Computational Macroeconomics: The Toolkit

From Function Approximation to Dynamic Programming

The Imperative

Modern dynamic models rarely have analytical paper-and-pencil solutions. Complexities such as non-linearities, uncertainty, and high dimensionality require robust numerical techniques for analysis.

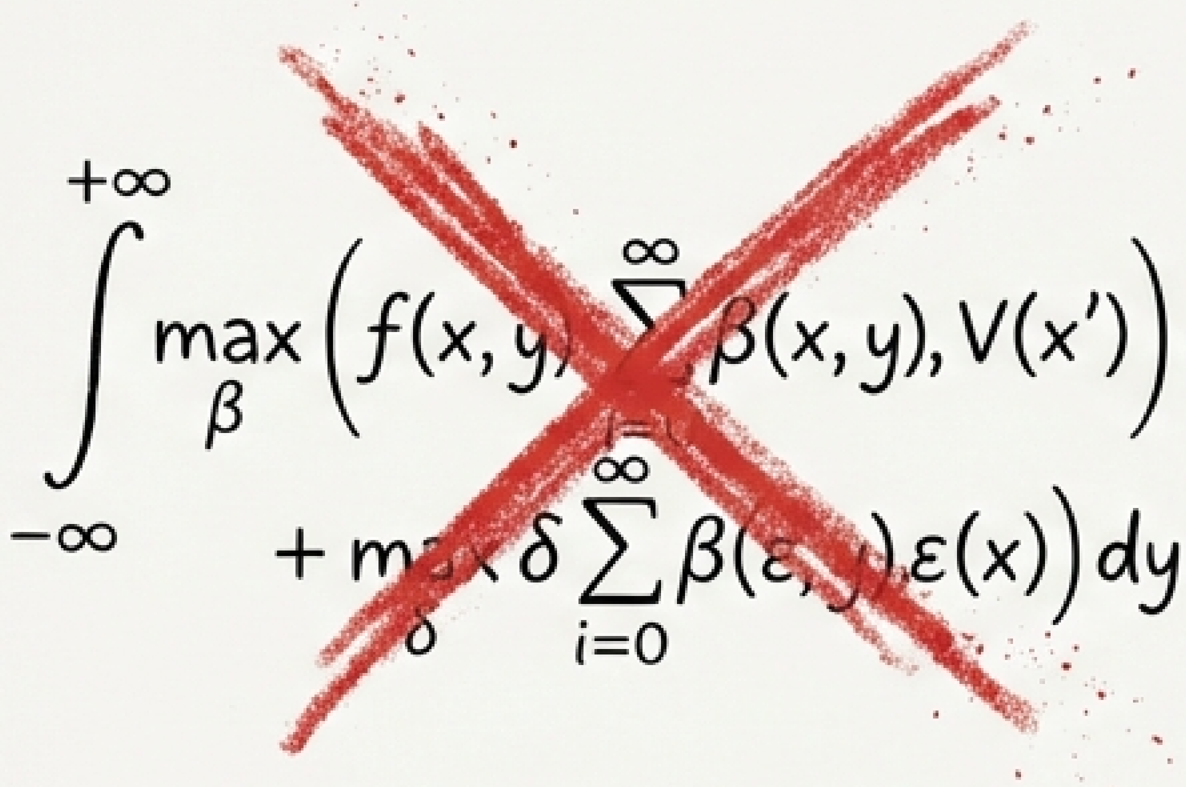
The Method

Translate economic theory into executable code using numerical algorithms. This involves techniques for function approximation, numerical integration, and root-finding to solve equilibrium conditions.

