Lecture 5: KKT Condition, Euler Equation and Time Iteration

Yasuyuki Sawada, Yaolang Zhong

University of Tokyo

yaolang.zhong@e.u-tokyo.ac.jp

October 29, 2025

Recap: Markov Decision Process (MDP)

- An MDP is defined by the tuple

$$(\mathcal{S},\ \mathcal{A},\ P,\ r,\ \beta)$$

  where:

  - State space $\mathcal{S}$: Possible system states.
  - Action space $\mathcal{A}(s)$: Feasible actions when in state s.
  - Transition kernel $P_t(s' \mid s, a)$: Probability of moving from s to s' given action a.
  - Reward function $r_t(s, a)$: Instantaneous payoff from taking action a in state s. (Sometimes expressed as a *cost* $-r(s, a)$.)
  - Discount factor $\beta \in (0, 1)$: Weights future rewards relative to current ones.

- Objective: Choose a policy $\pi : \mathcal{S} \to \mathcal{A}$ to maximize expected discounted rewards:

$$\mathbb{E}_\pi \left[\sum_{t=0}^{T} \beta^t r_t(s_t, a_t) + \beta^T R_T(s_T) \right].$$

Recap: Value Function Iteration (VFI)

- Problem Specification:
  - Infinite horizon (Finite horizon problems can be solved by backward-induction)
  - Time-homogeneous: known $P(s' \mid s, a), r(s' \mid s, a)$
  - Didn't specify State space $\mathcal{S}$,Action space $\mathcal{A}(s)$ to be continuous or discrete
- Value Function Iteration (VFI):
  - Iteratively do value update: for each state s,

$$Q_{V^{(k)}}(s,a) \;=\; r(s,a) + \beta\,\mathbb{E}\Big[V^{(k)}(S') \mid s,a\Big] \quad \text{for all } a \in \mathcal{A}(s),$$

$$V^{(k+1)}(s) \;\leftarrow\; \max_{a\in\mathcal{A}(s)} Q_{V^{(k)}}(s,a).$$

  - Policy extraction: greedy w.r.t. $Q_{V^{(k+1)}}$,

$$\pi^*(s) \;\in\; \arg\max_{a\in\mathcal{A}(s)} \Big\{ r(s,a) + \beta\,\mathbb{E}\Big[V^{(k+1)}(S') \mid s,a\Big] \Big\}.$$

Recap: Policy Function Iteration (PFI)

- Problem Specification:
  - Infinite horizon (Finite horizon problems can be solved by backward-induction)
  - Time-homogeneous: known $P(s' \mid s, a), r(s' \mid s, a)$
  - Didn't specify State space $\mathcal{S}$,Action space $\mathcal{A}(s)$ to be continuous or discrete
- Policy Function Iteration (PFI):

  Iteratively do
  - Policy evaluation: compute $V^{\pi^{(k)}}$ as the fixed point of $T^{\pi^{(k)}}$,

$$V^{\pi^{(k)}} = T^{\pi^{(k)}} V^{\pi^{(k)}},$$

  - Policy improvement: for each $s \in \mathcal{S}$,

$$\pi^{(k+1)}(s) \in \arg\max_{a \in \mathcal{A}(s)} \Big\{ r(s,a) + \beta \, \mathbb{E}\Big[V^{\pi^{(k)}}(S') \mid s, a \Big] \Big\}.$$

Motivation: Time Iteration (Coleman PFI)

- VFI and PFI: no free lunch
  - Work broadly for general state and action spaces.
  - Make no use of analytical information on functions or relationships between variables — act as black boxes.
- Potential issues when working with the value function V:
  - Flat V: updates remain small even when far from convergence.
  - When β is close to one, the Bellman operator is only weakly contractive. Over k iterations, the error shrinks as

    $$\|T^k V^{(0)} - V^*\|_\infty \leqslant \beta^k \|V^{(0)} - V^*\|_\infty,$$

    so convergence becomes slow when $\beta \approx 1$.
  - Interpolation errors in continuous state spaces.
- Economic models often have exploitable structure:
  - Utility u and production f are typically continuous, monotone, smooth, and concave.
  - Optimal decisions satisfy first-order conditions, except when constraints bind.

