Computing the Krusell-Smith Model: A Personal Experience

Toshihiko Mukoyama
Georgetown University
tm1309@georgetown.edu

May 2019

Abstract

The purpose of this note is to record my experiences in computing Krusell and Smith (1998) model and its variants. The sample computer code can be downloaded from: `https://sites.google.com/site/toshimukoyama/KrusellSmithlog.zip`.

1 Introduction

The purpose of this note is to record my experiences in computing Krusell and Smith (1998) model and its variants. I have computed this class of model (with my coauthors) in four papers: Mukoyama and Şahin (2006), Krusell et al. (2009), Krusell et al. (2010), and Krusell et al. (2012). (I also worked with a similar model in Lee and Mukoyama (2008, 2018), but not as the main model.) The ways I compute this class of models have evolved over time, as I gain experiences.

I have never written any computational-method papers in the past, and the main reason that I haven't done it is I feel that computation is not my comparative (and absolute) advantage. There are so many people who are far better at computation than me. I am also aware that there are already many textbook-type treatments, lecture notes, and survey papers for the computation of the Krusell-Smith model.

The main reason I decided to write this note is that I have many times been asked lessons I have learned in computing this type of models, especially from young scholars and graduate students. Some of the papers I have written involve more complex models than the baseline model, and there are some general lessons I learned during the process of trying to solve these models. I am also aware that there are many new computational methods being developed in recent years. Many of these new methods (which I have never used in my papers, although I am trying to learn some of them) come out with ambitions of creating "computational suites" in a similar style as Dynare. In a way, these new methods are intended to help creating papers like factory-produced Ford Model T cars, while my codes are like older cars with handmade parts. I wanted to record the handmade methods before that type of skill becomes forgotten. The handmade skills often become useful when one tackles a type of models that are not very similar to the standard model.

I have started learning how to compute heterogeneous-agent models after I started my first job at Concordia University. My Ph.D. thesis was about growth theory, and I could barely use Matlab at the time I started my job. Sun-Bin Kim, my colleague at that time, was so kind that he shared his Fortran code for Chang and Kim (2006), which was not published yet. He also shared his routines and I still use some of them. He also answered many questions of mine. My coauthor Ayşegül has an engineering background and she is by far a better programmer than me; reading Sun-Bin's code with her also was a great learning opportunity. The way Mukoyama and Şahin (2006) is computed was therefore heavily influenced by the Chang and Kim (2006) code.

The project I started working on after that paper (Krusell et al. (2010)) required me to have a better skill, and I kept learning. Heer and Maußner's (2005) book came out, and it recommended iterating over the density function in computing the distribution of agents (often referred to as "non-stochastic simulation"), and I started using that method (the standard reference is Young (2010)) in simulation. I also read the original Krusell and Smith (1998) code on Tony Smith's website. Writing codes for Krusell et al. (2012) added further challenges.

More recently, I started working on papers with firm heterogeneity that involves similar computation. Lee and Mukoyama (2008, 2018) did not require us to use Krusell and Smith (1998) method, because the model had a "block-recursive" structure, but the alternative model in the Appendix did. The experience of writing Mukoyama and Osotimehin (2019) made me think about what to do with the fat tails, although that paper does not involve business cycles. The issue of fat tail shows up in the original Krusell and Smith (1998) model as well, especially when the discount factor parameter is heterogeneous across agents (and stochastic), which is the one that is computed in the sample code.

2 Model

In this section, I will outline the Krusell and Smith (1998) model (with heterogeneous discount factor) just for the sake of completeness. Time is discrete and all markets are perfectly competitive. Firms produce goods by renting capital and hiring labor from consumers. Consumers make consumption-saving decision. Labor supply is governed by an exogenous stochastic process, which can be interpreted as unemployment shock.

Firms have Cobb-Douglas production function $Y = zK^\alpha L^{1-\alpha}$. The productivity z is subject to an aggregate shock and takes two points $z = z_g$ ("good" state) or $z = z_b$ ("bad" state), where $z_g > z_b$. The shock follows a Markov process. The aggregate labor is exogenous and the labor supply shock is constructed so that L takes two values (that are perfectly correlated with z), L_g and L_b, where $L_g > L_b$. In the competitive market, the wage is $w(K, z) = (1-\alpha)z(K/L)^\alpha$ and $r(K, z) = \alpha z(K/L)^{\alpha-1}$. I suppress L in the argument of r and w functions because z and L are one-to-one.

