Numerical Analysis and Computing

Lecture Notes #3 — Solutions of Equations in One Variable: Fixed Point Iteration; Root Finding; Error Analysis for Iterative Methods

Joe Mahaffy,
⟨mahaffy@math.sdsu.edu⟩

Department of Mathematics
Dynamical Systems Group
Computational Sciences Research Center
San Diego State University
San Diego, CA 92182-7720

http://www-rohan.sdsu.edu/∼jmahaffy

Spring 2010

Outline

Quick Recap

Last time we looked at the **method of bisection** for finding the root of the equation $f(x) = 0$.

Now, we are take a short detour in order to explore how

Root finding: $\quad f(x) = 0,$

is related to

Fixed point iteration: $\quad f(p) = p.$

Fixed Point Iteration ⇔ Root Finding

If $f(p) = p$, then we say that p is a **fixed point** of the function $f(x)$. We note a strong relation between root finding and finding fixed points:

To convert a fixed-point problem

$$g(x) = x,$$

to a root finding problem, define

$$f(x) = g(x) - x, \quad \text{and look for roots of} \quad f(x) = 0.$$

To convert a root finding problem

$$f(x) = 0,$$

to a fixed point problem, define

$$g(x) = f(x) + x, \quad \text{and look for fixed points} \quad g(x) = x.$$

Why Consider Fixed Point Iteration?

If fixed point iterations are (in some sense) equivalent to root finding, why not just stick to root finding???

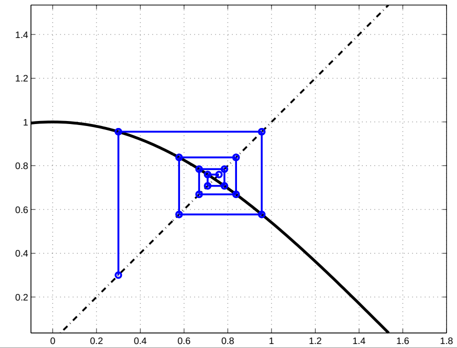
1. Sometimes easier to analyze.
2. What we learn from the analysis will help us find good root finding strategies.
3. Fixed point iterations pose some "cute" problems by themselves.

Example: The Bored Student Fixed Point

A "famous" fixed point is $p = 0.73908513321516$ (radians), *i.e.* the number you get by repeatedly hitting $\boxed{\cos}$ on a calculator. This number solves the fixed point equation:

$$\cos(p) = p$$

With a starting value of $p = 0.3$ we get:



Iteration #7:
$p = 0.71278594551835$,
$\cos(p) = 0.75654296195845$
$p = 0.75654296195845$

When Does Fixed-Point Iteration Converge?

The following theorem tells us when a fixed point exists:

Theorem (Convergence of Fixed Point Iteration)

a. *If $f \in C[a,b]$ and $f(x) \in [a,b]$, $\forall x \in [a,b]$, then f has a fixed point $p \in [a,b]$. (Brouwer fixed point theorem)*

b. *If, in addition, the derivative $f'(x)$ exists on (a,b) and $|f'(x)| \leq k < 1, \forall x \in (a,b)$, then the fixed point is* ***unique***. *(Contraction Mapping Principle)*

How does this apply to $\cos(x)$???

Note 1: $f \in C[a,b]$ — "f is continuous in the interval $[a,b]$." (input)

Note 2: $f(x) \in [a,b]$ — "f takes values in $[a,b]$." (output)

Proof of the Fixed Point Theorem

a. If $f(a) = a$, or $f(b) = b$, then we are done.

Otherwise, $f(a) > a$ and $f(b) < b$.

We define a new function $h(x) = f(x) - x$.

Since both $f(x)$ and x are continuous, we have $h(x) \in C[a,b]$, further $h(a) > 0$, and $h(b) < 0$ by construction.

Now, the **intermediate value theorem** guarantees $\exists p^* \in (a,b)$: $h(p^*) = 0$.

We have $0 = h(p^*) = f(p^*) - p^*$, or $\mathbf{p^* = f(p^*)}$.

Proof of the Fixed Point Theorem

b. $|f'(x)| \leq k < 1$. Suppose we have two fixed points $p^* \neq q^*$. Without loss of generality we may assume $p^* < q^*$.
The **mean value theorem** tells us $\exists r \in (p^*, q^*)$:

$$f'(r) = \frac{f(p^*) - f(q^*)}{p^* - q^*}$$

Now,

$$\begin{aligned} |p^* - q^*| &= |f(p^*) - f(q^*)| \\ &= |f'(r)| \cdot |p^* - q^*| \\ &\leq k|p^* - q^*| \\ &< |p^* - q^*| \end{aligned}$$

The contradiction $|\mathbf{p}^* - \mathbf{q}^*| < |\mathbf{p}^* - \mathbf{q}^*|$ shows that the supposition $p^* \neq q^*$ is false. Hence, the fixed point is unique.

Root Finding
Improved Algorithms for Root Finding
Error Analysis
Fixed Point Iteration
Detour: — Non-unique Fixed Points...

