A Young Person's Guide to the Heterogeneous Agents Resources and toolKit

Matthew N. White, University of Delaware

May 8, 2017

The Story So Far...

- Coding structural models is hard, with serious barriers to entry: basic programming, numeric techniques, economic modeling, solution methods, estimation, etc
- Could try to build on or combine existing code, but...
  - Written in language you don't know
  - For a very specific purpose / hard to extend
  - Cryptically documented, at best
  - Doesn't work with code from another economist
- Result: lots of wheel reinventing and duct tape coding

Goals of HARK

Heterogeneous Agents Resources and toolKit:

- Framework for dynamic models: *modularity* & *interoperability*
- Want elements/modules to play nicely with each other
- Reduce “Tower of Babel problem” and “duct tape coding”
- Universal structure: easy to combine models
- Make a “universal solver” so models speak the same language
- Frameworks for both “micro” and “macro” models

Agenda for Today

1. “Microeconomic” models in HARK: the `AgentType` class
2. Class inheritance of solvers
3. “Macroeconomic” models in HARK: the `Market` class
4. Other models in HARK and areas for near future work

Heterogeneous Agents

Two dimensions of heterogeneity

- *Ex post* heterogeneity: Agents differ because a different sequence of events or shocks has happened to them
- *Ex ante* heterogeneity: Agents differ in objectives, preferences, expectations, etc before anything in the model "happens"

Microeconomic Models

Microeconomic models in HARK:

- Concern decision-making of one agent
- Discrete time
- Sequence of choices
- Possibly subject to risk
- Agents treat inputs to problem as *exogenous*

Key restriction

Model solution can be interpreted as iteration on sequence of “one period problems”, conditional on solution to subsequent period.

HARK's microeconomic structure: `AgentType`

- General purpose class for representing economic agents
- Each model creates a subclass of `AgentType`
  - Includes model-specific attributes, functions, and methods...
  - ...And how to solve the "one period problem" for that model
  - Instances of subclass are *ex ante* heterogeneous "types"
- All `AgentType` subclasses use the same `solve()` method
  - Just a universal backward induction loop...
  - ...That lets different models "play nicely" together
- Complex models extend basic ones through class inheritance

Example Model: Basic Consumption-Saving

Consumption-saving model with idiosyncratic permanent and transitory shocks to income (normalized format):

$$u(c) = c^{1-\rho}/(1-\rho).$$

$$v_t(m_t) = \max_{c_t} u(c_t) + \beta \cancel{D}_t \mathbb{E}_t\left[(\psi_{t+1}\Gamma_t)^{1-\rho} v_{t+1}(m_{t+1})\right] \text{ s.t.}$$

$$a_t = m_t - c_t, \quad a_t \geq \underline{a},$$

$$m_{t+1} = R/(\Gamma_t \psi_{t+1}) a_t + \theta_{t+1},$$

$$\psi_{t+1} \sim F_{\psi t+1}(\psi), \quad \theta_{t+1} \sim F_{\theta t+1}(\theta), \;\; \mathbb{E}[\psi_t] = 1.$$

Example Model: Basic Consumption-Saving

Model solution in two lines:

$$\text{FOC: } u'(c_t) = R\beta\cancel{D}\mathbb{E}_t\left[(\psi_{t+1}\Gamma_t)^{-\rho}v'_{t+1}(m_{t+1})\right],$$

$$\text{EC: } v'_t(m_t) = u'(c_t).$$

Will use endogenous grid method:

$$\mathfrak{v}'_t(a_t) \equiv R\beta\mathbb{E}_t\left[(\psi_{t+1}\Gamma_t)^{-\rho}v'_{t+1}(m_{t+1})|a_t\right],$$

$$c_t = \mathfrak{v}'_t(a_t)^{-\rho}, \qquad m_t = a_t + c_t \text{ (for exogenous set of } \{a_t\}).$$

Defining An Agent's Problem (1/5)

- Must know what variables, distributions, information are needed to solve a one period problem
- Are those things the same in every period, or do they vary across periods?
- Basic consumption-saving model:
  - `time_inv = ['CRRA', 'Rfree', 'DiscFac', 'BoroCnstArt', 'vFuncBool', 'CubicBool', 'aXtraGrid']`
  - `time_vary = ['IncomeDstn','LivPrb','PermGroFac']`

Defining An Agent's Problem (2/5)

- Must know what the values that those variables, distributions, information have in each period
- What is the sequence of periods that agent will encounter?
- Basic consumption-saving model:
  - `CRRA = 3.2`
  - `DiscFac = 0.96`
  - `Rfree = 1.03`
  - `PermGroFac = [1.005,...,1.005,0.4,0.998,...,0.998]`
  - `LivPrb = [1,...,1,0.997,0.994,0.991,...,0]`
  - `IncomeDstn = [too much to put here]`

