Appendix

Assessing Simulations of Imperial Dynamics and Conflict in the Ancient World

Jim Madge[1], Giovanni Colavizza[2], James Hetherington[1], Weisi Guo[1,3], Alan Wilson[1]

[1] *The Alan Turing Institute*
[2] *University of Amsterdam*
[3] *Cranfield University*

Implementation

We implemented the model from Turchin et al. (2013) from scratch in Python. This implementation of the model is written in an object-oriented style, which aids in creating clear definitions of entities in the simulation such as polities and individual cells. Abstract classes were also created for analysis of the simulation in the form of an accumulator class for collecting data over the course of a simulation (such as imperial density), and a correlator class for comparing an accumulator against external data (such as historical population). The advantage of this design is that it greatly simplifies both making modifications to the simulation model and implementing new analyses. The trade-off is that the code is not optimized to reduce simulation time and is likely considerably slower than a more traditional implementation using contiguous arrays of data. However, as individual simulations are short (2–3 minutes) we felt that the small burden of extra calculation time was more than compensated for by a clear and versatile code.

Input for the simulation software is provided in the form of YAML files. These are human-readable text files, which again contributes to our approach of maximizing clarity and usability. The creation and parsing of YAML files are also well facilitated in a range of programming languages through free and open-source libraries.

In implementing the model, a number of issues arose. First, two simulation parameters were changed from the values stated in the paper and supporting information, following Bennett (2016). ε_{max}, a parameter determining the probability of ethnocide occurring, was set to 2 compared to the value of 1 stated in the supporting information and Δ, the amount that the sea attack distance is incremented each step, was set to 0.0025 instead of 0.025 as in the supporting information. All other parameters are equal to the stated values.

Corresponding author's e-mail: g.colavizza@uva.nl

Citation: Madge, Jim, Giovanni Colavizza, James Hetherington, Weisi Guo, and Alan Wilson. 2019. Assessing Simulations of Imperial Dynamics and Conflict in the Ancient World. *Cliodynamics* 10: 99–114.

Second, the spread of military technology occurs in a manner different from that described in Turchin et al. (2013). Rather than each agricultural cell attempting to spread technology to a random neighbor each turn, the spread is actually very closely tied to attacks. In particular, attempts to spread military technology only occur after an attempt to attack, and always occur in the same direction of the attempted attack. The technology spread is attempted whether or not the attack was successful, or even if the attack didn't happen (i.e., when a cell attempts to attack another cell belonging to the same polity). Otherwise, the paper's description of technology spread is consistent with the simulation, i.e., the random choice of a technology to attempt to share and the spread rate a.

Third, some peculiarities are associated with sea attacks between littoral cells. To be consistent with the original simulation, it is necessary that each littoral cell may attempt to attack itself when selecting a littoral neighbor to attack. These attacks are always rejected as cells are forbidden from attacking other cells belonging to the same polity. Furthermore, the distance between littoral cells, d_{sea} in Turchin et al. (2013), is the Euclidean distance between the centers of the two cells. The distance calculated does not necessarily correspond to a valid sea route between the two cells as it may cover land. Indeed, this means it is possible for any two littoral cells to attack each other even if there is no sea route between them (e.g., cells on the Caspian Sea attacking those in the Persian Gulf). However, as the sea attack distance remains small throughout the simulation (particularly in the light of the small increment described above), this is not considered to be a large source of unrealistic attacks.

Fourth, all cells must attack each of their neighbors with equal probability. Practically this is ensured by asserting that cells attack with equal probability in all four directions even if one or more of those directions is not a valid target for an attack, such as a desert cell or a cell belonging to the same polity.

Finally, the order of events in each step was not described in the original work. The order in Turchin's simulation code, and our own, is:

1. Each cell (in a random order) attacks and attempts to spread military technology to its target
2. Mutation of ultrasocial trait vectors in each cell
3. Polity disintegration