9. Unconstrained minimization

- terminology and assumptions
- gradient descent method
- steepest descent method
- Newton's method
- self-concordant functions
- implementation

Unconstrained minimization

$$\text{minimize} \quad f(x)$$

- f is convex and twice continuously differentiable; hence $\operatorname{dom} f$ is an open set
- we assume the optimal value $p^\star = \inf_x f(x)$ is finite and attained

Unconstrained minimization methods

- produce a sequence of points $x^{(k)} \in \operatorname{dom} f$, $k = 0, 1, \ldots$, with

$$f(x^{(k)}) \to p^\star$$

- can be interpreted as iterative methods for solving optimality condition

$$\nabla f(x^\star) = 0$$

Initial point

algorithms in this chapter require a starting point $x^{(0)}$ that satisfies two conditions

- feasiblity: $x^{(0)} \in \operatorname{dom} f$
- the initial sublevel set S is closed, where

$$S = \{x \mid f(x) \leq f(x^{(0)})\}$$

2nd condition is often hard to verify, except when *all* sublevel sets of f are closed

Closed function: a function with closed sublevel sets

- equivalent to property that the epigraph is a closed set
- convex f is closed if $\operatorname{dom} f = \mathbf{R}^n$
- convex f is closed if $\operatorname{dom} f$ is open and $f(x) \to \infty$ as $x \to \operatorname{bd} \operatorname{dom} f$

Examples

three convex differentiable functions

$$
\begin{aligned}
f(x) &= \log\left(\sum_{i=1}^m \exp(a_i^T x + b_i)\right), && \mathbf{dom}\, f = \mathbf{R}^n \\
g(x) &= -\sum_{i=1}^m \log(b_i - a_i^T x), && \mathbf{dom}\, g = \{x \mid a_i^T x < b_i,\ i = 1, \ldots, m\} \\
h(x) &= x_1^2 + x_2^2, && \mathbf{dom}\, h = \{(x_1, x_2) \mid x_1 > 1\}
\end{aligned}
$$

- f is closed; every $x^{(0)} \in \mathbf{R}^n$ satisfies closed initial sublevel set condition
- g is closed; every $x^{(0)} \in \mathbf{dom}\, g$ satisfies closed initial sublevel set condition
- h is not closed; no $x^{(0)}$ satisfies closed initial sublevel set condition

Strong convexity

many convergence results in this chapter require *strong convexity*

- f is strongly convex if there exists an $m > 0$ such that

$$f(x) - \frac{m}{2} x^T x \quad \text{is a convex function}$$

- equivalent definition for differentiable function: $\mathbf{dom}\, f$ is convex and

$$f(y) \geq f(x) + \nabla f(x)^T (y - x) + \frac{m}{2} \|y - x\|_2^2 \quad \text{for all } x, y \in \mathbf{dom}\, f \qquad (1)$$

- equivalent definition for twice differentiable function: $\mathbf{dom}\, f$ is convex and

$$\nabla^2 f(x) \succeq mI \quad \text{for all } x \in \mathbf{dom}\, f$$

Implications of strong convexity

- sublevel sets are bounded (follows from (1))
- optimal value $p^\star$ is finite and

$$f(x) - p^\star \leq \frac{1}{2m}\|\nabla f(x)\|_2^2 \quad \text{for all } x \in \mathbf{dom}\, f \qquad (2)$$

useful as stopping criterion (if you know m)

Proof: from (1)

$$\begin{aligned} p^\star &= \inf_y f(y) \\ &\geq \inf_y \left(f(x) + \nabla f(x)^T(y-x) + \frac{m}{2}\|y-x\|_2^2\right) \\ &= f(x) - \frac{1}{2m}\|\nabla f(x)\|_2^2 \end{aligned}$$

minimizer on second line is $y = x - (1/m)\nabla f(x)$

Descent methods

in a *descent method,* the iterates satisfy

$$f(x^{(k+1)}) < f(x^{(k)})$$

- the algorithms discussed in this chapter are of this type
- for convex f, requires that $v = x^{(k+1)} - x^{(k)}$ is a *descent direction, i.e.*,

$$\nabla f(x^{(k)})^T v < 0 \qquad (3)$$

(the *directional derivative* at $x^{(k)}$ in the direction v is negative)

necessity of (3) can be seen from the inequality

$$f(x^{(k+1)}) \geq f(x^{(k)}) + \nabla f(x^{(k)})^T v$$

General outline of a descent method

- different notation styles will be used for the update:

$$x^{(k+1)} = x^{(k)} + t^{(k)}\Delta x^{(k)}, \qquad x^+ = x + t\Delta x, \qquad x := x + t\Delta x$$

- Δx is the *step* or *search direction*
- t is the *step size* or *step length*

Descent method

given a starting point $x \in \mathrm{dom}\, f$
repeat
1. *search direction:* determine a descent direction Δx
2. *line search:* choose a step size $t > 0$
3. *update:* $x := x + t\Delta x$

until stopping criterion is satisfied

next, we discuss step 2 (line search) and then step 1 (choices for Δx)

Line search types

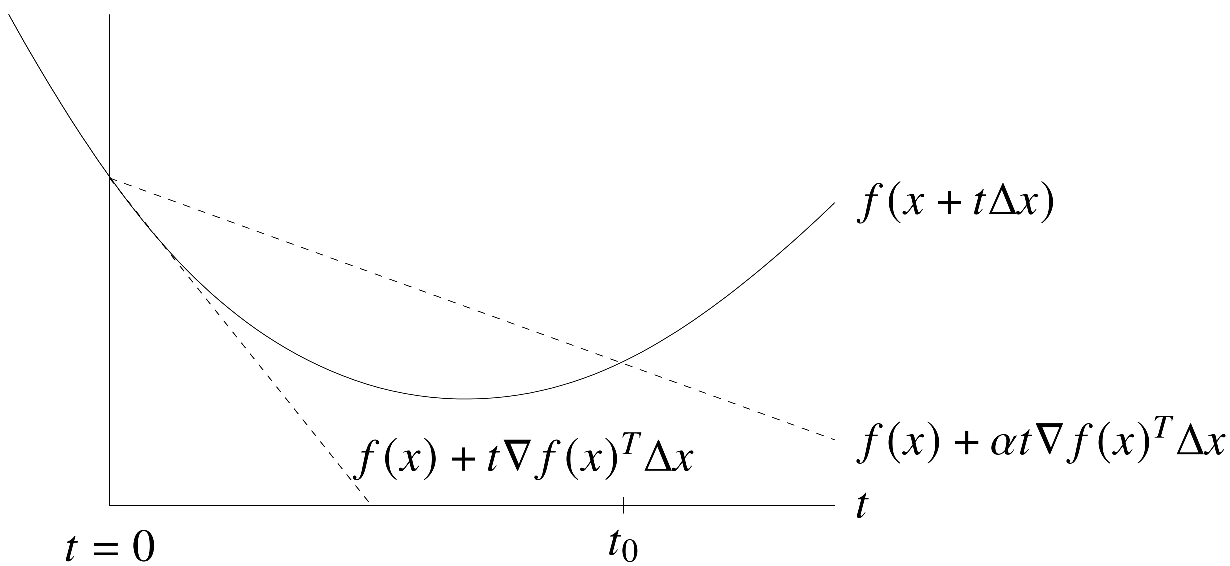
Exact line search: $t = \operatorname{argmin}_{t>0} f(x + t\Delta x)$

