Optimization Methods

Problem specification

Suppose we have a cost function (or objective function)

$$f(\mathbf{x}) : \mathbb{R}^N \longrightarrow \mathbb{R}$$

Our aim is to find values of the parameters (decision variables) $\mathbf{x}$ that minimize this function

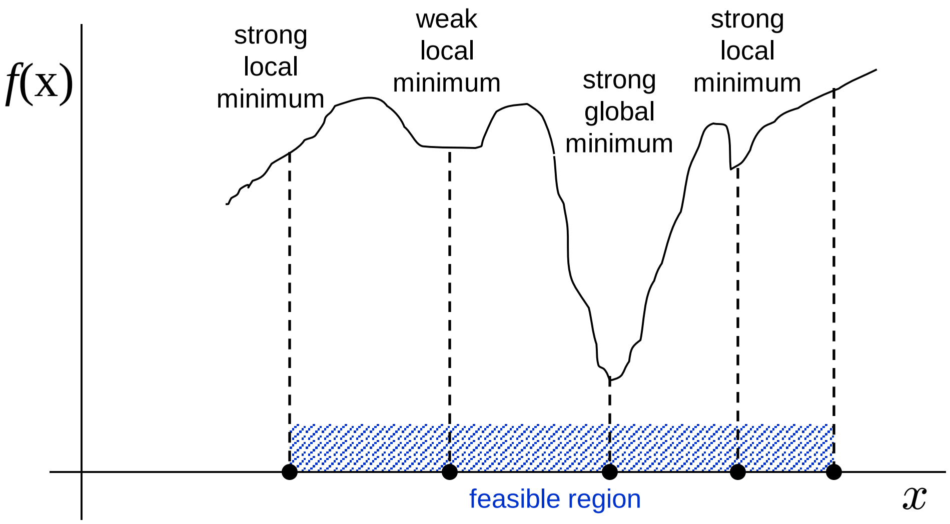
$$\mathbf{x}^* = \arg\min_{\mathbf{x}} f(\mathbf{x})$$

Subject to the following constraints:

- equality: $c_i(\mathbf{x}) = 0$
- nonequality: $c_j(\mathbf{x}) \geq 0$

If we seek a maximum of $f(\mathbf{x})$ (profit function) it is equivalent to seeking a minimum of $-f(\mathbf{x})$

Types of minima

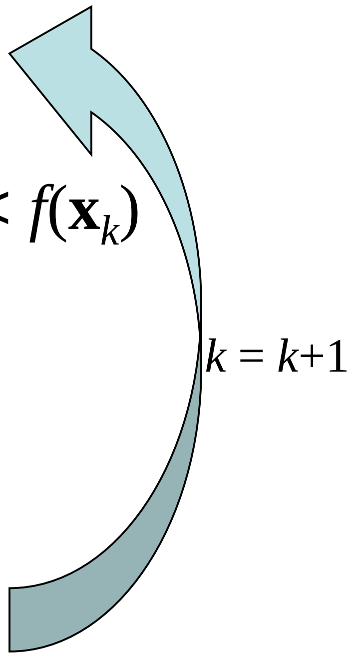


- which of the minima is found depends on the starting point

Iterative Optimization Algorithm

- Start at $\mathbf{x}_0$, $k = 0$.

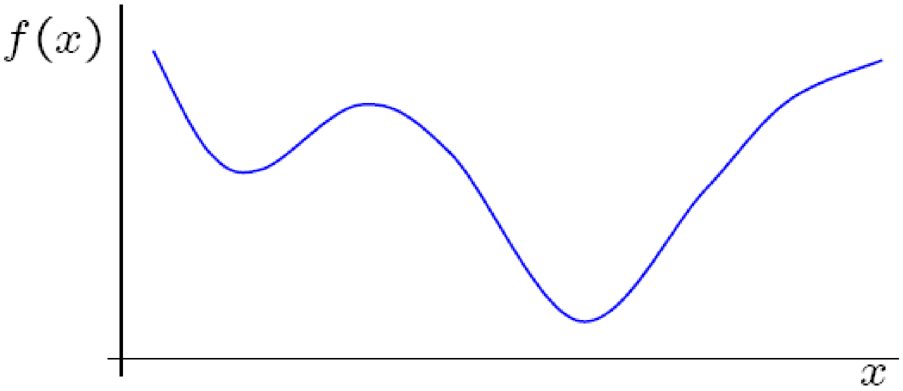
1. Compute a search direction $\mathbf{p}_k$
2. Compute a step length α_k, such that $f(\mathbf{x}_k + \alpha_k \mathbf{p}_k) < f(\mathbf{x}_k)$
3. Update $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k$
4. Check for convergence (stopping criteria)
   e.g. $df/d\mathbf{x} = \mathbf{0}$ or $\frac{\|x_{k+1} - x_k\|}{\|x_k\|} < \epsilon$



Reduces optimization in N dimensions to a series of (1D) line minimizations

Unconstrained univariate optimization

$$\min_x f(x)$$



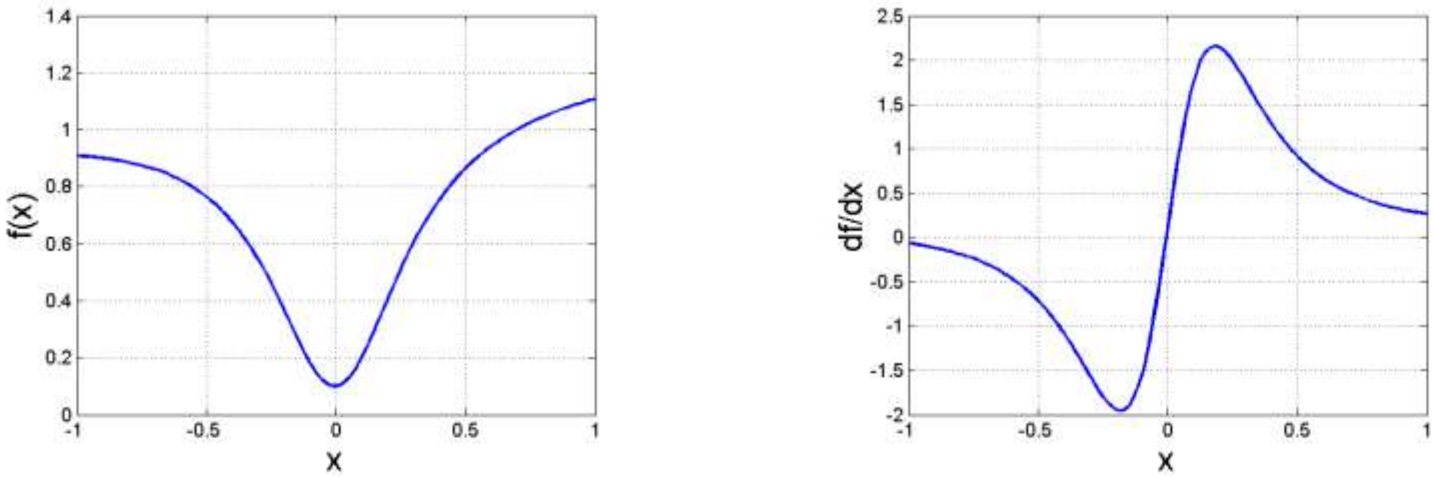
How to determine the minimum?

- Search methods (Dichotomous, Fibonacci, Golden-Section)
- Approximation methods
    1. Polynomial interpolation
    2. Newton method
- Combination of both (alg. of Davies, Swann, and Campey)
- Inexact Line Search (Fletcher)

1D function

As an example consider the function

$$f(x) = 0.1 + 0.1x + x^2/(0.1 + x^2)$$



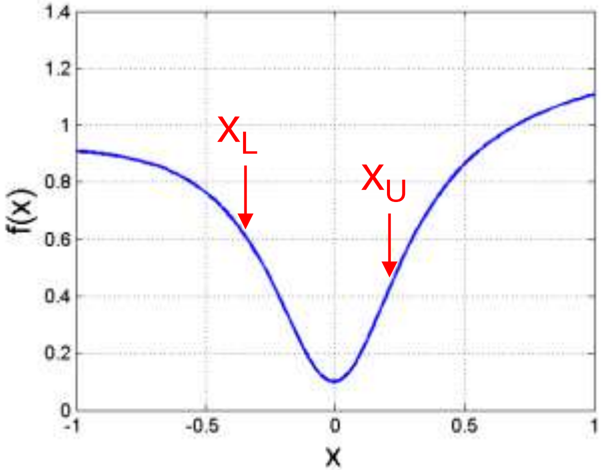
(Evaluation of the function is expensive.)

Search methods

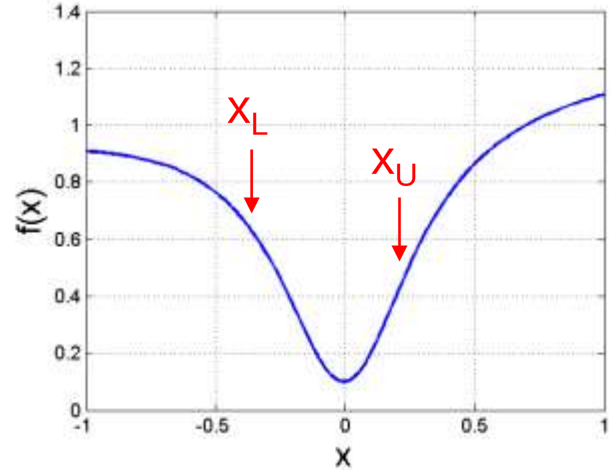
- Start with the interval ("bracket") $[x_L, x_U]$ such that the minimum x* lies inside.
- Evaluate $f(x)$ at two point inside the bracket.
- Reduce the bracket.
- Repeat the process.

- Can be applied to any function and differentiability is not essential.

Search methods



Dichotomous

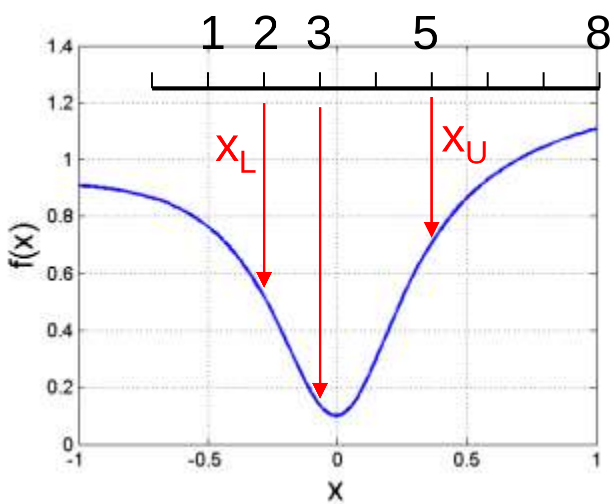


Fibonacci:

1 1 2 3 5 8 …

I_{k+5} I_{k+4} I_{k+3} I_{k+2} I_{k+1} I_k

$$I_k = I_{k+1} + I_{k+2}$$

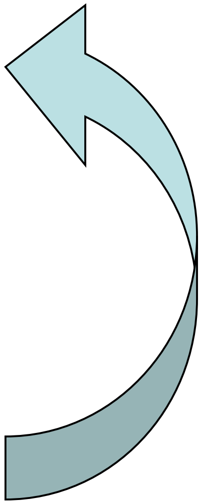


Golden-Section Search divides intervals by $K = 1.6180$

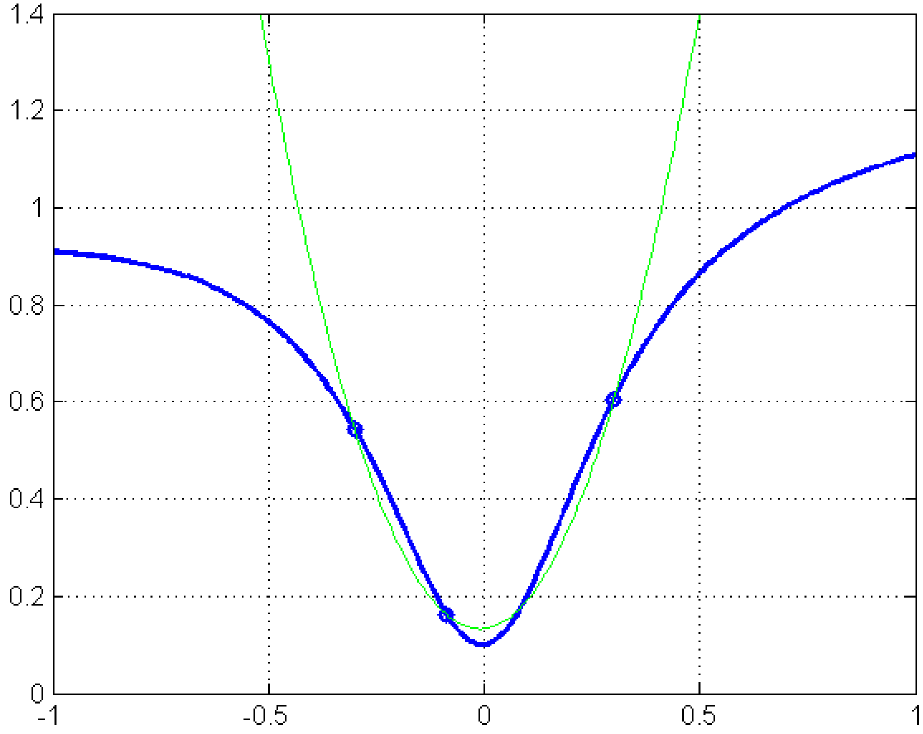
$$\frac{I_k}{I_{k+1}} = K$$

Polynomial interpolation

- Bracket the minimum.
- Fit a quadratic or cubic polynomial which interpolates $f(x)$ at some points in the interval.
- Jump to the (easily obtained) minimum of the polynomial.
- Throw away the worst point and repeat the process.



Polynomial interpolation



- Quadratic interpolation using 3 points, 2 iterations
- Other methods to interpolate?
  - 2 points and one gradient
  - Cubic interpolation

Newton method

Fit a quadratic approximation to $f(x)$ using both gradient and curvature information at x.

- Expand $f(x)$ locally using a Taylor series.

