Bayesian graduation on a spreadsheet

BRUNO KOLLER, Riehen

Bayesian Graduation on a Spreadsheet

1 Spreadsheets

1.1 Characteristics of spreadsheets

The first spreadsheet, Visicalc, was developed by Daniel Bricklin and Robert Frankston in 1978 (see *Licklider;* 1989). In the meantime there are many similar products; the most widespread is "1-2-3" from Lotus.
A spreadsheet consists of cells, arranged in a matrix. All cells together form the sheet. A cell is identified by a cell-address. A cell may contain a function assignment; together with the arguments of the function, it determines uniquely the cellvalue of the cell. The arguments are cellvalues of other cells; referencing the arguments is done by celladdress. All function assignments together form the spreadsheet program.
In a spreadsheet cells can be concatenated freely: a cellvalue may serve as input to another cell, its cellvalue in turn may be the input for another cell. On entering a new function assignment the cellvalues are recalculated, thus the sheet contains always the actual results.
The following example explains the working of a spreadsheet. The program calculates the present value of an annually payable annuity. The upper part of the table shows the cellvalues, as seen on the computer screen (formatted in a suitable manner). The numbers on the left and the letters above are not part of the sheet. They are coordinates for the cell-addresses. For instance, the cell with address $B3$ has the value 10. With the celladdress one can find in the lower part of the table the corresponding function assignment.
The simplest case of a function is the constant function. In cell $A1$ we have, for example, the constant function "interest i" and in cell $B1$ the constant function 0.03. Cell $B5$ has been assigned the function $\ddot{a}(v, n, R) = R \cdot (1 - v^n)/(1 - v)$; the argument v is stored in cell $B2$ – itself a function defined as $v(i) = 1/(1 + i)$.
The isolated constant functions which are not arguments to a function are not part of the program: they are only comments.
Function assignments may involve several operations. Besides the four basic arithmetic operations (+, −, *, /) and the exponential operation (^) most spreadsheets offer numerous higher functions. We shall use the exponential

	A	B	C
1	interest i	0.03	
2	discount factor v	0.970874	
3	duration n	10	
4	annuity R	1000	
5	present value ä	8786	
6			

```
A1: interest i
B1: +0.03
A2: discount factor v
B2: +1/(1+B1)
A3: duration n
B3: +10
A4: annuity R
B4: +1000
A5: present value ä
B5: +B4*(1-B2^B3)/(1-B2)
```

function e^x (written @EXP (x)) and the matrix-operations inversion and multiplication.

No programming language fits every application well. Spreadsheets are best suited for calculation-oriented problems: ledger, financial analysis, statistics, modelling, systems analysis and (Monte-Carlo-)simulation. Actuarial problems that can be solved well in spreadsheets are: calculation of present values, reserves, frequency and claim distributions, modelling of collectives, prognosis and graduation. Most spreadsheets offer preprogrammed financial functions. Spreadsheets are inappropriate for symbol manipulation. Also very calculation-intensive problems, like the evaluation of a aggregate claim distribution, may be out of reach.

1.2 Spreadsheets as objectbased, functional languages

Programming languages can be divided into generations. Normally machine and assembler languages are classified as first and second generation languages. Languages of the third and fourth generation differ in their organisations: imperative programming languages as Fortran, *C* and Pascal belong to the third generation, declarative languages (Spreadsheets, Lisp, Prolog, FP, Hope) to the fourth generation (*Schneider,* 1986).

For commercial applications imperative languages dominate. Imperative languages specify explicitly the order of computation in a program. Declarative languages on the other hand do not control the stepwise transformation of an initial state to a final state. They describe a problem, identifying directly the relations between input and output data. Declarative languages must have a mechanism to find the solution to a problem; the programmer does not specify how to solve it.

When the dependencies between input and output are formulated as functions from the admissible input data to the correct output data, we speak of functional or applicative programming languages. If the dependencies are in the form of relations, they are called relational programming languages. Spreadsheets, Lisp and Logo are functional languages; Prolog is a relational language.

A programming language is objectbased, if it supports "objects". Objects have evolved from modules. They are informationcarriers, which can be queried and manipulated by well-defined operations. Programs in objectbased languages are ensembles of objects exchanging messages. Ada is an example of a object-based programming language.

The cells of a spreadsheet can be understood as objects. A cell has a value, which can be queried by other cells and which can be changed subject to other cellvalues. Each cell acts as a small computer. A sheet consists of numerous, communicating objects. A. Kay calls spreadsheets "tissuelike superobjects" (*Kay,* 1984). Spreadsheets are object-based, but not objectoriented languages like Smalltalk or *C* + +; missing are the higher language elements "classbuilding" and "inheritance" (see *Wegner,* 1989).

Functional languages have several advantages over traditional imperative languages: simple grammar, clear and unique semantics. The so-called referential transparency (lack of a computing history) opens the possibility to program transformation: first write a program concentrating on clarity and let

a compiler transform it, in a second step, to an efficient, semantically equivalent program. Thanks to their data-centered approach declarative languages are a natural fit for parallel programming. On today's computers these languages are not as fast as imperative languages; this will certainly change with the upcoming of parallel computers.

2 Bayesian graduation

2.1 Graduation as statistical problem

Graduation is the process of drawing a curve through a sequence of sample data; the curve should be smooth, yet fit the given data to a certain degree.

There are numerous graduation methods. Most of them are easy to implement on a spreadsheet. Especially the method of "moving averages" can quickly be programmed for a first impression (see e.g. *Miller,* 1946, chapter 4).

