Initial Draft Language for Grant Proposal for the Development of HARK 2.0

Purpose: This document contains preliminary language that could be included in a request for funding for the development of "HARK 2.0", a software package for representing, solving, simulating, and estimating dynamic models with heterogeneous agents. It is not formatted as a grant proposal, but instead merely includes fragments of text that can be further developed and refined. The name "HARK 2.0" is used only as a shorthand in this paragraph.

Background: Over the past thirty years, economists have increasingly developed dynamic models of the decisions of individual agents (i.e. households or firms) that are intended to be interpreted *quantitatively* rather than only *qualitatively*. Instead of merely characterizing the general properties of the outcomes predicted by a particular theoretical model, researchers instead seek to quantitatively evaluate its performance in matching past observed outcomes, and to generate quantitatively plausible predictions of the effects of hypothetical future events and policies. This approach to economics is often referred to as structural modeling. When applied to macroeconomic topics, structural modeling involves generating aggregate outcomes as the accumulation of the individual decisions of many agents, each of whom may face different circumstances idiosyncratic to themselves. Hence this subfield is known as heterogeneous agents macroeconomics ("HA macro"), in contrast to the more traditional representative agent ("RA") approach in which idiosyncratic heterogeneity among households and firms is assumed away.

Many HA macro models include stochastic processes that realistically reflect the large uncertainties inherent to the agents' problem– potentially large shocks to their productivity, demand for their labor (or output), health, etc. In combination with risk averse preferences or constraints on access to capital, these sources of uncertainty generate model solutions for optimal behavior that are highly non-linear and require global (rather than local) methods for solving the model. Such methods are computationally complex and costly, requiring careful application of fundamental numeric tools like quadrature methods, functional approximation, and solving systems of non-linear equations. These tools are not usually taught as part of graduate economics training, and it takes years of experience to understand how they should be applied, and how to interpret (and correct) their apparent failures and shortcomings.

While economists have developed a standard and well-adopted set of tools for representing, solving, and examining RA macro models (including especially the Dynare software package), and there is commercially available software for statistical and econometric analysis (e.g. Stata), there is no widely adopted software nor standard for HA models, nor dynamic structural models in general. Each research team instead develops its own bespoke software for each new project, often inheriting or adapting pieces from their own prior work. The code for each model is designed to meet the specific needs of that project, and there is no interoperability or standards for the usage or application of numeric methods.

Proposal Summary: Econ-ARK seeks funding to develop a language for expressing dynamic structural models (with a particular focus on heterogeneous agents macroeconomics), specifying numeric methods, and describing simulation procedures to generate model output. The language is intended to provide a common format for describing dynamic structural models that will be widely adopted for precisely conveying model content in a human- and machine-readable way, but is independent of the software and code to actually solve and implement the model. To that end, the funding sought will also be used to develop a software package that interprets model statements in the new language, generates code for solving and simulating the model (with the provided numeric method choices), and allows the user to interactively develop and build a structural model and examine its solution and output.

Economic models are usually expressed with a combination of mathematical and natural language: most of the content can be concisely conveyed as a series of equations, inequalities, and other mathematical statements, while a few additional details are provided in plain English outside of the formal statement.[1] This representation of the theoretical model is intended to be readily understood by a human reader. Because dynamic structural models almost surely do not have a closed form solution, they can be solved only approximately using numeric methods. While it is increasingly common for researchers to publicly archive their project code, a description of the numeric methods used is usually omitted from the final published paper;[2] this information is (at best) relegated to an online appendix or (at worst) not documented anywhere outside of the code itself. Moreover, unlike the conventions of mathematically expressing a system of statements to compose a theoretical model, there is no straightforward and complete way to convey such numeric methods and details, even if an economist were so inclined. Even worse, there is no formal relationship between the model as expressed *on paper* and the problem as solved *in code*– the academic refereeing system focuses deeply on the economics and relies on trust with respect to the numerics.

[1]For example, few models of a person's life-cycle include a precise mathematical representation of what it means for an agent to "die" in the model; the authors trust that the reader understands the implications.

[2]E.g., economists rarely publish information about quadrature methods, the discretization of a continuous state variable, or convergence criteria.