Motivation: Time Iteration (Coleman PFI)

- Idea of time iteration:
  - Instead of re-solving the entire optimization problem at each step, directly impose the first-order conditions that define optimal behavior.
  - This removes the $\max$ operator, improves numerical stability, and supports efficient methods such as the Endogenous Grid Method.
- Goal: iterate directly on a policy $\pi : \mathcal{S} \to \mathcal{A}$ so that the first-order conditions hold given the continuation values implied by π.
- Coleman operator H: defines the updated policy as the solution to the Euler equation:

$$\pi^{(k+1)} = H(\pi^{(k)}) \quad \text{such that} \quad \text{Euler}\big(s, \pi^{(k+1)}(s); \pi^{(k)}\big) = 0 \ \forall s \in \mathcal{S}.$$

- GPI interpretation (Generalized Policy Iteration):
  - Evaluation step: use the current policy $\pi^{(k)}$ to compute the implied continuation values or marginal utilities (the right-hand side of the Euler condition).
  - Improvement step: update $\pi^{(k+1)}$ so that the first-order condition holds at each state — effectively replacing value-function updates with policy updates that satisfy the equilibrium condition.

Roadmap

1. Step 1: Unconstrained Optimization
   Interior solution: first-order condition $\nabla f(x^*) = 0_n$. Interpretation: gradient flatness and Hessian-based classification.
2. Step 2: Equality Constraints — Lagrange Multipliers
   Introduce constraint $g(x) = 0_m$, define Lagrangian $\mathcal{L}(x, \lambda) = f(x) + \lambda^\top g(x)$. Geometric view: tangency between level sets of f and g.
3. Step 3: Inequality Constraints — KKT System
   Generalize to $g(x) = 0_m$, $h(x) \leqslant 0_p$. First-order optimality: stationarity, feasibility, and complementary slackness.
4. Step 4: Dynamic Optimization as a KKT Problem
   Reformulate the intertemporal consumption–saving problem $\max \sum_t \beta^t u(c_t)$ under resource and nonnegativity constraints. Derive the Euler equation and interpret it as the KKT condition over time.
5. Connection to Time Iteration Methods
   Use the Euler condition directly to update policies — leading to the Coleman operator and generalized policy iteration (GPI).

Step 1: Unconstrained Optimization

- Consider an unconstrained optimization problem:

$$\min_{x \in \mathbb{R}^n} f(x), \quad f : \mathbb{R}^n \to \mathbb{R}.$$

- A necessary condition for an interior optimum x^* is that the gradient vanishes:

$$\nabla f(x^*) = 0_n,$$

  where $\nabla f(x^*) \in \mathbb{R}^n$ is the $n \times 1$ gradient vector.
- This identifies critical points where $f(x)$ stops increasing or decreasing in any direction.
- Whether a critical point is a minimum, maximum, or saddle depends on the Hessian $\nabla^2 f(x^*) \in \mathbb{R}^{n \times n}$:
  - $\nabla^2 f(x^*) \succ 0 \Rightarrow$ local minimum
  - $\nabla^2 f(x^*) \prec 0 \Rightarrow$ local maximum
  - $\nabla^2 f(x^*)$ indefinite $\Rightarrow$ saddle point
- The condition $\nabla f(x^*) = 0_n$ is necessary but not sufficient unless $f(x)$ is convex, in which case every stationary point is globally optimal.

Step 2: Lagrange Multipliers (Equality Constraints)

- Consider an optimization problem with equality constraints:

$$\min_{x \in \mathbb{R}^n} f(x) \quad \text{s.t.} \quad g(x) = 0_m, \quad g : \mathbb{R}^n \to \mathbb{R}^m.$$

- Lagrange first-order condition: at optimum x^*, the gradient of f must lie within the span of constraint gradients g_i:

$$\nabla f(x^*) = -\sum_{i=1}^{m} \lambda_i \nabla g_i(x^*) \quad \text{or compactly} \quad \nabla f(x^*) = -\nabla g(x^*)^\top \lambda,$$

  where

$$\nabla f(x^*) \in \mathbb{R}^{n \times 1}, \quad \nabla g(x^*) \in \mathbb{R}^{m \times n}, \quad \lambda \in \mathbb{R}^m.$$

- Define the Lagrangian function $\mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$:

$$\mathcal{L}(x, \lambda) = f(x) + \lambda^\top g(x)$$

- The first-order (stationarity) conditions become:

$$\nabla_x \mathcal{L}(x^*, \lambda^*) = 0_n, \qquad g(x^*) = 0_m.$$

Geometric intuition of the Lagrange condition

- Equality-constrained problem:

$$\min f(x) \text{ s.t. } g(x) = 0_m, \quad f : \mathbb{R}^n \to \mathbb{R},\; g : \mathbb{R}^n \to \mathbb{R}^m.$$

- Feasible set:

$$\mathcal{C} = \{x \in \mathbb{R}^n \mid g(x) = 0_m\},$$

  an $(n-m)$-dimensional surface in $\mathbb{R}^n$.