The consumer's decision can be summarized by the Bellman equation

$$v(k, \epsilon, \beta; \Gamma, z) = \max_{c,k'} \{\log(c) + \beta E_{z',\epsilon'}[v(k', \epsilon', \beta'; \Gamma', z')|\Gamma, \epsilon]\}$$

subject to

$$c + k' = r(K, z)k + w(K, z)\ell\epsilon + b(1-\epsilon) + (1-\delta)k, \tag{1}$$

$$\Gamma' = H(\Gamma, z, z'), \tag{2}$$

and

$$k' \geq \underline{k}. \tag{3}$$

Here, the consumers make consumption-saving decision (between consumption c and next-period capital k') subject to the budget constraint (1) and the borrowing constraint (3), where $\underline{k} \leq 0$ is a parameter that is larger than the natural borrowing limit. On the right-hand side of (1), k is the current period capital holding of the individual and $\epsilon \in \{0, 1\}$ is the employment state, where $\epsilon = 1$ represents employment and $\epsilon = 0$ represents unemployment. The parameter $\ell > 0$ represents (fixed) hours and $b > 0$ is a parameter representing the home production income when the individual is not working. The parameter $\delta \in (0, 1)$ is the depreciation rate. The employment state ϵ follows a Markov process, and the transition probability from ϵ to ϵ' depends on z and z'. Note that ϵ shock is idiosyncratic, and the only way to insure against that shock (and other shocks) is to self-insure by accumulating capital. The stochastic process for ϵ is constructed so that, with the continuum assumption, the unemployment rate (and therefore L) takes two values that is perfectly correlated with z. The discount factor β also is idiosyncratic and follows a Markov process.

The value of the aggregate capital, K, matters for the agent's decision because K affects the prices r and w. This fact implies that the individual has to forecast K', because K' is in the next period value function. However, in equilibrium, the law of motion for K' cannot be written only by K and z. This is because the individuals are heterogeneous (and therefore the saving rates can differ across individuals) and how the assets are distributed across people matter for the total saving next period. This implication is one of the differences of this model compared to the representative-agent model, because in the representative agent model only the current K matter for K'. In general, the entire (joint) distribution of (ϵ, k) matter for determining K' for the next period, and this distribution is denoted as Γ here. Note that Γ contains the information of K (because we can integrate Γ along k to obtain K) so that K is not repeated in the value function. This dependence of the value function on Γ means that the law of motion for Γ' is necessary, and it is the equation (2). Note that (2) contain z' in the right-hand side, because the distribution of ϵ' also depend on z'.

3 Computation

The Bellman equation above is very difficult to solve, because the value function includes Γ, which is a very large object. The main innovation of Krusell and Smith (1998) is to come up with a practical method of solving this problem.

Krusell and Smith (1998) assume that the individuals are boundedly rational in the following two senses: (i) individuals use only a subset of the information contained in Γ when making decisions; and (ii) individuals do not use the exact law of motion (2). They argue that if these boundedly-rational agents can predict K' perfectly in equilibrium, this equilibrium is also an equilibrium under rational expectations, because the behavior of these agents are exactly the same as perfectly rational agents. Perfectly rational agents can gain nothing by using extra information and a more sophisticated law of motion beyond (i) and (ii).

The practical question is what subset of information and what law of motion to use. In principle, any subset of Γ and any law of motion can be used for a boundedly-rational agent. Krusell and Smith (1998) found that in the context of this particular model, K, z, and a log-linear decision rule delivers almost perfect forecast of K'. Instead of (2), they suppose

$$\log(K') = \begin{cases} a_0^K + a_1^K \log(K) \text{ when } z = z_g \\ b_0^K + b_1^K \log(K) \text{ when } z = z_b. \end{cases} \tag{4}$$

Krusell and Smith (1998) provide two intuitions for why this approximation works. First, agents are in general fairly well-insured in this class of model. The reason is that they live for infinite periods and they have some chances to accumulate k to be sufficiently far away from $\underline{k}$. Well-insured agents' decision rules are close to linear, because the complete-market version of the model has the Gorman-aggregation property. (These preferences are often chosen because they typically have the balanced-growth properties, too. Both Gorman-aggregation property and balanced-growth property require some form of homotheticity.) Second, even though the agents who are close to $\underline{k}$ have fairly nonlinear behavior, their wealth holdings are too tiny to have any effect on the aggregate K. The market for capital is not "one person, one vote." This intuition is important because in the case of labor supply, for example, it is closer to "one person, one vote," leading to a difficulty in applying this method.