Backtracking line search

- starting at $t = 1$, repeat $t := \beta t$ until

$$f(x + t\Delta x) < f(x) + \alpha t \nabla f(x)^T \Delta x$$

- $\alpha \in (0, 1/2)$ and $\beta \in (0, 1)$ are algorithm parameters
- in the example of the figure, we backtrack until $t \leq t_0$



Gradient descent method

Gradient descent: the general descent method of page 9.8 with $\Delta x = -\nabla f(x)$

given: a starting point $x \in \mathbf{dom}\, f$
repeat
1. $\Delta x := -\nabla f(x)$
2. *line search:* choose step size t via exact or backtracking line search
3. *update:* $x := x + t\Delta x$

until stopping criterion is satisfied

- stopping criterion usually of the form $\|\nabla f(x)\|_2 \leq \epsilon$
- convergence result: for strongly convex f,

$$f(x^{(k)}) - p^\star \leq c^k (f(x^{(0)}) - p^\star)$$

  $c \in (0, 1)$ depends on m, $x^{(0)}$, line search type
- iteration is simple and inexpensive, but convergence is often very slow

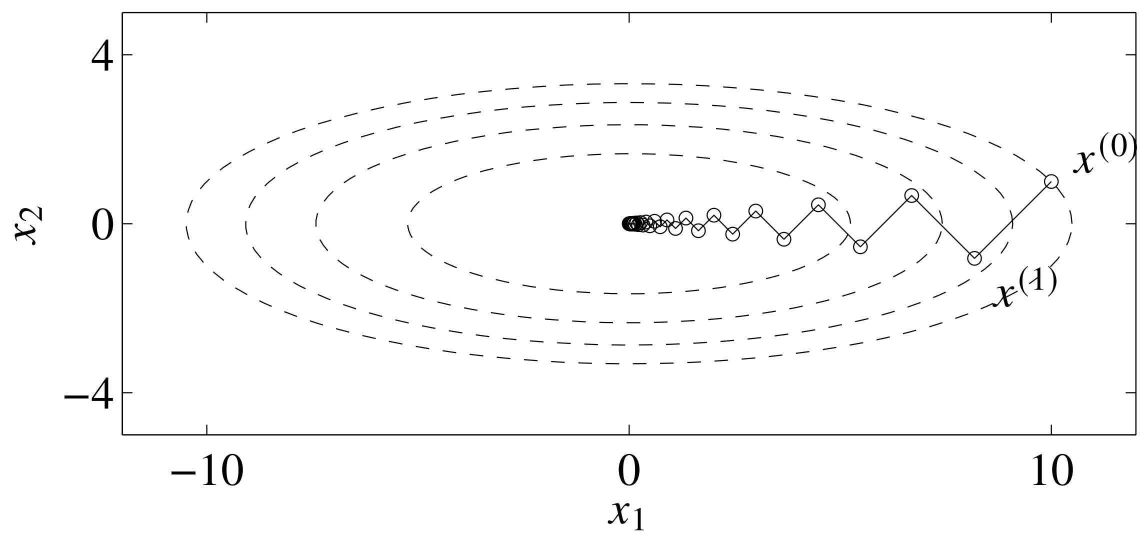
Quadratic problem in $\mathbf{R}^2$

$$f(x) = \tfrac{1}{2}(x_1^2 + \gamma x_2^2) \qquad\qquad (\gamma > 0)$$

with exact line search, starting at $x^{(0)} = (\gamma, 1)$:

$$x_1^{(k)} = \gamma \left(\frac{\gamma - 1}{\gamma + 1}\right)^k, \qquad x_2^{(k)} = \left(-\frac{\gamma - 1}{\gamma + 1}\right)^k$$

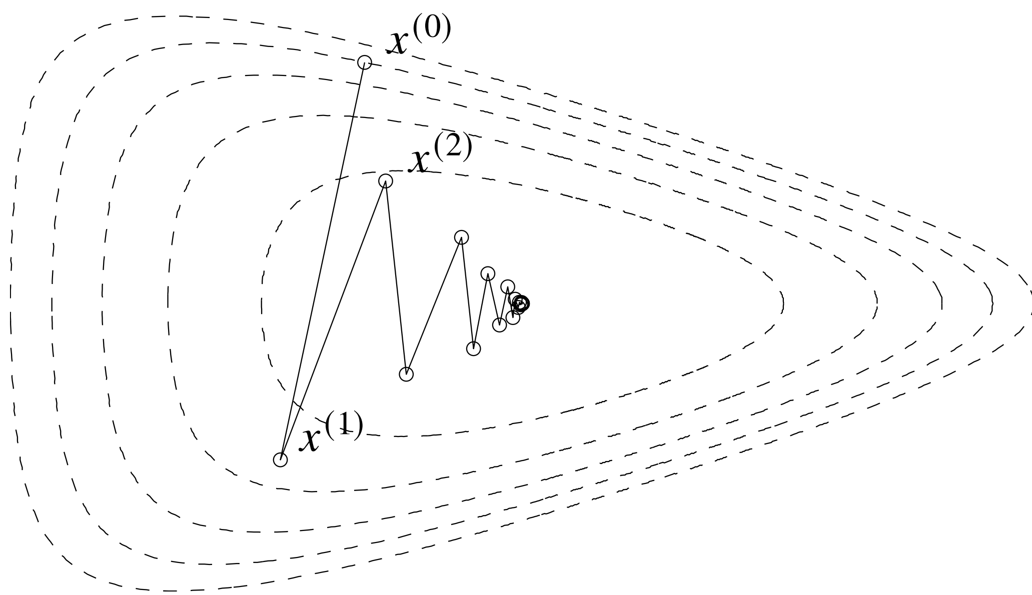
- very slow if $\gamma \gg 1$ or $\gamma \ll 1$
- example for $\gamma = 10$:



Nonquadratic example

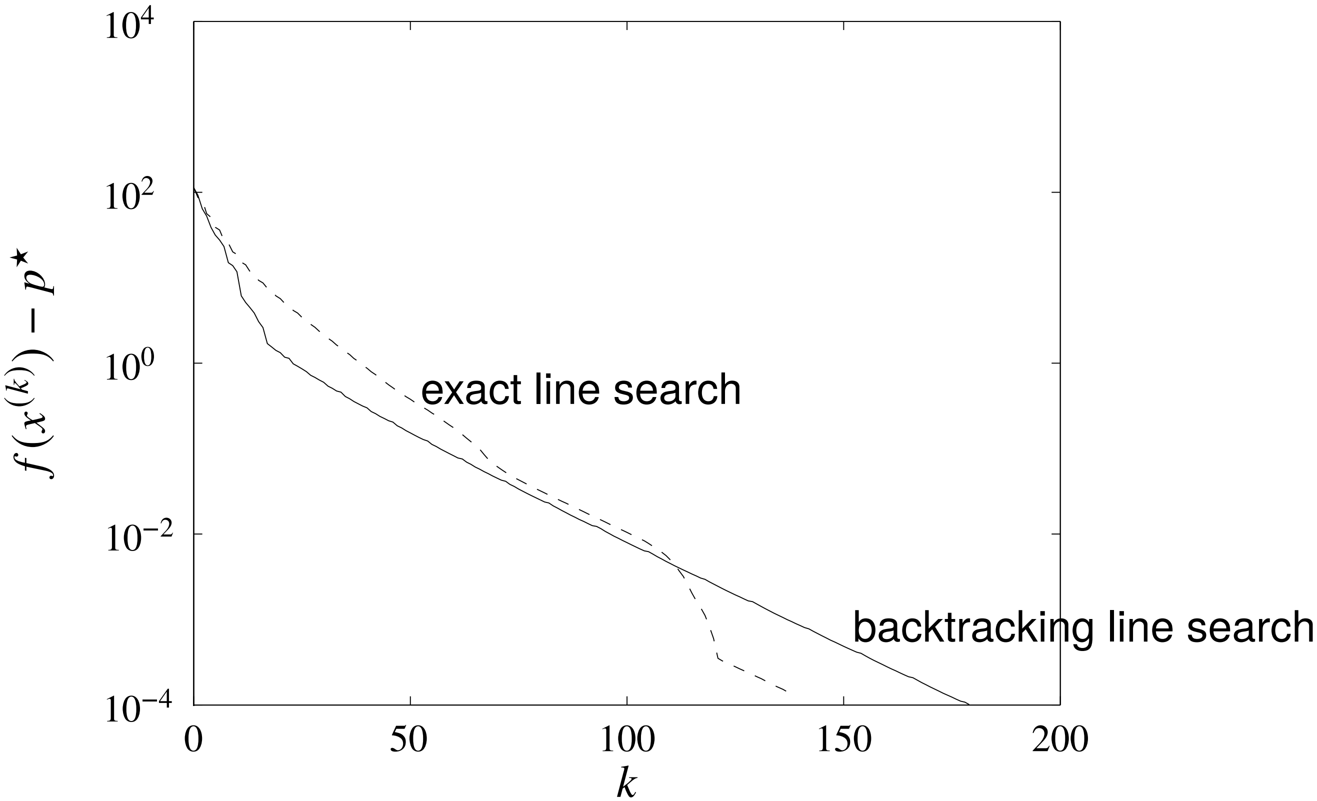
$$f(x_1, x_2) = e^{x_1+3x_2-0.1} + e^{x_1-3x_2-0.1} + e^{-x_1-0.1}$$

figure shows iterates with backtracking line search



Example in $\mathbf{R}^{100}$

$$f(x) = c^T x - \sum_{i=1}^{500} \log(b_i - a_i^T x)$$



shows linear convergence, *i.e.*, a straight line on a semilog plot

Steepest descent method

Normalized steepest descent direction (at x, for norm $\|\cdot\|$)

$$\Delta x_{\mathrm{nsd}} = \operatorname{argmin}\{\nabla f(x)^T v \mid \|v\| = 1\}$$

- direction Δx_{nsd} is unit-norm step with most negative directional derivative
- directional derivative in this direction is $\nabla f(x)^T \Delta x_{\mathrm{nsd}} = -\|\nabla f(x)\|_*$

  (recall definition of dual norm $\|v\|_* = \sup_{\|u\|=1} v^T u$)

(Unnormalized) steepest descent direction

$$\Delta x_{\mathrm{sd}} = \|\nabla f(x)\|_* \Delta x_{\mathrm{nsd}}$$

multiple of Δx_{nsd}, scaled to make $\nabla f(x)^T \Delta x_{\mathrm{sd}} = -\|\nabla f(x)\|_*^2$

Steepest descent method

- general descent method of page 9.8 with $\Delta x = \Delta x_{\mathrm{sd}}$
- convergence properties are similar to gradient descent

Steepest descent direction for quadratic norm

- for Euclidean norm ($\|\cdot\| = \|\cdot\|_* = \|\cdot\|_2$)

$$\Delta x_{\mathrm{nsd}} = \frac{-\nabla f(x)}{\|\nabla f(x)\|_2}, \qquad \Delta x_{\mathrm{sd}} = -\nabla f(x)$$

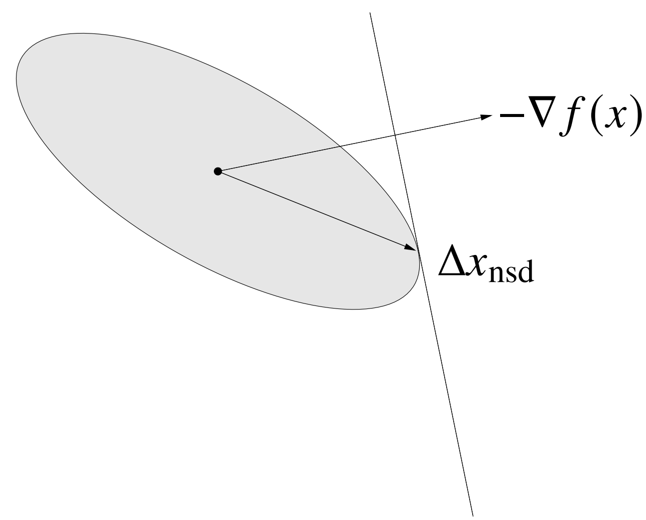
- as an extension, define quadratic norm and dual norm (for $P \in \mathbf{S}^n_{++}$)

$$\|x\| = (x^T P x)^{1/2} = \|P^{1/2} x\|_2, \qquad \|y\|_* = (y^T P^{-1} y)^{1/2} = \|P^{-1/2} y\|_2$$

