

Numerical Optimization

Unit 11: Penalty Method and Interior Point Method Filter Method and the Maratos Effect

Che-Rung Lee

Department of Computer Science
National Tsing Hua University

November 17, 2021

Penalty method

- The idea is to add penalty terms to the objective function, which turns a constrained optimization problem to an unconstrained one.

Quadratic penalty function

Example (For equality constraints)

$\min x_1 + x_2$ subject to $x_1^2 + x_2^2 - 2 = 0$ ($\vec{x}^* = (-1, -1)$)

\Rightarrow Define $Q(\vec{x}, \mu) = x_1 + x_2 + \frac{\mu}{2}(x_1^2 + x_2^2 - 2)^2$

For $\mu = 1$,

$$\nabla Q(\vec{x}, 1) = \begin{pmatrix} 1 + 2(x_1^2 + x_2^2 - 2)x_1 \\ 1 + 2(x_1^2 + x_2^2 - 2)x_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} \vec{x}_1^* \\ \vec{x}_2^* \end{pmatrix} = \begin{pmatrix} -1.1 \\ -1.1 \end{pmatrix}$$

For $\mu = 10$,

$$\nabla Q(\vec{x}, 10) = \begin{pmatrix} 1 + 20(x_1^2 + x_2^2 - 2)x_1 \\ 1 + 20(x_1^2 + x_2^2 - 2)x_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} \vec{x}_1^* \\ \vec{x}_2^* \end{pmatrix} = \begin{pmatrix} -1.0000001 \\ -1.0000001 \end{pmatrix}$$

Size of μ

- It seems the larger μ , the better solution is.
- When μ is large, matrix $\nabla^2 Q \approx \mu \nabla c \nabla c^T$ is ill-conditioned.

$$Q(\vec{x}, \mu) = f(\vec{x}) + \frac{\mu}{2}(c(\vec{x}))^2$$

$$\nabla Q = \nabla f + \mu c \nabla c$$

$$\nabla^2 Q = \nabla^2 f + \mu \nabla c \nabla c^T + \mu c \nabla^2 c$$

- μ cannot be too small either.

Example

$$\min_{\vec{x}} -5x_1^2 + x_2^2 \quad \text{s.t. } x_1 = 1.$$

$$Q(\vec{x}, \mu) = -5x_1^2 + x_2^2 + \frac{\mu}{2}(x_1 - 1)^2.$$

For $\mu < 10$, the problem $\min Q(\vec{x}, \mu)$ is unbounded.

Quadratic penalty function

- Picks a proper initial guess of μ and gradually increases it.

Algorithm: Quadratic penalty function

- 1 Given $\mu_0 > 0$ and \vec{x}_0
- 2 For $k = 0, 1, 2, \dots$
 - 1 Solve $\min_{\vec{x}} Q(:, \mu_k) = f(\vec{x}) + \frac{\mu_k}{2} \sum_{i \in \mathbb{E}} c_i^2(\vec{x})$.
 - 2 If converged, stop
 - 3 Increase $\mu_{k+1} > \mu_k$ and find a new \vec{x}_{k+1}

- Problem: the solution is not exact for $\mu \leq \infty$.

Augmented Lagrangian method

- Use the Lagrangian function to rescue the inexactness problem.
- Let

$$\mathcal{L}(\vec{x}, \vec{\rho}, \mu) = f(\vec{x}) - \sum_{i \in \varepsilon} \rho_i c_i(\vec{x}) + \frac{\mu}{2} \sum_{i \in \varepsilon} c_i^2(\vec{x})$$

$$\nabla \mathcal{L} = \nabla f(\vec{x}) - \sum_{i \in \varepsilon} \rho_i \nabla c_i(\vec{x}) + \mu \sum_{i \in \varepsilon} c_i(\vec{x}) \nabla c_i.$$

- By the Lagrangian theory, $\nabla \mathcal{L} = \nabla f - \underbrace{\sum_{i \in \varepsilon} (\rho_i - \mu c_i)}_{\lambda_i^*} \nabla c_i$.
- At the optimal solution, $c_i(\vec{x}^*) = \frac{-1}{\mu}(\lambda_i^* - \rho_i)$.
- If we can approximate $\rho_i \rightarrow \lambda_i^*$, μ_k need not be increased indefinitely,
- Algorithm: update ρ_i at each iteration.

