

Lecture 4: Unconstrained optimization algorithms

Method of choice

Consider the unconstrained optimization problem to

$$\underset{\mathbf{x} \in \mathbb{R}^n}{\text{minimize}} \ f(\mathbf{x}), \quad (1)$$

where $f \in C^0$ on \mathbb{R}^n (f is continuous). Mostly, we assume that $f \in C^1$ holds (f is continuously differentiable), sometimes even C^2

- Size of the problem (n)?
- Are $\nabla f(\mathbf{x})$ and/or $\nabla^2 f(\mathbf{x})$ available; to what cost?
- What is the goal? (Global/local minimum, stationary point?)
- What are the convexity properties of f ?
- Do we have a good estimate of the location of a stationary point \mathbf{x}^* ? (Can we use locally-only convergent methods?)

Example: curve fitting by least-squares

- Suppose we have m data points (t_i, b_i) believed to be related as

$$x_1 + x_2 \exp(x_3 t_i) + x_4 \exp(x_5 t_i) = b_i, \quad i = 1, \dots, m,$$

with unknown parameters x_1, \dots, x_5 . (Here, $\exp(x) = e^x$.) The best description minimizes the total “residual error,” given by the norm of the residual

$$f_i(\mathbf{x}) := b_i - [x_1 + x_2 \exp(x_3 t_i) + x_4 \exp(x_5 t_i)], \quad i = 1, \dots, m$$

- Resulting optimization problem:

$$\min_{\mathbf{x} \in \mathbb{R}^5} f(\mathbf{x}) := \sum_{i=1}^m |f_i(\mathbf{x})|^2 = \sum_{i=1}^m (b_i - [x_1 + x_2 \exp(x_3 t_i) + x_4 \exp(x_5 t_i)])^2$$

- Very often solved problem type within numerical analysis and mathematical statistics

Typical algorithm

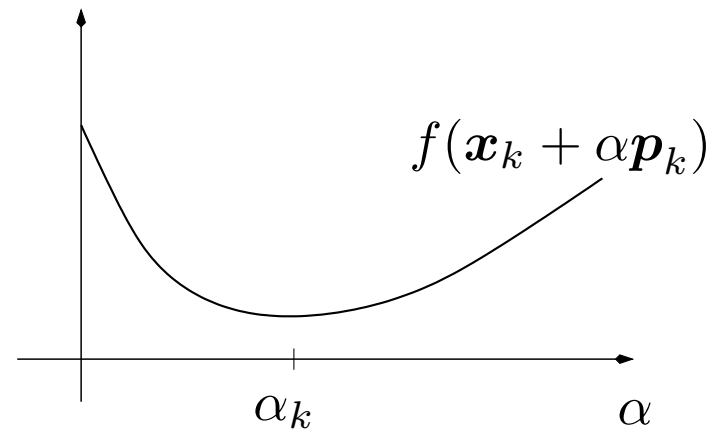
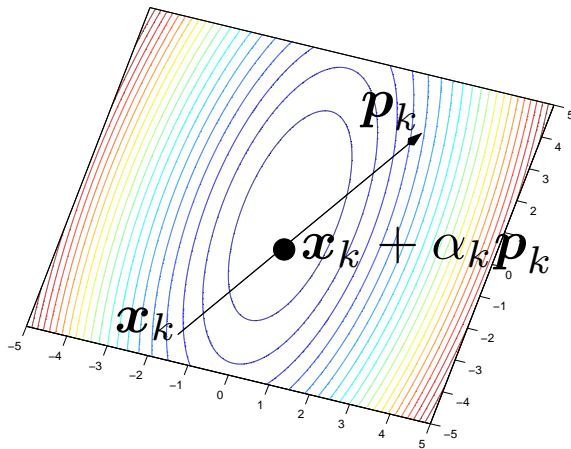
Step 0. Starting point: $\mathbf{x}_0 \in \mathbb{R}^n$. Set $k := 0$

Step 1. Search direction: $\mathbf{p}_k \in \mathbb{R}^n$

Step 2. Step length: $\alpha_k > 0$ such that $f(\mathbf{x}_k + \alpha_k \mathbf{p}_k) < f(\mathbf{x}_k)$ holds

Step 3. Let $\mathbf{x}_{k+1} := \mathbf{x}_k + \alpha_k \mathbf{p}_k$

Step 4. Termination criterion: If fulfilled, then stop! Otherwise, let $k := k + 1$ and go to step 1