- Tangent space at x^*: all feasible directions $d \in \mathbb{R}^n$ that keep us on the constraint surface:

$$\nabla g(x^*)\, d = 0_m, \quad \nabla g(x^*) \in \mathbb{R}^{m \times n}.$$

- At optimum, movement in any feasible direction cannot reduce f:

$$\nabla f(x^*)^\top d = 0 \quad \forall d \text{ s.t. } \nabla g(x^*) d = 0_m, \quad \nabla f(x^*) \in \mathbb{R}^{n \times 1}.$$

- Therefore, $\nabla f(x^*)$ must be orthogonal to all feasible directions, implying it lies in the row space of $\nabla g(x^*)$:

$$\nabla f(x^*) = -\nabla g(x^*)^\top \lambda, \quad \lambda \in \mathbb{R}^m.$$

Visualization: Lagrange Condition in Two Dimensions

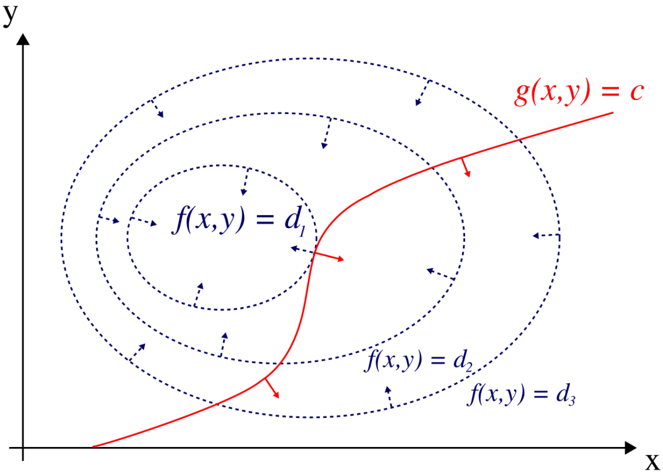

Figure 1: Lagrange multiplier illustration (public domain). Source: Wikipedia.

Step 3: Karush–Kuhn–Tucker Conditions (Inequality Constraints)

- Extend the constrained optimization problem to allow for inequality constraints:

$$\min_{x \in \mathbb{R}^n} f(x) \quad \text{s.t.} \quad \begin{cases} g_i(x) = 0, & i = 1, \ldots, m, \\ h_j(x) \leqslant 0, & j = 1, \ldots, p. \end{cases}$$

- Introduce multipliers:

$$\mathcal{L}(x, \lambda, \mu) = f(x) + \sum_{i=1}^{m} \lambda_i g_i(x) + \sum_{j=1}^{p} \mu_j h_j(x)$$

  where λ_i are unrestricted (for equality) and $\mu_j \geqslant 0$ (for inequality).

- First-order conditions (KKT system):

$$\begin{cases} \nabla_x f(x^*) + \sum_i \lambda_i^* \nabla g_i(x^*) + \sum_j \mu_j^* \nabla h_j(x^*) = 0 & \text{(stationarity)} \\ g_i(x^*) = 0, \quad h_j(x^*) \leqslant 0 & \text{(primal feasibility)} \\ \mu_j^* \geqslant 0 & \text{(dual feasibility)} \\ \mu_j^* \, h_j(x^*) = 0 & \text{(complementary slackness)} \end{cases}$$

Step 4: Dynamic Optimization as a KKT Problem

- In a competitive economy, firms' optimality implies the equilibrium return on capital:

$$f_k(k_t, z_t) + 1 - \delta = 1 + r_t.$$

  This links the production side to the household's saving decision.

- The representative household chooses $\{c_t, k_{t+1}\}$ to maximize lifetime utility:

$$\max_{\{c_t, k_{t+1}\}_{t \geqslant 0}} \sum_{t=0}^{\infty} \beta^t u(c_t) \quad \text{s.t.} \quad c_t + k_{t+1} = (1 + r_t) k_t, \quad k_{t+1} \geqslant \underline{k}.$$

- Problem elements:

$$\text{State: } k_t, z_t; \qquad \text{Controls: } c_t, k_{t+1}; \qquad u : \mathbb{R}_+ \to \mathbb{R}.$$

- Lagrangian:

$$\mathcal{L} = \sum_{t \geqslant 0} \beta^t \big[u(c_t) + \lambda_t ((1 + r_t) k_t - c_t - k_{t+1}) + \mu_t (k_{t+1} - \underline{k}) \big],$$

  where λ_t is the multiplier on the budget constraint and $\mu_t \geqslant 0$ on the borrowing constraint.

