Classical iterative methods for linear and nonlinear systems

Ed Bueler

MATH 615 Numerical Analysis of Differential Equations

Spring 2023

example linear systems

- suppose we want to solve the linear system

$$A\mathbf{x} = \mathbf{b} \tag{1}$$

  where $A \in \mathbb{R}^{m\times m}$ and $\mathbf{b} \in \mathbb{R}^m$
- the goal is to find $\mathbf{x} \in \mathbb{R}^m$
- throughout these notes we use 2 linear system examples:

  LS1

$$\begin{bmatrix} 2 & 1 & 0 \\ 0 & 2 & 1 \\ 1 & 0 & 3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 2 \\ 1 \\ 4 \end{bmatrix}$$

  LS2

$$\begin{bmatrix} 1 & 2 & 3 & 0 \\ 2 & 1 & -2 & -3 \\ -1 & 1 & 1 & 0 \\ 0 & 1 & 1 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 7 \\ 1 \\ 1 \\ 3 \end{bmatrix}$$

- it is trivial to find solutions of LS1, LS2 using the "`x=A\b`" black box in MATLAB (or similar)
- LS1 and LS2 stand-in for the large linear systems we get from applying finite difference (FD) schemes to ODE and PDE problems

residual

- the *residual* of a vector $\mathbf{v}$ in linear system (1) is the vector

$$\mathbf{r}(\mathbf{v}) = \mathbf{b} - A\mathbf{v} \tag{2}$$

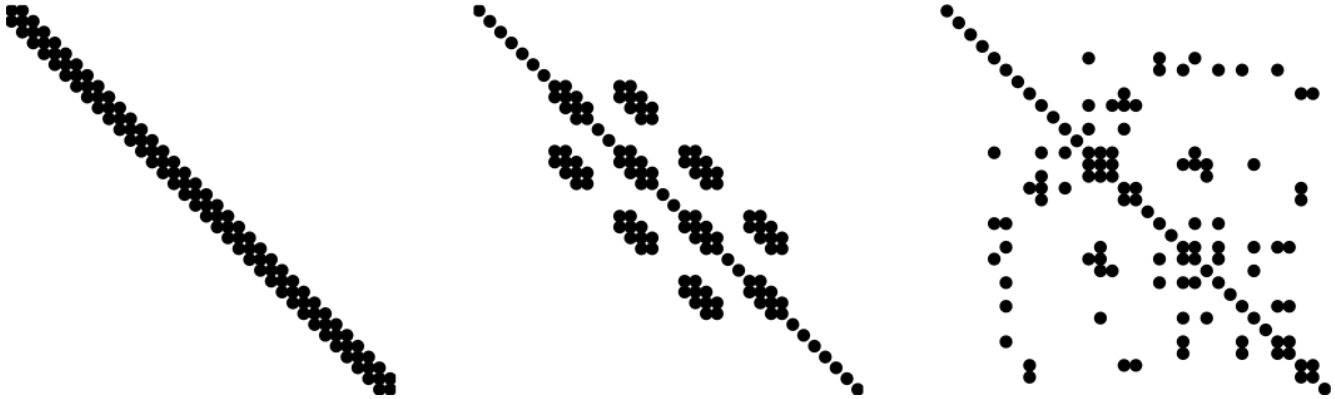
- making the residual zero is the same as solving the system:

$$A\mathbf{x} = \mathbf{b} \iff \mathbf{r}(\mathbf{x}) = 0$$

- evaluating $\mathbf{r}(\mathbf{v})$ needs a matrix-vector product and a vector subtraction
  - requires $O(m^2)$ operations at worst
  - by comparison, applying Gauss elimination to solve linear system (1) is an $O(m^3)$ operation in general

sparse matrices

- *definition.* a matrix with enough zeros to allow exploitation of that fact is called *sparse*
  - the figure shows `spy` plots of 3 matrices; nonzero entries are in black
  - numerical schemes for differential equations generate matrices A for which the majority, often 99% or more, of the entries are zero
- a non-sparse matrix is called *dense*, e.g. when most entries are nonzero
  - even if A is sparse, A^{-1} is generally dense
- evaluating the residual of a sparse matrix with at most z nonzero entries per row requires $O(m)$ operations
  - specifically, at most $(2z+1)m$ operations



