Quantitative Macroeconomics with AI and ML
Lecture 2: Introduction to Quant Macroeconomists

Zhigang Feng

January 2026

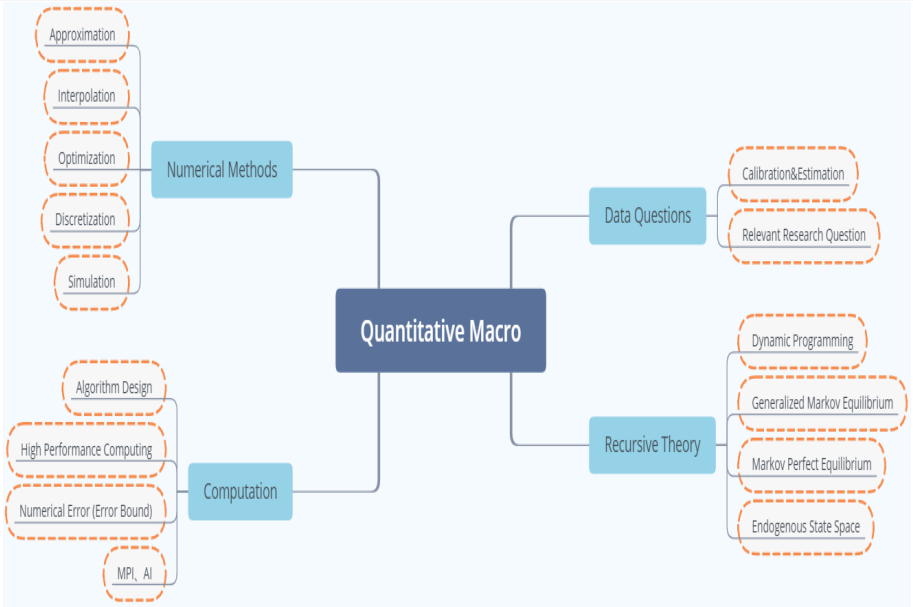
Lecture Roadmap

1. **What is Quantitative Macroeconomics?**
2. **The Optimal Growth Model**
   - Recursive formulation and the Bellman equation
   - Theoretical foundations: why computation works
3. **Two Constructive Approaches**
   - Dynamic programming (value function iteration)
   - Euler equation iteration (Greenwood-Huffman)
4. **Heterogeneous Agent Models**
   - Aiyagari-Huggett, Krusell-Smith
   - The challenge: infinite-dimensional state space
5. **From Classical Methods to AI/ML**
   - Why modern problems require new tools
   - Course preview

What is Quantitative Macroeconomics?

- A field that uses numerical methods and computational tools to solve and analyze macroeconomic models, providing precise, quantitative answers to economic questions.
- **The Methodological Triad:**
  1. *Theory:* Economic models with microfoundations
  2. *Data:* Calibration and estimation
  3. *Computation:* Numerical solution methods
- **Core Techniques:**
  - Dynamic programming and optimal control
  - Numerical optimization and root-finding
  - Monte Carlo simulation
- **This Course:** Adding machine learning to the toolkit—neural networks, stochastic optimization, and deep reinforcement learning.

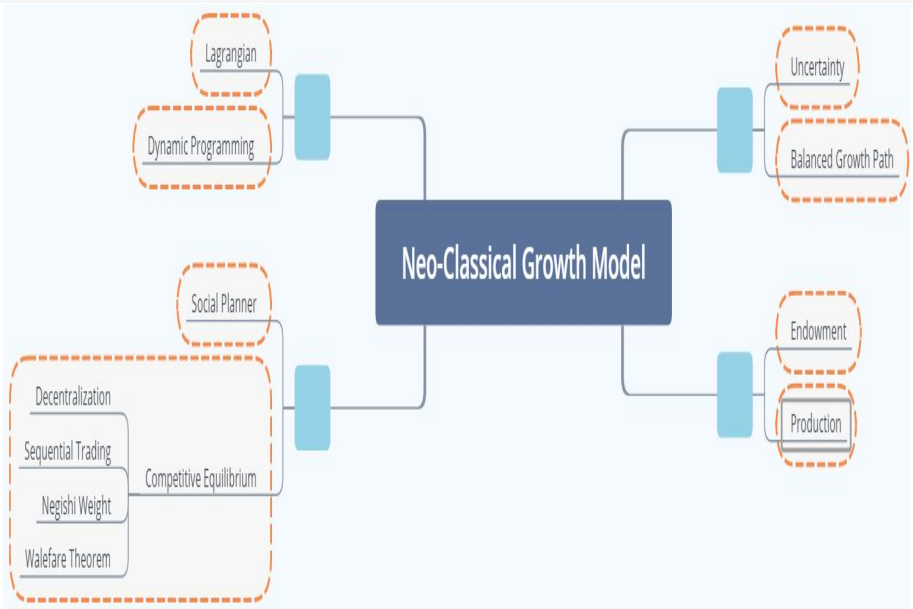
What is Quantitative Macroeconomics?



