Functional equations

Judd Chapters 10,11

David Childers (thanks to Y. Kryukov, K. Judd, and U. Doraszelski)

CMU, Tepper School of Business

April 5, 2023

The Plan: building methods for dynamic problems

- In dynamic problems, the unknown is often a function
  - Dynamic Programming: Value and policy as function of state
  - Optimal Control: State and policy as function of time
- Differential equations & finite difference method
  - ODE: $y'(x) = f(y(x), x)$
  - Finite difference method: build a spline approximation to y recursively
  - Econ. problems are optimization; one way to derive an ODE is Optimal Control approach (Hamiltonian conditions)
- Projection methods
  - Global function approximation
  - Coefficients approximately solve any conditions on the function, e.g. the Bellman equation of Dynamic programming
- Optimal Control will be covered in a later class

Motivation

- Optimal growth – social planner solves:

$$\begin{aligned} \max_c \quad & \int_0^\infty e^{-\rho t} u\{c(t)\} dt \\ \text{subject to:} \quad & \dot{k} \equiv k'(t) = f\{k(t)\} - c(t) \\ & k(0) = \bar{k}_0 \end{aligned}$$

  - $u(c) = \frac{c^{\gamma+1}}{\gamma+1}$ is the utility function
  - $f(k) = Ak^\alpha$ is the production function.
- Opt.Control: derive $c'(t) \equiv \dot{c}$ from optimality conditions, so $C'(k) = \dot{c}/\dot{k}$ is an ODE
  - Today's ODE: assume $c(t) = (1-s) f(k(t))$, so $k'(t) = sf\{k(t)\}$
- Dynamic progamming: solve Bellman equation for for $C(k)$ and $V(k)$.

Ordinary Differential Equation (ODE)

- First-order ODE:

$$\frac{dy}{dx} = f(y, x),$$

where $f : R^{n+1} \to \mathbb{R}^n$ is the known function and $y : [a, b] \subset \mathbb{R} \to \mathbb{R}^n$ is the unknown solution.

- A kth order ODE can be transformed into a system of k first-order ODEs.

$$\frac{d^2y}{dx^2} = f(\frac{dy}{dx}, y, x) \Leftrightarrow \left\{ \begin{array}{c} \frac{dy}{dx} = z \\ \frac{dz}{dx} = f(z, y, x) \end{array} \right\}$$

- Initial value problem (IVP) – conditions on the solution at a single point:

$$y(x_0) = y_0,$$

- Boundary value problem (BVP) – at several points:

$$g_i(y(x_i)) = 0, \qquad i = 1, \ldots, n$$

Linear ODE-IVP

- If the problem is linear in y' and y:

$$\begin{aligned} y'(x) + a(x)\,y &= b(x), \\ y(0) &= y_0 \end{aligned}$$

- There is a closed-form solution:

$$\begin{aligned} y(x) &= y_0 \exp\left\{-\int_0^x a(s)\,ds\right\} + \\ &\quad + \int_0^x \exp\left\{-\int_0^s a(r)\,dr\right\} b(s)\,ds \end{aligned}$$

- Constant coefficients simplify the problem:

$$\begin{aligned} y'(x) &= Ay, y(0) = y_0 \\ &\Rightarrow y(x) = y_0 \exp\{Ax\} \end{aligned}$$

General IVP: Euler methods

IVP: $y'(x) = f(y(x), x)$, $y(x_0) = y_0$; $y : [a, b] \to \mathbb{R}$, $x_0 = a$

- Grid $x_i = x_0 + ih$, where $i = 0, 1, \ldots, N$ and $h = \frac{b-a}{N}$.
- Goal: Find $Y_i = \widehat{y(x_i)}$

Explicit Euler method:

- Taylor approximation of y around x_i, evaluated at x_{i+1}:

$$y(x_{i+1}) \approx y(x_i) + y'(x_i)(x_{i+1} - x_i)$$

- Recall that $x_{i+1} - x_i = h$, and compute:

$$Y_{i+1} = Y_i + hf(Y_i, x_i), \quad Y_0 = y_0 \tag{EE}$$

- Straightforward computation $(i = 1, ..., N)$
- If $y \in C^3[a, b]$, $f \in C^2$, and f_y and f_{yy} are bounded, then error of Euler method shrinks in proportion to h.

