Netlogo Village Economy Model

This model represents a simple village economy. The model entities are the fields, taps, people, households, water storage, villages and cities. Fields and people belong to households. Fields can be connected to water storage that supplies irrigation. Households can be connected to taps that supply potable water. Villages contain households, taps and water storage.

The model was used as part of a research project dissemination meeting.

HOW IT WORKS

Entities
Fields: Grow agricultural crops as a function of water and labour applied.
Taps: Provide potable water to attached households.
People: Are available to work on fields in the village or can migrate to a city and pay remittances.
Households: Contain people as members. Own fields. Pay subsistence costs for members living in the village, pay maintenance cost share for fields connected to water storage. Allocate labour to own fields and through labour market to other fields. Decide if a member is to move away. Buy and sell fields.
Water Storage: Store water from a constant flow spring and a share of rainfall. Make water available to fields at start of growing season.
Villages: Contain village households, storage and taps. Track level of social capital. High social capital makes it easy to secure contributions from village members for construction, maintenance and repair of water storage.

Time Sequence

Time is divided into the kharif (rainy) season, the rabi (dry) season, and the policy period.

In both the kharif and rabi seasons, the same sequence is followed. First, the crop is set for each farmed field, based on season and access to irrigation. The labour market is then used to allocate labour to the fields, where labour demand is based on assuming that there will be sufficient water. Precipitation then comes, water storage systems are filled, and water is made available to the fields which are connected to storage. The crops are then grown, revenue is calculated, and seasonal household expenditures are paid. At the end of the season, households also decide if a member should migrate.

Policies are evaluated after the rabi season is concluded. Depending on the policy choices in place, at this time any new storage and new taps are built, failed storage is repaired, field level technology enhanced, social capital is increased, and direct cash payments made to households. All of these activities can be based on greatest need or distributed among the relevant entity at random. Greatest need was used for the simulations.

Initialization

The model reads information from a configuration file. The name of the configuration file is entered in an input box in the top left of the interface pane. The information contained in the configuration file defines most of the variables relevant to the model. This format was chosen so that it would be easy to adjust variables and a host of initial conditions. One option in this configuration file is to specify a Digital Elevation Model (DEM) that can be loaded to create the landscape on which the model runs.

Consult this file for the detail of the fields that are used and the information required.

HOW TO USE IT

Setup loads information from the configuration model and sets up the virtual world.
Go proceeds through one year.
Go to 10 and **Go to 20** run the model until 10 and 20 years have passed.
logFile contains the name of the log file to which detailed output is written. The log can be reset, and logging can be turned on and off.
The sliders **ltStorage**, **stStorage**, **repairStorage**, **irrigTech**, **drinkingWater**, **scBuilding**, and **emergeRelief** control the share of the policy budget that is applied to each of these activities. The corresponding switches with the subscript **_rnd** control whether the interventions are applied to randomly selected fields, households, or villages, or if they go to those with the greatest need. Specifically, the activities are building new long term storage, building new short term storage, repairing existing storage that isn't functioning, investing in improved field level water efficiency, installing driking water taps, building social capital, and paying a subsidy directly to households.
policyBudgetActive and **policyBudgetSpendShare** control whether policy interventions are active, and if they are what share of the policy budget is applied. The total policy budget is specified in the configuration file.
precipScale1 through **precipScale20** control the precipitation scaling factor for each of the years one through twenty. If the simulation control is set active, then the precipitation for a particular year will be read from the appropriate slider. The **simLtStorage** through **simEmergRelief** sliders control the levels of the policy variables during the policy period - years 11 to 20 - applied when a simulation is being run. All these simulation controlers are not used if the simulation switch is off. In this case, information read from the configuration file and entered in the relevant controlers is used.

The output panels show the averages for each village being modelled. The landscape is shown in the lower part of the interface. Differences in patch color reflect the elevation of the patch.

THINGS TO NOTICE

How policy choices impact on migration rates, social capital, and average income and wealth.

THINGS TO TRY

For the running model, most of the action is from the policy sliders. There is also much that can be explored by playing with the configuration file.

EXTENDING THE MODEL

* Local migration with return.
* Remittances diminishing over time.
* Migration cost function of previous migration.
* Transportation costs and investment in roads.
* Input markets.
* Human capital (skills, eduction).
* Education as a policy option.
* Age structured population.
* Gender structured population.
* Hydrologic detail.
* Environmental effects.
* Flooding risk as well as drought risk.

NETLOGO FEATURES

The main coding challenges lay in figuring out how to use a configuration file and in organizing the different entities. Reading the configuration file was quite easy, but a configuration file language had to be chosen that would make it simple to build a file and for NetLogo to process it. See the configuration file itself for how this was solved.

Organizing the entities was a challenge because NetLogo is not an object oriented programming environment. Multiple code files were used to approximate the organization of objects, but agent specific local variables could not be cleanly defined in the agent file itself. This made coding somewhat complicated.

CREDITS AND REFERENCES

Code generated by:

Dr. John Janmaat
Department of Economics
I.K. Barber School of Arts and Sciences
The University of British Columbia
Kelowna, British Columbia, Canada
john.janmaat@ubc.ca

Contact Dr. Janmaat for more extensive documentation and research results generated using this model.