4. Nonlinear equations

- nonlinear equations
- iterative methods
- bisection method
- Newton method

Set of nonlinear equations

consider n nonlinear equations in n variables $x_1, x_2, \ldots, x_n$:

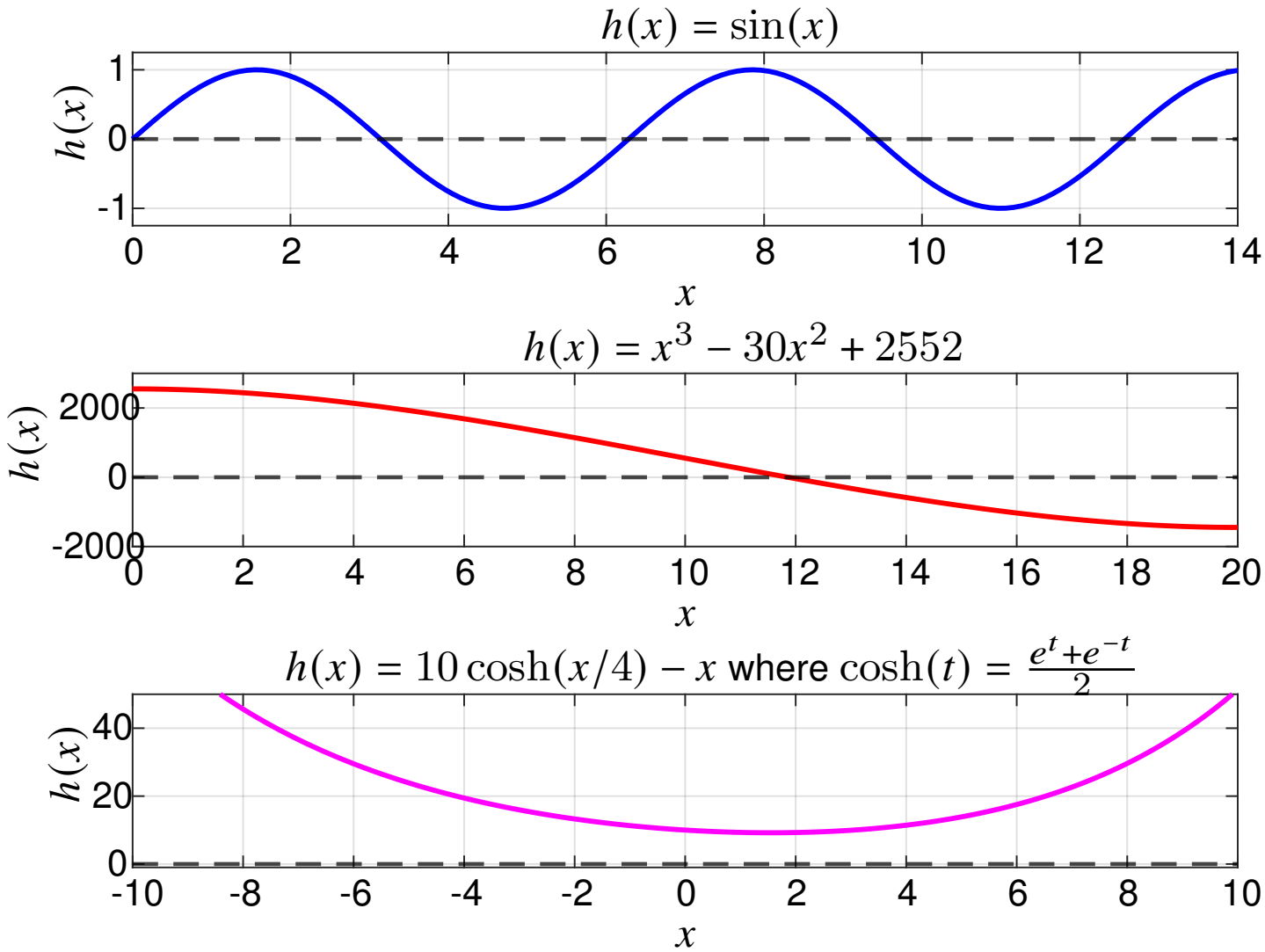
$$h_i(x_1, \ldots, x_n) = 0, \quad i = 1, \ldots, n$$