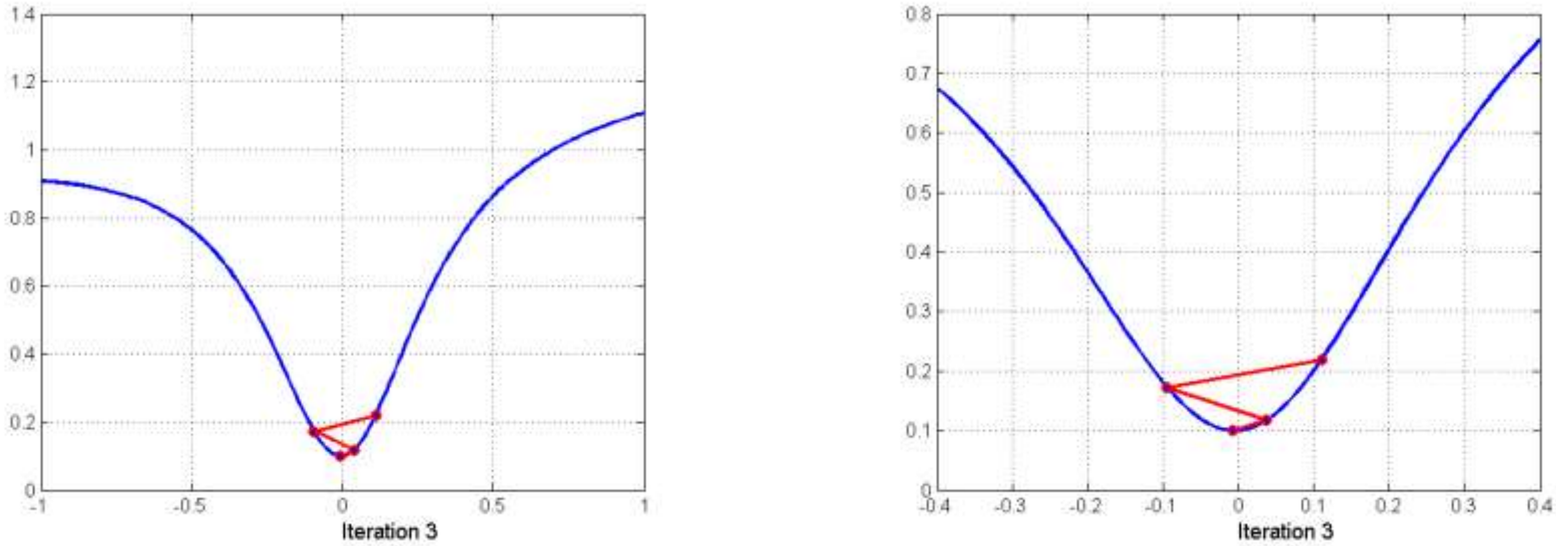
$$f(x+\delta x) = f(x) + f'(x)\delta x + \frac{1}{2}f''(x)\delta x^2 + o(\delta x^2)$$

- Find the δx which minimizes this local quadratic approximation.

$$\delta x = -\frac{f'(x)}{f''(x)}$$

- Update x. $$x_{n+1} = x_n - \delta x = x_n - \frac{f'(x)}{f''(x)}$$

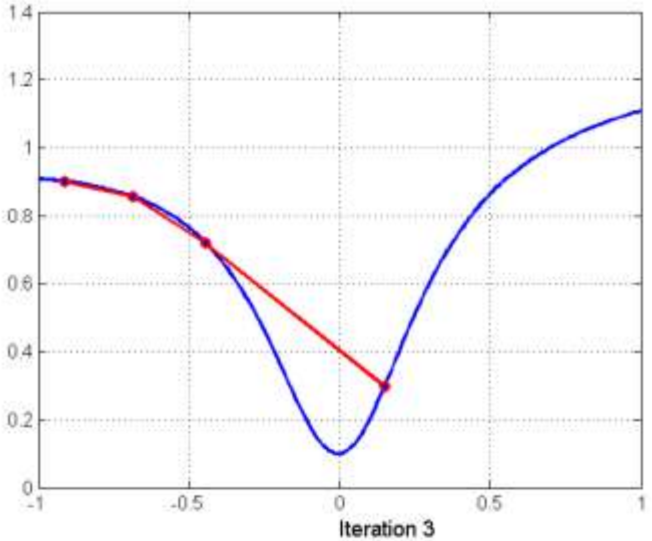
Newton method

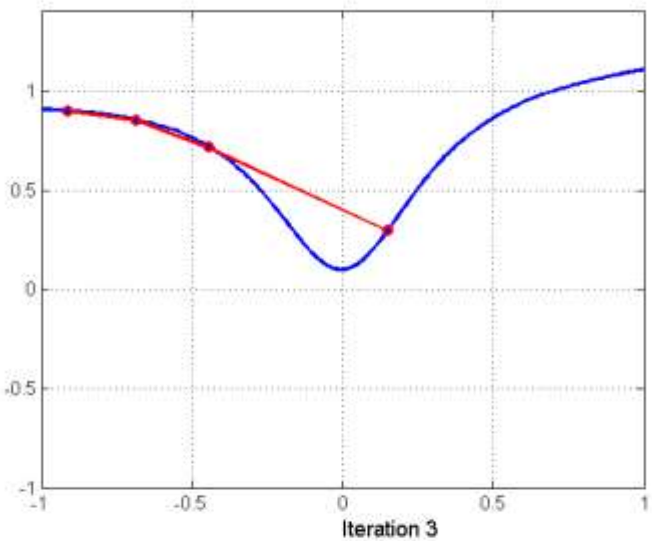


- avoids the need to bracket the root
- quadratic convergence (decimal accuracy doubles at every iteration)

Newton method

- Global convergence of Newton's method is poor.
- Often fails if the starting point is too far from the minimum.

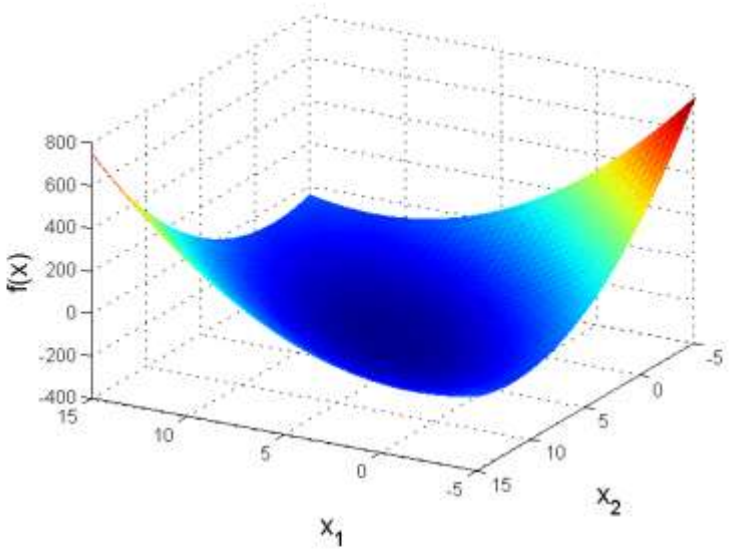


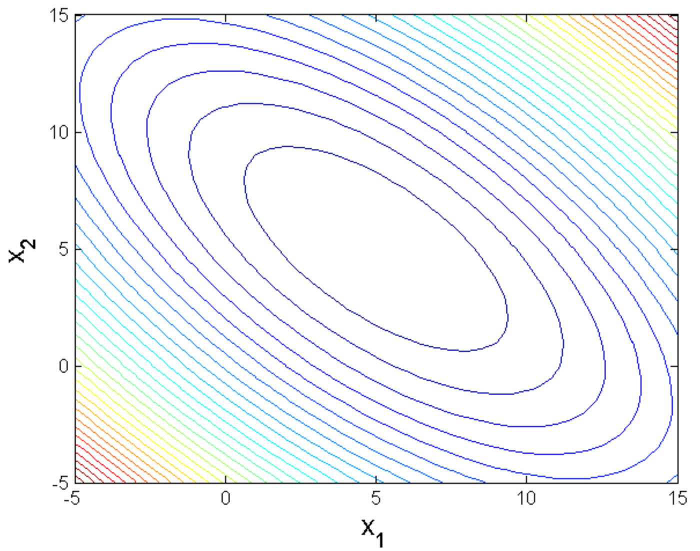


- in practice, must be used with a globalization strategy which reduces the step length until function decrease is assured

Extension to N (multivariate) dimensions

- How big N can be?
  - problem sizes can vary from a handful of parameters to many thousands
- We will consider examples for N=2, so that cost function surfaces can be visualized.





Taylor expansion

A function may be approximated locally by its Taylor series expansion about a point $\mathbf{x}^*$

$$f(\mathbf{x}^* + \mathbf{x}) \approx f(\mathbf{x}^*) + \nabla f^T \mathbf{x} + \frac{1}{2}\mathbf{x}^T \mathbf{H} \mathbf{x}$$

where the gradient $\nabla f(\mathbf{x}^*)$ is the vector

$$\nabla f(\mathbf{x}^*) = \left[\frac{\partial f}{x_1} \dots \frac{\partial f}{x_N}\right]^T$$

and the Hessian $\mathbf{H}(\mathbf{x}^*)$ is the symmetric matrix

$$\mathbf{H}(\mathbf{x}^*) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_N} \\ \vdots & & \vdots \\ \frac{\partial^2 f}{\partial x_N \partial x_1} & \cdots & \frac{\partial^2 f}{\partial x_N^2} \end{bmatrix}$$

Quadratic functions

$$f(\mathbf{x}) = a + \mathbf{g}^T\mathbf{x} + \frac{1}{2}\mathbf{x}^T\mathbf{H}\mathbf{x}$$

- The vector $\mathbf{g}$ and the Hessian $\mathbf{H}$ are constant.
- Second order approximation of any function by the Taylor expansion is a quadratic function.

We will assume only quadratic functions for a while.

Necessary conditions for a minimum

$$f(\mathbf{x}) = a + \mathbf{g}^T\mathbf{x} + \frac{1}{2}\mathbf{x}^T\mathbf{H}\mathbf{x}$$

Expand $f(\mathbf{x})$ about a stationary point $\mathbf{x}^*$ in direction $\mathbf{p}$

$$f(\mathbf{x}^* + \alpha\mathbf{p}) = f(\mathbf{x}^*) + \nabla f(\mathbf{x}^*)^T \alpha\mathbf{p} + \frac{1}{2}\alpha^2\mathbf{p}^T\mathbf{H}\mathbf{p}$$

$$= f(\mathbf{x}^*) + \frac{1}{2}\alpha^2\mathbf{p}^T\mathbf{H}\mathbf{p}$$

since at a stationary point $\nabla f(\mathbf{x}^*) = 0$

At a stationary point the behavior is determined by $\mathbf{H}$

- $\mathbf{H}$ is a symmetric matrix, and so has orthogonal eigenvectors

$$\mathbf{H}\mathbf{u}_i = \lambda_i \mathbf{u}_i \qquad \|\mathbf{u}_i\| = 1$$

$$\begin{aligned} f(\mathbf{x}^* + \alpha \mathbf{u}_i) &= f(\mathbf{x}^*) + \frac{1}{2}\alpha^2 \mathbf{u}_i^T \mathbf{H} \mathbf{u}_i \\ &= f(\mathbf{x}^*) + \frac{1}{2}\alpha^2 \lambda_i \end{aligned}$$

- As $|\alpha|$ increases, $f(\mathbf{x}^* + \alpha \mathbf{u}_i)$ increases, decreases or is unchanging according to whether λ_i is positive, negative or zero

Examples of quadratic functions

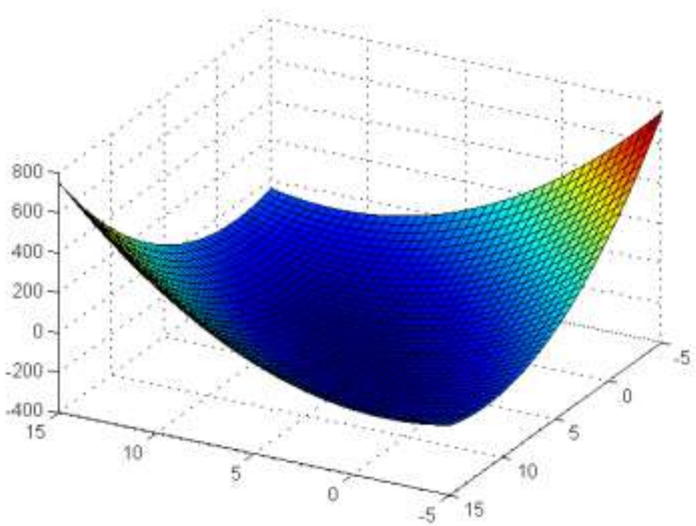
Case 1: both eigenvalues positive

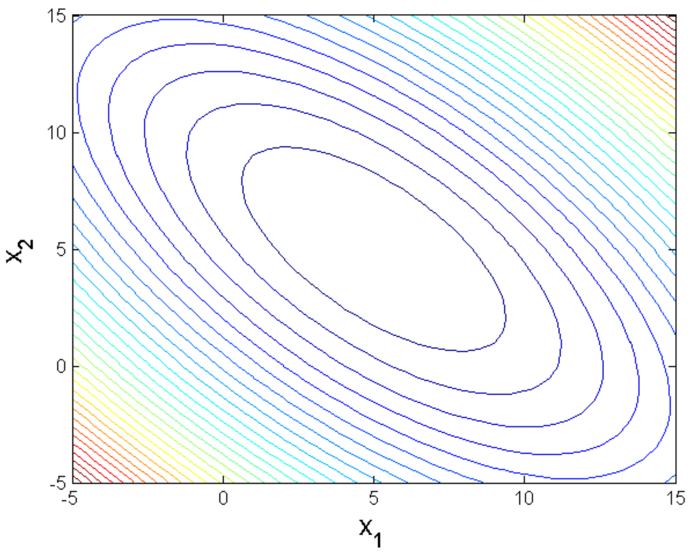
$$f(\mathbf{x}) = a + \mathbf{g}^T\mathbf{x} + \frac{1}{2}\mathbf{x}^T\mathbf{H}\mathbf{x}$$

with

$$a = 0\,, \qquad \mathbf{g} = \begin{bmatrix} -50 \\ -50 \end{bmatrix}\,, \qquad \mathbf{H} = \begin{bmatrix} 6 & 4 \\ 4 & 6 \end{bmatrix}$$

positive definite





minimum

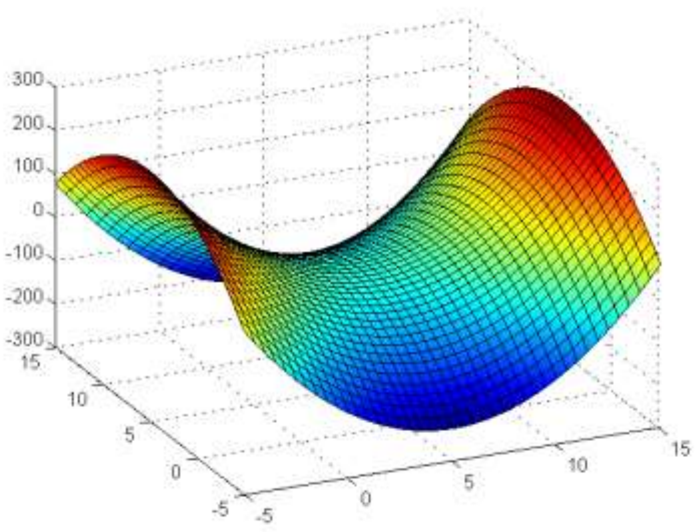
Examples of quadratic functions

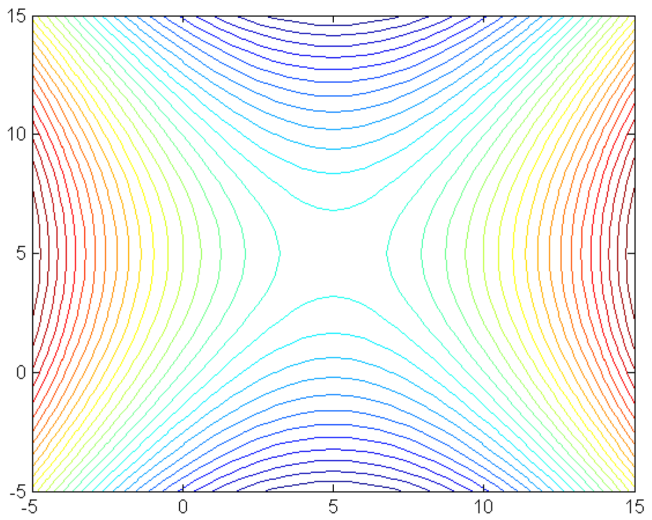
Case 2: eigenvalues have different sign

$$f(\mathbf{x}) = a + \mathbf{g}^T\mathbf{x} + \frac{1}{2}\mathbf{x}^T\mathbf{H}\mathbf{x}$$

with

$$a = 0\,, \qquad \mathbf{g} = \begin{bmatrix} -30 \\ 20 \end{bmatrix}\,, \qquad \mathbf{H} = \begin{bmatrix} 6 & 0 \\ 0 & -4 \end{bmatrix}$$ indefinite