- normalized and unnormalized steepest descent directions for this norm are

$$\Delta x_{\mathrm{nsd}} = \frac{-P^{-1}\nabla f(x)}{\|P^{-1/2}\nabla f(x)\|_2}$$

$$\Delta x_{\mathrm{sd}} = -P^{-1}\nabla f(x)$$

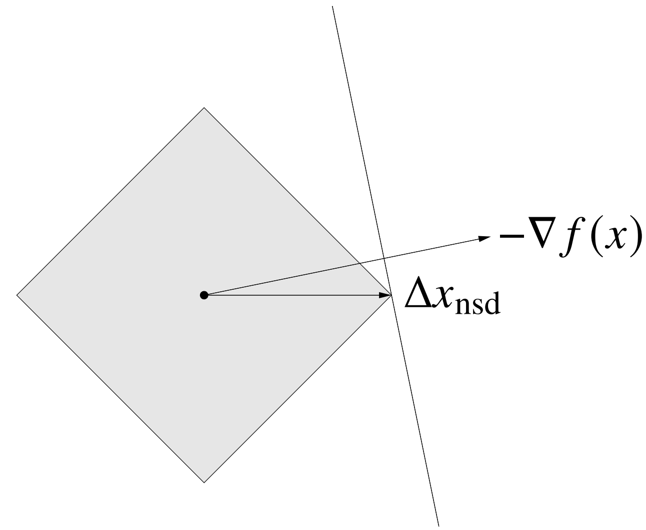


Steepest descent direction for 1-norm

- for $\|\cdot\| = \|\cdot\|_1$ and its dual norm $\|\cdot\|_\infty$, steepest descent directions are

$$\Delta x_{\rm nsd} = \frac{-\partial f(x)/\partial x_i}{|\partial f(x)/\partial x_i|}\, e_i$$

$$\Delta x_{\rm sd} = -\frac{\partial f(x)}{\partial x_i} e_i,$$

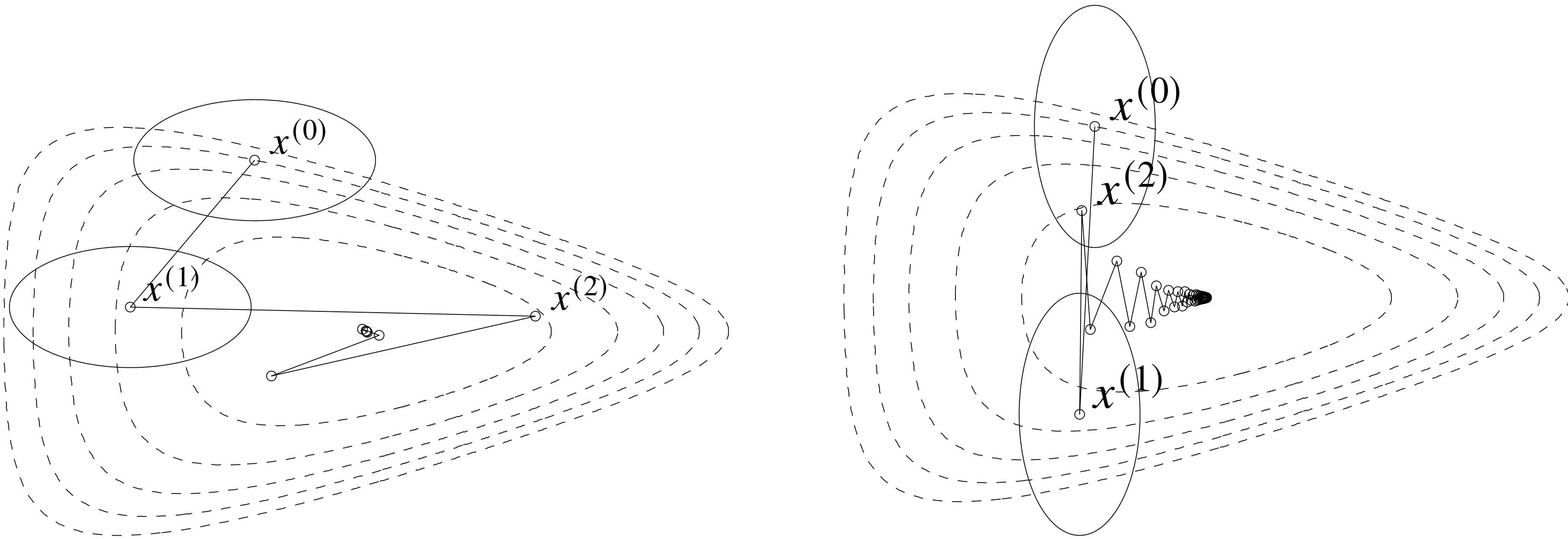


where i is an index with

$$\left|\frac{\partial f(x)}{\partial x_i}\right| = \|\nabla f(x)\|_\infty = \max_{k=1,\ldots,n}\left|\frac{\partial f(x)}{\partial x_k}\right|$$

- not necessarily unique
- steepest descent method for 1-norm updates one coordinate of x at a time

Choice of norm for steepest descent



- steepest descent with backtracking line search for two quadratic norms
- ellipses show $\{x \mid \|x - x^{(k)}\|_P = 1\}$
- equivalent to gradient descent after change of variables $\bar{x} = P^{1/2}x$
- figures show that shows choice of P has strong effect on speed of convergence

Newton step

$$\Delta x_{\rm nt} = -\nabla^2 f(x)^{-1} \nabla f(x)$$

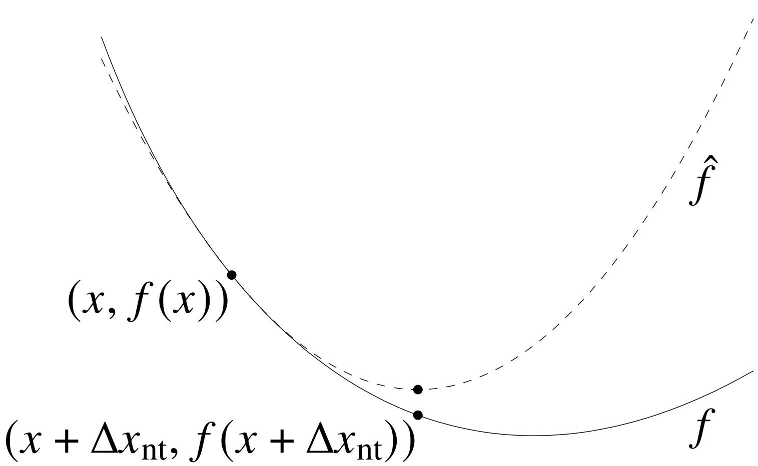
Interpretations

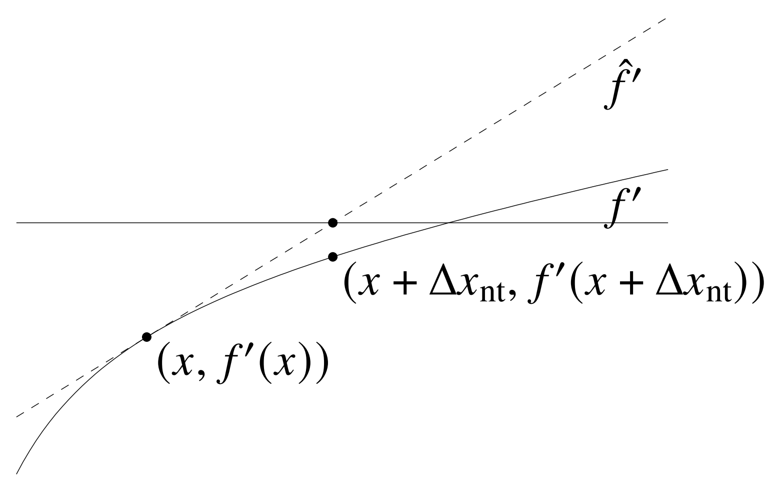
- $x + \Delta x_{\rm nt}$ minimizes second order approximation $\hat{f}$ of f at x

$$\hat{f}(x+v) = f(x) + \nabla f(x)^T v + \frac{1}{2} v^T \nabla^2 f(x) v \qquad (4)$$