Richardson iteration

- *iterative methods* for the linear system $A\mathbf{x} = \mathbf{b}$ attempt to solve it based only on computing the residual or applying A to a vector
  - one wants the sequence of approximations, the iterates, to *converge* to the solution $\mathbf{x} = A^{-1}\mathbf{b}$
  - Iterative methods always require an initial iterate $\mathbf{x}_0$
- for example, *Richardson iteration* adds a multiple ω of the last residual at each step:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \omega(\mathbf{b} - A\mathbf{x}_k) \tag{3}$$

- for system LS1, using initial iterate $\mathbf{x}_0 = 0$ and $\omega = 1/5$, Richardson gives:

$$\mathbf{x}_0 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \mathbf{x}_1 = \begin{bmatrix} 0.4 \\ 0.2 \\ 0.8 \end{bmatrix}, \mathbf{x}_2 = \begin{bmatrix} 0.6 \\ 0.16 \\ 1.04 \end{bmatrix}, \mathbf{x}_3 = \begin{bmatrix} 0.728 \\ 0.088 \\ 1.096 \end{bmatrix}, \ldots, \mathbf{x}_{10} = \begin{bmatrix} 0.998 \\ -0.017 \\ 1.01 \end{bmatrix}, \ldots$$

  these iterates seem to be converging to $\mathbf{x} = [1\ 0\ 1]^\top$, the solution to LS1
- when does Richardson iteration work?

recall: eigenvalues and vectors

- a complex number $\lambda \in \mathbb{C}$ is an *eigenvalue* of a square matrix $B \in \mathbb{R}^{m \times m}$ if there is a nonzero vector $\mathbf{v} \in \mathbb{C}^m$ so that $B\mathbf{v} = \lambda \mathbf{v}$
  - λ is a root of a polynomial
  - even if B is a real matrix, λ may be complex
  - if λ is complex and B is real then $\mathbf{v}$ must be complex
- the set of all eigenvalues of B is the *spectrum* $\sigma(B)$ of B
- the *spectral radius* $\rho(B)$ is the largest absolute value of an eigenvalue:

$$\rho(B) = \max_{\lambda \in \sigma(B)} |\lambda|$$

  - fact: $\rho(B) \leq \|B\|$ in any induced matrix norm

spectral properties and convergence of iterations

- properties of a matrix B which can be described in terms of eigenvalues are generically called *spectral properties*
- some examples:
  - the spectral radius $\rho(B)$ itself
  - the 2-norm $\|B\|_2 = \sqrt{\rho(B^\top B)}$
  - the 2-norm condition number $\kappa(B) = \|B\|_2 \|B^{-1}\|_2$
- a general idea:

  whether an iterative method for solving a linear system $A\mathbf{x} = \mathbf{b}$ converges, or not, depends on the spectral properties of A, or on the spectral properties of matrices built from A
- the right-hand side $\mathbf{b}$ in the linear system $A\mathbf{x} = \mathbf{b}$, and the initial iterate $\mathbf{x}_0$, generally *do not* determine whether an iteration converges
  - a good choice of $\mathbf{x}_0$ *can* speed up convergence when it happens

convergence of the Richardson iteration

- the Richardson iteration (3) can be rewritten as

$$\mathbf{x}_{k+1} = (I - \omega A)\mathbf{x}_k + \omega \mathbf{b}$$

  - confirm this!
- Richardson iteration converges if and only if all the eigenvalues of the matrix $I - \omega A$ are inside the unit circle:

$$\text{Richardson converges if and only if } \rho(I - \omega A) < 1$$

  - see the lemma on the next slide
  - note $\rho(I - \omega A) < 1$ means $(I - \omega A)\mathbf{x}_k$ is smaller in magnitude than $\mathbf{x}_k$
  - fact: *if* $\|I - \omega A\| < 1$ *then* Richardson converges

convergence lemma for linear iterations

Lemma

$$\mathbf{y}_{k+1} = M\mathbf{y}_k + \mathbf{c}$$

converges to the solution of $\mathbf{y} = M\mathbf{y} + \mathbf{c}$ *for all initial* $\mathbf{y}_0$ *if and only if*

$$\rho(M) < 1.$$

Proof.