Convergence of the Fixed Point Sequence

Or, *"how come hitting* $\boxed{\cos}$ *converges???"*

Take a look at the theorem we just proved

— part **(a)** guarantees the existence of a fixed point.

— part **(b)** tells us when the fixed point is unique.

We have no information about **finding** the fixed point!

We need one more theorem — one which guarantees us that we can find the fixed point!

Root Finding
Improved Algorithms for Root Finding
Error Analysis
Fixed Point Iteration
Detour: — Non-unique Fixed Points...

Convergence of the Fixed Point Sequence — Theorem

Suppose both part (a) and part (b) of the previous theorem (here restated) are satisfied:

Theorem (Convergence of Fixed Point Iteration)

a. *If $f \in C[a,b]$ and $f(x) \in [a,b]$, $\forall x \in [a,b]$, then f has a fixed point $p \in [a,b]$. (Brouwer fixed point theorem)*

b. *If, in addition, the derivative $f'(x)$ exists on (a,b) and $|f'(x)| \leq k < 1, \forall x \in (a,b)$, then the fixed point is unique.*

c. *Then, for any number $p_0 \in [a,b]$, the sequence defined by*

$$p_n = f(p_{n-1}), \quad n = 1, 2, \ldots, \infty$$

converges to the unique fixed point $p^* \in [a,b]$.

Root Finding
Improved Algorithms for Root Finding
Error Analysis
Fixed Point Iteration
Detour: — Non-unique Fixed Points...

Convergence of the Fixed Point Sequence — Proof

That's great news! — We can use **any** starting point, and we are guaranteed to find the fixed point.

The proof is straight-forward:

$$\begin{aligned} |p_n - p^*| &= |f(p_{n-1}) - f(p^*)| & \\ &= |f'(r)| \cdot |p_{n-1} - p^*| & \text{by \{MVT\}} \\ &\leq k|p_{n-1} - p^*| & \text{by b} \end{aligned}$$

Since $k < 1$, the distance to the fixed point is shrinking every iteration.

In fact,

$$|p_n - p^*| \leq k^n |p_0 - p^*| \leq k^n \max\{p_0 - a,\ b - p_0\}.$$

Example: $x^3 + 4x^2 - 10 = 0$

The equation $x^3 + 4x^2 - 10 = 0$ has a unique root in the interval $[1, 2]$.

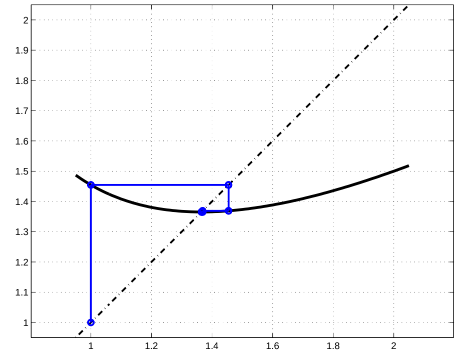
We make a couple attempts at finding the root:

1. Define $g_1(x) = x^3 + 4x^2 - 10 + x$, and try to solve $g_1(x) = x$. This fails since $g_1(1) = -4$, which is outside the interval $[1, 2]$.
2. Define $g_2(x) = \sqrt{10/x - 4x}$, and try to solve $g_2(x) = x$. This fails since $g_2(x)$ is not defined (or complex) at $x = 2$.
3. It turns out that the best form is solving $x = g_3(x)$, where

$$g_3(x) = x - \frac{x^3 + 4x^2 - 10}{3x^2 + 8x},$$

that's probably not obvious at first glance!!! (Continued...)

Example: $x^3 + 4x^2 - 10 = 0$



Iteration #4:
$p = 1.3652$,
$p^3 + 4p^2 - 10 = 0.0001$
$g_3(p) = 1.3652$
$p = 1.3652$

How did we come up with this crazy function $g_3(x)$???

It will be explained in the next section (on Newton's method).

The bottom line is that without more analysis, it is extremely hard to find the best (or even a functioning) fixed point iteration which finds the correct solution.

Non-uniqueness of the Fixed Point

Strange, and sometimes beautiful things happen when part (a) (existence) of the fixed-point theorem is satisfied, but part (b) is not...

Let us consider the family of functions $f_a(x)$ parametrized by a, defined as

$$f_a(x) = 1 - ax^2, \quad x \in [-1, 1]$$

Given a particular value of a, the fixed point iteration

$$x_n = f_a(x_{n-1}) = 1 - ax_{n-1}^2$$

has a fixed point for values of $a \in [0, 2]$.

Non-uniqueness of the Fixed Point

By solving the quadratic equation, $ax^2 + x - 1 = 0$, we get the fixed point to be

$$p^* = -\frac{1 - \sqrt{1 + 4a}}{2a}$$

the other root is outside the interval $[-1, 1]$.

