TFA 44 (1992-1994) 64-72

A NOTE ON LIFE OFFICE MODELS

BY A. S. MACDONALD, B.Sc., F.F.A.

ABSTRACT

A brief description is given of some problems encountered in designing a model office program. The key to an effective and flexible model lies in the structure which the model imposes on the data. An example of a model used for research is described.

1. INTRODUCTION

1.1 Despite a growing literature which presents results obtained with model offices, there has been little discussion of the models themselves, their features and designs. Every model is different, but all who have tried to design a model office will have discovered similar problems, which this paper attempts to describe.

The examples are based on the author's program MO, a small scale model of a mutual office, intended for research and for education

2. THE SCALE OF A MODEL

2.1 It is useful to pick out two typical uses of a model office; projecting revenue accounts over short periods, and asset-liability modelling over longer periods. The purpose of the model will influence the scale to which it is constructed.

2.2 A large scale model which describes the fine detail of the office would be preferred for the projection of revenue accounts. Typically, it would be capable of handling very many different policies (usually called model points) and it would use a time unit of less than a year.

2.3 For research use, including asset-liability modelling, a small scale model is more suitable. Large numbers of projections may be needed, so a short run-time is essential, and this can only be obtained by sacrificing detail. Moreover, the "accuracy" delivered by a large scale model is rarely warranted by the assumptions, and indeed it may impede the interpretation of the results.

2.4 Grouping the office's data creates problems of consistency. The aim is to find representative model points so that the order in which grouping and projection are performed is immaterial (see Figure 1).

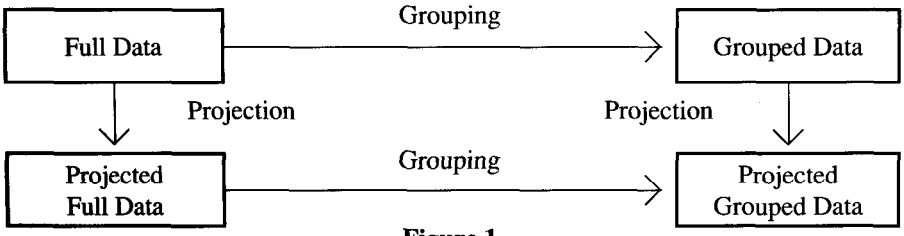

Figure 1

2.5 In many respects, a small scale model is harder to design, because it is less practical to rely on human supervision, and most if not all of the model must be automated. It is thus harder for the designer to leave problems for the user to solve. The following remarks mostly refer to small scale models.

3. DETERMINISTIC OR STOCHASTIC?

3.1 The uses of stochastic models are outside the scope of this note, but a few comments may be made in support of further research:

(a) Stochastic models must not be used blindly, but actuaries are not yet familiar with the techniques needed (for example, time series models).

(b) It is tempting to discard results which fall outside our experience, yet many "breaks with the past" were unbelievable until they happened.

(c) Stochastic models complement rather than replace deterministic models. Deterministic models help to interpret stochastic results.

(d) It would be unwise, given the current state of the art, to place much emphasis on absolute numbers obtained from stochastic models. However, comparative studies are often possible and these may have more to tell us.

(e) The features of individual models may be disputed, but the results from stochastic and deterministic models are often qualitatively different. Too much concern over details may obscure important questions, such as whether a qualitative effect is robust to changes in the model.

3.2 The main design problems associated with stochastic modelling are not, in fact, associated with the random nature of the inputs, but with the need for automated decision-making (called the "electronic actuary" by Ross [3]). Algorithms must be devised which respond dynamically to widely varying inputs, and in this respect, a stochastic model is no different from a dynamic, deterministic model.

3.3 Two problems have to be solved to make such algorithms work. First, a way must be found to gather the information which the algorithms need to make decisions; for example current yields may be needed to carry out a valuation. Second, the algorithms have to be applied to the appropriate parts of the office; for example, different bonus rules are needed for different with-profit funds. The solutions to both problems depend on how the data are structured within the model.

4. PROFIT TESTS v OFFICE MODELS

4.1 A first attempt might follow the obvious route from profit tests to model offices, aggregating the cash-flows found by profit testing each model point. This may be called the "policy-based" or "policy year" approach. It is described in Beard, Pentikäinen and Pesonen, Chapter 8 [1].

4.2 This approach leads to difficulties in implementing dynamic algorithms, which may be summed up as (i) lack of feedback, (ii) the need to make prior assumptions about outputs, (iii) failure to deal with cross-subsidy, (iv) failure to deal with global items and (v) lack of flexibility.

