MA 580; Iterative Methods for Nonlinear Equations

C. T. Kelley
NC State University
tim_kelley@ncsu.edu

Version of November 2, 2016

Read Chapter 4 sections 5.1--5.4 of the Red book

NCSU, Fall 2016
Part VIIa: Nonlinear Equations

Nonlinear Equations

Notation: $\mathbf{F} : R^N \rightarrow R^N$

$$\mathbf{F}(\mathbf{x}) = \begin{pmatrix} f_1(\mathbf{x}) \\ f_2(\mathbf{x}) \\ \vdots \\ f_N(\mathbf{x}) \end{pmatrix}$$

Two kinds of problems:

- Fixed point: Solve $\mathbf{x} = \mathbf{M}(\mathbf{x})$
- Root finding: Solve $\mathbf{F}(\mathbf{x}) = 0$.

Nonlinear Picard Iteration

Nonlinear Picard (Richardson, fixed point) iteration is

$$\mathbf{x}_{n+1} = \mathbf{M}(\mathbf{x}_n)$$

Similar to the linear case where $\mathbf{M}(\mathbf{x}) = \mathbf{M}\mathbf{x} + \mathbf{b}$.
So what's the analog to $\rho(\mathbf{M}) < 1$?

Contraction Mappings

Definition: $\mathbf{M}$ is a contraction on a closed set $\mathcal{D} \subset R^N$ if

- $\mathbf{M}(\mathbf{x}) \in \mathcal{D}$ if $\mathbf{x} \in \mathcal{D}$
- There is $\alpha \in (0, 1)$ such that

$$\|\mathbf{M}(\mathbf{x}) - \mathbf{M}(\mathbf{y})\| \leq \alpha \|\mathbf{x} - \mathbf{y}\|,$$

  for all $\mathbf{x}, \mathbf{y} \in \mathcal{D}$.

Contraction Mapping Theorem

Theorem: If $\mathbf{M}$ is a contraction on a closed set $\mathcal{D} \subset R^N$ then

- There is a unique solution $\mathbf{x}^* \in \mathcal{D}$ to $\mathbf{x} = \mathbf{M}(\mathbf{x})$.
- If $\mathbf{x}_0 \in \mathcal{D}$ then the Picard iteration converges to $\mathbf{x}^*$.

Proof of Uniqueness

Assume that $\mathbf{x} = \mathbf{M}(\mathbf{x})$ and $\mathbf{y} = \mathbf{M}(\mathbf{y})$, then

$$\|\mathbf{x} - \mathbf{y}\| = \|\mathbf{M}(\mathbf{x}) - \mathbf{M}(\mathbf{y})\| \leq \alpha \|\mathbf{x} - \mathbf{y}\|$$

which implies that $\|\mathbf{x} - \mathbf{y}\| = 0$ since $0 < \alpha < 1$.

Proof of convergence: I

We will prove that the sequence $\{\mathbf{x}_n\}$ is a Cauchy sequence. That means that

$$\lim_{m,n\to\infty} \|\mathbf{x}_n - \mathbf{x}_{m+n}\| \to 0.$$

It's a theorem that Cauchy sequences converge, to that will do it. Since $\mathbf{x}_n = \mathbf{M}(\mathbf{x}_{n-1})$ we have

$$\begin{aligned} \|\mathbf{x}_n - \mathbf{x}_{n+m}\| &= \|\mathbf{M}(\mathbf{x}_{n-1}) - \mathbf{M}(\mathbf{x}_{n+m-1})\| \le \alpha\|\mathbf{x}_{n-1} - \mathbf{x}_{n+m-1}\| \\ &\le \alpha^2\|\mathbf{x}_{n-2} - \mathbf{x}_{n+m-2}\| \le \cdots \le \alpha^n\|\mathbf{x}_0 - \mathbf{x}_m\| \end{aligned}$$

Proof of convergence: II

We're almost there. Note that

$$\mathbf{x}_0 - \mathbf{x}_m = \sum_{l=0}^{m-1} \mathbf{x}_l - \mathbf{x}_{l+1},$$

so

$$\begin{aligned} \|\mathbf{x}_0 - \mathbf{x}_m\| &\leq \textstyle\sum_{l=0}^{m-1} \alpha^l \|\mathbf{x}_0 - \mathbf{x}_1\| \\ &\leq \|\mathbf{x}_0 - \mathbf{x}_1\| \textstyle\sum_{l=0}^{\infty} \alpha^l = \frac{\|\mathbf{x}_0 - \mathbf{x}_1\|}{1-\alpha}. \end{aligned}$$

So

$$\|\mathbf{x}_n - \mathbf{x}_{n+m}\| \leq \frac{\alpha^n \|\mathbf{x}_0 - \mathbf{x}_1\|}{1-\alpha} \to 0$$

as $m, n \to \infty$.

Similarities to linear problems

- It doesn't matter what $\|\cdot\|$ is, or even if you know.
- You're not storing Krylov vectors for Picard.
- Small α means it's a winner.
  Future multigridders be happy.