Defining An Agent's Problem (3/5)

- How many times does that sequence of periods happen?
- Just once? Ten times? Indefinitely?
  - `cycles = 1` : Sequence happens once, lifecycle model
  - `cycles = 40` : Sequence occurs 40 times in a loop
  - `cycles = 0` : Sequence repeats forever, infinite horizon

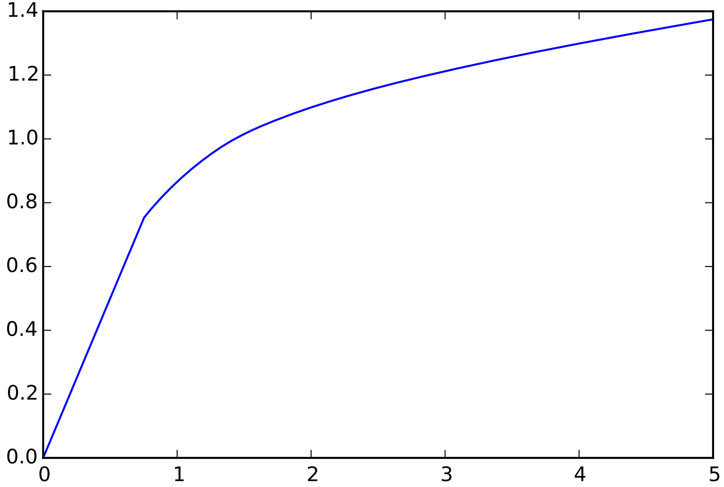
Defining An Agent's Problem (4/5)

- Must know *how* to solve a one period problem...
- ...*conditional* on solution to next period's problem...
- ...given values of time (in)variant parameters
- Basic consumption-saving model:
  - `solveOnePeriod = solveConsIndShock`
  - Inputs are named in `time_vary` and `time_inv`...
  - ...plus `solution_next`, the output from previous call
  - Attributes of `solution_next`: `cFunc`, `vPfunc`, `vPPfunc`, etc

Defining An Agent's Problem (5/5)

- Can solve by backward induction, but how do we start?
- Rather: how does the problem "end"? What comes "after"?
- Finite horizon: Need a terminal period solution or scrap value
- Infinite horizon: Need an initial "guess" of the solution
- `solution_terminal` often can be found in closed form
- Basic consumption-saving problem:
  - `solution_terminal.cFunc = consume everything`
  - `solution_terminal.vPfunc = u'(consume everything)`

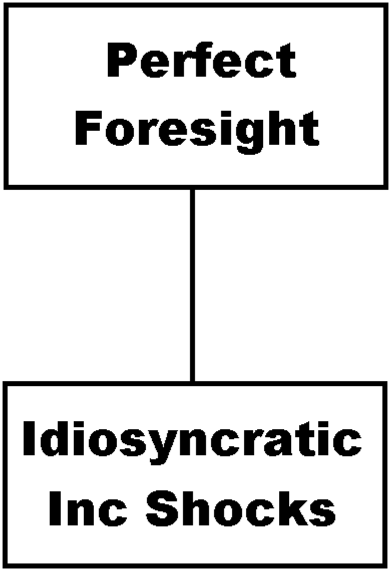
Basic Model Consumption Function



Object-Oriented Solution Methods (1/3)

- Models in HARK build up from each other
- “Parent” models are special cases of “child” models
- Solvers in HARK are objects that act (a lot) like functions
- Each model specifies a new class for its solver
- Inherit solution method from parent solver...
- ...and add or change its methods / subroutines.

Object-Oriented Solution Methods (2/3)

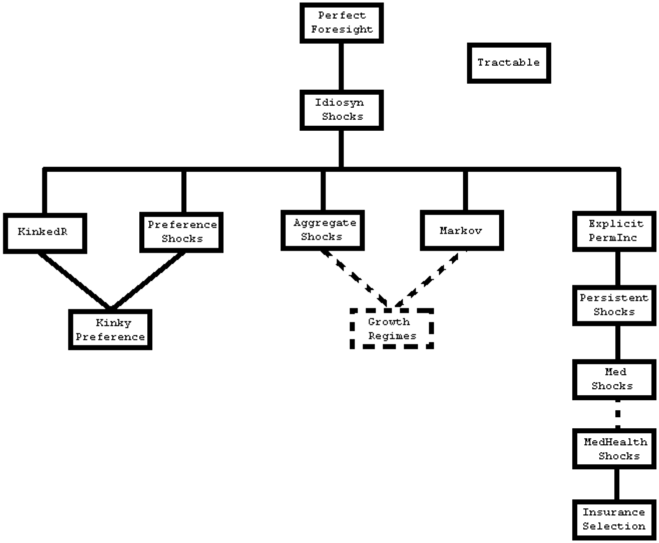


Object-Oriented Solution Methods (3/3)

