11

Solving nonlinear equations

Have: $f: \mathbb{R} \to \mathbb{R}$ function

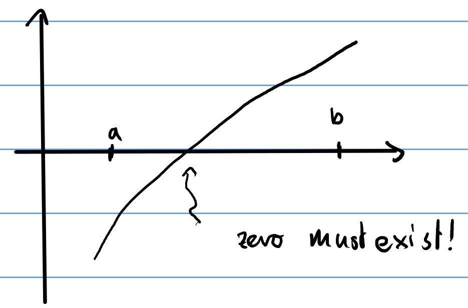
Want: x such that $f(x) = y$

Rewrite the problem so that we only need $g(x) = 0$. (i.e. no explicit right-hand side)

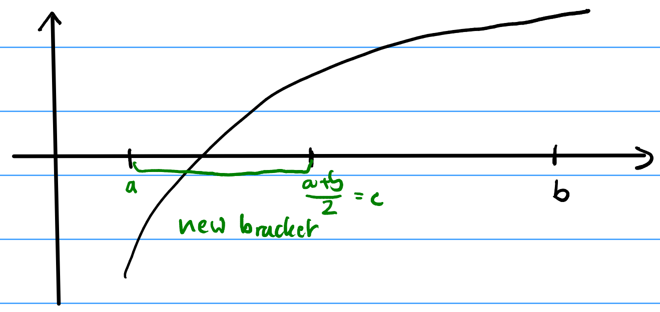
$$g(x) = f(x) - y \quad \rightsquigarrow \quad f(x) = y \Leftrightarrow g(x) = 0.$$

What if we know that f is continuous and $f(a) \cdot f(b) < 0$?

$f(a)$ and $f(b)$ have opposite signs.



Can we use this "bracket" to track down the zero?



- Check signs of function values to keep zero in new bracket
- Repeat until bracket small enough (halves in length with each step)

This is called the "bisection method".

Demo: Bisection Method

Convergence Rates of Iterative Procedures

Consider the "error" in the bisection method in the kth step:

$$e_k = |b_k - a_k| \leftarrow \text{length of the bracket}$$

What's the error in the next step, relative to e_k ?

$$e_{k+1} = \frac{1}{2} \cdot e_k$$

Generally, error behavior like this is called "linear convergence" ("order 1"):

$$e_{k+1} \leqslant C \cdot e_k \quad \text{with} \quad 0 \leqslant C < 1$$

Generally, error behavior like this is called "quadratic convergence" ("order 2"):

$$e_{k+1} \leqslant C \cdot e_k^2 \quad \text{with} \quad 0 \leqslant C < 1$$

Generally, error behavior like this is called "cubic convergence" ("order 3"):

$$e_{k+1} \leqslant C e_k^3 \quad \text{with} \quad 0 \leqslant C < 1$$

(... and so on) Which of these is fastest? cubic

Rewrite this so that the constant stands on its own, for a general order q :

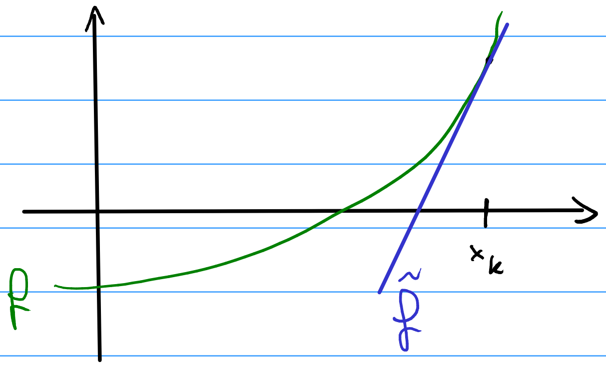
$$C \approx \frac{e_{k+1}}{e_k^q} \leftarrow \text{Print this, check for constant-ness to see if q-th order!}$$

Do not confuse this with "q-th order" convergence $\sim C \cdot h^q$ for a mesh width h!

Demo: Rates of Convergence

Newton's method

Suppose x_k is our current guess of the zero.



Idea: Build a solvable approximate version of f using $f(x_k)$ and $f'(x_k)$

$$\tilde{f}(x_k + h) = f(x_k) + h f'(x_k)$$

Find the zero of the approximate version.

$$\text{Solve} \quad 0 \overset{!}{=} \tilde{f}(x_k + h)$$

$$= f(x_k) + h f'(x_k)$$

$$\rightsquigarrow \quad h = -\frac{f(x_k)}{f'(x_k)}$$

$$\rightsquigarrow \quad x_{k+1} = x_k + h = x_k - \frac{f(x_k)}{f'(x_k)}$$

This is called Newton's method.

Demo: Newton's method

Demo: Convergence of Newton's method

Application: How does a computer find a square root?

Name some downsides of Newton's method.

Convergence only 'near' a zero, not far from it.

("Local", but not "global" convergence)

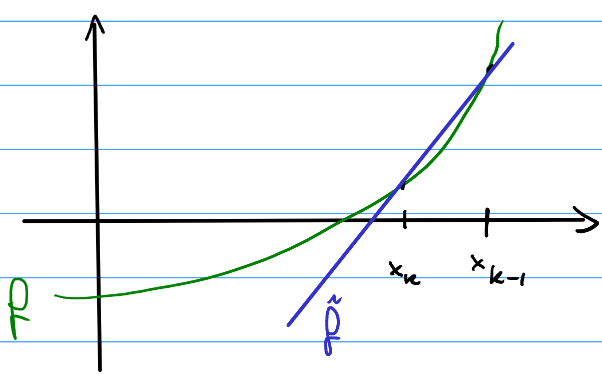
Need the derivative f', which we may or may not have.