- $x + \Delta x_{\rm nt}$ solves linearized optimality condition

$$\nabla f(x+v) \approx \nabla \hat{f}(x+v) = \nabla f(x) + \nabla^2 f(x) v = 0$$

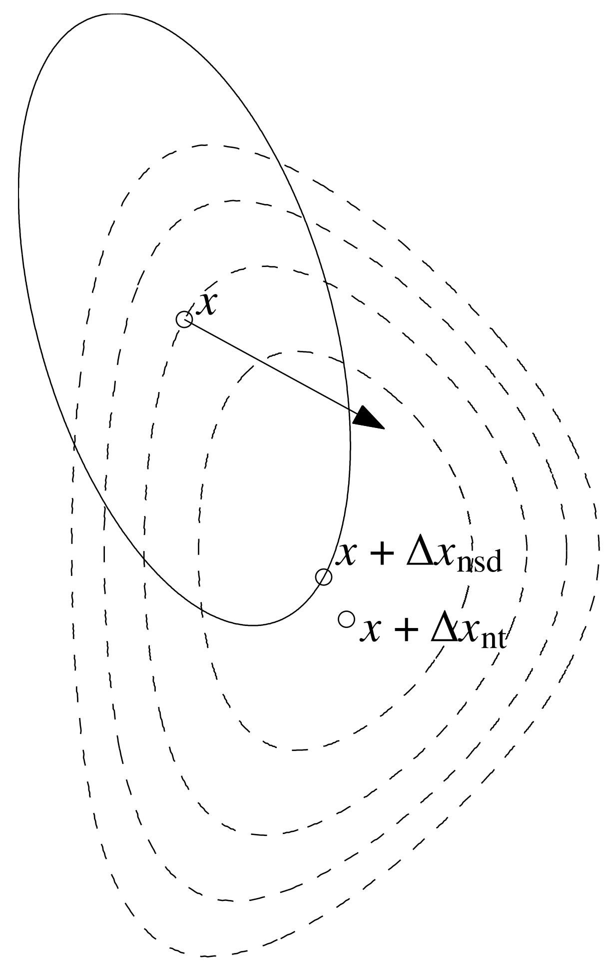




Interpretation as steepest descent direction in local norm

$\Delta x_{\rm nt}$ is steepest descent direction at x in local norm defined by Hessian

$$\|u\|_{\nabla^2 f(x)} = (u^T \nabla^2 f(x) u)^{1/2}$$



- dashed lines are contour lines of f
- ellipse is $\{x + v \mid v^T \nabla^2 f(x) v = 1\}$
- arrow shows $-\nabla f(x)$

Affine invariance of Newton step

suppose we make a change of variables $x = Ay$, with A nonsingular, and solve

$$\text{minimize} \quad g(y) = f(Ay)$$

- gradient and Hessian of g are

$$\nabla g(y) = A^T \nabla f(Ay), \qquad \nabla^2 g(y) = A^T \nabla^2 f(Ay) A$$

- Newton step of g at y is

$$\Delta y_{\mathrm{nt}} = -\nabla^2 g(y)^{-1} \nabla g(y) = -A^{-1} \nabla^2 f(Ay)^{-1} \nabla^2 f(Ay)$$

- if $y = A^{-1}x$, then

$$\Delta y_{\mathrm{nt}} = A^{-1} \Delta x_{\mathrm{nt}}, \qquad \text{where } \Delta x_{\mathrm{nt}} \text{ is Newton step of } f \text{ at } x$$

  Newton step is invariant under affine change of variables

Newton decrement

$$\lambda(x) = (\nabla f(x)^T \nabla^2 f(x)^{-1} \nabla f(x))^{1/2}$$

a measure of the proximity of x to the minimizer $x^\star$

- $\lambda(x)^2 = -\nabla f(x)^T \Delta x_{\mathrm{nt}}$ is negative of directional derivative in Newton direction
- $\lambda(x)$ is the norm of the Newton step in the quadratic Hessian norm

$$\lambda(x) = (\Delta x_{\mathrm{nt}}^T \nabla^2 f(x) \Delta x_{\mathrm{nt}})^{1/2}$$

- $\lambda(x)$ gives estimate of $f(x) - p^\star$, estimated using quadratic approximation (4)

$$f(x) - \inf_y \hat{f}(y) = \frac{1}{2}\lambda(x)^2$$

- $\lambda(x)$ is affine invariant (unlike $\|\nabla f(x)\|_2$)

Newton's method

given: a starting point $x \in \operatorname{dom} f$, tolerance $\epsilon > 0$
repeat

1. *compute Newton step and decrement*
$$\Delta x_{\mathrm{nt}} := -\nabla^2 f(x)^{-1} \nabla f(x); \quad \lambda(x) := (\nabla f(x)^T \nabla^2 f(x)^{-1} \nabla f(x))^{1/2}$$
2. *stopping criterion:* quit if $\lambda(x)^2/2 \leq \epsilon$
3. *line search:* choose step size t by backtracking line search
4. *update:* $x := x + t\Delta x_{\mathrm{nt}}$

- we use line search of page 9.9: starting at $t = 1$, backtrack ($t := \beta t$) until
$$\begin{aligned} f(x + t\Delta x_{\mathrm{nt}}) &< f(x) + \alpha t \nabla f(x)^T \Delta x_{\mathrm{nt}} \\ &= f(x) - \alpha t \lambda(x)^2 \end{aligned}$$

- typical values of line search parameters are $\alpha = 0.01$, $\beta = 1/2$

Affine invariance of Newton's method

- we already noted that Newton step and Newton decrement are affine invariant
- affine invariance of $\lambda(x)$ makes line search, stopping criterion affine invariant
- hence, for Newton method applied to $g(y) = f(Ay)$, started at $y^{(0)} = A^{-1}x^{(0)}$,

$$y^{(k)} = A^{-1}x^{(k)} \quad \text{for all } k$$

  where $x^{(k)}$ are iterates of Newton method applied to $f(x)$, started at $x^{(0)}$
- number of iterates is independent of linear changes of coordinates