Notes

- The figure was plotted using several thousands of function evaluations
- Never possible in reality! (And total waste of time)
- An “orienteering map” never exists
- Most algorithms are inherently *local*, only based on info at the current point \mathbf{x}_k , that is, $f(\mathbf{x}_k)$, $\nabla f(\mathbf{x}_k)$, and $\nabla^2 f(\mathbf{x}_k)$
- Possibly also on previous points passed
- An algorithm is a “near-sighted mountain climber” when trying to reach the summit (for a max problem!)
- The mountain climber is in a deep fog and can only check her barometer for the height and feel the steepness of the slope under her feet

Step 1: Search directions

- If $\nabla f(\mathbf{x}_k) \neq \mathbf{0}^n$, then $\mathbf{p} = -\nabla f(\mathbf{x}_k)$ is a descent direction for f at \mathbf{x}_k (Part of necessary condition proof!)
- This *steepest descent direction* solves the problem to

$$\underset{\mathbf{p} \in \mathbb{R}^n: \|\mathbf{p}\|=1}{\text{minimize}} \quad \nabla f(\mathbf{x}_k)^\top \mathbf{p}$$

- Suppose $\mathbf{Q} \in \mathbb{R}^{n \times n}$ is a symmetric, positive definite matrix. Then $\mathbf{p} = -\mathbf{Q}\nabla f(\mathbf{x}_k)$ is a descent direction for f at \mathbf{x}_k , because

$$\nabla f(\mathbf{x}_k)^\top \mathbf{p} = -\nabla f(\mathbf{x}_k)^\top \mathbf{Q} \nabla f(\mathbf{x}_k) < 0,$$

due to the positive definiteness of \mathbf{Q}

- Special case: $\mathbf{Q} = \mathbf{I}^n$ yield steepest descent
- Special case: $\mathbf{Q}^{-1} = \nabla^2 f(\mathbf{x}_k)$, if the Hessian is positive definite. This is *Newton's method*

Additional requirements

$$|\nabla f(\mathbf{x}_k)^T \mathbf{p}_k| \geq s_1 \|\nabla f(\mathbf{x}_k)\|^2, \quad \text{and} \quad \|\mathbf{p}_k\| \leq s_2 \|\nabla f(\mathbf{x}_k)\|,$$

or

$$\frac{\nabla f(\mathbf{x}_k)^T \mathbf{p}_k}{\|\nabla f(\mathbf{x}_k)\| \cdot \|\mathbf{p}_k\|} \geq s_1, \quad \text{and} \quad \|\mathbf{p}_k\| \geq s_2 \|\nabla f(\mathbf{x}_k)\|$$

- Purpose: prevent the descent directions to deteriorate in quality, and prevent premature convergence
- $\nabla f(\mathbf{x}_k)^T \mathbf{p}_k$ is the directional derivative of f at \mathbf{x}_k in the direction of \mathbf{p}_k . Make sure it stays away from zero!
- Also, make sure that \mathbf{p}_k stays bounded and that it tends to zero if and only if $\nabla f(\mathbf{x}_k)$ does
- These conditions hold for the above examples

Newton's method

- Steepest descent is most often not a very good algorithm. Why?
- It fails to take into account more than information about ∇f
- Let

$$f(\mathbf{x} + \mathbf{p}) - f(\mathbf{x}) \approx \varphi_{\mathbf{x}}(\mathbf{p}) = \nabla f(\mathbf{x})^{\top} \mathbf{p} + \frac{1}{2} \mathbf{p}^{\top} \nabla^2 f(\mathbf{x}) \mathbf{p}$$

Minimize by setting gradient of $\varphi_{\mathbf{x}}(\mathbf{p})$ to zero:

$$\nabla_{\mathbf{p}} \varphi_{\mathbf{x}}(\mathbf{p}) = \nabla f(\mathbf{x}) + \nabla^2 f(\mathbf{x}) \mathbf{p} = \mathbf{0}^n$$

- $n = 1$: $f'(x) + f''(x)p = 0 \implies p = -f'(x)/f''(x)$
- Provides descent if $f''(x) > 0$: $f'(x)p = -[f'(x)]^2/f''(x) < 0$
- Corresponding story in \mathbb{R}^n : $\mathbf{p} = -[\nabla^2 f(\mathbf{x})]^{-1} \nabla f(\mathbf{x})$, yields descent at non-stationary points if $\nabla^2 f(\mathbf{x})$ is positive definite!

Why do we not always choose Newton directions?

