THE UNIVERSITY OF CHICAGO

S310 NUM OPT

Sec 15: Fundamentals of Constrained Optimization

15.1 TYPES OF CONSTRAINED OPTIMIZATION ALGORITHMS
Quadratic Programming Problems

- Algorithms for such problems are interested to explore because
  - 1. Their structure can be efficiently exploited.
  - 2. They form the basis for other algorithms, such as augmented Lagrangian and Sequential quadratic programming problems.

$$\begin{aligned} \min_x \quad & q(x) = \tfrac{1}{2} x^T G x + x^T c \\ \text{subject to} \quad & a_i^T x = b_i, \qquad i \in \mathcal{E}, \\ & a_i^T x \geq b_i, \qquad i \in \mathcal{I}, \end{aligned}$$

Penalty Methods

Solve with unconstrained optimization

- Idea: Replace the constraints by a penalty term.
- Inexact penalties: parameter driven to infinity to recover solution. Example:

$$x^* = f(x) \text{ subject to } c(x) = 0 \Leftrightarrow$$

$$x^\mu = f(x) + \frac{\mu}{2} \sum_{i \in \mathcal{E}} c_i^2(x), \quad x^* = \lim_{\mu \to \infty} x^\mu$$

- Exact but nonsmooth penalty – the penalty parameter can stay finite.

$$x^* = f(x) \text{ subject to } c(x) = 0 \Leftrightarrow f(x) + \mu \sum_{i \in \mathcal{E}} |c_i(x)|, \quad \mu \geq \mu_0$$

- But how do we produce reduction?

Augmented Lagrangian Methods

- Mix the Lagrangian point of view with a penalty point of view.

$$x^* = f(x) \text{ subject to } c(x) = 0 \Leftrightarrow$$

$$x^{\mu,\lambda} = f(x) - \sum_{i \in \mathcal{E}} \lambda_i c_i(x) + \frac{\mu}{2} \sum_{i \in \mathcal{E}} c_i^2(x),$$

$$x^* = \lim_{\lambda \to \lambda^*} x^{\mu,\lambda}, \ \mu \geq \mu_0 > 0$$

Sequential Quadratic Programming Algorithms

- Solve successively Quadratic Programs.

$$\begin{aligned} \min_p \quad & \frac{1}{2} p^T B_k p + \nabla f(x_k) \\ \text{subject to} \quad & \nabla c_i(x_k) d + c_i(x_k) = 0 \quad i \in \mathcal{E} \\ & \nabla c_i(x_k) d + c_i(x_k) \geq 0 \quad i \in \mathcal{I} \end{aligned}$$

- It is the analogous of Newton's method for the case of constraints if

$$B_k = \nabla^2_{xx} \mathcal{L}(x_k, \lambda_k)$$

- But how do you solve the subproblem? It is possible with extensions of simplex.
- An option is BFGS which makes it convex.

Interior Point Methods

- Reduce the inequality constraints with a barrier

$$\min_{x,s} \quad f(x) - \mu \sum_{i=1}^{m} \log s_i$$

$$\text{subject to} \quad c_i(x) = 0, \quad i \in \mathcal{E}$$

$$c_i(x) - s_i = 0, \quad i \in \mathcal{I}$$

- An alternative, is use to use a penalty as well:

$$\min_{x,s} f(x) - \mu \sum_{i=1}^{m} \log s_i + \frac{1}{2\mu} \sum_{i \in \mathcal{I}} (c_i(x) - s_i)^2 + \frac{1}{2\mu} \sum_{i \in \mathcal{E}} c_i((x))^2$$

- And I can solve it as a sequence of unconstrained problems!
- Also, note, you need not start with a feasible point.

Observation: Combinatorial Difficulty of Inequality Constraints..

- A natural question is: can I not easily reduce problems with inequality constraints to problem with equality constraints?
- I could try to use quadratic slacks – not a terrible idea, though I lose structure.
- What if I aim to preserve convexity for example? Not without a price.
- KKT conditions suggest that there is one equality constrained problem equivalent to the one I am trying to solve, but I do not know which one ….

