Matlab con Aplicaciones a Programación Dinámica

MODELO NEOCLÁSICO DETERMINÍSTICO DE CRECIMIENTO

Considere el modelo neoclásico de crecimiento:

$$\begin{aligned} \max_{\{c_t,k_{t+1}\}_{t=0}^{\infty}} \quad & \sum_{t=0}^{\infty} \beta^t u(c_t) \\ s.t. \quad & c_t + k_{t+1} - (1-\delta)k_t \leq f(k_t) \\ & c_t \geq 0 \\ & k_{t+1} \geq 0 \end{aligned}$$

Donde $k_0 > 0$ está dado, $u(\cdot)$ y $f(\cdot)$ son estrictamente cóncavas y satisfacen las condiciones de Inada, $\beta \in (0,1)$ y $\delta \in [0,1]$. En clases encontró una solución analítica para ciertas especificaciones funcionales y parametrizaciones. El objetivo ahora será resolverlo numéricamente mediante el método de la Iteración de la Función Valor.
Considere la siguiente parametrización:

$$\begin{aligned} u(c) &= \frac{c^{1-\sigma}-1}{1-\sigma} \\ f(k) &= k^{\alpha} \\ \beta = 0.95, \quad \alpha = 0.33, \quad \delta = 0.1, \quad \sigma &= 2 \\ \text{Tolerancia} &= 10^{-6} \end{aligned}$$

1. A partir de las condiciones de primer orden del problema, encuentre el estado estacionario de k. Esto es, encuentre k_{EE}, tal que $k_t = k_{t+1} = k_{EE}, \forall t$.

2. Lo que buscaremos es una aproximación discreta para el problema continuo. Para eso genere una grilla para k, que contenga 100 valores equidistantes entre 0.1 y $2k_{EE}$.

3. Resuelva el problema enunciado usando iteración de la función valor. Para esto:

   (a) Comience con un valor inicial para $V^{(0)}(k)$. Este debiese ser un vector de dimensión 100×1. El valor más simple es el vector $0_{100\times 1}$.

   (b) Actualice la función valor. Se recomienda realizar esto en un loop, de la siguiente manera: primero fije la variable estado k_i, en algún punto de la grilla (comience con $i = 1$). Luego, para cada valor de k'_j factible de la grilla ($j = 1, \ldots, 100$) calcule:

   $$T_{i,j} = u\left(f(k_i) + (1-\delta)k_i - k'_j\right) + \beta V^{(0)}(k'_j)$$

   Para esto se recomienda construir una matriz C de 100×100 que contenga todos los posibles consumos $c_{i,j}$ y una matriz R de 100×100 que contenga los valores de $u(c)$. Como posteriormente maximizará sobre dichos valores, imponga $u(c) = -10^{10}$ para valores no factibles de consumo (consumo negativo).

   (c) Encuentre la ubicación j que maximiza $T_{i,j}$ y a su vez el valor de la función maximizada.

   (d) Repita el procedimiento para todos los valores de k_i, $\forall i = 1, \ldots, 100$.

   (e) Compute la distancia entre $V^{(0)}$ y $V^{(1)}$, es decir:

   $$d\left(V^{(1)}, V^{(0)}\right) = \max_{i=1,\ldots,100} \left|V_i^{(1)} - V_i^{(0)}\right|$$

   Si la distancia es menor a la tolerancia para todo i, entonces hemos encontrado el punto fijo. En caso contrario, actualice el guess inicial y repita el procedimiento hasta converger.

¿Cuánto tiempo demora en converger? ¿En cuántas iteraciones? Grafique $V(k)$ contra la grilla de capital. Grafique la función de política para el capital y una recta de 45 grados contra la grilla de k. Documente la cantidad de iteraciones y el tiempo que se demoró el programa en converger. Verifique si la convergencia fue monotónica o no analizando la distancia entre las 10 primeras iteraciones y el valor obtenido tras la convergencia, es decir, $d(V^{(n)}, V^*)$, donde V^* es el valor de la función valor cuando converge.

4. ¿Cómo podría traducir el loop en operaciones vectoriales? Recuerde que MATLAB es más eficiente trabajando con vectores y matrices.

5. Replique el ejercicio, pero con los siguientes cambios de parametrización y comente resultados:

   (a) $\alpha = 0.66$, ¿cómo cambia el estado estacionario? ¿Cómo cambia la velocidad de convergencia?

   (b) $\sigma = 5$, ¿cómo cambia la concavidad de la función valor? ¿Cómo cambia la velocidad de convergencia?

6. Fije $\delta = 1$ y $\sigma = 1$ (lo que implica que $u(c) = \log(c)$). Ud. se percatará que bajo esta parametrización existe solución analítica. Resuelva el modelo y luego grafique la función de política, la solución analítica vista en clases y la recta de 45° contra la grilla de capital.

7. **(Desafío)** Este ítem es opcional. Si hace este ítem con éxito significa que usted ha entendido bien conceptos fundamentales de programación dinámica para este curso.

   Algoritmos de aceleración usando propiedades de la función valor:

   (a) **Concavidad de la value function**: Note que la value funciton que resuelve el problema anterior es cóncava. Por lo que, dado i, $T_{i,j}$ tendrá una única solución cuando busque en j. Esto es, fije i, luego a medida que avanza en la grilla de capital para el próximo periodo (j) el valor de $T_{i,j}$, será creciente, alcanzará un máximo en j^* y luego será decreciente. ¿Tiene sentido seguir computando valores de $T_{i,j}$ luego de j^*?

   (b) **Monotonicidad de la policy function en** k. Denote por $g(\cdot)$ la policy function. Luego, suponga que el máximo de $V(k_{i-1})$ se alcanza en $k'_{j^*} = g(k_{i-1})$. Luego, para encontrar el máximo de $T_{i,j}$ ¿tiene sentido buscar en $j < j^*$, si sabemos que $k'_{j^*} = g(k_{i-1}) \leq g(k_i)$?

   Para ver las grandes ganancias de eficiencia compare el tiempo de solución usando estas dos propiedades de la función valor con el tiempo de solución usando: (i) loops para resolver la iteración de la función valor y (ii) vectores y matrices para resolver la iteración de la función valor.