Implicit Euler method

- Same approximation, but around x_{i+1} & evaluated at x_i:

$$y(x_i) \approx y(x_{i+1}) + y'(x_{i+1})(x_i - x_{i+1})$$

- Again, $x_i - x_{i+1} = -h$ and solve:

$$Y_{i+1} = Y_i + hf(Y_{i+1}, x_{i+1}) \tag{IE}$$

- (IE) is a nonlinear equation in Y_{i+1} $\Rightarrow$ slower step
  - Implement as Newton's method at each i,
  - or as a fixed-point iteration (need $|hf_y| < 1$),
  - or as a large system of nonlinear equations.
- More stable $\Rightarrow$ can use higher h $\Rightarrow$ less steps
- Both Euler methods assume $y'(x)$ is a step-function, so $y(x)$ is piecewise linear

Trapezoid method

- Combine explicit and implicit Euler:

$$Y_{i+1} = Y_i + \frac{h}{2}\left[f(Y_i, x_i) + f(Y_{i+1}, x_{i+1})\right] \tag{T}$$

- Assumes $y'(x)$ is piecewise linear,
- so $y(x)$ is piecewise quadratic.
- Convergence is quadratic: error $\sim h^2$
- Method is also implicit since (T) defines Y_{i+1} as the solution to an equation.

- Runge-Kutta scheme generalizes finite difference methods:

$$Y_{i+1} = Y_i + \frac{h}{2}\left[f(Y_i, x_i) + f(Y_i + hf(x_i, Y_i), x_{i+1})\right]$$

  - Also quadratic, but explicit

Runge-Kutta 4th order scheme

$$Y_{i+1} = Y_i + \frac{h}{6}\left[z_1 + 2z_2 + 2z_3 + z_4\right], \quad Y_0 = y_0,$$

where

$$\begin{aligned}
z_1 &= f(Y_i, x_i), \\
z_2 &= f(Y_i + \frac{h}{2} z_1, x_i + \frac{h}{2}), \\
z_3 &= f(Y_i + \frac{h}{2} z_2, x_i + \frac{h}{2}), \\
z_4 &= f(Y_i + h z_3, x_i + h).
\end{aligned}$$

- Explicit
- Converges at rate h^4.
- Each step evaluates f four times.

Example: constant growth rate

- Growth model with constant savings rate s:

$$\dot{k} \equiv k'(t) = sf\{k(t)\},$$
$$k(0) = \bar{k}_0$$

- 1st order ODE,
  - nonlinear if $f(\cdot)$ is nonlinear
  - IVP: $k(0) = \bar{k}_0$
- Grid over the argument of the function: $t_i = ih$
- Runge-Kutta: Compute $k_{i+1} = \widehat{k(t_{i+1})}$ using k_i and t_{i+1}

Boundary value problem (BVP)

General form (with two point conditions):

$$\begin{aligned} \dot{x}(t) &= f(x(t),y(t),t), \quad x(0)=\bar{x}_0, \\ \dot{y}(t) &= g(x(t),y(t),t), \quad y(T)=\bar{y}_T. \end{aligned}$$

Solution = **Shooting**

- Initialization: Guess y_0 and choose stopping criterion ϵ.
- Step 1: Solve the IVP

$$\begin{aligned} \dot{x}(t) &= f(x(t),y(t),t), \quad x(0)=\bar{x}_0, \\ \dot{y}(t) &= g(x(t),y(t),t), \quad y(0)=y_0. \end{aligned}$$

  - Let $Y(T,y^0)$ be the resulting approximation for $y(T)$.
- Step 2: If $||Y(T,y^0)-\bar{y}_T||<\epsilon$, stop; otherwise, adjust y_0 – using Bisection, Secant, etc.
- I.e. solve the nonlinear equation $Y(T;y^0)=\bar{y}_T$.