- **Lack of positive definiteness.** $\nabla^2 f(\mathbf{x})$ is not positive definite (PD). Solution: add diagonal matrix so that the result is PD:
 $\nabla^2 f(\mathbf{x}) + \gamma \mathbf{I}^n$ for $\gamma > 0$ large enough
- Note: If value of γ is very large $\implies \approx$ steepest descent
- Name: *Levenberg–Marquardt*
- **Lack of enough differentiability.** If $f \notin C^2$, what do we do?
- $n = 1$: the *secant method*:

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

- $n > 1$: *quasi-Newton*: choose approximate matrix \mathbf{B}_k so that

$$\mathbf{B}_k(\mathbf{x}_k - \mathbf{x}_{k-1}) = \nabla f(\mathbf{x}_k) - \nabla f(\mathbf{x}_{k-1}),$$

and more choices (the above does not specify the entire matrix!)

- **Computational burden.** It may be too much to ask for to solve a linear system many times when $n > 1000$ or so; it is enough to do *some* work on the linear system and still get a descent property. (See book for an example)
- Specific choices of matrices \mathbf{B}_k lead to *quasi-Newton* methods

Step 2: Line search

- Approximately solve the one-dimensional problem to

$$\underset{\alpha \geq 0}{\text{minimize}} \varphi(\alpha) := f(\mathbf{x}_k + \alpha \mathbf{p}_k)$$

Its optimality conditions are that

$$\varphi'(\alpha^*) \geq 0, \quad \alpha^* \cdot \varphi'(\alpha^*) = 0, \quad \alpha^* \geq 0,$$

that is,

$$\nabla f(\mathbf{x}_k + \alpha^* \mathbf{p}_k)^\top \mathbf{p}_k \geq 0, \quad \alpha^* \cdot \nabla f(\mathbf{x}_k + \alpha^* \mathbf{p}_k)^\top \mathbf{p}_k = 0, \quad \alpha^* \geq 0,$$

holds

- If $\alpha^* > 0$, then $\varphi'(\alpha^*) = 0$ holds, hence $\nabla f(\mathbf{x}_k + \alpha^* \mathbf{p}_k)^\top \mathbf{p}_k = 0$
- The search direction \mathbf{p}_k is orthogonal to the gradient of f at the point $\mathbf{x}_k + \alpha^* \mathbf{p}_k$

