14. Nonlinear equations

- Newton method for nonlinear equations
- damped Newton method for unconstrained minimization
- Newton method for nonlinear least squares

Set of nonlinear equations

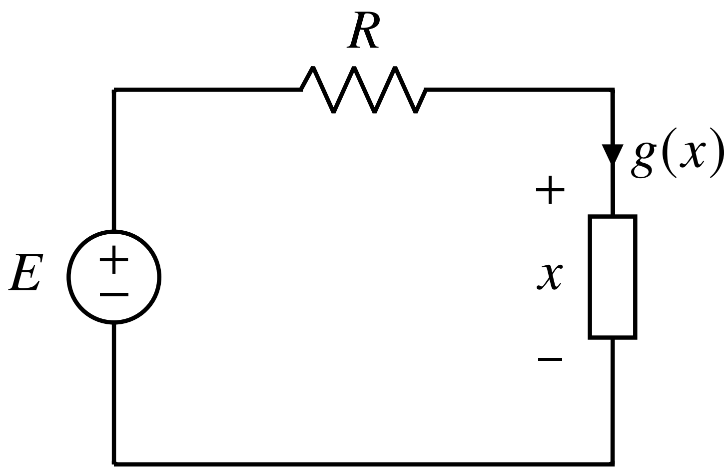
n nonlinear equations in n variables $x_1, x_2, \ldots, x_n$:

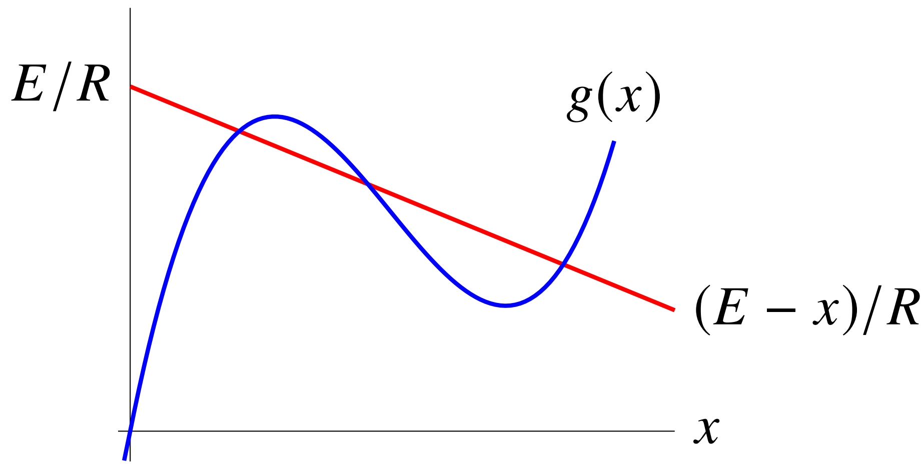
$$\begin{aligned} f_1(x_1, \ldots, x_n) &= 0 \\ f_2(x_1, \ldots, x_n) &= 0 \\ &\vdots \\ f_n(x_1, \ldots, x_n) &= 0 \end{aligned}$$

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

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, \qquad f(x) = \begin{bmatrix} f_1(x_1, \ldots, x_n) \\ f_2(x_1, \ldots, x_n) \\ \vdots \\ f_n(x_1, \ldots, x_n) \end{bmatrix}$$

Example: nonlinear resistive circuit





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

a nonlinear equation in the variable x, with three solutions

Newton method

assume $f : \mathbf{R}^n \to \mathbf{R}^n$ is differentiable

Algorithm: choose $x^{(1)}$ and repeat for $k = 1, 2, \ldots$

$$x^{(k+1)} = x^{(k)} - Df(x^{(k)})^{-1} f(x^{(k)})$$

- each iteration requires one evaluation of $f(x)$ and $Df(x)$
- each iteration requires factorization of the $n \times n$ matrix $Df(x)$
- we assume $Df(x)$ is nonsingular

Interpretation

$$x^{(k+1)} = x^{(k)} - Df(x^{(k)})^{-1} f(x^{(k)})$$

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

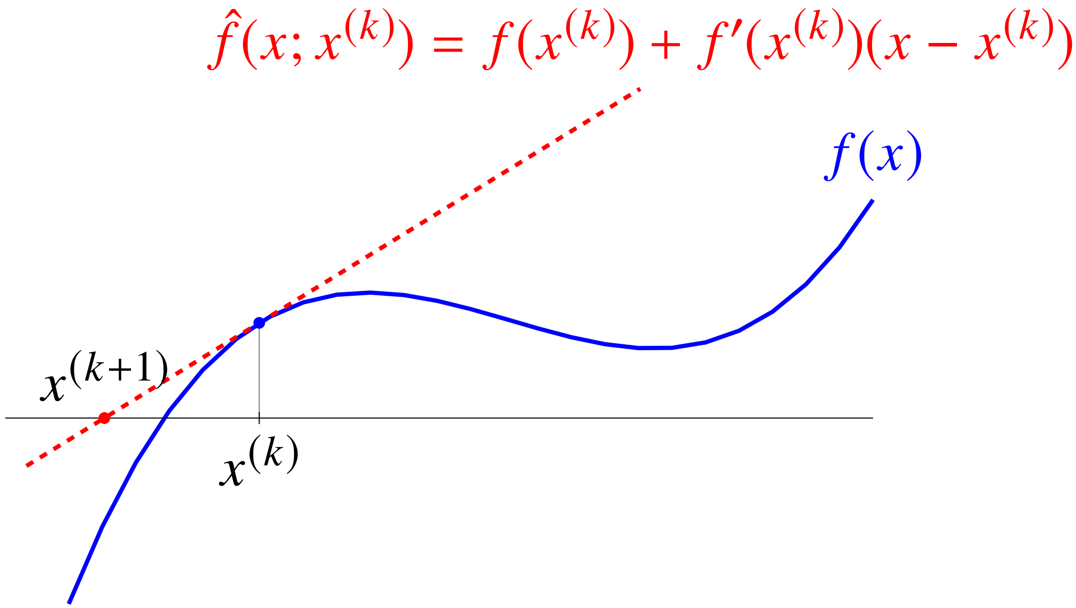
$$\hat{f}(x; x^{(k)}) = f(x^{(k)}) + Df(x^{(k)})(x - x^{(k)})$$

- solve the linearized equation $\hat{f}(x; x^{(k)}) = 0$; the solution is

$$x = x^{(k)} - Df(x^{(k)})^{-1} f(x^{(k)})$$

- take the solution x of the linearized equation as the next iterate $x^{(k+1)}$

One variable



- affine approximation of f around $x^{(k)}$ is

$$\hat{f}(x; x^{(k)}) = f(x^{(k)}) + f'(x^{(k)})(x - x^{(k)})$$

- solve the linearized equation $\hat{f}(x; x^{(k)}) = 0$ and take the solution as $x^{(k+1)}$:

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

Relation to Gauss–Newton method

recall Gauss–Newton method for nonlinear least squares problem

$$\text{minimize} \quad \|f(x)\|^2$$

where f is a differentiable function from $\mathbf{R}^n$ to $\mathbf{R}^m$

- Gauss–Newton update

$$x^{(k+1)} = x^{(k)} - \left(Df(x^{(k)})^T Df(x^{(k)})\right)^{-1} Df(x^{(k)})^T f(x^{(k)})$$

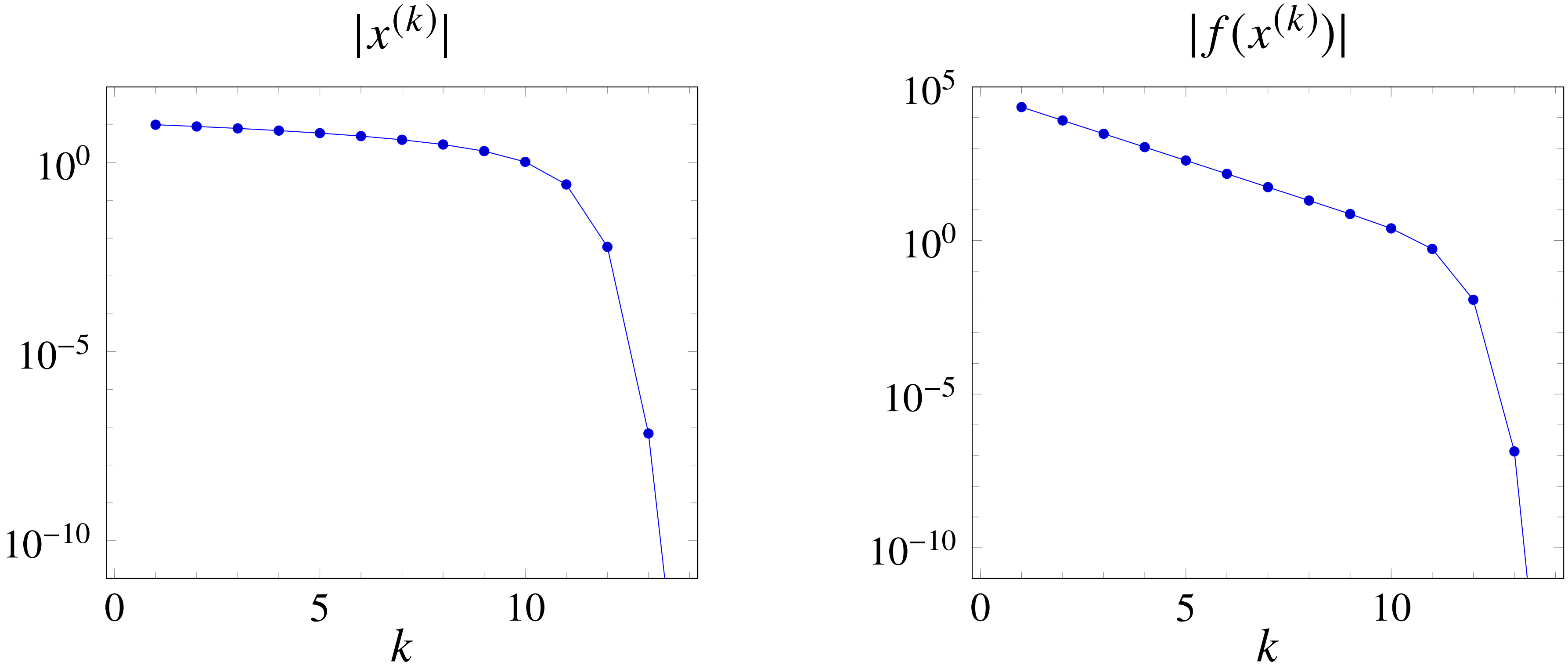
- if $m = n$, then $Df(x)$ is square and this is the Newton update