First-Order Optimality Condition Theorem

- Optimality Conditions.

$$
\begin{aligned}
\nabla_x \mathcal{L}(x^*, \lambda^*) &= 0, \\
c_i(x^*) &= 0, \quad \text{for all } i \in \mathcal{E}, \\
c_i(x^*) &\geq 0, \quad \text{for all } i \in \mathcal{I}, \\
\lambda_i^* &\geq 0, \quad \text{for all } i \in \mathcal{I}, \\
\lambda_i^* c_i(x^*) &= 0, \quad \text{for all } i \in \mathcal{E} \cup \mathcal{I}.
\end{aligned}
$$

- Note the either-or nature of inequality/complementarity constraints – the origin of taking a long time to believe that linear program may have an exponential complexity …
- Though it was later proved that the problem can be solved with polynomial Complexity.

15.2 EXPLICIT USE OF FEASIBLE SET.

Problems with linear equality constraints

- If a problem has equality constaints, why not use IFT and eliminate the variables?
- If the constraints are nonlinear, this may turn to be more expensive then the optimization problem itself, as explicit, algebraic, stable elimination is unlikely.
- One could try to do some way of recomputation at every step, but that may end up being expensive.
- But if the constraints, are linear, it may make sense (and this will be used for QPs with EC==EQP)
- Consider:

$$\min \ f(x) \quad \text{subject to } Ax = b,$$

Reduction Strategies for Linear Constraints

- Idea: Use a special form of the Implicit Function Theorem

$$Y \in \mathbf{R}^{n\times m}; \qquad Z \in \mathbf{R}^{n\times(n-m)} \qquad [Y \mid Z] \in \mathbf{R}^{n\times n} \text{ is nonsingular}, \quad AZ = 0.$$

- In turn, this implies that $A[Y \mid Z] = [AY \mid 0]$ and thus AY is full rank and invertible.
- We can thus parametrize the feasible set along the components in the range of Y and Z.

$$x = Yx_{\mathrm{Y}} + Zx_{\mathrm{Z}},$$

Using the Y,Z parameterization

- This allows easy identification of the Y component of the feasible set:

$$x = Yx_Y + Zx_Z, \quad \Rightarrow \quad x_Y = (AY)^{-1}b. \quad \Rightarrow \quad x = Y(AY)^{-1}b + Zx_Z$$

- The reduced optimization problem.

$$\min_{x_Z} \; f(Y(AY)^{-1}b + Zx_Z).$$

How do we obtain a "good" YZ parameterization?

- Idea: use a version of the QR factorization

$$Ax = b \Rightarrow A^T \Pi = \begin{bmatrix} \underbrace{Q_1}_{n\times m} & \underbrace{Q_2}_{n\times(n-m)} \end{bmatrix} \begin{bmatrix} \overbrace{R}^{m\times m} \\ 0 \end{bmatrix}$$

- After which, define

$$A^T \overbrace{\Pi}^{m\times m} = \begin{bmatrix} \underbrace{Q_1}_{n\times m} & \underbrace{Q_2}_{n\times(n-m)} \end{bmatrix} \begin{bmatrix} \overbrace{R}^{m\times m} \\ 0 \end{bmatrix} \Rightarrow Y = Q_1, Z = Q_2; Q_1^T Q_2 = 0_{m\times(n-m)} \Rightarrow$$

$$\left[AZ\right]^T \Pi = Q_2^T A^T \Pi = \begin{bmatrix} Q_1 & \underbrace{Q_2}_{} \end{bmatrix} \begin{bmatrix} \overbrace{R}^{m\times m} \\ 0 \end{bmatrix} = 0_{(n-m)\times(m)}$$

$$\left[AY\right]^T \Pi = Q_1^T A^T \Pi = R \Rightarrow AY = \left[R\Pi^T\right]^T = \Pi R^T$$

Inequality Constraints

- Unfortunately, when inequality contraints are present, while theoretically possible, the elimination is not practical.
- That is since, many times it destroys the structure, for example by making bound constraints harder …

15.3 MERIT FUNCTIONS AND FILTERS

Feasible algorithms

- Merit function: a function that allows me to measure progress to optimality weighing in both feasibility and optimality
- If I can afford to maintain feasibility at all steps, then I just monitor decrease in objective function.
- I accept a point if I have enough descent.
- But this works only for very particular constraints, such as linear constraints or bound constraints (and we will use it).
- Algorithms that do that are called feasible algorithms.