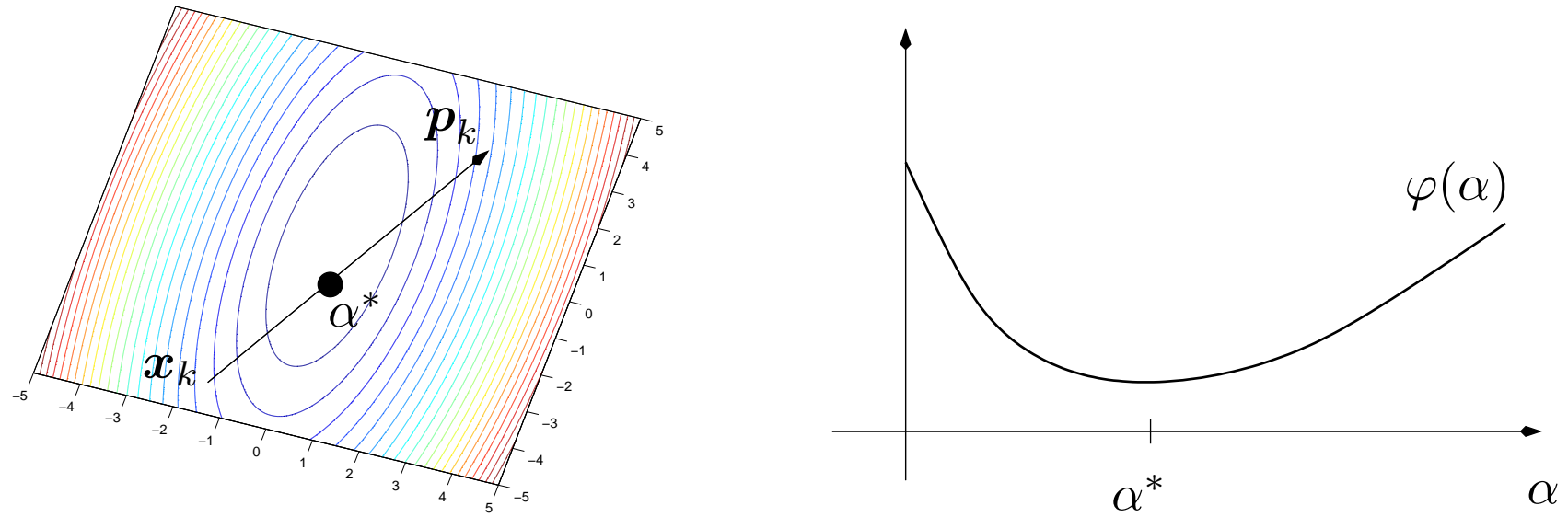


Figure 1: A line search in a descent direction

Approximate line search

- No point solving the one-dimensional problem exactly! Why? The optimum to the entire problem lies elsewhere!
- **Interpolation:** Use $f(\mathbf{x}_k), \nabla f(\mathbf{x}_k), \nabla f(\mathbf{x}_k)^T \mathbf{p}_k$ to model a quadratic function approximating f along \mathbf{p}_k . Minimize it by using the analytic formula for quadratics
- **Newton's method:** Repeat the improvements gained from a quadratic approximation: $\alpha := \alpha - \varphi'(\alpha)/\varphi''(\alpha)$
- **Golden section:** Derivative-free method that shrinks an interval where $\varphi'(\alpha) = 0$ lies

Armijo rule

- Idea: quickly generate a step α which provides “sufficient” decrease in f . Note: $f(\mathbf{x}_k + \alpha \mathbf{p}_k) \approx f(\mathbf{x}_k) + \alpha \cdot \nabla f(\mathbf{x}_k)^\top \mathbf{p}_k$, valid for small values of $\alpha > 0$
- Requirement: we get a decrease in f which is at least a *fraction* of that predicted in the right-hand side above. Let $\mu \in (0, 1)$ be this fraction. Acceptable step lengths are $\alpha > 0$ satisfying

$$\varphi(\alpha) - \varphi(0) \leq \mu \alpha \varphi'(0), \quad (2a)$$

that is,

$$f(\mathbf{x}_k + \alpha \mathbf{p}_k) - f(\mathbf{x}_k) \leq \mu \alpha \nabla f(\mathbf{x}_k)^\top \mathbf{p}_k \quad (2b)$$

- Can add condition making α also large enough (*Wolfe*)