Workhorse Models in Quantitative Macro

- **Representative Agent Models:**
  - Neoclassical (stochastic) optimal growth model
  - Real Business Cycle (RBC) model
  - New Keynesian DSGE model
- **Heterogeneous Agent Models:**
  - Bewley-Aiyagari-Huggett (incomplete markets, idiosyncratic risk)
  - Krusell-Smith (aggregate uncertainty + heterogeneity)
  - Overlapping generations (OLG) models
- **Optimal Policy Models:**
  - Ramsey taxation problems
  - Time-inconsistency and commitment
- **Common Thread:** All require solving dynamic optimization problems—the Bellman equation is central.

Optimal Growth Model at the Core



Optimal Growth Model: Setup

- The household solves the following dynamic optimization problem:

$$\max_{\{c_t, k_{t+1}\}_{t=0}^{\infty}} \sum_{t=0}^{\infty} \beta^t \log c_t$$

$$\text{s.t. } c_t + k_{t+1} = Ak_t^{\alpha} + (1-\delta)k_t$$

$$\beta \in (0,1), \quad A > 0, \quad \alpha \in (0,1), \quad \delta \in [0,1]$$

$$k_t, c_t \geq 0, \quad k_0 \text{ given}, \quad t = 0, 1, 2, \ldots$$

- **Recursive Formulation (Bellman Equation):**

$$v(k) = \max_{k' \in \Gamma(k)} \{\log(Ak^{\alpha} + (1-\delta)k - k') + \beta v(k')\}$$

  where $\Gamma(k) := \{k' \mid 0 \leq k' \leq Ak^{\alpha} + (1-\delta)k\}$

- **Solution:** Value function $v(k)$ and policy function $k' = g(k)$.

From Sequence Problem to Recursive Problem

- **The Challenge:** The sequence problem requires choosing an infinite sequence $\{c_t, k_{t+1}\}_{t=0}^{\infty}$ simultaneously—computationally intractable.
- **Key Insight:** The problem has *recursive structure*—optimal decisions depend only on the current state k, not on history or calendar time.
- **Bellman's Principle of Optimality (1957):**
  "An optimal policy has the property that whatever the initial state and initial decision are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decision."
- **Implication:** Replace infinite-dimensional optimization with a functional equation—the Bellman equation—reducing the problem to finding a fixed point in function space.

The Bellman Operator

- Define the **Bellman operator** $T : \mathcal{C}(X) \to \mathcal{C}(X)$:

$$(Tv)(k) = \max_{k' \in \Gamma(k)} \{u\left(Ak^{\alpha} + (1-\delta)k - k'\right) + \beta v(k')\}$$

- The value function v^* is a **fixed point**: $Tv^* = v^*$
- **Central Questions:**
  1. Does a fixed point exist?
  2. Is it unique?
  3. Can we compute it via iteration $v_{n+1} = Tv_n$?
  4. What properties does v^* inherit (continuity, concavity)?
- The following theoretical results answer these questions affirmatively.

Contraction Mapping Theorem

- **Definition:** Let (S, d) be a metric space. An operator $T : S \to S$ is a *contraction* with modulus $\beta \in (0, 1)$ if:

$$d(Tv, Tw) \leq \beta \cdot d(v, w) \quad \forall\, v, w \in S$$

- **Contraction Mapping Theorem (Banach, 1922):** If (S, d) is a complete metric space and T is a contraction, then:
  1. T has a *unique* fixed point $v^* \in S$
  2. For any $v_0 \in S$, the sequence $v_{n+1} = Tv_n$ converges to v^*
  3. Rate of convergence: $d(v_n, v^*) \leq \frac{\beta^n}{1-\beta} d(v_1, v_0)$

- **For Economics:** Guarantees that Value Function Iteration (VFI) converges to the unique solution of the Bellman equation.