The derivative of $f_a(x) = 1 - ax^2$ at the fixed point is:

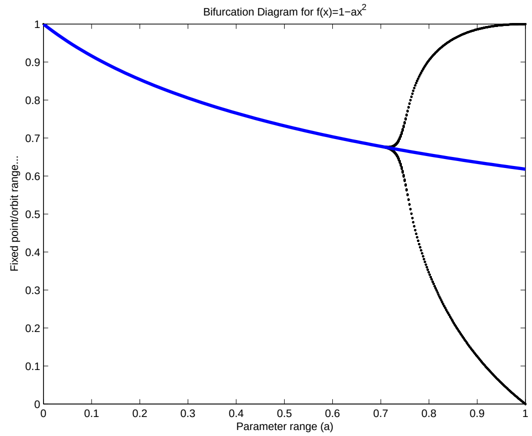
$$f_a'(p^*) = -2a\left[\frac{1 - \sqrt{1 + 4a}}{2a}\right] = \sqrt{1 + 4a} - 1 > 0$$

$$|f_a'(p^*)| = \sqrt{1 + 4a} - 1$$

as long as $a < \frac{3}{4}$ we have $|f_a'(p^*)| < 1$, but something definitely breaks when $a > \frac{3}{4}$...

Non-uniqueness of the Fixed Point

Let's look at the theoretical fixed point, and the computed values of the fixed point iteration... for values of a between 0 and 1.



Indeed, something strange happened around $a = 0.75$.

Root Finding
Improved Algorithms for Root Finding
Error Analysis
Fixed Point Iteration
Detour: — Non-unique Fixed Points...

Non-uniqueness of the Fixed Point

It turns out that when $|f'(p^*)| > 1$ the fixed point exists, but is no longer "attractive," *i.e.* the fixed point iteration does not converge to it.

Instead we settled in to a 2-orbit; — where the iteration "jumps" between the upper and lower branches of the diagram.

It turns out that the function

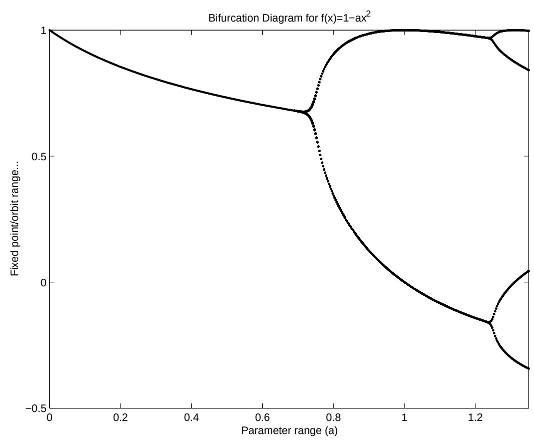
$$f_a^2(x) \equiv f_a(f_a(x)) = 1 - a(1 - ax^2)^2$$

has a unique fixed point for a in the range $[0.75,\ 1]$ (at least).

For some other critical value of a, the fixed point for $f_a(f_a(x))$ (the 2-orbit of $f_a(x)$) also becomes unstable... it breaks into a 4-orbit.

Root Finding
Improved Algorithms for Root Finding
Error Analysis
Fixed Point Iteration
Detour: — Non-unique Fixed Points...

Non-uniqueness of the Fixed Point



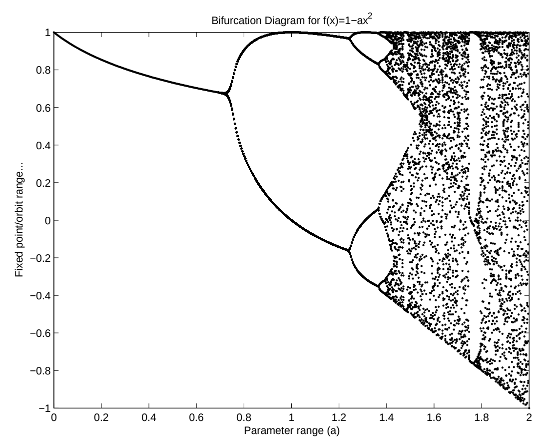
We see how iterations of $f_a(x)$ go from fixed-points, to 2-orbits, to 4-orbits...

Root Finding
Improved Algorithms for Root Finding
Error Analysis
Fixed Point Iteration
Detour: — Non-unique Fixed Points...

Non-uniqueness of the Fixed Point

As we may imagine, a 4-orbit corresponds to an attractive (stable) fixed point for the function $f_a^4(x) \equiv f_a^2(f_a^2(x)) \equiv f_a(f_a(f_a(f_a(x))))$.

It turns out we can play this game "forever..."



Summary: Playing with Fixed Point Iterations

The analysis of such "bifurcation diagrams" is done in Math 538 "Dynamical Systems and Chaos..."

The dynamics of $f_a(x) = 1 - ax^2$ is one of the simplest examples of chaotic behavior in a system.

Back to the Program: Quick Recap and Look-ahead

So far we have looked at two algorithms:

1. Bisection for **root finding**.
2. Fixed point iteration.

We have see that fixed point iteration and root finding are strongly related, but it is not always easy to find a good fixed-point formulation for solving the root-finding problem.

