2,\, \ldots,\, xD})^2],$$

where $0 < k_i < 1$ for all k_i's and $0 < \sum_{i=1}^{D} k_i < 1$; and F_T is the maximum value of F; and S_T is the maximum value of S.

Determining the graduated data amounts to solving the equation

$$Au = [(1 - \sum_{i=1}^{D} k_i)/F_T]\, Wu''.$$

$$A = \frac{1 - \sum_{i=1}^{D} k_i}{F_T} W + \frac{1}{S_T} \sum_{i=1}^{D} k_i K_i^T K_i \ ;$$

$u = (u_1, u_2, \ldots, u_N)$, the raveled graduated data;

$u'' = (u''_1, u''_2, \ldots, u''_N)$, the raveled ungraduated data;

W = An $N \times N$ matrix with the weights $(w_1, w_2, \ldots, w_N)$ down the diagonal and zeros everywhere else;

K = Another $N \times N$ matrix with binomial coefficients that are needed to determine $\Delta_i^{z_i}$;

K_i^T = The transpose of matrix K_i.

EXAMPLES

The extension of the Whittaker-Henderson graduation technique to higher dimensions, using the modification of the smoothness constant, is illustrated by the following examples. The data used in these examples were taken from the *1979 Reports* [8]. Although actual data were used, the analysis needed to perform the proper multidimensional graduation was not carried out. These examples are meant only to be illustrations of the graduation method.

Select Mortality Rates

The two dimensions represent policy duration and issue age group. The data represent select mortality rates of nonmedical males between 1977 and 1978 anniversaries ([8], pp. 40–45). Third differences are minimized across each dimension and smoothness constants of 0.199 and 0.8 were used. The exposures, which were used as weights, as well as the crude rates and graduated rates are shown in table 2. Figure 3 shows the crude rates and graduated rates for three of the nine issue age groups. Including all nine groups in one graph would have produced a graph too complicated to understand.

TABLE 2

STANDARD NONMEDICAL ISSUES OF 1963–77

MALES LIVES

EXPERIENCE BETWEEN 1977 AND 1978 ANNIVERSARIES

POLICY DURATION	ISSUE AGE								
	5–9	10–14	15–19	20–24	25–29	30–34	35–39	40–44	45–49
	Exposed to Risk (in $1,000s)								
1	432	449	1,613	4,718	4,493	1,806	471	113	36
2	339	380	1,321	3,521	3,396	1,297	365	93	46
3	263	320	1,131	2,889	2,697	1,053	327	86	30
4	256	345	1,216	2,822	2,308	884	284	74	25
5	249	362	1,187	2,653	2,056	817	280	80	18
6	208	307	989	2,443	1,806	715	262	66	12
7	148	219	800	2,225	1,537	644	245	60	11
8	121	183	736	2,009	1,339	579	233	57	7
9	109	174	654	1,669	1,226	546	216	52	6
10	107	170	644	1,535	1,171	533	208	50	6
11	105	166	621	1,258	981	447	162	34	4
12	103	165	607	1,042	839	374	139	25	3
13	99	161	683	1,069	803	375	145	25	2
14	94	149	541	968	765	369	154	25	2
15	85	137	486	839	691	346	147	24	2
	Crude Mortality Rates (per 1,000)								
1	.24	.32	1.08	.76	.61	.60	1.26	1.84	2.11
2	.23	.24	1.00	.68	.64	.79	1.11	2.52	.26
3	.17	.66	1.29	.72	.88	1.04	1.30	1.66	8.43
4	.24	.62	1.09	.82	.80	1.11	1.67	1.85	9.36
5	.34	.98	1.25	.78	.74	1.19	2.29	3.10	2.28
6	.39	1.22	1.16	.76	.70	1.54	2.29	3.88	4.42
7	.47	1.12	1.18	.81	.82	.97	2.65	4.85	2.09
8	.54	.98	1.24	.83	.95	1.60	2.60	4.91	1.43
9	.50	1.41	1.14	.86	.99	1.59	3.51	4.88	6.17
10	.96	1.11	.79	.87	1.08	1.84	3.72	6.80	1.50
11	.47	1.34	1.00	1.06	1.06	2.09	3.81	5.65	6.75
12	1.28	.85	.97	.99	1.10	1.86	3.00	7.28	7.00
13	.98	1.48	1.08	.85	1.30	2.95	5.83	7.00	13.00
14	1.16	.91	1.31	1.12	1.76	2.86	5.30	7.48	6.50
15	1.15	1.09	1.22	1.39	1.82	3.43	5.71	9.00	11.00

NOTE.—$D = 2$, $N = 135$, $n_1 = 9$, $n_2 = 15$, $z_1 = 3$, $z_2 = 3$, $k_1 = .199$, $k_2 = .8$.

TABLE 2—*Continued*

	Graduated Mortality Rates (per 1,000)								
1......	.35	.66	.78	.74	.66	.70	.92	1.31	1.87
2......	.42	.67	.77	.74	.73	.85	1.20	1.81	2.68
3......	.53	.73	.81	.79	.82	1.01	1.46	2.25	3.41
4......	.68	.82	.85	.82	.87	1.13	1.70	2.65	3.99
5......	.86	.91	.88	.81	.88	1.22	1.92	2.99	4.44
6......	1.04	.97	.87	.80	.90	1.32	2.12	3.29	4.84
7......	1.18	1.01	.87	.81	.96	1.44	2.33	3.60	5.25
8......	1.29	1.02	.86	.83	1.04	1.61	2.58	3.96	5.72
9......	1.37	1.02	.83	.84	1.13	1.79	2.87	4.37	6.29
10.....	1.44	1.02	.80	.84	1.20	1.96	3.18	4.84	6.96
11.....	1.52	1.03	.80	.86	1.27	2.13	3.50	5.36	7.72
12.....	1.61	1.05	.80	.89	1.38	2.36	3.89	5.96	8.57
13.....	1.69	1.08	.83	.96	1.55	2.69	4.39	6.65	9.48
14.....	1.73	1.12	.90	1.11	1.81	3.08	4.94	7.40	10.45
15.....	1.74	1.15	1.00	1.30	2.11	3.51	5.53	8.19	11.48

NOTE.—$D = 2$, $N = 135$, $n_1 = 9$, $n_2 = 15$, $z_1 = 3$, $z_2 = 3$, $k_1 = .199$, $k_2 = .8$.

Disability Termination Rates

The three dimensions represent elimination period, age at disablement, and duration since disablement. The data represent rates of termination from disability, and the weights are the exposures to termination ([8], pp. 371–97). Since nine-month elimination periods do not exist, the data and weights were set equal to zero. This was done to make the values of elimination period equidistant. The weights, crude rates, and graduated rates are shown in table 3A. The smoothness of the graduated rates is shown numerically in table 3B. Since the set of graduated termination rates is a three-dimensional array, smoothness is measured in three different directions. Figure 4 presents the ungraduated and graduated data graphically, showing smoothness across age. Graphs also could be constructed showing smoothness across duration, by drawing one line for each age-elimination period combination. The smoothness across elimination period can be shown similarly.

CONSIDERATIONS FOR ADDITIONAL ENHANCEMENTS

An attempt was made to have the APL program shown in the appendix as general as possible for the Whittaker-Henderson method presented here.

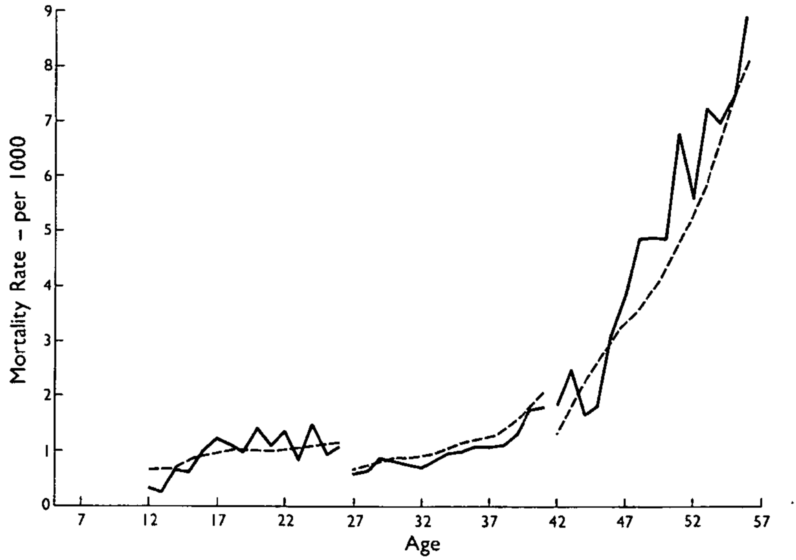


FIG. 3.—Standard nonmedical issues of 1963–77; male lives; experience between 1977 and 1978 anniversaries; ungraduated (———) and graduated (– – –).

The method, on the other hand, may be altered to allow for further variations. The following are some possible enhancements that come to mind.