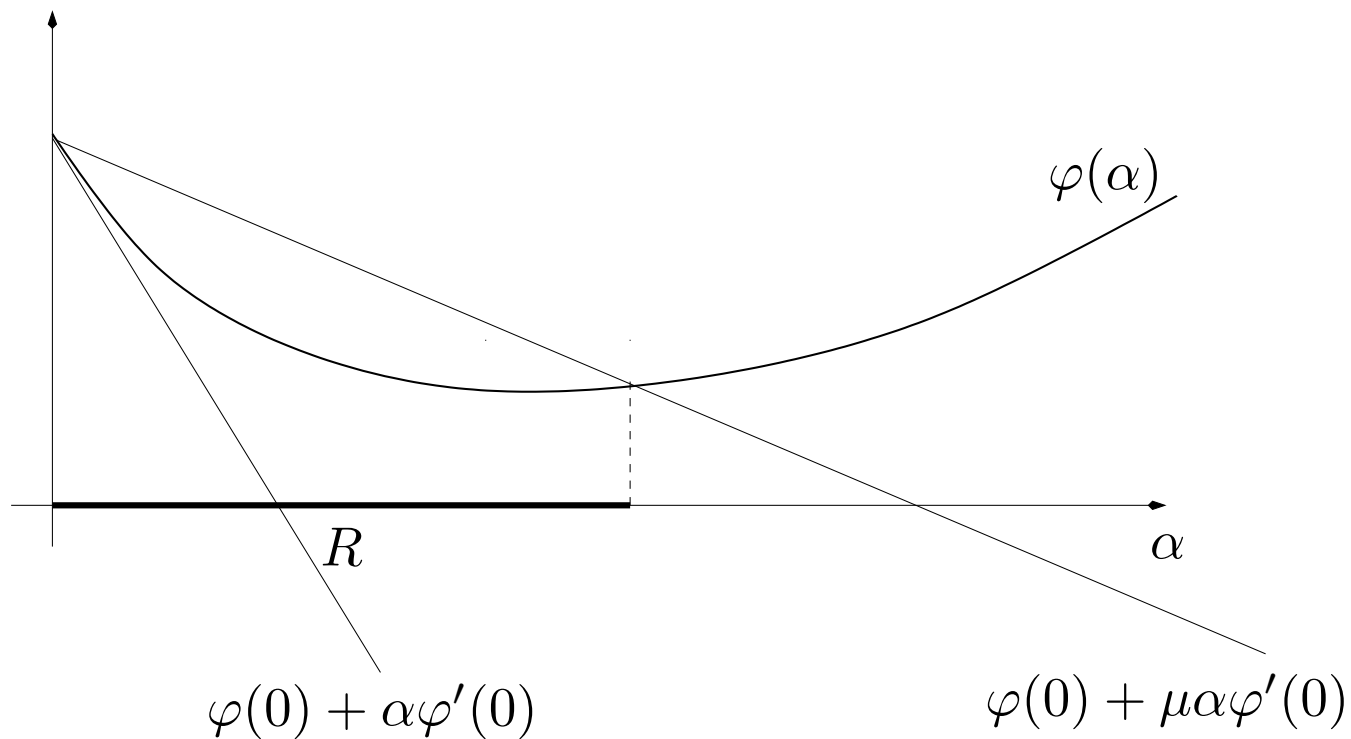


Figure 2: The interval (R) accepted by the Armijo step length rule

Typical convergence result

- Suppose that $f \in C^1$, and that for the starting point \mathbf{x}_0 it holds that the level set $\text{lev}_f(f(\mathbf{x}_0)) = \{\mathbf{x} \in \mathbb{R}^n \mid f(\mathbf{x}) \leq f(\mathbf{x}_0)\}$ is bounded. Consider the iterative algorithm defined on Page 1, with the following choices for each k :
 - \mathbf{p}_k satisfies the second sufficient descent condition on Page 6;
 - $\|\mathbf{p}_k\| \leq M$, where M is some positive constant; and
 - the Armijo step length rule is used

Then, the sequence $\{\mathbf{x}_k\}$ is bounded, the sequence $\{f(\mathbf{x}_k)\}$ is descending, lower bounded and therefore converges, and every limit point of $\{\mathbf{x}_k\}$ is stationary □

- For convex f much stronger convergence properties:

Optimum exists $\iff \{\mathbf{x}_k\}$ converges to an optimal solution

(Theorem proved for gradient projection method later)

Step 4: Termination criteria

- Lesson number one: Cannot terminate based on the exact optimality conditions, because $\nabla f(\mathbf{x}) = \mathbf{0}^n$ rarely happens!
- The recommendation is the combination of the following:
 1. $\|\nabla f(\mathbf{x}_k)\| \leq \varepsilon_1(1 + |f(\mathbf{x}_k)|)$, $\varepsilon_1 > 0$ small;
 2. $f(\mathbf{x}_{k-1}) - f(\mathbf{x}_k) \leq \varepsilon_2(1 + |f(\mathbf{x}_k)|)$, $\varepsilon_2 > 0$ small; and
 3. $\|\mathbf{x}_{k-1} - \mathbf{x}_k\| \leq \varepsilon_3(1 + \|\mathbf{x}_k\|)$, $\varepsilon_3 > 0$ small
- Why? Need to cover cases of very steep and very flat functions
- May need to use ∞ -norm: $\|\mathbf{x}\|_\infty := \max_{1 \leq j \leq n} |x_j|$, for large n

- Problem with the scaling of the problem: If

$$\mathbf{x}_{k-1} = (1.44453, 0.00093, 0.0000079)^T,$$

$$\mathbf{x}_k = (1.44441, 0.00012, 0.0000011)^T;$$

$$\begin{aligned}\|\mathbf{x}_{k-1} - \mathbf{x}_k\|_\infty &= \|(0.00012, 0.00081, 0.0000068)^T\|_\infty \\ &= 0.00081\end{aligned}$$

- Small absolute error but large relative error!
- Better to apply the algorithm from a scaled problem where elements of \mathbf{x} have similar magnitude
- Newton methods define good such scalings

Why is the C^1 property important?

- Suppose f is only in C^0 , not C^1 . Example:

$$f(\mathbf{x}) := \underset{i \in \{1, \dots, m\}}{\text{maximum}} \{ \mathbf{c}_i^T \mathbf{x} + b_i \}, \quad \mathbf{x} \in \mathbb{R}^n$$

- This is a piece-wise linear and convex function (see next page)
- It is differentiable almost everywhere, but *not* at the optimal solution!
- Ignoring non-differentiability may lead to the convergence to a non-optimal point
- Convex functions always has *subgradients*, corresponding to all the possible slopes of the function
- More on these when looking at Lagrangian duality!