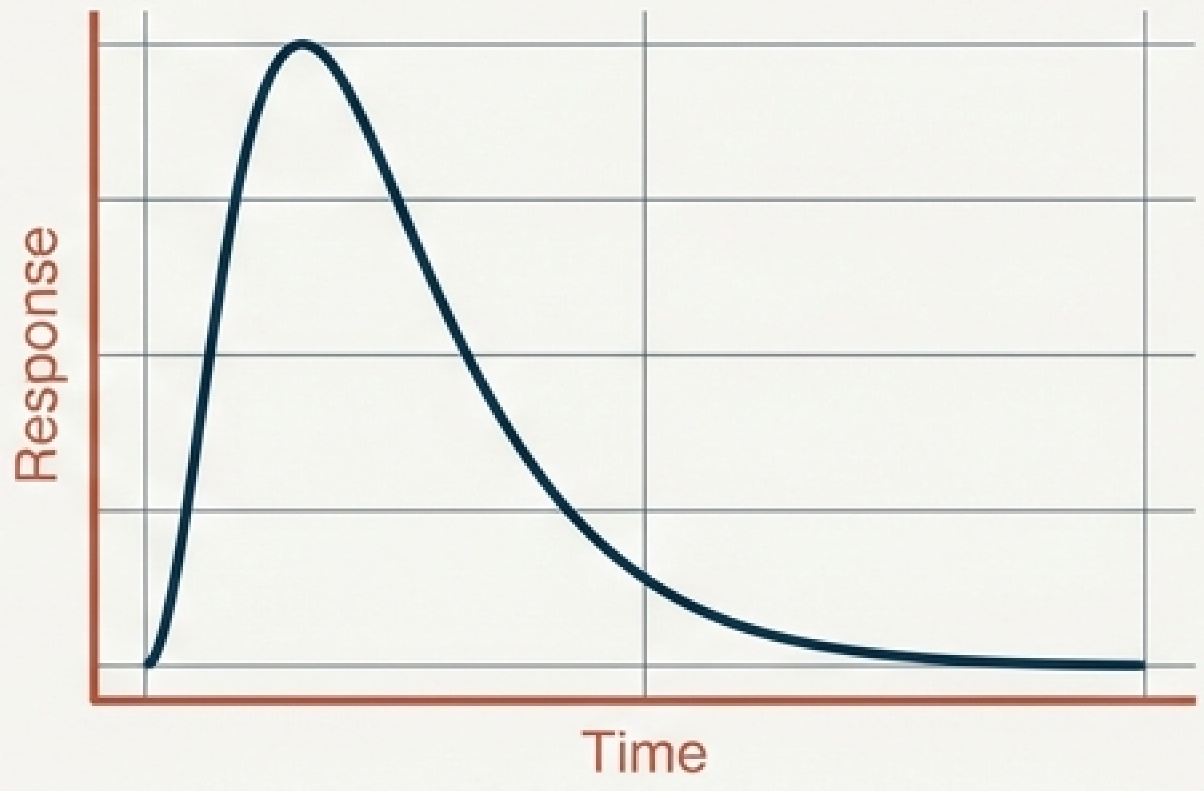
The Goal

Deconstruct the atomic 'building blocks' required to simulate complex economies. This includes solving bellman equations, policy function iteration, and simulating time series for model validation.

The Computational Imperative



Intractable Analytical Solutions



Numerical Approximations

"Our task as I see it...is to write a FORTRAN program that will accept specific economic policy rules as 'input' and will generate as 'output' statistics describing the operating characteristics of time series we care about."

— Robert E. Lucas Jr. (1980)

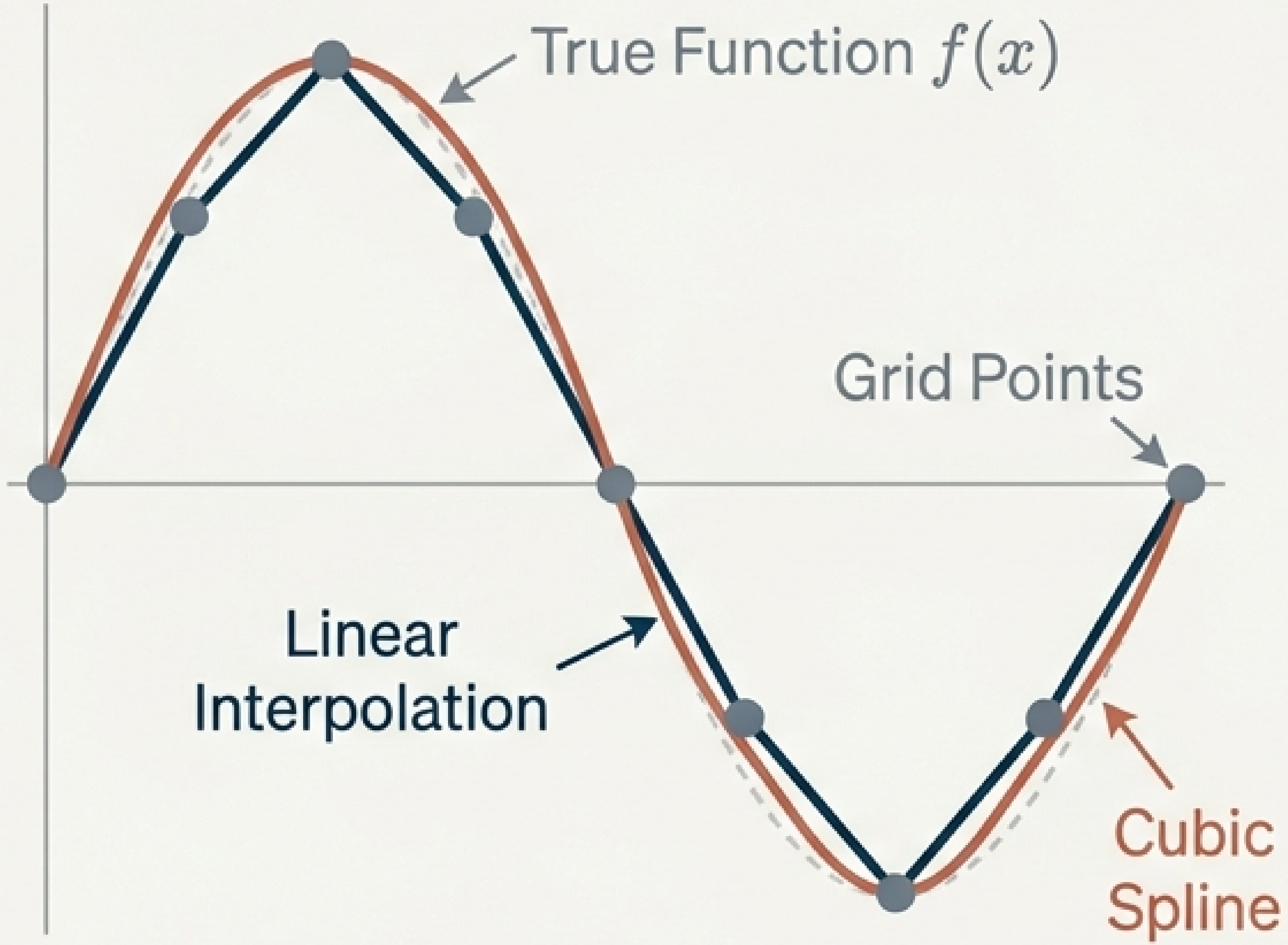
Tool 1: Approximating Functions

How do we store infinite points in finite memory?

Interpolation (Grid Points)

- **Linear:** Connects dots. Robust, local information only. Jagged (non-differentiable).

- **Cubic Spline:** Ensures continuous 1st and 2nd derivatives. Smoother, but errors propagate globally.



Known Functional Forms

- Weighted sum of polynomials (e.g., Chebyshev).

- Uses regression logic to minimize distance between true function and approximation.

$$\hat{f}(x) = \sum a_i \phi_i(x)$$

Tool 2: Root Finding (The Robust Approach)

Bisection: The Tortoise

The Problem:

Find scalar x where $f(x) = 0$.