in vector notation: $h(x) = 0$ with

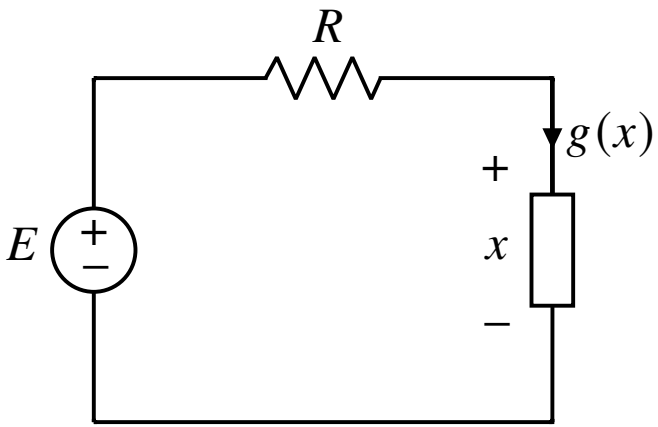
$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, \quad h(x) = \begin{bmatrix} h_1(x_1, \ldots, x_n) \\ h_2(x_1, \ldots, x_n) \\ \vdots \\ h_n(x_1, \ldots, x_n) \end{bmatrix}$$

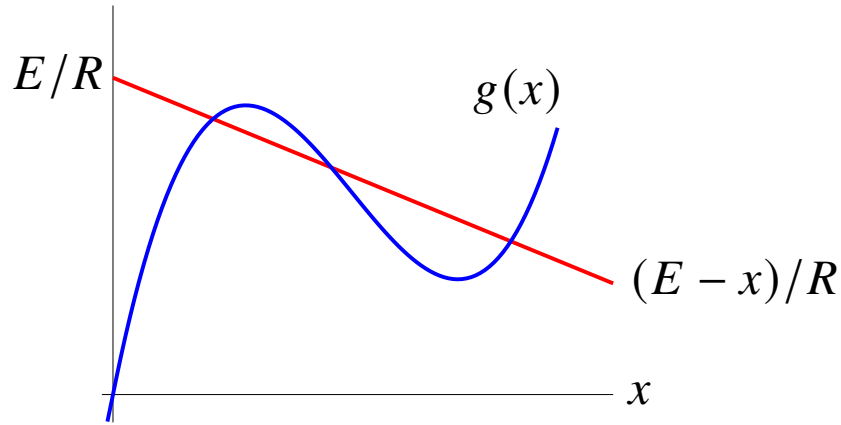
- $h_i : \mathbb{R}^n \to \mathbb{R}$
- $h_i(x)$ is ith *residual*; $h(x)$ is residual vector
- the *root* or *zero* is any solution of the above equation
- may have unique solution, no solution, or infinitely many solutions

Examples


$h(x) = \sin(x)$
$h(x)$
1
0
-1
0
2
4
6
8
10
12
14
x
$h(x) = x^3 - 30x^2 + 2552$
$h(x)$
2000
0
-2000
0
2
4
6
8
10
12
14
16
18
20
x
$h(x) = 10\cosh(x/4) - x$ where $\cosh(t) = \frac{e^t + e^{-t}}{2}$
$h(x)$
40
20
0
-10
-8
-6
-4
-2
0
2
4
6
8
10
x