$$x^{(k+1)} = x^{(k)} - Df(x^{(k)})^{-1} f(x^{(k)})$$

Example 1

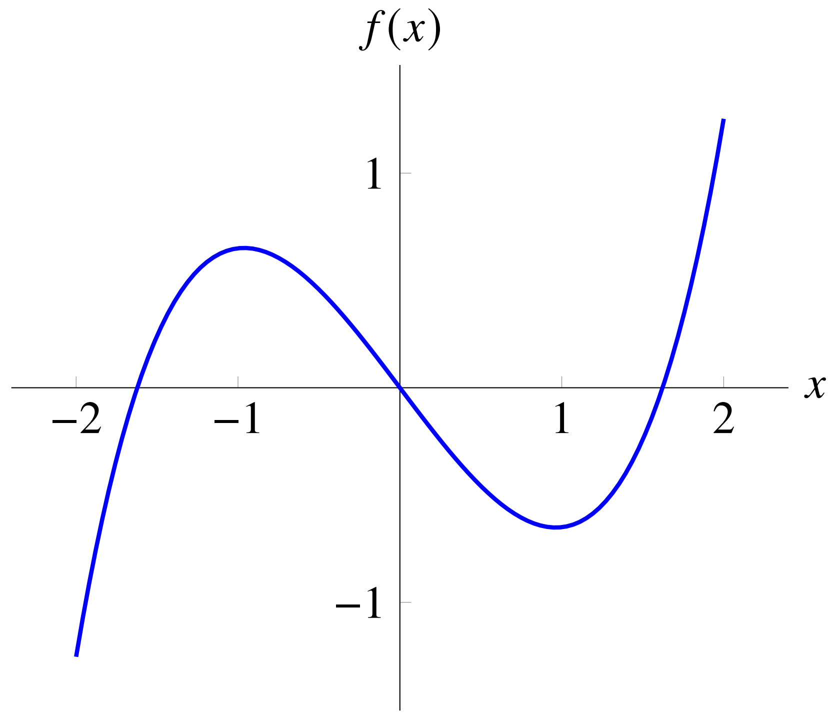
Newton method applied to

$$f(x) = e^x - e^{-x}, \qquad x^{(1)} = 10$$



Example 2

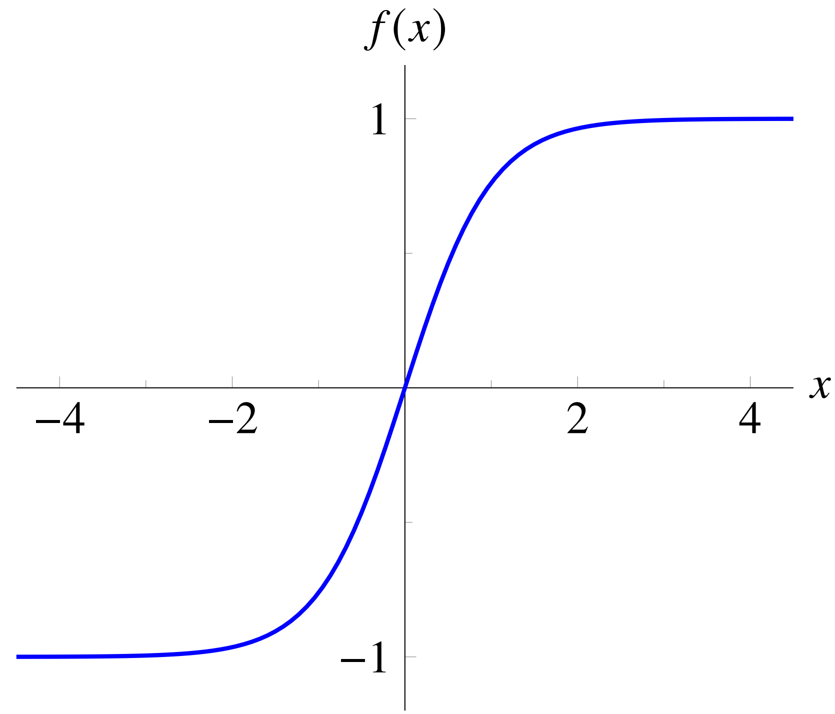
$$f(x) = e^x - e^{-x} - 3x$$



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

Example 3

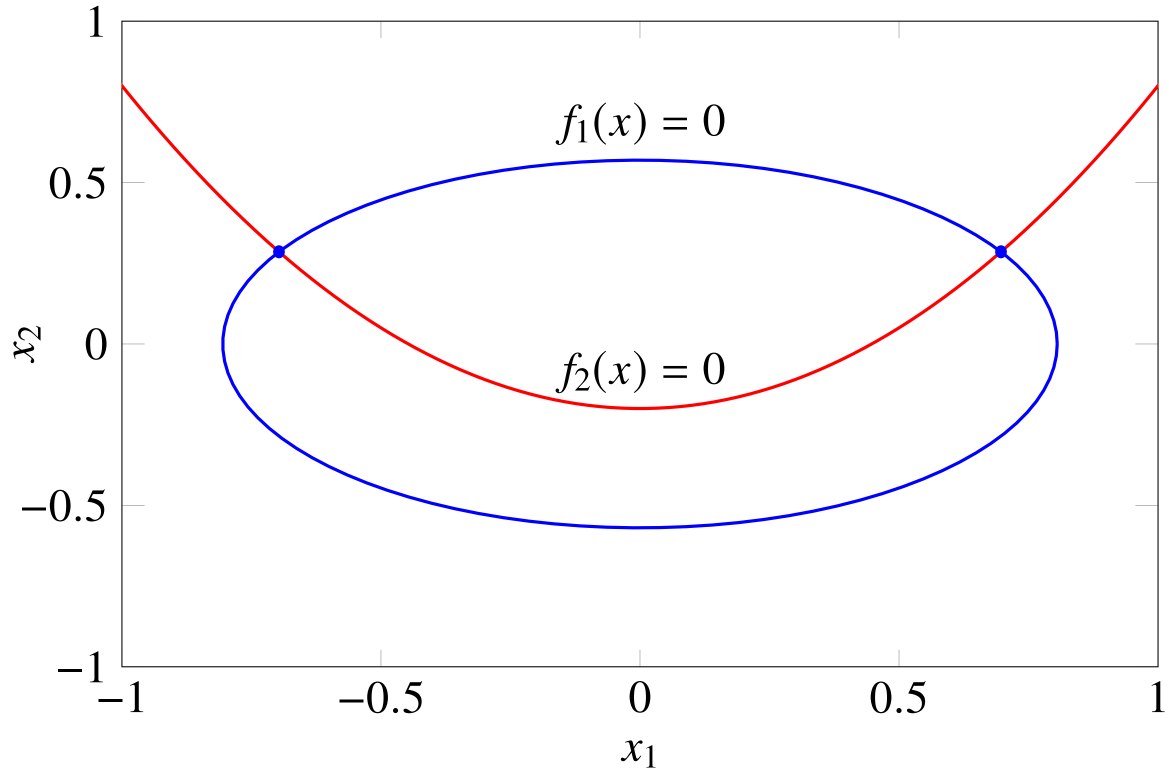
$$f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$



- starting point $x^{(1)} = 0.9$: converges very rapidly to $x^\star = 0$
- starting point $x^{(1)} = 1.1$: does not converge

Example 4

$$
\begin{aligned}
f_1(x_1, x_2) &= \log(x_1^2 + 2x_2^2 + 1) - 0.5 = 0 \\
f_2(x_1, x_2) &= x_2 - x_1^2 + 0.2 = 0
\end{aligned}
$$



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

Example 4

Newton iteration

- evaluate $g = f(x)$ and

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

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

Results

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

Observations

- Newton's method works very 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 $f(x^\star) = 0$ and $Df(x^\star)$ is nonsingular, and $x^{(1)}$ is sufficiently close to $x^\star$, then

$$x^{(k)} \to x^\star, \qquad \|x^{(k+1)} - x^\star\| \le c\|x^{(k)} - x^\star\|^2$$

for some $c > 0$

- this is called quadratic convergence
- explains fast convergence when started near solution

Outline

- Newton's method for sets of nonlinear equations
- **damped Newton for unconstrained minimization**
- Newton method for nonlinear least squares

Unconstrained minimization problem

$$\text{minimize} \quad g(x_1, x_2, \ldots, x_n)$$

g is a function from $\mathbf{R}^n$ to $\mathbf{R}$

- $x = (x_1, x_2, \ldots, x_n)$ is n-vector of optimization *variables*
- $g(x)$ is the *cost function* or *objective function*
- to solve a maximization problem (*i.e.*, maximize $g(x)$), minimize $-g(x)$
- we will assume that g is twice differentiable

Local and global optimum

- $x^\star$ is an *optimal point* (or a *minimum*) if

$$g(x^\star) \leq g(x) \quad \text{for all } x$$

  also called *globally* optimal

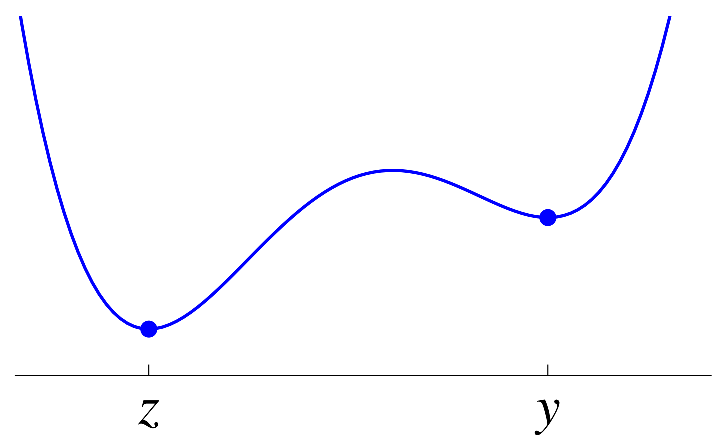
- $x^\star$ is a *locally optimal point* (*local minimum*) if for some $R > 0$

$$g(x^\star) \leq g(x) \quad \text{for all } x \text{ with } \|x - x^\star\| \leq R$$

