Referee Report on “The Approximate Solution of Finite-Horizon Discrete Choice Dynamic Programming Models: Revisiting Keane and Wolpin (1994)”

On the first round I pointed out two main limitations of the paper. The first was that the author never wrote out the choice probabilities or the likelihood function for the model, and he didn’t explain how they are simulated. This is now explained fully in appendix section B on “computational details,” especially the section labelled “function smoothing.” However, I am disappointed to see this relegated to an appendix. I think this short section on “function smoothing” is important enough that it should be included in the main text.

My second suggestion was that the author should do something to address the increase in computation speed since 1994. For example, one might ask “how much bigger of a problem could be solved today to the same accuracy and computation time. One easy way to look at this might be to expand the state space by increasing the time horizon (*T*).” The author now addresses this question in the short appendix section B.2 on “historical perspective.” Again, however, I am disappointed to see this relegated to an appendix.

I would argue that if the two short sections mentioned above were included in the main text it would increase the length of the paper by at most 1 page while making it much more interesting to readers. I hope that some arbitrary word/page limit doesn’t preclude this. Also, I think that the exposition on page 1 to 6 could be tightened up a bit with some careful editing (not requiring any loss of substance). For example, Section 2.3 is a pretty long-winded way to say you estimate the model by simulated maximum likelihood (SML).

Regardless, I think that what the author is doing here provides a very useful service in making DCDP model estimation accessible to a wider audience – particularly through the very useful computer code resources that he provides. I just have a few other expositional suggestions:

If I understand correctly, on pages 8 and 9 the author assesses the quality of both (i) approximations and (ii) estimations done based on those approximations, using a novel root mean square error (RMSE) criterion. For instance the model that KW estimated had 26 parameters (see Table A5). This raises the question of how one evaluates the <u>overall</u> quality of the SML estimates. KW simply assessed (over repeated estimations) how many of the 26 parameter estimates departed significantly from the true values (as is common in Monte Carlo studies). But of course, in this type of model we are not really interested in the estimates *per se*, but rather in the choice probabilities implied by those estimates. So the author assesses the overall quality of the estimated models by comparing (a) estimated choice probabilities (based on the approximation and/or parameter estimates) with (b) the true probabilities generated by the model solved exactly and with the true parameter values. Specifically, he calculates the RMSE of the difference between the true and estimated probabilities. I imagine this approach has some limitations of its own, but it strikes me as an interesting alternative to the usual approach.

However, I do think this RMSE approach has to be explained a bit more explicitly in the paper. It seems to be novel - at least, I haven't seen it done before – and so it is likely to confuse some readers (it took me a while to figure out what was going on).

A related question is that on the bottom of page 8 the author reports an RMSE of 0.06. How do we interpret that as good or bad? Could it be put in percentage terms?

A minor point is that Table 1 looks a bit odd. The two rows of all zeros don't convey any useful information.

Finally, I am confused about the timings reported in section B2 (the part I would like to see in the main paper). In comparing the speed of computers in 1994 to that in 2016 the author states that the fastest computer went from 143 GigaFLOPS in 1994 to 93 Peta FLOPS in 2016. That is about 650,000 times faster, at least in terms of floating point operations. So I am puzzled why the time to compute the exact solution of the KW model only decreases from 50 minutes to 9 minutes? Is it because solution of these models requires lots of other things besides floating point operations (e.g., integer operations), and those things haven't gotten as much faster? Is it because hardly anyone really has access to the currently fastest machine? I would love to hear an explanation of what is going on. It seems very surprising to hear that the time to solve this model has only improved by a factor of about 6 in 22 years.