Financial Frictions and the Wealth Distribution

Jesús Fernández Villaverde Samuel Hurtado Galo Nuño
Discussion: Lilia Maliar

July 23, 2020

Worum geht es in diesem Papier?

Excellent and ambitious paper with multiple contributions:

- **Financial frictions** in a DSGE model with heterogeneous agents;
- **Deep learning** in the context of Krusell and Smith's (1998) algorithm (henceforth, KS);
- **Multiple stochastic steady states** that generate **endogenous regime switches**;
- **Structural estimation** of a model with distributions evolving over time;
- **Accounting for some empirical regularities** that the representative agent KS model fails to reproduce.

Comment: What makes the individual decision functions to be so nonlinear in aggregate variables?

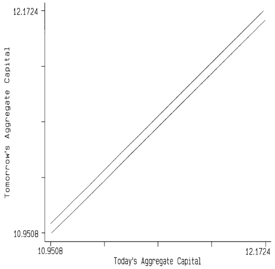
- The authors consider richer version of the Krusell and Smith (henceforth, KS) KS-type model:
  – heterogeneous agents like in KS & financial experts.
- In their model, the agent's decision function is
  – linear with respect to the agent's state variables;
  – but highly non-linear with respect to aggregate endogenous state variables (equity and debt).
  $\Longrightarrow$ Need a general flexible function of moments,
  $$K' = G\left(K\right).$$
  instead of KS polynomials $K' = b_0 + b_1 K$.
- Many recent algorithms, Boppart et al. (2018), Auclert et al. (2019), assume linearity of individual decisions in aggregate variables, which is not satisfied here.
- *Question:* What features or assumptions make the individual decision function to be nonlinear in aggregate variables?

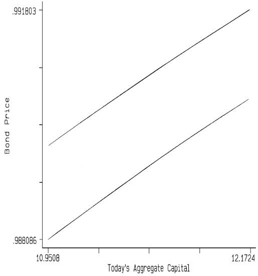
Comment: Multiplicity of SSSs is too sensitive to estimated volatility

- *Methodology:* calibrate all parameters, except of volatility σ of aggregate capital shock, which is estimated.
- The obtained point estimate: $\sigma = 0.014$.
- *Numerical result:* if $\sigma = 0.021$ – two stable SSS but if the volatility is higher – just one stable SSS.
- All non-trivial dynamics are due to multiple stable SSS.
  $\Longrightarrow$ The results are too sensitive to the estimated value of σ.
- To estimate σ use recent data 1984-2017.
- *Questions:*
  – *Q.1* How will the estimated value change if one considers a longer period (presumably, will lead to higher volatility)?
  – *Q.2* The model predicts supercycles of borrowing and deleveraging lasting centuries. If the data used for estimation comprise the most recent 40 years, can one compare the model to the data?

Comment: What is the reason for multiple SSSs?

Krusell and Smith (1997): a model with two endogenous aggregate states, capital and bond price $\Rightarrow$ no multiple SSSs.





Questions:

– Q.1 What are the assumptions of the present model that lead to multiple SSSs that KS does not have?

– Q.2 Neural network can stick to local minima. Can we rule out this possibility, namely, that neural network erroneously converges to two SSSs?

But deep learning is not used to a full potential

- Recall KS model with ℓ heterogeneous agents.
- Agents have capital and idiosyncratic productivity, $\left\{k_t^i, z_t^i\right\}_{i=1}^{\ell}$ and aggregate productivity follows Z_t.
- The state space has $2\ell + 1$ state variables, *for example, if we have* $\ell = 1000$ *agents, there are* 2001 *state variables.*
- To deal with the curse of dimensionality, KS use a reduced state space $\left(\left\{k_t^i, z_t^i\right\}_{i=1}^{\ell}, Z_t\right) \approx \left(k_t^i, z_t^i, Z_t, M_t\right)$

$$
\begin{aligned}
\text{agent solves } i \quad : \quad & k_{t+1}^i = K\left(k_t^i, z_t^i, Z_t, M_t\right) \\
\text{ALM:} \quad & M_{t+1} = a_0 + a_1 M_t
\end{aligned}
$$

  Instead of $2\ell + 1 = 2001$ state variables, we get just 4.
- The present paper uses neural network $M_{t+1} = DL\left(Z_t, M_t\right)$ instead polynomial.
- But all other objects (decision function, value function) are approximated using conventional rectangular grids like in KS.

Different application of DL for solving economic models

- Maliar, L., S. Maliar and P. Winant, (2019). Will artificial intelligence replace computational economists any time soon?
- The **key idea** is to convert the entire economic model (lifetime reward, Bellman equation, Euler equation)- into objective functions for deep learning.
- Feed the DL objective into machine using Google TensorFlow platform – the same software that is used for image recognition, operating self-driving cars, playing Go, etc.
- DL can solve models with **thousands of state variables** - examples of the code are available at *QuantEcon.org*.