Example

y is locally optimal

z is (globally) optimal



Gradient and Hessian

Gradient of $g : \mathbf{R}^n \to \mathbf{R}$ at $z \in \mathbf{R}^n$ is the n-vector

$$\nabla g(z) = \left(\frac{\partial g}{\partial x_1}(z),\ \frac{\partial g}{\partial x_2}(z),\ \ldots,\ \frac{\partial g}{\partial x_n}(z)\right)$$

Hessian of g at z: a symmetric $n \times n$ matrix $\nabla^2 g(z)$ with elements

$$\nabla^2 g(z)_{ij} = \frac{\partial^2 g}{\partial x_i \partial x_j}(z)$$

this is also the derivative matrix $Df(z)$ of $f(x) = \nabla g(x)$ at z

Quadratic (second order) approximation of g around z:

$$g_{\mathrm{q}}(x) = g(z) + \nabla g(z)^T (x - z) + \frac{1}{2}(x - z)^T \nabla^2 g(z)(x - z)$$

Examples

Affine function: $g(x) = a^T x + b$

$$\nabla g(x) = a, \qquad \nabla^2 g(x) = 0$$

Quadratic function: $g(x) = x^T P x + q^T x + r$ with P symmetric

$$\nabla g(x) = 2Px + q, \qquad \nabla^2 g(x) = 2P$$

Least squares cost: $g(x) = \|Ax - b\|^2 = x^T A^T A x - 2b^T A x + b^T b$

$$\nabla g(x) = 2A^T A x - 2A^T b, \qquad \nabla^2 g(x) = 2A^T A$$

Properties

Linear combination: if $g(x) = \alpha_1 g_1(x) + \alpha_2 g_2(x)$, then

$$\begin{aligned}\nabla g(x) &= \alpha_1 \nabla g_1(x) + \alpha_2 \nabla g_2(x) \\ \nabla^2 g(x) &= \alpha_1 \nabla^2 g_1(x) + \alpha_2 \nabla^2 g_2(x)\end{aligned}$$

Composition with affine mapping: if $g(x) = h(Cx + d)$, then

$$\begin{aligned}\nabla g(x) &= C^T \nabla h(Cx + d) \\ \nabla^2 g(x) &= C^T \nabla^2 h(Cx + d) C\end{aligned}$$

Example

$$g(x_1, x_2) = e^{x_1+x_2-1} + e^{x_1-x_2-1} + e^{-x_1-1}$$

Gradient

$$\nabla g(x) = \begin{bmatrix} e^{x_1+x_2-1} + e^{x_1-x_2-1} - e^{-x_1-1} \\ e^{x_1+x_2-1} - e^{x_1-x_2-1} \end{bmatrix}$$

Hessian

$$\nabla^2 g(x) = \begin{bmatrix} e^{x_1+x_2-1} + e^{x_1-x_2-1} + e^{-x_1-1} & e^{x_1+x_2-1} - e^{x_1-x_2-1} \\ e^{x_1+x_2-1} - e^{x_1-x_2-1} & e^{x_1+x_2-1} + e^{x_1-x_2-1} \end{bmatrix}$$

Gradient and Hessian via composition property

express g as $g(x) = h(Cx + d)$ with $h(y_1, y_2, y_3) = e^{y_1} + e^{y_2} + e^{y_3}$ and

$$C = \begin{bmatrix} 1 & 1 \\ 1 & -1 \\ -1 & 0 \end{bmatrix}, \qquad d = \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix}$$

Gradient: $\nabla g(x) = C^T \nabla h(Cx + d)$

$$\nabla g(x) = \begin{bmatrix} 1 & 1 & -1 \\ 1 & -1 & 0 \end{bmatrix} \begin{bmatrix} e^{x_1+x_2-1} \\ e^{x_1-x_2-1} \\ e^{-x_1-1} \end{bmatrix}$$

Hessian: $\nabla^2 g(x) = C^T \nabla h^2(Cx + d) C$

$$\nabla^2 g(x) = \begin{bmatrix} 1 & 1 & -1 \\ 1 & -1 & 0 \end{bmatrix} \begin{bmatrix} e^{x_1+x_2-1} & 0 & 0 \\ 0 & e^{x_1-x_2-1} & 0 \\ 0 & 0 & e^{-x_1-1} \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 1 & -1 \\ -1 & 0 \end{bmatrix}$$

Optimality conditions for twice differentiable g

Necessary condition: if $x^\star$ is locally optimal, then

$$\nabla g(x^\star) = 0 \quad \text{and} \quad \nabla^2 g(x^\star) \text{ is positive semidefinite}$$

Sufficient condition: if $x^\star$ satisfies

$$\nabla g(x^\star) = 0 \quad \text{and} \quad \nabla^2 g(x^\star) \text{ is positive definite}$$

then $x^\star$ is locally optimal

Necessary and sufficient condition for convex functions

- g is called *convex* if $\nabla^2 g(x)$ is positive semidefinite everywhere
- if g is convex then $x^\star$ is optimal if and only if $\nabla g(x^\star) = 0$

Examples ($n = 1$)

- $g(x) = \log(e^x + e^{-x})$

$$g'(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}, \qquad g''(x) = \frac{4}{(e^x + e^{-x})^2}$$

$g''(x) \geq 0$ everywhere; $x^\star = 0$ is the unique optimal point

- $g(x) = x^4$

$$g'(x) = 4x^3, \qquad g''(x) = 12x^2$$

$g''(x) \geq 0$ everywhere; $x^\star = 0$ is the unique optimal point

- $g(x) = x^3$

$$g'(x) = 3x^2, \qquad g''(x) = 6x$$

$g'(0) = 0$, $g''(0) = 0$ but $x = 0$ is not locally optimal

Examples

- $g(x) = x^T P x + q^T x + r$ (P is symmetric positive definite)

$$\nabla g(x) = 2Px + q, \qquad \nabla^2 g(x) = 2P$$

$\nabla^2 g(x)$ is positive definite everywhere, hence the unique optimal point is

$$x^\star = -(1/2)P^{-1}q$$

- $g(x) = \|Ax - b\|^2$ (A is a matrix with linearly independent columns)

$$\nabla g(x) = 2A^T Ax - 2A^T b, \qquad \nabla^2 g(x) = 2A^T A$$

$\nabla^2 g(x)$ is positive definite everywhere, hence the unique optimal point is

$$x^\star = (A^T A)^{-1} A^T b$$

Examples

example of page 14.20: we can express $\nabla^2 g(x)$ as

$$\nabla^2 g(x) = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -1 & 0 \end{bmatrix} \begin{bmatrix} e^{x_1+x_2-1} & 0 & 0 \\ 0 & e^{x_1-x_2-1} & 0 \\ 0 & 0 & e^{-x_1-1} \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 1 & -1 \\ 1 & 0 \end{bmatrix}$$