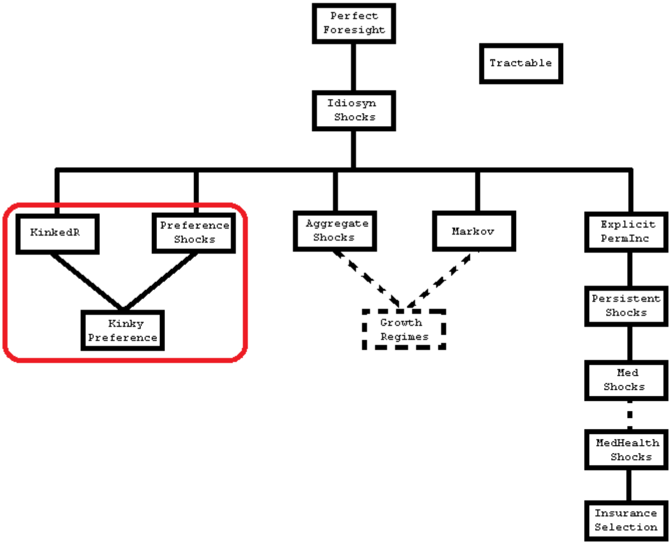
`ConsIndShockSolver` inherits from `ConsPerfForesightSolver`

- `setAndUpdateValues`: Calc constants from primitives: worst shocks, min and max MPC, human wealth, etc
- `defBoroCnst`: Find `cFunc` when borrowing constraint binds
- `prepareToCalcEndOfPrdvP`: Generate array of m_{t+1} values
- `calcEndOfPrdvP`: Calc end-of-period marginal value on $\{a_t\}$
- `getPointsForInterpolation`: Calc $\{m_t\}$ and $\{c_t\}$ points
- `usePointsForInterpolation`: Construct the `cFunc`
- `makevFunc`: Construct interpolated value function `vFunc`

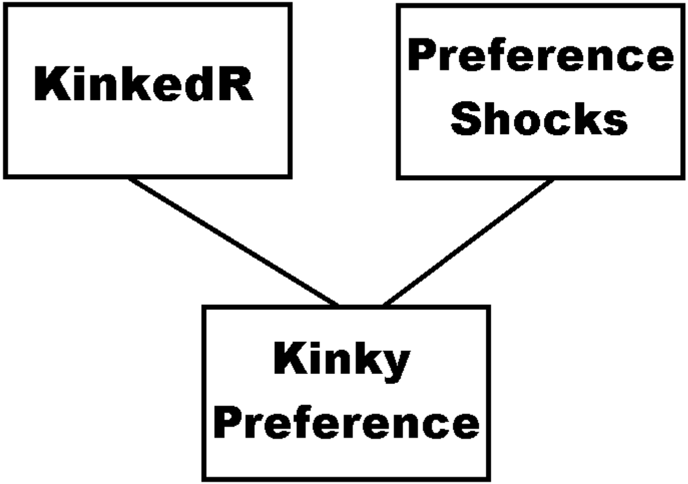
Consumption-Saving Model Tree



Consumption-Saving Model Tree



Consumption-Saving Model Tree



Kinked R: Costly Borrowing (1/3)

Make one small adjustment to idiosyncratic income shocks model: interest rate on borrowing is higher than rate on saving.

$$
\begin{aligned}
u(c) &= \frac{c^{1-\rho}}{1-\rho}, \\
v(m_t) &= \max_{c_t} u(c_t) + \beta \cancel{D}_{t+1} \mathbb{E}[v_{t+1}(m_{t+1})], \\
a_t &= m_t - c_t, \qquad a_t \geq \underline{a}, \\
m_{t+1} &= R/(\Gamma_{t+1}\psi_{t+1}) a_t + \theta_{t+1}, \\
\theta_{t+1} \sim F_{\theta t+1}, & \qquad \psi_{t+1} \sim F_{\psi t+1}, \quad \mathbb{E}[\psi_{t+1}] = 1, \\
R &= \begin{cases} R_{boro} & \text{if } a_t < 0 \\ R_{save} & \text{if } a_t > 0 \end{cases}, \qquad R_{boro} \geq R_{save}.
\end{aligned}
$$