The Algorithm:

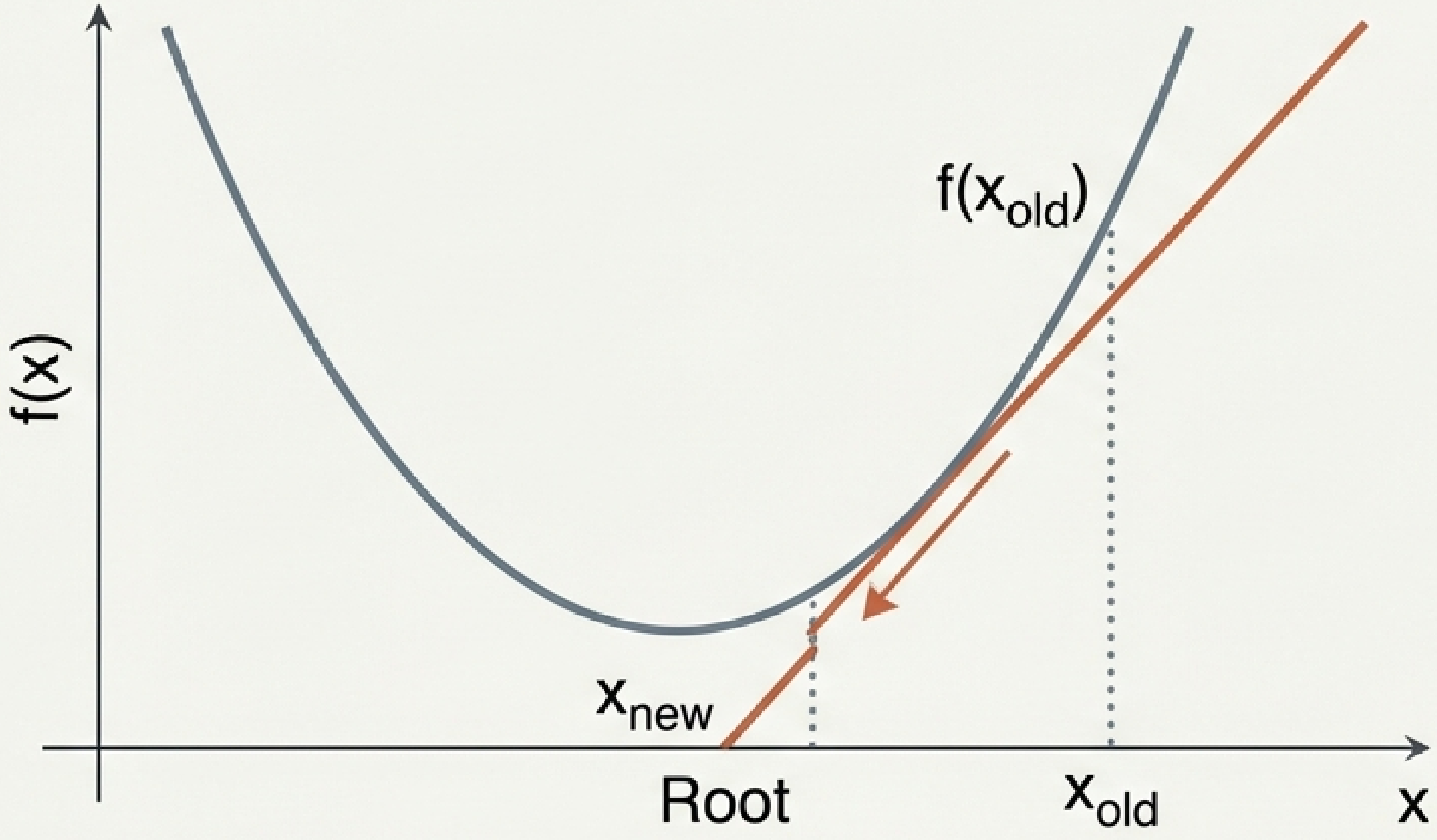
1. Bracket the root between a and b (where signs differ).
2. Evaluate midpoint c.
3. Replace bound to keep root bracketed.
4. Repeat until `|b − a| < Tolerance`.

Strategic Profile:

Pros: Guaranteed convergence.
No derivatives needed.
Cons: Slow (Linear convergence rate).

Tool 2: Root Finding (The Fast Approach)

Newton-Raphson: The Hare



$$x_{new} = x_{old} - \frac{f(x_{old})}{f'(x_{old})}$$

The Method:

- Approximates function locally using a tangent line (derivative).
- Jumps to where the tangent crosses zero.

Strategic Profile:

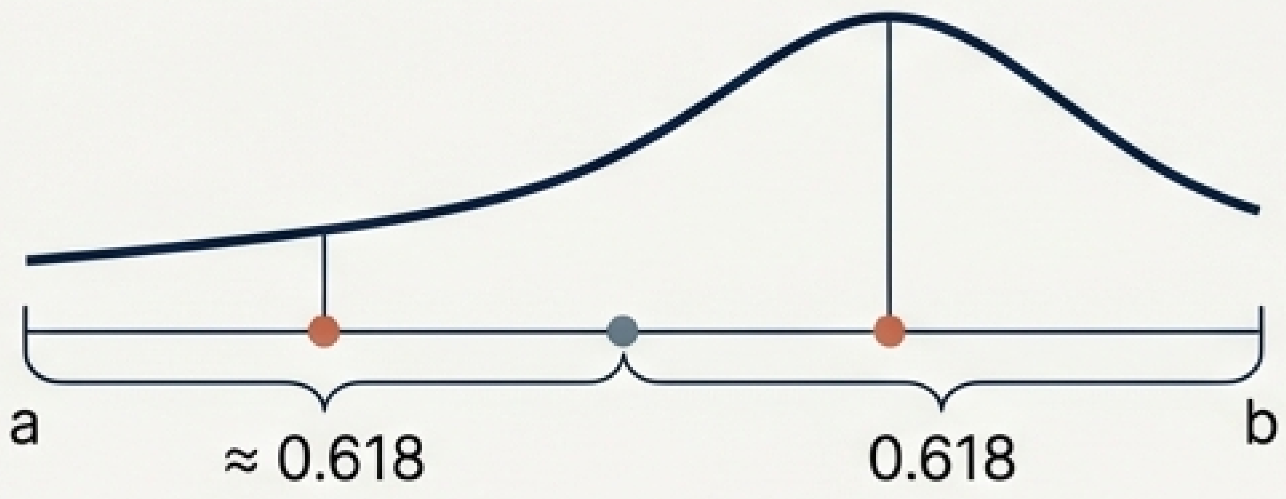
- Pros: Extremely fast (Quadratic convergence).
- Cons: Requires differentiability. Can diverge if function is ill-behaved.

Note: Derivatives often calculated numerically (Finite Difference).

Tool 3: Optimization (max F(x))

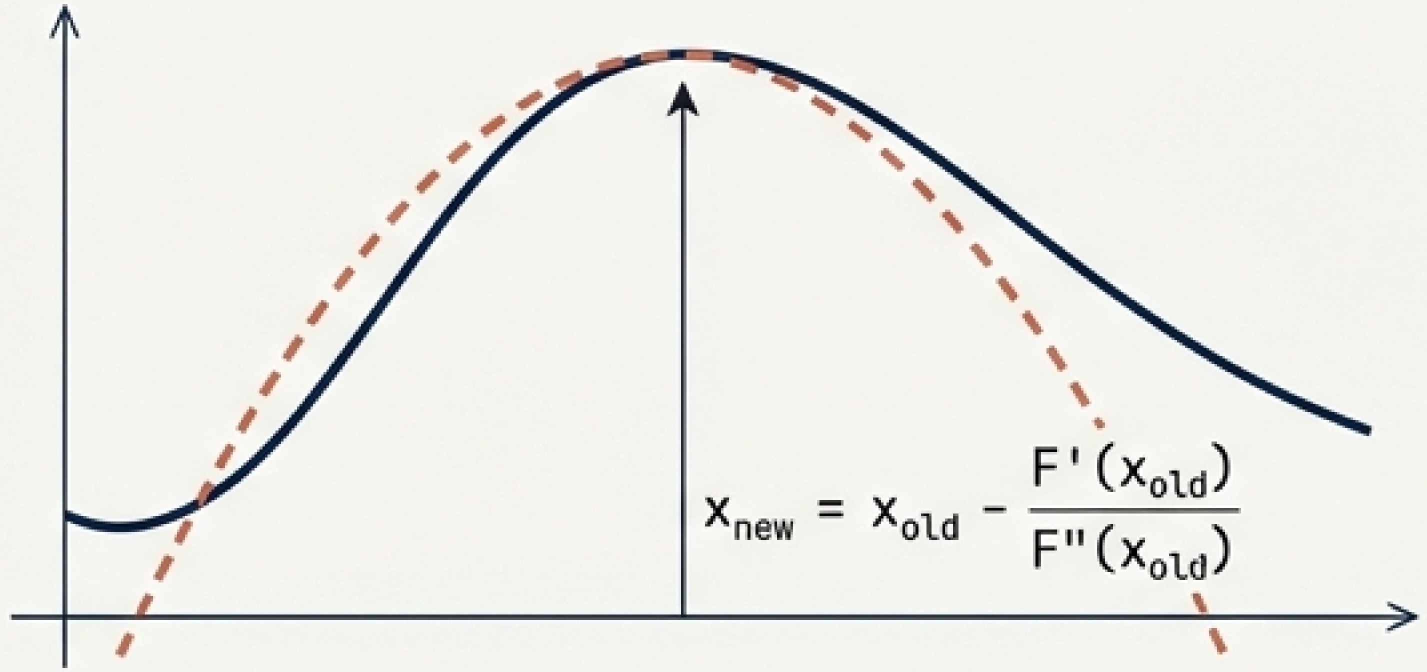
Maximizing F(x) is computationally equivalent to finding the root of the First Order Condition: $F'(x) = 0$.

Golden-Section Search



Robust, no derivatives. Shrinks interval by golden ratio. Use when function is non-differentiable.