saddle point

Examples of quadratic functions

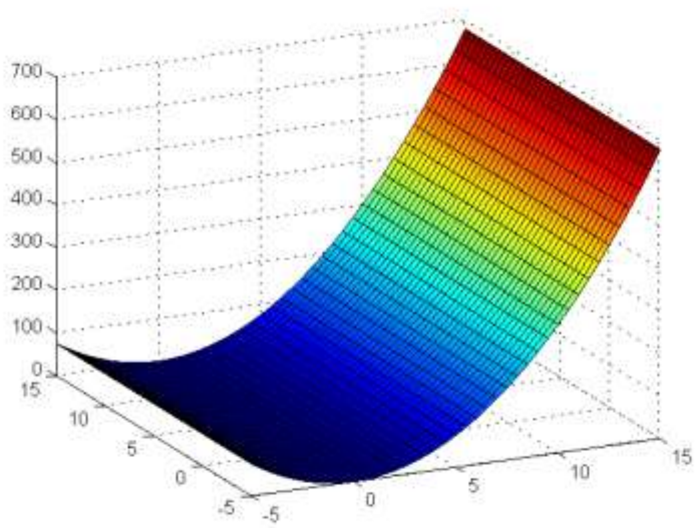
Case 3: one eigenvalues is zero

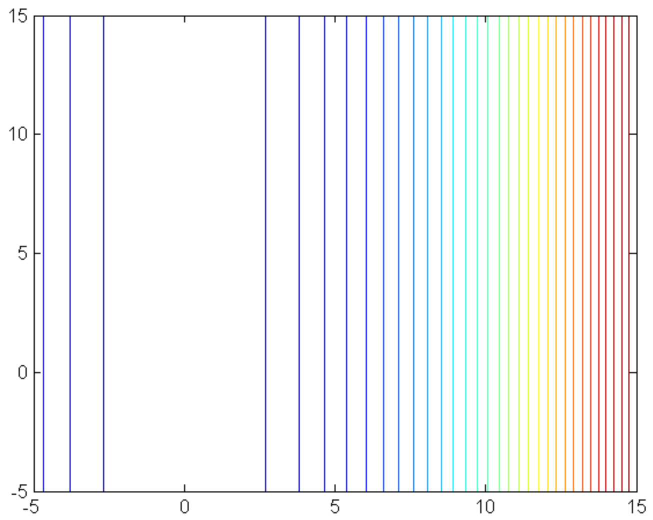
$$f(\mathbf{x}) = a + \mathbf{g}^T\mathbf{x} + \frac{1}{2}\mathbf{x}^T\mathbf{H}\mathbf{x}$$

with

$$a = 0\,, \qquad \mathbf{g} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}\,, \qquad \mathbf{H} = \begin{bmatrix} 6 & 0 \\ 0 & 0 \end{bmatrix}$$

positive semidefinite





parabolic cylinder

Optimization for quadratic functions

Assume that $\mathbf{H}$ is positive definite

$$f(\mathbf{x}) = a + \mathbf{g}^T\mathbf{x} + \frac{1}{2}\mathbf{x}^T\mathbf{H}\mathbf{x}$$

$$\nabla f(\mathbf{x}) = \mathbf{g} + \mathbf{H}\mathbf{x}$$

There is a unique minimum at

$$\mathbf{x}^* = -\mathbf{H}^{-1}\mathbf{g}$$

If N is large, it is not feasible to perform this inversion directly.

How to find descent directions?

- Start at $\mathbf{x}_0$, $k = 0$.

1. Compute a search direction $\mathbf{p}_k$
2. Compute a step length α_k, such that $f(\mathbf{x}_k + \alpha_k \mathbf{p}_k) < f(\mathbf{x}_k)$
3. Update $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k$
4. Check for convergence (stopping criteria)

Steepest descent

- Basic principle is to minimize the N-dimensional function by a series of 1D line-minimizations:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k$$

- The steepest descent method chooses $\mathbf{p}_k$ to be parallel to the gradient

$$\mathbf{p}_k = -\nabla f(\mathbf{x}_k)$$

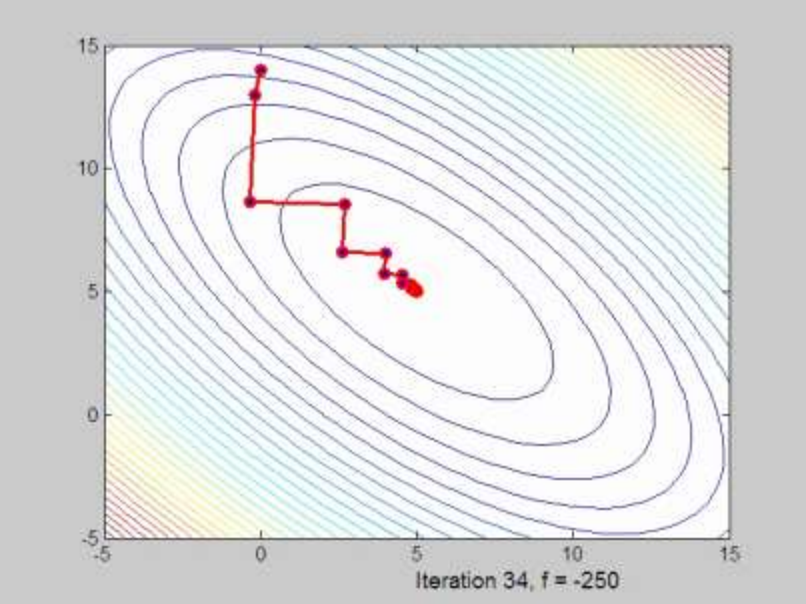
- Step-size α_k is chosen to minimize $f(\mathbf{x}_k + \alpha_k \mathbf{p}_k)$.

  For quadratic forms there is a closed form solution:

$$\alpha_k = \frac{\mathbf{p}_k^T \mathbf{p}_k}{\mathbf{p}_k^T \mathbf{H} \mathbf{p}_k}$$

Prove it!

Steepest descent



- The gradient is everywhere perpendicular to the contour lines.
- After each line minimization the new gradient is always *orthogonal* to the previous step direction (true of any line minimization).
- Consequently, the iterates tend to zig-zag down the valley in a very inefficient manner

Conjugate gradient

- Each $\mathbf{p}_k$ is chosen to be conjugate to all previous search directions with respect to the Hessian $\mathbf{H}$:

$$\mathbf{p}_i^T \mathbf{H} \mathbf{p}_j = 0 , \qquad i \neq j$$

- The resulting search directions are mutually linearly independent.

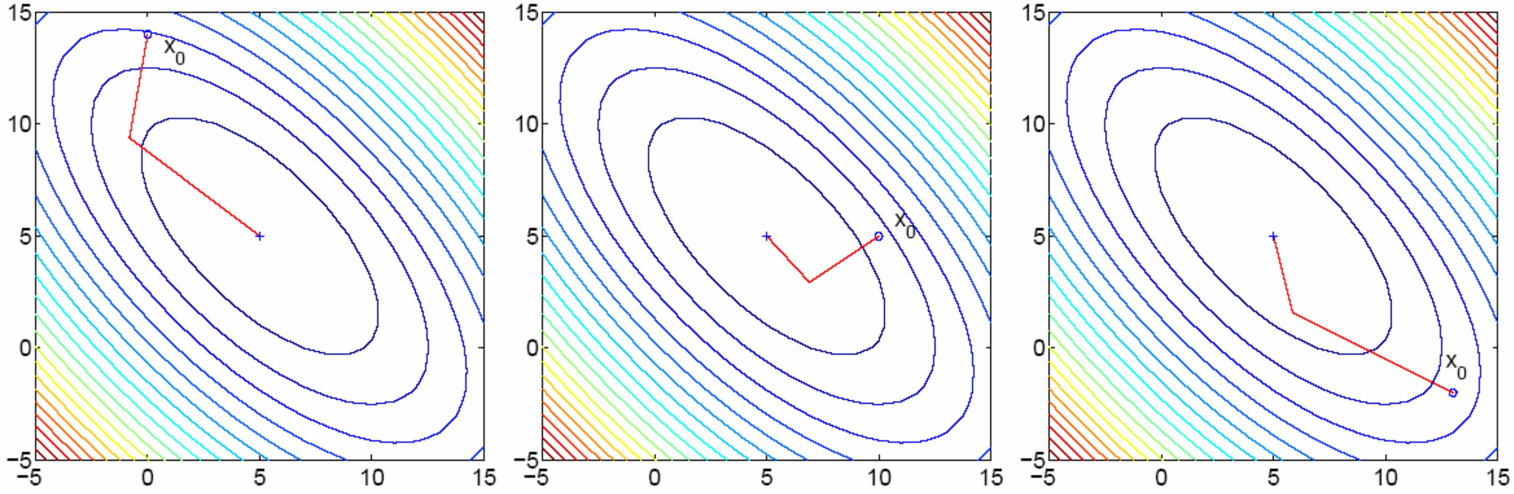
Prove it!

- *Remarkably*, $\mathbf{p}_k$ can be chosen using only knowledge of $\mathbf{p}_{k-1}$, $\nabla f(\mathbf{x}_{k-1})$, and $\nabla f(\mathbf{x}_k)$

$$\mathbf{p}_k = \nabla f_k + \left(\frac{\nabla f_k^\top \nabla f_k}{\nabla f_{k-1}^\top \nabla f_{k-1}} \right) \mathbf{p}_{k-1}$$

Conjugate gradient

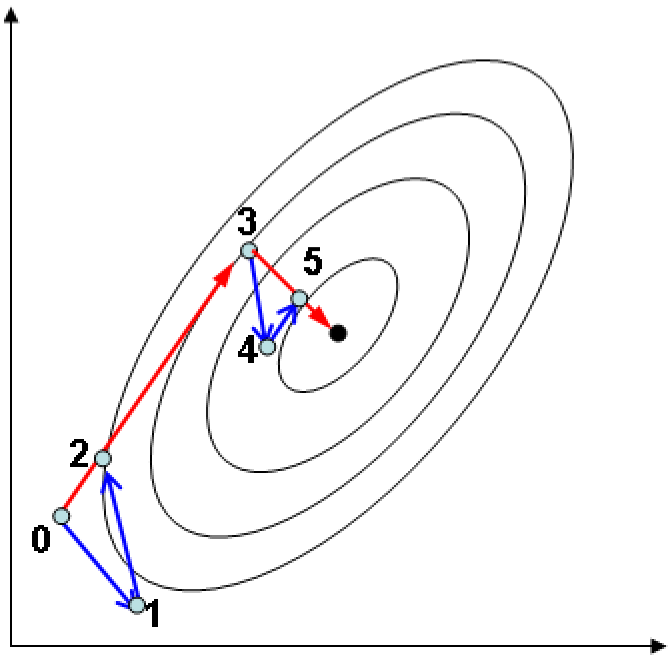
- An N-dimensional quadratic form can be minimized in at most N conjugate descent steps.



- 3 different starting points.
- Minimum is reached in exactly 2 steps.

Powell's Algorithm

- Conjugate-gradient method that does not require derivatives
- Conjugate directions are generated through a series of line searches

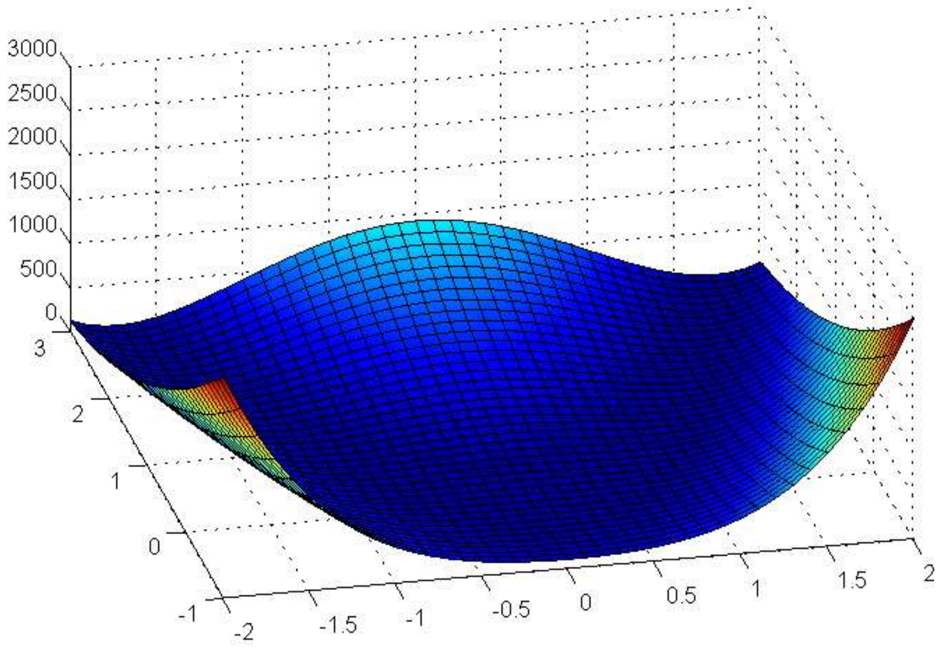


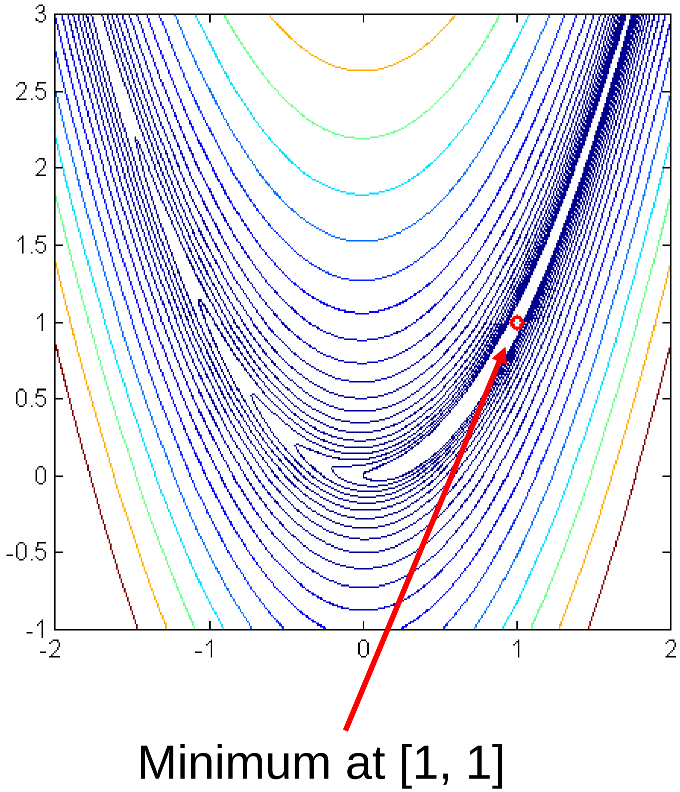
- N-dim quadratic function is minimized with N(N+1) line searches

Optimization of general functions

E.g., Rosenbrock's function:

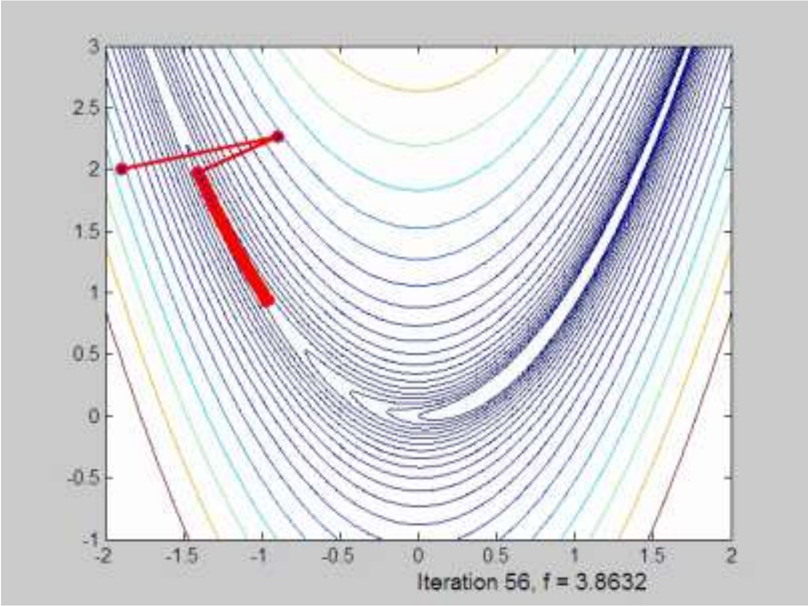
$$f(x, y) = 100(y - x^2)^2 + (1 - x)^2$$

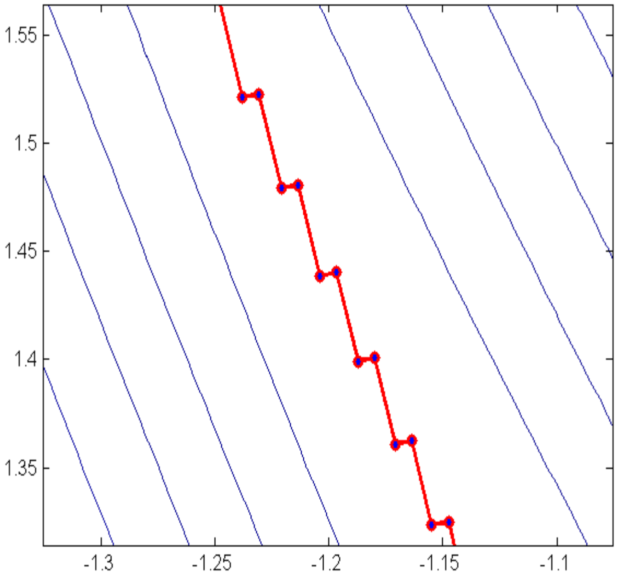




Steepest descent

- The 1D line minimization must be performed using one of the earlier methods (usually cubic polynomial interpolation)

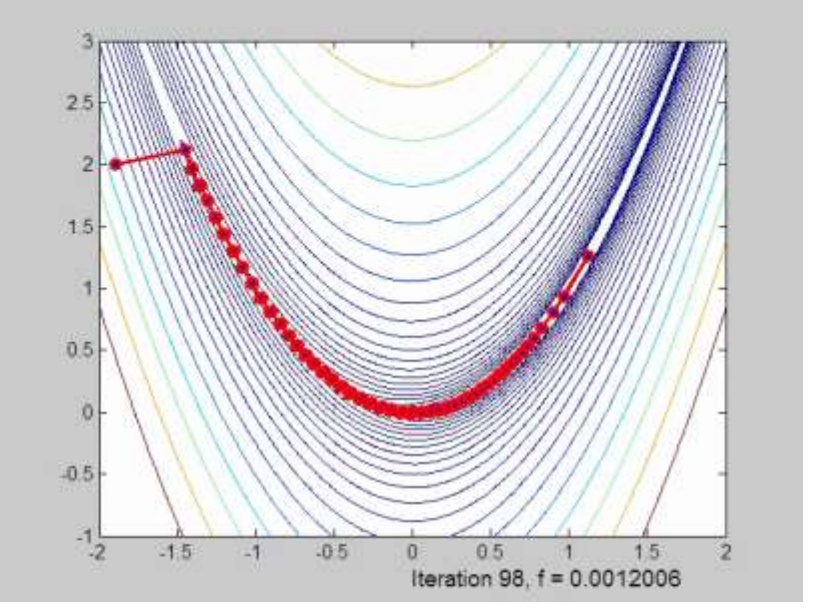




- The zig-zag behaviour is clear in the zoomed view
- The algorithm crawls down the valley

Conjugate gradient

- Again, an explicit line minimization must be used at every step



- The algorithm converges in 98 iterations
- Far superior to steepest descent

Newton method

Expand $f(\mathbf{x})$ by its Taylor series about the point $\mathbf{x}_k$

$$f(\mathbf{x}_k + \delta\mathbf{x}) \approx f(\mathbf{x}_k) + \mathbf{g}_k^T \delta\mathbf{x} + \frac{1}{2}\delta\mathbf{x}^T \mathbf{H}_k \delta\mathbf{x}$$

where the gradient is the vector

$$\mathbf{g}_k = \nabla f(\mathbf{x}_k) = \left[\frac{\partial f}{x_1} \dots \frac{\partial f}{x_N}\right]^T$$

and the Hessian is the symmetric matrix

$$\mathbf{H}_k = \mathbf{H}(\mathbf{x}_k) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_N} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_N \partial x_1} & \cdots & \frac{\partial^2 f}{\partial x_N^2} \end{bmatrix}$$

Newton method

For a minimum we require that $\nabla f(\mathbf{x}) = \mathbf{0}$, and so

$$\nabla f(\mathbf{x}) = \mathbf{g}_k + \mathbf{H}_k \delta\mathbf{x} = \mathbf{0}$$

with solution $\delta\mathbf{x} = -\mathbf{H}_k^{-1}\mathbf{g}_k$. This gives the iterative update

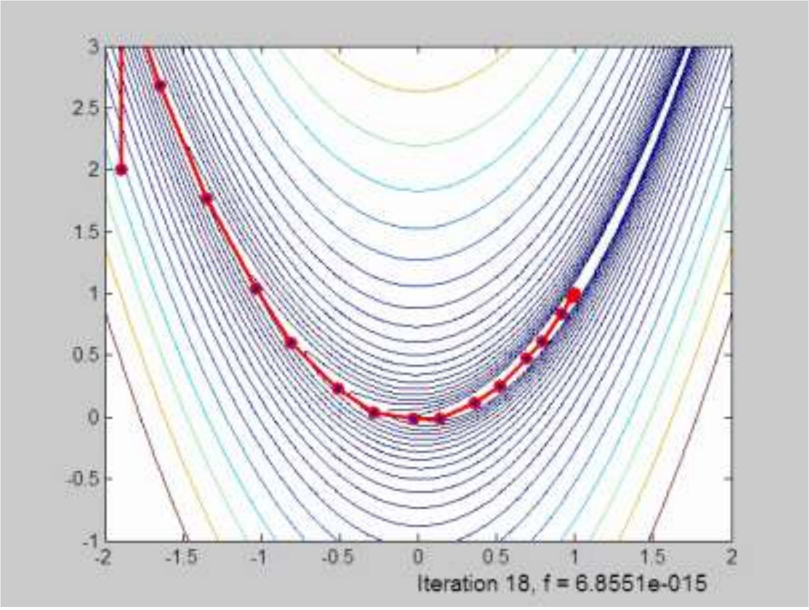
$$\mathbf{x}_{k+1} = \mathbf{x}_k - \mathbf{H}_k^{-1}\mathbf{g}_k$$

- If $f(\mathbf{x})$ is quadratic, then the solution is found in one step.
- The method has quadratic convergence (as in the 1D case).
- The solution $\delta\mathbf{x} = -\mathbf{H}_k^{-1}\mathbf{g}_k$ is guaranteed to be a downhill direction.
- Rather than jump straight to the minimum, it is better to perform a line minimization which ensures global convergence

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha_k \mathbf{H}_k^{-1}\mathbf{g}_k$$

- If $\mathbf{H}=\mathbf{I}$ then this reduces to steepest descent.

Newton method - example



- The algorithm converges in only 18 iterations compared to the 98 for conjugate gradients.
- However, the method requires computing the Hessian matrix at each iteration – this is not always feasible

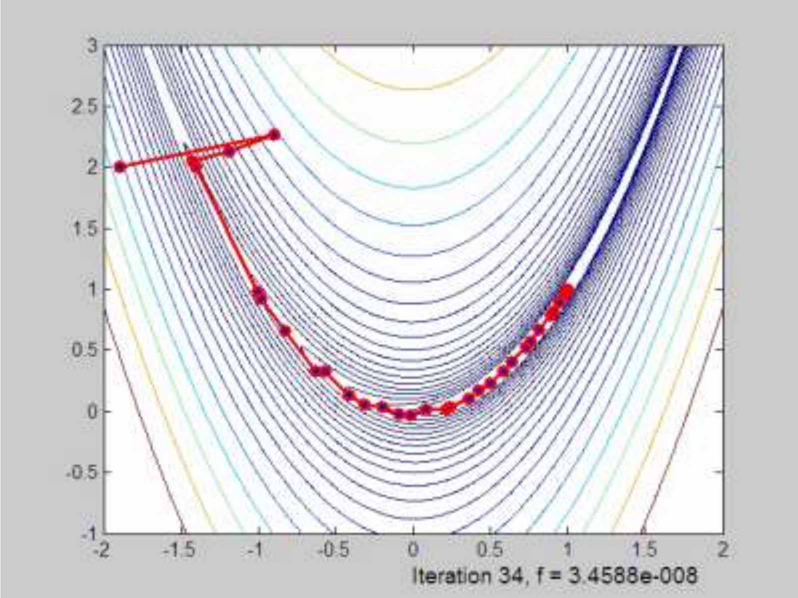
Quasi-Newton methods

- If the problem size is large and the Hessian matrix is dense then it may be infeasible/inconvenient to compute it directly.
- Quasi-Newton methods avoid this problem by keeping a "rolling estimate" of H(x), updated at each iteration using new gradient information.
- Common schemes are due to Broyden, Goldfarb, Fletcher and Shanno (BFGS), and also Davidson, Fletcher and Powell (DFP).
- The idea is based on the fact that for quadratic functions holds

$$\mathbf{g}_{k+1} - \mathbf{g}_k = \mathbf{H}(\mathbf{x}_{k+1} - \mathbf{x}_k)$$

and by accumulating $\mathbf{g}_k$'s and $\mathbf{x}_k$'s we can calculate $\mathbf{H}$.

BFGS example



- The method converges in 34 iterations, compared to 18 for the full-Newton method

Non-linear least squares

- It is very common in applications for a cost function $f(\mathbf{x})$ to be the sum of a large number of squared residuals

$$f(\mathbf{x}) = \sum_{i=1}^{M} r_i^2(\mathbf{x})$$

- If each residual depends non-linearly on the parameters $\mathbf{x}$ then the minimization of $f(\mathbf{x})$ is a non-linear least squares problem.

Non-linear least squares

$$f(\mathbf{x}) = \sum_{i=1}^{M} r_i^2(\mathbf{x})$$

- The M × N Jacobian of the vector of residuals r is defined as

$$J(\mathbf{x}) = \begin{bmatrix} \frac{\partial r_1}{\partial x_1} & \cdots & \frac{\partial r_1}{\partial x_N} \\ \vdots & \ddots & \vdots \\ \frac{\partial r_M}{\partial x_1} & \cdots & \frac{\partial r_M}{\partial x_N} \end{bmatrix}$$

- Consider

$$\frac{\partial f}{\partial x_k} = \frac{\partial}{\partial x_k} \sum_i r_i^2 = \sum_i 2 r_i \frac{\partial r_i}{\partial x_k}$$

- Hence

$$\nabla f(\mathbf{x}) = 2\mathbf{J}^T\mathbf{r}$$

Non-linear least squares

- For the Hessian holds

$$\frac{\partial^2 f}{\partial x_k \partial x_l} = \underbrace{2\sum_i \frac{\partial r_i}{\partial x_l}\frac{\partial r_i}{\partial x_k}}_{} + 2\sum_i r_i \frac{\partial^2 r_i}{\partial x_k \partial x_l}$$

$$\mathbf{H}(\mathbf{x}) \approx \quad 2\mathbf{J}^T\mathbf{J}$$

Gauss-Newton approximation

- Note that the second-order term in the Hessian is multiplied by the residuals r_i.
- In most problems, the residuals will typically be small.
- Also, at the minimum, the residuals will typically be distributed with mean = 0.
- For these reasons, the second-order term is often ignored.
- Hence, explicit computation of the full Hessian can again be avoided.

Gauss-Newton example

- The minimization of the Rosenbrock function

$$f(x, y) = 100(y - x^2)^2 + (1 - x)^2$$

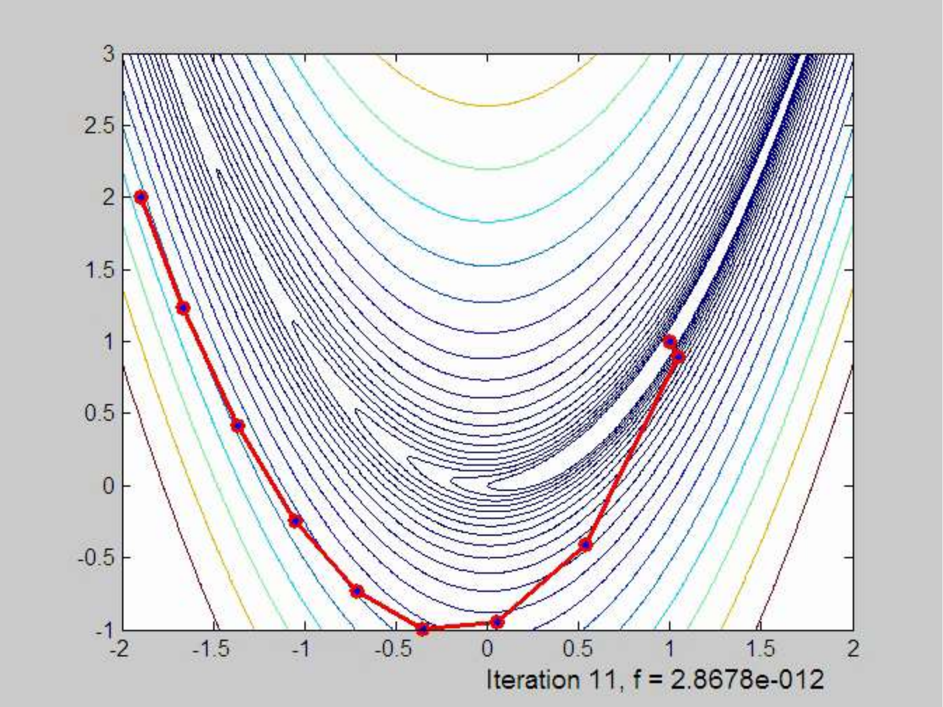
- can be written as a least-squares problem with residual vector

$$\mathbf{r} = \begin{bmatrix} 10(y - x^2) \\ (1 - x) \end{bmatrix}$$

$$\mathbf{J} = \begin{bmatrix} \frac{\partial r_1}{\partial x} & \frac{\partial r_1}{\partial y} \\ \frac{\partial r_2}{\partial x} & \frac{\partial r_2}{\partial y} \end{bmatrix} = \begin{bmatrix} -20x & 10 \\ -1 & 0 \end{bmatrix}$$

Gauss-Newton example

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha_k \mathbf{H}_k^{-1} \mathbf{g}_k \qquad \mathbf{H}_k = 2\mathbf{J}_k^T \mathbf{J}_k$$



- minimization with the Gauss-Newton approximation with line search takes only 11 iterations