Nonlinear Equations and Newton's Method

Scalar case from calculus

$$x_+ = x_c - \frac{f(x_c)}{f'(x_c)}$$

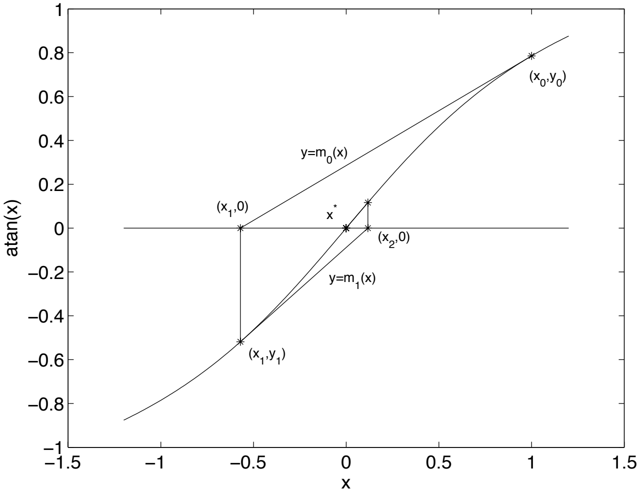
where x_c is the current point and x_+ is the new point

Interpretation: build the **local linear model** of f (first two terms of it's Taylor series) about x_c

$$m_c(x) = f(x_c) + f'(x_c)(x - x_c)$$

and solve $m_c(x) = 0$.

$f(x) = \arctan(x)$


1
0.8
0.6
0.4
0.2
0
−0.2
−0.4
−0.6
−0.8
−1
−1.5
−1
−0.5
0
0.5
1
1.5
x
atan(x)
(x0,y0)
y=m0(x)
(x1,0)
x*
(x2,0)
y=m1(x)
(x1,y1)

Notation

Objective: find a solution of

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

where $\mathbf{F} : R^N \rightarrow R^N$.
We write $\mathbf{F} = (f_1, \dots, f_N)^T$. The Jacobian matrix $\mathbf{F}'$ is

$$(\mathbf{F}')_{ij} = \partial f_i / \partial x_j$$

Example

If

$$\mathbf{F}\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x^2 \sin(y) \\ y^2 e^x \end{pmatrix}$$

then

$$\mathbf{F}'\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 2x \sin(y) & x^2 \cos(y) \\ y^2 e^x & 2ye^x \end{pmatrix} = \begin{pmatrix} \partial \mathbf{F}/\partial x & \partial \mathbf{F}/\partial y \end{pmatrix}.$$

Newton's Method

Transition from current point x_c to new one x_+.

$$\mathbf{x}_+ = x_c - \mathbf{F}'(\mathbf{x}_c)^{-1}\mathbf{F}(\mathbf{x}_c).$$

Interpretation: $\mathbf{x}_+$ is the root of the local linear model at $\mathbf{x}_c$

$$\mathbf{M}_c(\mathbf{x}) = \mathbf{F}(\mathbf{x}_c) + \mathbf{F}'(\mathbf{x}_c)(\mathbf{x} - \mathbf{x}_c)$$

Implementation of a Newton Iteration

Evaluate $\mathbf{F}(\mathbf{x}_c)$; terminate?
Solve $\mathbf{F}'(\mathbf{x}_c)\mathbf{s} = -\mathbf{F}(\mathbf{x}_c)$
$\mathbf{x}_+ = \mathbf{x}_c + \mathbf{s}$

You can write this at home!

Formulations of Newton differ in the way they solve for s.

Convergence Theory for Exact Linear Solves

Standard Assumptions (SA):

- $\mathbf{F}(\mathbf{x}^*) = 0$
- $\mathbf{F}'(\mathbf{x}^*)$ is nonsingular.
- $\mathbf{F}'(\mathbf{x})$ is Lipschitz continuous with Lipschitz constant γ

$$\|\mathbf{F}'(\mathbf{x}) - \mathbf{F}'(\mathbf{y})\| \leq \gamma \|\mathbf{x} - \mathbf{y}\|$$

What do SA mean in English?

- $\mathbf{F}(\mathbf{x}^*) = 0$ means we not trying to do the impossible, which rarely works.
- $\mathbf{F}'(\mathbf{x}^*)$ is nonsingular means we're avoiding a zero divide.
- Lipschitz continuity of $\mathbf{F}'$ means the linear model does what we need.

Lipschitz continuity and the linear model

The Taylor expansion of $\mathbf{F}(\mathbf{x})$ about $\mathbf{z}$ looks like

$$\mathbf{F}(\mathbf{x}) = \mathbf{F}(\mathbf{z}) + \mathbf{F}'(\mathbf{z})(\mathbf{x} - \mathbf{z}) + \dots$$

- What's the next term?
- How do you estimate the error?

Fundamental Theorem of Calculus and the Linear Model

FUNDY says "integrate the derivative and get the function back"

$$\int_a^b f(t)\ dt = f(b) - f(a).$$

Let $\mathbf{d} = \mathbf{x} - \mathbf{z}$, then

$$\int_0^1 \mathbf{F}'(\mathbf{z} + t\mathbf{d})\mathbf{d}\ \ dt = \mathbf{F}(\mathbf{x}) - \mathbf{F}(\mathbf{z}).$$

If $\mathbf{g}(t) = \mathbf{F}(\mathbf{z} + t\mathbf{d})$, then $\mathbf{g}(0) = \mathbf{F}(\mathbf{z}), \mathbf{g}(1) = \mathbf{F}(\mathbf{x})$ and

$$\mathbf{g}'(t) = \frac{d}{dt}\mathbf{F}(\mathbf{z} + t\mathbf{d}) = \mathbf{F}'(\mathbf{z} + t\mathbf{d})\mathbf{d}$$

Check this out with a hand calculation!

