Lecture IV

Root-finding Algorithms

Gianluca Violante

New York University

Quantitative Macroeconomics

The matching model of the labor market

- Risk-neutral households, that discount future at rate $\beta = 1/R$
- Linear production function: 1 worker-1 firm
- Aggregate productivity z drawn from $\Gamma(z', z)$ with N values
- Employed workers receive a wage $w(z)$ and separate at rate δ
- Firm profits: $z - w(z)$
- Unemployed workers receive benefits b and search for vacancies
- Aggregate CRS matching function: $m(u, v)$ with $p(\theta) = m/u$ and $q(\theta) = m/v$, $\theta \equiv v/u$
- Flow cost for firm of posting a vacancy: κ

The matching model of the labor market

- Value of unemployed and employed workers:

$$U(z) = b + \beta \mathbb{E}_z \left[p(\theta) W(z') + (1 - p(\theta)) U(z') \right]$$

$$W(z) = w(z) + \beta \mathbb{E}_z \left[\delta U(z') + (1 - \delta) W(z') \right]$$

- Values for vacant and matched firms:

$$V(z) = -\kappa + \beta \mathbb{E}_z \left[q(\theta) J(z') + (1 - q(\theta)) V(z') \right]$$

$$J(z) = z - w(z) + \beta \mathbb{E}_z \left[\delta V(z') + (1 - \delta) J(z') \right]$$

- Free entry of firms:

$$V(z) = 0 \qquad \forall z$$

Solving for θ

- Wage $w(z)$ determined by Nash bargaining btw worker and firm
- Wage solves (η bargaining power of worker):

$$\max_{w(z)} \left[W(z) - U(z)\right]^{\eta} \left[J(z) - V(z)\right]^{1-\eta}$$

- Solution:

$$w(z) = \eta z + (1-\eta)b + \eta\kappa\theta(z)$$

- After some algebra:

$$\frac{\kappa}{\beta q(\theta(z_i))} = \sum_{j=1}^{N} \left[(1-\eta)(z_j - b) - \eta\kappa\theta(z_j) + \frac{(1-\delta)\kappa}{q(\theta(z_j))}\right] \Gamma(z_j, z_i)$$

- System of N nonlinear equations in N unknowns $\theta(z_i)$

N-dimensional root-finding problem

- In matrix notation:

$$\begin{pmatrix} \frac{\kappa}{\beta q(\theta(z_1))} \\ \vdots \\ \frac{\kappa}{\beta q(\theta(z_N))} \end{pmatrix} = \Gamma \begin{pmatrix} (1-\eta)(z_1 - b) - \eta\kappa\theta(z_1) + \frac{(1-\delta)\kappa}{q(\theta(z_1))} \\ \vdots \\ (1-\eta)(z_N - b) - \eta\kappa\theta(z_N) + \frac{(1-\delta)\kappa}{q(\theta(z_N))} \end{pmatrix}$$

- where the (i, j) entry of $\Gamma = \Pr\left(z_{t+1} = z_j | z_t = z_i\right)$
- Given u_t and obtained θ_t, you can simulate u_{t+1}, v_t, w_t, y_t
- Recall the law of motion for the endogenous state:

$$u_{t+1} = u_t + \delta(1 - u_t) - p(\theta_t)u(t)$$

Rootfinding problem in one dimension

- We are interested in finding x^* that satisfies

$$f(x) = 0, \qquad f : \mathbb{R} \to \mathbb{R}$$

- The fixed point problem:

$$x = g(x)$$

is isomorphic to the rootfinding problem above.