1. The definition of F and S should not be limited to the sums of squares. Sums of absolute values may be one alternative. This was suggested by Donald R. Schuette ([6], p. 408). Also sums of positive square roots or sums of the absolute values of cubes should be available.
2. Smoothness may be redefined. Instead of using zth differences, a combination of differences has been suggested by Walter B. Lowrie in a recent paper in the *Transactions* ([5]). Using divided differences has also been suggested when the ordered values of a variable are not equidistant, as in the cases of graduating across elimination period where the values are 0, 7, 14, 30, 90, and 180 days. Perhaps smoothness could be defined in more general terms for cases where the underlying curves are neither polynomials, exponentials, nor close to either of these. If this is possible, then the definition of smoothness across one dimension should be independent of its definition across any other dimension.
3. Smoothness constants could be allowed to vary within each dimension

TABLE 3A

GROUP LONG-TERM DISABILITY INSURANCE TERMINATION EXPERIENCE OF 1962–77

DURATION	AGE AT DISABLEMENT				
	Under 30	30–39	40–49	50–59	60–64
	Exposures				
3-month elimination period:					
2........	256	504	1,374	3,674	1,633
3........	121	254	900	2,510	927
4........	63	146	583	1,756	394
5........	37	108	385	1,221	157
6........	34	59	274	695	0
6-month elimination period:					
2........	676	1,395	4,197	11,005	5,178
3........	314	816	2,823	7,998	3,135
4........	192	527	2,120	6,089	1,564
5........	127	377	1,631	4,642	667
6........	95	276	1,178	3,279	195
9-month elimination period:					
2........	0	0	0	0	0
3........	0	0	0	0	0
4........	0	0	0	0	0
5........	0	0	0	0	0
6........	0	0	0	0	0
12-month elimination period:					
2........	74	158	521	1,504	542
3........	42	76	330	965	260
4........	24	45	233	681	147
5........	9	34	163	508	0
6........	5	23	130	333	22

NOTE.—$D = 3$, $N = 100$, $n_1 = 4$, $n_2 = 5$, $n_3 = 5$, $z_1 = 2$, $z_2 = 3$, $z_3 = 3$, $k_1 = .1$, $k_2 = .29$, $k_3 = .59$.

and behave as weights of smoothness. This means rewriting each term of S from

$$k_i \sum_{x_1=1}^{n_1} \sum_{x_2=1}^{n_2} \cdots \sum_{x_{i-1}=1}^{n_{i-1}} \sum_{x_{i+1}=1}^{n_{i+1}} \cdots \sum_{x_D=1}^{n_D} \sum_{x_i=1}^{n_i-z_i} (\underset{i}{\Delta}^{z_i} u_{x_1,x_2,x_3,\ldots,x_D})^2$$

TABLE 3A—*Continued*

Duration	Age at Disablement: Under 30	30–39	40–49	50–59	60–64
	Crude Rates of Termination				
3-month elimination period:					
2	.3637	.2915	.2053	.1369	.1096
3	.1976	.1966	.1333	.0964	.1025
4	.0638	.0956	.0943	.0672	.0761
5	.1070	.1021	.0468	.0639	.0637
6	.0294	.0510	.0329	.0834	.0000
6-month elimination period:					
2	.3535	.2703	.1725	.1224	.1041
3	.2835	.1790	.1176	.0864	.0775
4	.1250	.1062	.0698	.0675	.0729
5	.0630	.0690	.0521	.0629	.0495
6	.0631	.0398	.0467	.0677	.0719
9-month elimination period:					
2	.0000	.0000	.0000	.0000	.0000
3	.0000	.0000	.0000	.0000	.0000
4	.0000	.0000	.0000	.0000	.0000
5	.0000	.0000	.0000	.0000	.0000
6	.0000	.0000	.0000	.0000	.0000
12-month elimination period:					
2	.2148	.2537	.1652	.1170	.1217
3	.1680	.1180	.1212	.0860	.0885
4	.2552	.0897	.1028	.0955	.0477
5	.1080	.0884	.0307	.0729	.0000
6	.3896	.0430	.0461	.0780	.0455

NOTE.— $D = 3$, $N = 100$, $n_1 = 4$, $n_2 = 5$, $n_3 = 5$, $z_1 = 2$, $z_2 = 3$, $z_3 = 3$, $k_1 = .1$, $k_2 = .29$, $k_3 = .59$.

to

$$\sum_{x_1=1}^{n_1} \sum_{x_2=1}^{n_2} \cdots \sum_{x_{i-1}=1}^{n_{i-1}} \sum_{x_{i+1}=1}^{n_{i+1}} \cdots \sum_{x_D=1}^{n_D} k_{x_1, x_2, \ldots, x_{i-1}, x_{i+1}, \ldots, x_D} \sum_{x_i=1}^{n_i - z_i}$$

$$(\Delta_i^{z_i} u_{x_1, x_2, x_3, \ldots, x_D})^2$$

Rewriting S in this manner can be incorporated in a program fairly easily. This, by the way, would not destroy the property of the Whittaker-Henderson method that preserves the total number of deaths or terminations.

TABLE 3A—*Continued*

Duration	Age at Disablement: Under 30	30–39	40–49	50–59	60–64
	Graduated Rates of Termination				
3-month elimination period:					
2.........	.3696	.2875	.2043	.1378	.1093
3.........	.2140	.1809	.1338	.0975	.0976
4.........	.1116	.1043	.0873	.0698	.0745
5.........	.0614	.0605	.0570	.0624	.0716
6.........	.0422	.0346	.0464	.0778	.1109
6-month elimination period:					
2.........	.3595	.2651	.1746	.1221	.1040
3.........	.2619	.1764	.1175	.0863	.0787
4.........	.1470	.1038	.0736	.0669	.0701
5.........	.0716	.0603	.0548	.0617	.0586
6.........	.0540	.0423	.0490	.0667	.0769
9-month elimination period:					
2.........	.3221	.2344	.1667	.1225	.1044
3.........	.2325	.1638	.1157	.0876	.0792
4.........	.1590	.1104	.0801	.0667	.0681
5.........	.1100	.0762	.0593	.0580	.0698
6.........	.0926	.0625	.0524	.0605	.0847
12-month elimination period:					
2.........	.2491	.2191	.1644	.1194	.1174
3.........	.1907	.1510	.1190	.0921	.0820
4.........	.1596	.1114	.0916	.0841	.0690
5.........	.1479	.0886	.0645	.0698	.0823
6.........	.1604	.0810	.0532	.0676	.1084

NOTE.— $D = 3, N = 100, n_1 = 4, n_2 = 5, n_3 = 5, z_1 = 2, z_2 = 3, z_3 = 3, k_1 = .1, k_2 = .29, k_3 = .59$.

CONCLUSION

The method for graduating a grid of data was expanded for a greater number of dimensions and for differences other than second differences. The general program presented here performs the Whittaker-Henderson graduation on multidimensional data as well as the more familiar one-dimensional data. This program also incorporates a new approach to smoothness constants, which limits them to values between zero (no smoothness) and one (ultimate smoothness). The program may be rendered useless if the actuary does not have access to an APL system with a workspace large enough for the graduation. In spite of this limitation, any actuary involved in graduating data should be aware that multidimensional data can be graduated using this enhancement of the Whittaker-Henderson method. Further enhancements are also possible and should be encouraged.

TABLE 3B

GROUP LONG-TERM DISABILITY INSURANCE
TERMINATION EXPERIENCE OF 1962–77
MEASURES OF SMOOTHNESS

DURATION	AGE AT DISABLEMENT				
	Under 30	30–39	40–49	50–59	60–64
	Smoothness across Elimination Period $\underset{1}{\Delta}^2 u_{x_1,x_2,x_3})$				
3-month elimination period:					
2	−.0273	−.0083	.0219	.0161	.0057
3	−.0773	−.0081	.0145	.0125	.0194
4	−.0234	.0072	.0202	.0027	.0024
5	.0282	.0161	.0067	−.0030	.0242
6	.0268	.0125	.0006	.0049	.0418
6-month elimination period:					
2	−.0356	.0154	.0056	−.0034	.0126
3	−.0124	−.0002	.0051	.0032	.0023
4	−.0114	−.0056	.0050	.0176	.0029
5	−.0005	−.0035	.0007	.0155	.0013
6	.0292	−.0016	−.0025	.0133	.0159
	Smoothness across Duration $\underset{2}{\Delta}^3 u_{x_1,x_2,x_3})$				
3-month elimination period:					
2	−.0010	.0025	−.0079	.0077	.0316
3	−.0212	−.0146	.0035	.0025	.0220
6-month elimination period:					
2	.0568	.0130	.0119	−.0022	−.0196
3	.0183	−.0036	−.0120	−.0040	.0327
9-month elimination period:					
2	.0084	.0020	−.0006	−.0018	−.0013
3	.0071	.0013	−.0009	−.0010	.0004
12-month elimination period:					
2	−.0079	−.0117	−.0177	−.0257	.0039
3	.0048	−.0015	.0155	.0184	−.0135

APPENDIX A

PROGRAM

Since the program is lengthy, it is presented here one logical step at a time. The length of the program would be reduced from seventy-three statements to thirty-four if all checks for the data's validity were left out.