this shows that $\nabla^2 g(x)$ is positive definite for all x

therefore $x^\star$ is optimal if and only if

$$\nabla g(x^\star) = \begin{bmatrix} e^{x_1^\star+x_2^\star-1} + e^{x_1^\star-x_2^\star-1} - e^{-x_1^\star-1} \\ e^{x_1^\star+x_2^\star-1} - e^{x_1^\star-x_2^\star-1} \end{bmatrix} = 0$$

two nonlinear equations in two variables

Newton's method for minimizing a convex function

if $\nabla^2 g(x)$ is positive definite everywhere, we can minimize $g(x)$ by solving

$$\nabla g(x) = 0$$

Algorithm: choose $x^{(1)}$ and repeat for $k = 1, 2, \ldots$

$$x^{(k+1)} = x^{(k)} - \nabla^2 g(x^{(k)})^{-1} \nabla g(x^{(k)})$$

- $v = -\nabla^2 g(x)^{-1} \nabla g(x)$ is called the *Newton step* at x
- converges if started sufficiently close to the solution
- Newton step is computed by a Cholesky factorization of the Hessian

Interpretations of Newton step

Affine approximation of gradient

- affine approximation of $f(x) = \nabla g(x)$ around $x^{(k)}$ is

$$\hat{f}(x; x^{(k)}) = \nabla g(x^{(k)}) + \nabla^2 g(x^{(k)})(x - x^{(k)})$$

- Newton update $x^{(k+1)}$ is solution of linear equation $\hat{f}(x; x^{(k)}) = 0$

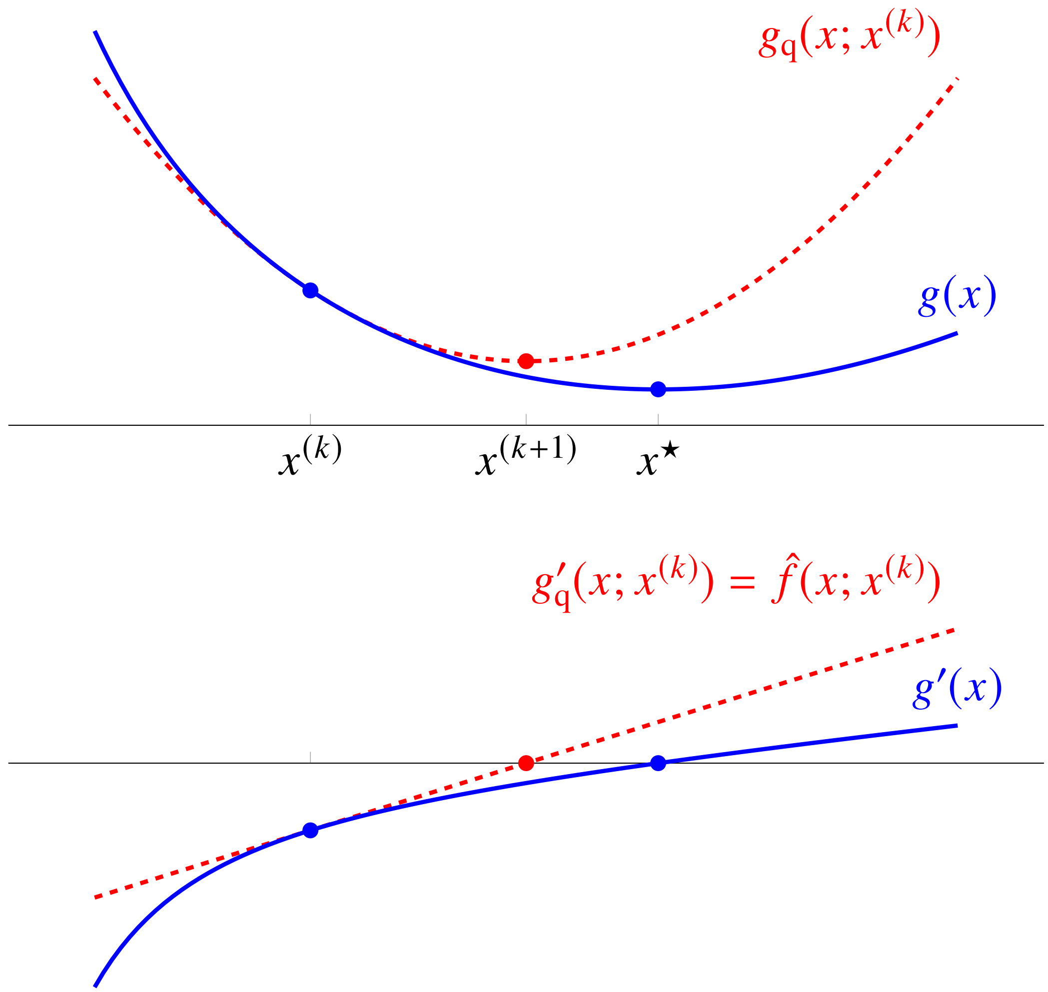
Quadratic approximation of function

- quadratic approximation of $g(x)$ around $x^{(k)}$ is

$$g_{\mathrm{q}}(x; x^{(k)}) = g(x^{(k)}) + \nabla g(x^{(k)})^T (x - x^{(k)}) + \frac{1}{2}(x - x^{(k)})^T \nabla^2 g(x^{(k)})(x - x^{(k)})$$

- Newton update $x^{(k+1)}$ minimizes $g_{\mathrm{q}}(x; x^{(k)})$ (satisfies $\nabla g_{\mathrm{q}}(x; x^{(k)}) = 0$)

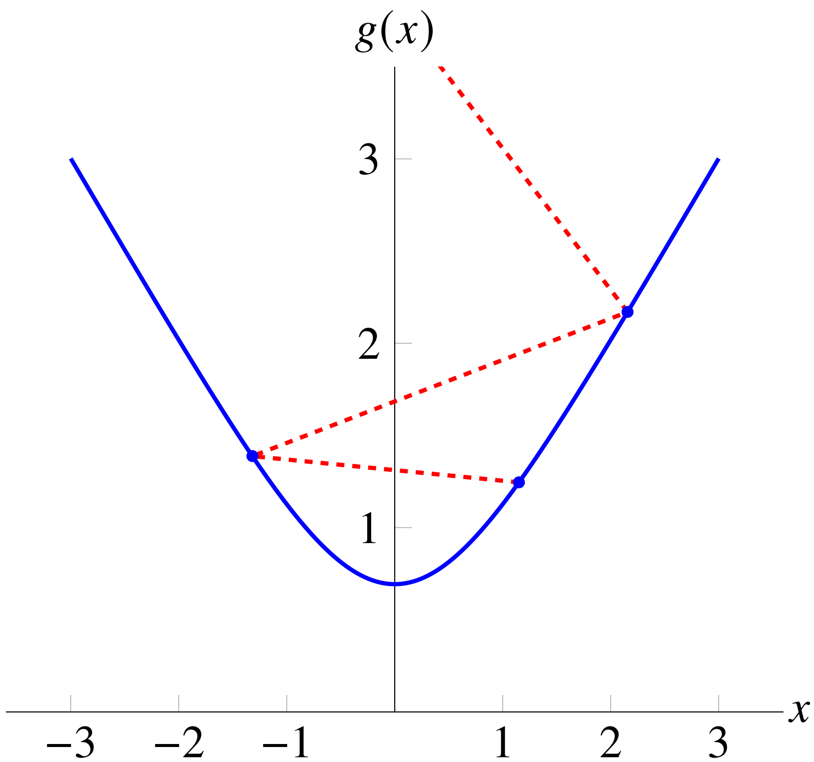
Example ($n = 1$)