One would expect that most graduation procedures treat graduation as a statistical problem, given the stochastic nature of sampling. This is not the case, for classical statistics deal only with observation. The sample is the only source of information – pre-existing experience is neglected. But without the assumption that "in reality" succeeding data are correlated, one has to draw the curve through the observed data points. A full integration of the graduation problem into the theory of statistics is possible with Bayesian statistics: the correlation of adjacent values is expressed as prior information. The proposal of graduation using Bayesian statistics stems from *George S. Kimeldorf* and *Donald A. Jones* (*Kimeldorf/Jones,* 1967).

Statistics deal with inference from observational data, produced by an unknown probability distribution, about the underlying distribution. For graduation of n data points we have to assess a n-dimensional probability distribution. We will use its n-dimensional vector of expectations as graduated data, but other moments or quantiles may serve better regarding the purpose of the graduation.

For Bayesian statistics the pre-existing information about the parametric values to be estimated is quantified as probability distribution over the parameter space. We quantify smooth paths in our prior distribution higher than unsmooth paths. Combining the prior information with the actual observations according to Bayes' theorem defines the posterior distribution.

The posterior distribution will still assign smooth curves higher probabilities than uneven curves, producing a graduation.

Bayesian graduation is computationally intensive – it can only be done on a computer. In turn the method is theoretically sound. The knowledge, which justifies graduation, is specified explicitly and integrated with never observations to give a accurate picture of the available information.

2.2 Graduation of health insurance data

We illustrate Bayesian graduation using an example from health insurance. More than realistic assumptions we shall stress the methodological aspects of the method.

The data stem from a Swiss investigation in the years 1972/73 about average sickness days of males, age $x = 30$ to 65. The average number of days, the 36-dimensional vector u, are stored in column B, rows 23 to 58 of the spreadsheet (see table in the appendix). As we see in graph 1 (see Appendix), the observed data vary around a climbing slope.

For simplicity we assume that the sample distribution may be approximated by a 36-dimensional normal distribution. A multivariate normal distribution is given by the vector of expectations and the covariance matrix. Let the covariances be known; the expectations have to be estimated. We make the usual assumption that the random variables of the sample distribution are independent; the covariance matrix $B = (b_{ij})$ becomes a diagonal matrix: $b_{ij} > 0$ for $i = j$ and $b_{ij} = 0$ otherwise. In the table we set the variances all to four, i.e. $b_{ii} = 4$.

The prior distribution weighs our pre-existing knowledge about the 36 expected values for the ages 30 to 65 of the sample distribution. As prior distribution we choose a 36-dimensional normal distribution. It is defined by the 36-dimensional expectation vector v and the 36×36 covariance matrix A. Let the vector v of the expectations be defined by

$$v(x) = k_1 \cdot e^{k2(x-30)/35} \quad x = 30, 31, \ldots, 65.$$

In the example k_1 and k_2 are 1 respectively 2.772589, giving: $v(30) = 1$ and $v(65) = 16$. The expected values of the prior distribution are plotted in graph 1.

We follow the advice of *Kimeldorf/Jones* and define the covariance matrix $A = (a_{ij})$ as

$$\begin{aligned} a_{ij} &= k_3 && \text{for } i = j, \\ a_{ij} &= k_3 \cdot k_4^{(i-j)} && \text{for } i > j, \\ a_{ij} &= k_3 \cdot k_4^{(j-i)} && \text{for } i < j, \\ k_3 &> 0, && 0 < k_4 < 1. \end{aligned}$$

We observe that the elements of the matrix diagonal, k_3, are the variances of the random variables; $k_4^{(i-j)}$ respectively $k_3^{(j-i)}$ are the correlations of the random variables i and j. This way two consecutive random variables are more correlated than distant random variables. One can easily show that the matrix A is positive definite. In our example k_3 has been set to 1 and k_4 to 0.9.

Let us state our prior information once again: we judge a exponential increase of sickness days as most probable. The uncertainty about the number of days has been expressed as a variance of 1. And we assumed a correlation of sickness days decreasing exponentially with increasing distance.

We now calculate the posterior distribution. One can show (see for instance *DeGroot*, 1970, page 175), that normal distributions form a conjugate family: Bayes' theorem transforms a normal prior distribution for any normal sample distribution into a normal posterior distribution. In our case we get a 36 dimensional normal distribution with expectation vector

$$w = (A^{-1} + B^{-1})^{-1} \cdot (B^{-1} \cdot u + A^{-1} \cdot v)$$

and covariance matrix

$$(A^{-1} + B^{-1})^{-1}.$$

The expectation vector w may be written in a form which is easier to calculate:

$$\begin{aligned} w &= u + (A^{-1} + B^{-1})^{-1} \cdot [(B^{-1} \cdot u + A^{-1} \cdot v) - (A^{-1} + B^{-1}) \cdot u] \\ &= u + (A^{-1} + B^{-1})^{-1} \cdot [A^{-1} \cdot (v - u)] \\ &= u + [(A^{-1} + B^{-1})^{-1} \cdot A^{-1}] \cdot (v - u) \\ &= u + [A \cdot (A^{-1} + B^{-1})]^{-1} \cdot (v - u) \\ &= u + (A \cdot B^{-1} + I)^{-1} \cdot (v - u). \end{aligned}$$

I is the unit matrix.