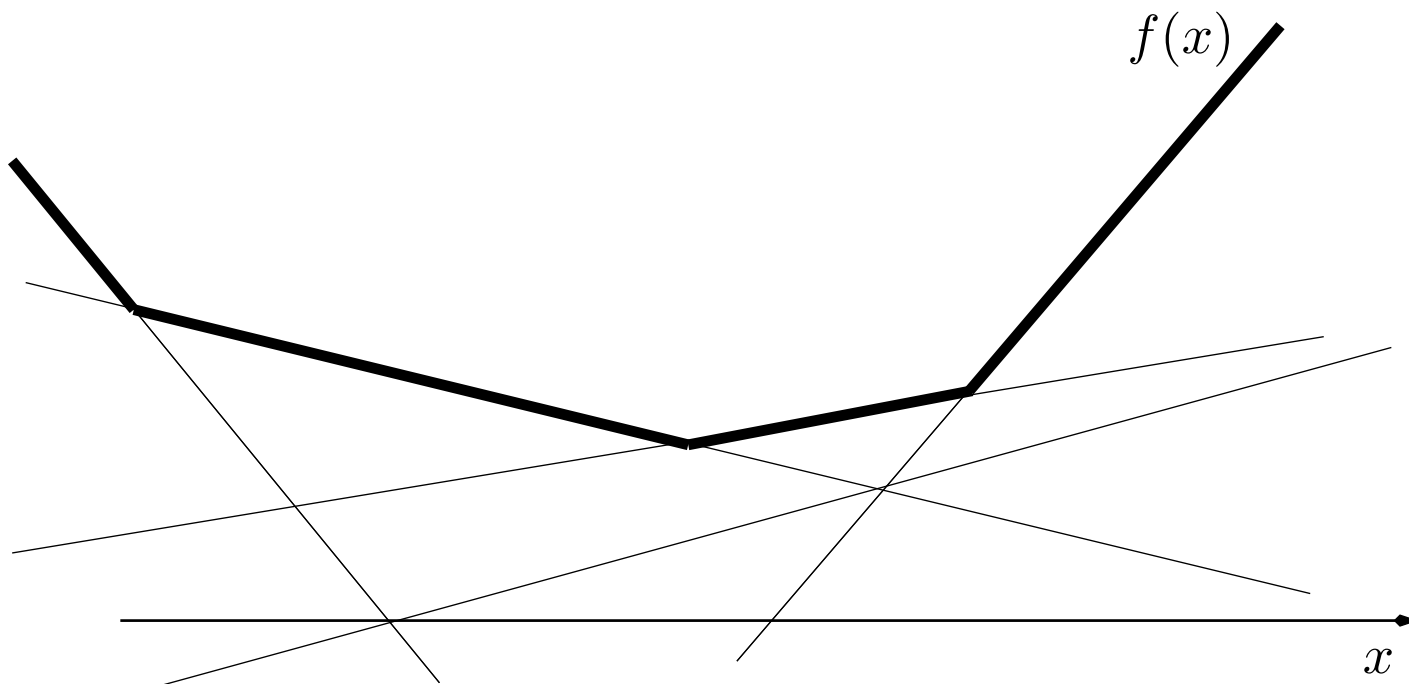


Figure 3: A piece-wise linear convex function

Trust region methods

- Trust region methods use quadratic models (as Newton)
- Avoids line searches by bounding the length of the search direction, at the same time influencing its direction
- Let $\psi_k(\mathbf{p}) := f(\mathbf{x}_k) + \nabla f(\mathbf{x}_k)^\top \mathbf{p} + \frac{1}{2} \mathbf{p}^\top \nabla^2 f(\mathbf{x}_k) \mathbf{p}$
- The model ψ_k is *trusted* in a neighbourhood of $\mathbf{x}_k : \|\mathbf{p}\| \leq \Delta_k$
- Very useful when $\nabla^2 f(\mathbf{x}_k)$ is not positive semi-definite
- Easy to minimize $\psi_k(\mathbf{p})$ subject to $\|\mathbf{p}\| \leq \Delta_k$
- Idea: when $\nabla^2 f(\mathbf{x}_k)$ is badly conditioned, Δ_k should be small (more of a steepest descent method); if well conditioned, Δ_k should be large to allow for unit steps (Newton! fast convergence)
- If Δ_k is small enough, $f(\mathbf{x}_k + \mathbf{p}_k) < f(\mathbf{x}_k)$ holds
- Even if $\nabla f(\mathbf{x}_k) = \mathbf{0}^n$ holds, $f(\mathbf{x}_k + \mathbf{p}_k) < f(\mathbf{x}_k)$ still holds, if