Classical convergence analysis

Assumptions

- f strongly convex with constant m
- $\nabla^2 f$ is Lipschitz continuous: there exists a constant L such that

$$\|\nabla^2 f(x) - \nabla^2 f(y)\|_2 \leq L\|x - y\|_2 \quad \text{for all } x, y \in \mathbf{dom}\, f$$

  constant L measures how well f is approximated by a quadratic function

Summary: there exist constants $\eta \in (0, m^2/L)$ and $\gamma > 0$ such that

- if $\|\nabla f(x^{(k)})\|_2 \geq \eta$, then

$$f(x^{(k+1)}) - f(x^{(k)}) \leq -\gamma$$

- if $\|\nabla f(x^{(k)})\|_2 < \eta$, then

$$\frac{L}{2m^2}\|\nabla f(x^{(k+1)})\|_2 \leq \left(\frac{L}{2m^2}\|\nabla f(x^{(k)})\|_2\right)^2$$

Classical convergence analysis

Damped Newton phase $(\|\nabla f(x^{(k)})\|_2 \geq \eta)$

- most iterations require backtracking steps
- function value decreases by at least γ
- if $p^\star > -\infty$, this phase ends after at most

$$\frac{f(x^{(0)}) - p^\star}{\gamma} \quad \text{iterations}$$

Quadratically convergent phase $(\|\nabla f(x^{(k)})\|_2 < \eta)$

- all iterations use step size $t = 1$
- gradient converges to zero quadratically: once $\|\nabla f(x^{(j)})\|_2 < \eta$,

$$\frac{L}{2m^2}\|\nabla f(x^k)\|_2 \leq \left(\frac{L}{2m^2}\|\nabla f(x^j)\|_2\right)^{2^{k-j}} \leq \left(\frac{1}{2}\right)^{2^{k-j}}, \qquad k \geq j$$

- inequality (2) shows that $(f(x^{(k)}) - p^\star) \to 0$ quadratically

Classical convergence analysis

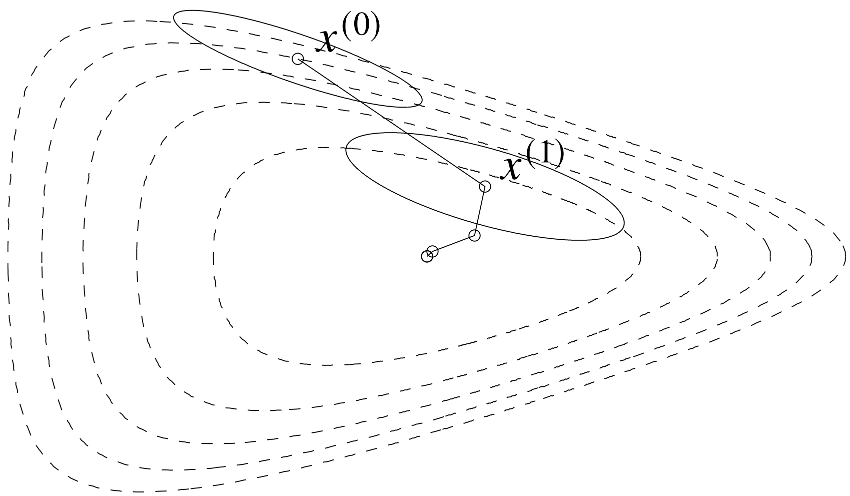
Conclusion: number of iterations until $f(x^{(k)}) - p^\star \leq \epsilon$ is bounded above by

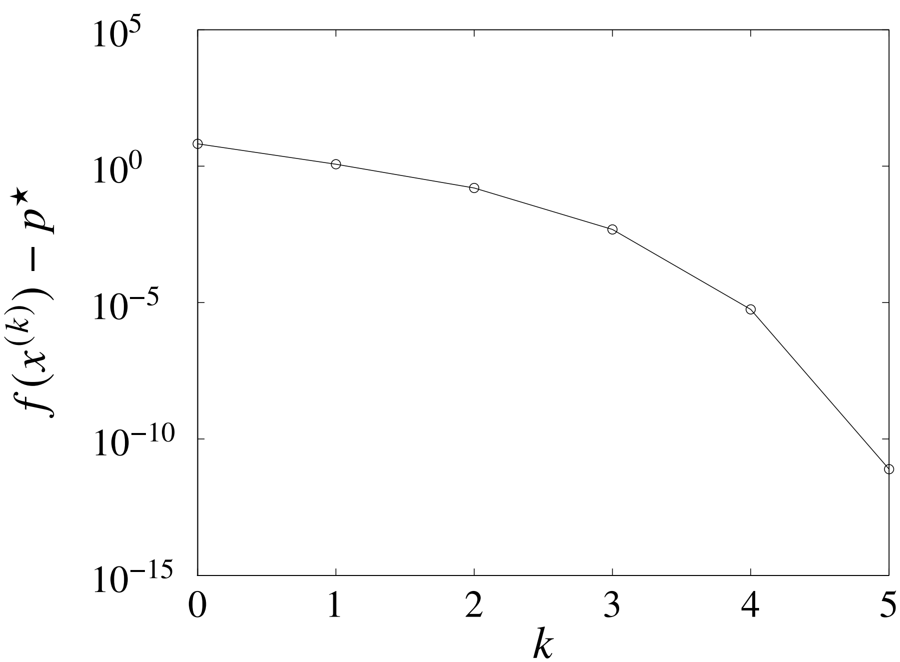
$$\frac{f(x^{(0)}) - p^\star}{\gamma} + \log_2 \log_2(\epsilon_0/\epsilon)$$

- γ, ϵ_0 are constants that depend on m, L, $x^{(0)}$
- 2nd term is small (of the order of 6), almost constant for practical purposes
- in practice, constants m, L (hence γ, ϵ_0) are usually unknown
- provides qualitative insight in convergence properties (two algorithm phases)

Examples

Example in $\mathbf{R}^2$ (page 9.12)