To calculate the vector we proceed as follows (see sheet in the appendix, rows 23 to 58): column B contains the observed data u. The diagonal elements $1/b_{jj}$ of the matrix B^{-1} are stored in column C. The vector v is calculated in column D using the parameters k_1 (cell $C7$) and k_2 (cell $C8$). The difference vector r of v and u is in column E. The 36×36-matrix $C = (c_{ij})$, columns I to AR, is defined as $(A \cdot B^{-1} + I)$:

$$
\begin{aligned}
c_{ij} &= k_3/b_{ij} + 1 && \text{for } i = j \\
c_{ij} &= (k_3/b_{ij}) \cdot k_4^{j-i} && \text{for } i < j \\
c_{ij} &= (k_3/b_{ij}) \cdot k_4^{i-j} && \text{for } i > j.
\end{aligned}
$$

Using the matrix operations we invert C and store the matrix $D = C^{-1}$ in columns AT to CC. The matrix D is then multiplied with the vector r to give the vector s (column F). The vector w in column G is the sum of the vectors u and s. We get:

$$
\begin{aligned}
C &= (A \cdot B^{-1} + I) \\
D &= C^{-1} \\
r &= v - u \\
s &= D \cdot r \\
w &= u + s.
\end{aligned}
$$

In graph 1 besides the observed values and the expectations of the prior distribution, one can see the expected values of the posterior distribution, which defines the graduation curve.

Graduations with different parameters k_3 and k_4 are plotted in the graphs 2 to 5. They give an impression of how the prior distribution influences the graduation curve. In graph 2 the variance parameter k_3 is increased to 2, i.e. the pre-information about the number of sickness days is considered less reliable. The graduation curve follows rather the observational data. An increase of the parameter k_4 from 0.9 to 0.95 has the effect of smoothing; this amounts to a higher interdependency of the yearly sickness days (graph 3).

When we decrease the variance from 1 to 0.5 – assuming one knows quite well the average number of sickness days – the graduation curve is nearly identical

with prior data (graph 4). Decreasing the postulated correlations from 0.9 to 0.75 produces a curve with better fit (graph 5).

B. Koller
Dinkelbergstrasse 21
4125 Riehen

Bibliography

Backus, J. (1978): Can programming be liberated from the von Neumann style? A functional style and its algebra of programs. Communications of the Association for Computing Machinery (ACM) 21.8, 613–641.

Darlington, J. (1982): Program transformation. In: Functional programming and its applications. Cambridge University Press, 193–215.

DeGroot, M.H. (1970): Optimal statistical decisions. McGraw-Hill.

Ghezzi, C./Jazayeri, M. (1989): Konzepte der Programmiersprachen. Oldenburg, München.

Hickman, J.C./Miller, R.B. (1977): Notes on bayesian graduation. Transactions of the Society of Actuaries 29, 7–49.

Hudak, P. (1989): Conception, evolution, and application of functional programming languages. ACM Computing Surveys 21.3, 359–411.

Kay, A. (1984): Computer software. Scientific American 251.3, 41–47.

Kimeldorf, G.S./Jones, D.A. (1967): Bayesian graduation. Transactions of the Society of Actuaries 19.1, 66–127.

Licklider, T.R. (1989): Ten years of rows and columns. Byte 14.13, 324–331.

Lindley, D.V. (1980): Bayesian statistics, a review. Society for Industrial and Applied Mathematics, Philadelphia.

Miller, M.D. (1946): Elements of graduation. Actuarial Society of America.

Pascoe, G.A. (1988): Elements of object-oriented programming. Byte 11.8, 139–144.

Schneider, H. (Ed.) (1986): Lexikon der Informatik und Datenverarbeitung. Oldenburg, München.

Wegner, P. (1989): Learning the language. Byte 14.3, 245–253.

Appendix

BAYESIAN GRADUATION OF HEALTH STATISTICS DATA

Prior distribution: multivariate normal distribution

Parameters for expectation vector v

k1 1.000000

k2 2.772589

Parameters for covariance matrix A

k3 1.0

k4 0.9

x	u	1/bjj	v	r	s	w
30	0.929	0.250	1.000	0.071	0.434	1.363
31	1.853	0.250	1.082	-0.771	-0.345	1.508
32	1.511	0.250	1.172	-0.339	0.137	1.648
33	5.386	0.250	1.268	-4.118	-3.580	1.806
34	1.680	0.250	1.373	-0.307	0.110	1.790
35	1.810	0.250	1.486	-0.324	-0.017	1.793
36	0.820	0.250	1.608	0.788	0.987	1.807
37	1.973	0.250	1.741	-0.232	-0.087	1.886
38	3.521	0.250	1.885	-1.636	-1.547	1.974
39	2.962	0.250	2.040	-0.922	-0.970	1.992
40	0.000	0.250	2.208	2.208	1.971	1.971
41	0.411	0.250	2.390	1.979	1.654	2.065
42	3.180	0.250	2.587	-0.593	-0.922	2.258
43	1.806	0.250	2.801	0.995	0.610	2.416
44	0.170	0.250	3.031	2.861	2.448	2.618
45	4.183	0.250	3.281	-0.902	-1.219	2.964
46	1.013	0.250	3.552	2.539	2.251	3.264
47	7.429	0.250	3.845	-3.584	-3.728	3.701
48	0.361	0.250	4.162	3.801	3.602	3.963
49	8.286	0.250	4.505	-3.781	-3.846	4.440
50	2.333	0.250	4.876	2.543	2.408	4.741
51	3.671	0.250	5.278	1.607	1.527	5.198
52	10.656	0.250	5.713	-4.943	-4.887	5.769
53	9.676	0.250	6.184	-3.492	-3.558	6.118
54	3.591	0.250	6.694	3.103	2.726	6.317
55	11.440	0.250	7.246	-4.194	-4.742	6.698
56	1.829	0.250	7.843	6.014	5.039	6.868
57	4.656	0.250	8.490	3.834	2.686	7.342
58	2.130	0.250	9.190	7.060	5.869	7.999
59	14.369	0.250	9.947	-4.422	-5.360	9.009
60	3.730	0.250	10.767	7.037	6.060	9.790
61	8.174	0.250	11.655	3.481	2.773	10.947
62	15.519	0.250	12.616	-2.903	-3.204	12.315
63	10.286	0.250	13.656	3.370	3.304	13.590
64	27.575	0.250	14.781	-12.794	-12.451	15.124
65	11.667	0.250	16.000	4.333	4.431	16.098