$\nabla^2 f(\mathbf{x}_k)$ is not positive definite

- Progress from stationary points if saddle points or local maxima
- Update of trust region size based on a measure of similarity between the model ψ_k and f : Let

$$\rho_k = \frac{f(\mathbf{x}_k) - f(\mathbf{x}_k + \mathbf{p}_k)}{f(\mathbf{x}_k) - \psi_k(\mathbf{p}_k)} = \frac{\text{actual reduction}}{\text{predicted reduction}}$$

If $\rho_k \leq \mu$ let $\mathbf{x}_{k+1} = \mathbf{x}_k$ (unsuccessful step), else

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{p}_k \text{ (successful step)}$$

Value of Δ_{k+1} depends on ρ_k :

$$\rho_k \leq \mu \implies \Delta_{k+1} = \frac{1}{2}\Delta_k,$$

$$\mu < \rho_k < \eta \implies \Delta_{k+1} = \Delta_k,$$

$$\rho_k \geq \eta \implies \Delta_{k+1} = 2\Delta_k$$

Figure 4 illustrates the trust region subproblem

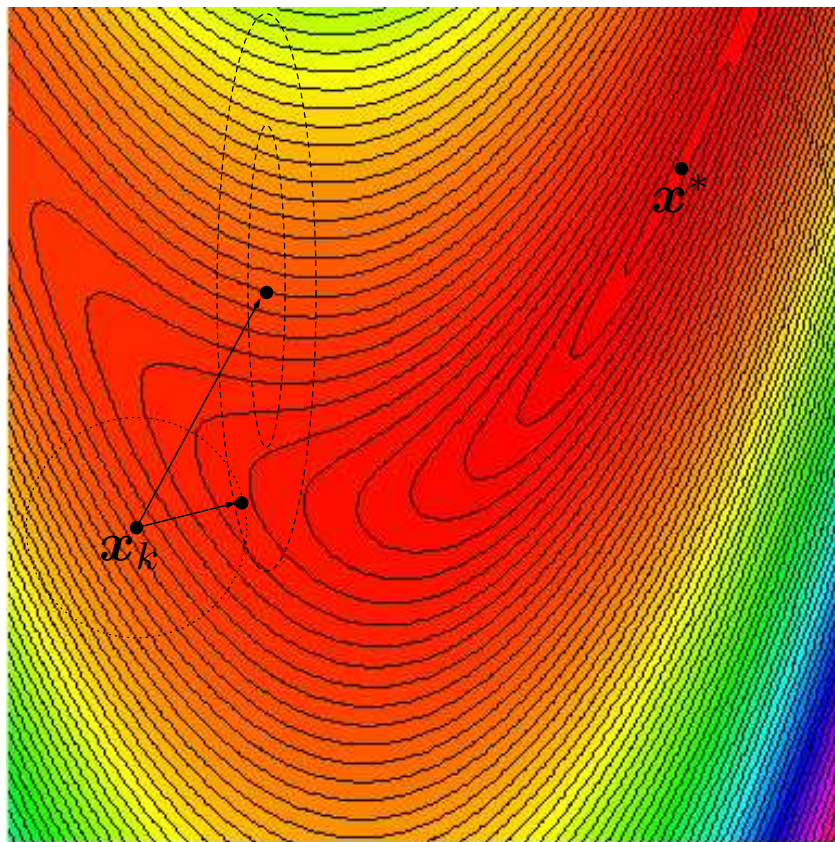
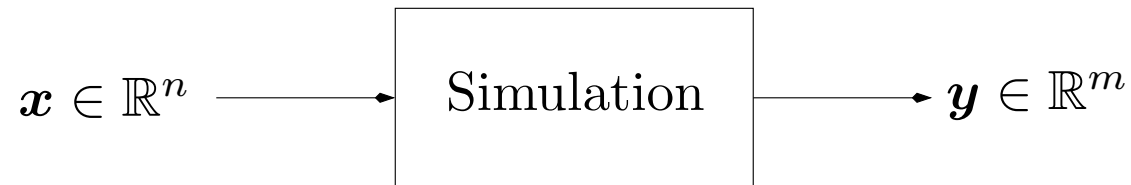


Figure 4: Trust region and line search step. The ellipses are level curves of the quadratic model; the circle defines the trust region

Minimizing implicit functions

- Common in engineering and natural science applications that f is not explicitly given but through a simulation:



- Wish is to minimize a function of both \mathbf{x} and \mathbf{y} : $f(\mathbf{x}, \mathbf{y})$; find the vector \mathbf{x} that gives the best *response* \mathbf{y} for f
- The form of the response $\mathbf{y} = \mathbf{y}(\mathbf{x})$ from the input \mathbf{x} is normally unknown
- Cannot differentiate $\mathbf{x} \mapsto f(\mathbf{x}, \mathbf{y}(\mathbf{x}))$
- Two distinct possibilities!