4.3 Few of the major decisions made in a life office relate to individual policies; as a rule they relate to aggregates of policies. For example, asset allocations and bonus rates may be decided at the with-profit/non-profit level. Such decisions are based on information gathered ("feedback") from the model in respect of the appropriate aggregates, to which a profit test clearly has no access.

4.4 In the absence of feedback, each profit test must be based on assumptions which may not be consistent with the larger picture. Moreover the effect of dynamic inputs on asset allocation, bonus rates and liabilities ought to be among the outputs from the model, so there is a real loss of utility if they must be treated as inputs.

4.5 It is often not true that the whole life office is equal to the sum of its parts. There are synergies (of tax, for example), and cross-subsidies (in the calculation of resilience reserves, for example). A policy-based model can only take these into account by making approximate adjustments when the profit tests are aggregated into global cash-flows, at which time the results may suggest changes which should have been made to the underlying profit tests.

4.6 Global items not depending on the in-force policies may affect policy cash-flows but lie outside the scope of the profit tests. Free assets, for example, may have a significant impact on asset allocation and statutory minimum valuations. Such influences further the aggregation of the policy cash-flows and may raise doubts about the cash-flows themselves.

4.7 These flaws might be tolerable if it were easier to build a model office out of existing profit testing software than to start afresh. However, profit tests may not be well suited to the needs of a model office. For example, if financial conditions are changing, each tranche of new business requires a separate cash-flow projection. Although such inflexibility is not a fundamental flaw, and may have software solutions, it suggests that the route from profit tests to model offices is not free of obstacles.

4.8 A different approach is to project the cash-flows of the whole office year by year. This may be called the "office-based" or "calendar year" method. Obtaining feedback in respect of aggregates of policies, or global items, is then simple and direct. This approach overcomes all of the flaws of the policy-based model, and its outputs should therefore be more reliable and easier to interpret. On the other hand, it is less likely to make use of existing software and may require greater effort in the first stages. Since it does not break the office down into very small units, it requires more capacious hardware, but this problem tends to diminish over time.

4.9 No studies have been published which would validate the results from a policy-based model by comparing them with those from an office-based model.

5. THE STRUCTURE OF A MODEL OFFICE

5.1 A life office has structure. Its business may be decomposed at several levels into funds or lines of business which reflect tax or operational differences. The task of modelling the office is easiest if the data structures in the model mirror the structure of the office.

5.2 A first step in the specification of data structures for a model is a minute dissection of the office to find all those "atomic" quantities which must be represented by single variables; for example the number of policies of a given type at a given time. A second step is the combination of "atomic" data into collectives, or sub-funds, which map the model's data onto the structure of the office.

5.3 Such objectives are needed for several purposes:

(a) The analysis of results is likely to be at the level of funds rather than of model points.

(b) The decision algorithms need feedback from relatively high levels.

(c) Each decision algorithm must be applied to a specific section of the office's business.

These requirements suggest that the analysis of office data into sub-funds may take place at several levels. For example, in the program MO there are five levels: the whole office, tax funds, with-profit/non-profit funds, lines of business, and policy types (distinguished by term and age). Each level can form the basis of reported results, feedback for algorithms, or the application of algorithms. MO structures an office as a strict hierarchy (or tree-like structure), the main advantage being that much of the processing can be carried out recursively, but other structures are possible.

5.4 "Atomic" items must be represented physically by numeric data, probably gathered into structures or arrays for convenience, but there are several ways in which the sub-funds may be represented, depending on the language being used. An economical method is to use "pointers", if they are supported. The physical object representing a sub-fund merely consists of a collection of pointers to the location of the sub-funds at the next level down, until the "atomic" data are reached. This scheme is not so economical, however, when calculations must make use of an aggregated quantity, such as the total asset shares. Unless the sub-funds have access to data, such a number must be recomputed whenever it is needed. Therefore the physical object representing a sub-fund may be enlarged to store, or otherwise have access to, the most useful aggregated asset, liability and cash-flow data.

5.5 The most important features of sub-funds are those which they all have in common. For example, the total of the asset shares is a potentially useful datum in any sub-fund. It is therefore sensible to provide a common physical representation for this property. The benefits of such abstraction are far-reaching, especially in the longer term; it is the key step in turning a model of a specific office into a general purpose program capable of modelling different offices:

(a) The code which performs the basic operations on sub-funds, for example to pick out the total asset shares whenever needed, is general purpose and need only be written once.

(b) Changes during later maintenance can often be restricted to a single section of code.

(c) The addition of a type of sub-fund to the model, for example to represent a new line of business only requires a new code to deal with the new features of the sub-fund.

