Non-linear equations

Judd Chapter 5

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

CMU, Tepper School of Business

March 20, 2023

Today: (systems of) nonlinear equations

- Bisection: simple univariate method
- Newton's method: from univariate to bivariate
  - Derivative computation
  - Secant / Broyden: avoiding derivatives
- Fixed point iteration: Gauss-Jacobi/Seidel
- Continuation & Homotopy methods

Systems of non-linear equations

$$F(x) = 0,$$

where $x \in \mathbb{R}^n$, and $F : \mathbb{R}^n \to \mathbb{R}^n$.

- Zero problem: $F(x) = 0$
- Fixed point problem: $F(x) = x$

Examples:

- Optimization FOC (*naive approach*)
- Games: multiple maximizing agents
- General equilibrium models: agents + market
- Z-estimators: estimator solves system

Issues:

- Direct solution methods rarely available: use iterative instead
- Potential multiplicity of solutions

Univariate problem: Bisection method

Solving $f(x) = 0, \qquad x \in \mathbb{R}^1, \qquad F : \mathbb{R}^1 \to \mathbb{R}^1$
Initialization: Find $x^L < x^R$ such that $f(x^L)f(x^R) < 0$.
Choose stopping criteria ϵ and δ.

1. Compute $x^M = \frac{1}{2}(x^L + x^R)$ or $x^L + \frac{f(x^L)}{f(x^L) - f(x^R)}(x^R - x^L)$
2. Compute $f(x^M)$, the new (x^L, x^R) is:

$$\begin{cases} (x^L, x^M) & \text{if} \qquad f(x^L)f(x^M) < 0, \\ (x^M, x^R) & \text{otherwise.} \end{cases}$$

3. Stop if $|f(x^M)| < \delta$ or $x^R - x^L < \epsilon(1 + |x^L| + |x^R|)$
   otherwise go to step 1.

Converges linearly to *a* solution, if f is continuous.

Bisection: illustration

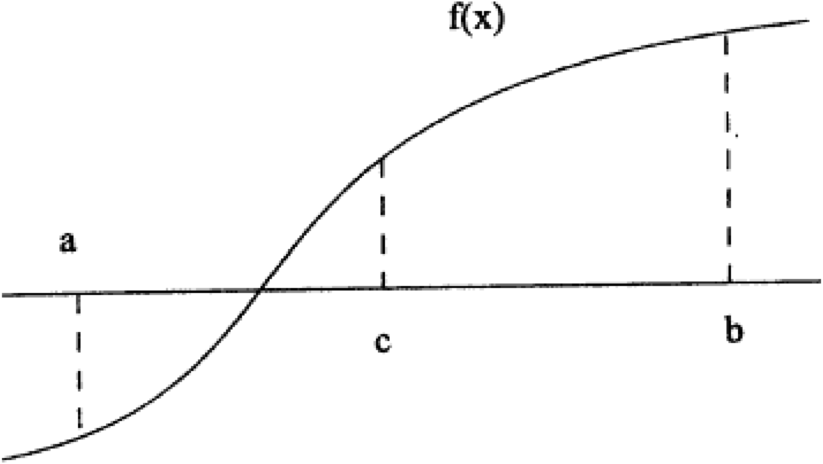

Figure 5.1
Bisection method

Univariate Newton(-Raphson)'s Method

Initialization: Choose initial guess x^0 and stopping criteria ϵ and δ.

1. Compute $f(x^k)$. Compute the step x^k as :

$$x^{k+1} = x^k - f(x^k)/f'(x^k).$$

2. If $|x^{k+1} - x^k| < \epsilon(1 + |x^k|)$, go to step 3; otherwise, to step 1
3. If $|f(x^{k+1})| < \delta$, stop and report success;
   otherwise stop and report failure.