Blackwell's Sufficient Conditions

- **Problem:** How do we verify that T is a contraction?
- **Blackwell (1965):** An operator $T : \mathcal{B}(X) \to \mathcal{B}(X)$ on bounded functions is a contraction if:
  1. **Monotonicity:** $v(x) \leq w(x)$ for all $x \Rightarrow (Tv)(x) \leq (Tw)(x)$
  2. **Discounting:** $\exists\, \beta \in (0,1)$ such that $T(v+a)(x) \leq (Tv)(x) + \beta a$ for all $a \geq 0$
- **Verification for the Bellman Operator:**
  - Monotonicity: Higher continuation value $\Rightarrow$ higher current value
  - Discounting: The factor β in $\beta v(k')$ provides the discounting
- **Key Insight:** The discount factor $\beta < 1$ is not just economically meaningful—it is mathematically essential for convergence.

Theorem of the Maximum

- **Theorem (Berge, 1963):** Let $f(x, a)$ be continuous and $\Gamma(x)$ be a continuous, compact-valued correspondence. Then:
  1. The *value function* $v^*(x) = \max_{a \in \Gamma(x)} f(x, a)$ is continuous
  2. The *policy correspondence* $G(x) = \arg\max_{a \in \Gamma(x)} f(x, a)$ is upper hemi-continuous

- **Economic Implications:**
  - Small changes in k lead to small changes in $v(k)$ and $g(k)$
  - Ensures stability of computed solutions
  - Justifies numerical approximation on discrete grids

- **Additional Properties:** Under strict concavity of $u(\cdot)$:
  - Policy correspondence is single-valued: a *function* $g(k)$
  - Value function $v(k)$ is strictly concave

Theoretical Foundation: Summary

Result	What It Guarantees
Principle of Optimality	Recursive formulation is valid
Contraction Mapping Thm	Existence, uniqueness, convergence
Blackwell's Conditions	Easy verification of contraction
Theorem of the Maximum	Continuity and stability

Together, these results establish:

1. The Bellman equation has a *unique* solution v^*
2. We can *compute* v^* by iterating $v_{n+1} = Tv_n$
3. The solution is *well-behaved* (continuous, concave)
4. The policy function $g(k)$ is *stable* under perturbations

$\Rightarrow$ **Solid foundation for computational methods**

Two Roads to Equilibrium

- We have two constructive algorithms for computing equilibrium:

- **Approach 1: Dynamic Programming (Value Function Iteration)**
  - Iterate on the Bellman equation: $v_{n+1} = Tv_n$
  - Convergence via *contraction mapping*
  - Recovers both value function and policy

- **Approach 2: Euler Equation Iteration (Greenwood-Huffman)**
  - Iterate directly on the policy function via Euler equation
  - Convergence via *monotonicity*
  - Works directly with equilibrium conditions

- Both approaches are *constructive*—they provide algorithms, not just existence proofs.

Dynamic Programming: The Classical Framework

- **Dynamic Programming (Bellman, 1957):** Solve sequential decision problems by exploiting recursive structure.
- **Core Idea:** Value of any state = max over (immediate reward + discounted continuation):

$$v(s) = \max_{a \in \Gamma(s)} \{r(s,a) + \beta \, \mathbb{E}\left[v(s') \mid s, a\right]\}$$

- **Value Function Iteration (VFI):**
  1. Discretize state space into finite grid $\{s_1, \ldots, s_N\}$
  2. Initialize $v_0(s_i)$ for each grid point
  3. Iterate: $v_{n+1}(s_i) = \max_a \{r(s_i,a) + \beta \sum_j P(s_j|s_i,a) v_n(s_j)\}$
  4. Stop when $\|v_{n+1} - v_n\| < \varepsilon$
- **Requirement:** Complete knowledge of the environment—$P(s'|s,a)$ and $r(s,a)$.