Euler Equation

$$
\begin{cases}
\text{(i) Stationarity in } c_t: & \frac{\partial \mathcal{L}}{\partial c_t} = \beta^t[u_c(c_t) - \lambda_t] = 0 \;\Rightarrow\; u_c(c_t) = \lambda_t, \\
\text{(ii) Stationarity in } k_{t+1}: & \frac{\partial \mathcal{L}}{\partial k_{t+1}} = \beta^t(-\lambda_t + \mu_t) + \beta^{t+1}\mathbb{E}_t[\lambda_{t+1}(1 + r_{t+1})] = 0 \\
 & \Rightarrow\; -\lambda_t + \beta\mathbb{E}_t[\lambda_{t+1}(1 + r_{t+1})] + \mu_t = 0, \\
\text{(iii) Primal feasibility:} & k_{t+1} \geqslant \underline{k}, \quad \text{(iv) Dual feasibility: } \mu_t \geqslant 0, \\
\text{(v) Complementary slackness:} & \mu_t(k_{t+1} - \underline{k}) = 0.
\end{cases}
$$

- Eliminating λ_t and μ_t gives either the Euler equation or the liquidity constraint binds in equilibrium

$$
\begin{aligned}
u_c(c_t) &= \beta\mathbb{E}_t[u_c(c_{t+1})(1 + r_{t+1})] \quad \text{if } k_{t+1} > \underline{k}, \\
u_c(c_t) &\geqslant \beta\mathbb{E}_t[u_c(c_{t+1})(1 + r_{t+1})] \quad \text{if } k_{t+1} = \underline{k}.
\end{aligned}
$$

Transversality Condition

- The Euler or liquidity condition ensures only local (per-period) optimality. Even if both hold, the household may still accumulate assets without bound:

$$u_c(c_t) = \beta\,\mathbb{E}_t[u_c(c_{t+1})(1 + r_{t+1})], \qquad k_t \to \infty,\ c_t \to 0.$$

  This can occur when $\beta(1 + r) = 1$, meaning the agent is so patient that saving forever still satisfies the Euler equation.

- Such a path is not globally optimal: lifetime utility can be raised by consuming more at finite dates since wealth grows indefinitely but is never used.

- The transversality condition (TVC) rules this out:

$$\lim_{T\to\infty} \beta^T\,\mathbb{E}_0[u_c(c_T)k_{T+1}] = 0,$$

  ensuring the discounted shadow value of assets vanishes.

- If the TVC fails, the present value of future wealth remains positive, so the agent could slightly reduce future saving to raise current consumption— contradicting optimality.

- In practice, we check $\beta(1 + r) < 1$: this ensures $\beta^T(1 + r)^T \to 0$, making the discounted value of wealth decay. Economically, the agent discounts the future faster than assets grow, preventing infinite accumulation and ensuring global optimality.

Recursive Bellman equation (consumption–saving model)

- The recursive problem:

$$V(k,z) = \max_{k' \geqslant \underline{k}} \Big\{ u\big((1+r_t)k - k'\big) + \beta\, \mathbb{E}[V(k',z') \mid z] \Big\}.$$

- Policy function: $\pi(k,z) = k'(k,z)$. Consumption follows from feasibility:

$$c(k,z) = (1+r_t)k - \pi(k,z).$$

- The Bellman fixed point:

$$V = TV, \qquad (TV)(k,z) = \max_{k' \geqslant \underline{k}} \Big\{ u((1+r_t)k - k') + \beta\, \mathbb{E}[V(k',z')] \Big\}.$$

The goal of time iteration is to find $\pi^*(k,z)$ directly using first-order and envelope conditions.