	I	J	K		AP	AQ	AR
21	ci1	ci2	ci3	. . .	ci34	ci35	ci36
22							
23	1.250	0.225	0.203		0.008	0.007	0.006
24	0.225	1.250	0.225		0.009	0.008	0.007
25	0.203	0.225	1.250		0.010	0.009	0.008
26	0.182	0.203	0.225		0.011	0.010	0.009
27	0.164	0.182	0.203		0.012	0.011	0.010
28	0.148	0.164	0.182		0.013	0.012	0.011
29	0.133	0.148	0.164		0.015	0.013	0.012
30	0.120	0.133	0.148		0.016	0.015	0.013
31	0.108	0.120	0.133		0.018	0.016	0.015
32	0.097	0.108	0.120		0.020	0.018	0.016
33	0.087	0.097	0.108		0.022	0.020	0.018
34	0.078	0.087	0.097		0.025	0.022	0.020
35	0.071	0.078	0.087		0.027	0.025	0.022
36	0.064	0.071	0.078		0.030	0.027	0.025
37	0.057	0.064	0.071		0.034	0.030	0.027
38	0.051	0.057	0.064		0.038	0.034	0.030
39	0.046	0.051	0.057		0.042	0.038	0.034
40	0.042	0.046	0.051		0.046	0.042	0.038
41	0.038	0.042	0.046	. . .	0.051	0.046	0.042
42	0.034	0.038	0.042		0.057	0.051	0.046
43	0.030	0.034	0.038		0.064	0.057	0.051
44	0.027	0.030	0.034		0.071	0.064	0.057
45	0.025	0.027	0.030		0.078	0.071	0.064
46	0.022	0.025	0.027		0.087	0.078	0.071
47	0.020	0.022	0.025		0.097	0.087	0.078
48	0.018	0.020	0.022		0.108	0.097	0.087
49	0.016	0.018	0.020		0.120	0.108	0.097
50	0.015	0.016	0.018		0.133	0.120	0.108
51	0.013	0.015	0.016		0.148	0.133	0.120
52	0.012	0.013	0.015		0.164	0.148	0.133
53	0.011	0.012	0.013		0.182	0.164	0.148
54	0.010	0.011	0.012		0.203	0.182	0.164
55	0.009	0.010	0.011		0.225	0.203	0.182
56	0.008	0.009	0.010		1.250	0.225	0.203
57	0.007	0.008	0.009		0.225	1.250	0.225
58	0.006	0.007	0.008		0.203	0.225	1.250
59							

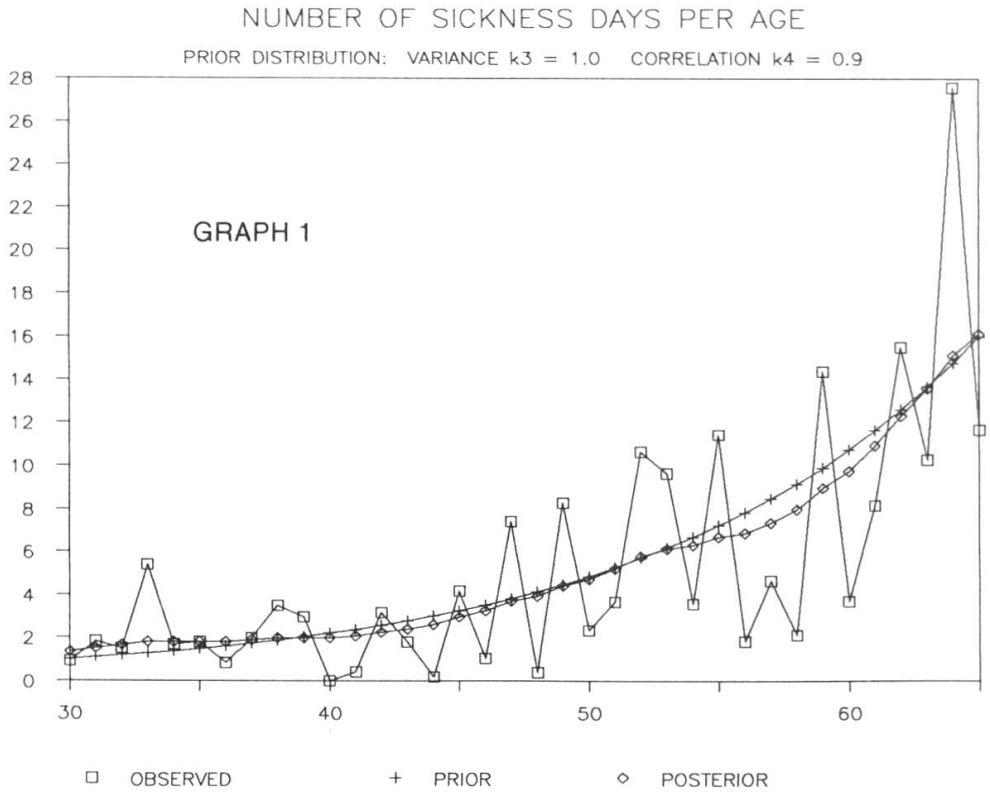
	AT	AU	AV		CA	CB	CC
21	di1	di2	di3	. . .	di34	di35	di36
22							
23	0.864	-0.106	-0.082		0.000	0.000	0.000
24	-0.106	0.877	-0.096		0.000	0.000	0.000
25	-0.082	-0.096	0.884		0.000	0.000	0.000
26	-0.064	-0.075	-0.090		0.000	0.000	0.000
27	-0.050	-0.058	-0.070		0.000	0.000	0.000
28	-0.039	-0.045	-0.054		0.000	0.000	0.000
29	-0.030	-0.035	-0.042		0.000	0.000	0.000
30	-0.023	-0.027	-0.033		0.000	0.000	0.000
31	-0.018	-0.021	-0.025		0.000	0.000	0.000
32	-0.014	-0.016	-0.020		0.000	0.000	0.000
33	-0.011	-0.013	-0.015		0.000	0.000	0.000
34	-0.009	-0.010	-0.012		0.000	0.000	0.000
35	-0.007	-0.008	-0.009		-0.001	0.000	0.000
36	-0.005	-0.006	-0.007		-0.001	-0.001	-0.001
37	-0.004	-0.005	-0.006		-0.001	-0.001	-0.001
38	-0.003	-0.004	-0.004		-0.001	-0.001	-0.001
39	-0.002	-0.003	-0.003		-0.002	-0.001	-0.001
40	-0.002	-0.002	-0.003		-0.002	-0.002	-0.001
41	-0.001	-0.002	-0.002	. . .	-0.003	-0.002	-0.002
42	-0.001	-0.001	-0.002		-0.003	-0.003	-0.002
43	-0.001	-0.001	-0.001		-0.004	-0.004	-0.003
44	-0.001	-0.001	-0.001		-0.006	-0.005	-0.004
45	-0.001	-0.001	-0.001		-0.007	-0.006	-0.005
46	0.000	0.000	-0.001		-0.009	-0.008	-0.007
47	0.000	0.000	0.000		-0.012	-0.010	-0.009
48	0.000	0.000	0.000		-0.015	-0.013	-0.011
49	0.000	0.000	0.000		-0.020	-0.016	-0.014
50	0.000	0.000	0.000		-0.025	-0.021	-0.018
51	0.000	0.000	0.000		-0.033	-0.027	-0.023
52	0.000	0.000	0.000		-0.042	-0.035	-0.030
53	0.000	0.000	0.000		-0.054	-0.045	-0.039
54	0.000	0.000	0.000		-0.070	-0.058	-0.050
55	0.000	0.000	0.000		-0.090	-0.075	-0.064
56	0.000	0.000	0.000		0.884	-0.096	-0.082
57	0.000	0.000	0.000		-0.096	0.877	-0.106
58	0.000	0.000	0.000		-0.082	-0.106	0.864
59							

