RNNs for Efficient Estimation of Parameters of Econometric Models

Jonathan Chassot* and Michael Creel†

July 27, 2022

Abstract

Using LSTM ...

Keywords: neural networks; ....

JEL codes: C??

*Faculty of Mathematics and Statistics, University of St. Gallen , Switzerland.

†Universitat Autònoma de Barcelona, Barcelona School of Economics, and MOVE, Bellaterra (Barcelona) 08193, Spain. michael.creel@uab.cat.

1 Motivation

1. Fact: under fairly general conditions, the posterior mean has the same asymptotic distributions as does the ML estimator, so is asymptotically efficient (Chernozhukov and Hong 2003 and similar refs.).

2. Conjecture, based on universal approximation theorem (Hornik, Stinchcombe, and White 1989): A recurrent net that fits samples to the parameters that generated them, using a MSE loss function, will converge to the posterior mean of the parameters, given the sample, or to something close to it. As far as I know, formal results for RNNs are still very vague. Nevertheless, much evidence has accumulated about good predictions under squared error loss.

Main implication:

With these two, the output of a net that is rich enough and well enough trained

1. will be a good point estimator for the parameters that generated the sample

2. can be used as moment conditions for the method of simulated moments. Because these moment conditions are close to the ML estimator, these moment conditions will give a MSM estimator that is approximately fully efficient. This solves one of the fundamental problems of MSM and GMM in general, which is the problem of selecting moments.

3. The reason to use MSM, or a Bayesian version thereof, instead of just the point estimator from the net, is that the net does not give standard error estimates, so it is not clear how to do inference. MSM does give standard error estimates, and allows for construction of confidence intervals. Furthermore, with statistics that come from a trained net, the MSM estimator is just identified, and inferences are more reliable than for overidentified MSM estimators (Creel 2021).

2 Potential paper(s)

Goals:

1. Show that RNNs can give good point estimates for several econometric models. When possible, compare to ML estimation, to verify full efficiency.

2. Go on to apply Bayesian MSM to do inferences, and apply the methods to a real problem with real data.

3. Formalize theory. This would require another coauthor. I have a contact, and can ask if he's interested.

Organization of paper(s)

Goal 1 could be done in a paper of it's own. This would be a relatively simple, short paper. It would probably not get published in a very high ranked journal, as it would be technical and without strong theoretical justification.

Then, another paper could build on the first, and do goal 2. This paper might be publishable in a reasonable quality applied econometrics journal.

A paper doing goal 3, with an example(s) from goal 2, could possibly be published in a high ranked journal. Howver, the math to achieve goal 3 may be beyond even a good mathematician.

3 Evidence so far

3.1 Earlier versions, need update

AR1 and Logit models have old code in the archive. These worked pretty well, but need an update. The AR1 is about the simplest time series model possible, so is a reasonable minimal example. Could use MA1 instead. The Logit model is a simple cross sectional model. These can be updated using the Garch code as a template.

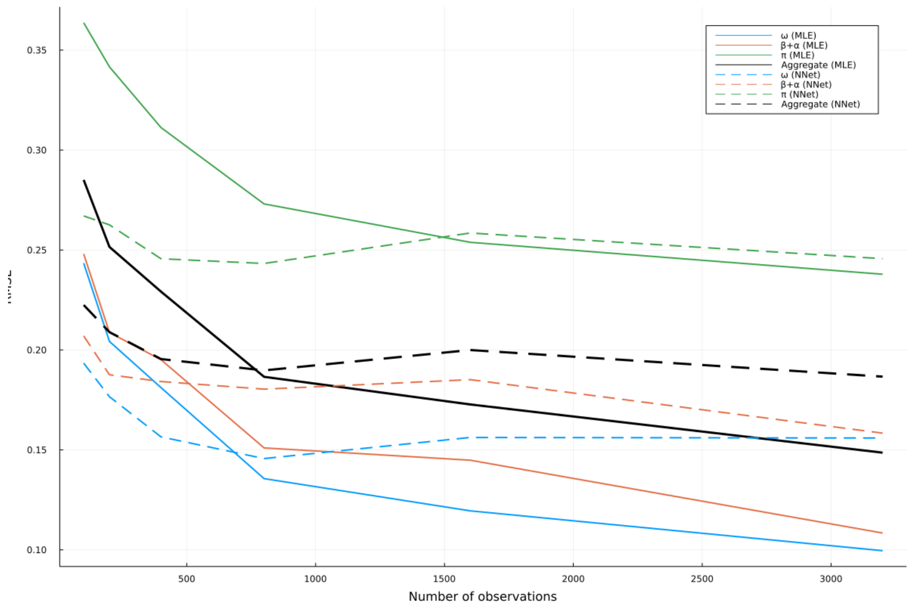
3.2 GARCH

This is in the current code, and is working quite well. The ML estimation is done using simulated annealing, which allows enforcing the bounds of the prior, which makes comparison with RNN fair, as the RNN also uses the prior. The RNN results show declining RMSE over samples of size 100, 200, 400, 800, but then as we go to 1600 and 3200, it is flat or slightly increasing. For reference, aggregate RMSE from the prior is 0.29, so both RNN and ML are improving over the prior. We would like to see RMSE for the net to decline to zero as N gets very large (consistent estimation).

A main objective for Goal 1 should be to find a way to make RMSE decline uniformly as N increases. Different net architecture, different training, or perhaps the LSTMs are forgetting the first part of the sample?

One idea is to get predictions for subsamples of say perhaps 400, where RMSE is declining well, and average them. The code is currently doing averaging of predictions for each observation, here:

Figure 1: Importances of statistics, jump diffusion model




```
# Compute prediction and error
#Ŷ = StatsBase.reconstruct(dtY, nnet(X[end]))
# Alternative:  this is averaging prediction at each observation in sample
Ŷ = mean([StatsBase.reconstruct(dtY, nnet(x)) for x ∈ X])
err_nnet[:, :, i] = Ŷ - Y
```

which is intended to have this effect. Perhaps it's not the best way, though.

4 Work plan

I suggest that you work with the Garch model and try to make RMSE decline for larger sample sizes. I will take the old code and update it to work with the newer framework. That will help to reinforce the current evidence, and may help to identify ways to improve RMSE for larger sample sizes.

References

Chernozhukov, Victor and Han Hong (Aug. 2003). "An MCMC approach to classical estimation". In: *Journal of Econometrics* 115.2, pp. 293–346. DOI: 10.1016/s0304-4076(03)00100-3. URL: http://dx.doi.org/10.1016/s0304-4076(03)00100-3.

Creel, Michael (2021). "Inference using simulated neural moments". In: *Econometrics* 9.4, p. 35. URL: https://doi.org/10.3390/econometrics9040035.

Hornik, K., M. Stinchcombe, and H. White (July 1989). "Multilayer feedforward networks are universal approximators". In: *Neural Netw.* 2.5, pp. 359–366. ISSN: 0893-6080. DOI: 10.1016/0893-6080(89)90020-8. URL: http://dx.doi.org/10.1016/0893-6080(89)90020-8.