Converges **quadratically** if:

- f is twice continuously differentiable with $f'(x) \neq 0$ and
- The initial guess is good (close to solution)

Bad initial guess can make Newton diverge, circle, or get stuck

Newton: illustration

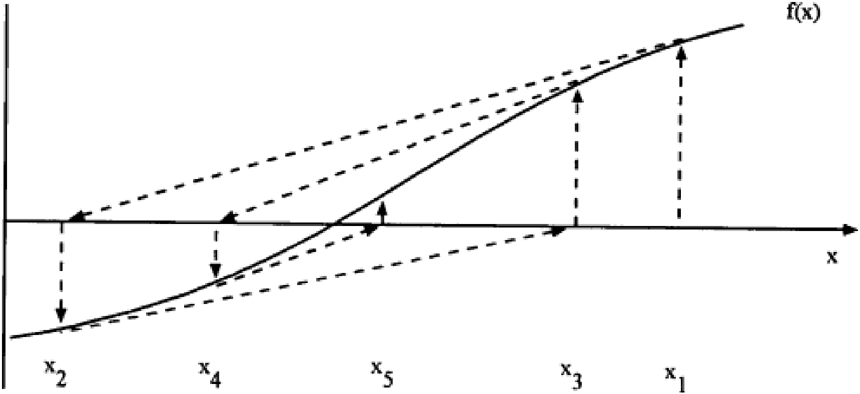

Figure 5.2
Newton method

Newton: Quadratic Convergence

- Let $f(x) \in \mathcal{C}^2$ with $f(x^*) = 0$ and $f'(x^*) \neq 0$
- Then $\exists \epsilon > 0$ s.t. $|x - x^*| < \epsilon$ implies $\lim_{k\to\infty} \frac{|x_{k+1}-x^*|}{|x_k-x^*|^2} = \frac{1}{2}\frac{|f''(x^*)|}{|f'(x^*)|}$
- Proof: By Taylor's theorem (with intermediate value remainder)

$$0 = f(x^*) = f(x_k) + f'(x_k)(x^* - x_k) + \frac{1}{2}f''(\tilde{x})(x^* - x_k)^2$$

- Rearrange and divide by $f'(x_k)$, then $x^* - x_{k+1} =$

$$\frac{f(x_k)}{f'(x_k)} + (x^* - x_k) = -\frac{1}{2}\frac{f''(\tilde{x})(x^* - x_k)^2}{f'(x_k)}$$

- Take absolute values on each side and divide: for small enough initial error, $|x^* - x_k| \to 0$, and so by continuity, limit holds

Newton: Cautions

- If $f'(x^*) = 0$, or only $\mathcal{C}^1$, may converge but not quadratically
- In fact, if derivative near 0, may be slow in practice, and have small radius of convergence
- If starting point not close enough, no guarantees
  - If derivative 0 at an iterate, will stop
  - Can also cycle or explode
  - $\rightarrow$ Start with good guess from more reliable but slower method
- Extensions exist based on higher order derivatives (Householder methods) with faster than quadratic convergence: rarely used since derivative computation may dominate cost

Newton: "pathological" example

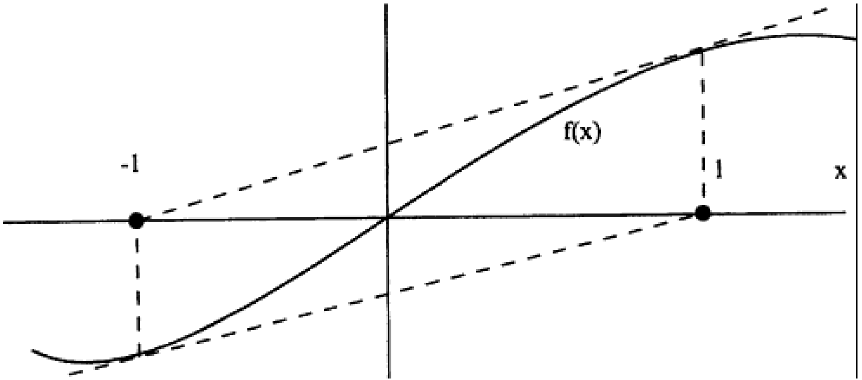

Figure 5.3
Newton method cycle

Multivariate Newton: idea

$$F(x) = 0, \qquad x \in \mathbb{R}^n, \qquad F : \mathbb{R}^n \to \mathbb{R}^n$$
$$F = [f^1(x), f^2(x), ..., f^n(x)]^T$$

- Univariate method is based on linear approximation around x^k
  - $\Rightarrow$ Approximate $F(x)$ by

$$\hat{F}^0(x) = F(x^0) + F_x(x^0)(x - x^0),$$

where $F_x(x)$ is the Jacobian of F at x.