Kinked R: Costly Borrowing (2/3)

`ConsKinkedRsolver` inherits from `ConsIndShockSolver`

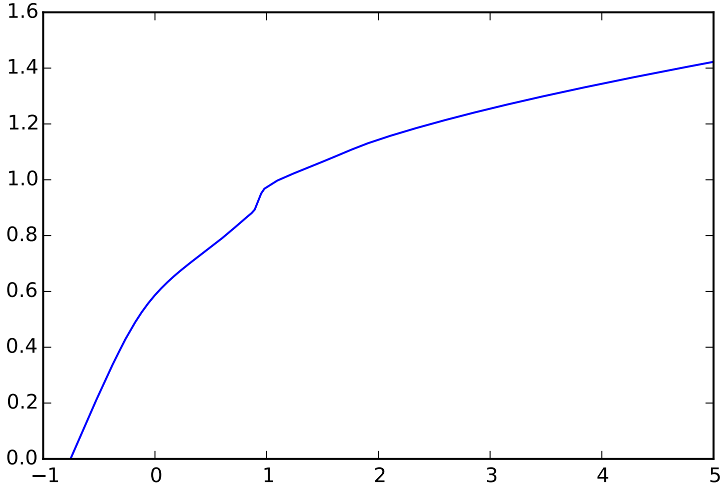
Additions to `__init__` method:

- Store new attributes `Rboro` and `Rsave`

Additions to `prepareToCalcEndOfPrdvP`:

- Four lines to use correct value of R for each value of a_t
- One line to apply that change to calculation of m_{t+1}
- Three lines to recalculate minimum MPC and human wealth

Kinked R: Costly Borrowing (3/3)



Marginal Utility Shocks (1/4)

Consider another small modification to `IndShockModel`:

- Multiplicative (idiosyncratic) shocks to utility each period.
- Consumption "more valuable" in some periods than others.

$$
\begin{aligned}
u(c;\eta) &= \eta\frac{c^{1-\rho}}{1-\rho}, \qquad \eta_t \sim F_\eta, \\
v(m_t,\eta_t) &= \max_{c_t} u(c_t;\eta_t) + \beta \cancel{D}_{t+1}\mathbb{E}[v_{t+1}(m_{t+1})], \\
a_t &= m_t - c_t, \qquad a_t \geq \underline{a}, \\
m_{t+1} &= R/(\Gamma_{t+1}\psi_{t+1})a_t + \theta_{t+1}, \\
\theta_{t+1} \sim F_{\theta t+1}, & \qquad \psi_{t+1} \sim F_{\psi t+1}, \quad \mathbb{E}[\psi_{t+1}] = 1.
\end{aligned}
$$

Marginal Utility Shocks (2/4)

New input `PrefShkDstn` is constructed:

- `PrefShkStd`: Standard deviation of (log) pref shocks
- `PrefShkCount`: Number of discrete shocks in “body”
- `PrefShkTailCount`: Discrete shocks in “augmented tail”

Marginal Utility Shocks (3/4)

`ConsPrefShockSolver` inherits from `ConsIndShockSolver`

Additions to `__init__` method:

- 2 lines: Store preference shock distribution `PrefShkDstn`

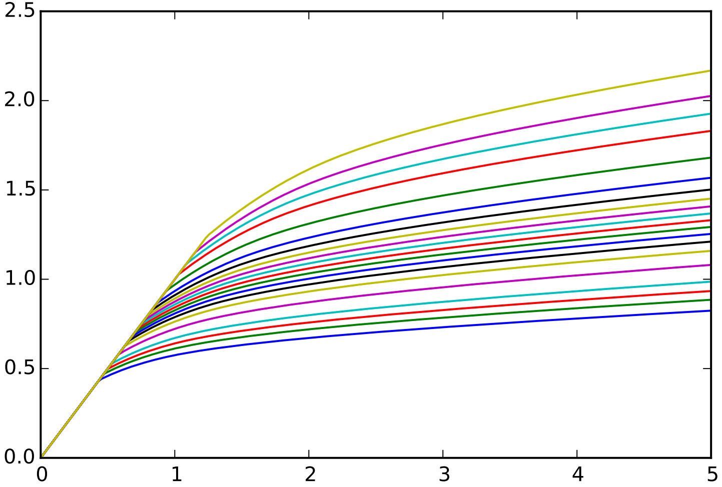
Replace `getPointsForInterpolation`

- 8 lines: Values of c_t and m_t for each η_t in `PrefShkDstn`

Replace `usePointsForInterpolation`

- 6 lines: Construct `cFunc` as a `LinearInterpOnInterp1D`
- 6 lines: Make `vPfunc` by integrating marginal utility across η_t

