THE UNIVERSITY OF
CHICAGO

S310

SEC 16 QUADRATIC PROGRAMMING

What is a QP?

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

- If Q is PSD Problem is called convex (can be unbounded or infeasible)
- If Q is PD it is called strongly convex (can be infeasible, but not unbounded)
- If Q is indefinite it can be a very hard problem. (NP-hard)

16.1 QPs WITH EQUALITY CONSTRAINTS

- Problem:

$$\begin{aligned} \min_x \quad & q(x) \stackrel{\text{def}}{=} \tfrac{1}{2} x^T G x + x^T c \\ \text{subject to} \quad & Ax = b, \end{aligned}$$

- Optimality Conditions:

$$\begin{bmatrix} G & A^T \\ A & 0 \end{bmatrix} \begin{bmatrix} -p \\ \lambda^* \end{bmatrix} = \begin{bmatrix} g \\ h \end{bmatrix},$$

$$h = Ax - b, \qquad g = c + Gx, \qquad p = x^* - x.$$

- When is a solution of KKT a solution of the minimization problem?

16.2 Direct Solution. Inertia of the KKT matrix

- Separate the eigenvalues of a symmetric matrix by sign. Define:

$$\text{inertia}(K) = (n_+, n_-, n_0).$$

- Result: (K=kkt matrix)

Let K be defined by (16.7), and suppose that A has rank m. Then

$$\text{inertia}(K) = \text{inertia}(Z^T G Z) + (m, m, 0).$$

Therefore, if $Z^T G Z$ is positive definite, $\text{inertia}(K) = (n, m, 0)$.

LDLT factorization

- Formulation:
- Solving the KKT system

$$P^T K P = L B L^T,$$

$$\begin{aligned}
&\text{solve } Lz = P^T \begin{bmatrix} g \\ h \end{bmatrix} && \text{to obtain } z;\\
&\text{solve } B\hat{z} = z && \text{to obtain } \hat{y};\\
&\text{solve } L^T \bar{z} = \hat{z} && \text{to obtain } \bar{z};\\
&\text{set} \begin{bmatrix} -p \\ \lambda^* \end{bmatrix} = P\bar{z}.
\end{aligned}$$

- Sylvester theorem:
- And B should have (n,m,0) inertia !!!

16.5 INTERIOR-POINT METHODS FOR CONVEX QP WITH INEQ CONSTRAINTS

- The form of the problem solved:

$$\begin{aligned} \min_x \quad & q(x) = \tfrac{1}{2} x^T G x + x^T c \\ \text{subject to} \quad & Ax \geq b, \end{aligned}$$

$$A = [a_i]_{i \in \mathcal{I}}, \quad b = [b_i]_{i \in \mathcal{I}}, \quad \mathcal{I} = \{1, 2, \ldots, m\}.$$

Optimality Conditions

- In original form:

$$\begin{aligned} Gx - A^T\lambda + c &= 0, \\ Ax - b &\geq 0, \\ (Ax - b)_i\lambda_i &= 0, \quad i = 1, 2, \ldots, m, \\ \lambda &\geq 0. \end{aligned}$$

- With slacks:

$$\begin{aligned} Gx - A^T\lambda + c &= 0, \\ Ax - y - b &= 0, \\ y_i\lambda_i &= 0, \quad i = 1, 2, \ldots, m, \\ (y, \lambda) &\geq 0. \end{aligned}$$

- Note: Extension when equalities are also included is straightforward

Idea: define an "interior" path to the solution

- Define the perturbed KKT conditions as a nonlinear system:

$$F(x, y, \lambda; \sigma\mu) = \begin{bmatrix} Gx - A^T\lambda + c \\ Ax - y - b \\ y\Lambda e - \sigma\mu e \end{bmatrix} = 0,$$

- Solve successively while taking mu to 0 solves KKT:

$$F\left(x(\mu), y(\mu), \lambda(\mu); \mu, \sigma\right) = 0;\ y_i(\mu)\lambda_i(\mu) = \mu\sigma > 0$$

$$\mu \to 0 \Rightarrow \left(x(\mu), y(\mu), \lambda(\mu)\right) \to \left(x^*, y^*, \lambda^*\right) \text{ satisfies KKT}$$

How to solve it ?