Euler Equation Approach: Setup

- **Alternative:** Work directly with first-order conditions (Euler equations).
- The Euler equation for the optimal growth model:

$$u'(c_t) = \beta \, \mathbb{E}_t \left[u'(c_{t+1}) \cdot F_k(k_{t+1}, z_{t+1}) \right]$$

- In a competitive equilibrium with $k = K$:

$$u_c(c(k, K, k', z)) = \beta \, \mathbb{E} \left[u_c(c(k', K', k'', z')) \cdot F_k(k', K', z') \mid z \right]$$

- **Equilibrium Requirement:** Find policy g and aggregate law of motion G such that:
  - The Euler equation holds for all (k, K, z)
  - Consistency: $G(K, z) = g(K, K, z)$ (individual = aggregate)

The Greenwood-Huffman Operator

- **Idea:** Construct a sequence of policy functions that converges to equilibrium.
- Let $H^0(K, z) = 0$. Define $H^{j+1}(K, z)$ as the x solving:

$$u_c(c(K, K, x, z)) = \beta \, \mathbb{E}\left[u_c(c(x, x, H^j(x, z'), z')) \cdot F_k(x, x, z')\right]$$

- **Interpretation:**
  - Given a conjecture H^j for future policy, solve for today's optimal x
  - The solution $x = H^{j+1}(K, z)$ becomes the new conjecture
  - Iterate until convergence: $H^{j+1} \approx H^j$
- **Key Question:** Does this sequence converge? To an equilibrium?

Existence of Unique Solution at Each Step

Consider the equation at each iteration:

$$u_c(F(K,K,z)-x) = \beta\,\mathbb{E}\left[u_c(F(x,x,z')-H^j(x,z'))\cdot F_k(x,x,z')\right]$$

- **Left-hand side:** Increasing in x (since $u_{cc}<0$)
- **Right-hand side:** Decreasing in x under conditions on:

$$u_{cc}\left(F_k+F_K-\frac{\partial H^j}{\partial x}\right)F_k+u_c(F_{kk}+F_{kK})$$

- **Greenwood-Huffman (1995):** Impose regularity conditions ensuring RHS is decreasing.
- **Result:** Unique intersection $\Rightarrow$ unique solution $x=H^{j+1}(K,z)$ exists.

Convergence via Monotonicity

- **Two approaches to fixed points:**
  - Dynamic programming: exploits **contraction** properties
  - Euler equation iteration: exploits **monotonicity** properties
- **Monotonicity of the Operator:**
  - If $H^j(K,z) > G^j(K,z)$ pointwise, then $TH^j > TG^j$
  - The sequence $\{H^j\}$ is monotone (increasing from $H^0 = 0$)
- **Boundedness:** Economic constraints ensure $H^j(K,z) \leq F(K,K,z)$
- **Convergence:** Monotone + bounded $\Rightarrow$ limit H^* exists
- **Equicontinuity:** Bounds on $\partial H^j/\partial K$ ensure H^* is continuous

Does the Limit Characterize Equilibrium?

- **By Construction:**
  - The limit H^* satisfies the Euler equation (by continuity)
  - Given H^*, individual optimization yields policy $k' = q(k, K, z)$
- **Consistency Check:**
  - When $k = K$: individual policy $q(K, K, z) = H^*(K, z)$
  - The Euler equation is satisfied at the aggregate level
- **Conclusion:** H^* is the equilibrium law of motion for aggregate capital.
- **Advantage of This Approach:**
  - Works directly with equilibrium conditions
  - Extends naturally to models where value functions are harder to characterize
  - Provides constructive proof of equilibrium existence

Comparing the Two Approaches

	Value Function Iteration	**Euler Equation Iteration**
Object	Value function $v(k)$	Policy function $H(K, z)$
Iteration	$v_{n+1} = Tv_n$	$H^{j+1} = TH^j$
Convergence	Contraction	Monotonicity
Theory	Banach fixed point	Monotone convergence
Strength	General, robust	Direct equilibrium
Weakness	Indirect (need to extract g)	Requires monotonicity

Both provide:

- Constructive algorithms (not just existence proofs)
- Theoretical guarantees for convergence
- Foundation for numerical implementation

Why Classical Macro Models Are "Easy"

- **Representative Agent + First Welfare Theorem:**
  - Competitive equilibrium $\Leftrightarrow$ Social planner's solution
  - No need to explicitly compute prices to find allocations