Marginal Utility Shocks (4/4)



Combination Inheritance: “Kinky Preferences” (1/4)

Combine those two extensions to `IndShockModel`:

- Borrowing has higher interest rate than saving...
- ...and there are shocks to marginal utility
- HARK makes this pretty easy

Combination Inheritance: “Kinky Preferences” (2/4)

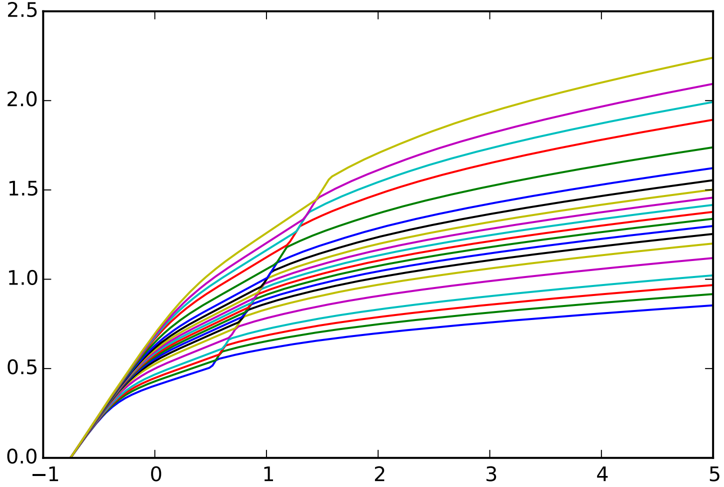
$$
\begin{aligned}
u(c, \eta) &= \eta \frac{c^{1-\rho}}{1-\rho}, \qquad \eta_t \sim F_\eta, \\
v(m_t, \eta_t) &= \max_{c_t} u(c_t) + \beta \not{D}_{t+1} \mathbb{E}[v_{t+1}(m_{t+1})], \\
a_t &= m_t - c_t, \qquad a_t \geq \underline{a}, \\
m_{t+1} &= R/(\Gamma_{t+1}\psi_{t+1})a_t + \theta_{t+1}, \\
\theta_{t+1} \sim F_{\theta t+1}, & \qquad \psi_{t+1} \sim F_{\psi t+1}, \quad \mathbb{E}[\psi_{t+1}] = 1, \\
R &= \begin{cases} R_{boro} & \text{if } a_t < 0 \\ R_{save} & \text{if } a_t > 0 \end{cases}, \qquad R_{boro} \geq R_{save}.
\end{aligned}
$$

Combination Inheritance: “Kinky Preferences” (3/4)

`ConsKinkyPrefSolver` inherits from two parent classes. Entirety of the code for the solver:

```
class ConsKinkyPrefSolver(ConsPrefShockSolver,ConsKinkedRsolver):
    def __init__(self,solution_next,...):
        ConsKinkedRsolver.__init__(self,solution_next,...)
        self.PrefShkPrbs = PrefShkDstn[0]
        self.PrefShkVals = PrefShkDstn[1]
```

Combination Inheritance: “Kinky Preferences” (4/4)



Macroeconomics in HARK (1/5)

- Some inputs to micro models are exogenous to each agent...
- ...But endogenous to collective whole of agents
- Might be static quantities or dynamic processes
- Equilibrium: consistency between what agents *believe* the endogenous objects are and what values / dynamic processes *actually occur* when agents act on those beliefs
- Fixed point in the space of beliefs
- Need a representation of beliefs about endogenous objects
- And a rule for how agents form beliefs from observing history

Macroeconomics in HARK (2/5)

"The computational algorithm has two key features. First, it is based on bounded rationality in the sense that we endow agents with boundedly rational perceptions of how the aggregate state evolves... Second, we use solution by simulation, which works as follows: (i) given the boundedly rational perceptions, we solve the individuals' problems using standard dynamic programming methods; (ii) we draw individual and aggregate shocks over time for a large number of individuals; (iii) ...we generate a time series for all aggregates; and finally (iv) we compare the perceptions about the aggregates to those in the actual simulations, and these perceptions are then updated. We think this approach... can be productive for other applications."
–Per Krusell and Tony Smith (2006)

Macroeconomics in HARK (3/5)

HARK operationalizes K-S method with a farming metaphor:

1. Solve agents' microeconomic problem for some beliefs
2. Simulate many agents for many periods by looping on:
    - `sow`: Distribute current aggregate variables to agents
    - `cultivate`: Agents act according to their micro solution
    - `reap`: Collect some individual variables from the agents
    - `mill`: Generate aggregate variables from individual vars
    - `store`: Record some information in a "history"
3. Use history to update beliefs about endogenous objects

`Market`'s method `solve` loops on this process until convergence.