KS (1998) model in MMW

- MMW also solve KS (1998) model *but work with the actual state space:* the entire KS economy is cast into DL objective.
- For example, if we have $\ell = 1000$ agents, we construct decision functions of 2001 state variables

$$k_{t+1}^i = K\left(\left\{k_t^i, z_t^i\right\}_{i=1}^{\ell}, Z_t\right).$$

- All equilibrium functions (decision rules, value functions, densities, ALMs) are found by deep learning using the same DL optimization problem.
- Our DL solution method for the KS model is extremely simple: *i) simulate the economy forward, ii) compute the aggregates, iii) train the agents*, and proceed until convergence.
- No assumptions are needed about ALM like what moments or statistics to include: just simulation of the panel $\left\{k_t^i, z_t^i\right\}_{i=1}^{\ell}$.

How can we solve so huge models?

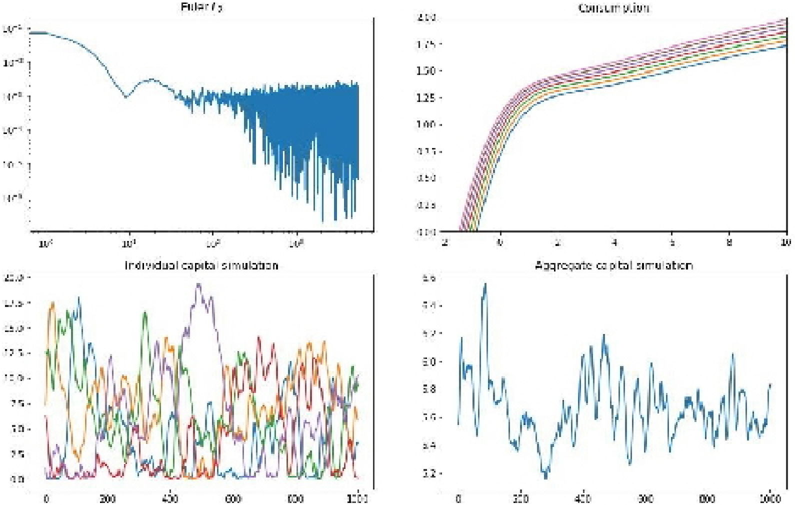
To deal with thousands of state variables, we rely on four results:

- Neural network **performs model reduction**:
  – It extracts information from 2001 inputs and condenses it into a small set of hidden layers (64 or 32).
  $$\left(k_t^i, z_t^i, \left\{k_t^i, z_t^i\right\}_{i=1}^{\ell}, Z_t\right) \Rightarrow (64 \ DL \ features)\,.$$
- Neural network **deals with ill-conditioning** by learning to ignore redundant and collinear variables:
  – For each agent i, we represent the state space as
  $$\left(k_t^i, z_t^i, \left\{k_t^i, z_t^i\right\}_{i=1}^{\ell}, Z_t\right).$$
- We solve the high-dimensional KS model using **stochastic simulation:** we focus on the ergodic set in which the solution "lives".
- We introduce **all-in-one-expectation** method that provides an unbiased estimator to an integral of any dimensionality with just two random draws.

How does the DL model reduction work?

- The decision function of each agent depends on thousands of state variables of all agents in the economy.
- KS (1998) perform the model reduction "by hand": they discovered that a single statistic – the mean of the wealth distribution – can effectively characterize the aggregate state of their model and replace the entire distribution.
- But this particular model reduction does not work for all heterogenous-agent models.
- In our DL analysis, the model reduction is automated: neural network learns a compact representation of the state space by extracting and condensing the relevant information (like Google Photo condenses and stores the pictures).
- Neural network will automatically search for all possible statistics (moments or any other statistics) that can effectively characterize the state space – this is what DL model reduction means.

A typical picture with the solution



Solving KS model using DL

ℓ	σ_y	$corr_{y,c}$	$Gini$	$Bot.\ 40\%$	$Top\ 1\%$	$T,\ sec.$	R^2
1	1.51	0.858	-	-	-	235	0.999999
5	1.51	0.772	0.335	0.176	0.031	234	0.993473
10	1.51	0.595	0.391	0.144	0.036	254	0.995091
50	1.51	0.635	0.497	0.099	0.050	467	0.995284
100	1.51	0.658	0.450	0.121	0.047	1020	0.997600
500	1.51	0.462	0.484	0.096	0.052	7552	0.996554
1000	1.51	0.268	0.501	0.092	0.047	16435	0.995415

Grain of salt about DL technology

- Neural network is a promising approximator but has a large number of parameters and is highly non-linear.
- Neural networks find local minima even for well behaved problems.
- Stochastic optimization is magical but its convergence rate is lower and not guaranteed.
- There are other promising AI technologies that can be used for solving economic models like reinforcement learning.

In conclusion: DL is not a panacea for all models. It is too early to retire computational economists.

Thank you!