Using Lipschitz continuity

Begin with

$$\mathbf{F}(\mathbf{x}) = \mathbf{F}(\mathbf{z}) + \int_0^1 \mathbf{F}'(\mathbf{z} + t\mathbf{d})\mathbf{d},$$

add and subtract $\mathbf{F}'(\mathbf{z})\mathbf{d}$ to the right side

$$\mathbf{F}(\mathbf{x}) = \mathbf{F}(\mathbf{z}) + \mathbf{F}'(\mathbf{z})\mathbf{d} + \mathbf{E},$$

where

$$\mathbf{E} = \int_0^1 (\mathbf{F}'(\mathbf{z} + t\mathbf{d}) - \mathbf{F}'(\mathbf{z}))\mathbf{d}\ dt$$

So how big is $\mathbf{E}$?

Taylor is good

$$
\begin{aligned}
\|E\| &\leq \int_0^1 \|(\mathbf{F}'(\mathbf{z}+t\mathbf{d}) - \mathbf{F}'(\mathbf{z}))\mathbf{d}\| \; dt \\
&\leq \int_0^1 \|\mathbf{F}'(\mathbf{z}+t\mathbf{d}) - \mathbf{F}'(\mathbf{z})\| \; dt \|\mathbf{d}\| \\
&\leq \int_0^1 \gamma t \;\; dt \|\mathbf{d}\|^2 = \frac{\gamma\|\mathbf{d}\|^2}{2}
\end{aligned}
$$

Convergence Theory: I

Theorem: SA and

$$\|\mathbf{e}_c\| \equiv \|\mathbf{x}_c - \mathbf{x}^*\| \leq \frac{\|\mathbf{F}'(\mathbf{x}^*)^{-1}\|^{-1}}{2\gamma}$$

then $\mathbf{F}'(\mathbf{x}_c)$ is nonsingular and $\|\mathbf{F}'(\mathbf{x}_c)^{-1}\| \leq 2\|\mathbf{F}'(\mathbf{x}^*)^{-1}\|$
Proof: Lipschitz continuity implies that

$$\|\mathbf{F}'(\mathbf{x}_c) - \mathbf{F}'(\mathbf{x}^*)\| \leq \gamma\|\mathbf{e}_c\| \leq \frac{\|\mathbf{F}'(\mathbf{x}^*)^{-1}\|^{-1}}{2}$$

and so ...

Convergence Theory: II

$$\|\mathbf{I} - \mathbf{F}'(\mathbf{x}^*)^{-1}\mathbf{F}'(\mathbf{x}_c)\| \le \|\mathbf{F}'(\mathbf{x}^*)^{-1}\|\|\mathbf{F}'(\mathbf{x}_c) - \mathbf{F}'(\mathbf{x}^*)\| \le 1/2$$

so $\mathbf{F}'(\mathbf{x}^*)^{-1}$ is an approximate inverse of $\mathbf{F}'(\mathbf{x}_c)$, and

$$\|\mathbf{F}'(\mathbf{x}_c)^{-1}\| \le 2\|\mathbf{F}'(\mathbf{x}^*)^{-1}\|.$$

Local Convergence Theory: III

Recall the fundamental theorem of calculus:

$$\mathbf{F}(\mathbf{x}) - \mathbf{F}(\mathbf{x}^*) = \int_0^1 \mathbf{F}'(\mathbf{x}^* + t(\mathbf{x} - x^*))(\mathbf{x} - x^*)\, dt.$$

$\mathbf{F}(\mathbf{x}^*) = 0$, so let $x = x_c$ and ...

$$\begin{aligned} \mathbf{F}(\mathbf{x}_c) &= \int_0^1 \mathbf{F}'(\mathbf{x}^* + t\mathbf{e}_c)\mathbf{e}_c\, dt \\ &= \mathbf{F}'(\mathbf{x}_c)\mathbf{e}_c + \int_0^1 (\mathbf{F}'(\mathbf{x}^* + t\mathbf{e}_c) - \mathbf{F}'(\mathbf{x}_c))\mathbf{e}_c\, dt \end{aligned}$$

Local Convergence Theory: IV

We are done since

$$\begin{aligned}\mathbf{e}_+ &= \mathbf{e}_c - \mathbf{F}'(\mathbf{x}_c)^{-1}(\mathbf{F}'(\mathbf{x}_c)\mathbf{e}_c + \int_0^1 (\mathbf{F}'(\mathbf{x}^* + t\mathbf{e}_c) - \mathbf{F}'(\mathbf{x}_c))\mathbf{e}_c\, dt) \\ &= -\mathbf{F}'(\mathbf{x}_c)^{-1}(\int_0^1 (\mathbf{F}'(\mathbf{x}^* + t\mathbf{e}_c) - \mathbf{F}'(\mathbf{x}_c))\mathbf{e}_c\, dt)\end{aligned}$$

So,

$$\|\mathbf{e}_+\| \le \frac{\|\mathbf{F}'(\mathbf{x}^*)^{-1}\|\gamma}{2}\|\mathbf{e}_c\|^2 \le \|\mathbf{e}_c\|/2.$$

Bottom Line

SA and good data ($\|\mathbf{e}_0\|$ small) imply that

- $\mathbf{x}_n \to \mathbf{x}^*$
- Convergence is q-quadratic ($\|e_+\| = O(\|e_c\|^2)$)

Things change if initial iterate is not close to $\mathbf{x}^*$ or you use an iterative method to compute $\mathbf{s}$.