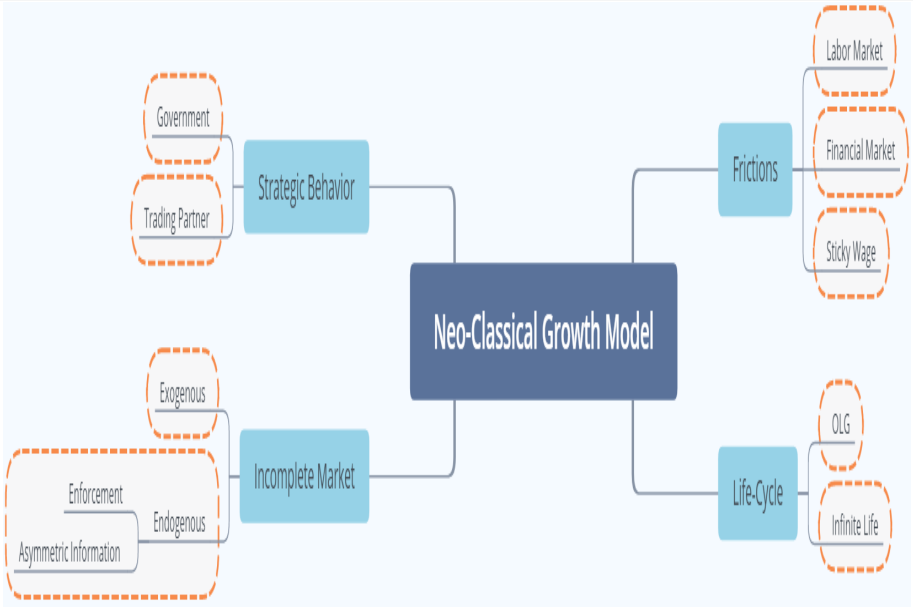
- **The Key Simplification:** Individual capital k = Aggregate capital K
  - Prices are deterministic functions of a single state variable:

$$r(K) = \alpha A K^{\alpha-1} - \delta, \qquad w(K) = (1-\alpha) A K^{\alpha}$$

  - The agent *knows* how prices evolve because $K' = g(K)$

- **By Design, Not by Accident:**
  - Economists constructed this framework for tractability
  - The "environment" is fully specified by $(A, \alpha, \beta, \delta)$
  - State space is low-dimensional (often just k or (k, z))

Extensions based on the Optimal Growth Model



Heterogeneous Agents: Breaking the Simplicity

- **Realistic Extension:** Agents differ in wealth, income, productivity
  - Individual state: (a_i, z_i) — assets and idiosyncratic shock
  - Now $k_i \neq K$; individual choices don't reveal aggregates

- **Prices Depend on the Distribution:**

$$r = r(\mu), \qquad w = w(\mu)$$

  where μ is the *cross-sectional distribution* of (a, z)

- **The Distribution Is a State Variable:**
  - To forecast prices, agents must forecast how μ evolves
  - But μ is *infinite-dimensional*—a measure over (a, z)
  - Evolution: $\mu' = \Phi(\mu; g)$ depends on everyone's policy

- **The Environment Is No Longer Obvious:**
  - Agents face a *perceived law of motion* they must approximate
  - First Welfare Theorem fails $\Rightarrow$ must solve for prices explicitly

Bewley-Aiyagari-Huggett Model

- **Setup:**
  - Continuum of infinitely lived households (measure 1)
  - Uninsurable idiosyncratic income shocks ϵ
  - *Ex-ante* identical, *ex-post* heterogeneous
- **Bellman Equation:**

$$v(a, \epsilon; \lambda) = \max_{c, a'} \left\{ u(c) + \beta \sum_{\epsilon'} v(a', \epsilon'; \lambda') \pi(\epsilon, \epsilon') \right\}$$

subject to:

$$c + a' = (1 + r(\lambda))a + w(\lambda)\epsilon$$
$$a' \geq -\underline{a}$$

- **Stationary Equilibrium:** Find λ^* such that $\lambda^* = \Phi(\lambda^*; g)$
  - Distribution is invariant under optimal policies
  - Incomplete markets endogenously generate wealth inequality

Krusell-Smith Model

- **Extension:** Add *aggregate uncertainty* to Aiyagari
- **Aggregate shock:** $z_t \in \{z_g, z_b\}$ (good/bad states)
  - Affects aggregate productivity
  - Unemployment higher in bad states: $u_b > u_g$
- **Household Problem:**

$$v(k, \epsilon; \Gamma, z) = \max_{c,k'} \{U(c) + \beta \, \mathbb{E}[v(k', \epsilon'; \Gamma', z') \mid z, \epsilon]\}$$

subject to:

$$c + k' = r(\bar{k}, \bar{l})k + w(\bar{k}, \bar{l})\epsilon + (1 - \delta)k$$
$$\Gamma' = H(\Gamma, z, z'), \qquad k' \geq 0$$

- **The Challenge:** Agents must forecast evolution of Γ—infinite-dimensional!