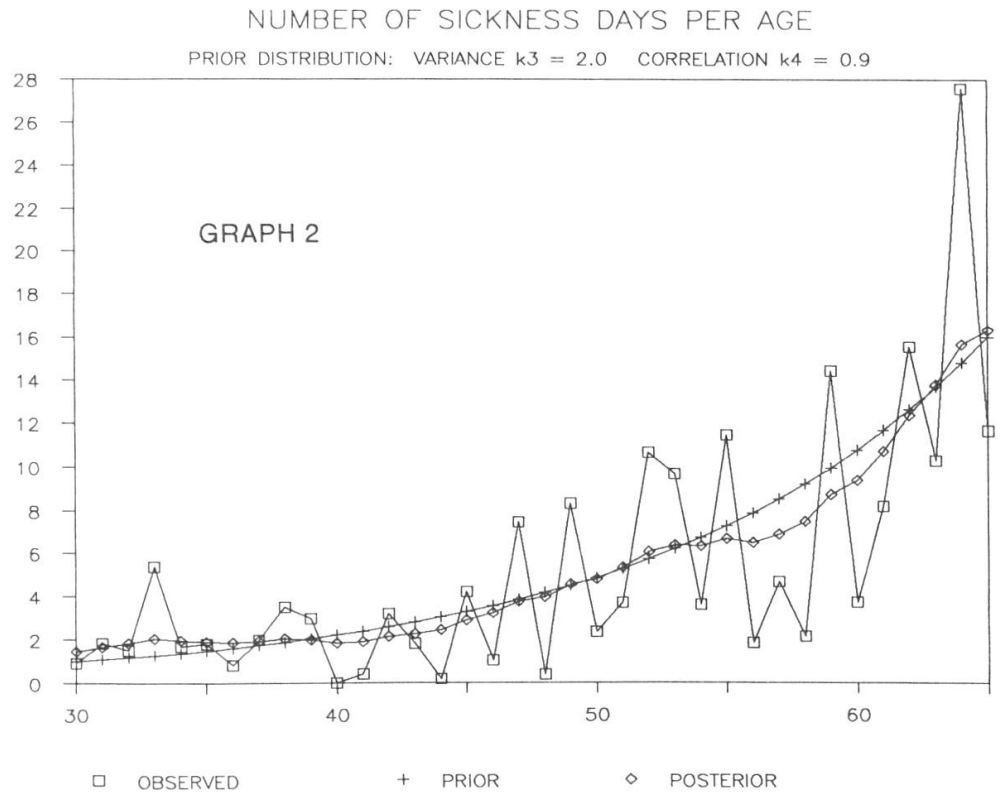
Row 23

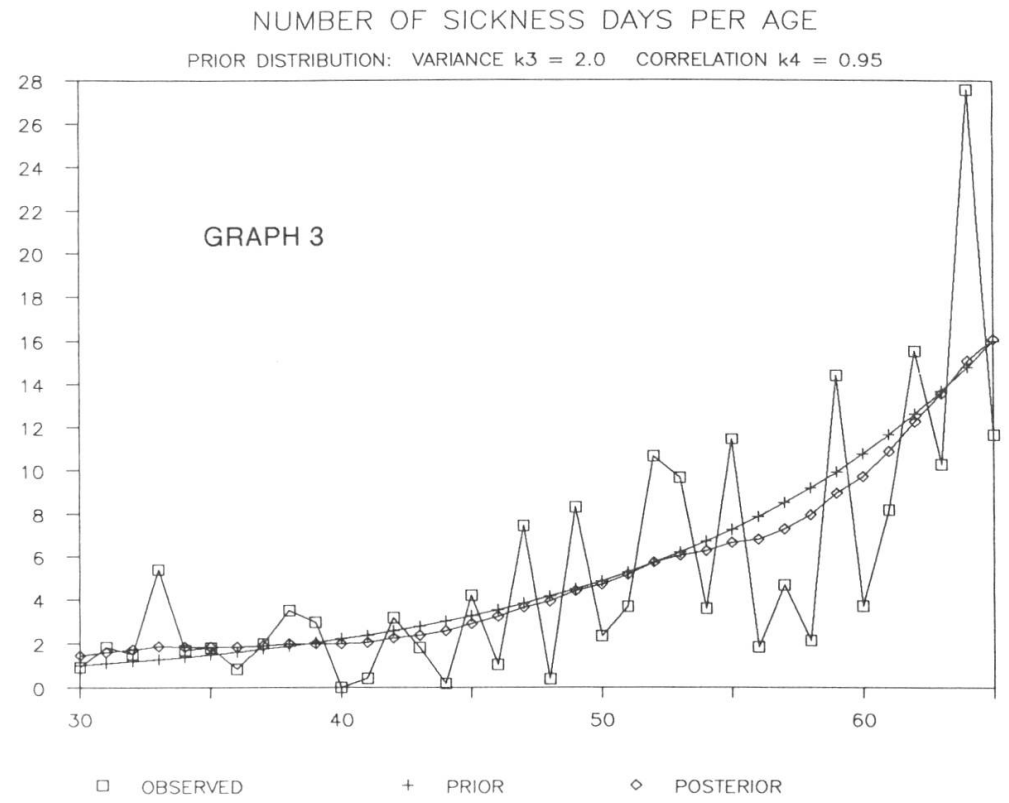
```
 A23:  +30                              AP23:  +C56*C10*C11^33
 B23:  +0.929                           AQ23:  +C57*C10*C11^34
 C23:  +0.25                            AR23:  +C58*C10*C11^35
 D23:  +C7*@EXP(C8*(A23-30)/35)         AT23:  +0.8635237062
 E23:  +D23-B23                         AU23:  -0.1060654641
 F23:  +0.4337027439                    AV23:  -0.0824310392
 G23:  +B23+F23                         AW23:  -0.0640630417
 I23:  +C23*C10+1                       AX23:  -0.0497879608
 J23:  +C24*C10*C11^1                   AY23:  -0.0386937773
 K23:  +C25*C10*C11^2                   AZ23:  -0.0300716963
 L23:  +C26*C10*C11^3                   BA23:  -0.0233708626
 M23:  +C27*C10*C11^4                   BB23:  -0.0181631673
 N23:  +C28*C10*C11^5                   BC23:  -0.0141158966
 O23:  +C29*C10*C11^6                   BD23:  -0.0109704749
 P23:  +C30*C10*C11^7                   BE23:  -0.0085259446
 Q23:  +C31*C10*C11^8                   BF23:  -0.0066261274
 R23:  +C32*C10*C11^9                   BG23:  -0.0051496462
 S23:  +C33*C10*C11^10                  BH23:  -0.0040021701
 T23:  +C34*C10*C11^11                  BI23:  -0.0031103882