- A fixed point of g is a root of the function:

$$f(x) \equiv x - g(x) = 0.$$

- A root of f is a fixed point of:

$$g(x) \equiv f(x) + x$$

Convergence of rootfinding algorithms

- A method is said to q-converge at rate k if:

$$\lim_{n\to\infty} \frac{|x^* - x^{n+1}|}{|x^* - x^n|^k} = C, \qquad \text{with } C \quad \text{positive and finite}$$

- A method is said to r-converge at rate k if:

  (i) the sequence of the algorithm's successive error terms is dominated by a sequence $\{v_n\}$ i.e.:

$$|x^* - x^n| \leq v_n \qquad \text{for all} \quad n$$

  (ii) and the sequence $\{v_n\}$ q-converges at rate k,

Function iteration

- If g is a contraction, we know that, if we start from x_0, the iteration

  $$x_{k+1} = g\left(x_k\right)$$

  converges to x^* for any initial value x_0

- If x_0 is close enough to x^*, g needs not be globally a contraction mapping. It is enough that it is locally around x^*, i.e.

  $$\left|g'\left(x^*\right)\right| < 1$$

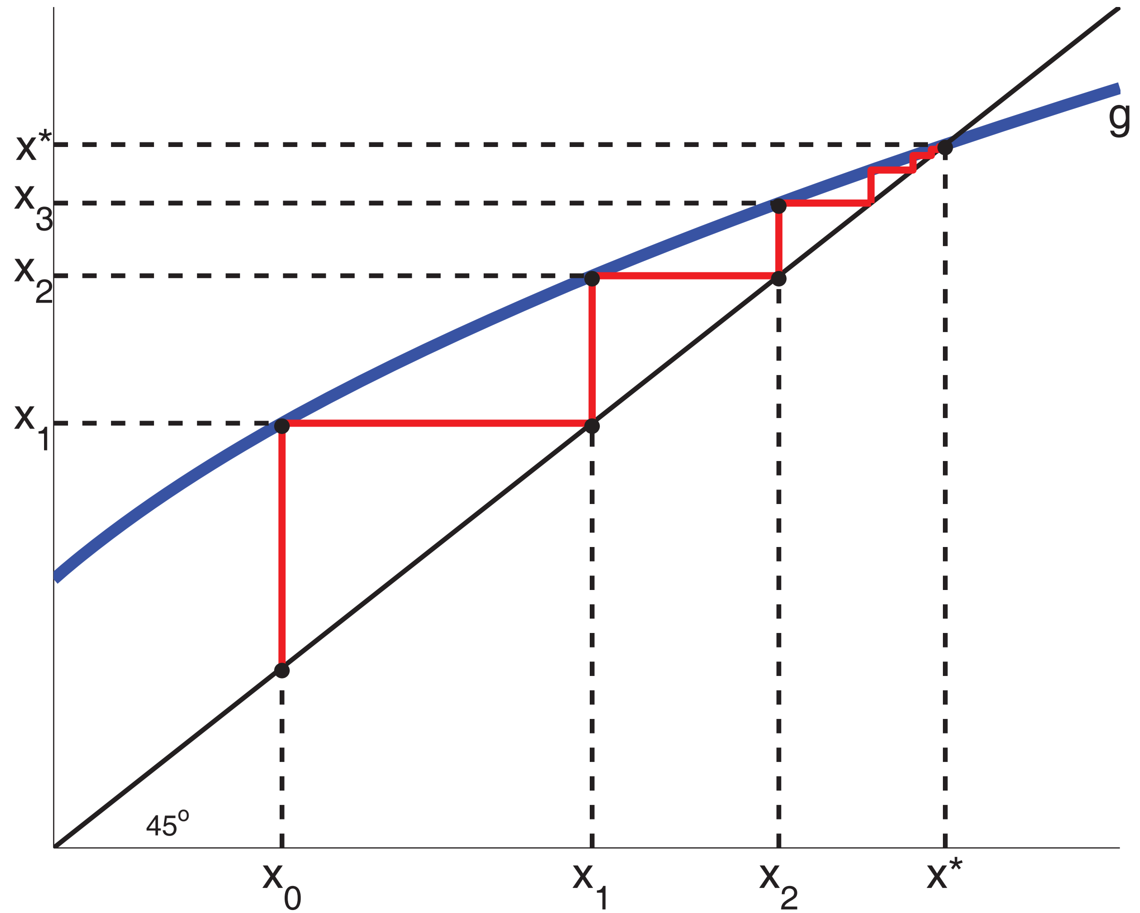
- It can be proven that:

  $$\left|x^* - x^{n+1}\right| \leq \left|g'\left(x^*\right)\right| \times \left|x^* - x^n\right|$$

  hence this method q-converges linearly $(k = 1)$ with $C = \left|g'\left(x^*\right)\right|$

Function Iteration

Computing Fixed-Point of g Using Function Iteration.



Bisection method

- Simplest and most robust algorithm for finding the root of a one-dimensional continuous function on a closed interval
- Suppose $f(x)$ is defined between $[a, b]$ where $f(a)$ and $f(b)$ have opposite signs, e.g. $f(a) < 0$ and $f(b) > 0$
- By the mean value theorem there exists at least a zero

1. Set $n = 1$, $a^n = a$ and $b^n = b$ and
2. Compute $c^n = \frac{a^n + b^n}{2}$
3. If $f(c^n) > 0$, set $a^{n+1} = c^n$ and $b^{n+1} = b$. Otherwise set $b^{n+1} = c^n$ and $a^{n+1} = a$
4. If your convergence criterion is satisfied stop, if not set $n = n + 1$ and go back to step 2

Advantages and disadvantages of bisection method

- Robust and stable: guaranteed to find an approximate solution within a given degree of accuracy in a known number of iterations.

- Step 1: solution is in $[a, b]$. Step 2 is in an interval of size $|b - a|/2$ and so on. At the n^{th} iteration, the solution is in an interval of size:

$$|b^n - a^n| = \frac{|b - a|}{2^{n-1}}$$

which means that if our tolerance error is δ, we have:

$$\delta = \frac{|b - a|}{2^{n-1}}$$

and we will will reach it in

$$n = 1 + \log_2 \left(\frac{b - a}{\delta} \right) \qquad \text{number of iterations}$$

Advantages and disadvantages of bisection method

- It does not require computation of derivatives, hence needs continuity but not differentiability
- However, because it does not use derivatives does not exploit information on the slope of the function, it is slow
- Note that:

$$
\begin{aligned}
|x^* - x^n| &\leq \frac{|b^n - a^n|}{2} = \frac{|b - a|}{2^n} \\
|x^* - x^{n+1}| &\leq \frac{|b^{n+1} - a^{n+1}|}{2} = \frac{|b - a|}{2^{n+1}}
\end{aligned}
$$

- Thus, the bisection method r-converges linearly: slow!
- Moreover, it cannot be generalized to the multi dimensional case

Newton's method

- As usual, we want to find the solution to: $f(x) = 0$
- Approximate the function f at point x^n in iteration n as:

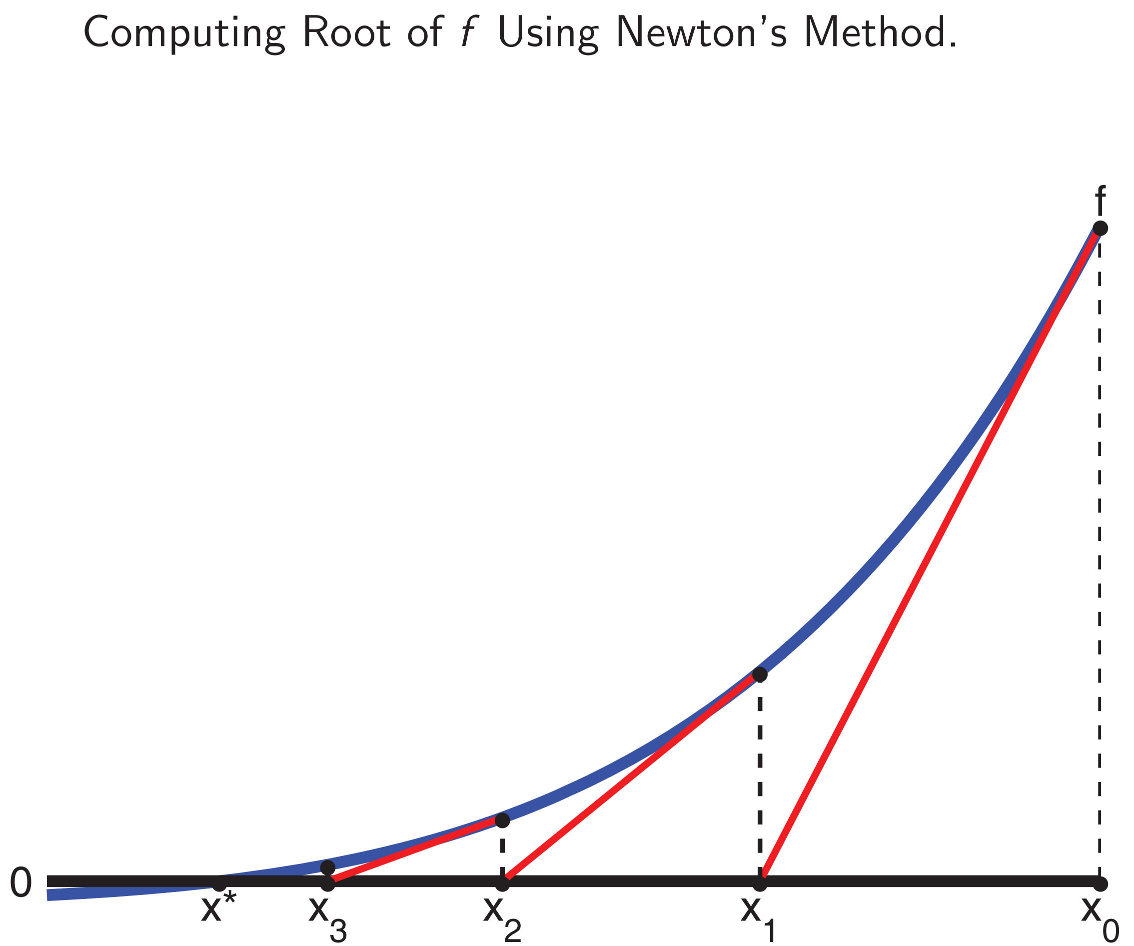
$$f(x) \simeq f(x^n) + f'(x^n)(x - x^n)$$

and update x as if you were solving for a zero of the appx. funct.:

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

- f must be differentiable and f' must be computed by hand or numerically by finite difference methods
- Unstable if function changes derivative quickly
- Backstepping: If the full Newton step $\Delta x \equiv x^{n+1} - x^n$ doesn't offer improvement over x^n, then one "backsteps" toward x^n by repeatedly cutting Δx in half until $x^n + \Delta x$ does offer improvement

Newton method



Definition of slope at $x_0 : -f'(x_0) = f(x_0)/(x_1 - x_0)$

Newton's method is fast (I)

- Suppose x^* is the solution to $f(x) = 0$
- At iteration n the Newton method's error is:

$$\varepsilon^n = x^n - x^*$$

- It follows that:

$$\varepsilon^{n+1} = x^{n+1} - x^* = x^n - \frac{f(x^n)}{f'(x^n)} - x^* = \varepsilon^n - \frac{f(x^n)}{f'(x^n)}$$

- Now consider the following approximations around the solution x^*:

$$\begin{aligned} f(x^n) &\simeq 0 + f'(x^*)\,\varepsilon^n + \frac{1}{2} f''(x^*)\,(\varepsilon^n)^2 \\ f'(x^n) &\simeq f'(x^*) \end{aligned}$$