Krusell-Smith: The Computational Innovation

- **Key Insight (Krusell-Smith, 1998):**
  - Agents don't need to track the entire distribution Γ
  - Approximate with low-dimensional moments: $\bar{K} = \int k \, d\Gamma$
- **Bounded Rationality:** Agents use simple forecasting rules

$$\log \bar{K}' = a_0(z) + a_1(z) \log \bar{K}$$

- **Algorithm:**
  1. Guess coefficients (a_0, a_1)
  2. Solve individual problem given forecasting rule
  3. Simulate economy, regress $\log \bar{K}'$ on $\log \bar{K}$
  4. Update coefficients; iterate until convergence
- **Remarkable Finding:** $R^2 > 0.999$—mean capital is nearly sufficient!

Theoretical Issues in Heterogeneous Agent Models

- **Existence of Recursive Equilibrium:**
  - When markets are complete/efficient: easier to prove
  - Incomplete markets: may require forward-looking variables
- **Uniqueness:** Generally *not* guaranteed
  - Multiple steady states possible
  - Policy functions may have discontinuities
- **Continuity:** Value and policy functions may not be continuous in aggregates
- **Computation:**
  - Curse of dimensionality (distribution as state)
  - Ensuring accuracy of approximations
  - Verifying equilibrium conditions
- **These issues become more severe in optimal policy (Ramsey) problems.**

Example: Non-Existence of Continuous Equilibrium

Santos (2002): Representative household, $\sum \beta^t \log(c_t)$, subject to:

$$c_t + k_{t+1} \leq \pi_t + (1 - \tau_t) r_t k_t + T_t$$

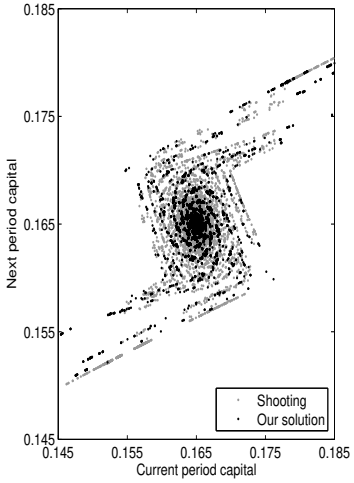
with piecewise linear tax on capital:

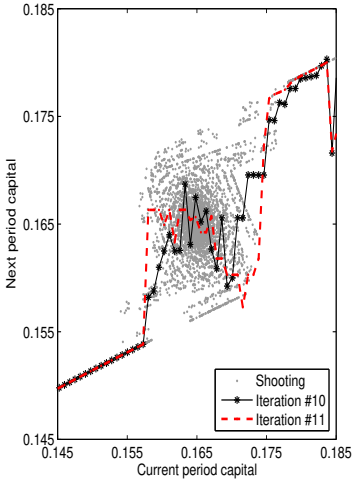
$$\tau(K) = \begin{cases} 0.10 & \text{if } K \leq 0.160 \\ 0.05 - 10(K - 0.165) & \text{if } 0.160 < K < 0.170 \\ 0 & \text{if } K \geq 0.170 \end{cases}$$

Result: A continuous Markov equilibrium *fails to exist*.

- Middle steady state: unstable (complex eigenvalues)
- Other steady states: saddle-path stable
- Equilibrium exhibits cycles or discontinuities

Theoretical Issues: Equilibrium Cycles





Ramsey Problem: Additional State Variables

- **Consumer FOCs:**
  - Consumption: $\lambda_t = u_{c_t}$
  - Labor: $u_n = (1 - \tau_t) u_c f_n$
  - Euler: $\lambda_t = \lambda_{t+1} \beta [(1 - \theta_{t+1}) f_k + 1 - \delta]$

- **The Problem:** Euler equation can be rewritten backward:

$$\lambda_{t-1} = \lambda_t \beta [(1 - \theta_t) f_k(k_t, n_t) + 1 - \delta]$$

- **Implication:** Must track λ_{t-1} as a state variable
  - Minimal state space (k_t) is *insufficient*
  - Need (k_t, λ_{t-1}) with unknown domain for λ

- **Consequence:** State space expands; recursive formulation becomes more complex.

The Curse of Dimensionality

- **Bellman (1961):** Named the "curse of dimensionality"—exponential growth of computational cost with state dimension.
- **Grid-Based Methods:**
  - n grid points per dimension, d dimensions $\Rightarrow n^d$ total points
  - Example: 100 points $\times$ 10 dimensions $= 10^{20}$ points
  - Infeasible for storage, let alone computation
- **Heterogeneous Agent Models:**
  - Distribution μ is infinite-dimensional
  - Even discretized: thousands of (a, z) pairs $\times$ aggregate states
  - Standard VFI becomes computationally prohibitive
- **This is exactly the challenge AI/RL research confronts**—in even more extreme settings.