 U23:  +C35*C10*C11^12                  BJ23:  -0.0024173256
 V23:  +C36*C10*C11^13                  BK23:  -0.0018787032
 W23:  +C37*C10*C11^14                  BL23:  -0.0014601091
 X23:  +C38*C10*C11^15                  BM23:  -0.0011347997
 Y23:  +C39*C10*C11^16                  BN23:  -0.0008819914
 Z23:  +C40*C10*C11^17                  BO23:  -0.0006855326
AA23:  +C41*C10*C11^18                  BP23:  -0.0005328716
AB23:  +C42*C10*C11^19                  BQ23:  -0.0004142553
AC23:  +C43*C10*C11^20                  BR23:  -0.0003221053
AD23:  +C44*C10*C11^21                  BS23:  -0.0002505342
AE23:  +C45*C10*C11^22                  BT23:  -0.0001949694
AF23:  +C46*C10*C11^23                  BU23:  -0.0001518611
AG23:  +C47*C10*C11^24                  BV23:  -0.000118455
AH23:  +C48*C10*C11^25                  BW23:  -0.0000926168
AI23:  +C49*C10*C11^26                  BX23:  -0.0000726958
AJ23:  +C50*C10*C11^27                  BY23:  -0.0000574193
AK23:  +C51*C10*C11^28                  BZ23:  -0.0000458112
AL23:  +C52*C10*C11^29                  CA23:  -0.00003713
AM23:  +C53*C10*C11^30                  CB23:  -0.0000308209
AN23:  +C54*C10*C11^31                  CC23:  -0.000026481
AO23:  +C55*C10*C11^32
```