Iterate. That is, write out a few cases:

$$\mathbf{y}_2 = M(M\mathbf{y}_0 + \mathbf{c}) + \mathbf{c} = M^2\mathbf{y}_0 + (I + M)\mathbf{c},$$
$$\mathbf{y}_3 = M(M^2\mathbf{y}_0 + (I + M)\mathbf{c}) + \mathbf{c} = M^3\mathbf{y}_0 + (I + M + M^2)\mathbf{c},$$

and so on. By induction we get $\mathbf{y}_k = M^k\mathbf{y}_0 + p_k(M)\mathbf{c}$ where $p_k(x) = 1 + x + x^2 + \ldots + x^{k-1}$. But $p_k(x) \to 1/(1-x)$ as $k \to \infty$ iff $x \in (-1, 1)$; it is a convergent Maclaurin series on that open interval. Also, $\rho(M) < 1$ iff $M^k \to 0$. Thus $\mathbf{y}_k \to (I - M)^{-1}\mathbf{c}$ iff $\rho(M) < 1$. □

convergence of the Richardson iteration 2

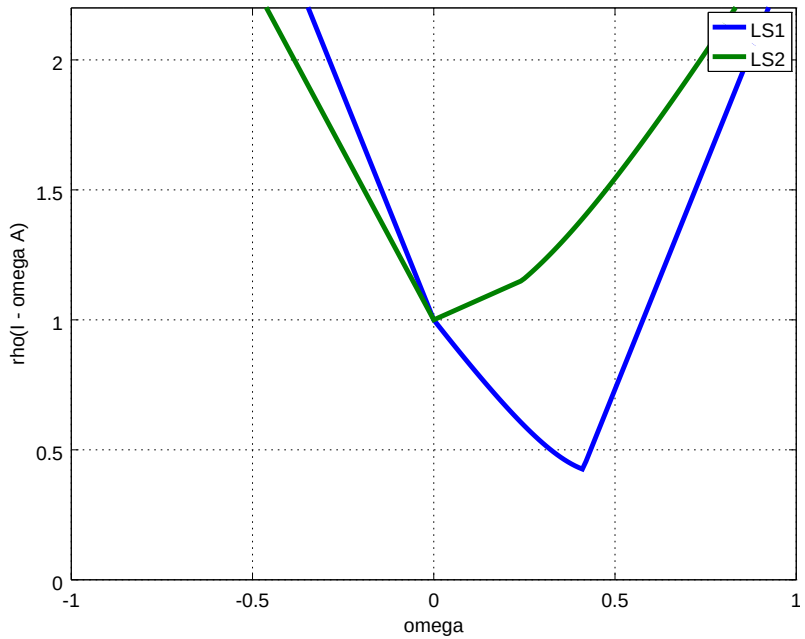
- since the Richardson iteration converges iff $\rho(I - \omega A) < 1$, we should choose ω based on the principle that

$$\omega A \text{ should be close to the identity } I$$

  - often not possible!

- in small cases we can graph $f(\omega) = \rho(I - \omega A)$:

```
omega = -1:.01:1;
rho = zeros(size(omega));
for j = 1:length(omega)
    M = eye(n) - omega(j) * A;
    rho(j) = max(abs(eig(M)));
end
plot(omega,rho)
```



for LS1: $\rho(I - \omega A)$ dips below 1 for $0 < \omega \lesssim 0.6$

for LS2: $\rho(I - \omega A) \geq 1$ always

- note $\rho(I - 0A) = 1$ ... so no convergence when $\omega \approx 0$
- for LS1, figure suggests $\omega \approx 0.4$ gives fastest convergence

matrix splitting

- several classical iteration methods "split" the matrix A before iterating
  - Richardson iteration is an exception
- the best known, and simplest, iteration based on splitting is *Jacobi iteration*, which extracts and inverts the diagonal of A
- the splitting we consider is

$$A = D - L - U$$

  where
  - D is the diagonal of A
  - L is strictly lower triangular ($\ell_{ij} = 0$ if $i \le j$)
  - U is strictly upper triangular ($u_{ij} = 0$ if $i \ge j$)
- you can split *any* matrix this way
- see section 4.2 of the textbook