Simple Newton code for scalar equations

```
function [x,hist]=newton_scalar(f,fp,x,atol,rtol
    )
fc=feval(f,x);
res0=abs(fc);
res=res0
hist=res;
maxit=20;
itc=0;
while res > atol + rtol*res0 && itc < maxit
    fpc=feval(fp,x);
    x=x - fc/fpc;
    fc=feval(f,x);
    res=abs(fc)
    hist=[hist,res];
    itc=itc+1;
end
```

$f(x) = x - e^{-x}\cos(x)$, $\mathbf{x}_0 = 2$

simple_demo.m

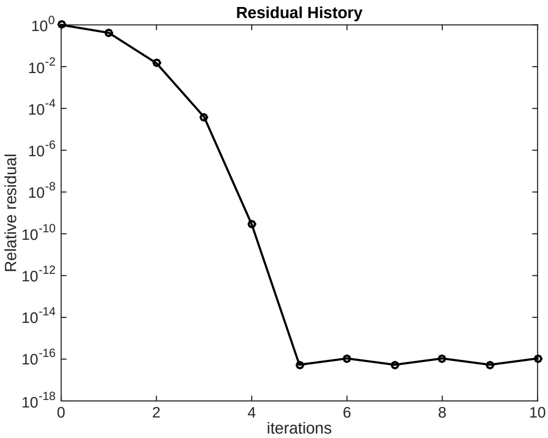
```
[x,hist]=newton_scalar(@fdemo,@fdemop,x,1.d
    -20,1.d-20);

function y=fdemo(x)
y=x - exp(-x)*cos(x);

function y=fdemop(x)
y=1 + exp(-x)*(sin(x)+cos(x));
```

$f(x) = x - e^{-x}\cos(x)$, $\mathbf{x}_0 = 2$

Tabular Residual History: $atol = rtol = 10^{-20}$

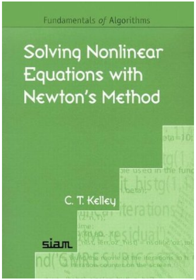
n	$\lvert f(x_n)\rvert / \lvert f(x_0)\rvert$
0	1e+00
1	4e-01
2	1e-02
3	4e-05
4	3e-10
5	5e-17
6	1e-16
7	5e-17
8	1e-16
9	5e-17
10	1e-16

Graphical Residual History



Software Book

C. T. KELLEY,
Solving Nonlinear Equations with Newton's Method, SIAM, 2003.
http://catalog.lib.ncsu.edu/record/NCSU2488300



Codes

- Latest version of Red book codes (Green book)
  - http://www4.ncsu.edu/~ctk/newtony.html
  - You should use `nsold.m` when using direct linear solvers.
- `knl.m` Better for iterative linear solvers.
  - http://www4.ncsu.edu/~ctk/knl.html

Implementation

newton($\mathbf{x}, \mathbf{F}, \tau_a, \tau_r, maxit$)

$itc = 0$

evaluate $\mathbf{F}(\mathbf{x})$; $\tau \leftarrow \tau_r \|\mathbf{F}(\mathbf{x})\| + \tau_a$.

while $\|\mathbf{F}(\mathbf{x})\| > \tau$ and $itc < maxit$ **do**

compute $\mathbf{F}'(\mathbf{x})$; factor $\mathbf{F}'(\mathbf{x}) = \mathbf{LU}$

solve $\mathbf{LUs} = -\mathbf{F}(\mathbf{x})$

$\mathbf{x} \leftarrow \mathbf{x} + \mathbf{s}$

evaluate $\mathbf{F}(\mathbf{x})$

$itc \leftarrow itc + 1$

end while

Cost Analysis

For every iteration you must,

- evaluate $\mathbf{F}$,
- evaluate $\mathbf{F}'$,
- factor $\mathbf{F}' = \mathbf{LU}$.

The general idea is that $\mathbf{F}'$ and the factorization dominate the work.

Cost of LU is $N^3/3 + O(N^2)$. As for the evaluation ...

Evaluation of $\mathbf{F}'$

- Most accurate way: analytic, but
  - not always possible
  - takes human effort
- Easy (for you) way: forward difference
  - Approximate jth column of $\mathbf{F}'(\mathbf{x})$ by

$$(\mathbf{F}(\mathbf{x} + h\mathbf{e}_j) - \mathbf{F}(\mathbf{x}))/h$$

    to get $\nabla_h \mathbf{F}(\mathbf{x})$
  - Cost: N extra calls to $\mathbf{F}$

Forward difference derivative

What's h? The scalar case illustrates the issues.

- Suppose you get $f(x) + \epsilon(x)$ when you call $f(x)$ and

$$\|\epsilon(x)\| \le \bar{\epsilon} \text{ for all } x.$$

- Then when you evaluate a forward difference derivative you really get

$$\frac{f(x+h) - f(x)}{h} + \frac{\epsilon(x+h) - \epsilon(x)}{h} = f'(x) + O\left(h^2 + \frac{\bar{\epsilon}}{h}\right).$$

So you don't want to let $h \to 0$.