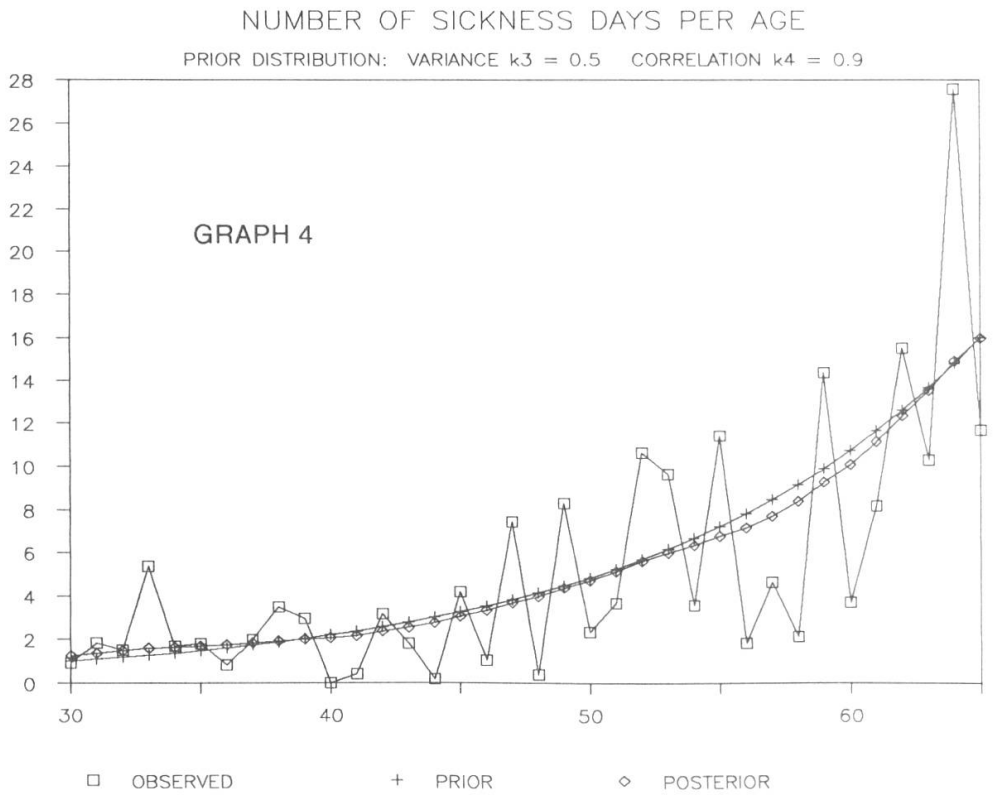
Row 58

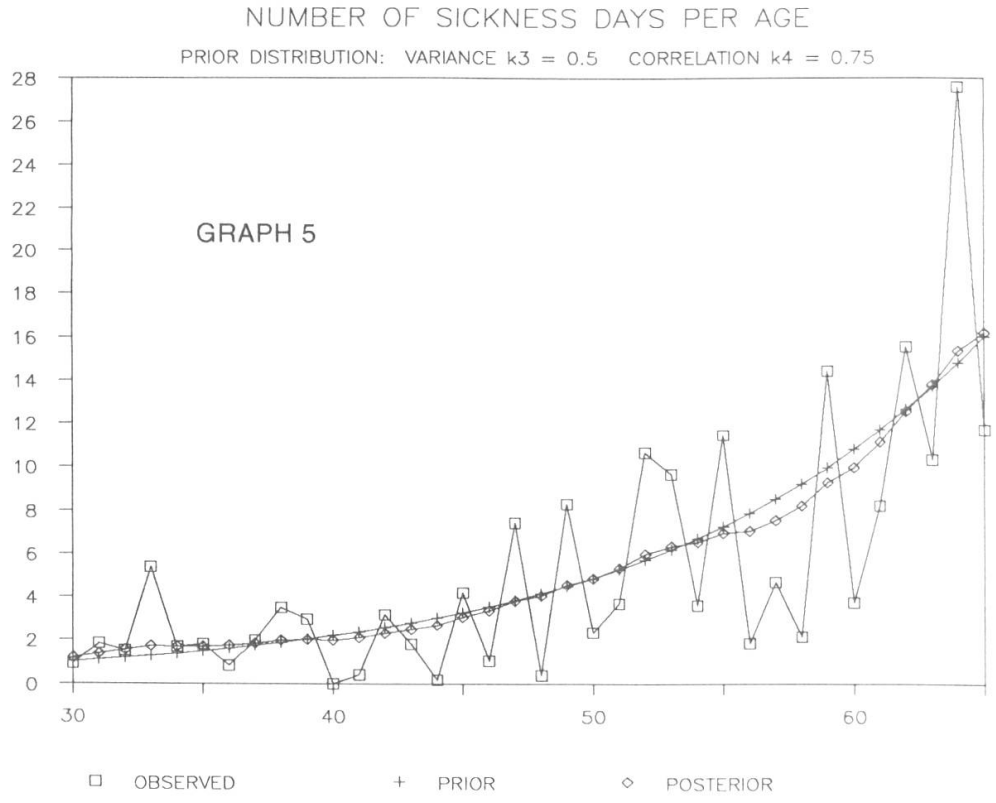
```
 A58:  +65
 B58:  +11.667
 C58:  +0.25
 D58:  +C7*@EXP(C8*(A58-30)/35)
 E58:  +D58-B58
 F58:  +4.4308432443
 G58:  +B58+F58
 I58:  +C23*C10*C11^35
 J58:  +C24*C10*C11^34
 K58:  +C25*C10*C11^33
 L58:  +C26*C10*C11^32
 M58:  +C27*C10*C11^31
 N58:  +C28*C10*C11^30
 O58:  +C29*C10*C11^29
 P58:  +C30*C10*C11^28
 Q58:  +C31*C10*C11^27
 R58:  +C32*C10*C11^26
 S58:  +C33*C10*C11^25
 T58:  +C34*C10*C11^24
 U58:  +C35*C10*C11^23
 V58:  +C36*C10*C11^22
 W58:  +C37*C10*C11^21
 X58:  +C38*C10*C11^20
 Y58:  +C39*C10*C11^19
 Z58:  +C40*C10*C11^18
AA58:  +C41*C10*C11^17
AB58:  +C42*C10*C11^16
AC58:  +C43*C10*C11^15
AD58:  +C44*C10*C11^14
AE58:  +C45*C10*C11^13
AF58:  +C46*C10*C11^12
AG58:  +C47*C10*C11^11
AH58:  +C48*C10*C11^10
AI58:  +C49*C10*C11^9
AJ58:  +C50*C10*C11^8
AK58:  +C51*C10*C11^7
AL58:  +C52*C10*C11^6
AM58:  +C53*C10*C11^5
AN58:  +C54*C10*C11^4
AO58:  +C55*C10*C11^3
AP58:  +C56*C10*C11^2
AQ58:  +C57*C10*C11^1
AR58:  +C58*C10+1
AT58:  -0.000026481
AU58:  -0.0000308209
AV58:  -0.00003713
AW58:  -0.0000458112
AX58:  -0.0000574193
AY58:  -0.0000726958
AZ58:  -0.0000926168
BA58:  -0.000118455
BB58:  -0.0001518611
BC58:  -0.0001949694
BD58:  -0.0002505342
BE58:  -0.0003221053
BF58:  -0.0004142553
BG58:  -0.0005328716
BH58:  -0.0006855326
BI58:  -0.0008819914
BJ58:  -0.0011347997
BK58:  -0.0014601091
BL58:  -0.0018787032
BM58:  -0.0024173256
BN58:  -0.0031103882
BO58:  -0.0040021701
BP58:  -0.0051496462
BQ58:  -0.0066261274
BR58:  -0.0085259446
BS58:  -0.0109704749
BT58:  -0.0141158966
BU58:  -0.0181631673
BV58:  -0.0233708626
BW58:  -0.0300716963
BX58:  -0.0386937773
BY58:  -0.0497879608
BZ58:  -0.0640630417
CA58:  -0.0824310392
CB58:  -0.1060654641
CC58:  +0.8635237062
```