- Solution: apply Newton's method for fixed mu:

$$\begin{bmatrix} G & 0 & -A^T \\ A & -I & 0 \\ 0 & \Lambda & \mathcal{Y} \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta \lambda \end{bmatrix} = \begin{bmatrix} -r_d \\ -r_p \\ -\Lambda \mathcal{Y} e + \sigma \mu e \end{bmatrix},$$

- New iterate:

$$r_d = Gx - A^T\lambda + c, \qquad r_p = Ax - y - b.$$

- Enforce:

$$(x^+, y^+, \lambda^+) = (x, y, \lambda) + \alpha(\Delta x, \Delta y, \Delta \lambda),$$

$$(y^+, \lambda^+) > 0;$$

How to PRACTICALLY solve it?

- Eliminate the slacks:

$$\begin{bmatrix} G & -A^T \\ A & \Lambda^{-1}\mathcal{Y} \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta\lambda \end{bmatrix} = \begin{bmatrix} -r_d \\ -r_p + (-y + \sigma\mu\Lambda^{-1}e) \end{bmatrix}.$$

- Eliminate the multipliers (note, the 22 block is diagonal and invertible)

$$(G + A^T\mathcal{Y}^{-1}\Lambda A)\Delta x = -r_d + A^T\mathcal{Y}^{-1}\Lambda[-r_p - y + \sigma\mu\Lambda^{-1}e],$$

- Solve (e.g by Cholesky)

Projected CG:

- The QP has the SAME structure as an EQP and can use projected CG:

$$\begin{bmatrix} G & 0 & -A^T \\ 0 & \mathcal{Y}^{-1}\Lambda & I \\ A & -I & 0 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta \lambda \end{bmatrix} = \begin{bmatrix} -r_d \\ -\Lambda e + \sigma \mu \mathcal{Y}^{-1} e \\ -r_p \end{bmatrix},$$

- We would have to do QR on [A –I] first.

How to choose the step?

- We need to enforce positivity.
- A typical approach:

$$\alpha_\tau^{\text{pri}} = \max\{\alpha \in (0,1] : y + \alpha \Delta y \geq (1-\tau)y\},$$
$$\alpha_\tau^{\text{dual}} = \max\{\alpha \in (0,1] : \lambda + \alpha \Delta\lambda \geq (1-\tau)\lambda\};$$

$$\alpha = \min(\alpha_\tau^{\text{pri}}, \alpha_\tau^{\text{dual}}),$$

- Here tau is a user defined parameter (say 0.1 – 0.01).

A practical primal-dual method

- First, compute an affine scaling step (that is, drive to solution and not to center) $\sigma = 0$. This allows us to move faster. Denote it by $(\Delta x^{\text{aff}}, \Delta y^{\text{aff}}, \Delta\lambda^{\text{aff}})$
- Then, move towards the center to make sure that, taking a Newton from this point to the center (16.66)

$$\begin{bmatrix} G & 0 & -A^T \\ A & -I & 0 \\ 0 & \Lambda & \mathcal{Y} \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta\lambda \end{bmatrix} = \begin{bmatrix} -r_d \\ -r_p \\ -\Lambda\mathcal{Y}e - \Delta\Lambda^{\text{aff}}\Delta\mathcal{Y}^{\text{aff}}e + \sigma\mu e \end{bmatrix}$$

- Note I reuse the same factorization! So the effort is small in changed right hand side.

A practical primal-dual algorithm

Algorithm 16.4 (Predictor-Corrector Algorithm for QP).

Compute (x_0, y_0, λ_0) with $(y_0, \lambda_0) > 0$;

for $k = 0, 1, 2, \ldots$

Set $(x, y, \lambda) = (x_k, y_k, \lambda_k)$ and solve (16.58) with $\sigma = 0$ for $(\Delta x^{\text{aff}}, \Delta y^{\text{aff}}, \Delta \lambda^{\text{aff}})$;

Calculate $\mu = y^T \lambda / m$;

Calculate $\hat{\alpha}_{\text{aff}} = \max\{\alpha \in (0, 1] \mid (y, \lambda) + \alpha(\Delta y^{\text{aff}}, \Delta \lambda^{\text{aff}}) \geq 0\}$;

Calculate $\mu_{\text{aff}} = (y + \hat{\alpha}^{\text{aff}} \Delta y^{\text{aff}})^T (\lambda + \hat{\alpha}^{\text{aff}} \Delta \lambda^{\text{aff}})/m$;