FOC and Envelope condition $\rightarrow$ Euler equation

- The Lagrangian for a given state (k, z):

$$\mathcal{L} = u((1 + r_t)k - k') + \beta\,\mathbb{E}[V(k', z') \mid z].$$

- First-order condition (w.r.t. k'):

$$\frac{\partial \mathcal{L}}{\partial k'} = -u_c(c(k, z)) + \beta\,\mathbb{E}[V_k(k', z') \mid z] = 0.$$

- Envelope condition (w.r.t. k):

$$V_k(k, z) = u_c(c(k, z))(1 + r_t).$$

- Substitute the envelope (for V_k) into the shifted FOC (at $t + 1$):

$$u_c(c_t) = \beta\,\mathbb{E}_t\Big[u_c(c_{t+1})(1 + r_{t+1})\Big].$$

This is the Euler equation, the basis of the time iteration (Coleman) method.

Envelope theorem: intuition and application

- The Envelope theorem simplifies how the value function changes with respect to a state variable when the decision rule is already optimal.

$$V(k,z) = \max_{k'} \mathcal{L}(k,k',z) = \max_{k'} \left[u((1+r_t)k - k') + \beta\, \mathbb{E}[V(k',z') \mid z] \right].$$

- Differentiating the maximized value function:

$$\frac{\partial V(k,z)}{\partial k} = \frac{\partial \mathcal{L}(k,k',z)}{\partial k} + \frac{\partial \mathcal{L}(k,k',z)}{\partial k'} \frac{dk'^*}{dk}.$$

- At the optimum, the first-order condition implies $\frac{\partial \mathcal{L}}{\partial k'} = 0$, so the second term vanishes:

$$V_k(k,z) = \frac{\partial \mathcal{L}(k,k',z)}{\partial k}\bigg|_{k'=k'^*(k,z)}.$$

- Hence we can treat the optimal policy as locally constant when differentiating: only the <u>direct</u> effect of k matters. In the consumption–saving model:

$$V_k(k,z) = u_c(c(k,z))(1+r_t).$$

Time iteration algorithm setup

- **Objective:** find the fixed point $\pi^*(k, z)$ satisfying the Euler condition and feasibility:

$$u_c\big((1 + r_t)k - \pi(k, z)\big) = \beta\,\mathbb{E}_z\Big[u_c\big((1 + r_{t+1})\pi(k, z) - \pi(\pi(k, z), z')\big)(1 + r_{t+1})\Big].$$

- Define the Coleman operator $H[\pi]$ mapping policy π to a new policy that satisfies the Euler condition given π.
- Iterate: $\pi^{(i+1)} = H[\pi^{(i)}]$ until convergence.
- Grids: $\{k_i\}_{i=1}^N$ for capital, $\{z_j\}_{j=1}^M$ for productivity.

Time iteration pseudocode (consumption–saving model)

1. **Inputs:** parameters β, r, σ, grid $\{k_i, z_j\}$, utility $u(c) = \frac{c^{1-\sigma}}{1-\sigma}$, tolerance ε.
2. **Initialize:** feasible policy $\pi^{(0)}(k, z)$ (e.g. constant savings rate).
3. **Iterate:**
   - For each (k_i, z_j):

$$\begin{aligned}
c &= (1+r)k_i - \pi^{(i)}(k_i, z_j), \\
\text{RHS}(k_i, z_j) &= \beta\, \mathbb{E}_{z'}\Big[u_c\big((1+r)\pi^{(i)}(k_i, z_j) - \pi^{(i)}(\pi^{(i)}(k_i, z_j), z')\big)(1+r)\Big], \\
c^{\text{new}} &= u_c^{-1}\big(\text{RHS}(k_i, z_j)\big), \\
\pi^{(i+1)}(k_i, z_j) &= (1+r)k_i - c^{\text{new}}.
\end{aligned}$$

   Enforce feasibility: $\pi^{(i+1)}(k_i, z_j) \in [\underline{k}, (1+r)k_i]$.
4. **Convergence:** if $\max_{i,j} |\pi^{(i+1)}(k_i, z_j) - \pi^{(i)}(k_i, z_j)| < \varepsilon$, stop.

Time iteration vs. VFI and PFI

- Methods recap:
  - VFI: updates $V^{(n+1)} = TV^{(n)}$ by full maximization over k'. Simple but slow; searches over action space each iteration.
  - PFI: evaluates V^{π} exactly for fixed π, then improves π. Fewer outer loops but costly evaluations.
  - Time iteration (TI): iterates directly in policy space via the Euler equation; replaces global maximization by local root-finds.
- When to use
  - TI – smooth, concave, continuous controls (e.g. consumption–saving).
  - VFI/PFI – discrete, non-smooth, or kinked problems.
- Key insight: TI exploits first-order and envelope structure, achieving much faster convergence when those conditions are valid.