$$\rho_i^{k+1} = \rho_i^k - \mu_k c_i(x_k)$$

Example

$$\min_{x_1, x_2} x_1 + x_2 \text{ subject to } x_1^2 + x_2^2 - 2 = 0$$

- The optimal solution is $x^* = (-1, -1)^T$ and $\lambda^* = -0.5$.
- The formulation of augmented Lagrangian

$$L_A(x, \mu) = x_1 + x_2 - \rho(x_1^2 + x_2^2 - 2) + \frac{\mu}{2}(x_1^2 + x_2^2 - 2)^2.$$

- The KKT conditions requires

$$\nabla_x L_A = \begin{bmatrix} 1 - 2\rho x_1 + 2\mu(x_1^2 + x_2^2 - 2)x_1 \\ 1 - 2\rho x_2 + 2\mu(x_1^2 + x_2^2 - 2)x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \quad (1)$$

- Let $\mu_0 = 1, \rho_0 = -0.4$. The solution of (1) is $\vec{x}_1 = (-1.02, -1.02)^T$.
- Update $\rho_1 = \rho_0 - \mu_0 c(\vec{x}_1) = -0.48$, which is closer to $\lambda^* = -0.5$.
- Use $\mu_1 = 1, \rho_1 = -0.48$ and repeat the procedure until convergence.

Example—continue

$$\nabla_{xx} L_A = \begin{bmatrix} 2\mu(3x_1^2 + x_2^2 - 2) - 2\rho & 4\mu x_1 x_2 \\ 4\mu x_1 x_2 & 2\mu(x_1^2 + 3x_2^2 - 2) - 2\rho \end{bmatrix}$$

- As it converges, $x_1^2 + x_2^2 \rightarrow 2$, the Hessian of L_A is not ill-conditioned.

$$\nabla_{xx} L_A = \begin{bmatrix} 4\mu x_1^2 - 2\rho & 4\mu x_1 x_2 \\ 4\mu x_1 x_2 & 4\mu x_2^2 - 2\rho \end{bmatrix}$$

- To make the solution exact, i.e. having the same solution as the original problem, what is the minimum μ^* ? (Theorem 17.5)
- Put $x^* = (-1, -1)^T$ and $\lambda^* = -0.5 (= \rho^*)$ to $\nabla_{xx} L_A$. The Hessian becomes $\nabla_{xx} L_A = \begin{bmatrix} 4\mu + 1 & 4\mu \\ 4\mu & 4\mu + 1 \end{bmatrix}$, whose eigenvalues are 1 and $8\mu + 1$.
- The minimum μ should be $\mu > -1/8$.

Inequality constraints

There are two approaches to handle inequality constraints.

- 1 Make the object function nonsmooth (non-differentiable at some points).
- 2 Add slack variable to turn the inequality constraints to equality constraints.

$$c_i \geq 0 \Rightarrow \begin{cases} c_i(\vec{x}) - s_i = 0 \\ s_i \geq 0 \end{cases}$$

But then we have bounded constraints for slack variables.

We will focus on the second approach here.

Inequality constraints

- Suppose the augmented Lagrangian method is used and all inequality constraints are converted to bounded constraints.
- For a fixed μ and $\vec{\lambda}$,

$$\begin{aligned} \min_{\vec{x}} \quad & \mathcal{L}(\vec{x}, \vec{\lambda}, \mu) = f(\vec{x}) - \sum_{i=1}^m \lambda_i c_i(\vec{x}) + \frac{\mu}{2} \sum_{i=1}^m c_i^2(\vec{x}) \\ \text{s.t.} \quad & \vec{\ell} \leq \vec{x} \leq \vec{u} \end{aligned}$$

- The first order necessary condition for \vec{x} to be a solution of the above problem is

$$\vec{x} = P(\vec{x} - \nabla_{\vec{x}} \mathcal{L}_A(\vec{x}, \vec{\lambda}, \mu), \vec{\ell}, \vec{u}),$$

where

$$P(\vec{g}, \vec{\ell}, \vec{u}) = \begin{cases} \ell_i, & \text{if } g_i \leq \ell_i; \\ g_i, & \text{if } g_i \in (\ell_i, u_i); \\ u_i, & \text{if } g_i \geq u_i. \end{cases} \quad \text{for all } i = 1, 2, \dots, n.$$