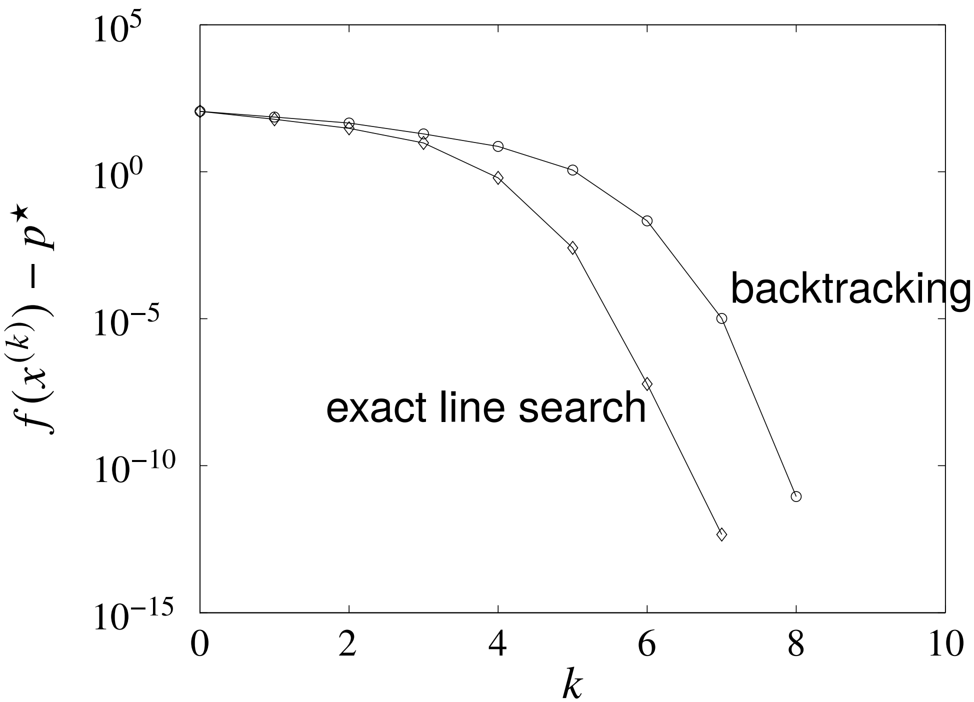


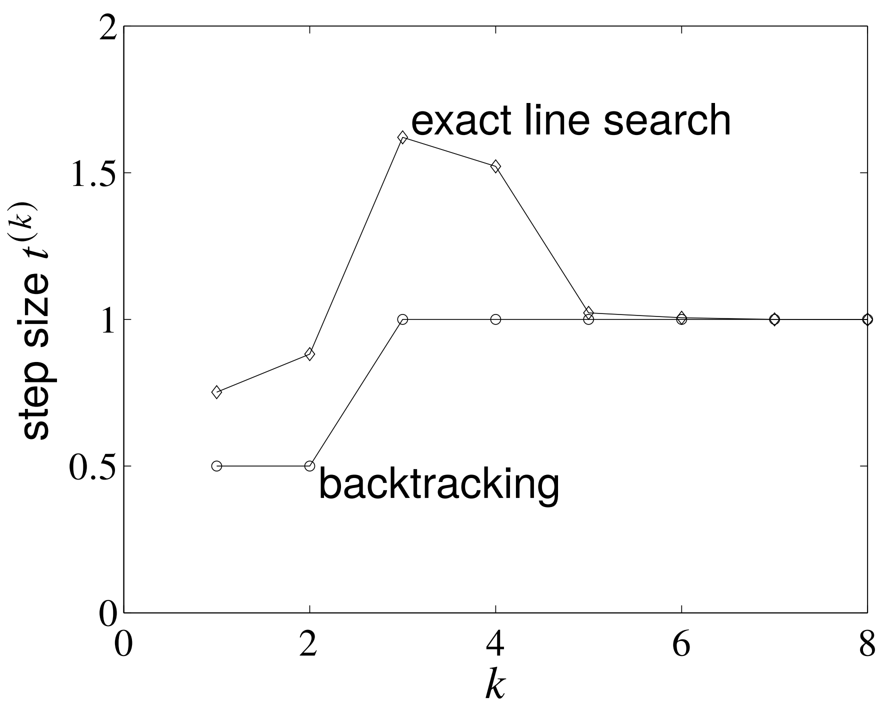


- backtracking parameters $\alpha = 0.1$, $\beta = 0.7$
- converges in only 5 steps
- quadratic local convergence

Examples

Example in $\mathbf{R}^{100}$ (page 9.13)



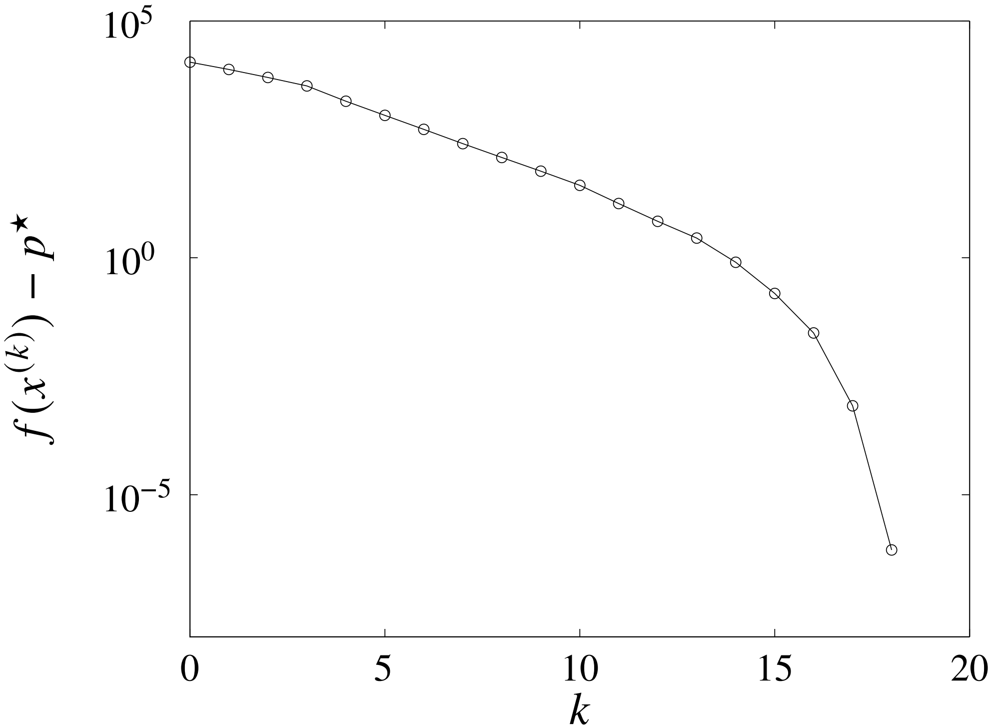


- backtracking parameters $\alpha = 0.01$, $\beta = 0.5$
- backtracking line search almost as fast as exact l.s. (and much simpler)
- clearly shows two phases in algorithm

Examples

Example in $\mathbf{R}^{10000}$ (with sparse a_i)

$$f(x) = -\sum_{i=1}^{10000} \log(1 - x_i^2) - \sum_{i=1}^{100000} \log(b_i - a_i^T x)$$



- backtracking parameters $\alpha = 0.01$, $\beta = 0.5$
- performance similar as for small examples

Self-concordance

Shortcomings of classical convergence analysis

- depends on unknown constants (m, L)
- bound is not affinely invariant, although Newton's method is