APL was chosen as the programming language for a number of reasons. First of all, it is well suited to multidimensional data. This comes in handy for the construction of the *K* matrices. Second, the matrix divide (⌹) makes the final step of the graduation very concise and eliminates the need for

TABLE 3B—*Continued*

DURATION	AGE AT DISABLEMENT: Under 30	AGE AT DISABLEMENT: 30–39	DURATION	AGE AT DISABLEMENT: Under 30	AGE AT DISABLEMENT: 30–39
	Smoothness across Age $(\Delta_3^3 u_{x_1,x_2,x_3})$			Smoothness across Age $(\Delta_3^3 u_{x_1,x_2,x_3})$	
3-month elimination period:			9-month elimination period:		
2	.0175	.0216	2	.0035	.0026
3	.0248	.0256	3	−.0006	−.0003
4	.0095	.0226	4	−.0014	−.0021
5	.0115	−.0051	5	−.0013	−.0025
6	.0002	−.0179	6	−.0018	−.0021
6-month elimination period:			12-month elimination period:		
2	.0341	−.0036	2	.0345	.0330
3	.0011	−.0041	3	−.0026	.0117
4	.0105	−.0136	4	−.0161	−.0199
5	.0066	−.0224	5	−.0058	−.0222
6	−.0077	−.0182	6	−.0091	−.0159

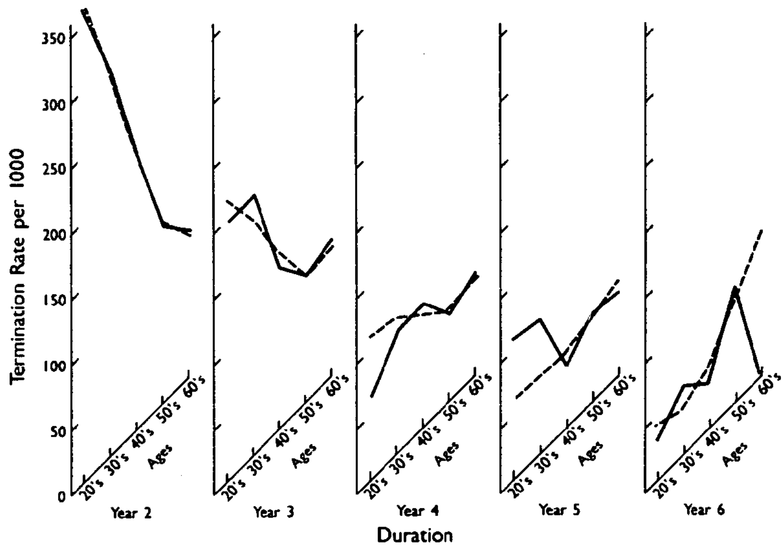


FIG. 4.—Group long-term disability insurance, termination experience of 1962–77, ungraduated (———) and graduated (– – –). A. three-month elimination period.

using the Choleski method. Also, APL is becoming the accepted standard in the actuarial community.

The main drawback of this program is the large amount of storage required

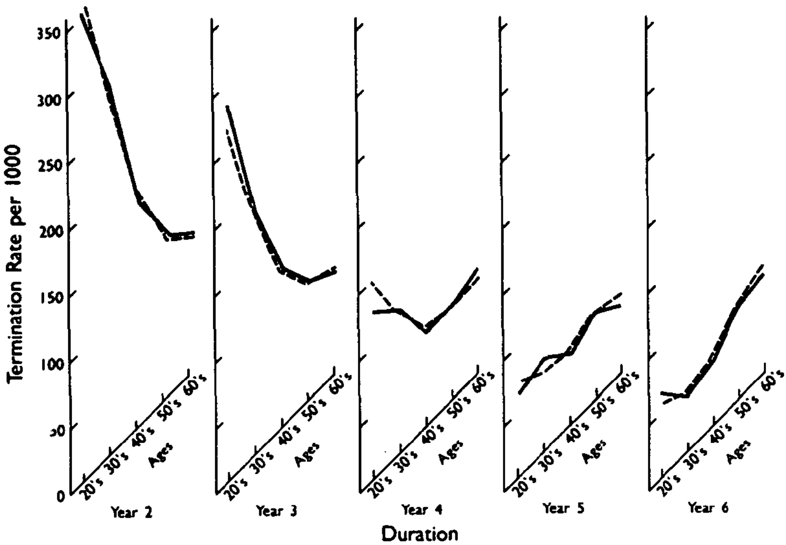


FIG. 4—B. Six-month elimination period.

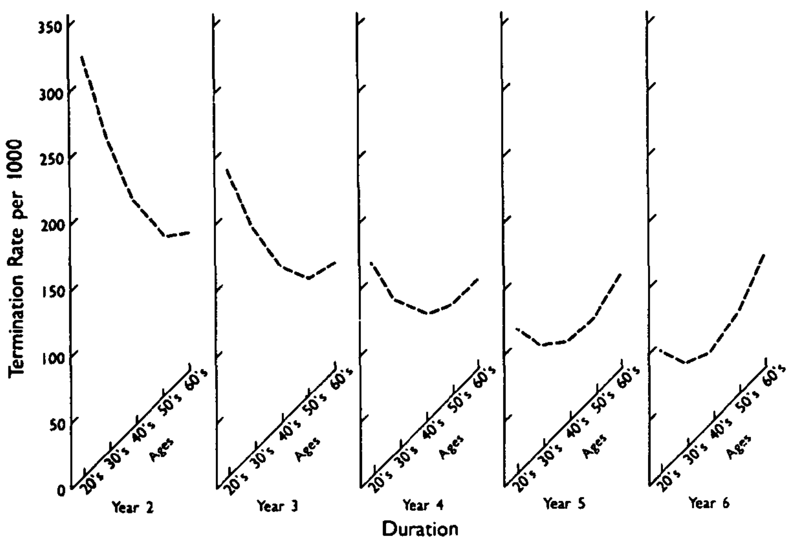


FIG. 4—C. Nine-month elimination period.

for the graduation of multidimensional data. One method of conserving storage is to store only the diagonal band of nonzero matrix elements. However,

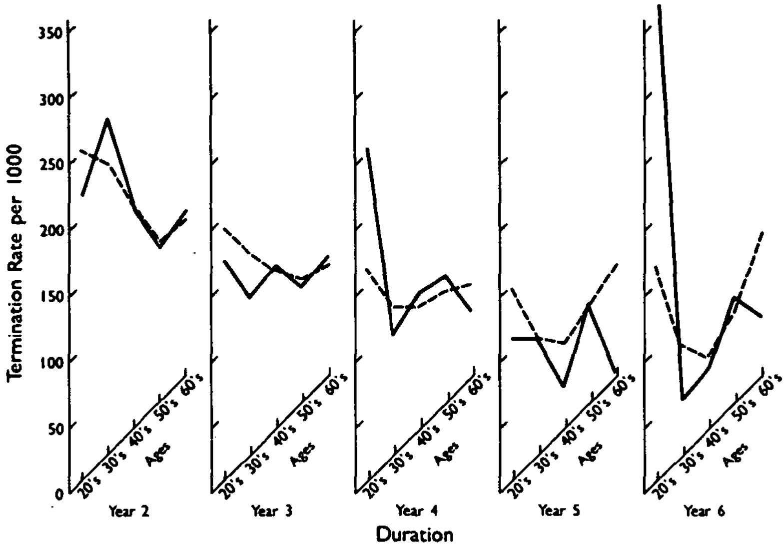


FIG. 4—D. Twelve-month elimination period.

additional dimensions expand the width of the band significantly making that method no longer practical. Also, the additional programming required would be very confusing.

This APL function returns the array of graduated data. It requires a left argument (the weights), a right argument (the ungraduated data), and two variables (*DIFF* and *SMOOTH*) that must be defined before the function is called. The variable *DIFF* must contain the differences that are to be minimized for each dimension. The variable *SMOOTH* must contain the smoothness constants for each dimension. The sum of these smoothness constants may not be greater than one.

```
[0]    U←WEIGHTS GRAD UNGRAD;D;N;N;Z;K;A;K;W;ET;ST;I;NOTI
```

Once *DIFF* and *SMOOTH* are defined, the graduation function, *GRAD*, may be called to display the graduated data. It may also be called by another APL function where the graduated data can be used for further processing. This graduation function requires two arguments. The left argument is the set of weights. The right argument is the ungraduated data. The data may have any number of dimensions, but the dimensions of the weights must match the dimensions of the ungraduated data.

The local variables that will not be saved after the function has been executed are listed after the right argument, *UNGRAD*. The names given the APL variables were designed to correspond with the names of the varibles used in the formulas presented in this paper.

Formulas	APL	Meaning
D	$\underline{D}$	Number of dimensions (variables)
N	$\underline{N}$	Number of cells
n and n_i	N and $N\ [I]$	Number of possible values for each variable
z and z_i	Z and $Z\ [I]$	Differences minimized
k and k_i	K and $K\ [I]$	Smoothness constants
u''	*UNGRAD*	Ungraduated array of data
u	U	Graduated array of data
$W_{x_1x_2,\ldots,x_D}$	*WEIGHTS* [; ; . . .;]	Array of weights
K	$\underline{K}$	$N \times N$ matrix of binomial coefficients
A	$\underline{A}$	$\dfrac{(1-\sum_{i=1}^{D} k_i)}{F_T} W + \dfrac{1}{S_T}\sum_{i=1}^{D} k_i K_i^T K_i$
W	$\underline{W}$	$N \times N$ matrix of weights
F_T	$\underline{FT}$	Maximum value of F
S_T	$\underline{ST}$ (actually $+/\underline{ST}$)	Maximum value of S

```
[1]   TESTRANK:→((ρρWEIGHTS)=ρρUNGRAD)/TESTSHAPE
[2]   'RANK OF WEIGHTS DO NOT EQUAL RANK OF UNGRADUATED DATA'
[3]   'NO GRADUATION WIL BE PERFORMED'
[4]   U←UNGRAD
[5]   →END
```

A test is made to be sure that the weights and the ungraduated data have the same number of dimensions, that is, the same rank. If they are not the same, then an error message is typed and the original ungraduated data are returned in place of the graduated data.

```
[6]   TESTSHAPE:→(∧/(ρWEIGHTS)=ρUNGRAD)/TESTWEIGHT
[7]   'SHAPE OF WEIGHTS DO NOT MATCH SHAPE OF UNGRADUATED DATA'
[8]   'NO GRADUATION WILL BE PERFORMED'
[9]   U←UNGRAD
[10]  →END
```