In the next section we will add three new algorithms for root finding:

1. Regula Falsi
2. Secant Method
3. Newton's Method

Newton's Method for Root Finding

Recall: we are looking for x^* so that $f(x^*) = 0$.

If $f \in C^2[a, b]$, and we know $x^* \in [a, b]$ (possibly by the intermediate value theorem), then we can formally Taylor expand around a point x close to the root:

$$0 = f(x^*) = f(x) + (x^* - x)f'(x) + \frac{(x - x^*)^2}{2} f''(\xi(x)), \quad \xi(x) \in [x, x^*]$$

If we are close to the root, then $|x - x^*|$ is small, which means that $|x - x^*|^2 \ll |x - x^*|$, hence we make the approximation:

$$0 \approx f(x) + (x^* - x)f'(x), \quad \Leftrightarrow \quad x^* \approx x - \frac{f(x)}{f'(x)}$$

Newton's Method for Root Finding

Newton's Method for root finding is based on the approximation

$$x^* \approx x - \frac{f(x)}{f'(x)}$$

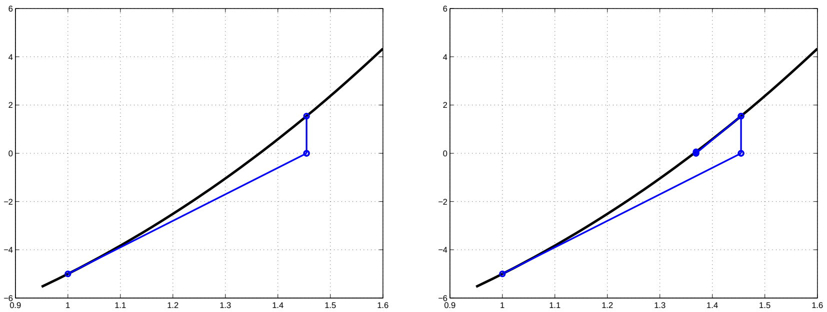
which is valid when x is close to x^*.

We use the above in the following way: given an approximation x_{n-1}, we get an **improved approximation** x_n by computing

$$x_n = x_{n-1} - \frac{f(x_{n-1})}{f'(x_{n-1})}$$

Geometrically, x_n is the intersection of the tangent of the function at x_{n-1} and the x-axis.

Two Steps of Newton for $f(x) = x^3 + 4x^2 - 10 = 0$



$$p_0 = 1$$

$$p_1 = p_0 - \frac{p_0^3 + 4p_0^2 - 10}{3p_0^2 + 8p_0} = 1.45454545454545$$

$$p_2 = p_1 - \frac{p_1^3 + 4p_1^2 - 10}{3p_1^2 + 8p_1} = 1.36890040106952$$

The Pros and Cons of Newton's Method

Strategy: Newton's Method

$$x_n = x_{n-1} - \frac{f(x_{n-1})}{f'(x_{n-1})}$$

Fast convergence: Newton's method converges the fastest of the methods we explore today. (Quadratic convergence).

Clearly, points where $f'(\cdot) = 0$ will cause problems!

It is especially problematic if $f(x^*) = f'(x^*) = 0$ — we cannot avoid the point where $f'(\cdot) = 0$ in this case; it is the point we are looking for!

Newton's method work best if $f'(\cdot) \geq k > 0$.

"Expensive:" We have to compute the derivative in every iteration.

Finding a Starting Point for Newton's Method

Recall our initial argument that when $|x - x^*|$ is small, then $|x - x^*|^2 \ll |x - x^*|$, and we can neglect the second order term in the Taylor expansion.

In order for Newton's method to converge we need a **good starting point!**

Theorem

Let $f(x) \in C^2[a, b]$. If $x^* \in [a, b]$ such that $f(x^*) = 0$ and $f'(x^*) \neq 0$, then there exists a $\delta > 0$ such that Newton's method generates a sequence $\{x_n\}_{n=1}^{\infty}$ converging to x^* for any initial approximation $x_0 \in [x^* - \delta, x^* + \delta]$.

The theorem is interesting, but quite useless for practical purposes. In practice: Pick a starting value x_0, iterate a few steps. Either the iterates converge quickly to the root, or it will be clear that convergence is unlikely.

Newton's Method as a Fixed Point Iteration

If we view Newton's method as a fixed point iteration...

$$x_n = \underbrace{x_{n-1} - \frac{f(x_{n-1})}{f'(x_{n-1})}}_{g(x_{n-1})}$$

Then (the fixed point theorem), we must find an interval $[x^* - \delta, x^* + \delta]$ that g maps into itself, and for which $|g'(x)| \leq k < 1$.

$g'(x)$ is quite an expression:

$$g'(x) = 1 - \frac{f'(x)f'(x) - f(x)f''(x)}{[f'(x)]^2} = \frac{f(x)f''(x)}{[f'(x)]^2}$$

By assumption, $f(x^*) = 0$, $f'(x^*) \neq 0$, so $g'(x^*) = 0$. By continuity $|g'(x)| \leq k < 1$ for some neighborhood of x^*... Hence the fixed point iteration will converge. (Gory details in the book).