(d) It is easier to cope with different office structures, since a particular office may be modelled by selecting standard building blocks in a particular order.

5.6 (a) – (d) above are reminiscent of the benefits claimed for object orientated programming languages, but it is not necessary to use such a language to obtain them. It is only necessary that a language has the facility to create, keep track of, and dispose of data structures at run-time rather than at compile-time; some languages such as C do so explicitly, while others such as APL do so implicitly.

5.7 Each sub-fund in MO has the physical representation shown in Figure 2.

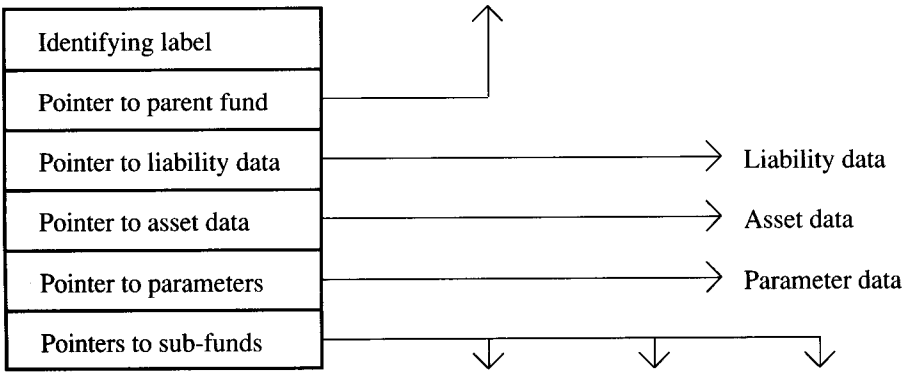

Figure 2

The "identifying label" allows the sub-fund to identify itself to the program, and this is the only essential difference between the representation of any two sub-funds. The liability and asset data structures are of a standard form, so for the most part the program can process a sub-fund without knowing what it represents. If necessary, it can interrogate the object to find out its type and, depending on the result, carry out an action or move up or down the office structure. The use of standard data structures inevitably brings some redundancy, but cost is well worth paying in return for flexibility.

5.8 An important feature of Figure 2 is that the structures shown are maps, which point to the location of numeric data but do not contain numeric data. The numeric data need not be in memory; for each pointer shown two pointers are provided; one into memory

and one into a disk database, so that the state of the office can be recorded at any time and retrieved later for analysis or to provide a starting point for further projections.

6. SYNERGIES AND CROSS-SUBSIDIES

6.1 Cross-subsidies arise in a fund when one sub-fund makes a profit and another makes a loss. If the data are structured on several levels, then cross-subsidies become apparent when the outputs are analysed by sub-fund.

6.2 Most cash-flows are additive, meaning that the cash-flows in the sub-funds at one level add up to the cash-flows at the next higher level. For investment cashflow and tax, however, this additivity may break down. If the tax computation is applied at the lowest levels in the office, each contract earns its own tax relief or accumulates its own unrelieved expenses on a stand-alone basis. Applying the tax rules to the aggregated income and expenses, synergies may arise; the tax charge at higher levels may be less than the total tax charges at lower levels. Over time, these differences will accumulate so that the assets will no longer be consistent at different levels. Instead of being a drawback, the "inconsistencies" which arise are a measure of the tax savings due to synergy, and hence are useful. For many purposes, they can be eliminated by imposing a predetermined tax basis.

6.3 A different sort of "inconsistency" may arise if different strategies are used at different levels in the office. For instance, the office might base terminal bonus on asset shares calculated using a notional investment history. It may or may not base its overall investment strategy on the underlying asset shares. If not, there will be actual and potential mismatching surpluses, which will appear as differences between the assets and total asset shares in a fund.

7. CONTROL PARAMETERS

7.1 A model is controlled by a large number of parameters. Some of these will represent economic conditions and perhaps the effect of competition. The majority may represent the office's strategies and its responses to the outside world.

7.2 A physical mechanism must be devised which allows different sets of parameters (such as different valuation bases) to be associated with different sub-funds (see 5.3). One method is suggested by Figure 2. In MO, parameters are first grouped into logical sets, such as valuation bases, or bonus rules. Then, sets of parameters are attached to suitable sub-funds; for example:

Investment conditions	<——————>	Whole office
Tax basis	<——————>	Tax fund
Asset allocation	<——————>	With-profit/non-profit fund
Valuation basis	<——————>	Policy class
Premium basis	<——————>	Policy type

7.3 Each object representing a sub-fund (Figure 2) can access the addresses of its parameter data, both in memory and in a disk database. The physical form of the parameters is a secondary matter. In MO, the parameters exist as binary entries in a database; changes are made via a screen interface. An alternative method, more flexible in some ways, is to store in each sub-fund the names of text files containing parameter data; changes to parameters can then be made with a text editor, but the program has to parse the text files.