Example: nonlinear resistive circuit





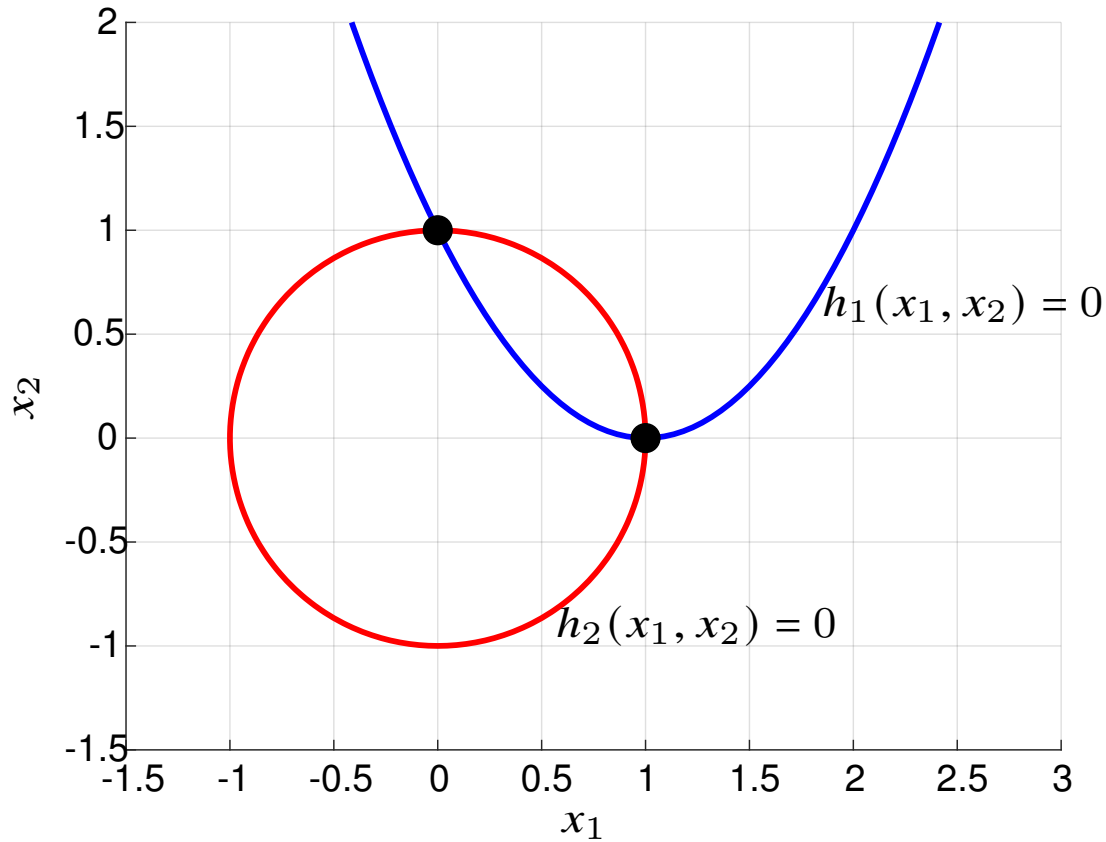
$$g(x) - \frac{E - x}{R} = 0$$

a nonlinear equation in the variable x, with three solutions

Example

$$h_1(x_1, x_2) = x_1^2 - 2x_1 - x_2 + 1 = 0$$
$$h_2(x_1, x_2) = x_1^2 + x_2^2 - 1 = 0$$

solutions at $x^{\star(1)} = (1, 0)$ and $x^{\star(2)} = (0, 1)$



Outline

- nonlinear equations
- **iterative methods**
- bisection method
- Newton method

Iterative methods

- nonlinear equations are much difficult to solve compared to linear equations
- obtaining a solution by finite-step algorithm is not feasible
- iterative algorithms start with *initial* or *starting point*, $x^{(0)}$ and compute estimates

$$x^{(1)}, x^{(2)}, \ldots, x^{(k)}, \ldots$$

- moving from $x^{(k)}$ to $x^{(k+1)}$ is called an *iteration* of the algorithm
- ideally converge to a root of the target function

$$x^{(k)} \longrightarrow x^{\star} \quad \text{as } k \longrightarrow \infty$$

where $h(x^{\star}) = 0$

Stopping iterative methods

- *function value:* stop when $\|h(x^{(k)})\| < \epsilon$
- *absolute error:* stop when $\|x^{(k)} - x^{(k-1)}\| < \epsilon$
- *relative error:* stop when $\|x^{(k)} - x^{(k-1)}\| / \|x^{(k)}\| < \epsilon$
- stop when $k = k_{\max}$ where $k_{\max}$ is a predetermined integer