Algorithm — Newton's Method

Algorithm: Newton's Method

`Input:` Initial approximation p_0; tolerance TOL; maximum number of iterations N_0.

`Output:` Approximate solution p, or failure message.

```
1.   Set i = 1
2.   While i ≤ N0 do 3--6
3.      Set p = p0 − f(p0)/f′(p0)
4.      If |p − p0| < TOL then
4a.         output p
4b.         stop program
5.      Set i = i + 1
6.      Set p0 = p.
7.   Output: "Failure after N0 iterations."
```


Root Finding
Improved Algorithms for Root Finding
Error Analysis
Newton's Method
The Secant Method
The Regula Falsi Method
Quick Summary

The Secant Method

The main weakness of Newton's method is the need to compute the derivative, $f'(\cdot)$, in each step. Many times $f'(\cdot)$ is far more difficult to compute and needs more arithmetic operations to calculate than $f(x)$.

What to do??? — **Approximate the derivative!**

By definition

$$f'(x_{n-1}) = \lim_{x \to x_{n-1}} \frac{f(x) - f(x_{n-1})}{x - x_{n-1}}$$

Let $x = x_{n-2}$, and approximate

$$f'(x_{n-1}) \approx \frac{f(x_{n-2}) - f(x_{n-1})}{x_{n-2} - x_{n-1}}.$$

Root Finding
Improved Algorithms for Root Finding
Error Analysis
Newton's Method
The Secant Method
The Regula Falsi Method
Quick Summary

The Secant Method

Using the approximation

$$f'(x_{n-1}) \approx \frac{f(x_{n-2}) - f(x_{n-1})}{x_{n-2} - x_{n-1}}$$

for the derivative in Newton's method

$$x_n = x_{n-1} - \frac{f(x_{n-1})}{f'(x_{n-1})},$$

gives us **the Secant Method**

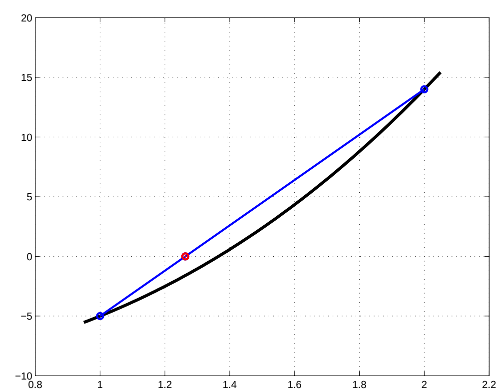
$$\begin{aligned} x_n &= x_{n-1} - \frac{f(x_{n-1})}{\left[\frac{f(x_{n-2}) - f(x_{n-1})}{x_{n-2} - x_{n-1}}\right]} \\ &= x_{n-1} - \frac{f(x_{n-1})\,[x_{n-2} - x_{n-1}]}{f(x_{n-2}) - f(x_{n-1})} \end{aligned}$$

Root Finding
Improved Algorithms for Root Finding
Error Analysis
Newton's Method
The Secant Method
The Regula Falsi Method
Quick Summary

The Secant Method

Strategy: The Secant Method

$$x_n = x_{n-1} - \frac{f(x_{n-1})\,[x_{n-2} - x_{n-1}]}{f(x_{n-2}) - f(x_{n-1})}$$



Instead of (as in Newton's method) getting the next iterate from the zero-crossing of the tangent line, the next iterate for the secant method is the zero-crossing of the secant line...

Algorithm — The Secant Method

Algorithm: The Secant Method

`Input:` Initial approximations p_0, p_1; tolerance TOL; maximum number of iterations N_0.

`Output:` Approximate solution p, or failure message.

1. Set $i = 2$, $\quad q_0 = f(p_0)$, $\quad q_1 = f(p_1)$
2. While $i \leq N_0$ do 3--6
3. Set $p = p_1 - q_1(p_1 - p_0)/(q_1 - q_0)$
4. If $|p - p_1| < TOL$ then
4a. output p
4b. stop program
5. Set $i = i + 1$
6. Set $p_0 = p_1$, $\quad q_0 = q_1$, $\quad p_1 = p$, $\quad q_1 = f(p_1)$
7. Output: "Failure after N_0 iterations."

Regula Falsi (the Method of False Position)

Regula Falsi is a combination of the Secant and the Bisection methods: We start with two points a_{n-1}, b_{n-1} which bracket the root, *i.e.* $f(a_{n-1}) \cdot f(b_{n-1}) < 0$. Let s_n be the zero-crossing of the secant-line, *i.e.*

$$s_n = b_{n-1} - f(b_{n-1}) \left[\frac{a_{n-1} - b_{n-1}}{f(a_{n-1}) - f(b_{n-1})} \right].$$

Update as in the bisection method:

$$\begin{array}{lll} \text{if } f(a_{n-1}) \cdot f(s_n) > 0 & \text{then } a_n = s_n, & b_n = b_{n-1} \\ \text{if } f(a_{n-1}) \cdot f(s_n) < 0 & \text{then } a_n = a_{n-1}, & b_n = s_n \end{array}$$

Regula Falsi is seldom used (it can run into some "issues" – to be explored soon), but illustrates how bracketing can be incorporated.