AI/Reinforcement Learning: The Robot-Taxi Problem

- **Example:** Training an autonomous vehicle to navigate a city
- **State Space:** Camera images, LiDAR, GPS, speed, ...
  - Single camera frame: $1920 \times 1080 \times 3 \approx 6$ million dimensions
  - Continuous, high-dimensional, partially observable
- **Action Space:** Steering, acceleration, braking (continuous)
- **Environment:** Transition dynamics $P(s'|s, a)$
  - Physics, other drivers, pedestrians, weather
  - *Unknown* and *impossible to fully specify*
  - No "First Welfare Theorem" to simplify
- **Contrast with Macro:** No closed-form model; must learn from experience

RL Formulation: The Same Bellman Equation

- **Markov Decision Process (MDP):** (S, A, P, r, β)
- **Bellman Equation for Optimal Value:**

$$v^*(s) = \max_a \left\{ r(s,a) + \beta \int v^*(s') P(ds'|s,a) \right\}$$

- **Bellman Equation for Q-Function:**

$$Q^*(s,a) = r(s,a) + \beta \int \max_{a'} Q^*(s',a') P(ds'|s,a)$$

- **Key Insight:** Theoretical foundation is *identical* to economics.
  - Bellman equation, contraction mapping, fixed point—all apply
- **The difference is computational:** How do we solve this when P is unknown and S is enormous?

Three Fundamental Challenges

Challenge	Classical DP	Modern AI/RL
Dimensionality	Low ($d \leq 5$)	Very high ($d \geq 10^6$)
Environment	Fully known	Unknown, learned
Data	Regular grids	Random samples

- **Dimensionality:** Cannot discretize; must use function approximation
- **Unknown Environment:** Cannot evaluate $\mathbb{E}[v(s')]$ analytically; estimate from samples (Monte Carlo)
- **Irregular Data:** Samples from trajectories, not grids; must generalize

Monte Carlo: Turning Samples into Expectations

- **The Problem:** Computing $\mathbb{E}[v(s')|s,a] = \int v(s')P(ds'|s,a)$
  - Classical DP: Enumerate all s', weight by known $P(s'|s,a)$
  - High dimensions or unknown P: Infeasible

- **Monte Carlo Estimation:**

$$\mathbb{E}[v(s')] \approx \frac{1}{N}\sum_{i=1}^{N} v(s'_i), \qquad s'_i \sim P(\cdot|s,a)$$

  - Generate samples by *simulating* or *interacting* with environment
  - Law of Large Numbers: Converges as $N \to \infty$

- **Key Advantage:** Cost scales with N, *not* dimension
  - Works equally well in 2 or 2 million dimensions
  - This is how RL escapes the curse of dimensionality

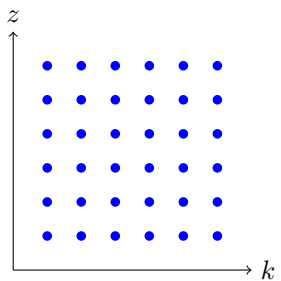
Function Approximation: Learning from Irregular Data

- **Classical DP:** Value function as table $\{v(s_i)\}_{i=1}^{N}$
  - Regular grid, pre-specified, covers entire state space
  - Interpolation straightforward
- **RL with Function Approximation:**

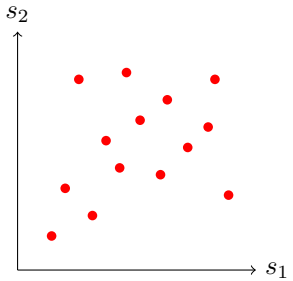
$$v(s) \approx v_\theta(s) = \text{NeuralNetwork}_\theta(s)$$

  - Parameters θ learned from sampled data
  - Data from trajectories: $(s_0, a_0, r_0, s_1, a_1, r_1, \ldots)$
  - Points concentrate where policy visits (not uniform)
- **Challenges:**
  - Must generalize to unseen states
  - Approximation errors can compound (instability)
  - No convergence guarantee (unlike tabular DP)