The computation steps are as follows. In the sample code, `Main.f90` is the main program file and `Globals.f90` defines the global variables and parameters.

1. Create discrete grids on k and K (`ConstructGrids.f90`). Set up the Markov matrices for shocks (`TransitionProbability.f90`). Initialize (`Initialize.f90`).

2. (Loop): Guess the coefficients on (4).

3. With the guess, one can solve the Bellman equation for consumers (`SolveValueFunction.f90`).

4. Simulate the economy for many periods (`SimulateData.f90`). Throw away the first part of the simulation. Record the time series of K and z.

5. Run an OLS regression, in the form of (4), using the simulated data above (`RegressLOM.f90`).

6. (Close/repeat loop) if the regressed coefficients are similar to the first guess, close the loop. If not, update the coefficient and repeat.

The sample code uses the golden section search (`goldensection.f90`) and linear interpolation in the optimization. This is for the ease of exposition, and there are better ways of computing these. The codes for Krusell et al. (2009) and Krusell et al. (2010) are in the public domain and they use better methods in general.

4 Specific issues

Here, I will list some of the detailed issues that I encountered in my experiences in computing this class of models.

4.1 Programming language and computing environment

Sun-Bin Kim used Compaq Fortran, and I adopted the same language. Now the Compaq Fortran is sold to Intel and is called Intel Fortran. I use MATLAB mainly for drawing figures using Fortran output. I have been using IMSL with my Fortran, and it has been convenient, but recently my young coauthors complained that they don't have IMSL so that I often had to rewrite the codes without it. The sample program uses IMSL. I do not have strong opinions about programming languages in general. If one is good at matrix operations, MATLAB can be quite fast, but I am very bad at matrices (I keep making coding mistakes with rows and columns) and Fortran has been more forgiving with my use of loops. For me, avoiding human errors is very important so that I try to keep my codes intuitive. Most of my routines come from Numerical Recipes.

My computations are mostly done in PC. I sometimes parallelize some part of codes (exploiting OpenMP) by adding a few lines, and it helps with the speed, but I don't do much more. I have no experience with MPI or clusters. I prefer to work with my PC (or workstation), rather than in the format of submitting a job and collecting the results later, because I like to see how the program runs in real time. I do a lot of back-and-forth between the outputs and the codes when I code. I don't really have formal education in programming, and thus for a lot of things I am sure my practice is not the best practice, but it has been working for me.

4.2 What is the best method for optimization?

In Sun-Bin Kim's program for Chang and Kim (2006), they used a variant of grid search for optimization. One of the reasons, I believe, that they used that method is that they were worried about the discrete-choice nature of the indivisible-labor model in Chang and Kim (2006). The discrete choice nature creates two issues: (i) the objective function may involve kinks; and (ii) the objective function may not be globally concave. I will talk more on this later. In their original code, Krusell and Smith (1998) wrote down the first-order conditions explicitly and used a simple Newton method to find roots. In his case, we know that the objective function is well-behaved, and a simple and straightforward method works well. In my earlier papers (like Krusell et al. (2009) and Krusell et al. (2010)), we used to do something similar—our model can be a bit more complex, but we could still use cubic-spline interpolation and combine it with something like Brent's method. When I get more worried, I use linear interpolation which can keep the "problematic behavior" local, and use non-derivative methods such as the golden section search for finding maximum. I am sure that there are more powerful off-the-shelf optimization routines, but I have never felt a need for it myself at this point. I like the transparency and stability of bisection and golden section search and I almost always use them as the first methods I try.

4.2.1 Choice of grids on individual wealth in optimization

Often I have to discretize a continuous state space. For the individual wealth, the first thing to do is to decide on the lower bound and the upper bound for the grids. Let me call these $\underline{A}$ and $\bar{A}$. The choice of $\underline{A}$ is fairly obvious—the borrowing constraint. I choose $\bar{A}$ by trial and error—raise $\bar{A}$ until the choice of $\bar{A}$ have no significant impact on the aggregate wealth. This typically requires $\bar{A}$ to be hundreds of times larger than the average wealth; this is more so with stochastic β as in the sample code. In such a situation, it is not so useful to have evenly-spaced grids; the number of grids become too large with reasonable amount of grids around the average value (where the majority of people are). We don't want to ignore the upper tail either, because their decision counts significantly to the average.