Nonlinear gradient projection method

Sequential quadratic programming + trust region method to solve

$$\min_{\vec{x}} f(\vec{x}) \quad \text{s.t.} \quad \vec{\ell} \leq \vec{x} \leq \vec{u}$$

Algorithm: Nonlinear gradient projection method

- 1 At each iteration, build a quadratic model

$$q(\vec{x}) = \frac{1}{2}(\vec{x} - \vec{x}_k)^T B_k (\vec{x} - \vec{x}_k) + \nabla f_k^T (\vec{x} - \vec{x}_k)$$

where B_k is SPD approximation of $\nabla^2 f(\vec{x}_k)$.

- 2 For some Δ_k , use the gradient projection method to solve

$$\begin{aligned} \min_{\vec{x}} \quad & q(\vec{x}) \\ \text{s.t.} \quad & \max(\vec{\ell}, \vec{x}_k - \Delta_k) \leq \vec{x} \leq \max(\vec{u}, \vec{x}_k + \Delta_k), \end{aligned}$$

- 3 Update Δ_k and repeat 1-3 until converge.

Interior point method

- Consider the problem

$$\begin{array}{ll}\min_{\vec{x}} & f(\vec{x}) \\ \text{s.t.} & C_E(\vec{x}) = 0 \\ & C_I(\vec{x}) - \vec{s} = 0 \\ & \vec{s} \geq 0\end{array}$$

where \vec{s} are slack variables.

- The interior point method starts a point inside the feasible region, and builds “walls” on the boundary of the feasible region.
- A barrier function goes to infinity when the input is close to zero.

$$\min_{\vec{x}, \vec{s}} f(\vec{x}) - \mu \sum_{i=1}^m \log(s_i) \quad \text{s.t.} \quad \begin{array}{l} C_E(\vec{x}) = 0 \\ C_I(\vec{x}) - \vec{s} = 0 \end{array} \quad (2)$$

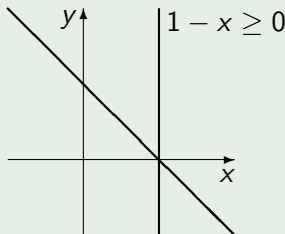
- The function $f(x) = -\log x \rightarrow \infty$ as $x \rightarrow 0$.
- μ : barrier parameter

An example

Example ($\min -x + 1$, s.t. $x \leq 1$)

$$\min_{\vec{x}} -x + 1 - \mu \ln(1 - x)$$

$\mu = 1,$	$x^* = 0.00005$
$\mu = 0.1,$	$x^* = 0.89999$
$\mu = 0.01,$	$x^* = 0.989999$
$\mu = 10^{-5},$	$x^* = 0.99993$



- The Lagrangian of (2) is

$$\mathcal{L}(\vec{x}, \vec{s}, \vec{y}, \vec{z}) = f(\vec{x}) - \mu \sum_{i=1}^m \log(s_i) - \vec{y}^T C_E(\vec{x}) - \vec{z}^T (C_I(\vec{x}) - \vec{s})$$

- 1 Vector \vec{y} is the Lagrangian multiplier of equality constraints.
- 2 Vector \vec{z} is the Lagrangian multiplier of inequality constraints.

The KKT conditions

The KKT conditions

The KKT conditions for (2)

$$\begin{aligned}\nabla_x \mathcal{L} = 0 &\Rightarrow \nabla f - A_E \vec{y} - A_I \vec{z} = 0 \\ \nabla_s \mathcal{L} = 0 &\Rightarrow S \vec{z} - \mu \vec{e} = 0 \\ \nabla_y \mathcal{L} = 0 &\Rightarrow C_E(\vec{x}) = 0 \\ \nabla_z \mathcal{L} = 0 &\Rightarrow C_I(\vec{x}) - \vec{s} = 0\end{aligned}\tag{3}$$

- Matrix $S = \text{diag}(\vec{s})$ and matrix $Z = \text{diag}(\vec{z})$.
- Matrix A_E is the Jacobian of C_E and matrix A_I is the Jacobian of C_I .