Macroeconomics in HARK (4/5)

Attributes of a `Market` instance (or subclass):

- `agents`: List of `AgentType` instances in market
- `dyn_vars`: Names of the endogenous objects
- `sow_vars`: Names of aggregate variables
- `reap_vars`: Individual variables that form aggregates
- `track_vars`: Aggregates that need to be recorded in history
- `millRule`: Function that transforms ind $\longrightarrow$ agg variables
- `calcDynamics`: Function that transforms history into beliefs

Macroeconomics in HARK (5/5)

Extra methods of a `Market`-compatible `AgentType`:

- `marketAction`: What agents *do* to generate `reap_vars`. Often just simulate one period with `simOnePeriod`
- `reset`: How to initialize for a new history: reset states

Trivial to add more *ex ante* heterogeneity: just add more `AgentType` instances to `agents`!

Consumption-Saving with Aggregate Productivity Shocks

$$
\begin{array}{rcl}
v_t(m_t, M_t) & = & \max_{c_t} u(c_t) + \beta \cancel{D}_{t+1} \mathbb{E}[v_{t+1}(m_{t+1}, M_{t+1})], \\
a_t & = & m_t - c_t, \qquad a_t \geq 0, \\
m_{t+1} & = & \dfrac{R_{t+1} a_t}{\Gamma_{t+1} \psi_{t+1} \Psi_{t+1}} + W_{t+1} \theta_{t+1} \Theta_{t+1} \ell, \\
A_t = \mathbf{A}(M_t), & & k_{t+1} = (1-\delta) A_t / (\Psi_{t+1} \ell), \\
R_{t+1} = \mathbf{R}(k_{t+1}/\Theta_{t+1}), & & W_{t+1} = \mathbf{W}(k_{t+1}/\Theta_{t+1}), \\
M_{t+1} & = & R_{t+1} k_{t+1} + W_{t+1} \Theta_{t+1} \ell \\
\theta_{t+1} \sim F_{\theta t+1}, & & \psi_{t+1} \sim F_{\psi t+1}, \;\; \mathbb{E}[\psi_{t+1}] = 1, \\
\Theta_{t+1} \sim F_{\Theta}, & & \Psi_{t+1} \sim F_{\Psi}, \;\; \mathbb{E}[\Psi_{t+1}] = \mathbb{E}[\Theta_{t+1}] = 1.
\end{array}
$$

Consumption-Saving with Aggregate Productivity Shocks

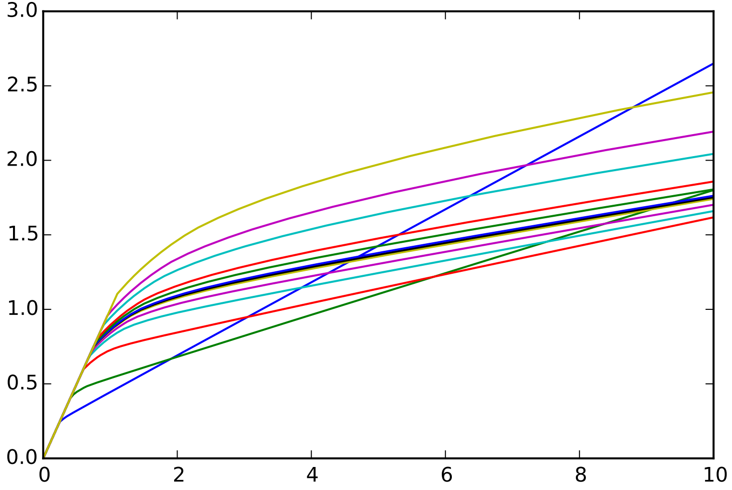
Some totally new inputs for an `AggShockConsumerType`:

- `Rfunc` and `wFunc`: Factor payments as function of k_t
- `Mgrid`: Grid of M_t state values (sort of constructed)
- `Afunc`: $\mathbb{E}[A_t|M_t] = \mathbf{A}(M_t)$

`IncomeDstn` combines idiosyncratic and aggregate shocks:

- Discrete approximation to aggregate shock distribution constructed like idiosyncratic shocks: `PermShkAggStd`, `TranShkAggStd`, `PermShkAggCount`, `TranShkAggCount`
- `IncomeDstn` has five elements: probs, idio shocks, agg shocks