NUMBER OF SICKNESS DAYS PER AGE
PRIOR DISTRIBUTION: VARIANCE k3 = 1.0 CORRELATION k4 = 0.9
GRAPH 1
28
26
24
22
20
18
16
14
12
10
8
6
4
2
0
30
40
50
60
OBSERVED
PRIOR
POSTERIOR


NUMBER OF SICKNESS DAYS PER AGE
PRIOR DISTRIBUTION: VARIANCE k3 = 2.0 CORRELATION k4 = 0.9
GRAPH 2
28
26
24
22
20
18
16
14
12
10
8
6
4
2
0
30
40
50
60
OBSERVED
PRIOR
POSTERIOR


NUMBER OF SICKNESS DAYS PER AGE
PRIOR DISTRIBUTION: VARIANCE k3 = 2.0 CORRELATION k4 = 0.95
GRAPH 3
28
26
24
22
20
18
16
14
12
10
8
6
4
2
0
30
40
50
60
OBSERVED
PRIOR
POSTERIOR


NUMBER OF SICKNESS DAYS PER AGE
PRIOR DISTRIBUTION: VARIANCE k3 = 0.5 CORRELATION k4 = 0.9
GRAPH 4
28
26
24
22
20
18
16
14
12
10
8
6
4
2
0
30
40
50
60
OBSERVED
PRIOR
POSTERIOR


NUMBER OF SICKNESS DAYS PER AGE
PRIOR DISTRIBUTION: VARIANCE k3 = 0.5 CORRELATION k4 = 0.75
GRAPH 5
28
26
24
22
20
18
16
14
12
10
8
6
4
2
0
30
40
50
60
OBSERVED
PRIOR
POSTERIOR

Summary

Bayesian statistics can be used for graduation. The prior probability distribution quantifies the preexisting information about the true position of the data points under examination. This multivariate distribution is transformed according to the observational data into a multivariate posterior distribution. Bayesian graduation is a theoretically sound, but computationally intensive method.
On a spreadsheet Bayesian graduation can be implemented elegantly. Spreadsheets are functional, objectbased programming languages. They are suited for most financial and acutarial problems.
The article first characterizes spreadsheet and explains their language concepts. In the second part Bayesian graduation is demonstrated using an example from health insurance.

Zusammenfassung

Ausgleichungen können mit Hilfe der Bayes-Statistik vorgenommen werden. Die Priori-Verteilung quantifiziert die vorbestehende Information über die wahre Lage der zu untersuchenden Werte. Diese mehrdimensionale Verteilung wird entsprechend den Beobachtungsdaten in eine mehrdimensionale Posteriori-Verteilung transformiert. Die Ausgleichung nach Bayes ist eine theoretisch gut fundierte, aber rechenintensive Methode.
Mittels einer Tabellenkalkulation können Bayes-Ausgleichungen elegant realisiert werden. Tabellenkalkulationen sind funktionale, objektbasierte Programmiersprachen. Sie eignen sich bestens zur Behandlung finanz- und versicherungsmathematischer Probleme.
Im ersten Teil der Arbeit wird die Tabellenkalkulation charakterisiert und ihr Sprach-Konzept erläutert. Im zweiten Teil wird die Bayes-Ausgleichung an einem Beispiel aus der Krankenversicherung demonstriert.

Résumé

Il est possible d'utiliser la statistique bayesienne pour réalisier des opérations de lissage. La distribution a priori quantifie l'information préexistante sur la position réelle de l'ensemble des points considérés. Cette distribution multivariée est transformée en accord avec les données observées en une distribution multivariée a posteriori. Le lissage bayesien est une méthode efficace en théorie, mais conduit à des opérations de calcul numérique nombreuses.
Le lissage bayesien peut être implanté élégamment sur un tableur. Les tableurs proposent des langages de programmation fonctionnels et orienté objets. Ils conviennent à la plupart des opérations numérique des domaines financiers et actuariels.
Le présent article caractérise premièrement la technique des tableurs et explique leurs concepts linguistiques. La seconde partie démontre la mise en oeuvre d'un lisssage bayesien par un exemple tiré de l'assurance-maladie.