- (1) *Numerical differentiation* of f by using a difference formula:
- Let $\mathbf{e}_i = (0, 0, \dots, 0, 1, 0, \dots, 0)^\top$ be the unit vector in \mathbb{R}^n . Then,

$$\begin{aligned} f(\mathbf{x} + \alpha \mathbf{e}_i) &= f(\mathbf{x}) + \alpha \mathbf{e}_i^\top \nabla f(\mathbf{x}) + (\alpha^2/2) \mathbf{e}_i^\top \nabla^2 f(\mathbf{x}) \mathbf{e}_i + \dots \\ &= f(\mathbf{x}) + \alpha \partial f(\mathbf{x}) / \partial x_i + (\alpha^2/2) \partial^2 f(\mathbf{x}) / \partial x_i^2 + \dots \end{aligned}$$

- So, for small $\alpha > 0$,

$$\frac{\partial f(\mathbf{x})}{\partial x_i} \approx \frac{f(\mathbf{x} + \alpha \mathbf{e}_i) - f(\mathbf{x})}{\alpha} \quad (\text{forward difference})$$

$$\frac{\partial f(\mathbf{x})}{\partial x_i} \approx \frac{f(\mathbf{x} + \alpha \mathbf{e}_i) - f(\mathbf{x} - \alpha \mathbf{e}_i)}{2\alpha} \quad (\text{central difference})$$

- Value of α typically set to a function of the machine precision; if too large, we get a bad approximation of the partial derivative, while a too small value might result in numerical cancellation
- May work well *if* the simulation is *accurate*, otherwise bad derivative information. Requires *cheap* simulations!

- (2) *Derivative-free methods* are available. (Not counting subgradient methods, because they demand f to be convex!)
Either builds explicit *models* \hat{f} of the objective function by evaluating f at test points, or evaluates f at grid points that are moved around, shrunk or expanded. *Names: Nelder–Mead, Pattern search*
- Check hand-out!

Methods based on interpolating/approximating f

- Alternative: create explicit algebraic (e.g., polynomial) model \tilde{f} based on visited points \mathbf{x}_k ; solve this problem with gradient methods; evaluate its optimum in the real problem (i.e., perform a simulation); update \tilde{f} with the new information \implies minimizes the number of simulations!
- Recent application: diesel engine optimization for Volvo Powertrain and Volvo Car Corporation
- Optimize fuel consumption, keep soot/nitrogen emissions at an acceptable level
- Simulations hard (42 hours each) and response contains noise
- New method developed based on approximate (surrogate) models

Conjugate gradient methods

- Algorithm for strictly convex quadratic programs to minimize $f(\mathbf{x}) := \frac{1}{2}\mathbf{x}^T \mathbf{Q}\mathbf{x} - \mathbf{q}^T \mathbf{x}$, that is, solve $\mathbf{Q}\mathbf{x} = \mathbf{q}$
- Non-quadratic extensions available
- Basic scheme:

$$\mathbf{p}_0 = -\nabla f(\mathbf{x}_0); \quad (3a)$$

$$\mathbf{p}_k = -\nabla f(\mathbf{x}_k) + \beta_k \mathbf{p}_{k-1}, \quad k = 1, 2, \dots, n-1, \quad (3b)$$

where

$$\beta_k = \frac{\nabla f(\mathbf{x}_k)^T \nabla f(\mathbf{x}_k)}{\nabla f(\mathbf{x}_{k-1})^T \nabla f(\mathbf{x}_{k-1})} \quad (3c)$$

- Use exact line search

- Crucial properties:
 - (a) Step k minimizes f over a k -dimensional manifold; after at most n steps we find the optimal solution
 - (b) All directions \mathbf{p}_k are “conjugate”, that is, $\mathbf{p}_i^T \mathbf{Q} \mathbf{p}_j = 0$ for $i \neq j$
(Type of orthogonality; generation a type of Gram–Schmidt procedure applied to the negative gradients)
 - (c) Need only store the previous gradient to get new direction
 - (d) Strictly better convergence than steepest descent
 - (e) Direction vector \mathbf{p}_i is an eigenvector corresponding to a largest eigenvalue λ_i not yet found
 - (f) Meaning: Takes care of most difficult part of the problem first.
Less sensitive to the size of the condition number $\kappa(\mathbf{Q}) := \lambda_n/\lambda_1$ than steepest descent