What's h

Minimize the error term as a function of h by setting

$$\frac{d}{dh}\left(h^2 + \frac{\bar{\epsilon}}{h}\right) = 0$$

and find that $h = \sqrt{\bar{\epsilon}}$.
So if ϵ is machine roundoff, you're perturbing the low-order bits.
BUT, if x is very large/small, you may miss too much, for example

$$\epsilon = 10^{-16}, h = 10^{-8}, x = 10^{9}$$

leads to $x + h = h$ in floating point arithmetic. The fix . . .

$$h = \sqrt{\epsilon}|x|.$$

$\nabla_h \mathbf{F}$

$h = \sqrt{\bar{\epsilon}}$, jth column of $\nabla_h \mathbf{F}(x)$ is

$$\frac{\mathbf{F}(\mathbf{x} + h\|\mathbf{x}\|\mathbf{e}_j) - \mathbf{F}(\mathbf{x})}{h\|\mathbf{x}\|}$$

for $\mathbf{x} \neq 0$.

Can we avoid $\mathbf{F}'$ and the factorization?

Yup. Why not take them out of the loop and get the **chord method**

chord$(\mathbf{x}, \mathbf{F}, \tau_a, \tau_r, maxit)$

$itc = 0$

evaluate $\mathbf{F}(x)$; $\tau \leftarrow \tau_r \|\mathbf{F}(x)\| + \tau_a$.

evaluate $\mathbf{F}'(x)$; factor $\mathbf{F}'(x) = LU$

while $\|\mathbf{F}(x)\| > \tau$ and $itc < maxit$ **do**

solve $LU\mathbf{s} = -\mathbf{F}(x)$

$\mathbf{x} \leftarrow \mathbf{x} + \mathbf{s}$

evaluate $\mathbf{F}(\mathbf{x})$

$itc \leftarrow itc + 1$

end while

What's changed?

Chord Method:

$$\mathbf{x}_+ = \mathbf{x}_c - \mathbf{F}'(\mathbf{x}_0)^{-1}\mathbf{F}(\mathbf{x}_c)$$

All we did was replace $\mathbf{F}'(\mathbf{x}_c)$ with $\mathbf{F}'(\mathbf{x}_0)$.
There's a story here. So far we have

- Swapped $\mathbf{F}'$ for $\nabla_h \mathbf{F}$
- Swapped $\mathbf{F}'(\mathbf{x}_c)$ for $\mathbf{F}'(\mathbf{x}_0)$
- Swapped $\mathbf{F}$ for $\mathbf{F} + \epsilon$

We do lots of things to cheat on Newton. This is a partial list.

Errors in $\mathbf{F}$ and $\mathbf{F}'$

Suppose that your approximation to Newton looks like

$$\mathbf{x}_+ = \mathbf{x}_c + \mathbf{s}$$

where

$$\|\mathbf{J}_c\mathbf{s} + (\mathbf{F}(x_c) + \epsilon(\mathbf{x}_c))\| \leq \eta_c \|\mathbf{F}(x_c) + \epsilon(\mathbf{x}_c)\|$$

and

$$\|\mathbf{J}_c - \mathbf{F}'(\mathbf{x}_c)\| \leq \Delta_c$$

What we have here is . . .

Errors in $\mathbf{F}$ and $\mathbf{F}'$: II

- Wrong $\mathbf{F}$: $\mathbf{F} + \epsilon$
- Wrong equation for the Newton step $\mathbf{J}_c \mathbf{s} = -(\mathbf{F} + \epsilon)$
- Poor solution for the wrong equation, leaving

$$\|\mathbf{J}_c \mathbf{s} + (\mathbf{F}(x_c) + \epsilon(\mathbf{x}_c))\| \leq \eta_c \|\mathbf{F}(x_c) + \epsilon(\mathbf{x}_c)\|$$

Even with all those bad things ...

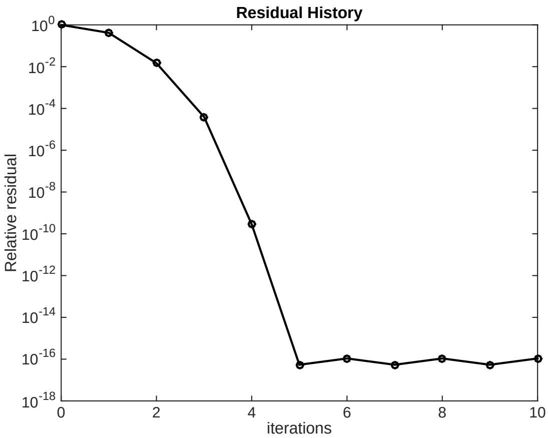
Errors in $\mathbf{F}$ and $\mathbf{F}'$: III

Cheating Theorem: SA and good data imply that ...

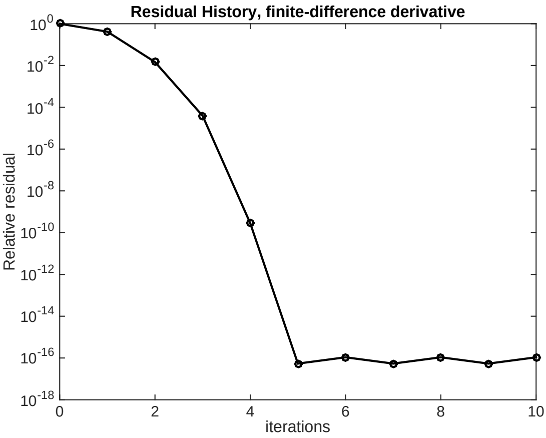
$$\|\mathbf{e}_+\| = O(\|\mathbf{e}_c\|^2 + (\|\eta_c\| + \|\Delta_c\|)\|\mathbf{e}_c\| + \|\epsilon(\mathbf{x}_c)\|)$$

- Analytic $\mathbf{F}$ and $\mathbf{F}'$: errors like floating point roundoff
  Ultimate accuracy is limited by errors in $\mathbf{F}$.
- Finite difference Jacobian: $\Delta = O(h)$, square root of roundoff
- Chord method: $\Delta = O(\|e_0\|)$
- Any idea what η might be for?