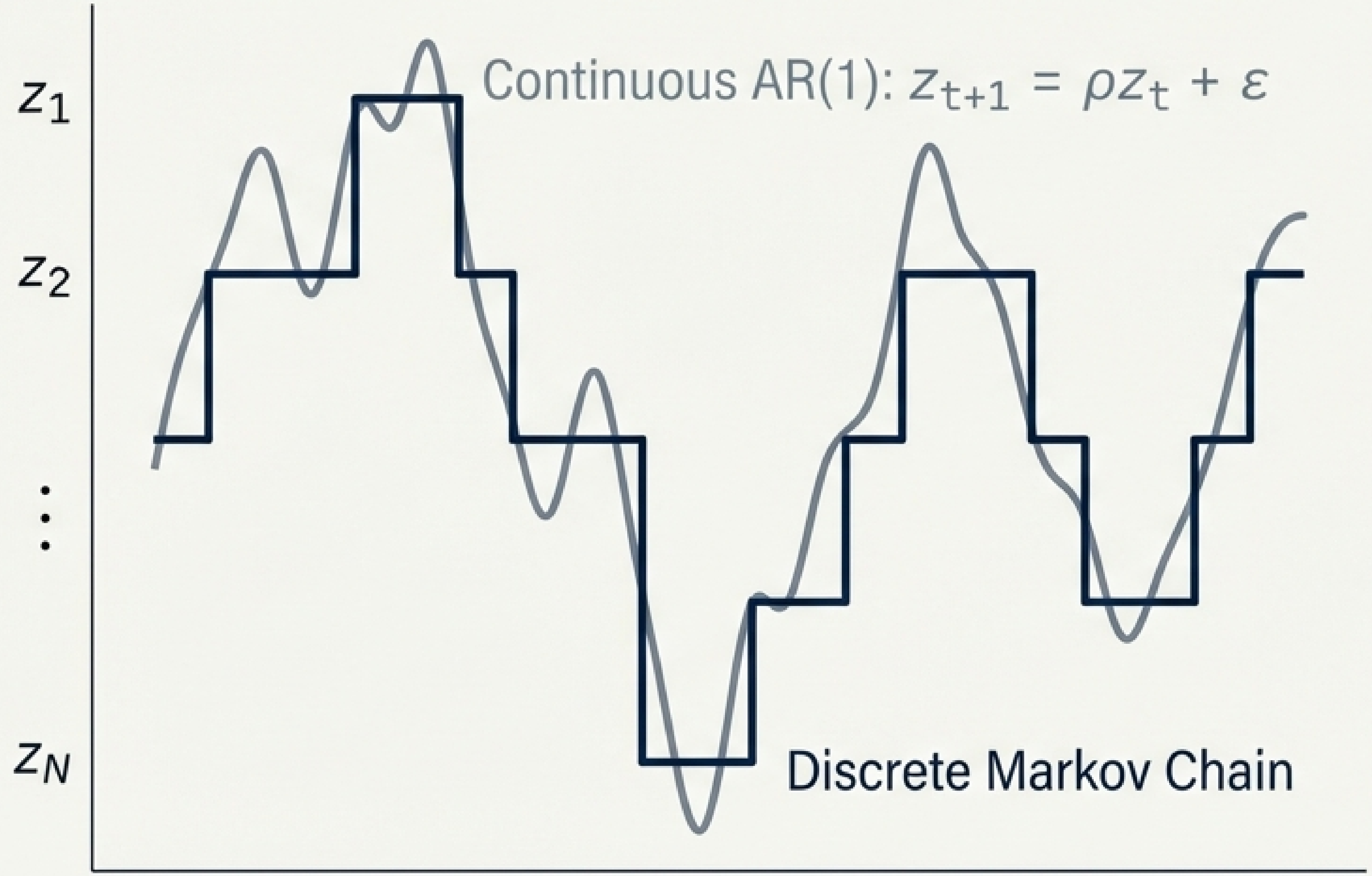
Newton's Method for Optimization



Uses quadratic approximation. Fast but requires 1st and 2nd derivatives. Use when speed is critical.

Tool 4: Discretizing Stochastic Processes

Turning continuous shocks into Markov matrices



The Problem: Computers solve Dynamic Programming via discrete matrices, but economic shocks are continuous.

Method A: Tauchen

- Uses integrals over intervals to calculate transition probabilities.
- Best for general purpose approximation.

Method B: Rouwenhorst

- Matches unconditional mean and variance exactly.
- Best for highly persistent processes (ρ close to 1), common in macro.

Solving The Model: Global Methods (VFI)

Optimization Tool

$$V(k) = \max \{ F(k, k') + \beta V(k') \}$$

Value Function (Recursive)

Interpolation Tool

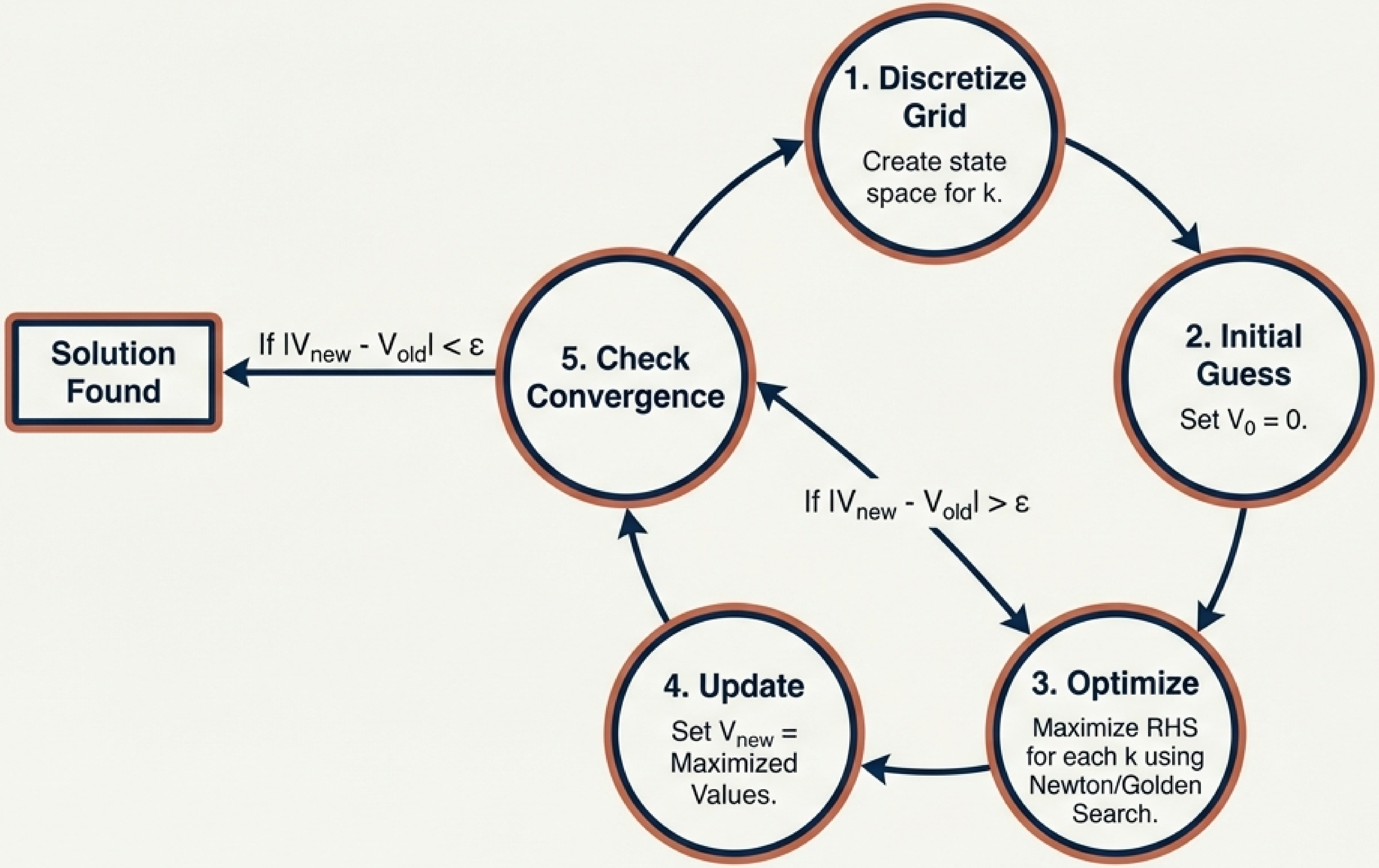
The Goal: **Solve for the Value Function** $V(k)$ and **Policy Function.**