Infeasible algorithms

- But, sometimes it is VERY HARD to enforce feasibility at all steps (e.g. nonlinear equality constraints).
- And I need feasibility only in the limit; so there is benefit to allow algorithms to move on the outside of the feasible set.
- But then, how do I measure progress since I have two, apparently contradictory requirements:
  - Reduce infeasibility (e.g. $\sum_{i \in E} |c_i(x)| + \sum_{i \in I} \max\{-c_i(x), 0\}$)
  - Reduce objective function.
  - It has a multiobjective optimization nature!

15.3.1 MERIT FUNCTIONS. Merit function

- One idea also from multiobjective optimization: minimize a weighted combination of the 2 criteria.

$$\phi(x) = w_1 f(x) + w_2 \left[\sum_{i \in E} |c_i(x)| + \sum_{i \in I} \max\{-c_i(x), 0\} \right]; \quad w_1, w_2 > 0$$

- But I can scale it so that the weight of the objective is 1.
- In that case, the weight of the infeasibility measure is called "penalty parameter".
- I can monitor progress by ensuring that $\phi(x)$ decreases, as in unconstrained optimization.

Nonsmooth Penalty Merit Functions

$$\phi_1(x;\mu) = f(x) + \mu \sum_{i\in\mathcal{E}} |c_i(x)| + \mu \sum_{i\in\mathcal{I}} [c_i(x)]^-, \qquad [z]^- = \max\{0, -z\}.$$

Penalty parameter

- It is called the l1 merit function.
- Sometimes, they can be even EXACT.

Definition 15.1 (Exact Merit Function).

A merit function $\phi(x;\mu)$ is exact *if there is a positive scalar μ^* such that for any $\mu > \mu^*$, any local solution of the nonlinear programming problem (15.1) is a local minimizer of $\phi(x;\mu)$.*

We show in Theorem 17.3 that, under certain assumptions, the ℓ_1 merit function $\phi_1(x;\mu)$ is exact and that the threshold value μ^* is given by

$$\mu^* = \max\{|\lambda_i^*|,\ i \in \mathcal{E} \cup \mathcal{I}\},$$

Smooth and Exact Penalty Functions

- Excellent convergence properties, but very expensive to compute.
- Fletcher's augmented Lagrangian:

$$\phi_F(x;\mu) = f(x) - \lambda(x)^T c(x) + \tfrac{1}{2}\mu \sum c_i(x)^2,$$

$$\lambda(x) = [A(x)A(x)^T]^{-1} A(x)\nabla f(x).$$

- It is both smooth and exact, but perhaps impractical due to the linear solve.
- There are versions which enforce this inexactly

Augmented Lagrangian

- Smooth, but inexact. $\phi(x) = f(x) - \sum_{i \in \mathcal{E}} \lambda_i c_i(x) + \frac{\mu}{2} \sum_{i \in \mathcal{E}} c_i(x)^2$

- An update of the Lagrange Multiplier is needed.
- We will not use it, except with Augmented Lagrangian methods themselves.

Line-search (Armijo) for Nonsmooth Merit Functions

$$\phi_1(x;\mu) = f(x) + \mu \sum_{i\in\mathcal{E}} |c_i(x)| + \mu \sum_{i\in\mathcal{I}} [c_i(x)]^-,$$

- How do we carry out the "progress search"?
- That is the line search or the sufficient reduction in trust region?
- In the unconstrained case, we had

$$f(x_k) - f(x_k + \beta^m d_k) \geq -\rho\beta^m \nabla f(x_k)^T d_k; \quad 0 < \beta < 1, 0 < \rho < 0.5$$

- But we cannot use this anymore, since the function is not differentiable.

Directional Derivatives of Nonsmooth Merit Function

$$\phi_1(x;\mu) = f(x) + \mu \sum_{i\in\mathcal{E}} |c_i(x)| + \mu \sum_{i\in\mathcal{I}} [c_i(x)]^-,$$

$$D\left(\phi(x,\mu);p\right) = \lim_{t\to 0, t>0} \frac{\phi(x+tp,\mu)-\phi(x,\mu)}{t}; \quad D\left(\max\{f_1,f_2\},p\right) = \max\{\nabla f_1 p, \nabla f_1 p\}$$

- Nevertheless, the function has a directional derivative (follows from properties of max function).

$$\phi(x_k,\mu)-\phi(x_k+\beta^m p_k,\mu) \geq -\rho\beta^m D\left(\phi(x_k,\mu),p_k\right);$$

- Line Search: $\phi(x_k,\mu)-\phi(x_k+\beta^m p_k,\mu) \geq -\eta_1\left(m(0)-m(p_k)\right);$
- Trust Region $0<\eta_1<0.5$

And …. How do I choose the penalty parameter?

