Tuesday Recap

Official solutions

Dynare part was correct and there should be no errors there (the output of the dynare file ran just once matched mine).

The simulation part of the model is the simple method which uses Monte Carlo integration (when you take the mean of the capital for each period, across N agents in the economy). The error obtained due to simulation can be evaluated (`https://web.northeastern.edu/afeiguin/phys5870/phys5870/node71.html`) and from there you can see that increasing the number of agents lowers your approximation error drastically. Two other methods that Wouter mentioned and did not talk about should be aviding the stochasticity error which may arise. References in the slides should be enough to grasp the error size between the methods (and the graphs Wouter provided). The volatility of the MC method is huge and if you are to concentrate on numerical precision, this is something to think about. Most of the papers simply use MC methods (as numerics is not the main priority there).

I did give a wrong lead when it comes to the **solve_model_agg_uncertainty.m**

- constructing the LHS matrix to be multiplied with **decision** - you have to have all variables corrected by their steady states, so the states in Dynare look like $(k_{t-1} - k_{ss})$, $(z_t - z_{ss})$, $(K^a_{t-1} - K^a_{ss})\ldots$. However, when you calculate the integral which is to be the input of the regression, K^a_t should not be a deviation, but the level.

- the steady state for individual capital k_t can be found in the decision matrix, but needs to be corrected for the second coefficient (named correction in the Dynare output - **decision(1,1)-decision(2,1)**). The aggregate capital steady state should come from our assumption $K^a_t = b_0 + b_K K^a_{t-1} + b_z(z_t - z_{ss})$.

- The misleading part were the coefficients that need to multiply the corresponding variables in the decision matrix. So, even though we do correct for the setady state, coefficients used remain the same - (**decision(1,1),decision(3:end,1)**). However, Wouter multiplies with **decision(2:end,1)** just because he constructs the variables in a different way, but gets to the same result.

- Wouter multiplies the vector of ones with **decision(2,1)**:

  ```
  k_sim_norm(:,t) = state_vector(t,k_sim_norm,z_sim_norm,
      Ka_sim_norm,e1_sim,e2_sim)*decision(2:end,1);
  ```

  and what he gets is

  $$k_{sim\_norm}(t) = decision(2,1) + rest.$$

  However, when the calculation of the policy ends the adjustment (back to $k_s im$ in the level form) is $k_{sim}(t) = k_{sim\_norm} + decision(2,1) + rest + k_{ss} = k_{sim\_norm} + decision(2,1) + rest + decision(1,1) - decision(2,1)$, which yields the same multiplication as I have stated above. Ultimately, he calculates the aggregate capital by integrating the level value of k_{sim} across the agents, for each time period.

My solution

My solution is slightly more student-type of solution, going step by step. I will provide it here also, works in matlab and produces the same thing as the official solutions. I define the huge matrix, calculate deviations every iteration explicitly and multiplying with $[decision(1,1), decision(3:end)]$.

Also, instead of **regress.m** in Matlab, I use the analytical form to calculate b_{new} and use **inv** command, which could be substituted with a left division (then the inverse of the matrix is numerically approximated which lowers the chance of singularity issues).

The reason behind the correction which needs to be accounted for from Dynare output: `https://stephane-adjemian.fr/dynare/slides/dsge-perturbation-method.pdf`. Around slide 42 you will see the loss of certainty equivalence property of the policy function (now the risk matters for agents policy, so we have an additional term on the RHS (in blue)).