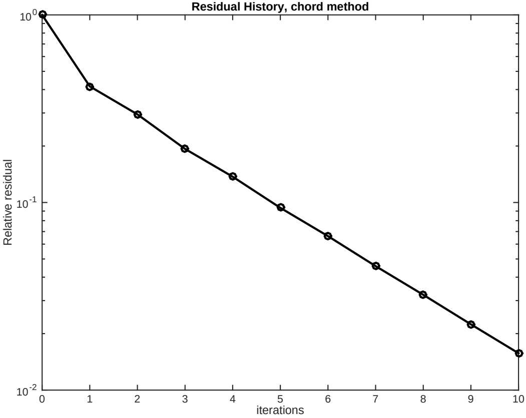
Remember this? Can you explain the stagnation?



Same example with forward difference derivative.



Chord method



Forward difference derivatives (usually) do no harm.

Assume:

- $\|\epsilon\| \le \bar{\epsilon}$ is floating point roundoff
- $h = \sqrt{\bar{\epsilon}}$
- All other errors are $O(\bar{\epsilon})$

Then the theorem says

$$\|e_+\| = O(\|e_c\|^2 + h\|e_c\| + h^2)$$

so you can't tell if it's a difference derivative or the real one.

Chord Method

Assume:

- $\|\epsilon\| \leq \bar{\epsilon}$ is floating point roundoff
- $h = \sqrt{\bar{\epsilon}}$
- $\Delta = O(\|e_0\|)$
- All other errors are $O(\bar{\epsilon})$

Then the theorem says

$$\|e_+\| = O(\|e_c\|^2 + \|e_0\|\|e_c\| + h^2)$$

which is what you observe.

Taxonomy of Convergence

- q-quadratic: $\mathbf{x}_n \to \mathbf{x}^*$ and $\|\mathbf{e}_+\| = O(\|\mathbf{e}_c\|^2)$
  Newton
- q-linear: $\|\mathbf{e}_+\| \le \alpha \|\mathbf{e}_c\|$ for some $\alpha \in [0, 1)$
  Chord
- q-superlinear: $\|\mathbf{e}_{n+1}\| / \|\mathbf{e}_n\| \to 0$
  Stay tuned and do your homework.

Computing Jacobians by Inspection

$\mathbf{F}'(\mathbf{x})$ is the linear operator for which

$$\frac{d}{d\epsilon}\mathbf{F}(\mathbf{x}+\epsilon\mathbf{w})\Big|_{\epsilon=0} = \mathbf{F}'(\mathbf{x})\mathbf{w} \text{ for all } \mathbf{w}\ .$$

Examples:

- Constant functions: $\mathbf{F}(\mathbf{x}) = \mathbf{z}$, $\mathbf{F}'(\mathbf{x}) = 0$.
- Linear functions: $\mathbf{F}(\mathbf{x}) = \mathbf{A}\mathbf{x} - \mathbf{b}$, $\mathbf{F}'(\mathbf{x}) = \mathbf{A}$
  So Newton's method converges in one iteration.

Substitution Operators

Given $\phi : R \to R$ define

$$\mathbf{\Phi}(\mathbf{x}) = \begin{pmatrix} \phi(x_1) \\ \phi(x_2) \\ \vdots \\ \phi(x_N) \end{pmatrix}$$

Use the formula to see that

$$\mathbf{\Phi}'(\mathbf{x}) = \mathrm{diag}(\phi'(x_1), \phi'(x_2), \dots, \phi'(x_N)).$$

Chain Rule

If $\mathbf{F}(\mathbf{x}) = \mathbf{\Phi}(\mathbf{G}(\mathbf{x}))$ then

$$\mathbf{F}'(\mathbf{x}) = \mathbf{\Phi}'(\mathbf{G}(\mathbf{x}))\mathbf{G}'(\mathbf{x})$$

Example: $\mathbf{G}(\mathbf{x}) = \mathbf{A}\mathbf{x} - \mathbf{b}$ (linear) and $\mathbf{\Phi}$ is a substitution operator,

$$\mathbf{F}(\mathbf{x}) = \mathbf{\Phi}(\mathbf{A}\mathbf{x} + \mathbf{b}) \text{ implies } \mathbf{F}'(\mathbf{x}) = \mathbf{\Phi}'(\mathbf{A}\mathbf{x} + \mathbf{b})\mathbf{A}$$

is the product of a diagonal matrix and the matrix $\mathbf{A}$.

Boundary Value Problem

$$F(u)(x) = -u''(x) + \sin(u(x)) - 1, \;\; u(0) = u(1) = 0$$

Here's the sum of a linear operator

$$\frac{d^2}{dx^2} \text{ and boundary conditions } u(0) = u(1) = 0$$

with a substitution operator $u \leftarrow \sin(u)$.
So the Fréchet derivative is the linear operator A where

$$Aw(x) = -w'' + \cos(u(x))w(x) \text{ bc } w(0) = w(1) = 0.$$

Discrete Version

$$\mathbf{F}(\mathbf{u}) = D_2\mathbf{u} + \mathbf{\Phi}(\mathbf{u}) - 1$$

where $\phi(u) = \sin(u)$. So,

$$\mathbf{F}'(\mathbf{u}) = D_2 + \text{diag}(\cos(u_1), \ldots \cos(u_2))$$

is tridiagonal and

$$\mathbf{F}'(\mathbf{u})_{i,i} = \frac{2}{h^2} + \cos(u_i), \mathbf{F}'(\mathbf{u})_{i,i\pm1} = \frac{-1}{h^2}.$$

Some Matlab

- For MA 580 you should write your own simple Newton code.
- Using my code `nsold.m` is ok, but too fancy.
- Save time by writing your own.