Levenberg-Marquardt Algorithm

- For non-linear least square problems
- Combines Gauss-Newton with Steepest Descent
- Fast convergence even for very "flat" functions
- Descend direction $\delta \mathbf{x}$:
  - Newton

    $$\mathbf{H} \delta \mathbf{x} = -\mathbf{g}$$

    $$\mathbf{J}^T \mathbf{J} \delta \mathbf{x} = -\mathbf{J}^T \mathbf{r}$$

  - Steepest Descent

    $$\delta \mathbf{x} = -\mathbf{g}$$

$$(\mathbf{J}^T \mathbf{J} + \lambda \mathbf{I}) \delta \mathbf{x} = -\mathbf{J}^T \mathbf{r}$$

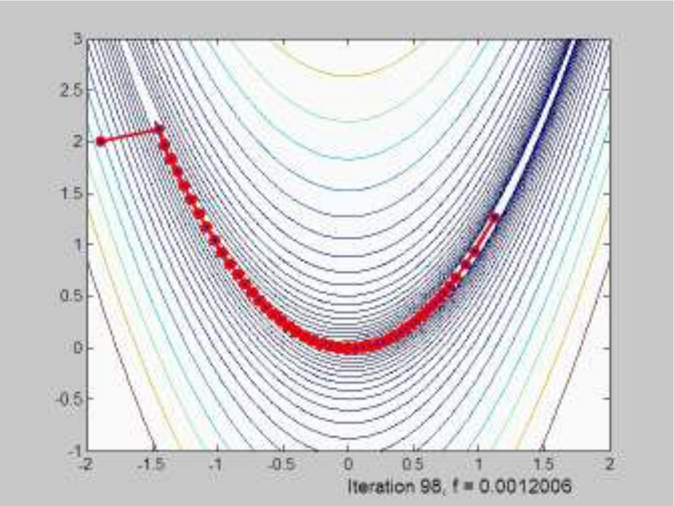
$$(\mathbf{J}^T \mathbf{J} + \lambda \mathrm{diag}(\mathbf{J}^T \mathbf{J})) \delta \mathbf{x} = -\mathbf{J}^T \mathbf{r}$$

Gauss-Newton:

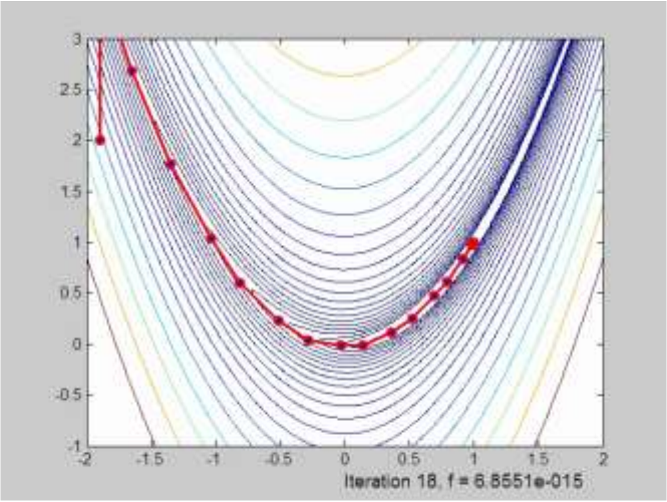
$$\mathbf{g} = 2\mathbf{J}^T \mathbf{r}$$

$$\mathbf{H} = 2\mathbf{J}^T \mathbf{J}$$

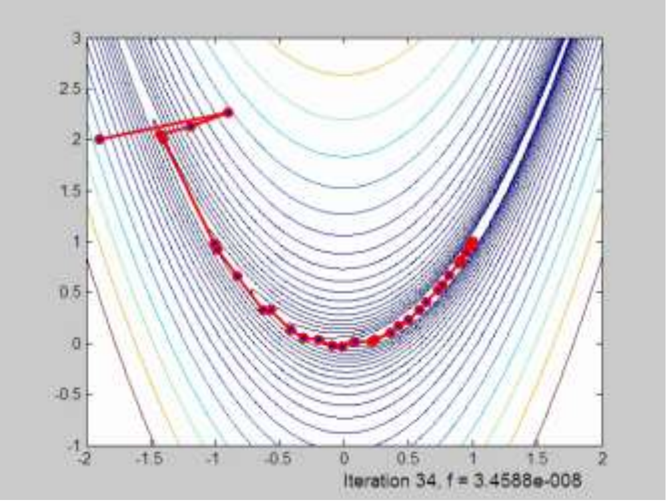
Comparison



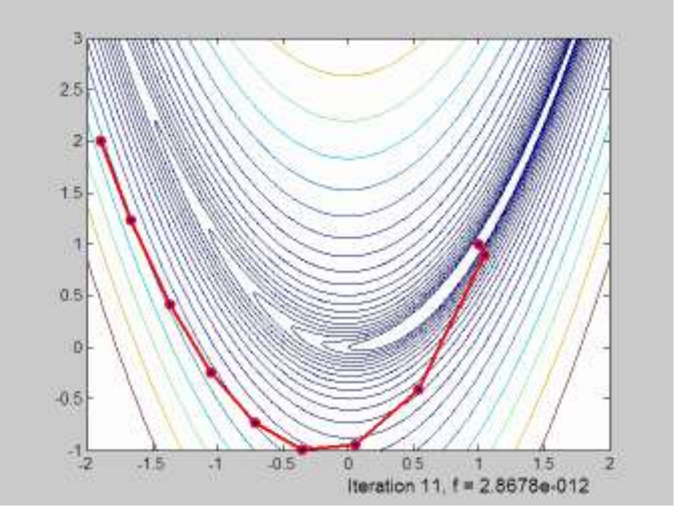
CG



Newton



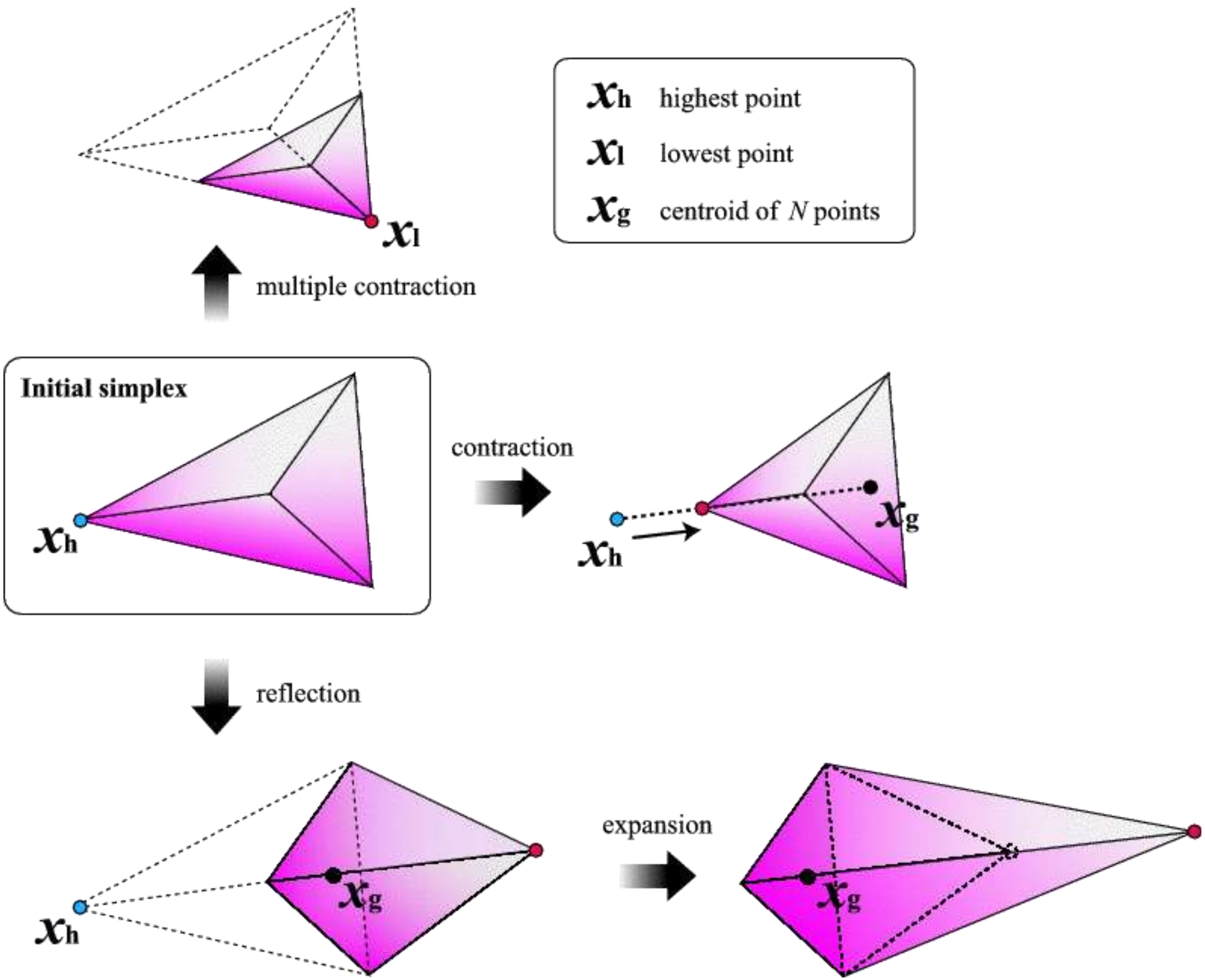
Quasi-Newton



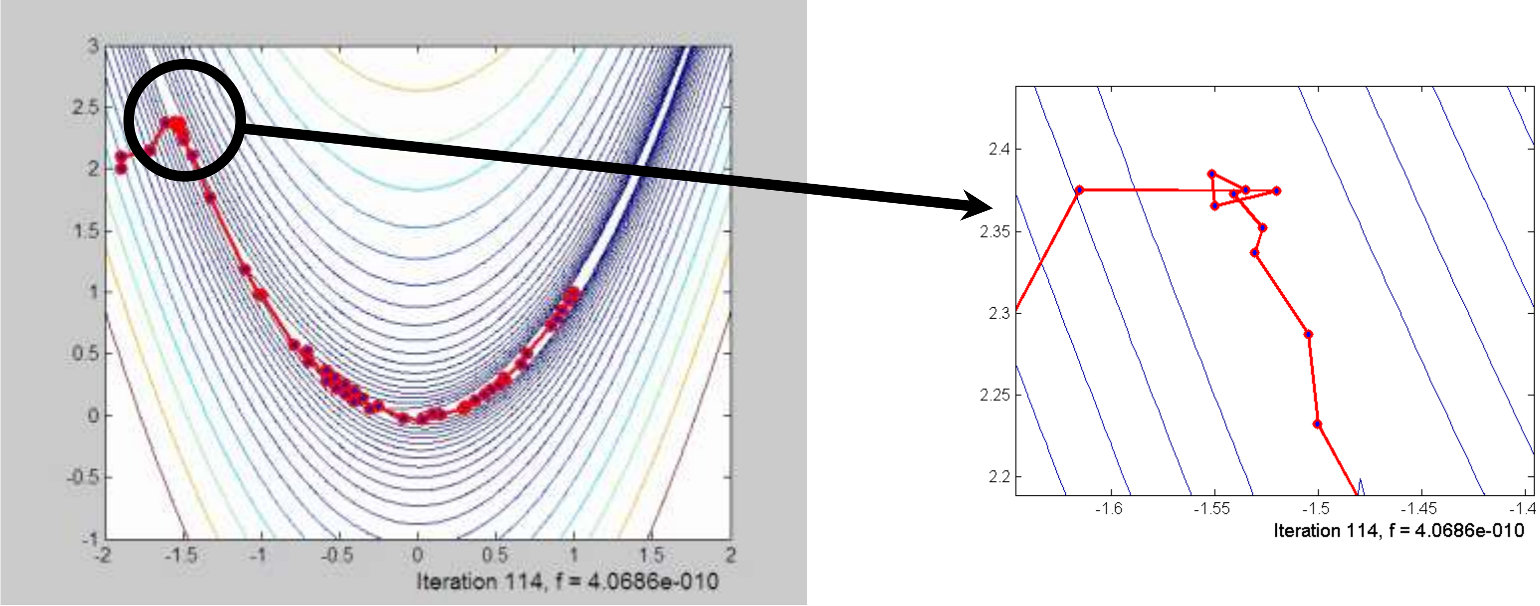
Gauss-Newton

Derivative-free optimization

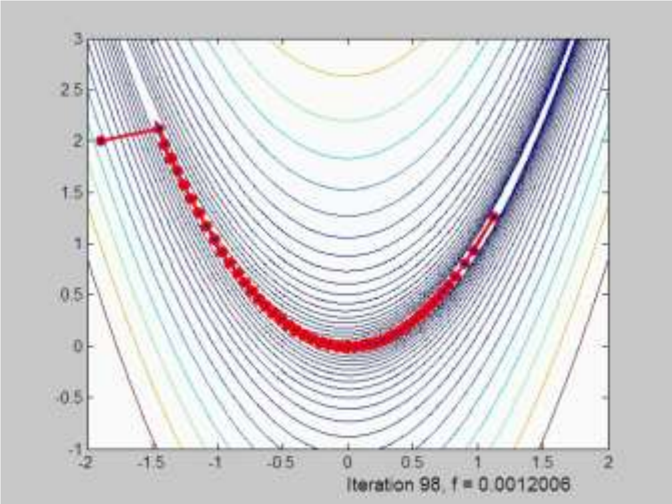
Downhill simplex method



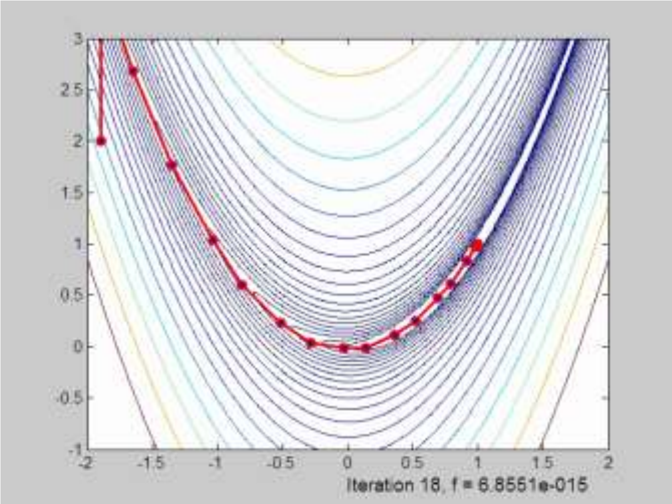
Downhill Simplex



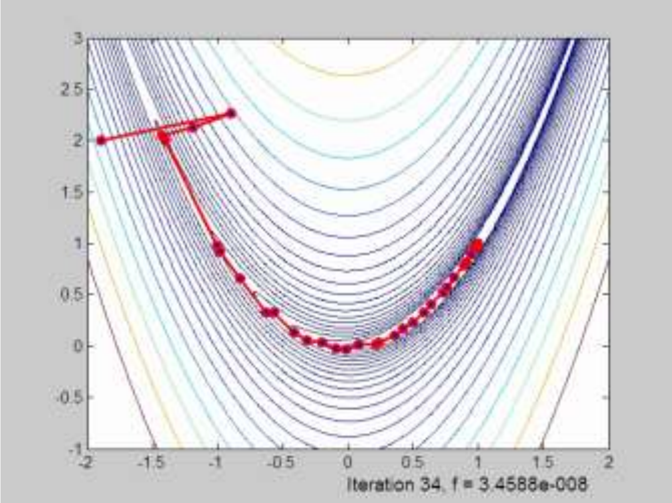
Comparison



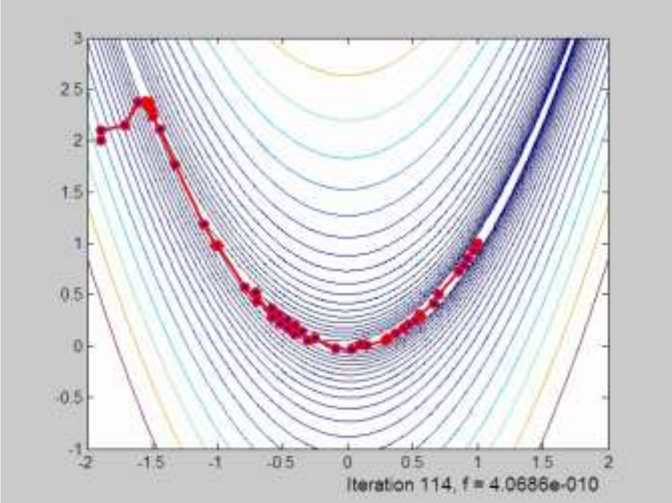
CG



Newton



Quasi-Newton



Downhill Simplex

Rates of Convergence

$$\beta = \lim_{k\to\infty} \frac{\|x_{k+1} - x^*\|}{\|x_k - x^*\|^p}$$

x^* … minimum

p … order of convergence

β … convergence ratio

Linear conv.:	p=1, $\beta<1$
Superlinear conv.:	p=1, $\beta=0$ or p=>2
Quadratic conv.:	p=2