Newton's method is fast II

- Combining equations:

$$\begin{aligned} \frac{f\left(x^n\right)}{f'\left(x^n\right)} &\simeq \varepsilon^n + \frac{1}{2}\frac{f''\left(x^*\right)}{f'\left(x^*\right)}\left(\varepsilon^n\right)^2 \\ \varepsilon^n - \varepsilon^{n+1} &\simeq \varepsilon^n + \frac{1}{2}\frac{f''\left(x^*\right)}{f'\left(x^*\right)}\left(\varepsilon^n\right)^2 \\ \varepsilon^{n+1} &\simeq -\frac{1}{2}\frac{f''\left(x^*\right)}{f'\left(x^*\right)}\left(\varepsilon^n\right)^2 \\ \frac{\left|\varepsilon^{n+1}\right|}{\left|\varepsilon^n\right|^2} &\simeq \left|\frac{1}{2}\frac{f''\left(x^*\right)}{f'\left(x^*\right)}\right| = C > 0 \end{aligned}$$

- It follows that the q-convergence rate of the Newton's method is quadratic

- A lot faster than bisection

Quasi-Newton method (derivative-free)

$$f(x) \simeq f(x^n) + f'(x^n)(x - x^n)$$

- If computing derivative is too costly, instead of f' use the slope of the secant function going through $f(x^n)$ and $f\left(x^{n-1}\right)$:

$$\frac{f(x^n) - f\left(x^{n-1}\right)}{x^n - x^{n-1}} = \frac{f(x^n) - f\left(x^n - \left[x^n - x^{n-1}\right]\right)}{\left[x^n - x^{n-1}\right]}$$

  which converges to the one-sided derivative as the distance between successive points gets close

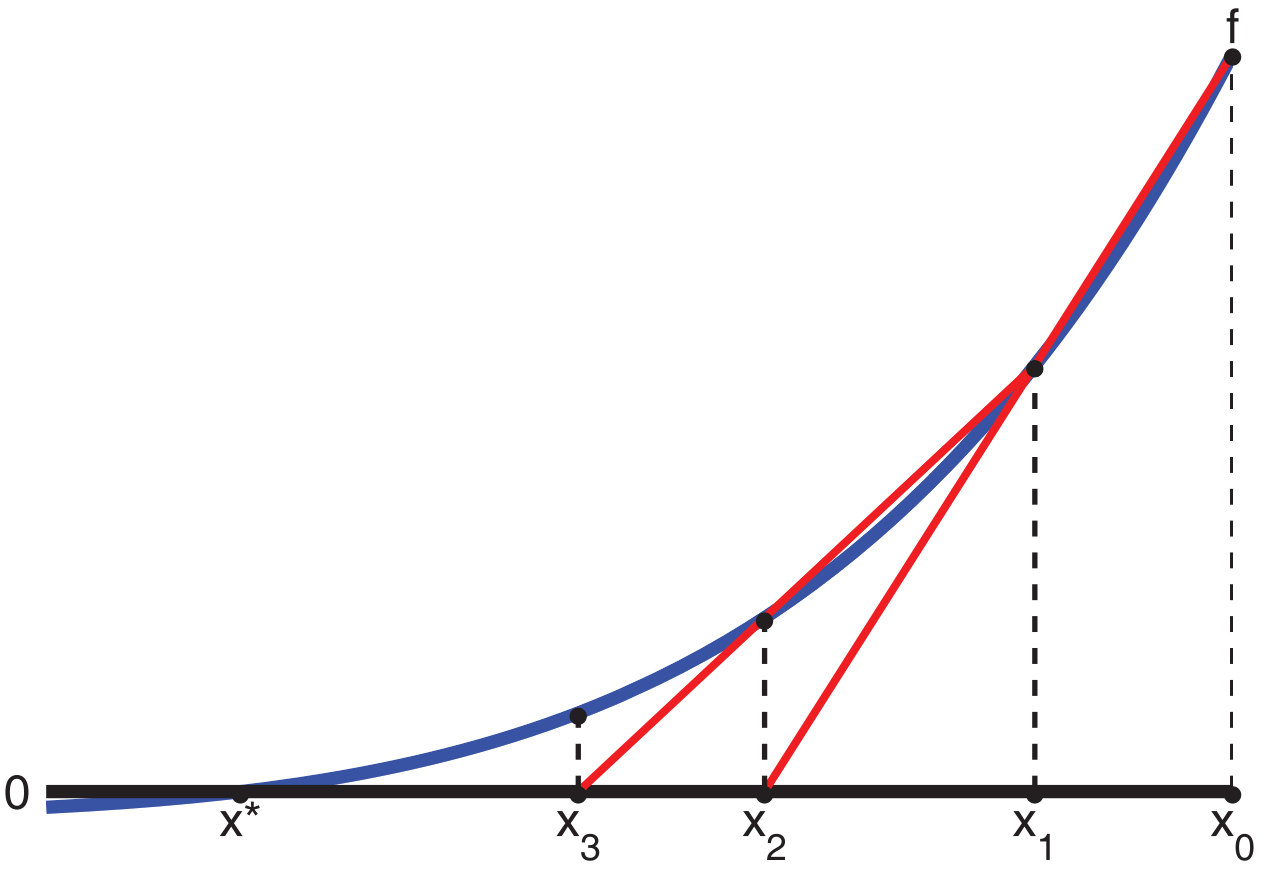
- Iteration scheme becomes:

$$x^{n+1} = x^n - \left[\frac{x^n - x^{n-1}}{f(x^n) - f(x^{n-1})}\right] f(x^n)$$

- Convergence is slower though: q-convergence rate is superlinear (between linear and quadratic).

Quasi-Newton method

Computing Root of f Using Secant Method.



Broyden's algorithm

- Multidimensional version of the univariate quasi-Newton
- Let $\mathbf{f} : \mathbf{R}^k \to \mathbf{R}^k$ so we have a system of k equations and k unknowns (as in the matching model)
- Given $\mathbf{x}^n$, the updated root $\mathbf{x}^{n+1}$ is found by solving the linear problem obtained by replacing $\mathbf{f}$ with its first order Taylor expansion around $\mathbf{x}^n$

$$\mathbf{f}(\mathbf{x}) \simeq \mathbf{f}(\mathbf{x}^n) + \mathbf{J}^n (\mathbf{x} - \mathbf{x}^n)$$

  where $\mathbf{J}^n$ is the $(k \times k)$ Jacobian evaluated at $\mathbf{x}^n$.
- This yields the rule for updating:

$$\mathbf{x}^{n+1} = \mathbf{x}^n - (\mathbf{J}^n)^{-1} \mathbf{f}(\mathbf{x}^n)$$

Broyden's algorithm

- To avoid computing $\mathbf{J}^n$, use $\mathbf{A}^n$ given by secant method:

$$\mathbf{A}^{n+1} = \mathbf{A}^n + \left[\mathbf{f}\left(\mathbf{x}^{n+1}\right) - \mathbf{f}\left(\mathbf{x}^n\right) - \mathbf{A}^n \mathbf{d}_n\right] \frac{\left(\mathbf{d}^n\right)'}{\left(\mathbf{d}^n\right)' \mathbf{d}^n}$$

  where $\mathbf{d}^n = \mathbf{x}^{n+1} - \mathbf{x}^n$.

- Guess of $\mathbf{A}^0$: scaled identity matrix
- Superlinear q-convergence, like secant
- The method can be accelerated by avoiding the inverse operation in the updating stage. Instead of guessing/updating the jacobian, we guess/update the inverse jacobian
- Brent method: instead of linear approximation of f uses a quadratic approximation of f^{-1}.