Our proposed modeling language seeks to rectify these systemic issues with the workflow of economic research that uses dynamic structural models. Using the YAML data format, we will provide a common platform for precisely representing the mathematical mechanics of dynamic structural models, providing syntax to specify a wide range of model features. The representation of a model in this language will be both human-readable (due to the concise syntax and simple structure of YAML) and machine-readable (due to the precision and completeness of the modeling language). If a model specification file is used to generate a numeric solution and model output, a reader or evaluator can be confident that the model presented on paper matches its execution in code. Furthermore, our language will include a format for specifying the methods and choices used to solve the model numerically, transparently providing this information alongside the "pure" mathematical content of the model. In combination with our proposed software package, we seek to provide a platform for testing and evaluating the performance of a numeric solution to a theoretical model.[3]

Language Overview: The core element of our proposed modeling language is a "stage", representing a sequence of events for an agent (or actor). An agent's dynamic programming problem is constructed by linking these stages together, potentially recurring on itself. Model stages are meant to compartmentalize information, explicitly specifying the variables that are "inbound" to the stage as information (i.e. "state variables") and objects that can be observed from its successor when solving (e.g. the continuation value function). Each stage has a declaration of its parameters, distributions, functions, and algebraic substitutions. The *values* or *details* of these parameters and distributions do are not required when a block is defined, and can instead be left symbolic.

The dynamic model content of a stage could be as small as one model "step", or it could encompass as much as what economists would ordinarily describe as entire period of the model. For example, in the context of a basic consumption-saving model with transitory labor income risk and a riskless return, a stage could be as small as going from beginning-of-period capital holdings k_t to bank balances b_t by multiplying the former by return factor R; or it could be the next step, where labor income y_t is drawn from some distribution and added to b_t to yield market resources m_t. Alternatively, a stage could be specified as having *all* of the steps in one period of the consumption-saving problem: earning interest on capital, receiving stochastic labor income, choosing consumption, and retaining non-

[3]E.g., a tool for computing the accuracy of an approximate solution as measured by its adherence to a known characteristic of the solution, such as an automatic test of Euler equation errors.

consumed market resources as assets.

Suppose a user chose to specify the simple consumption-saving model using (say) four very small stages. Under our proposed modeling language, they could further specify that these stages are sequentially linked together. The resulting linked group of stages would itself then represent a stage (the entire period) that could be further linked.

The Econ-ARK team has been developing draft specifications of this modeling language, formatted as YAML files. This allows model information to exist alongside parameterization choices, but not inextricably paired together. Instead, the modeling language will include a directive for assigning parameters (or distribution information, etc) from a YAML dictionary entry to a stage, filling in previously unspecified information. As discussed below, our proposed software package would also allow a user to import a stage into their modeling environment without parameter values, and *then* to assign values.

Our draft language specification includes concepts for flow control among stages, so that (e.g.) an agent's discrete control can govern the *nature* of the next choice they face, or a mortality shock can determine whether they continue to the next period or simply terminate. A group of connected stages thus acts like (and can be represented by) a directional graph among a set of nodes. A graph that loops back on itself is interpreted as an infinite horizon model in which the solution for optimal controls is recursively defined, while a graph that reaches a definite terminal node is a finite lifecycle problem.

The language will also have syntax for how information interacts among agents, including agent-to-agent transfer and market-level aggregation– how idiosyncratic actions and states accumulate into macroeconomic outcomes. In our modeling language, such interactions are conducted by an actor that is also constructed from an assemblage of stages, albeit one that has no *agency*– it makes no choices and has no motivation or preferences. This structure allows for the macroeconomic endogenous objects (e.g. equilibrium prices) to be specified in the same way as microeconomic endogenous objects (e.g. optimal policy functions).

Software Package: The software package we plan to develop will act as an interpreter of the YAML model statements, interactive model development environment, and platform for applications using the models (e.g. structural estimation). Several features of the model specification language will also be included as programming directives in our software package– the user can link stages, create a sequence of periods, assign or change parameters, etc.

Our proposed software package will read in YAML model files and generate representations of model objects using `sympy`'s various capabilities, which now include so-called "pleuripotent" functions: objects that are simultaneously both a symbolic representation of a function's mathematical form *and* a numeric approximation of it (e.g. a representation of the optimal policy or value function) that can actually be executed. The `sympy` package now has well developed interfaces with various numeric back-ends (including `numpy` and `jax`), so that carrying symbolic representations of the model does not (necessarily) impose grave efficiency costs on computations using them.

Likewise, `sympy`'s automatic differentiation and algebraic substitution capabilities have now been sufficiently well developed that an Econ-ARK team member has prototyped code for generating an algebraic representation of the first order and envelope conditions for an intertemporal optimization problem. These representations can be automatically converted into code that numerically solves the choice problem using the endogenous grid method (EGM). Our software package will thus be able to quickly generate (a draft of) solver code from the model input, which could be used as-is or manually refined by the user; it will also allow the user to specify fully custom solver code.

We have also already prototyped automatic Monte Carlo simulation based on YAML model input, ensuring consistency between intended and actual model behavior. The new version of HARK will support multiple modes of population dynamics, including explicit mortality Monte Carlo, cumulative survival probability weights, and discretizations of continuous state variables (for Markov-style approximations).

Current Tools: [describe existing packages and their insufficiency]