What does Newton need?

- Code for $\mathbf{F}$, $\mathbf{F}'$
- Algorithmic parameters
- Way to pass precomputed stuff.

Putting this example together

$$\mathbf{F}(\mathbf{u}) = D2\mathbf{u} + \sin(\mathbf{u}) - 1$$

So precompute $D2$ as we've done before. Then

```
function f=bvp(u,D2)
n=length(u);
f = D2*u + sin(u) - ones(n,1);
```

The Jacobian

$$\mathbf{F}'(\mathbf{u}) = D2 + \mathrm{diag}(\cos(\mathbf{u}))$$

```
function fp=bvpprime(u,D2)
n=length(u);
c=cos(u);
C = spdiags([c],0:0,n,n);
fp = D2 + C;
```

Roll-your-own in-line Newton: I

See bvp_demo.m in Nonlinear Solver Examples on Moodle.
Set up the iteration

```
h=1/(n+1); x=h:h:1-h; x=x';
e = ones(n,1);
D2 = spdiags([e -2*e e], -1:1, n, n); D2=D
   2/(h*h);
u0=sin(10*pi*x)+10*sin(pi*x);
res0=bvp(u0,D2); nres0=norm(res0);
u=u0; res=res0; nres=nres0;
```

Roll-your-own in-line Newton: II

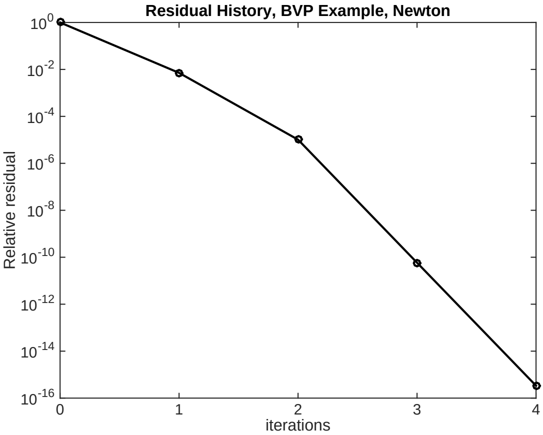
Newton time!

```
atol=1.d-12; rtol=1.d-12; maxit=20;
itc=0; ihist=nres;
while nres > (atol + rtol*nres0)  && itc <
   maxit
     fp=bvpprime(u,D2); [L,U]=lu(fp);
     step = -U\(L\res); u=u+step;
     res=bvp(u,D2); nres=norm(res);
     ihist=[ihist,nres]; itc=itc+1;
end
```

How'd I do? Herewith, some residuals.

```
6.9591e+03
4.9699e+01
7.1040e-02
4.0831e-07
2.3554e-12
```

and a plot.


Residual History, BVP Example, Newton
Relative residual
iterations
10^0
10^-2
10^-4
10^-6
10^-8
10^-10
10^-12
10^-14
10^-16
0
1
2
3
4

Integro-Differential Equation

$$F(u)(x) = -u''(x) + \int_0^1 \cos(xu(y))\, dy$$

Use the formula to see that

$$F'(u)w = -w''(x) - \int_0^1 x \sin(xu(y))w(y)\, dy$$

And you get a similar story for the discrete problem.

Discretization

$$\mathbf{F}(\mathbf{u})_i = (D_2 u)_i + \sum_{j=1}^{N} \cos(x_i u_j) w_j$$

where x_i is the ith interior grid point and $w_j = h = 1/(N+1)$ for all j.
You should verify that, in Matlab,

```
f = D2*u + cos(x*u')*w;
```

and

```
jac = D2-diag(x)*sin(x*u')*diag(w);
```

Passing precomputed data around

Precompute `D2, x, w` and then

```
R_data=struct('D2',D2,'w',w,'x',x);

function f=ieq(u,R_data)
x=R_data.x; D2=R_data.D2; w=R_data.w;
f = D2*u + cos(x*u')*w;

function jac=ieqprime(u,R_data);
x=R_data.x; D2=R_data.D2; w=R_data.w;
jac=D2-diag(x)*sin(x*u')*diag(w);
```

See `ieq_newton.m` in the examples.

Chandrasekhar H-Equation

$$\mathbf{F}(H)(\mu) = H(\mu) - \left(1 - \frac{c}{2}\int_0^1 \frac{\mu H(\nu)\, d\nu}{\mu + \nu}\right)^{-1} = 0.$$

- $0 \le \mu \le 1$; $0 \le c \le 1$
- Unknown $H \in C[0,1]$
- Two solutions for $0 < c < 1$. Unique for $c = 0, 1$.

Discretization

We will approximate the integrals by the composite midpoint rule.

$$\int_0^1 f(\mu)\, d\mu \approx \frac{1}{N} \sum_{j=1}^{N} f(\mu_j)$$

where $\mu_i = (i - 1/2)/N$ for $1 \leq i \leq N$.
Discrete equation in R^N:

$$\mathbf{F}(\mathbf{h})_i = h_i - \left(1 - \frac{c}{2N} \sum_{j=1}^{N} \frac{\mu_i h_j}{\mu_i + \mu_j}\right)^{-1}.$$

Midpoint rule accuracy?