A test is made to be sure each dimension of the ungraduated data matches the corresponding dimension of the weights. If any one of them does not match, an error message is typed and the original data are returned.

```
[11]  TESTWEIGHT:→((∧/,WEIGHTS≥0)∧((+/,WEIGHTS)>0))/TESTDIFF
[12]  'WEIGHTS MAY NOT BE NEGATIVE AND MUST HAVE AT LEAST ONE POSITIVE VALUE'
[13]  'NO GRADUATION WILL BE PERFORMED'
[14]  U←UNGRAD
[15]  →END
```

A test is made to be sure that there are no negative weights and that there is at least one weight that is greater than zero. If either of these is not true, an error message is typed and the original data are returned.

```
[16]  TESTDIFF:→((ρ,DIFF)=ρρUNGRAD)/ADJDIFF
[17]  '''DIFF'' NOT DEFINED PROPERLY'
[18]  'ONE VALUE FOR EACH DIMENSION'
[19]  'NO GRADUATION WILL BE PERFORMED'
[20]  U←UNGRAD
[21]  →END
```

A test is made to be sure that there is one difference to be minimized for each dimension. If the number of differences does not match the number of dimensions, then an error message is typed and the original data are returned.

```
[22]  ADJDIFF:→((∧/DIFF≥1)∧(∧/DIFF≤ρUNGRAD)∧(∧/DIFF=⌊DIFF))/TESTSMOOTH
[23]  '''DIFF'' NOT DEFINED PROPERLY'
[24]  'ADJUSTED FROM ',(⍕DIFF),' TO ',⍕1⌈(⌊DIFF)⌊ρUNGRAD
[25]  'GRADUATION WILL CONTINUE'
```

If any one of the differences is less than one or greater than the number of possible values in its corresponding dimension or is not an integer, then a message is typed, and the differences are corrected and processing continues.

```
[26]  TESTSMOOTH:→((∧/(,SMOOTH)>0)∧((+/,SMOOTH)≤1)∧((ρ,SMOOTH)=ρρUNGRAD))/TOTAL
      SMOOTHNESS
[27]  'SMOOTHNESS FACTORS NOT DEFINED PROPERLY'
[28]  'ALL FACTORS MUST BE GREATER THAN ZERO'
[29]  'THE SUM OF THE FACTORS MUST BE LESS THAN ONE'
[30]  'ONE FACTOR FOR EACH DIMENSION'
[31]  'NO GRADUATION WILL BE PERFORMED'
[32]  U←UNGRAD
[33]  →END
```

The smoothness constants are tested to be sure they are all greater than zero, their sum is not greater than one, and there is one smoothness constant for each dimension. If any one of these is not true, an error message is typed and the original data are returned.

```
[34]  TOTALSMOOTHNESS;Z←(1⌈(⌊DIFF)⌊⍴UNGRAD)-1
[35]  U←WEIGHTS LSQ UNGRAD
```

The multidimensional least squares data are determined with z_i defined to be the degree of the polynomial across the *i*th dimension. Once the APL variable *Z* is defined, the least-squares function (*LSQ*) can be called similar to the *GRAD* function. This function contains thirty statements and is shown in appendix B. The data that are returned by *LSQ* are data with the ultimate smoothness. In other words, this is the limit of the graduated data as the smoothness constants approach one.

```
[36]  →((+/,SMOOTH)=1)/END
[37]  E̲T←+/,WEIGHTS×(U-UNGRAD)*2
[38]  →(E̲T>0)/INTIALIZE
[39]  'DATA ALREADY SMOOTH'
[40]  →END
```

If the sum of the smoothness constants is equal to one or if the ungraduated data are already smooth, no more processing is required. The rest of the program is bypassed and the smooth data are returned.

```
[41]  INTIALIZE:
[42]  D̲←⍴⍴UNGRAD
[43]  N←⍴UNGRAD
[44]  N̲←⍴,UNGRAD
[45]  Z←Z+1
[46]  K←,SMOOTH
```

Most of the variables are initialized here.

```
[47]  A̲←(N̲,N̲)⍴0
[48]  S̲T←⍳0
[49]  I←0
[50]  LOOP:I←I+1
[51]  K̲←((⍳Z[I])!Z[I])×(¯1)*⍳Z[I]
[52]  K̲←N[I]↑⌽1,K̲
[53]  K̲←((N[I]-Z[I]),N[I])⍴K̲
[54]  K̲←(N[I],N[I])↑K̲
[55]  NOTI←(I≠⍳D̲)/⍳D̲
```

```
[56]  K←((×/N[NOTI]),N[I],1,N[I])⍴K
[57]  K←((×/N[NOTI]),N[I],(×/N[NOTI]),N[I])↑K
[58]  K←(N,N)⍴K
[59]  K←(1-⍳N)⌽K
[60]  K←(N[NOTI],N[I],N[NOTI],N[I])⍴K
[61]  K←((NOTI,I),D+NOTI,I)⍉K
[62]  K←(N,N)⍴K
[63]  ST←ST,(,UNGRAD)+.×(⍉K)+.×K+.×(,UNGRAD)
[64]  K←(⍉K)+.×K
[65]  K←K[I]×K
[66]  A←A+K
[67]  →(I<D)/LOOP
```

This loop is required to produce a K array for each dimension. These are needed in the development of the A array since

$$A = \frac{(1 - \sum_{i=1}^{D} k_i}{F_T} W + \frac{1}{S_T} \sum_{i=1}^{D}, k_i K_i^T K_i.$$

Each K array is used to define the smoothness along the given dimension. Since smoothness is defined in terms of differences, binomial coefficients are needed. In other words, since

$$\Delta^3 u_x = 1u_{x+3} - 3u_{x+2} + 3u_{x+1} - 1u_x,$$

and, in general,

$$\Delta^z u_x = \binom{z}{z} u_{x+z} - \binom{z}{z-1} u_{x+z-1} + \binom{z}{z-2} u_{x+z-2} + \ldots + (-1)^z \binom{z}{0} u_x$$

the binomial coefficients $-1, 3, -3, 1$ and, in general,

$$(-1)^z \binom{z}{0}, (-1)^{z-1} \binom{z}{1}, (-1)^{z-2} \binom{z}{2}, \ldots, \binom{z}{z}$$

are needed. Positioning these coefficients properly in the $N \times N$ array is the reason for the major processing effort. The proper position is determined by the dimension and by the number of possible values in the other dimensions. In one case the difference is taken between an element and the next element of the raveled data; in this case the coefficients are positioned next to each other in the K array. In another case the difference is taken between an element and another element n_D places away in the raveled data. In this case the coefficients are separated by $n_D - 1$ zeros. There is a different case for each dimension. In the last case the difference is taken between an element and another one $n_D \times n_{D-1} \times \ldots \times n_3 \times n_2$ places away; in this case the coefficients are separated by $(n_D \times n_{D-1} \times \ldots \times n_3 \times n_2) - 1$ zeros.

The smoothness constant for a given dimension is applied to the matrix product of K^T and K.

```
[68]  A←A÷+/ST
```

The A array is adjusted by S_T, the standardizing value for smoothness.

```
[69]  ⎕EX 'K'
```

The final K array is erased to make more storage available for the W array.

```
[70]  W←(N,N)⍴(,((⍳N)∘.=⍳N))\,WEIGHTS×(1−+/K)÷FT
```

W is constructed as an $N \times N$ array whose elements are zeros everywhere except down the diagonal. The diagonal elements are the ravel weights adjusted by (a) one minus the sum of the smoothness constants and (b) F_T, the standardizing value for fit.

```
[71]  A←A+W
```

The W array is added to complete the A array.

```
[72]  GRADUATION:U←N⍴(W+.×,UNGRAD)⌹A
```

The graduation is performed by solving the equation $Au = Wu''$ for u. This is possible if and only if the A array has an inverse. Since we have constructed the A array to be a positive definite matrix, an inverse is assured.

```
[73]  END:
```

Processing ends and the data in the APL variable U are returned.

APPENDIX B

THE LEAST SQUARES FUNCTION

```
C←W LSQ FX;X;I;A;R;COEF;XI
⍝
⍝ INITIALIZING DATA
⍝
N̲←×/N←⍴FX
W←(N̲,N̲)⍴(,(⍳N̲)∘.=⍳N̲)\,W
Z←,Z
X← 1 1 ⍴1
⍝
⍝ LOOPING THROUGH EACH DIMENSION
⍝ TO SET UP X ARRAY - ASSUME ALL DATA IS EQUIDISTANT
⍝
I←0
LOOP:I←I+1
XI←(⍳N[I])∘.*0,⍳Z[I]
X←(((⍴X)[1]×N[I]),(⍴X)[2]×1+Z[I])⍴ 1 3 2 4 ⍉X∘.×XI
→(I<⍴⍴FX)/LOOP
⍝
⍝ TWO ARRAYS MUST BE SET UP
⍝
A←(⍉X)+.×W+.×X
R←(W+.×,FX)+.×X
⍝
⍝ DETERMINE COEFFICIENTS FOR THE LEAST SQUARES POLYNOMIAL
⍝
COEF←R⌹A
⍝
⍝ USE THE COEFFICIENTS TO CALCULATE THE VALUE OF
⍝ THE POLYNOMIAL AT EVERY POINT
⍝
C←N⍴X+.×COEF
```

REFERENCES

1. BAYO F., and WILKIN, J. C. *Experience of Disabled Worker Benefits under OASDI 1965–74*. Actuarial Study No. 74, DHEW Publication No. (SSA) 77–11521, 1977.
2. COLEMAN, JAMES S. *Introduction to Mathematical Sociology*. Toronto: Collier-Macmillan Canada, 1964.
3. GREVILLE, T. N. E. *Part 5 Study Notes: Graduation*. Chicago: Society of Actuaries, 1974.
4. HENDERSON, ROBERT. *Graduation of Mortality and Other Tables*. New York: Actuarial Society of America, 1919.
5. LOWRIE, WALTER B. "An Extension of the Whittaker-Henderson Method of Graduation," *TSA*, XXXIV (1982), 329.
6. SCHUETTE, DONALD R. "A Linear Programming Approach to Graduation," *TSA*, XXX (1978), 407.
7. SPOERL, CHARLES A. "Actuarial Note," *TASA* XXXII (1931), 60.
8. *TSA 1979 Reports* (1980), p. 341.