Constrained Optimization

$$f(\mathbf{x}) : \mathbb{R}^N \longrightarrow \mathbb{R}$$

$$\mathbf{x}^* = \arg\min_{\mathbf{x}} f(\mathbf{x})$$

Subject to:

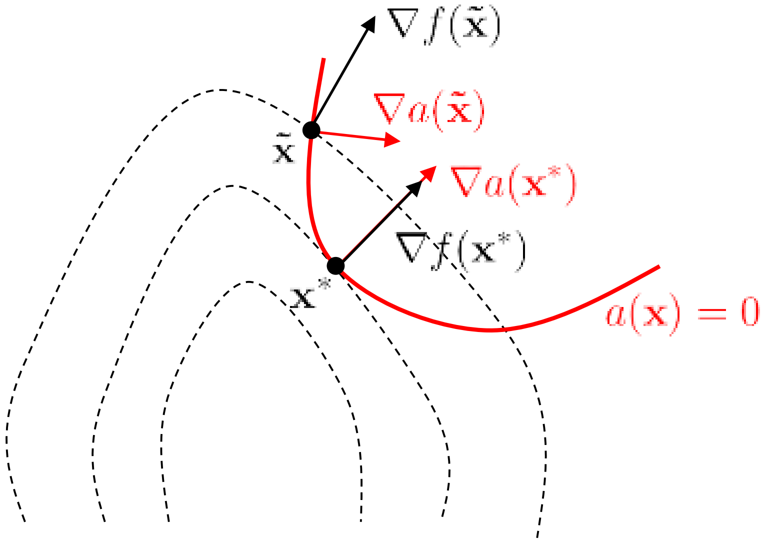
- Equality constraints: $a_i(\mathbf{x}) = 0 \quad i = 1, 2, \ldots, p$
- Nonequality constraints: $c_j(x) \leq 0 \quad j = 1, 2, \ldots, q$

- Constraints define a feasible region, which is nonempty.
- The idea is to convert it to an unconstrained optimization.

Equality constraints

- Minimize $f(\mathbf{x})$ subject to: $a_i(\mathbf{x}) = 0$ for $i = 1, 2, \ldots, p$
- The gradient of $f(\mathbf{x})$ at a local minimizer is equal to the linear combination of the gradients of $a_i(\mathbf{x})$ with Lagrange multipliers as the coefficients.

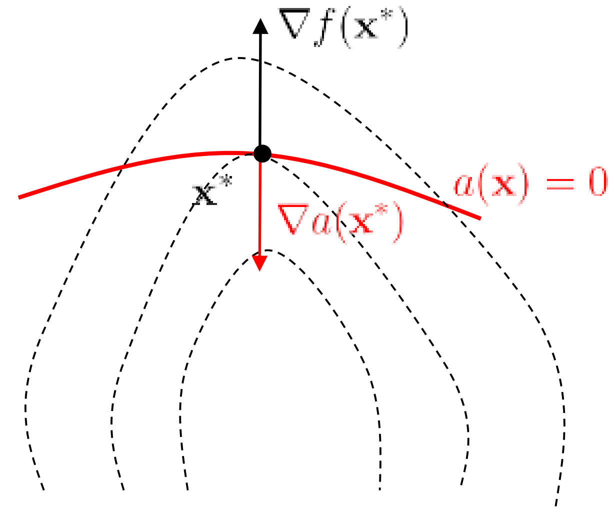
$$\nabla f(\mathbf{x}^*) = \sum_{i=1}^{p} \lambda_i^* \nabla a_i(\mathbf{x}^*)$$



$f_3 > f_2 > f_1$

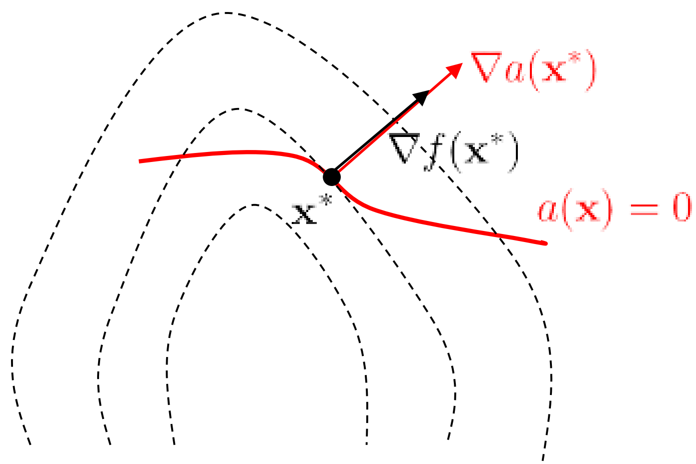
$\tilde{\mathbf{x}}$ is not a minimizer

x* is a minimizer, $\lambda^*>0$



$f_3 > f_2 > f_1$

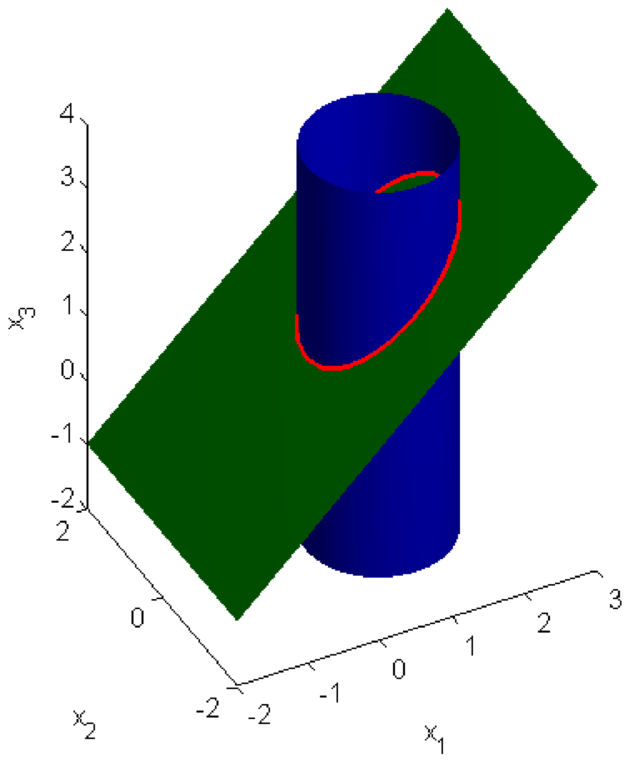
x* is a minimizer, $\lambda^*<0$



$f_3 > f_2 > f_1$

x* is not a minimizer

3D Example

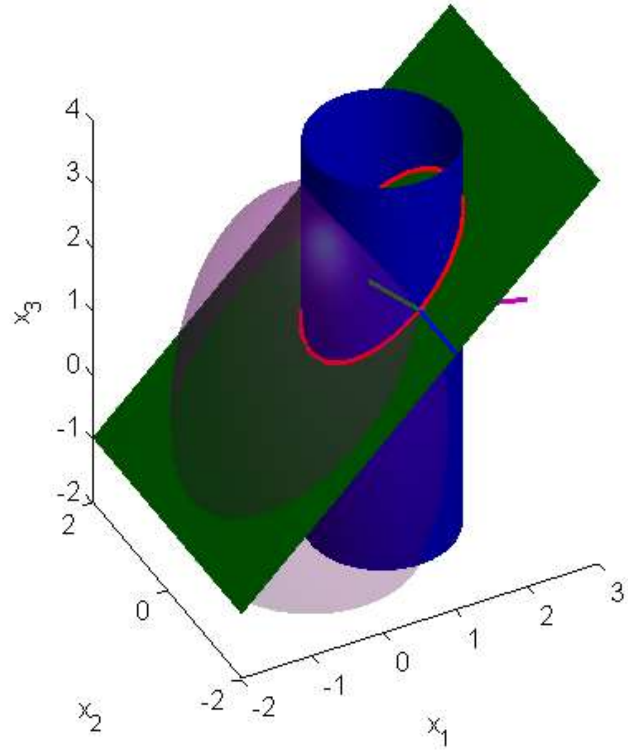


$$a_1(\mathbf{x}) = -x_1 + x_3 - 1 = 0$$

$$a_2(\mathbf{x}) = x_1^2 + x_2^2 - 2x_1 = 0$$

3D Example

$$f(\mathbf{x}) = x_1^2 + x_2^2 + \frac{1}{4}x_3^2$$

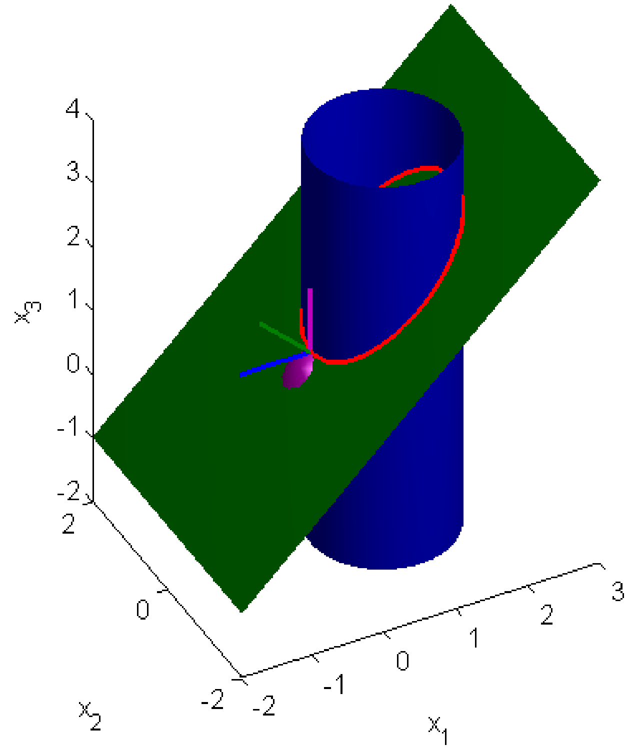


$$f(\mathbf{x}) = 3$$

Gradients of constraints and objective function are linearly independent.

3D Example

$$f(\mathbf{x}) = x_1^2 + x_2^2 + \frac{1}{4}x_3^2$$



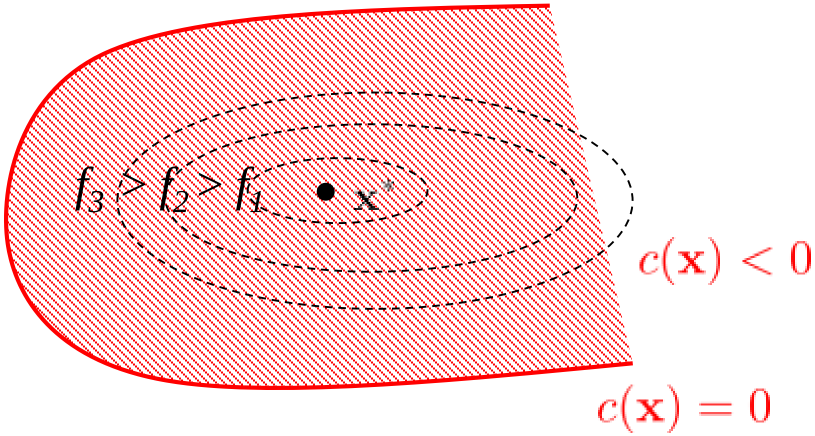
$$f(\mathbf{x}) = 1$$

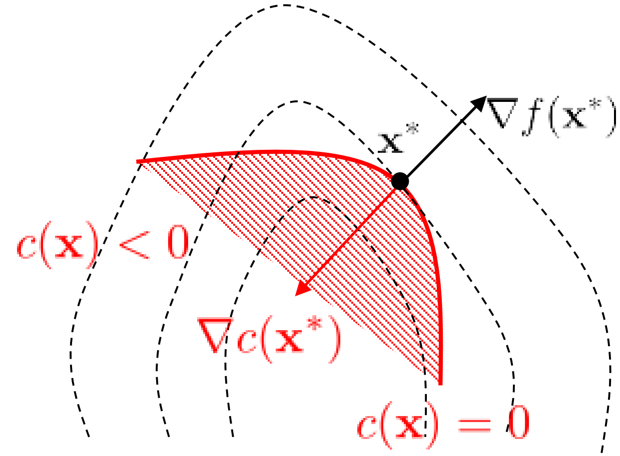
Gradients of constraints and objective function are linearly dependent.

Inequality constraints

- Minimize $f(\mathbf{x})$ subject to: $c_j(x) \leq 0$ for $j = 1, 2, \ldots, q$
- The gradient of $f(\mathbf{x})$ at a local minimizer is equal to the linear combination of the gradients of $c_j(\mathbf{x})$, which are active ($c_j(\mathbf{x}) = 0$)
- and Lagrange multipliers must be positive, $\mu_j \geq 0,\ j \in A$

$$\nabla f(\mathbf{x}^*) = -\sum_{j \in A} \mu_j^* \nabla c_j(\mathbf{x}^*)$$

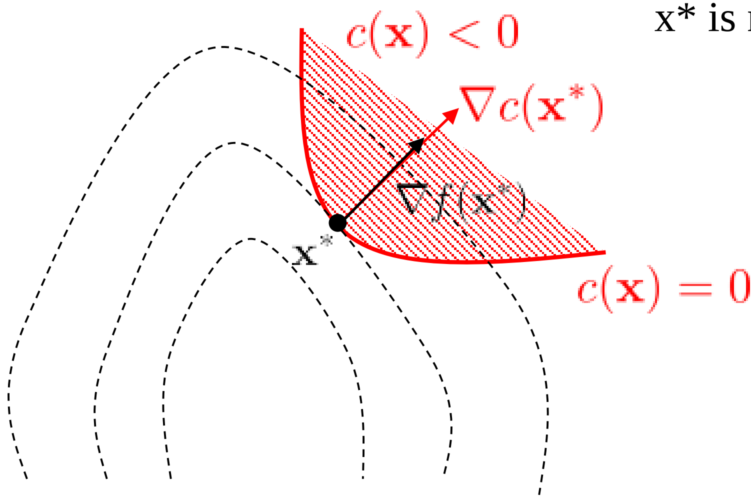




$f_3 > f_2 > f_1$

No active constraints at x*, $\nabla f(\mathbf{x}) = \mathbf{0}$

x* is not a minimizer, μ<0



$f_3 > f_2 > f_1$

x* is a minimizer, μ>0

Lagrangien

- We can introduce the function (Lagrangian)

$$L(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = f(\mathbf{x}) + \sum_{i=1}^{p} \lambda_i a_i(\mathbf{x}) + \sum_{j=1}^{q} \mu_j c_j(\mathbf{x})$$

- The necessary condition for the local minimizer is

$$\nabla L(x, \boldsymbol{\lambda}, \boldsymbol{\mu}) = 0 \iff \frac{\partial L}{\partial x} = 0, \quad \frac{\partial L}{\partial \boldsymbol{\lambda}} = 0, \quad \frac{\partial L}{\partial \boldsymbol{\mu}} = 0$$

and it must be a feasible point (i.e. constraints are satisfied).