BVP example: Life-cycle consumption

- Consumer with lifespan T solves:

$$\begin{array}{rc} \max_c & \int_0^T e^{-\rho t} u(c) dt \\ \text{subject to} & \dot{A} = f(A) + w - c, \\ & A(0) = A(T) = 0 \end{array}$$

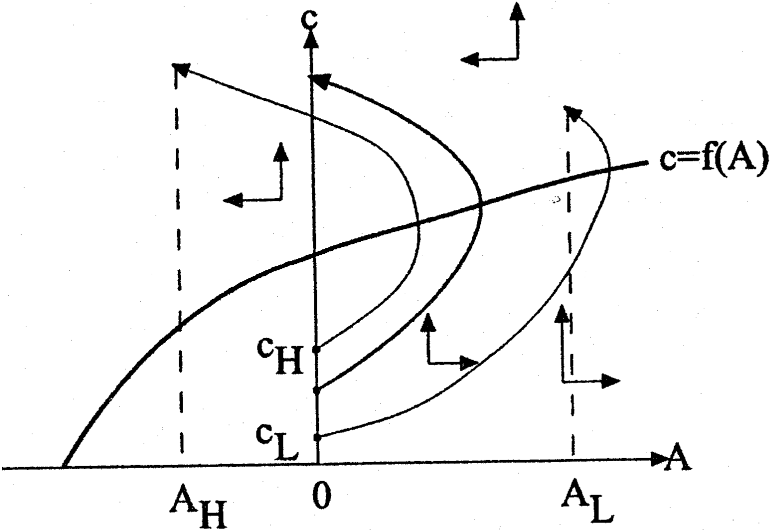
  - $A(t)$ = assets that consumer holds at time t
  - w = wage, $f(\cdot)$ = return on investment

- Optimal Control methods (Hamiltonian + Pontryagin conditions, topic 9) tell us that:

$$\begin{aligned} \dot{c} &= \frac{u'(c)}{u''(c)} \left(\rho - f'(A)\right), \\ \dot{A} &= f(A) + w - c \end{aligned}$$

- Shooting: pick $c(0)$ to make $A(T) = 0$ (Fig.10.2)

Finite-Horizon Optimal Control Problems



Shooting in a life-cycle consumption problem. Source: Judd, K. (1998), Figure 10.2.

Projection: idea

- Solving $y'(x) = f(y(x), x)$, $y(0) = y_0$, for $x \in [0, \bar{X}]$
- Finite difference methods construct $y(\cdot)$ from splines
- Projection chooses function representation to minimize global measure of error

$$\hat{y}(x) = y_0 + \sum_{j=1}^{n} a_j \phi_j(x)$$

  - $a_0 \equiv y_0$ to fit the initial condition (here, $\phi_j(0) \equiv 0$)
- Define function operator: $[\mathcal{L}y](x) = [y'(x) - f(y(x), x)](x)$
- Solution $y^*(\cdot)$ must satisfy $[\mathcal{L}y^*](x) = 0$, $\forall x \in [0, \bar{X}]$
- Idea: find coefficients a that make $\mathcal{L}\hat{y}(x)$ "small",
  - To this end, define **residual function**:

$$R(x; a) \equiv \mathcal{L}\hat{y}(x) = \sum_{j=1}^{n} a_j \phi_j'(x) - f\left(y_0 + \sum_{j=1}^{n} a_j \phi_j(x), x\right)$$

Projection: approaches

- Want to find a that makes $R\left(x;a\right)=\mathcal{L}\left[\sum_{j=1}^{n}a_j\phi_j\right]\left(x\right)$ small
- Different approaches use different function representation, different def of "small"
- *Spectral* methods solve exactly $\langle R(x,a),\psi_i(x)\rangle=0\ \forall i$
  - n unknowns (a) $\Rightarrow$ need n equations
  - Representation should approximate solution well
  - Choose ψ to approximate range space well
  - Use knowledge of function class (differentiable, etc)
- Finding a: solve system of n equations
  - If $\mathcal{L}$ and $\hat{y}$ linear, system is linear
  - Otherwise, use nonlinear solution method: e.g. fixed point, Newton
- Warning: $\frac{d}{da}\mathcal{L}$ can be very badly conditioned
  - E.g. $\mathcal{L}=\int_s^t k(x,z)\hat{y}(z)dz=0$ Fredholm eq of 1st kind
  - $\rightarrow$ use penalized method

Spectral approaches: examples

- *Galerkin*: Use $\{\psi\}$ an orthonormal basis
  - Typically: $\{\psi\} = \{\phi\}$ orthogonal polynomials
  - Can differentiate, integrate exactly in many cases
  - *Petrov-Galerkin* $\{\psi\} \neq \{\phi\}$
  - o.n.b. ensures L^2 approx of input and output
  - Exponentially fast approximation in n *if* solution known to be smooth
- *Least squares*:

$$\min_a \int_0^{\bar{X}} \left[R\left(x;a\right)\right]^2 dx$$

  - FOC's: one equation for each a_j: $\left\langle R(x,a), \frac{\partial}{da_j} R(x,a) \right\rangle = 0$
  - Special case solvable by optimization

