Computational Problem Set 1

Michael B. Nattinger

September 14, 2021

1 State the dynamic programming problem

Households choose future capital levels to solve:

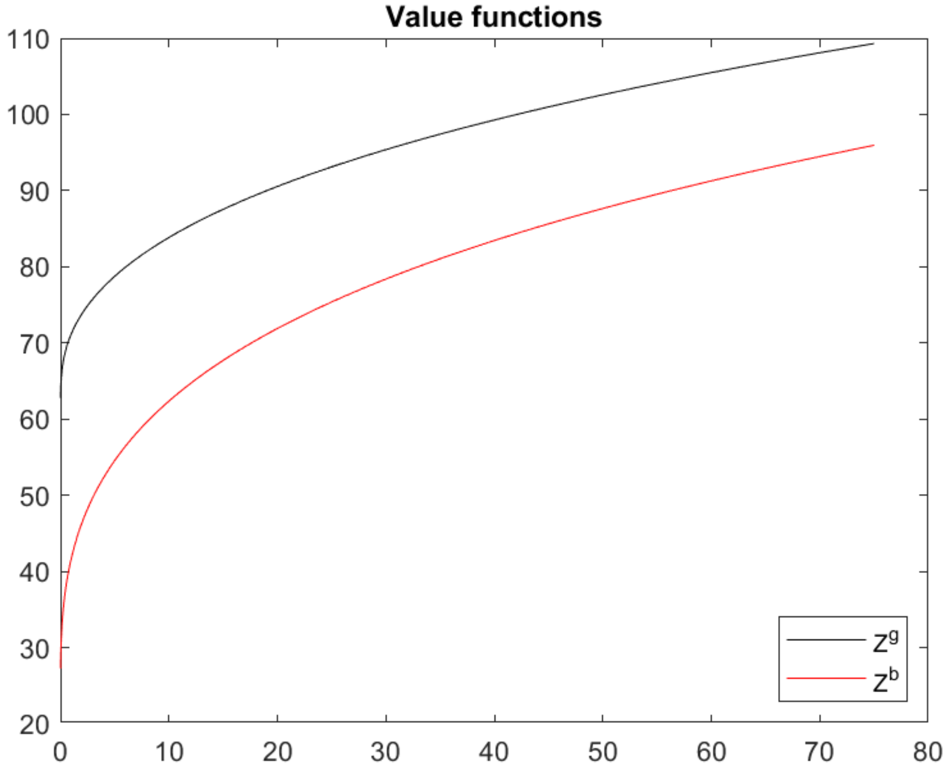
$$\begin{aligned}
V(K,Z) &= \max_{K'} log(ZK^{\alpha} + (1-\delta)K - K') + \beta E_t[V(K',Z')] \\
&= \max_{K'} log(ZK^{\alpha} + (1-\delta)K - K') + \\
&+ \beta[V(K',Z')P(Z' = Z^g|Z) + V(K',Z^b)P(Z' = Z^b|Z)]
\end{aligned}$$

We will proceed to solve in Matlab, Julia, Fortran, and parallelized Fortran, as instructed.

Matlab results: Runtime is 18 seconds (not including figure making). Capital grid was extended to 75 from 45, and has 1000 elements. In Julia, computation is substantially quicker and convergence was achieved in 2.6 seconds to solve the same problem. Note: this is with difference computed as the sup norm of value function changes, and not normalized to the scale of value function, rather absolute size with a tolerance of 10^{-4}. Fortran runs in 5.8 seconds. Figures below are from Matlab solution to the stochastic problem, with a capital grid with 3000 elements instead of 1000. Figures from Julia and Fortran are identical.