- These are Karush-Kuhn-Tucker conditions

Dual Problem

Primal problem: minimize $f(x)$

subject to: $c(x) \leq 0$

Lagrangian: $L(x, \mu) = f(x) + \mu c(x)$

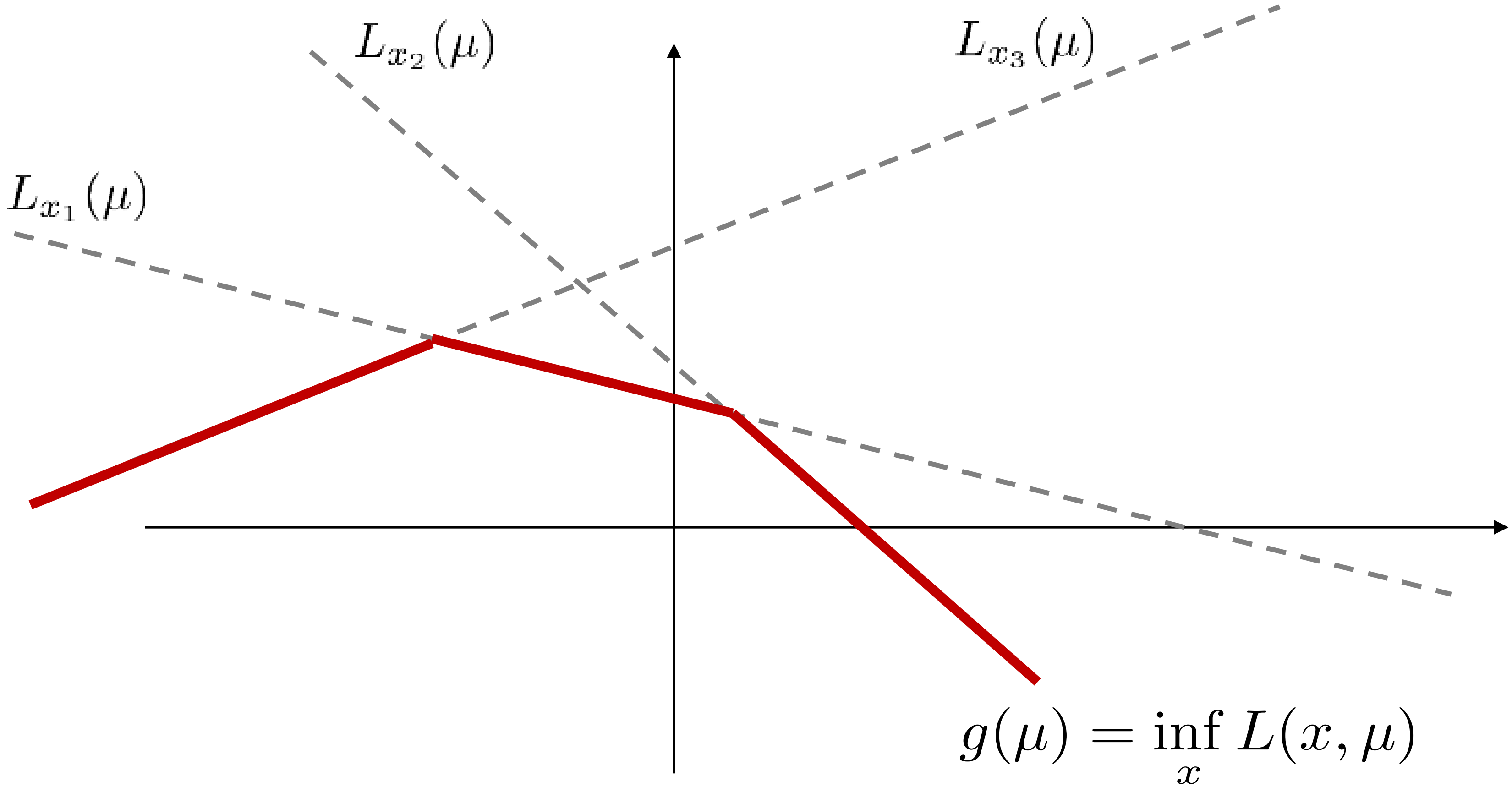
Dual function: $g(\mu) = \inf_x L(x, \mu)$ is always concave!

Dual problem: maximize $g(\mu)$

subject to: $\mu \geq 0$

If f and c convex ➔ sup g = inf f (almost always)

- Linear functions: $L_x(\mu) \equiv L(x,\mu) = f(x) + \mu c(x)$



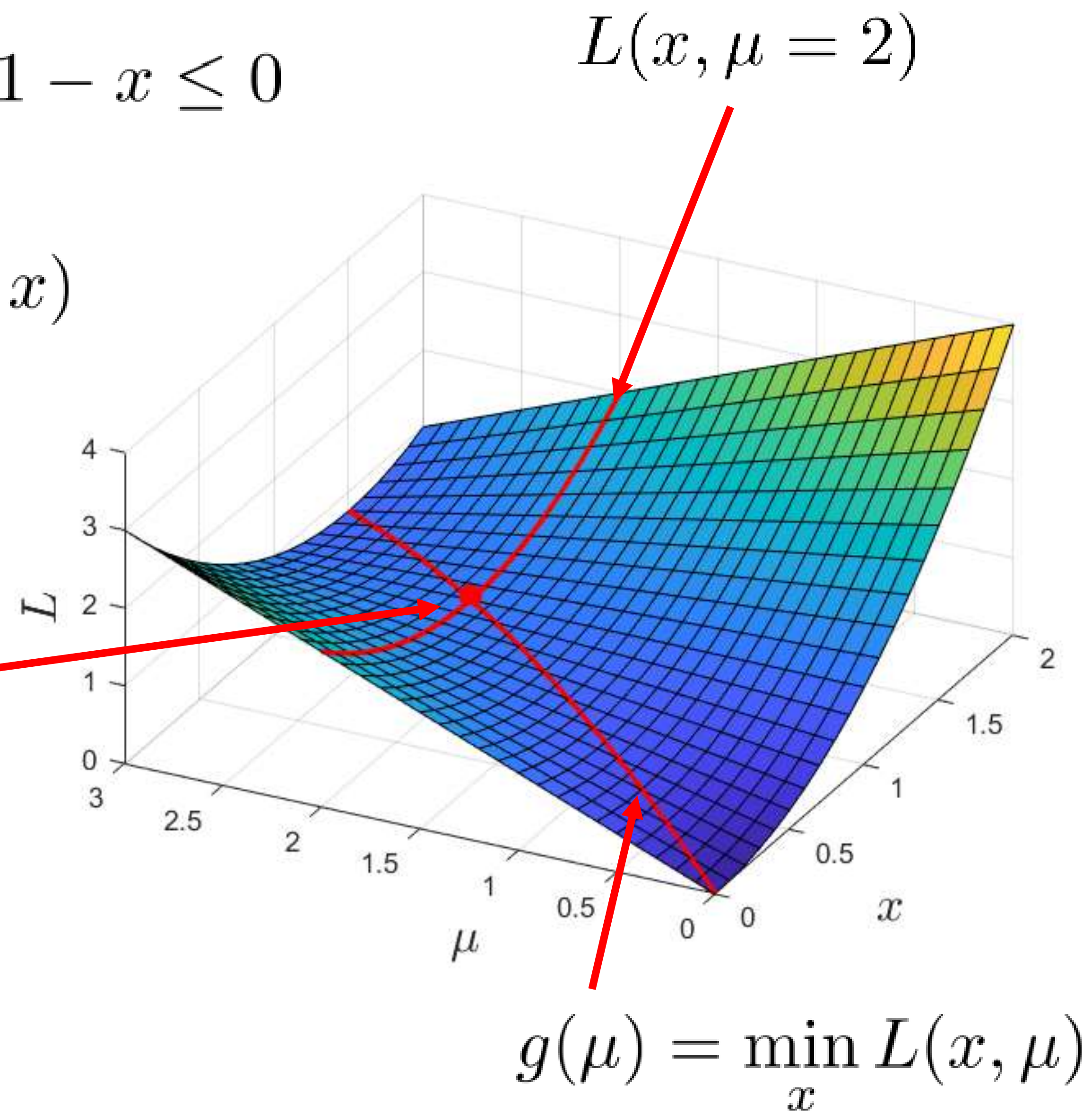
Toy Case

$$\min_x x^2 \quad \text{subject to} \quad 1 - x \leq 0$$

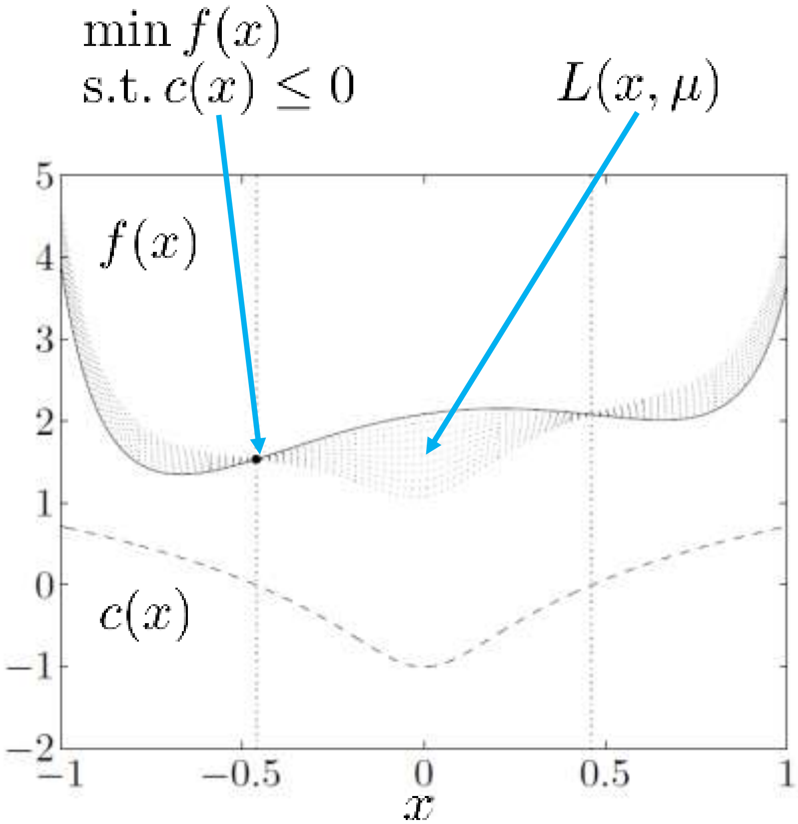
$$L(x, \mu) = x^2 + \mu(1 - x)$$

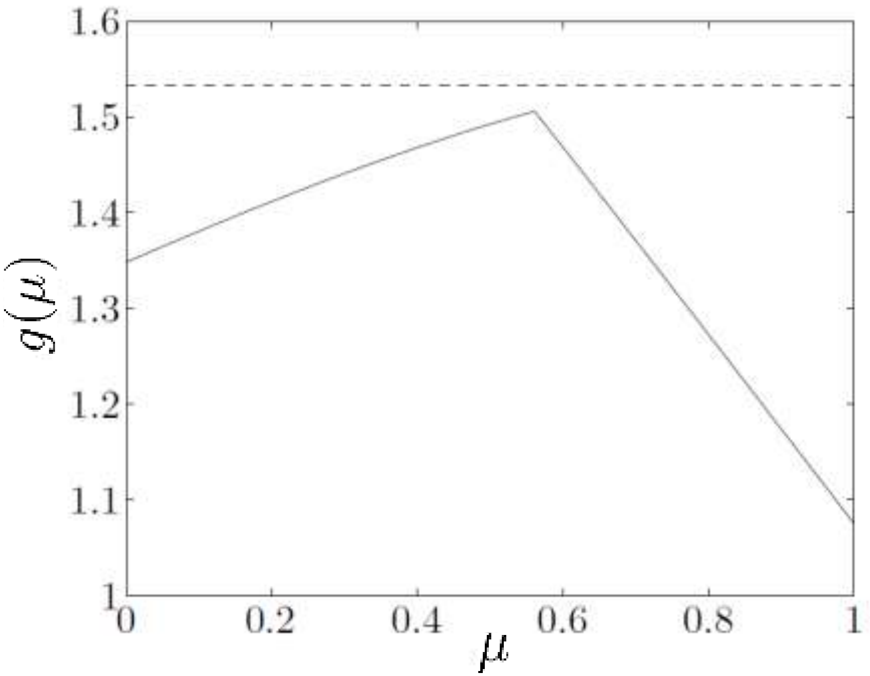
Solution is a saddle point

$$x = 1, \mu = 2$$



Dual Function





Proximal operator

- Problems of type:

$$\color{red}{a^*} = \arg\min_a \frac{1}{2}\|a - \color{blue}{b}\|_2^2 + \lambda\rho(a) = \mathbf{prox}_{\lambda\rho}(\color{blue}{b})$$

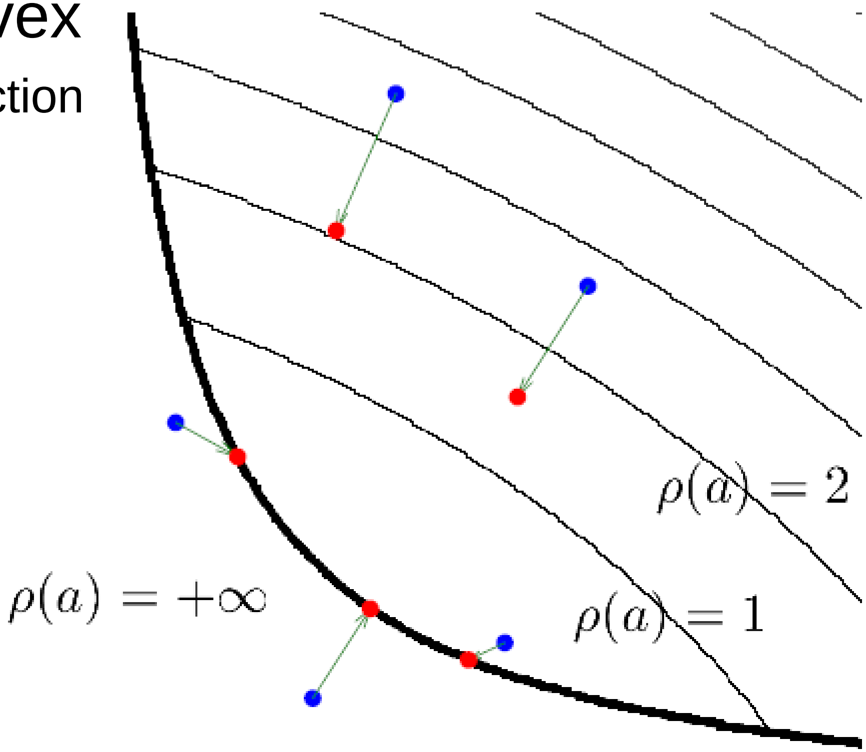
- If $\rho(a)$ closed proper convex

  => e.g. indicator function

$\mathbf{prox}_\rho(b)$ strictly convex

=>

unique minimizer



Examples of prox operators

- L1 norm ->

soft thresholding

$$\rho = \| \quad \|_1 \quad \rightarrow \quad \mathbf{prox}_{\| \|_1}(b) := S_\lambda(b)$$

