GMSE: an R package for generalised management strategy evaluation

Supporting Information 2

A. Bradley Duthie, Jeremy J. Cusack, Isabel L. Jones, Erlend B. Nilsen, Rocío Pozo, O. Sarobidy Rakotonarivo, Bram Van Moorter, and Nils Bunnefeld

2017-11-06

Extended introduction to the genetic algorithm applied in GMSE

A genetic algorithm is called in the predefined GMSE manager and user models to simulate human decision making. As of GMSE version 0.3.1.9, this includes one independent call to the genetic algorithm for each decision-making agent in every GMSE time step. Therefore, one run of the genetic algorithm occurs to simulate the manager's policy setting decisions in each time step (unless otherwise defined through non-default `manage_freq` values greater than 1), and one run occurs to simulate each individual user's action decisions in each time step (unless otherwise defined through non-default `group_think = TRUE`, in which case one user makes decions that all other users follow identically. Each run of the genetic algorithm mimics the evolution by natural selection of a populaiton of potential manager or user strategies over multiple generations, with the highest fitness strategy in the terminal generation being selected as the one that the manager or user decides to implement. For clarity, as in the main text, we use 'time step' to refer to a full GMSE cycle (in which multiple genetic algorithms may be run) and 'generation' to refer to a single, non-overlapping, generation of potential strategies that evolve within a genetic algorithm (see Figure 1 of the main text). Below, we explain the genetic algorithm in extended detail as it occurs in GMSE v0.3.1.9 (future versions of GMSE might expand upon this framework). We first explain the key data structure used, then provide an overview of the processes of crossover, mutation, cost constraint, fitness evaluation, tournament selection, and replacement. We then explain the fitness functions of managers and users in more detail.

Key data structures used

General overview of key aspects of the genetic algorithm

Crossover

Mutation

Cost constraint

Fitness evaluation

Tournament selection

Replacement

Detailed explanation of manager and user fitness functions

Manager fitness function

User fitness function

Thanks for the clarification regarding the equation. I'll try to answer as best as I can – apologies if this has been unclear. At the broadest scale, the equation for user fitness would be on L367 in the strategy_fitness function (https://github.com/bradduthie/gmse/blob/master/src/game.c#L376). Here's what's going on: Users are predicting how their actions will change the quantities of things in the model (either resources or landscape output), and these changes are individually multiplied by the users' utilities for that thing. The change multiplied by utility for each thing is summed across all things to get a value for fitness. Note that positive change times positive utility, and negative change times negative utility, will increase fitness (i.e., increasing the thing users want more of and decreasing the things they want less of). Hence, an equation describing user fitness would be the below,

$$F_{user} = \sum_{i=1}^{N} \Delta A_i \times U_i$$

.

Where F_{user} is user fitness, N is the total number of things that might be of interest (at the moment $N = 2$ in GMSE, one resource and, potentially, one landscape value), ΔA_i is the change in the abundance of thing i, and U_i is the utility of thing i from the perspective of the user (apologies for the LaTeX code – attached a PNG of the conversion). I want to stress though that I would not consider this equation to be central to the GMSE framework – if someone else has a better approach for defining fitness, or defining any of the terms listed above, or wants to expand upon it to include new things, then that would be awesome! The above just works well as a heuristic tool to get users to act in such a way as to maximise their interests in harvesting or getting more crop yield (as is my intent), but it's not based on first principles and I don't claim it to be particularly special.

The values of ΔA_i are calculated for resources and the landscape in the functions res_to_counts and land_to_counts, respectively (and U_i is specified a priori in the model depending on other parameters – namely land_ownership). Again, a bit of heuristic is needed here because there cannot be any perfect way of exactly predicting how a users actions will increase or decrease resources – there are too many complex factors (e.g., behaviour of other stakeholders, demographic stochasticity, movement of resources on the landscape, and interactions between resources and the landscape). Even if we could include all of these things somehow, it would be a bit unrealistic in that real stakeholders would never have this much information. The predicted direct effect of actions on resources is shown in lines 268-272 (https://github.com/bradduthie/gmse/blob/master/src/game.c#L268), and the array 'jaco' (a sort of Jacobian matrix) accounts for interactions between landscape and resources on line 286. Something similar happens in the land_to_counts function. The manager's genetic algorithm works in a similar way (the above equation applies), but with the need to dynamically update utility values based on current resource abundance, and to account for the predicted actions of users in finding ΔA_i.