The Mechanism: **Contraction Mapping Theorem**

- The operator T is a contraction mapping.
- **Guarantee:** Iterating on the Bellman equation converges to the true solution ($V_i \to V$) from *any* initial guess.

Modern Academic Precision

The Algorithm: Value Function Iteration (VFI)



Side Note

This process relies on the "Black Box" tools defined earlier:

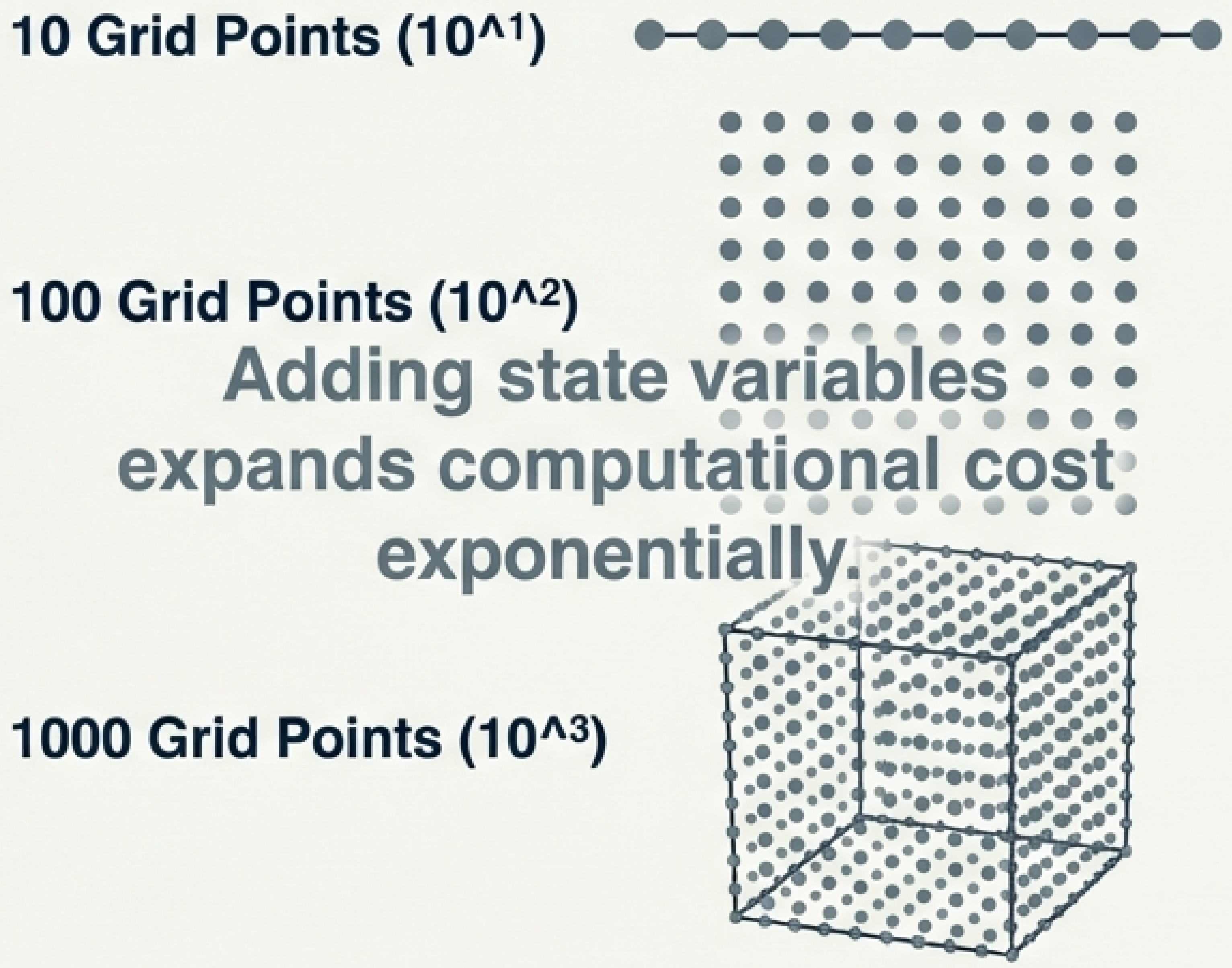
Approximation for off-grid values and Root Finding/Optimization for the max operator.