Newton's step

$$\text{Let } F = \begin{pmatrix} \nabla_x \mathcal{L} \\ \nabla_s \mathcal{L} \\ \nabla_y \mathcal{L} \\ \nabla_z \mathcal{L} \end{pmatrix} = \begin{pmatrix} \nabla f - A_E \vec{y} - A_I \vec{z} \\ S \vec{z} - \mu \vec{e} \\ C_E(\vec{x}) \\ C_I(\vec{x}) - \vec{s} \end{pmatrix}.$$

The interior point method uses Newton's method to solve $F = 0$.

$$\nabla F = \begin{bmatrix} \nabla_{xx} \mathcal{L} & 0 & -A_E(\vec{x}) & -A_I(\vec{x}) \\ 0 & Z & 0 & S \\ A_E(\vec{x}) & 0 & 0 & 0 \\ A_I(\vec{x}) & -I & 0 & 0 \end{bmatrix}$$

Newton's step

$$\nabla F \begin{pmatrix} \vec{p}_x \\ \vec{p}_s \\ \vec{p}_y \\ \vec{p}_z \end{pmatrix} = -F$$
$$\begin{aligned} \vec{x}_{k+1} &= \vec{x}_k + \alpha_x \vec{p}_x \\ \vec{s}_{k+1} &= \vec{s}_k + \alpha_s \vec{p}_s \\ \vec{y}_{k+1} &= \vec{y}_k + \alpha_y \vec{p}_y \\ \vec{z}_{k+1} &= \vec{z}_k + \alpha_z \vec{p}_z \end{aligned}$$

Algorithm: Interior point method (IPM)

Algorithm: Interior point method (IPM)

- ① Given initial $\vec{x}_0, \vec{s}_0, \vec{y}_0, \vec{z}_0$, and μ_0
- ② For $k = 0, 1, 2, \dots$ until converge
 - (a) Compute $\vec{p}_x, \vec{p}_s, \vec{p}_y, \vec{p}_z$ and $\alpha_x, \alpha_s, \alpha_y, \alpha_z$
 - (b) $(\vec{x}_{k+1}, \vec{s}_{k+1}, \vec{y}_{k+1}, \vec{z}_{k+1}) = (\vec{x}_k, \vec{s}_k, \vec{y}_k, \vec{z}_k) + (\alpha_x \vec{p}_x, \alpha_s \vec{p}_s, \alpha_y \vec{p}_y, \alpha_z \vec{p}_z)$
 - (c) Adjust $\mu_{k+1} < \mu_k$

Some comments of IPM

Some comments of the interior point method

- 1 The complementarity slackness condition says $s_i z_i = 0$ at the optimal solution, by which, the parameter μ , $SZ = \mu I$, needs to decrease to zero as the current solution approaches to the optimal solution.
- 2 Why cannot we set μ zero or small in the beginning? Because that will make \vec{x}_k going to the nearest constraint, and the entire process will move along constraint by constraint, which again becomes an exponential algorithm.
- 3 To keep \vec{x}_k (or \vec{s} and \vec{z}) away from any constraints, IPM also limits the step size of \vec{s} and \vec{z}

$$\alpha_{\vec{s}}^{\max} = \max\{\alpha \in (0, 1], \vec{s} + \alpha \vec{p}_s \geq (1 - \tau) \vec{s}\}$$
$$\alpha_{\vec{z}}^{\max} = \max\{\alpha \in (0, 1], \vec{z} + \alpha \vec{p}_z \geq (1 - \tau) \vec{z}\}$$

Interior point method for linear programming

- We will use linear programming to illustrate the details of IPM.

The primal	The dual
$\min_{\vec{x}} \quad \vec{c}^T \vec{x}$ s.t. $A\vec{x} = \vec{b},$ $\vec{x} \geq 0.$	$\max_{\vec{\lambda}} \quad \vec{b}^T \vec{\lambda}$ s.t. $A^T \vec{\lambda} + \vec{s} = \vec{c},$ $\vec{s} \geq 0.$

- KKT conditions

$$A^T \vec{\lambda} + \vec{s} = \vec{c}$$

$$A\vec{x} = \vec{b}$$

$$x_i s_i = 0$$

$$\vec{x} \geq 0, \vec{s} \geq 0$$

Solve problem

$$\text{Let } X = \begin{pmatrix} x_1 & & & \\ & x_2 & & \\ & & \ddots & \\ & & & x_n \end{pmatrix}, S = \begin{pmatrix} s_1 & & & \\ & s_2 & & \\ & & \ddots & \\ & & & s_n \end{pmatrix},$$
$$F = \begin{bmatrix} A^T \vec{\lambda} + \vec{s} - \vec{c} \\ A\vec{x} - \vec{b} \\ X\vec{s} - \mu\vec{e} \end{bmatrix}$$