- The approximation $\hat{F}^0(x)$ is equal to zero at:

$$x^1 = x^0 - [F_x(x^0)]^{-1}F(x^0).$$

This suggests the iteration:

$$x^{k+1} = x^k - [F_x(x^k)]^{-1}F(x^k).$$

Multivariate Newton: details

- Same stopping rules as univariate version:
  1. If $\left\|x^{k+1} - x^k\right\| > \epsilon(1 + \left\|x^k\right\|)$, continue iterating
  2. If $\left\|F(x^{k+1})\right\| < \delta$, report success
- Starting value can be crucial:
  - Make your best guess
  - E.g. solution to a simpler version of this problem
  - Continuation method (later in this lecture)
- Potential for multiple solutions:
  - Try many different starting values: a grid for each x_j^0, or random values from some reasonable interval
- One way to prove uniqueness: FOC of concave maximization

Where do we get the Jacobian?

- Analytic Jacobian:
  - By hand – can be labor-intensive
  - Symbolic derivatives (e.g. in Maple) – available in some cases; still have to code it
- Numeric Jacobian (*Finite Difference*, next slide):
  - Precision is low, but Newton's method is robust to that
  - Can be slow to compute
- Automatic differentiation:
  - Takes code for the function, returns code for derivative
  - If efficiently implemented, cost is O(cost of function eval)
  - Fortran (ADIFOR), C (ADIC), Julia (Juliadiff, etc), Python (Autograd/JAX/Torch), Matlab
- Estimate the Jacobian within the method:
  - *Secant* (univariate), *Broyden* (multivariate)
  - Useful if $F_x(x)$ is hard to evaluate

Finite difference derivatives

$$f'(x) = \lim_{h\to 0} \frac{f(x+h) - f(x)}{h} = \lim_{h\to 0} \frac{f(x) - f(x-h)}{h}$$

- One-sided finite difference. Pick $h > 0$, and

$$\widehat{f'_+(x)} = \frac{f(x+h) - f(x)}{h}$$

  - Biased on curved functions
- Two-sided finite difference:

$$\widehat{f'(x)} = \frac{f(x+h) - f(x-h)}{2h}$$

- Multivariate: separate FD for each x_i
  - Two-sided needs twice as many evaluations of f
- Trade off approx vs floating point error
  - $O(h)$ or $O(h^2)$ for 1, 2 sided vs $O(\frac{\epsilon}{h}) \rightarrow$ set $h \propto \sqrt{\epsilon}$ or $\epsilon^{2/3}$

Secant method: univariate aprox. derivative

- Newton method without $f'(x)$.
- Replace the update formula

$$x^{k+1} = x^k - \frac{f(x^k)}{f'(x^k)}$$

with

$$x^{k+1} = x^k - \frac{f(x^k)(x^k - x^{k-1})}{f(x^k) - f(x^{k-1})}.$$

- $f(x)$ evaluations are used to approximate the derivative
- Converges at rate $\frac{1+\sqrt{5}}{2} \approx 1.62$, i.e. superlinear: faster than linear, but slower than quadratic

Improving reliability

- As with Newton, Secant may fail if $f(x^k) - f(x^{k-1}) \approx 0$
- Popular solution is *Brent's Method*
- Starting with bracketing points, perform secant update if $\left|f(x^k) - f(x^{k-1})\right| > \delta$
- Perform bisection update otherwise
- Each iteration series of criteria used to decide between methods
- Ensures continued convergence at linear rate of bisection even if secant would get stuck
- But since most updates are secant updates, usually converges superlinearly
- Version of this is `fzero` in Matlab , option `Brent()` in `Optim.jl`, `brentq` in SciPy

Broyden method: multivariate aprox. derivative

- Approximates Jacobian F_x as J, updated at each iteration
- Update the Jacobian estimate as:

$$J^{k+1} = J^k + \frac{1}{s^{k\prime} s^k}(y^k - J^k s^k) s^{k\prime},$$

where $y^k = F(x^{k+1}) - F(x^k)$, $s^k = x^{k+1} - x^k$.