We also know that the decision rules are close to $\log(k')$ being a linear function of $\log(k)$ when k is large—this is because (i) the consumers are well-insured when a is large, and (ii) for a well-insured consumer, the saving rate is close to constant given the log utility, and the income for a high-k agent mostly come from k. The natural choice, therefore, is to make the grids "log-spaced": I define

$$\underline{M} \equiv \log(\underline{A} + \gamma)$$

and

$$\bar{M} \equiv \log(\bar{A} + \gamma),$$

where $\gamma > 0$. I usually choose γ so that

$$\gamma = -\underline{A} + 1,$$

so that $\underline{M} = 0$ (note that $\underline{A}$ is typically a negative number). There is no particular reason that I make $\underline{M} = 0$ (other than the requirement $\underline{A} + \gamma > 0$), but this has worked well for me. Then I can define the equally spaced grids on $[\underline{M}, \bar{M}]$: call them $m_0, m_1, ..., m_n$. Then we can define the asset grids $k_0, k_1, ..., k_n$ by

$$k_i = \exp(m_i) - \gamma.$$

This construction also is in line with Tony Smith's recommendation of having more grids around where there are more curvatures in the value function and the decision rules—these functions typically have more curvature close to the borrowing constraint. Note that we can also control the concentration of grids at the lower part of the interval by changing the value of γ.

4.2.2 Interpolation methods

Optimization requires interpolating the value function off the grid points. I typically like the cubic spline interpolation when the problem is well-behaved; it is of a similar speed to linear interpolation and the function is smooth (which means I can even take first-order conditions). In the sample code, I use linear interpolation. This has a downside of being (typically) less accurate than the cubic spline interpolation and it is not differentiable. The good part of the linear interpolation is that, if there is something wrong in the grid interval $[k_i, k_{i+1}]$, it doesn't affect what happens at another interval $[k_j, k_{j+1}]$. This is not the case for cubic spline. This property can be important when there is a suspicion that the value function may contain some parts that are ill-behaved (such as sudden changes in slopes due to discrete choice). Most of the time, I start from linear interpolation anyway, and change it to cubic spline as I become more confident that the problem is well-behaved.

4.3 How should we compute the stationary/moving distribution?

There are potentially many methods in computing the stationary distribution in the Bewley-Huggett-Aiyagari model. I have always opted for some kind of simulations, but there are other methods, such as directly finding the stationary distribution from the Markov transition matrix. I prefer simulation methods because these are typically very stable (due to the fact that the mapping is typically a contraction mapping). Initially I have used "simulating many people (or one person) for a long time" method, as in the original Krusell and Smith (1998) paper (note that we have to make sure to use the same seed in each loop for generating the random number for idiosyncratic shocks), but I have moved on to working with the density function directly. The main reason is that it requires really large number of people (in the order of millions) to achieve a satisfactory accuracy for the simulation with this method. The law of large numbers work only with a really large number.

The method is described in Heer and Maußner (2005) and Young (2010). The method first define the measure of agents on the grids of (ϵ, k) as a histogram. That is, for each combination of (ϵ_i, k_j), there are $\mu(\epsilon_i, k_j)$ number of people who are in that state. Using the Markov transition matrix for ϵ and the decision rule for k', from (abusing) the law of large numbers, we know *exactly* how many people will be at each ϵ' and k'. One issue is that k' described in the decision rule may not be exactly on one of the grid points. In that case, we linearly interpolate. For example, if k' falls on the interval between the grid points k_i and k_{i+1}, we can describe k' as $k' = \eta k_i + (1-\eta)k_{i+1}$ where $\eta \in [0,1]$. In this case, we move η fraction of people to the grid point k_i and $(1-\eta)$ fraction of people to the grid point k_{i+1}. Note that this interpolation preserves the aggregate value of k': if there are N people on a particular (ϵ, k) grid, their next-period total wealth Nk' is equal to $N(\eta k_i + (1-\eta)k_{i+1})$, while with the linear interpolation $N\eta$ people will be placed at k_i and $N(1-\eta)$ people will be placed at Nk_{i+1}, summing up to the same value. This "aggregation-preserving" property is regardless of how the grids are spaced. For the stationary distribution, repeat this until the measure (histogram) converges.