- The problem is to solve $F = 0$

Newton's method

Using Newton's method

$$\nabla F = \begin{bmatrix} 0 & A^T & I \\ A & 0 & 0 \\ S & 0 & X \end{bmatrix}$$

$$\nabla F \begin{bmatrix} \vec{p}_x \\ \vec{p}_\lambda \\ \vec{p}_s \end{bmatrix} = -F \quad \begin{aligned} \vec{x}_{k+1} &= \vec{x}_k + \alpha_x \vec{p}_x \\ \vec{\lambda}_{k+1} &= \vec{\lambda}_k + \alpha_\lambda \vec{p}_\lambda \\ \vec{z}_{k+1} &= \vec{z}_k + \alpha_s \vec{p}_s \end{aligned}$$

- How to decide μ_k ?
 - $\mu_k = \frac{1}{n} \vec{x}_k \cdot \vec{s}_k$ is called duality measure.

The central path

The central path

- The central path: a set of points, $p(\tau) = \begin{pmatrix} x_\tau \\ \lambda_\tau \\ z_\tau \end{pmatrix}$, defined by the solution of the equation

$$A^T \vec{\lambda} + \vec{s} = \vec{c}$$

$$A\vec{x} = \vec{b}$$

$$x_i s_i = \tau \quad i = 1, 2, \dots, n$$

$$\vec{x}, \vec{s} > 0$$

Algorithm: The interior point method for solving linear programming

- ① Given an interior point \vec{x}_0 and the initial guess of slack variables \vec{s}_0
- ② For $k = 0, 1, \dots$

(a) Solve
$$\begin{pmatrix} 0 & A^T & I \\ A & 0 & 0 \\ S^k & 0 & X^k \end{pmatrix} \begin{pmatrix} \Delta x_k \\ \Delta \lambda_k \\ \Delta s_k \end{pmatrix} = \begin{pmatrix} \vec{b} - A\vec{x}_k \\ \vec{c} - \vec{s}_k - A^T \vec{\lambda}_k \\ -X^k S^k \vec{e} + \sigma_k \mu_k \vec{e} \end{pmatrix} \text{ for } \sigma_k \in [0, 1].$$

(b) Compute $(\alpha_x, \alpha_\lambda, \alpha_s)$ s.t.
$$\begin{pmatrix} \vec{x}_{k+1} \\ \vec{\lambda}_{k+1} \\ \vec{s}_{k+1} \end{pmatrix} = \begin{pmatrix} \vec{x}_k + \alpha_x \Delta x_k \\ \vec{\lambda}_k + \alpha_\lambda \Delta \lambda_k \\ \vec{s}_k + \alpha_s \Delta s_k \end{pmatrix}$$
 is in the neighborhood of the central path

$$\mathcal{N}(\theta) = \left\{ \begin{pmatrix} \vec{x} \\ \vec{\lambda} \\ \vec{s} \end{pmatrix} \in \mathcal{F} \mid \|XS\vec{e} - \mu\vec{e}\| \leq \theta\mu \right\} \text{ for some } \theta \in (0, 1].$$

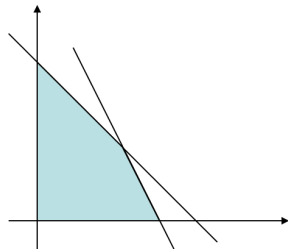
Example

The primal problem is

$$\begin{aligned} \min_x \quad & -x_1 + x_2 \\ \text{s.t.} \quad & x_1 + x_2 + x_3 = 40 \\ & 2x_1 + x_2 + x_4 = 60 \\ & x_1, x_2, x_3, x_4 \geq 0 \end{aligned}$$