Algorithm — Regula Falsi

Algorithm: Regula Falsi

`Input:` Initial approximations p_0, p_1; tolerance TOL; maximum number of iterations N_0.

`Output:` Approximate solution p, or failure message.

1. Set $i = 2$, $\quad q_0 = f(p_0)$, $\quad q_1 = f(p_1)$
2. While $i \leq N_0$ do 3--7
3. Set $p = p_1 - q_1(p_1 - p_0)/(q_1 - q_0)$
4. If $|p - p_1| < TOL$ then
4a. output p
4b. stop program
5. Set $i = i + 1$, $\quad q = f(p)$
6. If $q \cdot q_1 < 0$ then set $p_0 = p_1$, $\quad q_0 = q_1$
7. Set $p_1 = p$, $\quad q_1 = q$
8. Output: "Failure after N_0 iterations."

Summary — Next Iterate

Method	Next Iterate
Bisection	Midpoint of bracketing interval: $m_{n+1} = (a_n + b_n)/2$: if $f(c_{n+1})f(b_n) < 0$, then $\{a_{n+1} = m_{n+1},\ b_{n+1} = b_n\}$, else $\{a_{n+1} = a_n,\ b_{n+1} = m_{n+1}\}$.
Regula Falsi	Zero-crossing of secant line: $s_{n+1} = x_n - f(x_n) \cdot \frac{x_n - x_{n-1}}{f(x_n) - f(x_{n-1})}$: if $f(s_{n+1})f(b_n) < 0$, then $\{a_{n+1} = s_{n+1},\ b_{n+1} = b_n\}$, else $\{a_{n+1} = a_n,\ b_{n+1} = s_{n+1}\}$.
Secant	Zero-crossing of secant line: $x_{n+1} = x_n - f(x_n) \cdot \frac{x_n - x_{n-1}}{f(x_n) - f(x_{n-1})}$
Newton	Zero-crossing of tangent line: $x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$

Summary — Convergence

Method	Convergence
Bisection	**Linear** — Slow, each iteration gives 1 binary digit. We need about 3.3 iterations to gain one decimal digit...
Regula Falsi	**Linear** — Faster than Bisection (ideally).
Secant	**Linear/Superlinear?** — Slower than Newton. Generally faster than Regula Falsi. Burden-Faires says convergence rate is $\alpha \approx 1.62$. After some digging around, I found a proof that the convergence rate should be $\frac{1+\sqrt{5}}{2}$, given sufficient smoothness of f, and $f'(x^*) \neq 0$.
Newton	**Quadratic.** — In general. The fastest of the lot, when it works.

Summary — Cost

Method	Cost
Bisection	Each iteration is cheap — one function evaluation, one or two multiplications and one or two comparisons. Comparable to Regula Falsi.
Regula Falsi	Higher cost per iteration compared with Secant (conditional statements), Requires more iterations then Secant. Higher cost per iteration compared with Bisection, but requires fewer iterations.
Secant	Cheaper than Newton's Method – no need to compute f'(x). Slightly cheaper per iteration than Regula Falsi.
Newton	*"Expensive"* — We need to compute $f'(x)$ in every iteration.

Summary — Comments

Method	Comments
Bisection	Can be used to find a good starting interval for Newton's method (if/when we have a problem finding a good starting point for Newton).
Regula Falsi	The combination of the Secant method and the Bisection method. All generated intervals bracket root (i.e. we carry a "built-in" error estimate at all times.)
Secant	Breaks down if $f(x_n) = f(x_{n-1})$ [division by zero]. Unknown basin of attraction (*c.f.* Newton's method).
Newton	If $f'(x_k) = 0$ we're in trouble. Works best when $\lvert f'(x)\rvert \geq k > 0$. Iterates do not bracket root. Unknown basin of attraction (How do we find a good starting point?). **In practice**: Pick a starting point x_0, iterate. It will very quickly become clear whether we will converge to a solution, or diverge...

Newton's Method and Friends — Things to Ponder...

- How to start
- How to update
- Can the scheme break?
  $\rightarrow$ Can we fix breakage? (How???)
- Relation to Fixed-Point Iteration

In the next section we will discuss the convergence in more detail.

Introduction: Error Analysis

In the previous section we discussed four different algorithms for finding the root of $f(x) = 0$.

We made some (sometime vague) arguments for why one method would be faster than another...

Now, we are going to look at the error analysis of iterative methods, and we will quantify the speed of our methods.

Note: The discussion may be a little "dry," but do not despair! In the "old days" before fancy-schmancy computers were commonplace is was almost true that `numerical analysis` $\equiv$ `error analysis`, here we only look at some basics...