From Grids to Samples: A Paradigm Shift



Classical DP
Regular Grid



Modern RL
Sampled Trajectories

- **Left:** Compute $v(s)$ at every grid point; interpolate
- **Right:** Observe (s, a, r, s') tuples; fit $v_\theta(s)$ to minimize prediction error

Heterogeneous Agent Models: Between Two Worlds

- **Individual Problem:** Low-dimensional (a, z)—classical methods work
- **Aggregate State:** Distribution μ—infinite-dimensional
- **Traditional Approaches:**
  - Krusell-Smith: Approximate μ by moments $(\bar{K}, \sigma_K, \ldots)$
  - Bounded rationality: Simple forecasting rules
  - Iterate until forecasts approximately consistent
- **Modern Approaches (Research Frontier):**
  - Neural networks to represent policy: $g_\theta(a, z, \mu)$
  - Monte Carlo simulation of agent distribution
  - Learn equilibrium conditions end-to-end
- **Key Insight:** Heterogeneous agent macro is a middle ground—we *know* economic structure but face RL-like computational challenges.

Overview: Two Traditions, Shared Foundations

	Quantitative Macro	AI / Reinforcement Learning
Theory	Bellman equation	Bellman equation
Foundation	Contraction mapping	Contraction mapping
Environment	Known (model-based)	Unknown (learn from data)
State Space	Low-dimensional	Very high-dimensional
Computation	Grid + interpolation	Samples + neural nets
Strength	Exploits structure	Scales to complexity
Weakness	Curse of dimensionality	Sample inefficiency

The Opportunity: Combine economic structure with ML scalability

- Use theory to reduce dimensionality where possible
- Use neural networks where analytical solutions fail
- Use Monte Carlo to handle high-dimensional distributions

Numerical Techniques: Classical Methods

- **Function Approximation and Interpolation**
  - Linear/cubic splines, Chebyshev polynomials
  - Approximating policy and value functions
- **Numerical Optimization**
  - Newton-Raphson, gradient descent, grid search
  - Finite differences, automatic differentiation (PyTorch)
- **Stochastic Process Approximation**
  - Tauchen, Rouwenhorst methods for discretizing AR(1)
- **Perturbation and Projection**
  - Linearization around steady state
  - Higher-order perturbation (2nd/3rd order for risk)
  - Projection methods with basis functions

Numerical Techniques: The AI Frontier

- **Deep Neural Networks**
  - Universal approximators for high-dimensional functions
  - Policy networks $\pi_\theta(s)$, value networks $v_\theta(s)$
- **Monte Carlo and Simulation**
  - Estimating expectations without discretization
  - Variance reduction techniques
- **Stochastic Optimization**
  - SGD, Adam, and variants
  - Training on mini-batches of sampled data
- **Reinforcement Learning**
  - Q-learning, policy gradient, actor-critic
  - Solving DP when environment is unknown or complex
- **Adaptive Methods:** Sparse grids, active learning for efficient sampling

What This Course Will Explore

- **Classical Foundations:**
  - Dynamic programming theory (existence, uniqueness, convergence)
  - Numerical methods: VFI, PFI, Euler iteration, projection, perturbation
- **Machine Learning Tools:**
  - Function approximation with neural networks
  - Monte Carlo methods and variance reduction
  - Stochastic optimization (SGD, Adam)
- **Bridging the Gap:**
  - Deep learning for solving Bellman equations
  - Neural network solutions to heterogeneous agent models
  - Exploiting economic structure in ML architectures
- **Goal:** Equip you to work at the frontier where economic theory meets computational innovation.

Programming Environment

- **Python**
  - High-level, readable, vast ecosystem (NumPy, SciPy, Pandas)
  - Excellent for prototyping; vectorized libraries for speed
- **PyTorch**
  - GPU-accelerated tensor computing
  - Dynamic computational graphs with automatic differentiation
  - Essential for modern high-dimensional solving
- **Installation:**
  - Anaconda: https://www.anaconda.com/products/individual
  - PyCharm IDE: https://www.jetbrains.com/pycharm/download/
  - Google Colab: Zero-setup with free GPU access

Summary: The Big Picture

1. **Theory:** Bellman equation + contraction mapping = foundation for both economics and AI
2. **Classical Macro:** Exploits known structure; limited by dimensionality
3. **Heterogeneous Agents:** Distribution as state breaks classical methods
4. **Modern AI/RL:** Scales via Monte Carlo + neural networks; sample-intensive
5. **The Opportunity:** Combine economic structure with ML scalability

This course: Learn to work at the intersection.