The dual problem is

$$\begin{aligned} \max_y \quad & 40y_1 + 60y_2 \\ \text{s.t.} \quad & y_1 + 2y_2 + s_1 = -1 \\ & y_1 + y_2 + s_2 = 1 \\ & y_1 + s_3 = 0 \\ & y_2 + s_4 = 0 \\ & s_1, s_2, s_3, s_4 \geq 0 \end{aligned}$$



$$A = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 2 & 1 & 0 & 1 \end{bmatrix}, c = \begin{bmatrix} -1 \\ 1 \\ 0 \\ 0 \end{bmatrix}, x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}, s = \begin{bmatrix} s_1 \\ s_2 \\ s_3 \\ s_4 \end{bmatrix}, b = \begin{bmatrix} 40 \\ 60 \end{bmatrix}$$

Example—solving by Newton's method directly

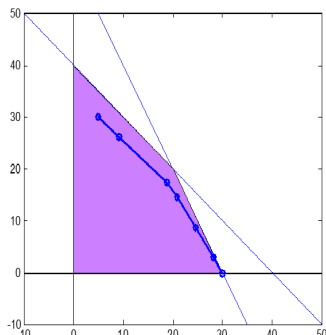
$$F = \begin{bmatrix} r_c \\ r_b \\ r_s \end{bmatrix} = \begin{bmatrix} A^T y + s - c \\ Ax - b \\ Xs \end{bmatrix}, J = \begin{bmatrix} \nabla_x r_c & \nabla_y r_c & \nabla_s r_c \\ \nabla_x r_b & \nabla_y r_b & \nabla_s r_b \\ \nabla_x r_s & \nabla_y r_s & \nabla_s r_s \end{bmatrix} = \begin{bmatrix} O & A^T & I \\ A & O & O \\ S & O & X \end{bmatrix}$$

Algorithm:

- 1 Given an initial guess of x_0, y_0 and s_0 ,
- 2 Solve $Jp = -F$, where $p = [p_x^T, p_y^T, p_s^T]^T$.
- 3 Update $x_{k+1} = x_k + \alpha_k p_x$, $y_{k+1} = y_k + \alpha_k p_y$, $s_{k+1} = s_k + \alpha_k p_s$.
- 4 If not converged, go to step 2.

Different initial points could have varied results.

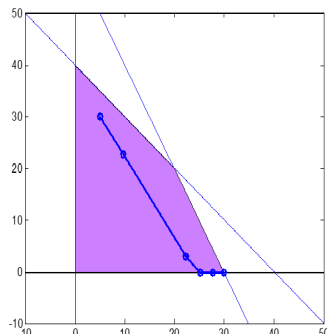
Example of Newton's method



$$x_0 = [5, 30, 5, 10]^T$$

$$s_0 = [.1, 2, .1, .1]^T$$

$$y = [-0.96, -0.06]^T$$



$$x_0 = [5, 30, 5, 10]^T$$

$$s_0 = [.1, .1, .1, .1]^T$$

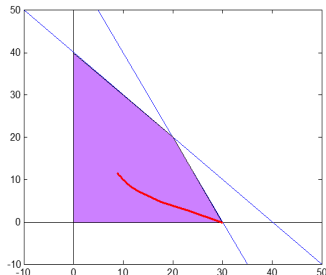
$$y = [0.266, -0.366]^T$$

The central path and path following algorithm

The central path is the trajectory that solves

$$F_c = \begin{bmatrix} A^T y + s - c \\ Ax - b \\ Xs - \tau e \end{bmatrix} = 0$$

for $\tau = [0, 300]$.



Path following algorithm:

- 1 Compute the search direction

$$Jp = - \begin{bmatrix} A^T y + s - c \\ Ax - b \\ Xs - \sigma_k \mu e \end{bmatrix}$$

- 2 Compute the step length so the steps are in the neighborhood of the central path.