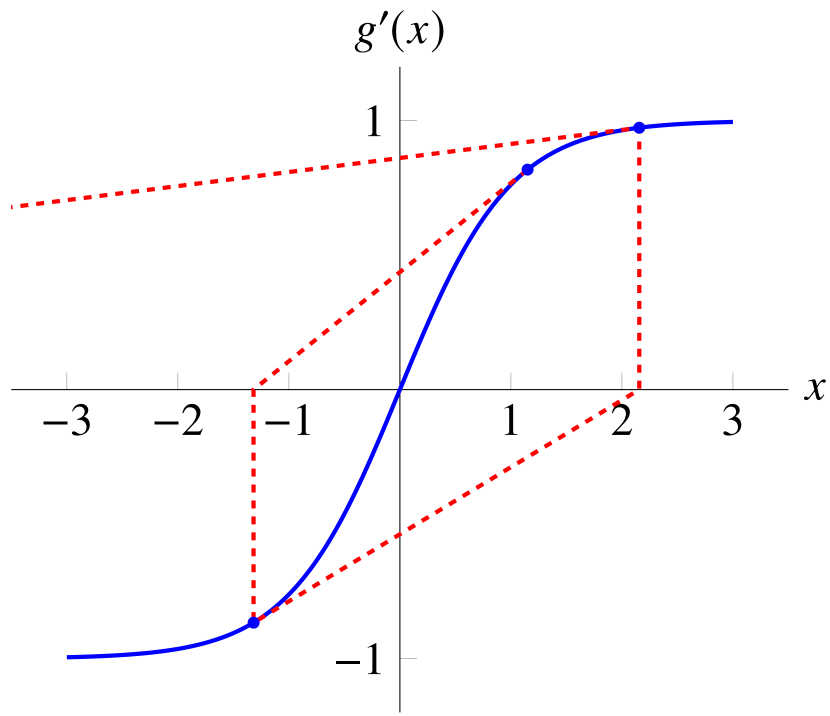


$$g_{\mathrm{q}}(x; x^{(k)}) = g(x^{(k)}) + g'(x^{(k)})(x - x^{(k)}) + \frac{g''(x^{(k)})}{2}(x - x^{(k)})^2$$

Example

$$g(x) = \log(e^x + e^{-x}), \qquad g'(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}, \qquad g''(x) = \frac{4}{(e^x + e^{-x})^2}$$





does not converge when started at $x^{(1)} = 1.15$

Damped Newton method

- use *damped* update $x^{(k+1)} = x^{(k)} - t\nabla^2 g(x^{(k)})^{-1}\nabla g(x^{(k)})$
- choose *step size* t so that $g(x^{(k+1)}) < g(x^{(k)})$

Algorithm: choose $x^{(1)}$ and repeat for $k = 1, 2, \ldots$

1. compute Newton step $v = -\nabla^2 g(x^{(k)})^{-1}\nabla g(x^{(k)})$
2. find largest t in $\{1,\ 0.5,\ 0.5^2,\ 0.5^3, \ldots\}$ that satisfies

$$g(x^{(k)} + tv) \leq g(x^{(k)}) + \alpha t \nabla g(x^{(k)})^T v$$

   and take $x^{(k+1)} = x^{(k)} + tv$

- α is an algorithm parameter (small and positive, *e.g.*, $\alpha = 0.01$)
- step 2 in algorithm is called *line search*

Interpretation of line search

to determine a suitable step size, consider the function $h : \mathbf{R} \to \mathbf{R}$

$$h(t) = g(x^{(k)} + tv)$$

$x^{(k)}$ is the current iterate; v is the Newton step at $x^{(k)}$

- $h'(0) = \nabla g(x^{(k)})^T v$ is the *directional derivative* of g at $x^{(k)}$ in direction v
- affine approximation of h at $t = 0$ is

$$\hat{h}(t) = h(0) + h'(0)t = g(x^{(k)}) + t\nabla g(x^{(k)})^T v$$

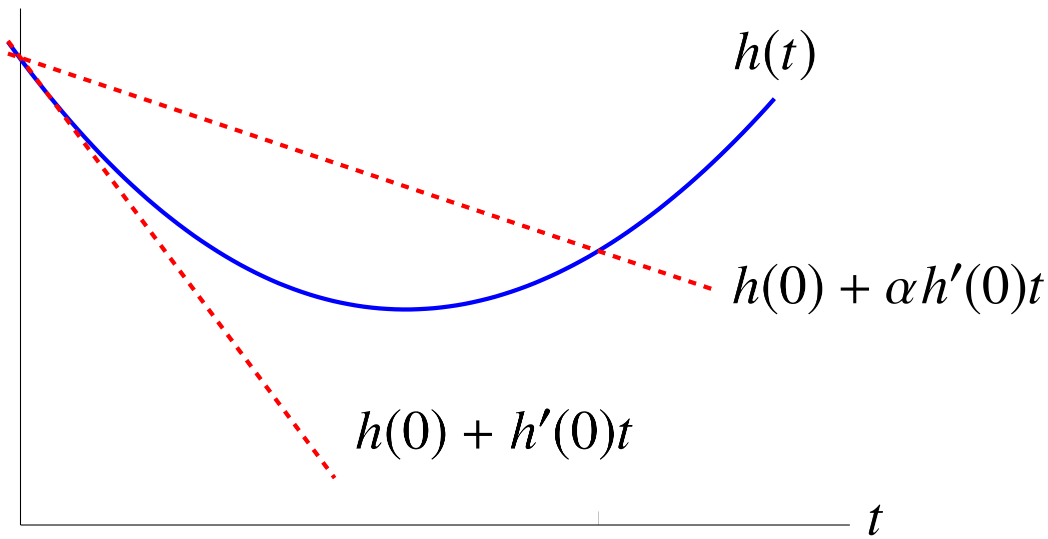
- condition $g(x^{(k)} + tv) \le g(x^{(k)}) + \alpha t \nabla g(x^{(k)})^T v$ means that t is accepted if

$$h(t) - h(0) \le \alpha(\hat{h}(t) - h(0))$$

actual decrease $h(t) - h(0)$ is at least α times what is expected based on $\hat{h}$