here, ϵ is a tolerance level constant determined by the user

Convergence rates

assume the sequence $x^{(k)}$ converges to a limit $x^\star$

Linear convergence: if there exists a constant $c \in (0, 1)$ such that

$$\|x^{(k)} - x^\star\| \leq c\|x^{(k-1)} - x^\star\| \quad \text{for sufficiently large } k$$

example: $x^{(k)} = 1 + (1/2)^k$ linearly converges to $x^\star = 1$,

$$|x^{(k+1)} - x^\star| = (1/2)^{k+1} = \frac{1}{2}|x^{(k)} - x^\star|$$

satisfies the definition with $c = 1/2$

R-linear convergence: if a positive constant M and a value $c \in (0, 1)$ exist such that

$$\|x^{(k)} - x^\star\| \leq Mc^k \quad \text{for sufficiently large } k$$

linear convergence implies R-linear convergence (reverse is not necessarily true)

Superlinear convergence: if a sequence $c_k > 0$ with $c_k \longrightarrow 0$ exists such that

$$\|x^{(k)} - x^\star\| \leq c_k \|x^{(k-1)} - x^\star\| \quad \text{for large } k$$

example: $x^{(k)} = 1 + (1/(k+1))^k$ has superlinear convergence to $x^\star = 1$, as

$$|x^{(k)} - x^\star| = \frac{1}{(k+1)^k} = \frac{k^{k-1}}{(k+1)^k}\frac{1}{k^{k-1}} = \frac{k^{k-1}}{(k+1)^k}|x^{(k-1)} - x^\star|$$

satisfies the definition with $c_k = k^{k-1}/(k+1)^k$, which approaches zero

Quadratic convergence: if a constant $c > 0$ exists such that

$$\|x^{(k)} - x^\star\| \leq c \|x^{(k-1)} - x^\star\|^2 \quad \text{for large } k$$

example: $x^{(k)} = 1 + (1/2)^{2^k}$ has quadratic convergence to $x^\star = 1$, as

$$|x^{(k+1)} - x^\star| = (1/2)^{2^{k+1}} = \left((1/2)^{2^k}\right)^2 = |x^{(k)} - x^\star|^2$$

satisfies the definition with $c = 1$

Outline

- nonlinear equations
- iterative methods
- **bisection method**
- Newton method

The bisection method

one equation in one variable

$$h(x) = 0, \quad x \in \mathbb{R}$$

---

given: a, b with $a < b$, $h(a)h(b) < 0$, and tolerance ϵ

repeat

1. $x = (a + b)/2$
2. compute $h(x)$; **if** $h(x) = 0$, **return** x
3. **if** $h(x)h(a) < 0$, $b = x$, **else**, $a = x$
4. **stop** if $b - a \leq \epsilon$

- condition $h(a)h(b) < 0$ ensures a root exists between a, b
- a, b can be chosen from graphing the function
- h is assumed to be continuous

Convergence

let $[a^{(k)}, b^{(k)}]$ be the interval after iteration k, then

$$b^{(k)} - a^{(k)} = (b-a)/2^k$$

- after k iterations, the midpoint $x^{(k)} = (b^{(k)} + a^{(k)})/2$ satisfies

  $$|x^{(k)} - x^\star| \leq b^{(k)} - a^{(k)} \leq (1/2)^k (b-a)$$

  thus, it is R-linearly convergent with $c = 1/2$ and $M = b - a$

- the exit condition $b^{(k)} - a^{(k)} \leq \epsilon$ will be satisfied if

  $$\log_2 \left(\frac{b-a}{2^k} \right) = \log_2 (b-a) - k \leq \log_2 \epsilon$$

  the algorithm therefore terminates after

  $$k \approx \left\lceil \log_2 \left(\frac{b-a}{\epsilon} \right) \right\rceil$$

  iterations where $\lceil \alpha \rceil$ is the smallest integer greater than or equal to α

