Zero-Intelligence Economic Trade

By: Mike Tamillow

The problem space

Trade arises as a mutually beneficial aspect of healthy economies. Market prices arise from trading, not vice-versa. Can we make efficient markets with agents who never modify their bidding or asking prices?

Model Rules

To use:

- All traders are divided into two categories, producers and consumers.
- Users set the number of each type, and how much cash and asset to start each trader with.
- Users set a cost, utility, return-rates, economic-rates, trade-rates, account-limit (and an option for volatility or trend).

In run:

1. During setup all traders pick a price to trade at in the simulation that remains unchanged.
2. Producers place sell orders, consumers place buy orders.
3. Markets clear all orders, find a trade price and volume that has been filled. (using Auction)
4. Agents update location in the world by account value vs. account-limit
5. Agents perform economic actions, producers spend cash to get asset, consumers expend asset to get cash.
6. If volatility or trend modify utility and cost
7. Stop if no trading 20 ticks, else go again.

Auctions as ABM?

An ideal auction should satisfy as many agents as possible at a price that does not penalize an unsuspecting agent. The auction creates an official protocol that agents must abide by, so that the rules of the auction become an alternative environment, and traders the ecosystem to it. Auctions are often represented with multi-agent systems for this reason.

Auction Mechanism

Auctions are a unique type of agent-based game used to model potentially optimal and/or stable outcomes of distribution and utility maximization amongst agents.

Here is a small piece of the extension written in Scala:

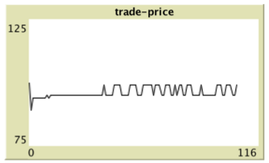
```
object Clear extends api.Command {
  /* clear all orders from market:
   * find the fill price where supply meets demand
   * fill all bids above, all offers below, and the matching orders at the price
   * returns assets and capital to respective agents
   */
  override def getSyntax = commandSyntax(right = List(AgentType))
  def perform(args: Array[api.Argument], context: api.Context) {
    val market = Router.routes(args(0).getTurtle)

    // before finding fill price, sort bids and asks
    // these are arrays of tuples for the standing bids and asks in a market
    // they are (price, sum(size))  Buyer/Seller orders that were placed
    val sortedBids = market.sortBids()
    val sortedAsks = market.sortAsks()

    // pss  - stands for Price, Size, Side (side = "bid"/"ask")
    val pss = equilibriumPriceCalc(sortedBids, sortedAsks, (0, 0, "no-trade"))

    // pss will equal None if there have been no fills, in that case all orders
    // are removed from the market and assets/money are returned
    if (pss._3 != "no-trade")
      // this procedure adds assets or money to traders according to the fills
      // and returns any assets or money that is being held until execution
      market.fillAllTraders(pss)
    else
      market.resetLastTrade()
    market.returnAllHoldings(pss)
    market.resetAllOrders()
  }
```

Clear is the extension primitive that must be called any round when all orders have been placed and trades are ready to be executed.

Results: pricing

In balanced runs prices would find a single trading price for a series of ticks. This displays stability and efficiency of markets

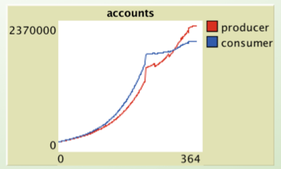
Eventually instability would emerge as trading at a price became more favorable to either the buy side or sell side

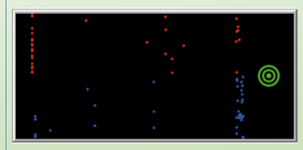


As traders exit the market prices swing rapidly. Rapid price swings result in more favorable outcomes for one side, allowing reversion to the mean.

Results: accounting

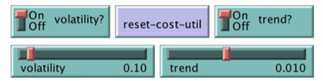
Producers and consumers see exponential account gains. The individuals with the greatest account gains will drive prices in the run



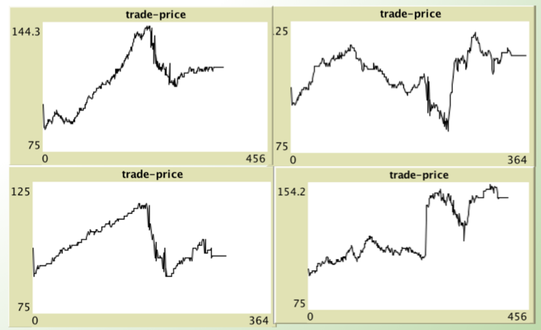


A distribution of traders' accounts shows us that not all traders will win. Agents with prices that don't trade will fail to make money

Modeling real markets



With a little bit of stochastic price behavior and exponential price growth, we can model the price movement in real markets with this model. The model isolates the cyclical price swings and account growth of agents. By adding uncertainty to outcomes in utility or cost, agents display a similar type of account growth while altering prices to be consistent with the new metrics.



References

- *Building the Santa Fe Stock Market,* Blake LeBaron, Brandeis University. 2002
- *Allocative Efficiency of Markets with Zero Intelligence Traders: Market as a Partial Substitute for Individual Rationality.* Dhananjay K. Gode; Shyam Sunder. The Journal of Political Economy. Vol 101, No. 1 Feb., 1993.
- *Fundamentals of Multiagent Systems with NetLogo Examples* Jose M Vidal. 2010
- *Autonomous Bidding Agents: Strategies and Lessons from the Trading Agent Competition*, Michael P. Wellman, Amy Greenwald, Peter Stone. 2007 MIT press
- *NetLogo: http://ccl.northwestern.edu/netlogo/*. Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL. Uri Wilensky. 1997
- *An Introduction to Agent-Based Modeling: Modeling Natural, Social, and Engineered Complex Systems with Netlogo.* Uri Wilensky and William Rand. 2015