Second-order in h for smooth functions. However,

$$
\begin{aligned}
H'(\mu) &= \left(1 - \frac{c}{2}\int_0^1 \frac{\mu H(\nu)\, d\nu}{\mu+\nu}\right)^{-2} \int_0^1 H(\nu) \frac{d}{d\mu}\left(\frac{\mu}{\mu+\nu}\right)\, d\nu \\
&= H^2(\mu) \int_0^1 H(\nu) \frac{\nu}{(\mu+\nu)^2}\, d\nu.
\end{aligned}
$$

So $H'(0) = \infty$!!

Computing the Jacobian

F is the sum of

- a linear map **I**
- a composition of
  - a substitution operator with $\phi(x) = 1/x$ with
  - a linear operator $1 - \mathbf{Ah}$, where

$$\mathbf{A}_{ij} = \frac{c\mu_i}{2N(\mu_i + \mu_j)}$$

Compact Form

Precompute **A**, then **F**(**h**) can be rapidly evaluated as

$$\mathbf{F}(\mathbf{h})_i = x_i - (1 - (\mathbf{A}\mathbf{h})_i)^{-1},$$

and

$$\mathbf{F}(\mathbf{h})_{ij} = \delta_{ij} - \frac{\mathbf{A}_{ij}}{(1 - (\mathbf{A}\mathbf{h})_i)^2}.$$

The Jacobian-vector product is given by

$$(\mathbf{F}'(\mathbf{h})\mathbf{v})_i = v_i - \frac{(\mathbf{A}\mathbf{v})_i}{(1 - (\mathbf{A}\mathbf{h})_i)^2}.$$

So, after you compute $\mathbf{F}(\mathbf{h})$, $\mathbf{F}'(\mathbf{h})\mathbf{v}$ takes very little work.

Analytic is better than finite differences

in this example because

- Cost of function is $O(N^2)$ so
  Cost of forward difference Jacobian is $O(N^3)$
- Cost of analytic Jacobian is $O(N^2)$
- Human time for analytic Jacobian is small.

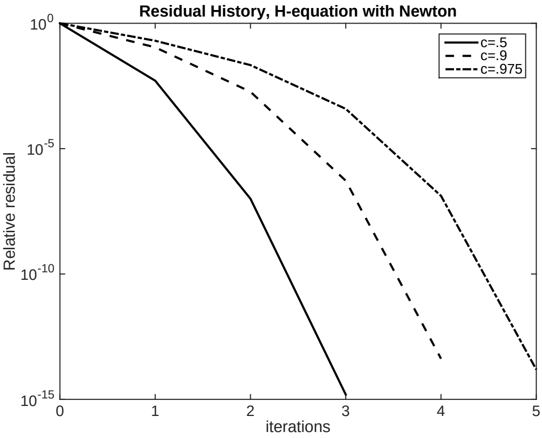
Some answers

- Look 'em up at the source
  - S. Chandrasekhar, Radiative Transfer, Dover, New York, 1960, pg 125.
  - "*Using the method described, Mrs. Frances H. Breen and the writer have evaluated over forty H-functions in the context of various problems.*" (pg 124)
- or trust me . . .

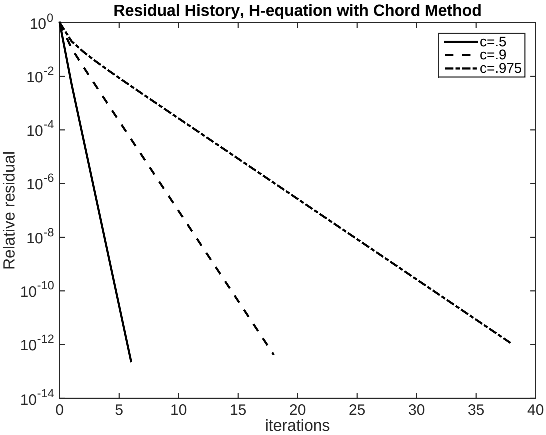
Values of $H(\mu)$: $N = 500$

μ/c	.5	.9	.975	1.0
0.0	1.0000	1.0000	1.0000	1.0000
0.1	1.0724	1.1721	1.2111	1.2473
0.2	1.1135	1.2914	1.3703	1.4504
0.3	1.1439	1.3914	1.5117	1.6425
0.4	1.1680	1.4785	1.6414	1.8293
0.5	1.1877	1.5560	1.7621	2.0128
0.6	1.2043	1.6259	1.8753	2.1941
0.7	1.2186	1.6893	1.9822	2.3740
0.8	1.2309	1.7474	2.0834	2.5527
0.9	1.2417	1.8008	2.1795	2.7306
1.0	1.2513	1.8501	2.2710	2.9078

Residual History Plots: $H_0 \equiv 1$



Residual History Plots: $H_0 \equiv 1$



Newton vs Chord: timings on MacBook Air

- Residual histories are independent of N, but not of c.
- For $N = 5000$ and $c = .975$ the timings for a solve are
  - Newton: 14 secs, 5 iterations
  - Chord: 8 secs, 39 iterations
- For $N = 5000$ and $c = .5$ the timings for a solve are
  - Newton: 10 secs, 3 iterations
  - Chord: 3.5 secs, 6 iterations