We can use the same method for the Krusell-Smith model as well, this time with simulation. Starting from a distribution (histogram), for each period, given the realizations of the z shock, we can compute the histograms $\mu_t(\epsilon_i, k_j)$. Throw away the first part of the simulation (2000 periods in the case of the sample code) for the outcome to be unaffected by the initial condition. Then use the rest for the purpose of running the OLS regression.

The k grids for this part of the code does not have to be the same as the ones in the optimization part. Overall, it is more desirable to have more grid points than the optimization part, for the sake of accuracy. Regarding how to place the grids, I have thought of two options.

Heer and Maußner (2005) and Young (2010), implicitly or explicitly, recommend evenly-spaced grids. It seemed a reasonable choice, and I have been using evenly-spaced grids in all of my published papers. However, now I believe that there is a better way of placing grids, especially when we need to take a wide space due to thick tails in the asset distribution (such as the case with stochastic β in Krusell and Smith (1998)). In these situations, I recommend using log-spaced grids, similarly to how I described in Section 4.2.1, with more grids than the optimization grids. It turns out that the latter method can save the total number of grids dramatically (the log-spaced grids can achieve a similar accuracy as evenly-spaced grids with 1/10 number of grids), and thus I use this method in the sample code. Linear interpolation is performed similarly, as the "aggregation-preserving" property described above is an attractive property in this context.

4.4 Initial values

The initial conditions for value function iteration and the simulation can be fairly arbitrary, as mappings that generate these have contraction properties. The outer loop of updating OLS coefficients for (4) is not a contraction, and the choice of the initial value is important. The method Sun-Bin recommended me is to run a corresponding representative agent model and perform the same OLS regression on its model-generated data, and use that coefficient as the initial value. I think it is a useful recommendation in general; if we have a model that is similar and easy to compute, use that as a starting point.

There are two suggestions that I can make when the coefficients don't converge. First is to start from small values of a_1^K and b_1^K. When $a_1^K = 0$ and $b_1^K = 0$, the agents hold a static expectation on future K with $\log(K) = a_0^K$ and $\log(K) = b_0^K$. With a reasonable values of a_0^K and b_0^K (for example, the steady-state value of the corresponding Aiyagari model or even the representative-agent model), the simulation outcome won't be very unreasonable. Then gradually move all coefficients using very small weight (e.g. 0.01) on the new OLS coefficients and the rest of the weight on the pre-updating values. The second suggestion is to "think like the agents in the economy." If the simulated K increases rapidly, it means that the returns from saving is too high. This is likely a result of the expected value of r in the future being too high, which is caused by the expected values of K in the future that are too low. And the expectations on the future K can be managed through the coefficients on (4). This logic helps finding a reasonable initial set of values by trying to change the coefficients in the direction of removing the wrong bias from the agents' expectations.

Another related tip is to start from a wide range of grids on K. Typically I end up putting 15% above and below the steady-state value for the grids on K at the end, similarly to the typical RBC model. In the equilibrium simulation, the simulated series of K almost never goes beyond these upper and lower bounds. But when I try to find a good initial values for coefficients on (4), I start from much wider range, say, 90% above and below the steady-state values. This is because when the coefficients are not at the right values, the simulated K drifts up or down quickly. These drifting time series still have some information on the true time series of K, so that we don't want to waste that information. By taking a wide range of grids, we can capture the behavior of K before it hits the upper or lower bound.

4.5 Extending to more complex models

Some of the models that I have worked on, especially Krusell et al. (2010) and Krusell et al. (2012), are substantially more complex than the basic model. The principles are the same as the basic model, but I would like to mention three points that I think are relevant in solving these complex models.

4.5.1 Other markets that have to clear every period

In some applications, there are other markets that have to (nontrivially) clear at each period. For example, in Krusell and Smith (1998), there is an example where the agents make labor supply decisions and then the labor market has to clear. The same is the case for Chang and Kim (2006). In Krusell and Smith (1997), the bond price has to be set so that the asset markets clear (for bond and capital). There are two potential approaches to tackle this situation.