Convergence analysis via self-concordance (Nesterov and Nemirovski)

- does not depend on any unknown constants
- gives affine-invariant bound
- applies to special class of convex functions (*self-concordant* functions)
- developed to analyze interior-point methods for convex optimization

Self-concordant functions

Definition

- a convex function $f : \mathbf{R} \to \mathbf{R}$ is self-concordant if

$$|f'''(x)| \leq 2f''(x)^{3/2} \quad \text{for all } x \in \operatorname{dom} f$$

- a convex function $f : \mathbf{R}^n \to \mathbf{R}$ is self-concordant if restriction to a line

$$g(t) = f(x + tv)$$

  is a self-concordant function of t for all $x \in \operatorname{dom} f$ and v

Affine invariance

- if $f : \mathbf{R}^n \to \mathbf{R}$ is self-concordant, then $\tilde{f}(y) = f(Ay)$ is self-concordant
- this is easily checked for $f : \mathbf{R} \to \mathbf{R}$ and $\tilde{f}(y) = f(ay)$:

$$|\tilde{f}'''(y)| = |a|^3 |f'''(ay)| \leq 2|a|^3 f''(ay)^{3/2} = 2\tilde{f}''(y)^{3/2}$$

Examples of self-concordant functions

- linear and quadratic functions
- negative logarithm: $f(x) = -\log x$
- negative entropy plus negative logarithm: $f(x) = x \log x - \log x$
- logarithmic barrier for set of linear inequalities

$$f(x) = -\sum_{i=1}^{m} \log(b_i - a_i^T x), \qquad \mathbf{dom}\, f = \{x \mid a_i^T x < b_i,\ i = 1, \ldots, m\}$$

- log-det barrier

$$f(X) = -\log \det X, \qquad \mathbf{dom}\, f = \mathbf{S}_{++}^n$$

- logarithmic barrier for second order cone

$$f(x) = -\log(y^2 - x^T x), \qquad \mathbf{dom}\, f = \{(x, y) \mid \|x\|_2 < y\}$$

Newton's method for self-concordant functions

Newton's method of page 9.22

Summary: there exist constants $\eta \in (0, 1/4]$, $\gamma > 0$ such that

- if $\lambda(x^{(k)}) > \eta$, then
$$f(x^{(k+1)}) - f(x^{(k)}) \leq -\gamma$$
- if $\lambda(x^{(k)}) \leq \eta$, then
$$2\lambda(x^{(k+1)}) \leq (2\lambda(x^{(k)}))^2$$

η and γ only depend on backtracking parameters α, β

Complexity bound: number of iterations until $f(x^{(k)}) - p^\star \leq \epsilon$ is bounded by

$$\frac{f(x^{(0)}) - p^\star}{\gamma} + \log_2 \log_2(1/\epsilon)$$

Implementation of Newton's method

main effort in each iteration: evaluate derivatives and solve Newton system

$$H\Delta x = -g$$

where $H = \nabla^2 f(x)$ and $g = \nabla f(x)$

Via Cholesky factorization

$$H = LL^T, \qquad \Delta x_{\mathrm{nt}} = -L^{-T}L^{-1}g, \qquad \lambda(x) = \|L^{-1}g\|_2$$

- cost: $(1/3)n^3$ flops for unstructured system, plus cost of evaluating derivatives
- cost $\ll (1/3)n^3$ if H sparse or highly structured (for example, banded)

Structured-plus-low-rank matrices

a type of structured linear equations, common in optimization:

$$(A + BC)x = b \qquad \text{with } A \in \mathbf{R}^{n\times n},\ B \in \mathbf{R}^{n\times p},\ C \in \mathbf{R}^{p\times n} \tag{5}$$

- A has some property that makes $Ax = b$ easy to solve, for example, diagonal
- B, C are dense, with $p \ll n$
- using an auxilary variable y, equation can be written as

$$\left[\begin{array}{cc} A & B \\ C & -I \end{array}\right] \left[\begin{array}{c} x \\ y \end{array}\right] = \left[\begin{array}{c} b \\ 0 \end{array}\right] \tag{6}$$

- instead of solving (5) directly, can solve (6) by eliminating x: first solve equation

$$(I + CA^{-1}B)y = CA^{-1}b$$

to find y; then solve $Ax = b - By$ to find x

Matrix inversion lemma

if A and $A+BC$ are nonsingular, then $I+CA^{-1}B$ is nonsingular and

$$(A+BC)^{-1} = A^{-1} - A^{-1}B(I+CA^{-1}B)^{-1}CA^{-1} \tag{7}$$

- easily verified by multiplying $A+BC$ and right-hand side of (7)
- can be derived via method on previous page: $x=(A+BC)^{-1}b$ is equal to

$$\begin{aligned} x &= A^{-1}(b-By) \\ &= A^{-1}(b-B(I+CA^{-1}B)^{-1}CA^{-1}b) \\ &= (A^{-1}-A^{-1}B(I+CA^{-1}B)^{-1}CA^{-1})b \qquad (8) \end{aligned}$$

  since this is true for all b, matrix on the right-hand side of (8) is $(A+BC)^{-1}$
- method on previous page can be viewed as evaluating $(A+BC)^{-1}b$ via (8)

Example

Newton method for unconstrained optimization with cost function $f : \mathbf{R}^n \to \mathbf{R}$,

$$f(x) = \sum_{i=1}^{n} \psi_i(x_i) + \phi(Ax + b)$$

- functions $\psi_i : \mathbf{R} \to \mathbf{R}$ and $\phi : \mathbf{R}^p \to \mathbf{R}$ are convex
- assume $A \in \mathbf{R}^{p \times n}$, dense, with $p \ll n$
- Hessian of f is diagonal plus low rank:

$$H = D + A^T G A$$

where D is diagonal with $D_{ii} = \psi_i''(x_i)$, and $G = \nabla^2 \phi(Ax + b)$

Example

compare two methods for solving Newton equation $(D + A^T GA)\Delta x = -g$

Method 1: form $D + A^T GA$, solve via dense Cholesky

cost dominated by cost of factorization ($(1/3)n^3$ flops)

Method 2: follow idea on page 9.35

- compute Cholesky factorization $G = LL^T$ and write Newton system as

$$\begin{bmatrix} D & A^T L \\ L^T A & -I \end{bmatrix} \begin{bmatrix} \Delta x \\ y \end{bmatrix} = \begin{bmatrix} -g \\ 0 \end{bmatrix}$$

- eliminate Δx from first equation: solve two equations

$$(I + L^T AD^{-1} A^T L)y = -L^T AD^{-1} g, \qquad D\Delta x = -g - A^T Ly$$

- cost is roughly $2p^2 n$ flops, dominated by computation of $L^T AD^{-1} A^T L$

complexity of method 2 is linear in n