- 3 Update $\mu = \frac{1}{n} \sum_{i=1}^n x_i s_i = \frac{x^T s}{n}$

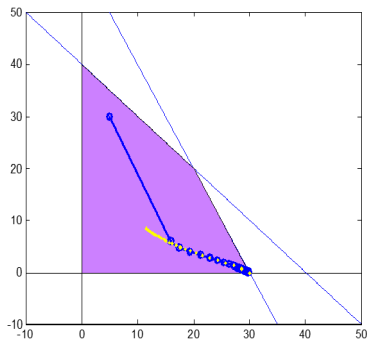
- 4 Centering parameter σ

Examples of path following algorithms

Use the same initial $x_0 = [5, 30, 5, 10]^T$, $s_0 = [1, 2, 1, .25]^T$

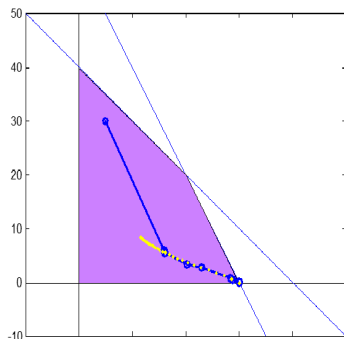
Short-step path following

- No line search
- Fixed σ .
- Using 2-norm



Long-step path following

- Line search
- Varied σ .
- Using L_∞ -norm



Filter method

- There are two goals of constrained optimization:
 - 1 Minimize the objective function.
 - 2 Satisfy the constraints.

Example

Suppose the problem is

$$\begin{array}{ll}\min_{\vec{x}} & f(\vec{x}) \\ \text{s.t.} & c_i(\vec{x}) = 0 \quad \text{for } i \in \mathcal{E} \\ & c_i(\vec{x}) \geq 0 \quad \text{for } i \in \mathcal{I}\end{array}$$

- Define $h(\vec{x})$ penalty functions of constraints.

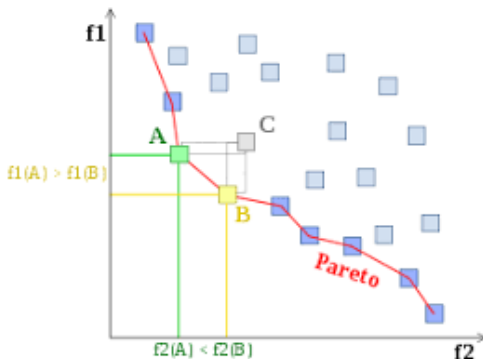
$$h(\vec{x}) = \sum_{i \in \mathcal{E}} |c_i(\vec{x})| + \sum_{i \in \mathcal{I}} [c_i(\vec{x})]^{-},$$

in which the notation $[z]^{-} = \max\{0, -z\}$.

- The goals become $\begin{cases} \min f(\vec{x}) \\ \min h(\vec{x}) \end{cases}$

Pareto set

- A pair (f_k, h_k) dominates (f_l, h_l) if $f_k < f_l$ and $h_k < h_l$.
- A filter is a list of pairs (f_k, h_k) such that no pair dominates any other.
- This is also called the Pareto set in multi-objective optimization.



The filter method

- The filter method only accepts the steps that are not dominated by other pairs.

Algorithm: The filter method

- ➊ Given initial \vec{x}_0 and an initial trust region Δ_0 .
- ➋ For $k = 0, 1, 2, \dots$ until converge
 - ➊ Compute a trial \vec{x}^+ by solving a local quadric programming model
 - ➋ If (f^+, h^+) is accepted to the filter
 - Set $\vec{x}_{k+1} = \vec{x}^+$, add (f^+, h^+) to the filter, and remove pairs dominated by it.
 - Else
 - Set $\vec{x}_{k+1} = \vec{x}_k$ and decrease Δ_k .

The Maratos effect

- The Maratos effect shows the filter method could reject a good step.

Example

$$\begin{aligned} \min_{x_1, x_2} f(x_1, x_2) &= 2(x_1^2 + x_2^2 - 1) - x_1 \\ \text{s.t. } x_1^2 + x_2^2 - 1 &= 0 \end{aligned}$$

- The optimal solution is $\vec{x}^* = (1, 0)$
- Suppose $\vec{x}_k = \begin{pmatrix} \cos \theta \\ \sin \theta \end{pmatrix}$, $\vec{p}_k = \begin{pmatrix} \sin^2 \theta \\ -\sin \theta \cos \theta \end{pmatrix}$

$$\vec{x}_{k+1} = \vec{x}_k + \vec{p}_k = \begin{pmatrix} \cos \theta + \sin^2 \theta \\ \sin \theta (1 - \cos \theta) \end{pmatrix}$$