DISCUSSION OF PRECEDING PAPER

STEVEN F. MCKAY:

Mr. Knorr has done a commendable job in expanding the usefulness of multidimensional Whittaker-Henderson graduation. As he pointed out in his paper, we developed the concept of graduating a grid of data here at the Social Security Administration about ten years ago. In our applications, we saved a considerable amount of computer storage by using the band structure of the matrices involved in the calculations. Such techniques should be expandable to higher dimensions. I suggest that anyone having problems with limited computer storage when using Mr. Knorr's approach should look into the possibility of using the band structure.

I believe any discussion of advances in Whittaker-Henderson graduation techniques should give credit to Dr. T. N. E. Greville, whose lucid Part 5 Study Note was a great help to a decade of actuarial students.

JEFFREY L. KUNKEL:

Mr. Knorr's paper is a useful addition to the literature on graduation. I particularly like his idea of standardizing fit and smoothness. My discussion is limited to the author's justification for the large amount of storage required by his APL program used to implement his generalization of Whittaker-Henderson graduation.

Mr. Knorr contends that taking advantage of the band structure of the matrix A, as described in his article, is impractical because "additional dimensions expand the width of the band significantly." While it is true that the band width is significantly expanded with an increase in the number of dimensions, one should recall that A is a symmetric matrix. Thus, only slightly more than half of the band (the main diagonal and the nontrivial diagonals above it) needs to be stored. The width of the band will depend on the particular function used to transform the ungraduated array into a vector. It will also depend on the degree of differencing associated with that dimension for which the transformation requires the largest "jumps" within the resulting vector when moving from one position to the next. Let me clarify this last statement with an example. Suppose we have a three-dimensional array with N rows, M columns, and L layers. We choose to map the (i, j, k) position in the array to the $(k - 1)NM + (i - 1)N + j$ position in a vector, so that in moving from one layer to the next (i.e., in the height dimension), the largest jump will occur in the resulting vector. The width of the band, in this example, would be $2dNM + 1$, where d is the degree of differencing specified for smoothness among the layers. Again, because of symmetry, we need only store $dNM + 1$ diagonals of this band.

In general, if the ungraduated array has D dimensions, and the total number of cells is $N = n_1,n_2,n_3,. . . ,n_D$, then the number of entries required for the matrix A is N^2. However, taking full advantage of the symmetric band structure, we may store the main diagonal of A and the nontrivial diagonals above it as the columns of a matrix A'. If the jth dimension is chosen as the one for which "jumps" are largest under the transformation previously described, and if d_j is the degree of differencing associated with this dimension, then A' will have

$$N(n_1,n_2,. . .,n_{j-1}, n_{j+1}, . . .,n_D d_j + 1)$$

entries. The ratio of the size of A' to A is thus about d_j to n_j. Of course, we are free to choose j in order to make this ratio as small as possible.

For example, suppose we have a 20 by 9 by 5 array of ungraduated data. The full matrix A would require $(20 \times 9 \times 5)^2 = 810{,}000$ entries—probably too many for most computers. However, if we assume that only order 2 differencing is needed among the 20 rows, then the reduced matrix A' will require $(20 \times 9 \times 5) \times (9 \times 5 \times 2 + 1) = 81{,}900$ entries. Storage requirements have now been cut by almost 90 percent.

Thus, significant amounts of storage space can be saved by utilizing the band structure of matrix A instead of the whole matrix. The degree of savings will depend on the structure of the ungraduated array and the degree of differencing selected for each dimension.

I have written a short (129 lines total, or 82 lines excluding comments) FORTRAN program (see Appendix) which performs a conventional Whittaker-Henderson graduation and conserves storage space as I've described. The program will handle one or two dimensional data. The question remains whether the method employed here can be practically extended to more than two dimensions.

Perhaps my FORTRAN program may serve as a foundation upon which an extension to higher dimensions can be made by those who both have the need for graduating data in more than two dimensions and who wish to save computer storage space at the same time.

APPENDIX

```
      C  Program for Whittaker-Henderson graduation of one or two dimensional
      C  data.
      C
            PARAMETER MD=4,MR=50,MC=14,MP=MR*MC,MAC=MD*MC+1
      C
      C  MD = Maximum order of Differencing used in smoothing
      C  MR = Maximum # Rows of ungraduated data
      C  MC = Maximum # Columns of ungraduated data
      C  MP = Product of Maximum # rows & max # columns of ungraduated data
      C  MAC = Maximum # Columns needed in matrix A(I,J) described below
      C
            CHARACTER*11 TN(2)/'UNGRADUATED',' GRADUATED '/
            INTEGER DOP(MD+1,2),ORD(2),D(2),SM(MR,MR,2),BW,Z
            DIMENSION UNGRAD(MR,MC),WEIGHT(MR,MC),A(MP,MAC),U(MP),SW(2)
      C
      C  A = matrix, in reduced form, whose columns are the nontrivial
      C     diagonals of "W + V*SM( , ,1)+ H*SM( , ,2)".   V & H are numbers
      C     which give the relative weight placed on vertical and horizontal
      C     smoothing, respectively.
      C  BW = # nontrivial columns of matrix A minus 1
      C  DOP = Difference Operator.  2nd dimension refers to whether DOP is
      C     for vertical smoothing -- DOP(I,1), or horizontal -- DOP(I,2).
      C  ORD = Order of smoothing (1=vert, 2=horiz)
      C  SM( , ,1) = coefficient matrix for vertical smoothing.   Each
      C     coefficient refers to a multiple of the identity submatrix of
      C     size = to # columns of ungraduated data.
      C  SM( , ,2) = coefficient submatrix for horizontal smoothing.  The whole
      C     matrix consists of R repititions of this submatrix along the main
      C     diagonal (where R = # rows of ungrad data) and zeros elsewhere.
      C  SW = smoothing weights (1=vert, 2=horiz)
      C  U = vector containing the graduated data.
      C  UNGRAD = matrix of ungraduated data.
      C  WEIGHT = matrix of weights, to be applied to the ungraduated data.
      C
         10 FORMAT(//10X,'NUMBER OF ROWS =',I3/7X,'NUMBER OF COLUMNS =',I3//
           &'  ORDER',I2,' DIFFERENCING FOR VERTICAL SMOOTHING'/
           &'  ORDER',I2,' DIFFERENCING FOR HORIZONTAL SMOOTHING'//
           &'     VERTICAL SMOOTHING WEIGHT =',F5.1/
           &'   HORIZONTAL SMOOTHING WEIGHT =',F5.1)
         20 FORMAT(1H1,20X,'THE MATRIX OF ',A11,' NUMBERS:'/' ROW')
         30 FORMAT(I4,14F9.3)
      C
      C  Read the # of rows D(1) & # of columns D(2) of UNGRAD.  Read the
      C    order of differencing for vertical & horizontal smoothing followed
      C    by the corresponding weights for each type of smoothing.
            READ(5,*) D,ORD,SW
            WRITE(6,10) D,ORD,SW
      C  Read the ungraduated data followed by their respective weights.
            READ(5,*) ((UNGRAD(I,J),J=1,D(2)),I=1,D(1))
            READ(5,*) ((WEIGHT(I,J),J=1,D(2)),I=1,D(1))
            WRITE(6,20) TN(1)
            DO 40 I=1,D(1)
         40 WRITE(6,30) I,(UNGRAD(I,J),J=1,D(2))
      C  Compute the difference operators for vert & horiz smoothing:
            DO 140 K=1,2
            N = ORD(K) +  1
      C  GAMMA(N) = (N-1)! = ORD(K)!
            DO 100 J=1,N
        100 DOP(J,K) = (-1)**(J+N)*GAMMA(N)/(GAMMA(J)*GAMMA(N-J+1))
      C  Zero-out SM array on & above the main diagonal:
            DO 110 I=1,D(K)
            DO 110 J=I,D(K)
        110 SM(I,J,K) = 0
      C  Compute SM:
            DO 120 J=1,D(K)
            DO 120 I=MAX(1,J-ORD(K)),J
            DO 120 IR=MAX(1,I-ORD(K),J-ORD(K)),MIN(I,D(K)-ORD(K))
        120 SM(I,J,K) = SM(I,J,K)+DOP(I-IR+1,K)*DOP(J-IR+1,K)
            DO 130 I=2,D(K)
            DO 130 J=1,I-1
```

```
  130 SM(I,J,K) = SM(J,I,K)
  140 CONTINUE
      NRC = D(1)*D(2)
      BW = MAX(ORD(1)*D(2),ORD(2))
C  Fill in the first column of matrix A:
      DO 190 I=1,D(1)
      DO 190 J=1,D(2)
  190 A((I-1)*D(2)+J,1) = WEIGHT(I,J) +SW(1)*SM(I,I,1) +SW(2)*SM(J,J,2)
C  Zero-out the remaining BW columns of matrix A
      DO 200 J=1,BW
      DO 200 I=1,NRC
  200 A(I,J+1) = 0.
C  Fill out BW columns of A beginning with column 2:
      DO 220 K=1,2
      DO 220 Z=1,D(3-K)
      DO 220 I=1,D(K)
      IA = ((2-K)*I+(K-1)*Z-1)*D(2) + (2-K)*Z+(K-1)*I
      DO 220 J=I+1,MIN(I+ORD(K),D(K))
      JA = ((2-K)*J+(K-1)*Z-1)*D(2) + (2-K)*Z+(K-1)*J
  220 A(IA,JA-IA+1) = A(IA,JA-IA+1) + SW(K)*SM(I,J,K)
C  Compute the decomposition of matrix A as the product L*L'.
C  The nontrivial diagonals of L are stored in matrix A.
      DO 270 J=1,NRC
      IF(J .EQ. 1) THEN
        A(1,1) = SQRT(A(1,1))
      ELSE
        SUM = 0.
        DO 240 IH=MAX(1,J-BW),J-1
  240   SUM = SUM + A(IH,J-IH+1)**2
        A(J,1) = SQRT(A(J,1)-SUM)
      END IF
      DO 260 I=J+1,MIN(NRC,J+BW)
      SUM = 0.
      DO 250 IH=1,J-1
  250 IF(I-IH .LE. BW) SUM=SUM+A(IH,I-IH+1)*A(IH,J-IH+1)
  260 A(J,I-J+1) = (A(J,I-J+1)-SUM)/A(J,1)
  270 CONTINUE
C  Compute the graduated numbers by solving
C                AU =  LL'U = WU"
C  for U in 2 stages, where U" are the ungraduated numbers.
      DO 280 I=1,D(1)
      DO 280 J=1,D(2)
  280 U((I-1)*D(2)+J) = WEIGHT(I,J)*UNGRAD(I,J)
      U(1) = U(1)/A(1,1)
      DO 300 I=2,NRC
      SUM = 0.
      DO 290 J=MAX(1,I-BW),I-1
  290 SUM = SUM + U(J)*A(J,I-J+1)
  300 U(I) = (U(I)-SUM)/A(I,1)
      U(NRC) = U(NRC)/A(NRC,1)
      DO 320 I=NRC-1,1,-1
      SUM = 0.
      DO 310 J=I+1,MIN(NRC,I+BW)
  310 SUM = SUM + U(J)*A(I,J-I+1)
  320 U(I) = (U(I)-SUM)/A(I,1)
      WRITE(6,20) TN(2)
      DO 330  I=1,D(1)
  330 WRITE(6,30) I,(U(J),J=(I-1)*D(2)+1,I*D(2))
      END
```