- so that D is an invertible matrix, for the remaining slides we assume

  all diagonal entries of A are nonzero: $\quad a_{ii} \neq 0$

Jacobi iteration

- the Jacobi iteration is

$$D\mathbf{x}_{k+1} = \mathbf{b} + (L + U)\mathbf{x}_k \tag{4}$$

- if it converges then $D\mathbf{x} = \mathbf{b} + (L + U)\mathbf{x}$, which is the same as $A\mathbf{x} = \mathbf{b}$
- we could also write it as $\mathbf{x}_{k+1} = D^{-1}\left(\mathbf{b} + (L + U)\mathbf{x}_k\right)$ or as

$$x_i^{(k+1)} = \frac{1}{a_{ii}}\left(b_i - \sum_{j \neq i} a_{ij} x_j^{(k)}\right) \tag{5}$$

where $x_j^{(k)}$ denotes the jth entry of the kth iterate $\mathbf{x}_k$

- make sure you understand why (4) and (5) are the same!

Gauss-Seidel iteration

- *Gauss-Seidel iteration* instead extracts and inverts the non-strict lower-triangular part of A
- if $A = D - L - U$ then Gauss-Seidel is

$$(D - L)\mathbf{x}_{k+1} = b + U\mathbf{x}_k \tag{6}$$

  - we could also write it as "$\mathbf{x}_{k+1} = (D - L)^{-1}(b + U\mathbf{x}_k)$", but *don't* . . . that would miss the point!
- instead we write it as $\quad D\mathbf{x}_{k+1} = b + U\mathbf{x}_k + L\mathbf{x}_{k+1} \quad$ or equivalently:

$$x_i^{(k+1)} = \frac{1}{a_{ii}}\left(b_i - \sum_{j>i} a_{ij}x_j^{(k)} - \sum_{j<i} a_{ij}x_j^{(k+1)}\right) \tag{7}$$

- the lower-triangular entries of A apply to *those entries of* $\mathbf{x}_{k+1}$ *which have already been computed*
- form (7) is actually *easier* to implement than Jacobi (5) (why?)

convergence conditions for Jacobi and Gauss-Seidel

- the convergence lemma says that
  - Jacobi iteration converges if and only if $\rho(D^{-1}(L+U)) < 1$
  - Gauss-Seidel iteration converges if and only if $\rho((D-L)^{-1}U) < 1$
- these conditions are hard to use in practice because computing a spectral radius can be just as hard as solving the original system

diagonally-dominant matrices

- *definition*. A is *strictly diagonally-dominant* (SDD) if $|a_{ii}| > \sum_{j \neq i} |a_{ij}|$
  - A in LS1 is SDD
  - A in LS2 is not
  - SDD is a common, but not universal, property of the matrices coming from FD schemes on ODEs and PDEs
- facts:[1]
  - if A is strictly diagonally-dominant then both the Jacobi and Gauss-Seidel iterations converge; see problem **P14**
  - if A is symmetric positive definite then Gauss-Seidel iteration converges
  - these are only *sufficient* conditions, e.g. there are nonsymmetric A, which are *not* diagonally-dominant, for which the iterations converge
- unlike the "$\rho(\dots) < 1$" conditions on the last slide, it is easy to check SDD

[1] section 11.2 of Golub and van Loan, *Matrix Computations*, 4th edition 2013

interlude: past and future

- the Jacobi and Gauss-Seidel iterations are from the 19th century
- Richardson iteration first appears in a 1910 publication
- the early history of numerical partial differential equations, e.g. in the 1920 to 1970 period, heavily used these classical iterations
  - a generalization of Gauss-Seidel iteration called *successive over-relaxation*, was a particular favorite around 1970; see section 4.2 of the textbook
- none of these iterations work on system LS2
- there are better iterative ideas; they flourished starting in the 1980-90s
  - among the best known are CG = *conjugate gradients* ($\sim$1950) and GMRES = *generalized minimum residuals* (Saad and Schultz, 1986)
  - GMRES works (i.e. converges at some rate) on LS2
  - *but* there is no "good iteration" with a universally-fast convergence rate for all matrices[2]
- iterative methods for solving linear systems will dominate the future:
  - they are obligatory on sufficiently-big systems
  - they work better in parallel than direct methods like Gauss elimination
  - they can exploit partial knowledge of the underlying model/question