Consumption-Saving with Aggregate Productivity Shocks



Cobb-Douglas Economy in the `Market` Framework

`CobbDouglasEconomy` is a subclass of `Market`:

- `sow`: Distribute $(M_t, R_t, W_t, \Theta_t, \Psi_t)$ to consumers
- `cultivate`: Consumers draw (θ_t, ψ_t), choose c_t
- `reap`: Collect assets a_t and productivity P_t from consumers
- `mill`: Calc A_{t+1}, draw $(\Theta_{t+1}, \Psi_{t+1})$, calc k_{t+1}, M_{t+1}, get (R_{t+1}, W_{t+1})
- `store`: Record M_t and A_t in their histories

Loop that process for (say) 1000 periods

- `calcDynamics`: Regress $log(A_t)$ on $log(M_t)$, make new $\mathbf{A}$
- Distribute new $\mathbf{A}$ to consumers as `Afunc`

Other Applications of `Market`

Krusell and Smith were right: method is applicable to other topics

- Premiums of medical insurance contracts: actuarial constraint maps *who* buys each contract to break even premium, subject to informational constraints (sex, age, health, etc)
- Papageorge et al risky sex framework: probability of contracting HIV from risky sex act depends on HIV infection rate and risky sex choices of the population
- Agent-to-agent interaction: could `sow` a permutation of what is `reaped`: imperfect knowledge, contagion of information, moves closer to "agent-based modeling"

Other Consumption-Saving Models in HARK

But wait, there's more!

- `TractableBufferStock`: Highly specialized idiosync shocks
- `MarkovModel`: $\cancel{D}, \Gamma, F_\theta, F_\psi, R$ vary by discrete state
- `ExplicitPermInc`: Same as `IndShock`, but not normalized
- `PersistentShock`: "Permanent" shocks not fully permanent
- `MedShock`: 2nd cons good with random marginal utility
- `MedHealthShock`:* Medical shocks plus discrete health states
- `DynInsSel`:* ...plus choice over medical insurance contracts

And even more to come...

The Future of HARK: Small To-Do Items

Contributions that would get your feet wet in HARK:

- Extend perfect foresight to handle borrowing constraint
- Bequest motives: warm glow, other?
- Fix/improve aggregate shocks model: handle life cycle models
- Fix/generalize `ExplicitPermInc` models: `PermGroFunc`
- Advanced features on more solvers: cubic spline interpolation
- Various numeric methods detached from particular models

The Future of HARK: Incorporating Labor (1/4)

Model of labor supply on intensive margin:

$$u(c,\ell) = ((1-\ell)^{\alpha} c)^{1-\rho}/(1-\rho),$$

$$v_t(b_t,\theta_t) = \max_{c_t,\ell_t} u(c_t,\ell_t) + \beta \cancel{D}_t \mathbb{E}_t\left[(\psi_{t+1}\Gamma_t)^{1-\rho} v_{t+1}(b_{t+1},\theta_{t+1})\right] \text{ s.t.}$$

$$y_t = \ell_t \theta_t, \qquad \ell_t \in [0,1],$$

$$a_t = m_t + y_t - c_t, \quad a_t \geq \underline{a},$$

$$b_{t+1} = R/(\Gamma_t \psi_{t+1}) a_t,$$

$$\psi_{t+1} \sim F_{\psi t+1}(\psi), \quad \theta_{t+1} \sim F_{\theta t+1}(\theta), \;\; \mathbb{E}[\psi_{t+1}] = 1.$$

The Future of HARK: Incorporating Labor (2/4)

Model of labor supply on extensive margin:

$$u(c,\ell) = c^{1-\rho}/(1-\rho) - \alpha\ell,$$

$$v_t(b_t,\theta_t,\ell_{t-1}) = \max_{c_t,\ell_t} u(c_t,\ell_t) + \beta \not{D}_t \mathbb{E}_t \left[(\psi_{t+1}\Gamma_t)^{1-\rho} v_{t+1}(b_{t+1},\theta_{t+1},\ell_t) \right]$$

$$y_t = \ell_t\theta_t, \qquad \ell_t \in \{0,\ell_{t-1}\},$$

$$a_t = m_t + y_t - c_t, \quad a_t \geq \underline{a},$$

$$b_{t+1} = R/(\Gamma_t\psi_{t+1})a_t,$$

$$\psi_{t+1} \sim F_{\psi t+1}(\psi), \quad \theta_{t+1} \sim F_{\theta t+1}(\theta), \ \mathbb{E}[\psi_{t+1}] = 1.$$

The Future of HARK: Incorporating Labor (3/4)