7.4 At a more detailed level, the mechanism should allow changes to be made to any parameter at any time. As a projection proceeds, the program must locate the most recent version of each parameter set. The asset and liability (output) data may appear to be free of this complication, but the problem of writing time-stamped output to a disk file is very like that of reading the most up-to-date parameters from a disk file. In MO, both parameters and output are handled using linked lists indexed by time. Use of the same general purpose mechanism makes maintenance much easier, and program code more transparent.

7.5 The most detailed calculations are carried out at the lowest level, that of the individual policy of a particular duration in force. These calculations must have access to virtually every parameter in order to update asset shares, allot bonus, deal with movements and so on. The program locates parameters by searching up the office's hierarchy until it finds what it needs. For example, to allocate bonus to a contract, the program will search the lowest (policy type) and next lowest (policy class) levels before finding the bonus parameters in the third (with-profit fund) level. Therefore, any bonus changes need be made in only one place, and the change will automatically flow down to all lower levels in the fund. This benefit is not trivial, as will be appreciated by anyone who has ever tried to ensure consistency across a number of profit tests.

7.6 Moreover, it is possible to associate parameters of the same type with more than one level in the fund. For example, if there were some reason to use different bonus rates in a sub-fund of the with-profit fund, a set of bonus parameters could be attached to the object representing that sub-fund. Any contracts contained in the sub-fund would find these parameters first and use them in preference to those at the higher level.

8. ALGORITHMS

8.1 Algorithms for asset allocation, bonus distribution and pricing have been described elsewhere (see Macdonald [2], Ross [3], Ross & McWhirter [4]). The main point here is to note how the suggestions made above meet the requirements in 3.3.

8.2 Feedback is provided by the access to aggregated outputs at each level in the hierarchy. The only problem is to ensure that any datum needed by an algorithm is calculated before the algorithm is applied. Sometimes each of two algorithms need an output calculated by the other, and there is deadlock. Two solutions are (i) to use iteration, with the danger that the solutions do not converge, or (ii) to introduce a time-

lag, so that one algorithm uses output from the previous time period. MO only uses iteration when the asset allocation is driven by solvency.

8.3 The application of different algorithms to different parts of the office is determined by the attachment of parameter sets to the sub-funds at one or more levels, and is therefore also automatic. Note that it is not necessary for the feedback to an algorithm to be supplied by the sub-fund to which it is applied.

9. ANALYSIS OF OUTPUTS

9.1 Provided the main items of asset, liability and cash-flow data for each sub-fund can be retrieved from the disk, the printing of reports requires no more than a list of which sub-funds to include. Two useful outputs are:

(a) time series output, which displays in each table a chronological series of results, and

(b) the results at a single time, in which each sub-fund shown is split up into its own sub-funds, following the structure of the office.

9.2 Problems do arise in the analysis of stochastic outputs, mainly because of the sheer volume of data. It is sensible for the program to allow a choice of outputs to be recorded for later analysis. The main decision is whether the program itself should analyse the outputs, or whether this is better left to other programs.

9.3 The main requirements are:

(a) to analyse the distributions of time series data;

(b) to compare different sets of projections made using the same set of stochastic scenarios;

(c) to inspect the results of individual projections;

(d) to obtain graphical output.

Given the range of statistical packages with good graphics, (such as S+) there is little reason to re-invent the wheel, and the method adopted for MO is to write the selected data to a text file at the end of each scenario, in a specified format. It is then straightforward for other programs to extract data for statistical analysis. It would perhaps be convenient if the model office program could analyse its own outputs, but this seems to be less flexible in the long run.

REFERENCES

(1) R. E. Beard, T. Pentikäinen, E. Pesonen (1984). *Risk Theory*. Third edition, Chapman & Hall, London.

(2) A. S. Macdonald (1993). *What is the Value of a Valuation?*. Proceedings of the 3rd International A.F.I.R. Colloquium. Rome, 1993

(3) M. D. Ross (1989). *Modelling a With-Profits Life Office*. Journal of the Institute of Actuaries Vol.116 p.691.

(4) M. D. Ross, M. R. McWhirter (1991). *The Impact on Solvency and Policy Results of the Valuation Regulations Restrictions on Equity Yields*. Unpublished paper.