MATLAB implementation

```
function [x,k] = bisect(h,a,b,tol)
% assuming h is a defined input  function
% this function returns in x a value such that |x - root| < atol
% and in k the number of iterations required.
ha=h(a);hb=h(b);
if (a >= b) | (ha*hb >= 0) | (tol <= 0)
disp('something wrong with the input: quitting');
x = NaN; k=NaN;
return
end
k = ceil(log2(b-a) - log2(tol));
for i=1:k
x = (a+b)/2;
hx = h(x);
if abs(hx) < eps, k = i; return, end
if ha * hx < 0
b = x; hb = hx;
else
a = x; end
end
end
```

Examples

- for $h(x) = x^3 - 30x^2 + 2552$, starting from interval $[0, 20]$ with a tolerance of 1×10^{-8}, the method converges to $x^\star \approx 11.86150151$ after 31 iterations

  the associated MATLAB script is:

```
format long g
h = @(x) x^3 - 30*x^2 + 2552;
[x,k] = bisect(h,0,20,1.e-8)
```

- for $h(x) = 2.5 \sinh(x/4) - 1$, starting with interval $[-10, 10]$ with a tolerance of 1×10^{-10}, the method converges to $x^\star \approx 1.5601412791$ after 38 iterations

  the associated MATLAB script is:

```
format long g
h =@(x) 2.5 * sinh (x/4) - 1;
[x,k] = bisect(h,-10,10,1.e-10)
```

Outline

- nonlinear equations
- iterative methods
- bisection method
- **Newton method**

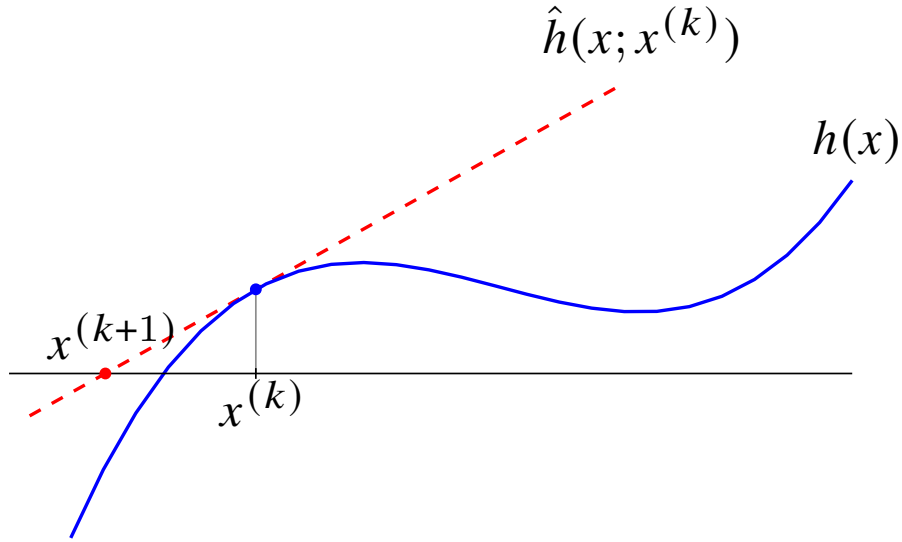
Deriving Newton method

- linearize h (*i.e.*, make affine approximation) around current iterate $x^{(k)}$

$$\hat{h}(x; x^{(k)}) = h(x^{(k)}) + Dh(x^{(k)})(x - x^{(k)})$$

- take solution x of linearized equation $\hat{h}(x; x^{(k)}) = 0$ as the next iterate $x^{(k+1)}$:

$$x = x^{(k)} - Dh(x^{(k)})^{-1} h(x^{(k)})$$



Newton method for nonlinear equations

assume $h : \mathbb{R}^n \longrightarrow \mathbb{R}^n$ is differentiable

---

given a starting point $x^{(0)}$ and solution tolerance ϵ

repeat for $k = 1, 2, \ldots$

1. compute $Dh(x^{(k)})$ and $h(x^{(k)})$
2. **if** $\|h(x^{(k)})\| < \epsilon$, stop and **return** $x^{(k)}$
3. solve for $v^{(k)}$ in $Dh(x^{(k)})v^{(k)} = -h(x^{(k)})$
4. update $x^{(k+1)} = x^{(k)} + v^{(k)}$