Model of endogenous employment search:

$$u(c,s) = ((1-s)^{\alpha} c)^{1-\rho}/(1-\rho),$$

$$v_t(m_t, e_t) = \max_{c_t, s_t} u(c_t, s_t) + \beta \cancel{D}_t \mathbb{E}_t \left[(\psi_{t+1}\Gamma_t)^{1-\rho} v_{t+1}(m_{t+1}, e_{t+1}) \right] \text{ s.t.}$$

$$a_t = m_t - c_t, \quad a_t \geq \underline{a}, \quad s_t \in [0,1],$$

$$m_{t+1} = R/(\Gamma_t^e \psi_{t+1}) a_t + \theta_t e_{t+1} + \underline{b}(1 - e_{t+1}),$$

$$\text{Prob}(e_{t+1} = 1 | e_t = 0) = s_t, \qquad \text{Prob}(e_{t+1} = 0 | e_t = 1) = \mho,$$

$$\psi_{t+1} \sim F^e_{\psi t+1}(\psi), \quad \theta_{t+1} \sim F_{\theta t+1}(\theta), \quad \mathbb{E}[\psi_t] = 1.$$

The Future of HARK: Incorporating Labor (4/4)

Applications of `Market` for labor models:

- Non-trivial calculation of $L_t = \int_0^1 \ell_{it} P_{it} \theta_{it} di$ for Cobb-Douglas
- Disutility of employment search and probability of job loss depend on labor market slackness
- Can look at behavior in response to change in SS, etc

The Future of HARK: Durable Goods (1/3)

General durable goods model:

$$u(c,d) = (c^{\alpha}, d^{1-\alpha})^{1-\rho}/(1-\rho).$$

$$v_t(m_t, d_t) = \max_{c_t, i_t} u(c_t, d_t) + \beta \cancel{D}_t \mathbb{E}_t \left[(\psi_{t+1}\Gamma_t)^{1-\rho} v_{t+1}(m_{t+1}, d_{t+1}) \right] \text{ s.t.}$$

$$a_t = m_t - c_t, \quad a_t \geq \underline{a},$$

$$D_t = d_t + g(i_t), \quad d_{t+1} = (1-\delta_{t+1})D_t, \quad \delta_{t+1} \sim F_{\delta}(\delta),$$

$$m_{t+1} = R/(\Gamma_t \psi_{t+1}) a_t + \theta_{t+1},$$

$$\psi_{t+1} \sim F_{\psi t+1}(\psi), \quad \theta_{t+1} \sim F_{\theta t+1}(\theta), \quad \mathbb{E}[\psi_t] = 1.$$

The Future of HARK: Durable Goods (2/3)

Variations of durable goods model require different solvers:

- Easiest case: $g(i_t)$ is concave, $i_t \in \mathbb{R}$. Every end-of-period state (a_t, D_t) associated with *some* beginning of period state.
- Slightly harder: $i_t \geq 0$, must handle constraint
- Somewhat harder: $g(i_t) = \pi i_t$. One locus in (a_t, D_t) space is optimal; each point on optimal (a_t, D_t) locus associated with locus in (m_t, d_t) space.
- Even harder: $g(i_t) = \widehat{g}(i_t) + K\mathbf{1}(i_t \neq 0)$, with $\widehat{g}(0) = 0$ and $\widehat{g}(\cdot)$ concave. Must check $i_t = 0$ soln everywhere, discont.
- Just ugh: $g(i_t) = \pi i_t + K\mathbf{1}(i_t \neq 0)$, $i_t \geq 0$.

The Future of HARK: Durable Goods (3/3)

Applications for `Market` with durable goods:

- Endogenous pricing of durable good: housing market
- Dynamics of demand for durables after an aggregate shock
- Some specifications overlap with health models

The Future of HARK: Heavy Lifting

If you're feeling ambitious or are comfortable with HARK:

- Incorporate `opencl4py` with basic consumption-saving model. "Repack" model inputs into memory buffers, pass to OpenCL solver. OpenCL simulator: easier, big gains for some models.
- Aggregate shocks with explicit permanent income. One endogenous state variable, two exogenous state variables. Future candidate for GPU computing.
- Generalized Markov solver: make "solution schema" so that Markov state can be added to any correctly specified solver
- Models of firm creation / bankruptcy / investment / hiring

Concluding Remarks

- HARK isn't just for heterogeneous agents macroeconomics
- Useful for structural health/labor, industrial organization, agent-based models, etc
- First piece of the larger Econ-ARK
- Many of the tools in HARK will eventually be "promoted"
- Momentum is key: development and additions spur more work
- More eyes are needed: double check our work, improve it