↑ Let's try and address this last issue.

Secant Method

How else could we find a line approximating a function?

Use last two guesses: x_k and x_{k-1}



Estimate the slope of the approximating line:

$$\frac{\text{Rise}}{\text{Run}} = \frac{f(x_k) - f(x_{k-1})}{x_k - x_{k-1}} \approx f'(x_k)$$

Now use this estimate in Newton's method:

$$x_{k+1} = x_k + h = x_k - \frac{f(x_k)}{\cancel{f'(x_k)}}$$

$$= x_k - \frac{f(x_k)}{\frac{f(x_k) - f(x_{k-1})}{x_k - x_{k-1}}}$$

Demo: Secant Method   Demo: Convergence of the Secant Method

Solving systems of nonlinear equations

Want to solve $\vec{f}(\vec{x}) = \vec{0}$. $\qquad f: \mathbb{R}^n \to \mathbb{R}^n$! why?

Let's try to carry over our 1-dimensional ideas.

Let's first get an idea of what behavior can occur.

Demo: Three quadratic functions

Based on the demo: Does bisection stand a chance?

Not really--no easy equivalent of 'bracket'.

Let's try Newton's method then. What's the linear approximation of f ?

1D: $\tilde{f}(x+h) = f(x) + f'(x)\cdot h \approx f(x+h)$

nD: $\tilde{\vec{f}}(\vec{x}+\vec{h}) = \vec{f}(\vec{x}) + J_f(\vec{x})\,\vec{h} \approx \vec{f}(\vec{x}+\vec{h})$

where $J_f(\vec{x}) = \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & & \vdots \\ \frac{\partial f_n}{\partial x_1} & \cdots & \frac{\partial f_n}{\partial x_n} \end{pmatrix}$. "Jacobian matrix"

OK, now solve that for h.

a linear system (surprised?)

$\tilde{\vec{f}}(x+h) = \vec{f}(\vec{x}) + J_f(\vec{x})\,\vec{h} \doteq \vec{0} \quad \rightsquigarrow \quad J_f(\vec{x})\,\vec{h} = -\vec{f}(\vec{x})$

$\rightsquigarrow \quad \vec{h} = -J_f(\vec{x})^{-1}\,\vec{f}(\vec{x})$

$\rightsquigarrow \quad x_{k+1} = x_k - J_f(\vec{x}_k)^{-1}\,\vec{f}(\vec{x}_k)$

Let's do an example of that:

$$f(x,y) = \begin{pmatrix} x + 2y - 2 \\ x^2 + 4y^2 - 4 \end{pmatrix}$$

$$J_f(x,y) = \begin{pmatrix} 1 & 2 \\ 2x & 8y \end{pmatrix}$$

Demo: Newton's method in n dimensions

What are the downsides of this method?

- Local convergence only
- Need the Jacobian

So how about (an n-dimensional analog of) the secant method?

Idea: Find enough information to allow reconstructing the Jacobian

$$\underbrace{\underset{\in \mathbb{R}^n}{f(x_k)} \text{ and } \underset{\in \mathbb{R}^n}{f(x_{k-1})}}_{\text{20 numbers}}. \qquad \text{But: } \underbrace{\underset{\in \mathbb{R}^{n \times n}}{J_f(x)}}_{\text{100 numbers}}$$

$n = 10$:

not enough!

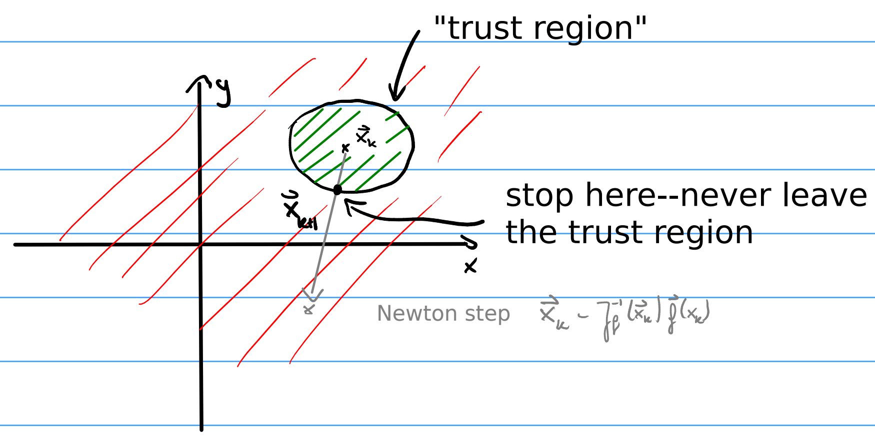
So carrying over the secant method to n dimensions is not easy.

It's possible, but beyond the scope of our class.

Here are two starting points to search:

- Broyden's method
- Secant updating methods

Here's one more idea: If we could figure out where the linear approximation in Newton is 'trustworthy', would that buy us anything?



These are called "trust region methods".

They can help make Newton's method a little more robust.