- $Dh(x^{(k)})$ is derivative (Jacobian) matrix of h at $x^{(k)}$ assumed to be nonsingular
- also called Newton-Raphson algorithm
- for one equation in one variable, the update reduces to

$$x^{(k+1)} = x^{(k)} - \frac{h(x^{(k)})}{h'(x^{(k)})}$$

Example

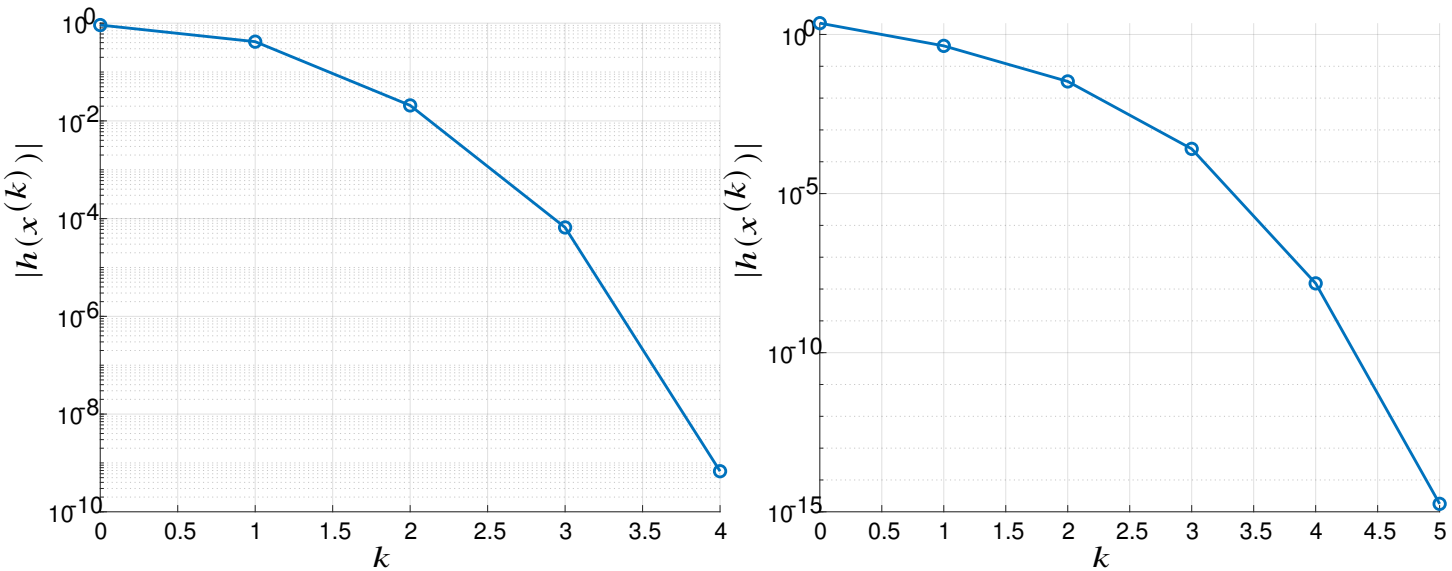
applying Newton's method on $h(x) = 2\cosh(\frac{x}{4}) - x$ gives

$$x^{(k+1)} = x^{(k)} - \frac{2\cosh(x^{(k)}/4) - x^{(k)}}{0.5\sinh(x^{(k)}/4) - 1}$$

where $\cosh(u) = (e^u + e^{-u})/2$ and $\sinh(u) = (e^u - e^{-u})/2$

with tolerance of 1×10^{-8}:

- starting from $x_0 = 4$, 4 iterations are needed to get $x_1^\star = 2.35755106$
- from $x_0 = 10$, 5 iterations are enough to reach $x_2^\star = 8.50719958$