- Why updates? Linear approximation gives us n secant equations:

$$F(x^{k+1}) - F(x^k) = J^{k+1}(x^{k+1} - x^k)$$

– not enough to determine the n^2 elements of Jacobian J^{k+1}.

- Solution: Impose $J^{k+1} q = J^k q$ whenever $q' s^k = 0$,
to keep J^{k+1} "close" to J^k.
- x^k converges superlinearly; J^k might not

Newton & Quasi-Newton in high dimensions

- In large dimensions, inverse Jacobian is large linear system
- Helpful to approximate even when derivatives fast to calculate
- w/ Broyden, update equation allows fast inversion by Sherman-Morrison formula $(A+uv^T)^{-1} = A^{-1} - \frac{A^{-1}uv^TA^{-1}}{1+v^TA^{-1}u}$
- Given initial inverse, update needs only matrix vector multiplies
- $J_{k+1}^{-1} = J_k^{-1} + \frac{s_k - J_k^{-1}y_k}{s_k^\top J_k^{-1}y_k} s_k^\top J_k^{-1}$
- Each update is $O(n^2)$ instead of $O(n^3)$ for generic linear system
- Trade off larger # of iterates needed for faster iterates

Newton & Quasi-Newton in high dimensions

- Newton valid up to $n = \infty$: use for PDEs, functional equations
- Approximation methods give large but finite matrices
- In many problems, Jacobian is ill-conditioned matrix
  - Especially (approximations of) integral equations
- Multivariate analog of (near) failure of $f' \neq 0$ condition
- Similarly causes slow or non-convergence, small basin
- *Regularize*: Replace Jacobian by invertible surrogate
  - Tikhonov: $(J_k + \lambda_k I)^{-1}$ for $\lambda_k \to 0$
  - Spectral cutoff: SD_k^+V Zero out smallest singular values, invert remainder

Fixed point iteration

- Solving a fixed point problem: $G(x) = x$
  - Transforming $F(x) = 0$: carry out x_i out of each $f^i(x)$
- Iterate on $x^{k+1} = G(x^k)$
- Starting in neighborhood, converges to solution x^* if F Lipschitz & $\rho(G_x(x^*)) < 1$
  - Linear convergence rate $= \rho(G_x(x^*))$
  - We do not know x^*, and $G_x(\cdot)$ can be hard to compute
- Dampening and acceleration work as with linear eq's.
- If there are multiple solutions:
  - "Basin of convergence" – set of starting values that lead to a given solution
  - Some of multiple solutions will be unstable, i.e. we can't converge to them

Fixed point problem and contraction mapping

- Contraction mapping: $G : \mathbb{R}^n \to \mathbb{R}^n$ such that

$$\|G(x) - G(y)\| \leq \beta \|x - y\|, \qquad \forall x, y \in \mathbb{R}^n$$

  for some $\beta \in (0, 1)$
- Contraction mapping theorem (Banach's fixed point): $G(x)$ is a contraction $\Rightarrow$
  - There exists a unique fixed point $G(x^*) = x^*$
  - $x^{k+1} = G(x^k)$ converges to x^*, for any x^0
  - Convergence is linear at rate β
- Converse also true: if iteration converges linearly to unique fixed point, $\exists$ metric in which function is contraction
- Many constructive existence theorems are Banach in disguise: implicit function theorem, Picard iteration for ODEs