Reject a good step

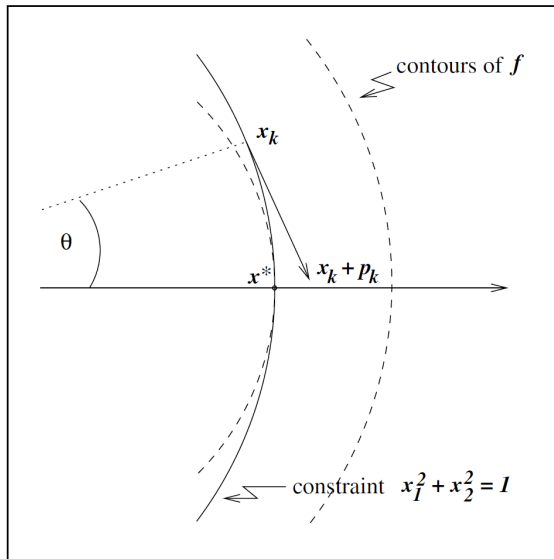
- $\|\vec{x}_k - \vec{x}^*\| = \left\| \begin{pmatrix} \cos \theta - 1 \\ \sin \theta \end{pmatrix} \right\|$
 $= \sqrt{\cos^2 \theta - 2 \cos \theta + 1 + \sin^2 \theta} = \sqrt{2(1 - \cos \theta)}$
- $\|\vec{x}_{k+1} - \vec{x}^*\|$
 $= \left\| \begin{pmatrix} \cos \theta + \sin^2 \theta - 1 \\ \sin \theta - \sin \theta \cos \theta \end{pmatrix} \right\| = \left\| \begin{pmatrix} \cos \theta(1 - \cos \theta) \\ \sin \theta(1 - \cos \theta) \end{pmatrix} \right\|$
 $= \sqrt{\cos^2 \theta(1 - \cos \theta)^2 + \sin^2 \theta(1 - \cos \theta)^2} = \sqrt{(1 - \cos \theta)^2}$
- Therefore $\frac{\|\vec{x}_{k+1} - \vec{x}^*\|}{\|\vec{x}_k - \vec{x}^*\|^2} = \frac{1}{2}$. This step gives a quadratic convergence.
- However, the filter method will reject this step because

$$f(\vec{x}_k) = -\cos \theta, \text{ and } c(\vec{x}_k) = 0,$$

$$f(\vec{x}_{k+1}) = -\cos \theta - \sin 2\theta = \sin^2 \theta - \cos \theta > f(\vec{x}_k)$$

$$c(\vec{x}_{k+1}) = \sin^2 \theta > 0 = c(\vec{x}_k)$$

Example of Maratos effect.



Maratos Effect:
Note that the constraint is no longer satisfied after the step from x_k to $x_k + p_k$, and the objective value has increased. (From textbook Figure 15.8).

The second order correction

The second order correction could help to solve this problem.

- Instead of $\nabla c(\vec{x}_k)^T \vec{p}_k + c(\vec{x}_k) = 0$, use quadratic approximation

$$c(\vec{x}_k) + \nabla c(\vec{x}_k)^T \vec{d}_k + \frac{1}{2} \vec{d}_k^T \nabla_{xx}^2 c(\vec{x}) \vec{d}_k = 0. \quad (4)$$

- Suppose $\|\vec{d}_k - \vec{p}_k\|$ is small. Use Taylor expansion to approximate quadratic term

$$c(\vec{x}_k + \vec{p}_k) \approx c(\vec{x}_k) + \nabla c(\vec{x}_k)^T \vec{p}_k + \frac{1}{2} \vec{p}_k^T \nabla_{xx}^2 c(\vec{x}) \vec{p}_k.$$

$$\frac{1}{2} \vec{d}_k^T \nabla_{xx}^2 c(\vec{x}) \vec{d}_k \approx \frac{1}{2} \vec{p}_k^T \nabla_{xx}^2 c(\vec{x}) \vec{p}_k \approx c(\vec{x}_k + \vec{p}_k) - c(\vec{x}_k) - \nabla c(\vec{x}_k)^T \vec{p}_k.$$

- Equation (4) can be rewritten as

$$\nabla c(\vec{x}_k)^T \vec{d}_k + c(\vec{x}_k + \vec{p}_k) - \nabla c(\vec{x}_k)^T \vec{p}_k = 0$$

- Use the corrected linearized constraint:

$$\nabla c(\vec{x}_k)^T \vec{p} + c(\vec{x}_k + \vec{p}_k) - \nabla c(\vec{x}_k)^T \vec{p}_k = 0.$$

(The original linearized constraint is $\nabla c(\vec{x}_k)^T \vec{p} + c(\vec{x}_k) = 0$.)