[2] a remarkable 1992 theorem by Nachtigal, Reddy, and Trefethen

interlude: biographies

- of course, Gauss (1777–1855) did lots of big stuff:
  `en.wikipedia.org/wiki/Carl_Friedrich_Gauss`
- Jacobi (1804–1851) also has his name on the "Jacobian", the matrix of derivatives appearing in Newton's method for systems of equations:
  `en.wikipedia.org/wiki/Carl_Gustav_Jacob_Jacobi`
- Seidel (1821–1896) is relatively little known:
  `en.wikipedia.org/wiki/Philipp_Ludwig_von_Seidel`
- Richardson (1881–1953) is the most interesting. He invented numerical weather forecasting, doing it by-hand for fun during WWI. Later, as a pacifist and quaker, he quit the subject when he found his meteorological work was being used by chemical weapons engineers and the military:
  `en.wikipedia.org/wiki/Lewis_Fry_Richardson`

nonlinear systems

- generally, systems of equations are *not* linear, so they cannot be written in the form $A\mathbf{x} = \mathbf{b}$
- instead, we can write any square system of equations using a function $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^n$:

$$\mathbf{f}(\mathbf{x}) = \mathbf{0}$$

  - "square" simply means the number of unknowns (inputs to $\mathbf{f}$) equals the number of equations (outputs from $\mathbf{f}$)
- if $\mathbf{f}$ is smooth then its first derivative is a matrix-valued function called the *Jacobian* of $\mathbf{f}$:

$$J_{ij} = \frac{\partial f_i}{\partial x_j}$$

  - for each $\mathbf{x}$, $J(\mathbf{x})$ is an $n \times n$ matrix
- *example*: a function and its Jacobian:

$$\mathbf{f}(\mathbf{x}) = \begin{bmatrix} \frac{1}{2}e^{2x_1} - x_2 \\ \\ x_1^2 + x_2^2 - 1 \end{bmatrix}, \qquad J(\mathbf{x}) = \begin{bmatrix} e^{2x_1} & -1 \\ \\ 2x_1 & 2x_2 \end{bmatrix}$$

example of a nonlinear system

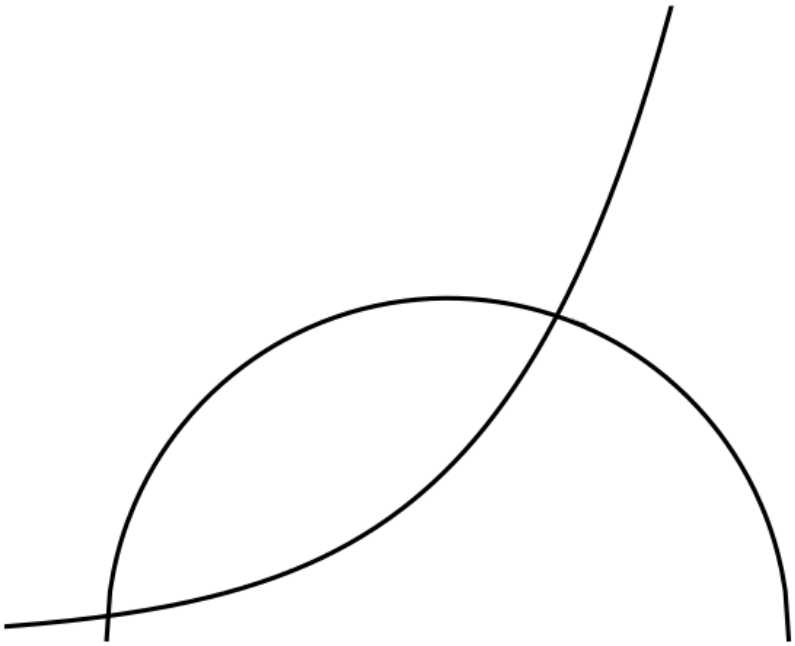
- here is an example of a nonlinear system with $\mathbf{f} : \mathbb{R}^2 \to \mathbb{R}^2$
- find where the circle of radius 1 intersects the graph of an exponential:

$$x^2 + y^2 = 1$$
$$y = \tfrac{1}{2} e^{2x}$$

  - note there are two intersection points
  - I *don't* know how to find them by hand
- renaming $x = x_1, y = x_2$ allows us to write this as $\mathbf{f}(\mathbf{x}) = \mathbf{0}$:

$$\mathbf{f}(\mathbf{x}) = \begin{bmatrix} \frac{1}{2} e^{2x_1} - x_2 \\ \\ x_1^2 + x_2^2 - 1 \end{bmatrix}$$