Example

$$h(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} = 0$$

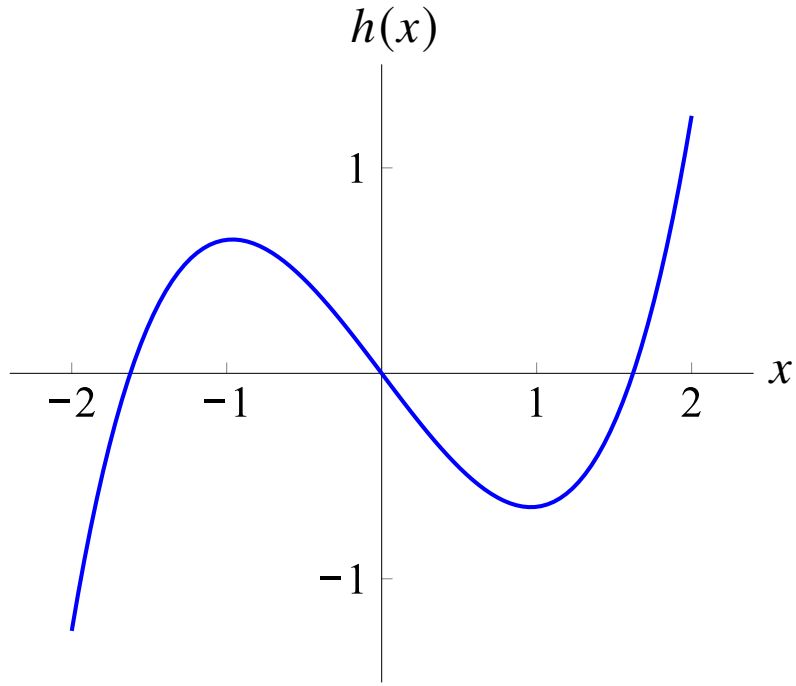
the derivative is $h'(x) = 4/(e^x + e^{-x})^2$, so the Newton iteration is

$$x^{(k+1)} = x^{(k)} - \frac{1}{4}(e^{2x^{(k)}} - e^{-2x^{(k)}}), \quad k = 0, 1, \ldots$$

- method converges rapidly from $x^{(0)} = 0.85$
- does not converge from $x^{(0)} = 1.15$