ELIAS S. W. SHIU:

As two-dimensional Whittaker-Henderson graduation has been included in the Society of Actuaries' new Part 5 study note ([3], section 8.4), this is indeed a timely paper. The author has clearly demonstrated that APL is the natural tool for solving problems involving matrices. I hope that future revisions of the graduation syllabus will include the study of APL algorithms. I have the following comments on this paper.

The definition of S is perhaps not complete since mixed difference terms are not included. Let f be a function in two variables. The conditions

$$\Delta_x^2 f(x, y) = 0$$

and

$$\Delta_y^2 f(x, y) = 0$$

do not necessarily imply that f is a linear function. An example is

$$f(x, y) = xy.$$

The additional condition

$$\Delta_x \Delta_y f(x, y) = 0$$

is needed to ensure that f is of the form

$$a + bx + cy.$$

For a select life table, it is expected that

$$q_{[x]} < q_{[x-1]+1} < q_{[x-2]+2} < \cdots$$

The mortality rates graduated by the method in this paper need not satisfy the previous inequalities. Indeed if there is a lot of emphasis on smoothness, some of the graduated values may turn out to be negative or greater than one. Thus, we propose that appropriate linear constraints be included in the formulation of the problem. The resulting optimization problem is then solved with a quadratic-programming algorithm. (See section 7 of [5]).

Consider the function

$$g(\boldsymbol{u}, k) = \frac{1 - k}{F_T} F + \frac{k}{S_T} S.$$

In this paper the value of k is first fixed, and then g is minimized by varying $\boldsymbol{u}$. Perhaps one should minimize g by varying both $\boldsymbol{u}$ and k. If we ignore the natural constraints on $\boldsymbol{u}$ and k and apply the method of multivariate calculus, the problem becomes solving the equations

$$\frac{\partial}{\partial k} g = 0 \tag{1}$$

and

$$\frac{\partial}{\partial \boldsymbol{u}} g = \boldsymbol{0}, \tag{2}$$

simultaneously. Equation (1) is simply

$$F/F_T = S/S_T.$$

As shown in [4], the left-hand side of equation (2) is

$$2\left[\frac{1-k}{F_T} W(u - u'') + \frac{k}{S_T} K^T K u\right].$$

Traditionally, the choice of k has been made by trial and error. In proposing the method above, we are trying to eliminate the problem of choosing k. However, upon examining figure 2 of the paper, we see that the *optimal* k is around 0.5. The graduated values for such a k may not be sufficiently smooth.

The author has raised the question whether the index of "smoothness could be defined in more general terms for cases where the underlying curves are neither polynomials, exponentials, nor close to either of these." An answer to this question can be found in [1].

First, it should be pointed out that, by the Weierstrass approximation theorem, any continuous function on a compact set can be uniformly approximated by polynomials (compare [2], p. 486). In fact, since we are dealing with a finite set of points, there are polynomials interpolating these points. Of course, this is not the answer the author is looking for.

Dr. T. N. E. Greville ([1], pp. 389–90) has suggested that S may be generalized as

$$\sum_x [p(E)\, u_x]^2,$$

where $p(E)$ is a polynomial in the forward-shift operator E. By the theory of difference equations, the solutions of

$$p(E)u_x = 0$$

are linear combinations of products of polynomial and exponential functions. It is interesting to note that, nearly one hundred years ago, T. B. Sprague ([6], p. 110) used this result to discredit the method of moving-weighted-average graduation.

The last comment concerns Appendix B. We wish to find the coefficient vector c such that the quadratic form

$$(y - Xc)^T W (y - Xc) \qquad (3)$$

is minimized (compare [3], section 6.5.1).

Equating the derivative of (3) with respect to c with the zero vector, we have

$$-2X^TW(y - Xc) = \mathbf{0},$$

or

$$c = (X^TWX)^{-1}X^TWy. \quad (4)$$

Equation (4) is programmed as lines 20, 21, and 25 in Appendix B.

On the other hand, (3) can be expressed as

$$(W^{1/2}y - W^{1/2}X\,c)^T\,(W^{1/2}y - W^{1/2}Xc).$$

Thus the coefficient vector c is simply given by

$(W^{1/2}y)$ ⌹ $(W^{1/2}X)$.

REFERENCES

1. GREVILLE, T.N.E. "A Fortran Program for Generalized Whittaker-Henderson Graduation," *ARCH*, 1982.1, 385–401.
2. GREVILLE, T.N.E., AND VAUGHAN, H. Review of Discussion of "Polynomial Interpolation in Terms of Symbolic Operators," *TSA*, VI (1954), 484–87.
3. LONDON, R.L. *Graduation: The Revision of Estimates*. Part 5 Study Note 54-01-84. Itasca, Illinois: Society of Actuaries, 1984.
4. SHIU, E.S.W. "Matrix Whittaker-Henderson Graduation Formula," *ARCH*, 1977.1.
5. ———. Discussion of "An Extension of the Whittaker-Henderson Method of Graduation," *TSA*, XXXIV (1982), 362–68.
6. SPRAGUE, T.B. "The Graphic Method of Adjusting Mortality Tables," *JIA*, XXVI (1886), 77–112.

WILLIAM J. TAYLOR:

First, on behalf of all of the members of the Committee to Recommend New Disability Tables for Valuation, we congratulate Mr. Knorr on his lucid description of the extensions he has made to Whittaker-Henderson graduation in connection with our committee's charge. In addition to the obvious multidimensional nature of our problem, his standardization of smoothness and fit constants not only makes the graduation process simpler, it also makes it a lot easier to describe to actuaries not intimately involved in graduation.

In applying the author's program to graduating incidence rates, we encountered difficulties in obtaining satisfactory results and thought it might be caused by the invalidity of the assumption of equal intervals between the variable values in some of our dimensions. We investigated extending the program to handle divided differences in addition to normal differences and found the extension to be quite simple. Unfortunately, the improvements provided by this extension did not produce satisfactory results. We concluded that our problem was that we had too few variable values to graduate the complex mathematical form with the underlying probabilities and ended up using Lotus 1, 2, 3 to perform multidimensional graphic graduation as described in our committee report.

Revised listings incorporating the extensions and excluding many of the comments are attached. Two new global variables have been defined, one function added and small changes made to each of LSQ and GRAD, including the addition of the letter T to the name of each of these two functions.

