

Operations manual for optimising system dynamics models using genetic algorithms

Patrick E. McSharry^{a,b,c,*},

^a*Department of Engineering Science, University of Oxford, Oxford, OX1 3PJ, UK*

^b*Mathematical Institute, University of Oxford, Oxford OX1 3LB, UK*

^c*Commissioned by The World Bank, 1818 H Street, NW, Washington, D.C. 20433*

Abstract

System dynamics models provide a detailed mathematical description of the complex interactions between numerous variables. Software packages, such as *iThink* and *Vensim*, allow policy makers to easily adapt, implement and explore these models. By testing various scenarios through an analysis of the model with different parameter values it is possible to investigate and visualise the long-term effects of changes that could be implemented over the short-term. Genetic algorithms (GA) are shown to provide an efficient and accurate method for identifying optimal scenarios from among the vast number of possible scenarios that are available. The combination of the GA parameter search and human intuition can be utilised to arrive at better strategies for government policy. This approach to optimisation is demonstrated using a model of the malaria-control program in Bolivia.

* Department of Engineering Science, University of Oxford, Oxford, OX1 3PJ, UK
Email address: patrick@mcsharry.net (Patrick E. McSharry).

1 Introduction

The availability of user-friendly packages, such as *iThink* [1] and *Vensim* [2], allows a wide range of practitioners to construct and implement system dynamics models. Their easy to understand building-block approach and visual appeal also allows non-experts to easily adapt and employ these models. At present, however, the ability of these packages to provide optimal solutions is limited.

The simplest procedure for identifying optimal solutions, the one currently used by the most advanced system dynamics software package, is to divide the range of each parameter value into discrete points (forming a grid) and to test each combination in turn. This procedure works for systems with a small number of parameters but quickly becomes intractable for larger systems. Another feature of complicated, nonlinear system dynamics models is that the parameter space often exhibits multiple equilibria and many optimisation techniques tend to give sub-optimal solutions.

In this paper, a stochastic optimisation technique based on a Genetic Algorithm (GA) is proposed as a means of addressing both (i) the dimensionality and (ii) multiple equilibria problems. Furthermore, this approach to optimisation does not rely on specific information about the problem (gradients, linearity or continuity) and has been shown to be highly efficient in comparison to competing optimisation techniques. In addition, the GA gives a set of solutions, which may be used to analyse the robustness of the optimal solution.

Providing policy makers with a means of exploring models built in order to facilitate their decision-making gives them the chance to better acquaint themselves with the likely effects of making small changes to the current system or completely switching to an alternative one. The large number of possible changes implies that it would take a long time for one person to investigate all the different scenarios. In contrast, in this paper, a technique for providing a set of optimal scenarios is proposed. Each scenario is specified by a collection of parameter values, to help guide the policy maker towards both relevant and viable outcomes. This approach will still allow for sufficient human intervention without losing the policy makers' interest because of the need for exhaustive searches.

Operating the Genetic Algorithm

The task of constructing and optimising a system dynamics model can be separated into four phases: (a) data collection, (b) *iThink* model construction,

(c) *iThink* model conversion to *Matlab* and (d) GA optimisation. All four phases are explained in the following sections.

Data collection

The first step to obtaining a realistic model is to collect all the relevant data. This involves an analysis of the different dynamical variables (stocks in *iThink*) and the influences between them. For this reason there should be some iteration between this step and the model construction. An initial value will be required for each dynamical variable. This phase usually involves contact with experts in the field to establish the important stocks. Furthermore many of the parameters may vary slightly over time, e.g. prices. This information should also be provided to the model builder.

Model construction

The *iThink* package provides an intuitive, dynamic and graphic modelling approach. It facilitates the integration of a range of sub-models and sub-processes. The simplicity of the graphical interface and the ease of integration means that a number of different people can be involved in developing different sub-models at the same time. A thorough description of the *iThink* model for the malaria-control program is given in [3].