Interpretation of line search

start with $t = 1$; divide t by two until $h(t) \leq h(0) + \alpha h'(0)t$

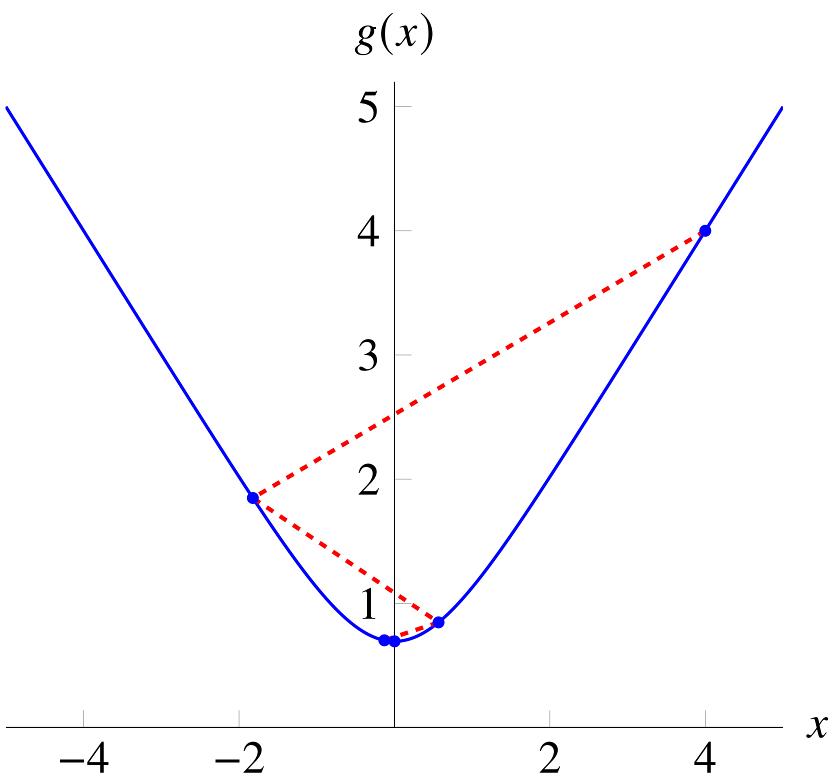


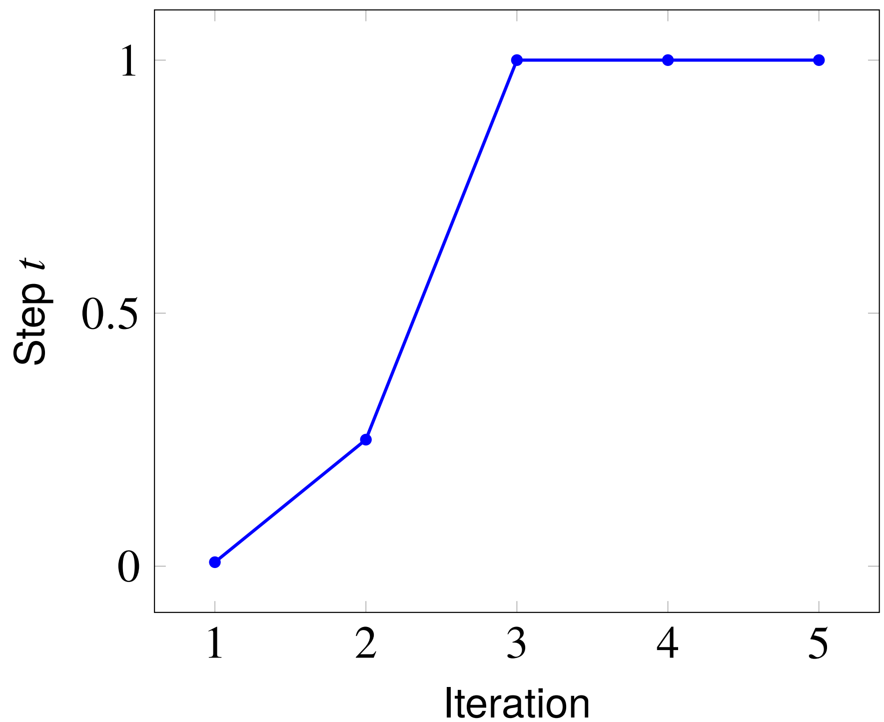
- works if $h'(0) = \nabla g(x)^T v < 0$ (v is a *descent direction*)
- if $\nabla^2 g(x^{(k)})$ is positive definite, the Newton step is a descent direction

$$h'(0) = \nabla g(x^{(k)})^T v = -v^T \nabla^2 g(x^{(k)}) v < 0$$

Example

$$g(x) = \log(e^x + e^{-x}), \qquad x^{(0)} = 4$$





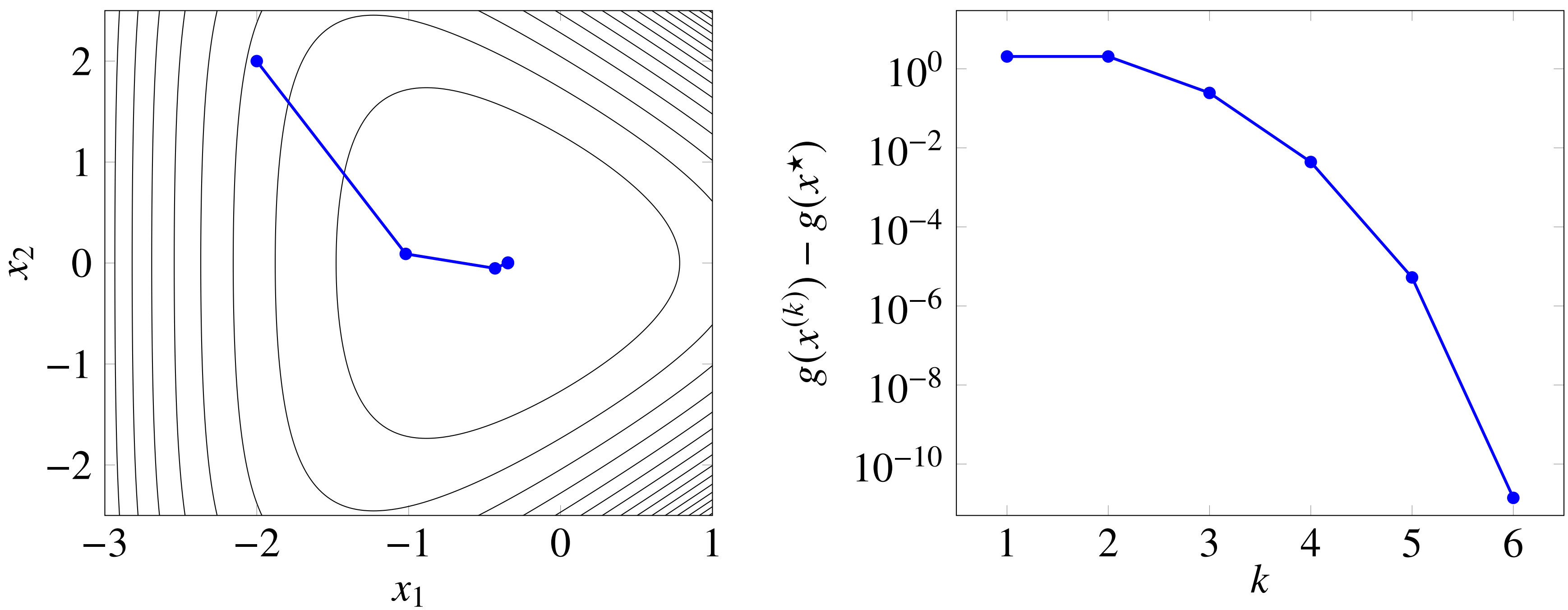
close to the solution: very fast convergence, no backtracking steps