VFI in Practice: The Curse of Dimensionality

Case Study: The Stochastic Cake Eating Problem

The Problem:

- Agent maximizes utility of consumption over infinite time.
- Stochastic extension: New cake arrives randomly (z).
- Value Function becomes V(a, z).



Strategic Choice: VFI is robust (handles risk/non-linearities) but computationally expensive.

Solving The Model: Local Methods (Linearization)

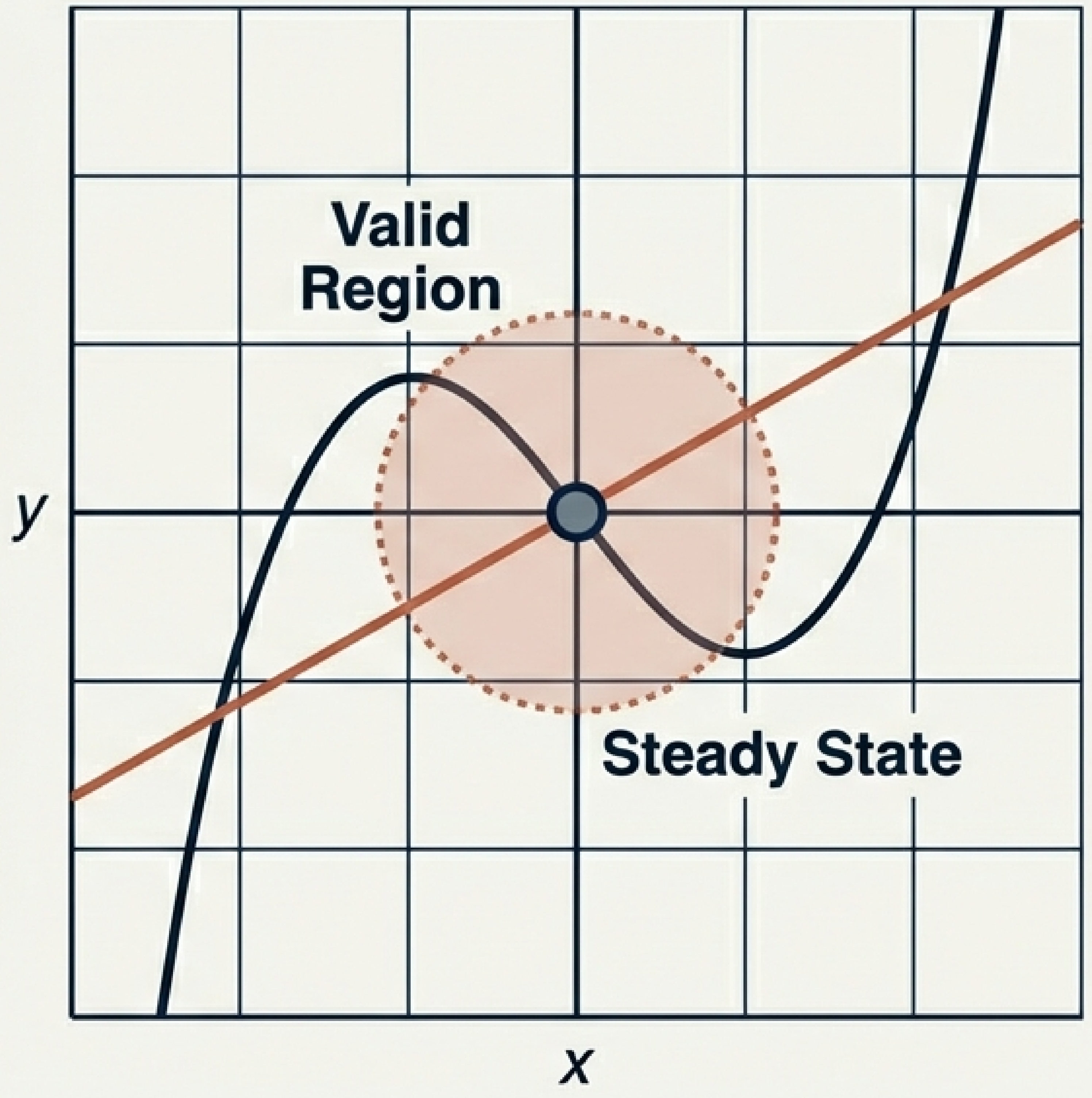
The Perturbation Approach

The Concept:

- Instead of solving the whole grid (Global), solve locally around the deterministic Steady State.
- Approximate system using Taylor expansion.

Key Assumptions:

1. Shocks are small.
2. Economy stays near Steady State.
3. Certainty Equivalence: In linear systems, risk premiums disappear.



The Linearization Workflow

1. Classification

Predetermined Variables (x_t):

- Fixed from previous period
- Capital K_t
- Shocks z_t

Jump Variables (y_t):

- Determined in current period
- Consumption C_t
- Labor H_t

2. Log-Linearization

$$\hat{x}_t = \log(X_t/\bar{X})$$

Convert Euler equations and constraints to log-deviations.