Sufficient conditions for contraction

- **Blackwell**'s sufficient conditions for contraction ($x \in \mathbb{R}^n$):
  - *Monotonicity*: $x \leq y$ implies $G(x) \leq G(y)$
  - *Discounting*: $\exists\ \beta \in (0,1)$ such that for any x and $a \in \mathbb{R}^1$: $g_j(\{x_i + a\}_i) \leq g_j(x) + \beta a$ for all j, where $\{x_i + a\}_i$ is vector x with a added to all components.
- Alternately: $G(x)$ is a *differentiable contraction map*
- Global convergence on compact convex set $D \subseteq \mathbb{R}^n$ if
  - $G \in \mathcal{C}^1$
  - $G(D) \subseteq D$
  - $\max_{x \in D} \|G_x(x)\|_\infty < 1$

Other Fixed Point Theorems

- Brouwer/Kiyotaki/Schauder less practical than Banach
  - Every (upper hemi-)continuous function (correspondence) from closed ball to itself has a fixed point
- Used to show GE, Nash equilibria exist, but nonconstructive
- Recent work suggests worst case takes exponential time to find even approximate solution
  - Problem is *PPAD complete* (c.f. Daskalakis, Papadimitriou)
  - Special cases can be tractable (zero sum, potential games, etc)
  - Weaker equilibrium concepts (correlated) also tractable
- Tarski's fixed point theorem sometimes practical
  - Order preserving (monotone) function on complete lattice (all subsets have sup and inf) has nonempty ordered set of fixed points
  - Can find smallest/largest fixed point by iteration, not others

Other methods

- Re-state as least squares problem:

$$\min \sum_{i=1}^{n} [f^i(x)]^2 = SSR(x)$$

  - Optimization is better studied than equations
  - But can get local min, and problem is badly conditioned
- Powell's hybrid method (a version of Dog-Leg or Safeguarding):
  - Check if Newton reduces SSR
  - If not, switch to least squares
- Direction search along Newton's s^k:
  $f(\lambda) = SSR(F(x_k + \lambda s^k))$
  - Trust region: limit λ so Taylor's approximation is accurate
- Transform the problem to reduce curvature:
  - E.g. $e^x h(x) = 0 \Leftrightarrow h(x) = 0$

Continuation method: smart initial guess

- Introduce parameter t ($x \in \mathbb{R}^n$ is still the variable):

$$H(x;t) = 0, \qquad t \in [0,1]$$

  - $t = 0 \implies H(x;t)$ is a problem with known solution x^0
  - $t = 1 \implies H(x;t) = F(x)$, the problem of interest

1. Pick sequence $0 = t^0 < t^1 < \ldots < t^K = 1$
2. Solve problem $H(x^{k+1}; t^{k+1}) = 0$ for x^{k+1}, **using x^k as the initial guess**.

- Constructing H:
  - "Natural" parameter that makes the model simple
  - "Artificial" parameter: $H(x,t) = (1-t)x + tF(x)$

Homotopy method

Exact approach to continuation

- We want the *solution path* though the (x, t)-space: $y(s) = (x(s), t(s))$
- Solution path is described by:

$$H(y(s)) = 0$$

  - Differentiate both sides w.r.t. s:

$$H_y y'(s) = 0$$

  - This is a differential equation, and can be solved numerically; Starting value: $y^0 = (x^0, 0)$
- Path guaranteed to reach $t = 1$ under reasonable conditions.
- Can be labor-intensive to implement (HOMPACK90 in Fortran)
- Can find multiple solutions $\Rightarrow$ good way to explore effects of a natural parameter

References

- Judd, Ch 4 (Solvers) SciML Book Ch 8, 10 (Differentiation)
- QuantEcon. Solvers, Optimizers, and Automatic Differentiation. `https://julia.quantecon.org/more_julia/optimization_solver_packages.html`
  - QuantEcon tutorial on NLsolve.jl, Optim.jl, Roots.jl, LeastSquaresOptim.jl, and differentiation libraries in Julia
  - Use these in practice, code methods yourself on problem sets
- Quantecon Python Newton Tutorial `https://python.quantecon.org/newton_method.html`
- Ron N. Borkovsky, Ulrich Doraszelski, Yaroslav Kryukov (2010) A User's Guide to Solving Dynamic Stochastic Games Using the Homotopy Method. Operations Research 58(4-part-2) 1116-1132.