Example

example of page 14.20

$$g(x_1, x_2) = e^{x_1+x_2-1} + e^{x_1-x_2-1} + e^{-x_1-1}$$

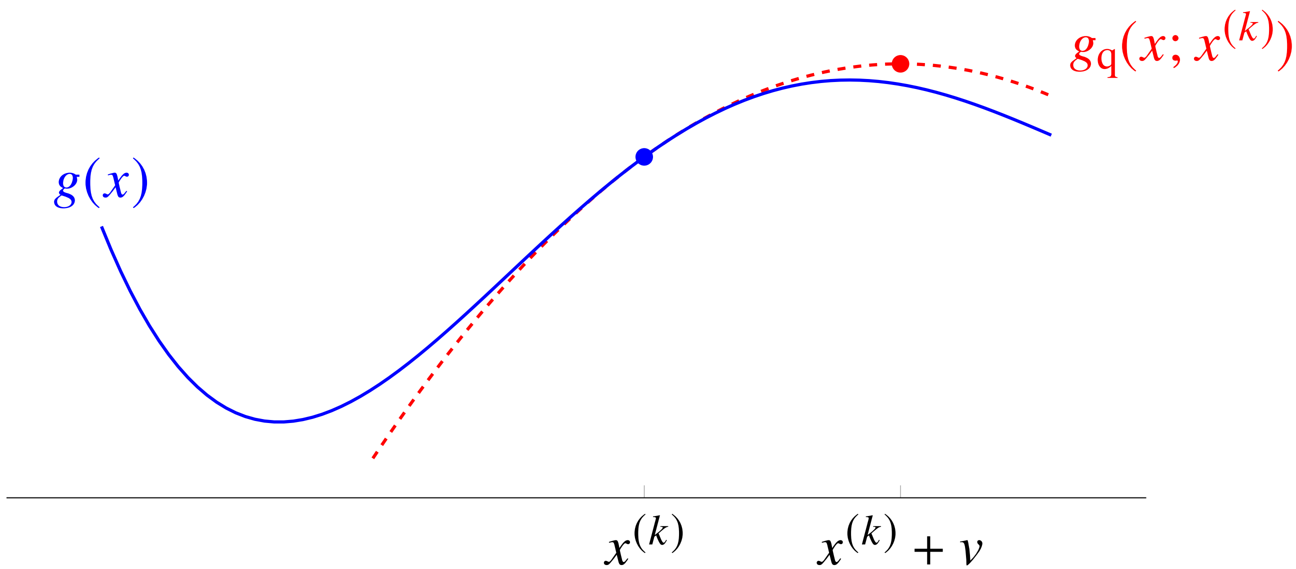
damped Newton method started at $x = (-2, 2)$



Newton method for nonconvex functions

if $\nabla^2 g(x^{(k)})$ is not positive definite, it is possible that Newton step v satisfies

$$\nabla g(x^{(k)})^T v = -\nabla g(x^{(k)})^T \nabla^2 g(x^{(k)})^{-1} \nabla g(x^{(k)}) > 0$$



- if Newton step is not descent direction, replace it with descent direction
- simplest choice is $v = -\nabla g(x^{(k)})$; practical methods make other choices

Outline

- Newton's method for sets of nonlinear equations
- damped Newton for unconstrained minimization
- **Newton method for nonlinear least squares**

Hessian of nonlinear least squares cost

$$g(x) = \|f(x)\|^2 = \sum_{i=1}^{m} f_i(x)^2$$

- gradient (from page 13.14):

$$\nabla g(x) = 2\sum_{i=1}^{m} f_i(x)\nabla f_i(x) = 2Df(x)^T f(x)$$

- second derivatives:

$$\frac{\partial^2 g}{\partial x_j \partial x_k}(x) = 2\sum_{i=1}^{m}\left(\frac{\partial f_i}{\partial x_j}(x)\frac{\partial f_i}{\partial x_k}(x) + f_i(x)\frac{\partial^2 f_i}{\partial x_j \partial x_k}(x)\right)$$

- Hessian

$$\nabla^2 g(x) = 2Df(x)^T Df(x) + 2\sum_{i=1}^{m} f_i(x)\nabla^2 f_i(x)$$

Newton and Gauss–Newton steps

(Undamped) Newton step at $x = x^{(k)}$:

$$
\begin{aligned}
v_{\mathrm{nt}} &= -\nabla^2 g(x)^{-1} \nabla g(x) \\
&= -\left(Df(x)^T Df(x) + \sum_{i=1}^{m} f_i(x) \nabla^2 f_i(x) \right)^{-1} Df(x)^T f(x)
\end{aligned}
$$

Gauss–Newton step at $x = x^{(k)}$ (from pages 13.17):

$$
v_{\mathrm{gn}} = -\left(Df(x)^T Df(x) \right)^{-1} Df(x)^T f(x)
$$

- can be written as $v_{\mathrm{gn}} = -H_{\mathrm{gn}}^{-1} \nabla g(x)$ where $H_{\mathrm{gn}} = 2Df(x)^T Df(x)$
- H_{gn} is the Hessian without the term $\sum_i f_i(x) \nabla^2 f_i(x)$

Comparison

Newton step

- requires second derivatives of f
- not always a descent direction ($\nabla^2 g(x)$ is not necessarily positive definite)
- fast convergence near local minimum

Gauss–Newton step

- does not require second derivatives
- a descent direction (if columns of $Df(x)$ are linearly independent):

$$\nabla g(x)^T v_{\text{gn}} = -2v_{\text{gn}}^T Df(x)^T Df(x) v_{\text{gn}} < 0 \quad \text{if } v_{\text{gn}} \neq 0$$

- local convergence to $x^\star$ is similar to Newton method if

$$\sum_{i=1}^{m} f_i(x^\star)\nabla^2 f_i(x^\star)$$

is small (*e.g.*, $f(x^\star)$ is small, or f is nearly affine around $x^\star$)