Example

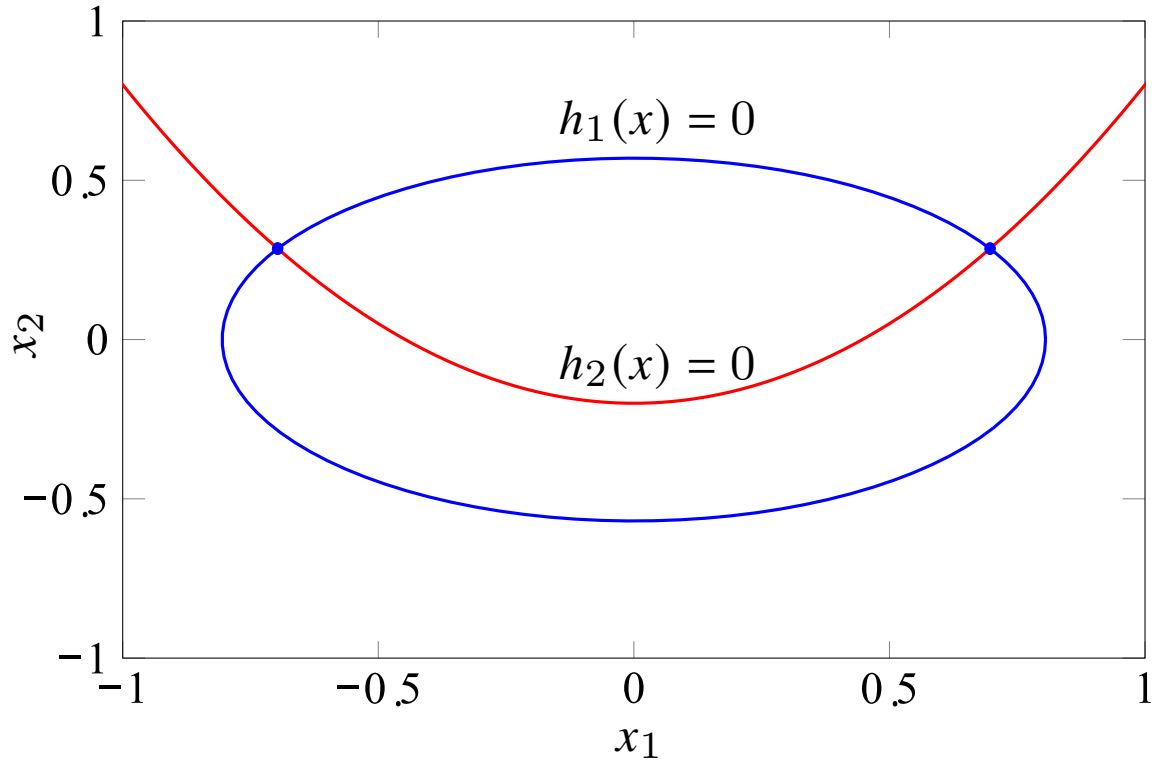
$$h(x) = e^x - e^{-x} - 3x$$



- starting point $x^{(0)} = -1$: converges to $x^\star = -1.62$
- starting point $x^{(0)} = -0.8$: converges to $x^\star = 1.62$
- starting point $x^{(0)} = -0.7$: converges to $x^\star = 0$

Example

$$h_1(x_1, x_2) = \log(x_1^2 + 2x_2^2 + 1) - 0.5 = 0$$
$$h_2(x_1, x_2) = x_2 - x_1^2 + 0.2 = 0$$



two equations in two variables; two solutions $(0.70, 0.29), (-0.70, 0.29)$

Newton iteration

- evaluate $g = h(x)$ and

$$H = Dh(x) = \begin{bmatrix} 2x_1/\left(x_1^2 + 2x_2^2 + 1\right) & 4x_2/\left(x_1^2 + 2x_2^2 + 1\right) \\ -2x_1 & 1 \end{bmatrix}$$

- solve $Hv = -g$ (two linear equations in two variables)
- update $x := x + v$

Results

- $x^{(0)} = (1, 1)$: converges to $x^\star = (0.70, 0.29)$ in about 4 iterations
- $x^{(0)} = (-1, 1)$: converges to $x^\star = (-0.70, 0.29)$ in about 4 iterations
- $x^{(0)} = (1, -1)$ or $x^{(0)} = (-1, -1)$: does not converge

Observations

- Newton's method works well if started near a solution; may not work otherwise
- can converge to different solutions depending on the starting point
- does not necessarily find the solution closest to the starting point

Convergence of Newton's method

if $h(x^\star) = 0$ and $Dh(x^\star)$ is nonsingular, and $x^{(0)}$ is sufficiently close to $x^\star$, then

$$x^{(k)} \longrightarrow x^\star, \quad \|x^{(k+1)} - x^\star\| \leq c\|x^{(k)} - x^\star\|^2$$

for some $c > 0$

- has quadratic convergence when started near a solution
- explains fast convergence when started near solution

Secant method

- consider a single equation with one variable $h(x) = 0$
- the secant method modifies Newton's approach by estimating the derivative:

$$h'(x^{(k)}) \approx \frac{h(x^{(k)}) - h(x^{(k-1)})}{x^{(k)} - x^{(k-1)}}$$

- incorporating this into Newton's formula, we get the secant method equation:

$$x^{(k+1)} = x^{(k)} - \frac{h(x^{(k)})(x^{(k)} - x^{(k-1)})}{h(x^{(k)}) - h(x^{(k-1)})}, \quad k = 1, 2, \ldots$$

Example: $h(x) = 2\cosh(x/4) - x$, tolerance 10^{-8}, $x_0 = 10$

k	0	1	2	3	4	5	6
$h(x^{(k)})$	2.26	−4.76e − 1	−1.64e − 1	2.45e − 2	−9.93e − 4	−5.62e − 6	1.30e − 9

References and further readings

- E. K.P. Chong, Wu-S. Lu, and S. H. Zak, *An Introduction to Optimization: With Applications to Machine Learning*. John Wiley & Sons, 2023. (Ch. 7)

- U. M. Ascher. *A First Course on Numerical Methods*. Society for Industrial and Applied Mathematics, 2011. (Ch. 3)

- L. Vandenberghe. *EE133A lecture notes,* Univ. of California, Los Angeles. (`http://www.seas.ucla.edu/~vandenbe/ee133a.html`)