The first is the natural extension of Krusell and Smith (1998) method. In fact, (I recently realized that) Krusell and Smith (1998) use this method in the endogenous labor supply example. Suppose, in that context, that the total quantity of labor L in equilibrium follows

$$\log(L) = \begin{cases} a_0^L + a_1^L \log(K) \text{ when } z = z_g \\ b_0^L + b_1^L \log(K) \text{ when } z = z_b. \end{cases} \tag{5}$$

Similarly to the forecasting equation on K', we can make a guess the coefficients on (5) first (it is weird to call (5) as a forecasting equation because the agents have to "see" the prices at each period, but I will use that terminology anyway). Knowing state variables K, z, and with the forecasting equation (5), the consumers can tell the prices r and w, and thus can solve the Bellman equation. With individual decisions on the labor supply, we can simulate the actual aggregate values of L in the simulation. At the same time as we update (4), we can update (5) with the OLS regression using the simulated data. This is the method we used in Krusell et al. (2012). Chang and Kim (2006) opted for something different: they created forecasting equations on r and w. I personally prefer to forecast K' and L, but the idea is similar.

The second method is employed by Krusell and Smith (1997) and also recommended by Ríos-Rull (1999). The idea is to first solve the Bellman equation using the forecasting equation (5). Then add an extra step of solving for an auxiliary decision rule where the current period L (and therefore r and w) is given exogenously. This would provide a decision rule that is conditional on the one-period deviation of L. (We applied a similar idea in the computation of the Nash-bargained wages in Krusell et al. (2010).) Then this decision rule is used in the simulation stage—in each period of the simulation, we search for a value of L that clears the labor market using the auxiliary decision rule. This method is basically adding two extra steps (computing the auxiliary decision rule and ensuring the market clearing at each period) to the first method, and thus this method is slower. These extra steps were necessary in Krusell and Smith (1997), where they computed the market-clearing bond price q in each period. In their situation, bond and capital stock are very close substitutes as assets, because consumers are fairly well-insured. Thus only a small deviation of q from the equilibrium value can shift everyone's portfolio choice to "everyone tries to hold all assets capital (and short-sell bond)" or "everyone tries to hold all assets in bond (and short-sell capital)" quite wildly. Thus, trying to find the equilibrium time series of q by moving the coefficients of (5) up and down was not practical. Krusell et al. (2010) used both the first and the second method.

As a side note, in Krusell et al. (2010), we have a similar portfolio choice problem (capital and equity) as Krusell and Smith (1997), but we didn't have much of an issue with corner solutions. The reason is that although capital and equity are close substitutes, we did not let the consumers own capital and equity directly. We assume that there is a mutual fund that owns all capital and equity, convert them into two "aggregate Arrow securities" (security that pays one unit of the consumption good only in one of the aggregate states), and sell them to consumers. In our setting, there are two (uncertain) aggregate states ("good" and "bad"), thus two independent securities (capital and equity) are sufficient to span the two aggregate states and create the aggregate Arrow securities. This "aggregate Arrow security" setup was nice by four reasons. First, because people typically don't want to lose all wealth in one of the aggregate states, the demand for the aggregate Arrow security is typically positive for almost everyone, avoiding corner solutions. Second, because the aggregate Arrow security represents the price of the consumption good at each state, this means that everyone who owns the asset agree on the state prices, Thus these security prices can be used for the basis of firm investment, circumventing a well-known difficulty in the firm decision in an incomplete-market framework. Third, the aggregate Arrow securities can be used for the asset pricing purposes (we used this formulation fully in Krusell et al. (2011)). Fourth, the borrowing constraints based on the aggregate Arrow securities intuitively makes more sense than the constraints based on the individual asset holdings such as capital and equity, because the aggregate Arrow security represents the net asset position at each state.

4.5.2 What should be the right-hand side variables?

As I emphasized earlier, Krusell and Smith (1998) solution method starts from assuming that the agents make decisions based on limited information. There is no fixed formula for what kind of limited information is appropriate—we can use whatever combination of variables (as long as it is in the information set of the agents) that works. In the case of Krusell and Smith (1998), K was sufficient. For another example, in the case of Krusell et al. (2012), we put the capital-labor ratio in the previous period in the right-hand side, as it turned out to be a good predictor for the capital-labor ratio this period.