Definition of Convergence for a Sequence

Definition

Suppose the sequence $\{p_n\}_{n=0}^{\infty}$ converges to p, with $p_n \neq p$ for all n. If positive constants λ and α exists with

$$\lim_{n\to\infty} \frac{|p_{n+1} - p|}{|p_n - p|^{\alpha}} = \lambda$$

then $\{p_n\}_{n=0}^{\infty}$ **converges to** p **of order** α**, with asymptotic error constant** λ**.**

An iterative technique of the form $p_n = g(p_{n-1})$ is said to be of **order** α if the sequence $\{p_n\}_{n=0}^{\infty}$ converges to the solution $p = g(p)$ of order α.

Bottom line: High order $(\alpha) \Rightarrow$ Faster convergence (more desirable).
λ has an effect, but is less important than the order.

Special Cases: $\alpha = 1$, and $\alpha = 2$

When $\alpha = 1$ the sequence is **linearly convergent.**

When $\alpha = 2$ the sequence is **quadratically convergent.**

When $\alpha < 1$ the sequence is **sub-linearly convergent** (very undesirable, or "painfully slow.")

When (($\alpha = 1$ and $\lambda = 0$) or $1 < \alpha < 2$), the sequence is **super-linearly convergent**.

Linear vs. Quadratic

Suppose we have two sequences converging to zero:

$$\lim_{n\to\infty} \frac{|p_{n+1}|}{|p_n|} = \lambda_p, \qquad \lim_{n\to\infty} \frac{|q_{n+1}|}{|q_n|^2} = \lambda_q$$

Roughly this means that

$$|p_n| \approx \lambda_p |p_{n-1}| \approx \lambda_p^n |p_0|, \qquad |q_n| \approx \lambda_q |q_{n-1}|^2 \approx \lambda_q^{2^n - 1} |q_0|^{2^n}$$

Linear vs. Quadratic

Now, assume $\lambda_p = \lambda_q = 0.9$ and $p_0 = q_0 = 1$, we get the following

n	p_n	q_n
0	1	1
1	0.9	0.9
2	0.81	0.729
3	0.729	0.4782969
4	0.6561	0.205891132094649
5	0.59049	0.0381520424476946
6	0.531441	0.00131002050863762
7	0.4782969	0.00000154453835975
8	0.43046721	0.00000000000021470

Table (Linear vs. Quadratic): A dramatic difference! After 8 iterations, q_n has 11 correct decimals, and p_n still *none*. q_n roughly doubles the number of correct digits in every iteration. Here p_n needs more than 20 iterations/digit-of-correction.

Root Finding
Improved Algorithms for Root Finding
Error Analysis
Convergence
Practial Application of the Theorems
Roots of Higher Multiplicity

Convergence of General Fixed Point Iteration

Theorem

Let $g \in C[a,b]$ be such that $g(x) \in [a,b]$ for all $x \in [a,b]$. Suppose, in addition that $g'(x)$ is continuous on (a,b) and there is a positive constant $k < 1$ so that

$$|g'(k)| \leq k, \quad \forall x \in (a,b)$$

If $\mathbf{g'(p^*) \neq 0}$, then for any number p_0 in $[a,b]$, the sequence

$$p_n = g(p_{n-1}), \quad n \geq 1$$

converges **only linearly** *to the unique fixed point p^* in $[a,b]$.*

In a sense, this is bad news since we like fast convergence...

Root Finding
Improved Algorithms for Root Finding
Error Analysis
Convergence
Practial Application of the Theorems
Roots of Higher Multiplicity

Convergence of General Fixed Point Iteration

Proof

The existence and uniqueness of the fixed point follows from the fixed point theorem.

We use the mean value theorem to write

$$p_{n+1} - p^* = g(p_n) - g(p^*) = g'(\xi_n)(p_n - p^*), \quad \xi_n \in (p_n, p^*)$$

Since $p_n \to p^*$ and ξ_n is between p_n and p^*, we must also have $\xi_n \to p^*$. Further, since $g'(\cdot)$ is continuous, we have

$$\lim_{n\to\infty} g'(\xi_n) = g'(p^*)$$

Thus,

$$\lim_{n\to\infty} \frac{|p_{n+1} - p^*|}{|p_n - p^*|} = \lim_{n\to\infty} |g'(\xi_n)| = |g'(p^*)|$$

So if $g'(p^*) \neq 0$, the fixed point iteration converges linearly with asymptotic error constant $|g'(p^*)|$.

Root Finding
Improved Algorithms for Root Finding
Error Analysis
Convergence
Practial Application of the Theorems
Roots of Higher Multiplicity

Speeding up Convergence of Fixed Point Iteration

Bottom Line: The theorem tells us that if we are looking to design rapidly converging fixed point schemes, we must design them so that $g'(p^*) = 0$...