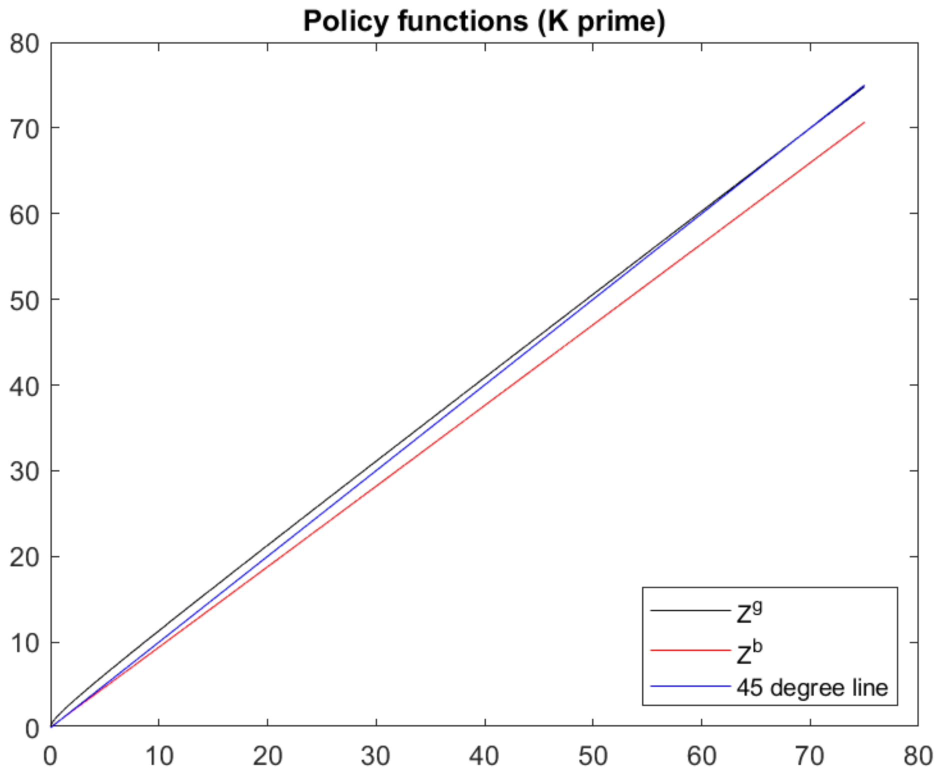
Note also: Matlab code is structured very differently from the Fortran/Julia code and is not optimized for performance. I would not consider this to be a fair comparison of the speed of Matlab; rather, I see it as a reasonably fast baseline for sub-optimal ("naively-written") code.

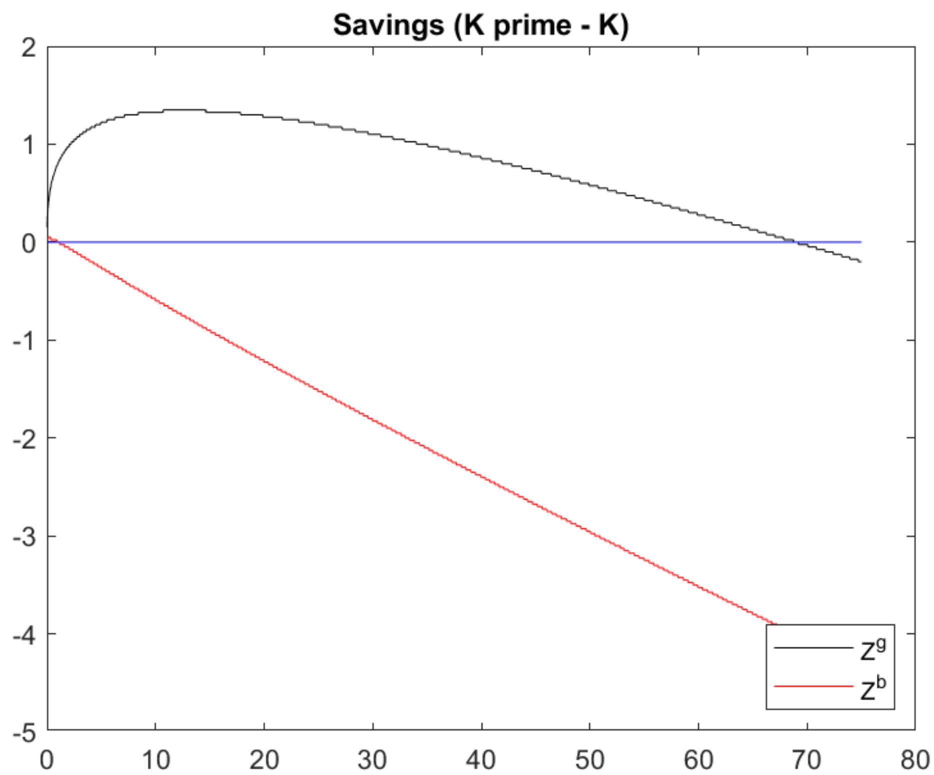
2 Value function



From the above figure, we can see that the value function is indeed increasing and concave.

3 Decision rule





From the above figures, we can see that the decision rule is increasing in K and Z. Saving is not always increasing in K - if households have sufficiently large capital levels, they are better off dissaving. However, saving is increasing in Z - under bad conditions, one will save less than they would under good conditions (conditional on K).