- Indicator function of a convex set C ->

projection onto C

$$\rho = I_C \quad \rightarrow \quad \mathbf{prox}_{I_C}(b) := \Pi_C(b)$$

Alternating Direction Method of Multipliers

Gabay et al., 1976

- f, g convex but not necessary smooth

$$\min_{\mathbf{x}} f(\mathbf{x}) + g(\mathbf{A}\mathbf{x})$$

- e.g.: g is $L1$ norm or positivity constraint

Deconvolution with TV regularization

$$\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{H}\mathbf{x} - \mathbf{g}\|_2^2 + \lambda\|\mathbf{D}\mathbf{x}\|_1$$

Alternating Direction Method of Multipliers

Gabay et al., 1976

- f, g convex but not necessary smooth

$$\min_{\mathbf{x}} f(\mathbf{x}) + g(\mathbf{A}\mathbf{x})$$

- e.g.: g is $L1$ norm or positivity constraint
- variable splitting

$$\min_{\mathbf{x},\mathbf{z}} f(\mathbf{x}) + g(\mathbf{z}) \quad \text{s.t.} \quad \mathbf{A}\mathbf{x} - \mathbf{z} = 0$$

- Augmented Lagrangian:

$$L(\mathbf{x}, \mathbf{z}, \mathbf{y}) = f(\mathbf{x}) + g(\mathbf{z}) + \mathbf{y}^T(\mathbf{A}\mathbf{x} - \mathbf{z}) + (\rho/2)\|\mathbf{A}\mathbf{x} - \mathbf{z}\|_2^2$$

Alternating Direction Method of Multipliers

$$L(\mathbf{x}, \mathbf{z}, \mathbf{y}) = f(\mathbf{x}) + g(\mathbf{z}) + \mathbf{y}^T(\mathbf{A}\mathbf{x} - \mathbf{z}) + (\rho/2)\|\mathbf{A}\mathbf{x} - \mathbf{z}\|_2^2$$

- ADMM

$$\mathbf{x}^{k+1} := \arg\min_{\mathbf{x}} L(\mathbf{x}, \mathbf{z}^k, \mathbf{y}^k)$$ x minimization

$$\mathbf{z}^{k+1} := \arg\min_{\mathbf{z}} L(\mathbf{x}^{k+1}, \mathbf{z}, \mathbf{y}^k)$$ z minimization

$$\mathbf{y}^{k+1} := \mathbf{y}^k + \rho(\mathbf{A}\mathbf{x}^{k+1} - \mathbf{z}^{k+1})$$ dual update

ADMM with scaled dual variable

- combine linear and quadratic terms

$$L(\mathbf{x},\mathbf{z},\mathbf{y}) = f(\mathbf{x}) + g(\mathbf{z}) + \mathbf{y}^T(\mathbf{A}\mathbf{x} - \mathbf{z}) + (\rho/2)\|\mathbf{A}\mathbf{x} - \mathbf{z}\|_2^2$$
$$= f(\mathbf{x}) + g(\mathbf{z}) + (\rho/2)\|\mathbf{A}\mathbf{x} - \mathbf{z} + \mathbf{u}\|_2^2 + \mathrm{const.}$$

with

$$\mathbf{u} = (1/\rho)\mathbf{y}$$

- ADMM (scaled dual form):

$$\mathbf{x}^{k+1} := \arg\min_{\mathbf{x}} \left(f(\mathbf{x}) + (\rho/2)\|\mathbf{A}\mathbf{x} - \mathbf{z}^k + \mathbf{u}^k\|_2^2\right)$$
$$\mathbf{z}^{k+1} := \arg\min_{\mathbf{z}} \left(g(\mathbf{z}) + (\rho/2)\|\mathbf{A}\mathbf{x}^{k+1} - \mathbf{z} + \mathbf{u}^k\|_2^2\right)$$
$$\mathbf{u}^{k+1} := \mathbf{u}^k + (\mathbf{A}\mathbf{x}^{k+1} - \mathbf{z}^{k+1})$$

ADMM - example

- Deconvolution with TV regularization

$$\min_{\mathbf{x}} (1/2)\|\mathbf{Hx} - \mathbf{g}\|_2^2 + \lambda\|\mathbf{Dx}\|_1$$

- Augmented Lagrangian

$$L(\mathbf{x}, \mathbf{z}, \mathbf{v}) \propto (1/2)\|\mathbf{Hx} - \mathbf{g}\|_2^2 + \lambda\|\mathbf{z}\|_1 + (\rho/2)\|\mathbf{Dx} - \mathbf{z} + \mathbf{v}\|_2^2$$

- ADMM

1) $\mathbf{x} \leftarrow \arg\min_{\mathbf{x}} L(\mathbf{x}, \mathbf{z}, \mathbf{v})$ System of linear equations (CG):

$$\mathbf{x} \leftarrow (\mathbf{H}^T\mathbf{H} + \rho\mathbf{D}^T\mathbf{D})\mathbf{x} = \mathbf{H}^T\mathbf{g} + \rho\mathbf{D}^T(\mathbf{z} - \mathbf{v})$$

2) $\mathbf{z} \leftarrow \arg\min_{\mathbf{z}} L(\mathbf{x}, \mathbf{z}, \mathbf{v})$ Proximal operator (soft-thresholding)

$$\mathbf{z} \leftarrow S_{\lambda/\rho}(\mathbf{Dx} + \mathbf{v})$$

3) $\mathbf{v} \leftarrow \mathbf{v} + (\mathbf{Dx} - \mathbf{z})$

Quadratic Programming (QP)

- Like in the unconstrained case, it is important to study quadratic functions. Why?
- Because general nonlinear problems are solved as a sequence of minimizations of their quadratic approximations.
- QP with constraints

Minimize $$f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T\mathbf{H}\mathbf{x} + \mathbf{x}^T\mathbf{p}$$

subject to linear constraints.

- $\mathbf{H}$ is symmetric and positive semidefinite.

QP with Equality Constraints

- Minimize $f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T\mathbf{H}\mathbf{x} + \mathbf{x}^T\mathbf{p}$

  Subject to: $\mathbf{A}\mathbf{x} = \mathbf{b}$

- Ass.: $\mathbf{A}$ is $p \times N$ and has full row rank ($p<N$)
- Convert to unconstrained problem by variable elimination:

$$\mathbf{x} = \mathbf{Z}\phi + \mathbf{A}^+\mathbf{b}$$

$\mathbf{Z}$ is the null space of $\mathbf{A}$

$\mathbf{A}^+$ is the pseudo-inverse.

Minimize $\hat{f}(\phi) = \frac{1}{2}\phi^T\hat{\mathbf{H}}\phi + \phi^T\hat{\mathbf{p}}$

$$\hat{\mathbf{H}} = \mathbf{Z}^T\mathbf{H}\mathbf{Z}$$

$$\hat{\mathbf{p}} = \mathbf{Z}^T(\mathbf{H}\mathbf{A}^+\mathbf{b} + \mathbf{p})$$

This quadratic unconstrained problem can be solved, e.g., by Newton method.

QP with inequality constraints

- Minimize $f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T\mathbf{H}\mathbf{x} + \mathbf{x}^T\mathbf{p}$

  Subject to: $\mathbf{A}\mathbf{x} \geq \mathbf{b}$

- First we check if the unconstrained minimizer $\mathbf{x}^* = -\mathbf{H}^{-1}\mathbf{p}$ is feasible.

  If yes we are done.

  If not we know that the minimizer must be on the boundary and we proceed with an active-set method.

  - $\mathbf{x}_k$ is the current feasible point
  - $\mathcal{A}_k$ is the index set of active constraints at $\mathbf{x}_k$
  - Next iterate is given by $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{d}_k$

Active-set method

$$\mathbf{A}^T = [\mathbf{a}_1 \dots \mathbf{a}_p]$$

- $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{d}_k$ How to find $\mathbf{d}_k$?
  - To remain active $\mathbf{a}_j^T \mathbf{x}_{k+1} - b_j = 0$ thus $\mathbf{a}_j^T \mathbf{d}_k = 0 \quad j \in \mathcal{A}_k$
  - The objective function at $\mathbf{x}_k+\mathbf{d}$ becomes

$$f_k(\mathbf{d}) = \frac{1}{2}\mathbf{d}^T \mathbf{H} \mathbf{d} + \mathbf{d}^T \mathbf{g}_k + f(\mathbf{x}_k)$$

where $\mathbf{g}_k = \nabla f(\mathbf{x}_k)$

- The major step is a QP sub-problem

$$\mathbf{d}_k = \arg\min_{\mathbf{d}} \frac{1}{2}\mathbf{d}^T \mathbf{H} \mathbf{d} + \mathbf{d}^T \mathbf{g}_k$$

subject to: $\mathbf{a}_j^T \mathbf{d} = 0 \quad j \in \mathcal{A}_k$

- Two situations may occur: $\mathbf{d}_k = \mathbf{0}$ or $\mathbf{d}_k \neq \mathbf{0}$

Active-set method

- $\mathbf{d}_k = \mathbf{0}$

  We check if KKT conditions are satisfied

  $$\nabla_x L(\mathbf{x}, \boldsymbol{\mu}) = \mathbf{H}\mathbf{x}_k + \mathbf{p} - \sum_{j \in \mathcal{A}_k} \mu_j \mathbf{a}_j = \mathbf{0} \quad \text{and} \quad \mu_j \geq 0$$

  If YES we are done.

  If NO we remove the constraint from the active set $\mathcal{A}_k$ with the most negative μ_j and solve the QP sub-problem again but this time with less active constraints.

- $\mathbf{d}_k \neq \mathbf{0}$

  We can move to $\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{d}_k$ but some inactive constraints may be violated on the way.

  In this case, we move by $\alpha_k \mathbf{d}_k$ till the first inactive constraint becomes active, update $\mathcal{A}_k$, and solve the QP sub-problem again but this time with more active constraints.

General Nonlinear Optimization

- Minimize $f(\mathbf{x})$

  subject to: $a_i(\mathbf{x}) = 0$

  $c_j(\mathbf{x}) \geq 0$

  where the objective function and constraints are nonlinear.

1. For a given $\{\mathbf{x}_k, \boldsymbol{\lambda}_k, \boldsymbol{\mu}_k\}$ approximate Lagrangien by Taylor series → QP problem
2. Solve QP → descent direction $\{\boldsymbol{\delta}_x, \boldsymbol{\delta}_\lambda, \boldsymbol{\delta}_\mu\}$
3. Perform line search in the direction $\boldsymbol{\delta}_x \longrightarrow \mathbf{x}_{k+1}$
4. Update Lagrange multipliers → $\{\boldsymbol{\lambda}_{k+1}, \boldsymbol{\mu}_{k+1}\}$
5. Repeat from Step 1.

General Nonlinear Optimization

Lagrangien $$L(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = f(\mathbf{x}) - \sum_{i=1}^{p} \lambda_i a_i(\mathbf{x}) - \sum_{j=1}^{q} \mu_j c_j(\mathbf{x})$$

At the *k*th iterate: $\{\mathbf{x}_k, \boldsymbol{\lambda}_k, \boldsymbol{\mu}_k\}$
and we want to compute a set of increments: $\{\boldsymbol{\delta}_x, \boldsymbol{\delta}_\lambda, \boldsymbol{\delta}_\mu\}$

First order approximation of $\nabla_x L$ and constraints:

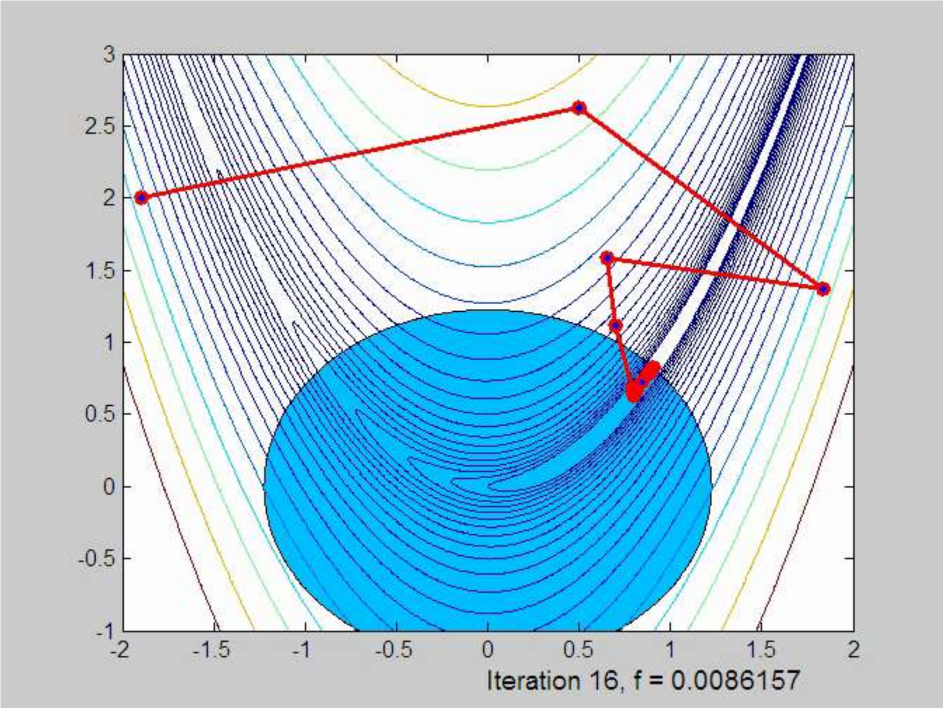
- $\nabla_x L(\mathbf{x}_{k+1}, \boldsymbol{\lambda}_{k+1}, \boldsymbol{\mu}_{k+1}) \approx \nabla_x L(\mathbf{x}_k, \boldsymbol{\lambda}_k, \boldsymbol{\mu}_k) +$
  $+\nabla_x^2 L(\mathbf{x}_k, \boldsymbol{\lambda}_k, \boldsymbol{\mu}_k)\boldsymbol{\delta}_x + \nabla_{x\lambda}^2 L(\mathbf{x}_k, \boldsymbol{\lambda}_k, \boldsymbol{\mu}_k)\boldsymbol{\delta}_\lambda + \nabla_{x\mu}^2 L(\mathbf{x}_k, \boldsymbol{\lambda}_k, \boldsymbol{\mu}_k)\boldsymbol{\delta}_\mu = \mathbf{0}$
- $c_i(\mathbf{x}_{k+1}) \approx c_i(\mathbf{x}_k) + \boldsymbol{\delta}_x^T \nabla_x c_i(\mathbf{x}_k) \geq 0$
- $a_i(\mathbf{x}_{k+1}) \approx a_i(\mathbf{x}_k) + \boldsymbol{\delta}_x^T \nabla_x a_i(\mathbf{x}_k) = 0$

These approximate KKT conditions corresponds to a QP program

SQP example

Minimize $f(x,y) = 100(y - x^2)^2 + (1 - x)^2$

subject to: $1.5 - x_1^2 - x_2^2 \geq 0$



Linear Programming (LP)

- LP is common in economy and is meaningful only if it is with constraints.
- Two forms:
  1. Minimize $f(\mathbf{x}) = \mathbf{c}^T\mathbf{x}$
     subject to: $\mathbf{A}\mathbf{x} = \mathbf{b}$, $\mathbf{x} \geq 0$

     $\mathbf{A}$ is $p \times N$ and has full row rank ($p<N$)
  2. Minimize $f(\mathbf{x}) = \mathbf{c}^T\mathbf{x}$
     subject to: $\mathbf{A}\mathbf{x} \geq \mathbf{b}$

- QP can solve LP.
- If the LP minimizer exists it must be one of the vertices of the feasible region.

  Prove it!
- A fast method that considers vertices is the Simplex method.