The variable VI is a logical scalar indicating the variable interval. A value of 1 indicates equal intervals and causes the original calculations to be performed. A value of 0 indicates unequal intervals.

If VI is equal to 0, VM must be defined as a matrix containing each of the values that each of the variables assumes. There is a row for each dimension and enough columns to accommodate the dimension with the most values. The change to LSQ simply substitutes the variable values for the natural numbers sequence if unequal intervals are used. The change to GRAD simply substitutes the divided difference coefficients for the binomial coefficients if unequal intervals are used. The appropriate set of coefficients are determined in the new function DCM and three lines of GRAD are replaced by a single line returning the result from DCM.

```
     ∇GRADT[⎕]∇
     ∇ U←WEIGHTS GRADT UNGRAD;D̲;N;N̲;Z;K;A̲;K̲;W̲;E̲T;S̲T;I;NOTI;VV
    TESTRANK:→((ρρWEIGHTS)=ρρUNGRAD)/TESTSHAPE
     'RANK OF WEIGHTS DO NOT EQUAL RANK OF UNGRADUATED DATA'
     'NO GRADUATION WIL BE PERFORMED'
     U←UNGRAD
     →END
    TESTSHAPE:→(∧/(ρWEIGHTS)=ρUNGRAD)/TESTWEIGHT
     'SHAPE OF WEIGHTS DO NOT MATCH SHAPE OF UNGRADUATED DATA'
     'NO GRADUATION WILL BE PERFORMED'
     U←UNGRAD
     →END
    TESTWEIGHT:→((∧/,WEIGHTS≥0)∧((+/,WEIGHTS)>0))/TESTDIFF
     'WEIGHTS MAY NOT BE NEGATIVE AND MUST HAVE AT LEAST ONE POSITIVE VALUE'
     'NO GRADUATION WILL BE PERFORMED'
     U←UNGRAD
     →END
    TESTDIFF:→((ρ,DIFF)=ρρUNGRAD)/ADJDIFF
     '''DIFF'' NOT DEFINED PROPERLY'
     'ONE VALUE FOR EACH DIMENSION'
     'NO GRADUATION WILL BE PERFORMED'
     U←UNGRAD
     →END
    ADJDIFF:→((∧/DIFF≥1)∧(∧/DIFF≤ρUNGRAD)∧(∧/DIFF=⌊DIFF))/TESTSMOOTH
     '''DIFF'' NOT DEFINED PROPERLY'
     'ADJUSTED FROM ',(⍕DIFF),' TO ',⍕1⌈(⌊DIFF)⌊ρUNGRAD
     'GRADUATION WILL CONTINUE'
    TESTSMOOTH:→((∧/(,SMOOTH)>0)∧((+/,SMOOTH)≤1)∧((ρ,SMOOTH)=ρρUNGRAD))/TOTAL
     SMOOTHNESS
     'SMOOTHNESS FACTORS NOT DEFINED PROPERLY'
     'ALL FACTORS MUST BE GREATER THAN ZERO'
     'THE SUM OF THE FACTORS MUST BE LESS THAN ONE'
     'ONE FACTOR FOR EACH DIMENSION'
     'NO GRADUATION WILL BE PERFORMED'
     U←UNGRAD
```

```
[33]  →END
[34] TOTALSMOOTHNESS:Z←(1⌈(⌊DIFF)⌊⍴UNGRAD)-1
[35]  U←WEIGHTS LSQT UNGRAD
[36]  →((+/,SMOOTH)=1)/END
[37]  E̲T←+/,WEIGHTS×(U-UNGRAD)*2
[38]  →(E̲T>0)/INTIALIZE
[39]  'DATA ALREADY SMOOTH'
[40]  →END
[41] INTIALIZE:
[42]  D̲←⍴⍴UNGRAD
[43]  N←⍴UNGRAD
[44]  N̲←⍴,UNGRAD
[45]  Z←Z+1
[46]  K←,SMOOTH
[47]  A̲←(N̲,N̲)⍴0
[48]  S̲T←⍳0
[49]  I←0
[50] LOOP:I←I+1
[51]  K̲←DCM
[52]  K̲←(N[I],N[I])↑K̲
[53]  NOTI←(I≠⍳D̲)/⍳D̲
[54]  K̲←((×/N[NOTI]),N[I],1,N[I])⍴K̲
[55]  K̲←((×/N[NOTI]),N[I],(×/N[NOTI]),N[I])↑K̲
[56]  K̲←(N̲,N̲)⍴K̲
[57]  K̲←(1-⍳N̲)⌽K̲
[58]  K̲←(N[NOTI],N[I],N[NOTI],N[I])⍴K̲
[59]  K̲←((NOTI,I),D̲+NOTI,I)⍉K̲
[60]  K̲←(N̲,N̲)⍴K̲
[61]  S̲T←S̲T,(,UNGRAD)+.×(⍉K̲)+.×K̲+.×(,UNGRAD)
[62]  K̲←(⍉K̲)+.×K̲
[63]  K̲←K[I]×K̲
[64]  A̲←A̲+K̲
[65]  →(I<D̲)/LOOP
[66]  A̲←A̲÷+/S̲T
[67]  ⎕EX 'K̲'
[68]  W̲←(N̲,N̲)⍴(,((⍳N̲)∘.=⍳N̲))\,WEIGHTS×(1-+/K)÷E̲T
[69]  A̲←A̲+W̲
[70] GRADUATION:U←N⍴(W̲+.×,UNGRAD)⌹A̲
[71] END:
     ∇


     ∇DCM[⎕]∇
     ∇ R←DCM;J;IA;T
[1]    →VI/EI
[2]    IA←((N[I]-Z[I]),T,T)⍴(T,T)⍴1,(T←1+Z[I])⍴0
[3]    R←÷×/IA+ 1 2 1 3 ⍉VV[J]∘.-VV[J←(¯1+⍳N[I]-Z[I])∘.+⍳T]
[4]    →0
[5]   EI:R←N[I]↑⌽1,((⍳Z[I])!Z[I])×¯1*⍳Z[I]
[6]    R←((N[I]-Z[I]),N[I])⍴R
     ∇


     ∇LSQT[⎕]∇
     ∇ C←W LSQT Y;X;I;A;R;COEF;XI
[1]    W←((⍴,Y),⍴,Y)⍴(,(⍳⍴,Y)∘.=⍳⍴,Y)\,W
[2]    Z←,Z
[3]    X← 1 1 ⍴1
[4]    I←0
[5]   L1:I←I+1
[6]    →VI/VVP,0⍴VV←⍳(⍴Y)[I]
[7]    VV←(⍴Y)[I]↑VM[I;]
[8]   VVP:XI←VV∘.*0,⍳Z[I]
[9]    X←(((⍴X)[1]×(⍴Y)[I]),(⍴X)[2]×1+Z[I])⍴ 1 3 2 4 ⍉X∘.×XI
[10]   →(I<⍴⍴Y)/L1
[11]   A←(⍉X)+.×W+.×X
[12]   R←(W+.×,Y)+.×X
[13]   COEF←R⌹A
[14]   C←(⍴Y)⍴X+.×COEF
     ∇
```

LEE GIESECKE:

Frank E. Knorr's article on multidimensional Whittaker-Henderson graduation, looks very promising. I would like to make two suggestions that will make the paper slightly more general. I also add a note of caution in using the standardized fidelity and smoothness constants F_T and S_T.

First I suggest moving the smoothness constants inside all but the innermost summations. This would permit many more smoothness constants and allow us to establish the smoothness coefficients using single dimensional graduations prior to the multidimensional run. Without doing this, we may be using some very large smoothness constants where data are sparse.

Second I suggest permitting regions of discontinuity inside the innermost smoothness summation. The use of different smoothness measures and smoothness constants on either side of the discontinuity is related to this. Even when there is no discontinuity, it may be desirable to use different smoothness measures in different rows (or columns) of a particular cross section of the multidimensional array. This suggestion is rather academic but may occasionally prove useful. Table 1 illustrates why I note caution in using the standardized fidelity constants F_T and S_T.

TABLE 1

x	q_x	W_x	u''_x	u_{1x}	u_{2x}
85	.12575	43205	.12742	.12747	.12598
86	.13650	37772	.13772	.13750	.13662
87	.14792	32616	.14338	.14371	.14829
88	.16013	27791	.15984	.15969	.16096
89	.17308	23341	.17608	.17587	.17440
90	.18674	19301	.18828	.18882	.18830
91	.20103	15697	.20610	.20528	.20233
92	.21562	12541	.21468	.21557	.21618
93	.23011	9837	.23172	.23136	.22961
94	.24417	7573	.25028	.24399	.24245
95	.25745	5724	.25028	.25173	.25458
96	.26959	4251	.26687	.26543	.26545
97	.28024	3105	.27451	.27872	.27654
98	.28977	2235	.30194	.28940	.28634
99	.29869	1587	.26490	.28303	.29537
100	.30696	1113	.30695	.29707	.30365
101	.31461	771	.31724	.31228	.31119
102	.32167	529	.30897	.32148	.31799
103	.32817	359	.34613	.33047	.32402
104	.33414	241	.31813	.33542	.32928
105	.33960	160	.35056	.34089	.33374
106	.34460	106	.34087	.34341	.33740
107	.34917	69	.34865	.34057	.34026
108	.35333	45	.32190	.33065	.34230
109	.35712	29	.31523	.31390	.34353

The q's are the true values we are trying to estimate. Using q and W, the u'' vector was created in a simulation trial. The u_1 vector was created using a Whittaker-Henderson graduation on W and u'' with third differences and a smoothing coefficient of 162.2. The value of 162.2 was obtained from $\frac{k}{(1-k)} \cdot \frac{F_t}{S_t}$ with $k = .5$.