- VERY tricky issue, highly dependent on the penalty function used.
- For the l1 function, guideline is:

$$\mu^* = \max\{|\lambda_i^*|,\ i \in \mathcal{E} \cup \mathcal{I}\},$$

- But almost always adaptive. Criterion: If optimality gets ahead of feasibility, make penalty parameter more stringent.
- E.g l1 function: the max of current value of multipliers plus safety factor (EXPAND)
  -

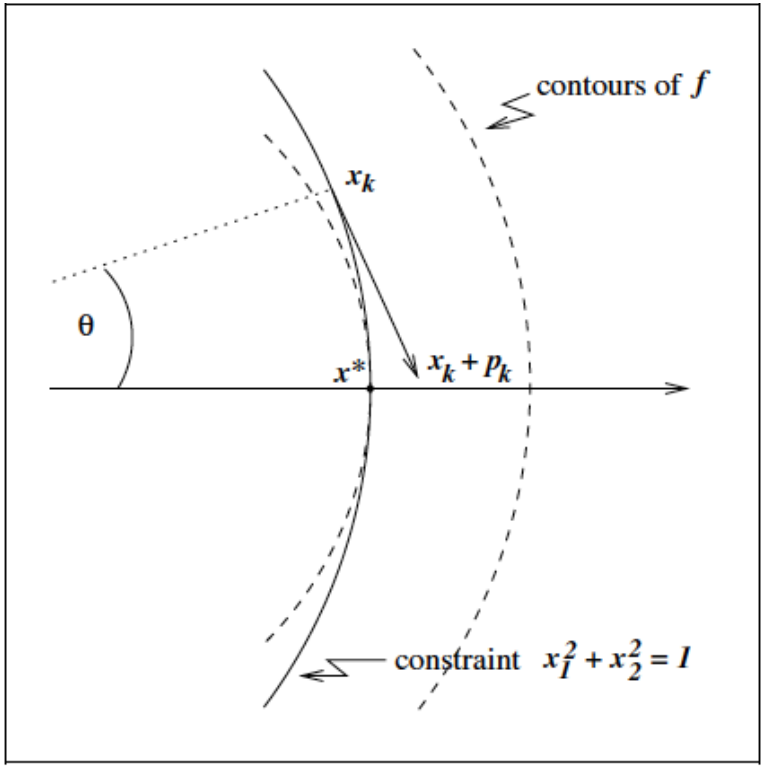
15.4 MARATOS EFFECT

Unfortunately, the Newton step may not be compatible with penalty

- This is called the Maratos effect.
- Problem:

$$\min\ f(x_1, x_2) = 2(x_1^2 + x_2^2 - 1) - x_1,$$

$$x_1^2 + x_2^2 - 1 = 0.$$



- From any feasible point a Newton-type step increases both infeasibility and objective function!!
- So fast convergence does not occur when coupled with globalization mechanisms.

Solutions?

- (a) Use Fletcher's function that does not suffer from this problem, but both filter and line search suffer from this.
- (b) Do curvilinear search:
  - Following a step computation: $A_k p_k + c(x_k) = 0.$
  - Use a correction that satisfies $A_k \hat{p}_k + c(x_k + p_k) = 0.$

    $\hat{p}_k = -A_k^T (A_k A_k^T)^{-1} c(x_k + p_k),$
  - Followed by the update or curvilinear line search:

    $x_k + p_k + \hat{p}_k$ $\qquad$ $x_k + \tau p_k + \tau^2 \hat{p}_k$
  - Since $c(x_k + p_k + \hat{p}_k) = O\left(\|x_k - x^*\|^3\right)$ compared to $c(x_k + p_k) = O\left(\|x_k - x^*\|^2\right)$ corrected Newton step is likelier to be accepted.