Model conversion from iThink to Matlab

The *Matlab* program `ithink2matlab.m` was developed in order to automate the conversion of a given *iThink* model into *Matlab*. This may be achieved by carrying out the following steps:

- construct an *iThink* model without using characters \$, &, % or accents
- view the underlying equations of the *iThink* model
- choose **Select All** from the **Edit** menu
- copy and paste into the Microsoft package *Wordpad*
- select **No Wrap** from **View > Options > Word Wrap**
- save as an ascii file, e.g. `filename.txt`

Within the *Matlab* environment, the execution of the command:

```
ithink2matlab(filename, optparam)
```

performs the task of reading in the *ithink* model from `filename.txt` and uses the parameters listed in `filename.par` to generate a *Matlab* model `filename.m`

and matlab derivative file `derivsfilename.m`. In addition, it generates a matlab script `optfilename.m`, which is a function of the parameters and gives the final value of the specific parameter chosen for optimisation.

The malaria-control program

In the case of the malaria-control program, the file `malaria.m` may be executed using

```
F = malaria(a)
```

where `a` is a vector containing the values of the 17 financial investments. The program runs the model, evolving all the stocks over time. Inside the file `malaria.m`, a call to `optmalaria.m` calculates the values of the variable selected for optimisation, API. The output `F` gives the value of the API at the final time step of the model simulation.

In summary, the file `malaria.m` calculates the final value of the API for any given investment scenario that is encoded in the vector `a`. The aim is now to minimise `F` with respect to `a` while maintaining the constraint that the sum over all the values in `a` must equate with the financial budget. The GA with built-in budget constraint may be operated using `gabudget.m`.

This genetic algorithm (GA) is then used to optimise the results of running the script `malaria.m` with respect to different input parameters. This is achieved by running the command

```
xopt = gabudget('malaria',Xlims,xstart,c)
```

where `xopt` is the optimal GA solution. The matrix `Xlims` gives the range of each of the 17 parameters used in the optimisation, `xstart` is the best-guess starting solution and `c` is the budget constraint.

The ability of `gabudget.m` to accept a *best-guess* starting solution, x_{start} is particularly useful in situations where the purpose of carrying out the optimisation is to test whether a given policy can be out-performed. If the GA returns the same optimal solution, x_{opt} as the best-guess starting solution, then the policy-makers can rest assured that their current strategy is optimal. On the other hand, if the GA returns a different optimal solution, this could then be implemented in order to improve the efficiency of the policy.

2 Conclusion

There are numerous challenges surrounding the optimisation of system dynamics models. A range of different possible dynamical regimes, fixed points,

limit cycles and chaos, complicate the process of optimising the output of the model. The number of parameters over which the optimisation takes place in many system dynamics models suggests that “brute force” approaches are not adequate. In addition, it may be extremely difficult, if not impossible to obtain information about the gradient of the model. For these reasons, a stochastic optimisation technique based on a Genetic Algorithm is suggested for determining a collection of solutions.

A system dynamics model for malaria-control in Bolivia has been used to demonstrate this approach [3]. A comparison of the 1998-2001 policy, the human-optimised model and the GA-optimised model demonstrated that the GA approach can outperform the others in terms of providing more accurate results and being more efficient [4]. An analysis of a collection of solutions in the form of investment strategies was used to determine the robustness of the optimal solution. This showed that the GA optimal solution is robust to small changes in the investment allocations.

The combination of user friendly system dynamics packages such as *iThink* and *Vensim* with a Genetic Algorithm for determining optimal solutions has important implications for policy making. Having easy access to relevant solutions makes it easier for the policy makers to explore and experiment with the model while incorporating their own intuition before deciding on a final plan of action.

References

- [1] www.iseesystems.com.
- [2] www.vensim.com.
- [3] J. Newman, Velasco M. A., L. Martin, and A.-M Fantini. A system dynamics approach to monitoring and evaluation at the country level: an application to the evaluation of malaria-control programs in bolivia. In *Proceedings of the Fifth Biennial World Bank Conference on Evaluation and Development, “Evaluating Development Effectiveness: Challenges and the Way Forward*, Washington, D.C., 2003. Operations Evaluation Department.
- [4] P. E. McSharry. *Optimisation of system dynamics models using Genetic Algorithms (GAs)*. World Bank, La Paz, Bolivia, May 2004.