We state the following without proof:

Theorem

Let p^* be a solution of $p = g(p)$. Suppose $g'(p^*) = 0$, and $g''(x)$ is continuous and strictly bounded by M on an open interval I containing p^*. Then there exists a $\delta > 0$ such that, for $p_0 \in [p^* - \delta, p^* + \delta]$ the sequence defined by $p_n = g(p_{n-1})$ converges **at least quadratically** *to p^*. Moreover, for sufficiently large n*

$$|p_{n+1} - p^*| < \frac{M}{2}|p_n - p^*|^2$$

Practical Application of the Theorems

The theorems tell us:

"Look for quadratically convergent fixed point methods among functions whose derivative is zero at the fixed point."

We want to solve: $f(x) = 0$ using fixed point iteration. We write the problem as an equivalent fixed point problem:

$g(x) = x - f(x)$	Solve: $x = g(x)$	
$g(x) = x - \alpha f(x)$	Solve: $x = g(x)$	α a constant
$g(x) = x - \Phi(x)f(x)$	Solve: $x = g(x)$	$\Phi(x)$ differentiable

We use the most general form (the last one).
Remember, we want $g'(p^*) = 0$ when $f(p^*) = 0$.

Practical Application of the Theorems... Newton's Method Rediscovered

$$g'(x) = \frac{d}{dx}\left[x - \Phi(x)f(x)\right] = 1 - \Phi'(x)f(x) - \Phi(x)f'(x)$$

at $x = p^*$ we have $f(p^*) = 0$, so

$$g'(p^*) = 1 - \Phi(p^*)f'(p^*).$$

For quadratic convergence we want this to be zero, that's true if

$$\Phi(p^*) = \frac{1}{f'(p^*)}.$$

Hence, our scheme is

$$g(x) = x - \frac{f(x)}{f'(x)},$$ **Newton's Method, rediscovered!**

Newton's Method

We have "discovered" Newton's method in two scenarios:

1. From formal Taylor expansion.
2. From convergence optimization of Fixed point iteration.

It is clear that we would like to use Newton's method in many settings. One major problem is that it breaks when $f'(p^*) = 0$ (division by zero).

The good news is that this problem can be fixed!
— We need a short discussion on the **multiplicity of zeroes**.

Multiplicity of Zeroes

Definition: Multiplicity of a Root

A solution p^* of $f(x) = 0$ is a **zero of multiplicity** m of f if for $x \neq p$ we can write

$$f(x) = (x - p^*)^m q(x), \qquad \lim_{x \to p^*} q(x) \neq 0$$

Basically, $q(x)$ is the part of $f(x)$ which does not contribute to the zero of $f(x)$.

If $m = 1$ then we say that $f(x)$ has a **simple zero**.

Theorem

$f \in C^1[a, b]$ has a simple zero at p^* in (a, b) if and only if $f(p^*) = 0$, but $f'(p^*) \neq 0$.

Multiplicity of Zeroes

Theorem (Multiplicity and Derivatives)

The function $f \in C^m[a,b]$ has a zero of multiplicity m at p^* in (a,b) if and only if

$$0 = f(p^*) = f'(p^*) = \cdots f^{(m-1)}(p^*), \quad \text{but } f^{(m)}(p^*) \neq 0.$$

We know that Newton's method runs into trouble when we have a zero of multiplicity higher than 1.

Newton's Method for Zeroes of Higher Multiplicity

Suppose $f(x)$ has a zero of multiplicity $m > 1$ at p^*...
Define the new function

$$\mu(x) = \frac{f(x)}{f'(x)}.$$

We can write $f(x) = (x-p^*)^m q(x)$, hence

$$\begin{aligned} \mu(x) &= \frac{(x-p^*)^m q(x)}{m(x-p^*)^{m-1}q(x) + (x-p^*)^m q'(x)} \\ &= (x-p^*)\frac{q(x)}{mq(x) + (x-p^*)q'(x)} \end{aligned}$$

This expression has a simple zero at p^*, since

$$\frac{q(p^*)}{mq(p^*) + (p^*-p^*)q'(p^*)} = \frac{1}{m} \neq 0$$

Newton's Method for Zeroes of Higher Multiplicity

Now we apply Newton's method to $\mu(x)$:

$$\begin{aligned} x &= g(x) = x - \frac{\mu(x)}{\mu'(x)} \\ &= x - \frac{\frac{f(x)}{f'(x)}}{\frac{[f'(x)]^2 - f(x)f''(x)}{[f'(x)]^2}} \\ &= x - \frac{f(x)f'(x)}{[f'(x)]^2 - f(x)f''(x)} \end{aligned}$$

This iteration will converge quadratically!

Newton's Method for Zeroes of Higher Multiplicity

Strategy: "Fixed Newton" (for Zeroes of Multiplicity ≥ 2)

$$x_{n+1} = x_n - \frac{f(x_n)f'(x_n)}{[f'(x_n)]^2 - f(x_n)f''(x_n)}$$

Drawbacks:

We have to compute $f''(x)$ — more expensive and possibly another source of numerical and/or measurement errors.

We have to compute a more complicated expression in each iteration — more expensive.

Roundoff errors in the denominator — both $f'(x)$ and $f(x)$ approach zero, so we are computing the difference between two small numbers; a serious cancelation risk.