Pseudo-spectral approaches

- Spectral methods: evaluate $R(x, a)$ and inner products
  - Need closed form for operator *and* integral
- Instead, can approximate inner product by quadrature
- If $\mathcal{L}$ itself involves integral, use quadrature here too

$$\int \int k(x,z) \sum_{j=1}^{n} a_j \phi_j(z) dz \psi_i(x) dx = \sum_{j=1}^{n} a_j \left\langle k(x,z), \phi_j \psi_i \right\rangle$$

- Choose quadrature method to fit function choice
  - Gaussian or other exact method for polynomials
- (Chebyshev) *Collocation*: pick n nodes x_i, and solve:

$$R\left(x_i; a\right) = 0, \qquad i = 1, \ldots, n$$

  - Nodes x_i = degree n Chebyshev points, mapped into $[0, \bar{X}]$
  - $\phi_j(x)$ any basis, ψ_j are *implicitly* Chebyshev polynomials
  - Uses interpolation (fast) in place of exact projection (slow)

Projection: applications

- If solving a system of ODE's:
  - Separate approximation for each univariate function
  - Separate residual f-n for each eq-n $\Rightarrow$ same number of nodes
- If solving BVP: need to account for boundary conditions
- Either add as equation $\hat{y}(x_0) = a_0$
- or choose basis that satisfies for all a: "solve out"
- Projection method works for all functional equations, not just ODE's:
  - Bellman $\Rightarrow$ eq's in terms of value and policy functions
  - Optimal control approach: ODE for policy function (with steady state as the "initial" value)
  - Euler equation: eq-n in terms of policy function (next slide)

Projection example: Euler equation

- Discrete-time growth model:

$$\max_{c,k} \sum_{t=1}^{\infty} \beta^t u(c_t)$$
$$\text{s.t.} \quad : \quad k_{t+1} = f(k_t) - c_t$$

- Substitute $c_t = f(k_t) - k_{t+1}$, take FOC w.r.t. k_{t+1}:

$$u'(c_t) = \beta u'(c_{t+1}) f'(k_{t+1})$$

- Rewrite in terms of policy function: $c_t = C(k_t)$:

$$\mathcal{L}C(k) \equiv \left[\begin{array}{l} u'[C(k)] \\ -\beta u'[C(f(k) - C(k))] f'(f(k) - C(k)) \end{array} \right] = 0$$

- Apply $\mathcal{L}$ to $\hat{C}(k; a) = \sum_{j=0}^{n} a_j \phi_j(k)$:

$$R(k; a) \equiv \mathcal{L}\hat{C}(k; a)$$

Projection implementation

- Need a good non-linear equation solver, or optimizer
  - Note that there is no coefficient regression anymore
- Starting values of coefficients are often crucial
  - Set low-degree coefficients to correct slope and curvature
  - Leave higher-order coefficients at zero
- Plot resulting function and residuals
  - Function: Check slope and curvature;
    reduce degree of polynomials if too many waves
  - Residuals: make sure they are small everywhere;
    increase degree if there is bias in their sign
- Code polynomial evaluation as a separate function
  - Or pre-compute polynomials at nodes

Projection Pros and Cons

- When solution well-approximated by known basis, can be really accurate
  - Error can decay exponentially in n if $y(.)$ analytic
  - Guarantees depend on "well-posedness" of $\mathcal{L}$
  - Combine with analytical results to ensure good behavior
- Requires solving system:
  - If linear, up to $O(n^3)$, if nonlinear, $O(n^3)$ each iteration
- Finite difference methods effectively solve system also, but due to locality get sparsity, fast $O(n)$ solves
- Big gains to using structure for faster system solves, integrals
- Advanced function representation can help:
  - Adaptive methods: add new basis/node using residuals
  - Neural nets, Smolyak can be applied if dimension large
- Choice often depends on needs in rest of multistep algorithm
  - May need continuity or other function property

References

- Ordinary Differential Equations
  - SciML Book, Docs, Tutorials
- Projection:
  - Boyd: Chebyshev and Fourier Spectral Methods
  - Chatelin: Spectral Approximation of Linear Operators
  - Econ-specific: Fernández-Villaverde et al Macro Handbook