  - the same example as on the last slide



Newton's method

- our classical linear iterations generated sequences $\mathbf{x}_k$ which solved $\mathbf{r}(\mathbf{x}) = \mathbf{b} - A\mathbf{x} = 0$, that is, so that $\mathbf{r}(\mathbf{x}_k) \to \mathbf{0}$
- similarly, *Newton's method* for a system of nonlinear equations generates a sequence $\mathbf{x}_k$ so that $\mathbf{f}(\mathbf{x}_k) \to \mathbf{0}$
  - as usual, an initial iterate $\mathbf{x}_0$ is needed
- it repeatedly linearizes $\mathbf{f}(\mathbf{x}) = 0$; one needs to solve a linear system to get each new iterate $\mathbf{x}_k$
- it is often surprisingly fast!
- Newton's method:

$$J(\mathbf{x}_k)\,\mathbf{s} = -\mathbf{f}(\mathbf{x}_k), \tag{8}$$

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{s} \tag{9}$$

  - eqn (8) is a system of linear equations which determines the *step* $\mathbf{s}$
  - eqn (9) takes the step to the next iterate

explanation of Newton's method

- suppose $\mathbf{x}_k$ is a current estimate of the solution $\mathbf{f}(\mathbf{x}) = \mathbf{0}$
- now linearize $\mathbf{f}$ around $\mathbf{x}_k$ using Taylor's theorem with remainder:

$$\mathbf{f}(\mathbf{x}) = \mathbf{f}(\mathbf{x}_k) + J(\mathbf{x}_k)(\mathbf{x} - \mathbf{x}_k) + O(h^2)$$

  where $h = \|\mathbf{x} - \mathbf{x}_k\|$ is the distance between the basepoint $\mathbf{x}_k$ and another value $\mathbf{x}$
- the linearization of $\mathbf{f}$ comes from dropping the $O(h^2)$ term:

$$\ell(\mathbf{x}) = \mathbf{f}(\mathbf{x}_k) + J(\mathbf{x}_k)(\mathbf{x} - \mathbf{x}_k)$$

- now define the next iterate $\mathbf{x}_{k+1}$ as the zero of the linearization ℓ:

$$\mathbf{0} = \mathbf{f}(\mathbf{x}_k) + J(\mathbf{x}_k)(\mathbf{x}_{k+1} - \mathbf{x}_k)$$

- denote the difference $\mathbf{x}_{k+1} - \mathbf{x}_k$ by $\mathbf{s}$, and thus get the previous form:

$$J(\mathbf{x}_k)\,\mathbf{s} = -\mathbf{f}(\mathbf{x}_k),$$
$$\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{s}$$

example

- recall the example, where a circle intersects an exponential: $x^2 + y^2 = 1$, $y = \frac{1}{2}e^{2x}$
- this nonlinear residual function and Jacobian:

$$\mathbf{f}(\mathbf{x}) = \begin{bmatrix} \frac{1}{2}e^{2x_1} - x_2 \\ \\ x_1^2 + x_2^2 - 1 \end{bmatrix}, \; J(\mathbf{x}) = \begin{bmatrix} e^{2x_1} & -1 \\ \\ 2x_1 & 2x_2 \end{bmatrix}$$

- for example, run Newton's method starting from $\mathbf{x}_0 = (1, 1)$:

$$\mathbf{x}_0 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \; \mathbf{x}_1 = \begin{bmatrix} 0.619203 \\ 0.880797 \end{bmatrix}, \; \mathbf{x}_2 = \begin{bmatrix} 0.394157 \\ 0.948623 \end{bmatrix},$$
$$\mathbf{x}_3 = \begin{bmatrix} 0.325199 \\ 0.948157 \end{bmatrix}, \; \mathbf{x}_4 = \begin{bmatrix} 0.319665 \\ 0.947547 \end{bmatrix} \cdots$$

- $\mathbf{x}_3$ is already at the intersection visually; note $\|\mathbf{f}(x_4)\|_2 = \texttt{7e-5}$ and $\|\mathbf{f}(x_5)\|_2 = \texttt{2e-9}$; we have solved accurately