The u_2 vector was also created using a Whittaker-Henderson graduation with third differences but with the smoothing coefficient set to 1,076,616. The larger smoothing coefficient gives a better graduation as may be seen by comparing u_1 and u_2 with q, which they are trying to estimate. The larger value for the smoothing coefficient could be obtained using the F_t/S_t approach; however, k would have to equal .99985.

My own practise is to iteratively reset the smoothing coefficient until a twenty-fifth, fiftieth, or seventy-fifth percentile Chi-square statistic is obtained. Although it is time consuming, this normally gives good results.

(AUTHOR'S REVIEW OF DISCUSSION)

FRANK E. KNORR:

I would like to thank everyone who expressed an interest in this paper, in particular those who have contributed their written discussions. I am especially grateful to Mr. McKay who first conceived the idea of graduating a grid of data and made the method for graduating higher dimensions much clearer. I would like to address the various problems and issues raised in the discussions.

Programs

I am certainly not the only one who is glad that Mr. Kunkel made his FORTRAN program available to everyone in his discussion. His example of a 20 by 9 by 5 array nicely illustrates that developing a similar program to graduate 3-dimensional or n-dimensional data to make multidimensional graduation available to people with limited computer capacity would be time well spent. When I said that this method was impractical, I was only thinking of data where the ratio of d_j to n_j is large, such as a 5 by 5 by 5 array minimizing third differences. Here only about 16 percent of the cells are outside the nonzero band of A. Note that the upper triangle of this array has 1,225 zeros outside the band; putting the band portion of the upper triangle into a rectangular array creates 2,850 new cells which contain zeros. This explains why there is only a 40 percent reduction in storage needs, while 57 percent of the cells are outside the band portion of the upper triangle.

Professor Shiu's final remark concerning Appendix B makes the least squares program more concise. This is very fitting since conciseness is one of APL's trademarks and strengths.

Using divided differences to define smoothness was thought to be a huge task to program but Mr. Taylor makes it look easy. In spite of the difficulties in applying this to the work of our committee, I am sure this will prove to be a very useful enhancement.

Smoothness Constants

Mr. Giesecke's first suggestion that smoothness constants vary within each dimension would make graduation quite flexible. For areas of sparse data, a smaller (or larger) smoothness constant can be selected. Sparse data many times mean erratic values that need more smoothness.

Mr. Giesecke's note of caution is expressed a little differently by Professor Shiu. A natural inclination is to select a standardized k value around .5. After all, the idea is to minimize the expression

$$\frac{1-k}{F_T} F + \frac{k}{S_T} S.$$

Why not allow k to vary, so we can find "the smallest minimum"? I did not intend to imply that the best graduation is achieved when fit and smoothness are equally balanced. The best graduation must be determined by the graduator. I personally must be able to see good smoothness when the graduated values are graphed. Mr. Giesecke uses Chi-square statistics. We both seem to agree that graduation using a k less than .95 will probably not be good. This is like statistical tests where a 50 percent confidence limit is uninteresting, but confidence limits 95 or 99 percent are interesting.

I would normally consider a smoothness constant of .99985 close enough to 1 that I would simply use the least squares values as the graduated values. Graduated mortality rates that have been presented using $k = .99985$ are visibly different than the least squares values at the extremely high ages. However, if the weights truly reflect their importance (i.e. age 87 data is almost 500 times more important than age 107 data) then the least squares values would be perfectly acceptable to me.

Using the Chi-square statistic is interesting because it is similar to the definition of Whittaker-Henderson's measure of fit:

$$F = \Sigma\, W(u-u'')^2 \qquad \text{Pearson } X^2 = \Sigma \frac{(Wu - Wu'')^2}{Wu}.$$

$$= \Sigma \frac{W}{u}(u-u'')^2$$

Although I have not tried this, it seems that if one knows the degrees of freedom, the Chi-square confidence limit, F_T, and has an idea of what the weighted average of $1/u$'s are, then a very good first guess of k can be made.

Problem of Discontinuities

As Professor Shiu points out, any *continuous* function can be approximated by a polynomial. I believe that points of discontinuity are less academic than Mr. Giesecke suggests. There are many insurance policies with options at defined durations, causing experience before that duration to be different from experience afterwards. One would expect the group of employed people less than 65 years old to be different than those over 65. One approach would be to graduate the two parts separately.

Another approach is to use divided differences. Mr. Taylor's program uses divided differences to define smoothness and can be applied here. Since equidistance between points is no longer assumed, values must be entered (into the VM array) so that the distances among the points can be determined. These values are typically 25, 35, 45, 55, 62 for the age dimension or 0, 7, 14, 30, 90, 180 for the elimination period dimension. However, any values can be entered. When the values of a standard table are entered, the result is a set of graduated values whose shape is related to the shape of the standard table. If second differences are minimized and total smoothness is used, then each graduated value is a linear function of its corresponding standard value ($u_i = a\, s_i + b$ for all i's). This is the case regardless of the shape of the standard curve. Illustration D shows how this can be used to graduate data that has a point of discontinuity.

It should be noted that the total number of deaths (or terminations) will be preserved when divided differences are used. The average age at death (or average duration) will only be preserved if the ages (or durations) are used to define the distances among ungraduated data.

Problem of Mixed Terms

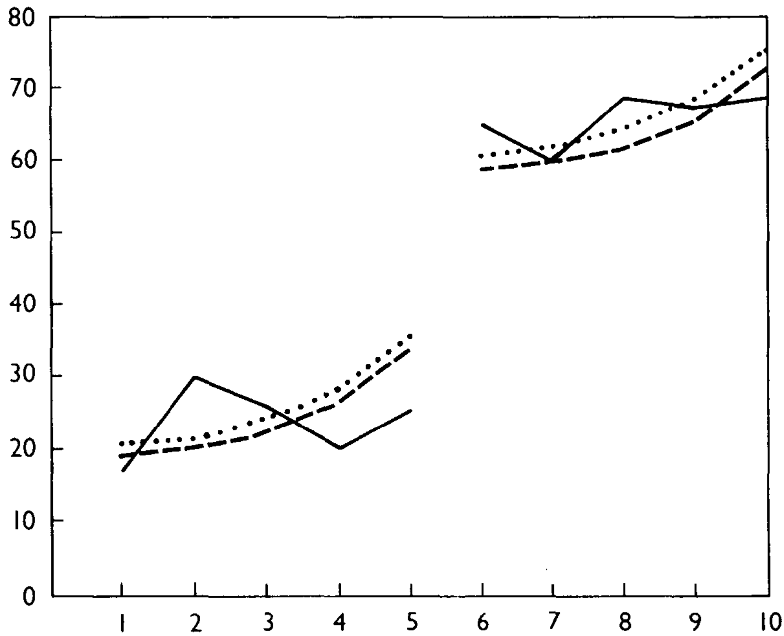
Professor Shiu suggests that the definition of S is not complete without mixed difference terms. When second differences are minimized in two directions, the graduated values can be thought of as lying on a surface. With total smoothness, we can be assured that any cross section of the surface produces a linear graph as long as the cutting plane is perpendicular to the x-axis or y-axis. But the surface is not necessarily a plane. This is because the surface is defined by the equation $a_{00} + a_{10}\, x + a_{01}\, y + a_{11}\, x\, y$ and a plane is defined by the equation $b_{00} + b_{10}\, x + b_{01}\, y$. I would solve this by finding the weighted average of $a_{11}\, x\, y$ for all x, y combinations. Then I would redefine the constant term to be a_{00} plus this weighted

ILLUSTRATION D

Problem of Discontinuities

Weights	95	72	65	60	57	25	22	20	19	18	
Ungraduated Data	17	30	26	20	25	65	60	68	67	68	(———)
Standard Data	21	22	24	28	36	61	62	64	68	76	(- - - - -)
Graduated Data	19.6	20.6	22.5	26.4	34.2	58.5	59.5	61.4	65.3	73.1	(– – –)

Use Standard Data to define the distances among Ungraduated Data and k = 1.



average. This may be all that is necessary to get the least squares plane. But, it seems that a simple counter example will show that further adjustments to the coefficients are necessary.

If this approach is a solution, it can be used to eliminate mixed terms (any terms, in fact) for higher differences and/or higher dimensions.

Problem of Boundaries

Professor Shiu's second point is that there are natural boundaries for mortality rates (between 0 and 1) which are ignored by this graduation method. Monotonically increasing select mortality rates can be put into this category since we want the differences between $q_{[x-i]+i}$ and $q_{[x-(i-1)]+i-1}$ to be greater than zero (or their ratios to be greater than one). Faced with this problem a number of times, my solution has been to graduate the logarithms of the data. No matter what the graduated numbers are, when they are converted from logarithms back to rates, only positive rates are possible. This is a simple practical solution which also has a price; some nice properties of Whittaker-Henderson graduation are lost.

Another solution might be to use divided differences with a standard table which I have described as a solution to the problem of discontinuities. Graduated data can usually be forced to take on the same shape as the standard data even if the standard data contain sudden jumps and/or sudden changes of slope. Therefore, sparse mortality data could be used to produce graduated rates which include the infamous hump in mortality rates for males in their twenties. I feel the use of divided differences in defining S is such a major step that it deserves its own name, like "Type D" (for divided differences—by-passing C entirely).