When the log-linear formulation for the forecasting rule doesn't work, one easy (and cheap) fix I recommend to try is to incorporate a quadratic (or other nonlinear) term in $\log(K)$. This is cheaper than adding more moments (such as second moment) as the only necessary change is adding one term to the forecasting equation. Often adding a nonlinear term improves the fit a lot better. If this method doesn't work, adding more moments, such as second moment or median, is probably the next thing to try.

4.5.3 Discrete choices

The existence of discrete choices, as in Chang and Kim (2006) and Krusell et al. (2012), can potentially cause troubles in computation. With discrete choices, the value function can have kinks and local nonconvexities. In practice, however, if there are sufficient amount of idiosyncratic shocks, these issues can be mitigated in the process of taking future expectations. This was the case for Krusell et al. (2012), which features idiosyncratic productivity shock at individual level. For the idiosyncratic shock to smooth the expected value function, the timing is somewhat important: (i) make a decision on the next period asset, (ii) the realizes for the next period, and (iii) the next period discrete choice is made. The shock has to realize before the discrete choice for the expectation operation to smooth the kink and nonconvexity. In a sense, the shock naturally introduces the Rogerson-style lottery into the indivisible decision.

When the timing does not work out, or there are no shocks to smooth the value function, an option is to introduce an extreme-value taste shocks as in the structural IO literature. I've written a separate note on this issue.
`https://sites.google.com/site/toshimukoyama/DiscreteChoice.pdf`

References

[1] Chang, Yongsung and Sun-Bin Kim (2006): "From Individual to Aggregate Labor Supply: A Quantitative Analysis based on a Heterogeneous-Agent Macroeconomy," *International Economic Review* 47, 1–27.

[2] Heer, Burkhard and Alfred Maußner (2005): *Dynamic General Equilibrium Modeling: Computational Methods and Applications,* Springer.

[3] Krusell, Per, Toshihiko Mukoyama, Richard Rogerson, and Ayşegül Şahin (2012): "Is Labor Supply Important for Business Cycles?," NBER Working Paper 17779.

[4] Krusell, Per, Toshihiko Mukoyama, and Ayşegül Şahin (2010): "Labour-Market Matching with Precautionary Savings and Aggregate Fluctuations," *Review of Economic Studies* 77, 1477-1507.

[5] Krusell, Per, Toshihiko Mukoyama, Ayşegül Şahin, and Anthony Smith, Jr. (2009): "Revisiting the Welfare Effects of Eliminating Business Cycles," *Review of Economic Dynamics* 12, 393–404.

[6] Krusell, Per, Toshihiko Mukoyama, and Anthony Smith, Jr. (2011): "Asset Pricing in a Huggett Economy," *Journal of Economic Theory* 146, 812–844.

[7] Krusell, Per and Anthony Smith Jr. (1997): ""Income and Wealth Heterogeneity, Portfolio Choice, and Equilibrium Asset Returns," *Macroeconomic Dynamics* 1, 387–422.

[8] Krusell, Per and Anthony Smith Jr. (1998): ""Income and Wealth Heterogeneity in the Macroeconomy," *Journal of Political Economy* 106, 867–896.

[9] Lee, Yoonsoo and Toshihiko Mukoyama (2008): "Entry, Exit, and Plant-level Dynamics over the Business Cycle," Cleveland Fed Working Paper 07-18R.

[10] Lee, Yoonsoo and Toshihiko Mukoyama (2018): "A Model of Entry, Exit, and Plant-level Dynamics over the Business Cycle," *Journal of Economic Dynamics and Control* 96, 1–25.

[11] Mukoyama, Toshihiko and Sophie Osotimehin (2019): "Barriers to Reallocation and Economic Growth: The Effects of Firing Costs," *American Economic Journal: Macroeconomics,* forthcoming.

[12] Mukoyama, Toshihiko and Ayşegül Şahin (2006): "Costs of Business Cycles for Unskilled Workers," *Journal of Monetary Economics* 53, 2179–2193.

[13] Ríos-Rull, José Víctor. (1999): "Computation of Equilibria in Heterogeneous Agent Models," In Ramon Marimon and Andrew Scott (Eds.) *Computational Methods for the Study of Dynamic Economies,* Oxford University Press.

[14] Young, Eric R. (2010): "Solving the Incomplete Markets Model with Aggregate Uncertainty Using the Krusell-Smith Algorithm and Non-Stochastic Simulations," *Journal of Economic Dynamics and Control* 34, 36–41.