

Some Guidelines for Reproducing the Krusell-Smith Algorithm

- I) Before starting with the algorithm, it is best to figure out what the exogenous joint transition probabilities for the aggregate and idiosyncratic states are, given the KS calibration in the paper and the restrictions that these transition probabilities have to satisfy.
- II) Draw a T vector of aggregate states using the aggregate transition probabilities. Suggestion: draw $T=1200$ and throw the first 200 away, when you run your updating regression for the forecasting rule. Hint: before you draw these series, I suggest you fix the seed of the random number generator, so that it always draws the same series for you (you want this to be able to compare different runs of your code). Matlab has two basic random number generators, one for a uniform (*rand*) and one for a standard normal (*randn*). Both allow fixing the seeds. Check the help sections of both commands!
- III) Draw a $T \times N$ vector of idiosyncratic states and make sure that you condition your transition probabilities on the aggregate states.
- IV) I like to write my code along the lines of the economic problem. Hence, I write an overall matlab code (a script as they call it in matlab that can be executed). This script then calls - in a loop - three subfunctions: (i) a value function iteration and policy function finding routine; (ii) a routine that computes and approximates the policy function on a much finer grid than (i), for simulation purposes; (iii) simulation and updating. I will describe each in turn.
- V) Input into this subfunction: the current guesses for the law of motion parameters. Output: a value function: $k' = k'(k, \varepsilon, z, \bar{k})$, i.e. values for k' for all grid combinations. Pick a grid on the (k, \bar{k}) plane. Be not equi-spaced along the first direction and very fine around zero. Go for equi-spaced in the second direction. Possible ranges can be picked up from the graphs in the paper. Two suggestions: (i) for the optimization, (ii) for the interpolation and approximation. (i) A good idea is to check for corner solutions first, so it may be best to first evaluate the maximand in the Bellman equation at $k=0$. And then at $k=\varepsilon$, where ε is small, maybe 10^{-3} . If the value goes down, you can invoke concavity and the corner is the optimum. For those states, where the value of the maximand goes up, use a search routine to compute the optimum. I would suggest a bracketing algorithm with the two corner values $[\varepsilon, \max(k\text{-grid})]$. The matlab function *fminbnd* should be good. Hint 1: this function can only search for an optimum one state combination at a time, which is slow (you may want to program a vectorized version of the bracketing algorithm in Judd). Hint 2: check that the upper corner of the k -grid is never hit. (ii) Do your interpolation in a hierarchical fashion, first for the \bar{k} direction: as an intermediate step compute values for the value function on the implied \bar{k}' grid, but with today's capital grid (just as an auxiliary device). Then view \bar{k}' as fixed and combine it with the truly exogenous states to search over the optimal k' for each combination. I would do the first interpolation, using the *interp1* function, with a '*linear*' or a '*pchip*' specification, the second one with a '*spline*' spec. Check help for guidance. Play around with these functions to make sure you understand their syntax well.
- VI) Input into this subfunction: the value function on the coarse grid. Output: a policy function on a much finer grid for both k and \bar{k} . Using the value function on the coarse grid and the interpolation methods from above, compute in a one-shot optimization problem a new policy function on a finer grid.
- VII) Input into this subfunction: the fine policy function. Output: an updated guess for the law of motion parameters and an R^2 . Simulate the economy using the fixed draws for ε and z and the fine policy function. Start with a mass point on k , but with the stationary distribution for ε . Discard the first 200 points in time and do an OLS on the remaining aggregate capital series.
- VIII) Iterate on V) to VII) until convergence and R^2 is high.
- IX) Final hint: matlab has so called global variables that can be declared in the script code and then used in all the subfunctions, if declared as global there. That allows fixing parameter values once and for all and have them in all the subfunctions as well.

GOOD LUCK and HAVE FUN!