3. Matrix System

$$A \begin{bmatrix} x_{t+1} \\ E_t[y_{t+1}] \end{bmatrix} = B \begin{bmatrix} x_t \\ y_t \end{bmatrix} + \text{Shocks}$$

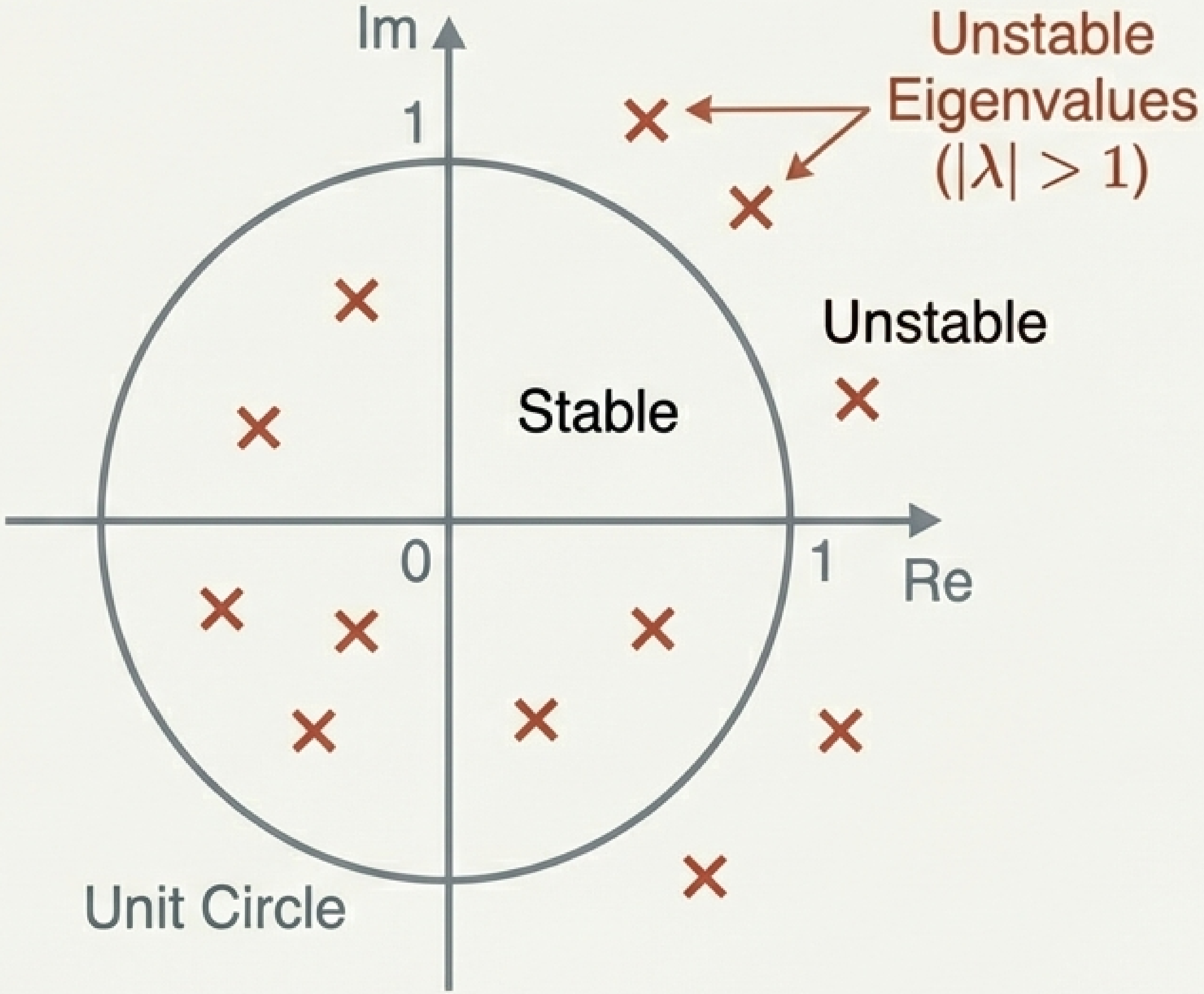
Ensuring Stability: The Blanchard-Kahn Condition

The Problem: Linear systems can explode. We need a unique path back to equilibrium.

The Condition ($h = m$):

- The number of unstable eigenvalues (h) must equal the number of jump variables (m).

- Ensures a unique initial jump for Consumption that places the economy on the stable Saddle Path.



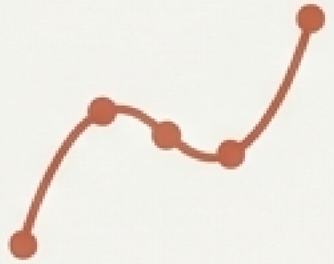
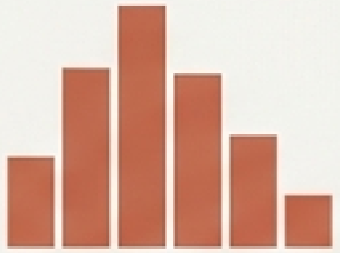
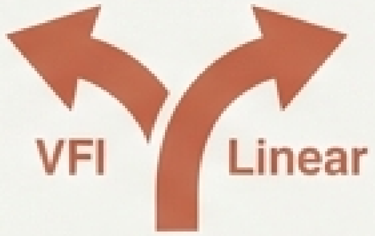
NotebookLM

The Strategic Choice: Global vs. Local

	Global Methods (VFI)	Local Methods (Linearization)
Scope	Valid everywhere in state space.	Valid only near Steady State.
Risk Handling	Accurate. Captures risk aversion & non-linearities.	Certainty Equivalence. Ignores risk premiums.
Cost & Speed	High Cost. Slow. Limited by Curse of Dimensionality.	Low Cost. Instant. Scalable to large systems.

Use VFI for asset pricing/crises. Use Linearization for standard business cycles.

The Modern Macroeconomist’s Toolkit

Approximation 	 Solvers
Dynamics 	 Solution Logic

Summary: We have deconstructed the “Black Box”.
- We store functions via Chebyshev/Splines.
- We solve equations via Newton/Bisection.
- We handle shocks via Tauchen/Rouwenhorst.
- We choose solution strategies based on the trade-off between robustness and speed.

Implementation: While software (Dynare) automates this, understanding the blocks is essential for diagnosis.

References: Chapter 10, Lucas (1980), Judd (1998), Numerical Recipes, QuantEcon.