Set centering parameter to $\sigma = (\mu_{\text{aff}}/\mu)^3$;

Solve (16.67) for $(\Delta x, \Delta y, \Delta \lambda)$;

Choose $\tau_k \in (0, 1)$ and set $\hat{\alpha} = \min(\alpha_{\tau_k}^{\text{pri}}, \alpha_{\tau_k}^{\text{dual}})$ (see (16.66));

Set $(x_{k+1}, y_{k+1}, \lambda_{k+1}) = (x_k, y_k, \lambda_k) + \hat{\alpha}(\Delta x, \Delta y, \Delta \lambda)$;

end (for)

16.6 GRADIENT PROJECTIONS FOR QPS WITH BOUND CONSTRAINTSn

- The problem:

$$\begin{aligned} \min_x \quad & q(x) = \tfrac{1}{2} x^T G x + x^T c \\ \text{subject to} \quad & l \le x \le u, \end{aligned}$$

- Like in the trust-region case, we look for a Cauchy point, based on a projection on the feasible set.
- G does not have to be psd (essential for AugLag)
- The projection operator:

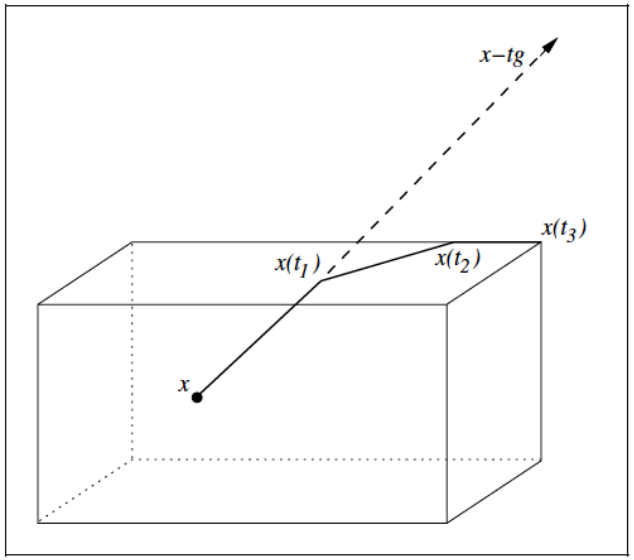
$$P(x, l, u)_i = \begin{cases} l_i & \text{if} \quad x_i < l_i, \\ x_i & \text{if} \quad x_i \in [l_i, u_i], \\ u_i & \text{if} \quad x_i > u_i. \end{cases}$$

The search path

- Create a piecewise linear path which is feasible (as opposed to the linear one in the unconstrained case) by projection of gradient.

$$x(t) = P(x - tg, l, u),$$

$$g = Gx + c;$$



Computation of breakpoints

- Can be done on each component individually

$$\bar{t}_i = \begin{cases} (x_i - u_i)/g_i & \text{if } g_i < 0 \text{ and } u_i < +\infty, \\ (x_i - l_i)/g_i & \text{if } g_i > 0 \text{ and } l_i > -\infty, \\ \infty & \text{otherwise.} \end{cases}$$

- Then the search path becomes on each component:

$$x_i(t) = \begin{cases} x_i - t g_i & \text{if } t \le \bar{t}_i, \\ x_i - \bar{t}_i g_i & \text{otherwise.} \end{cases}$$

Line Search along piecewise linear path

- Reorder the breakpoints eliminating duplicates and zero values to get
- The path: $0 < t_1 < t_2 < \ldots$
- Whose direction is:

$$x(t) = x(t_{j-1}) + (\Delta t)p^{j-1}, \qquad \Delta t = t - t_{j-1} \in [0, t_j - t_{j-1}],$$

$$p_i^{j-1} = \begin{cases} -g_i & \text{if } t_{j-1} < \bar{t}_i, \\ 0 & \text{otherwise.} \end{cases}$$

Line Search (2)

- Along each piece, $[t_{j-1}, t_j]$ find the minimum of the quadratic $\frac{1}{2}x^T Gx + c^T x$
- This reduces to analyzing a one dimensional quadratic form of t on an interval.
- If the minimum is on the right end of interval, we continue.
- If not, we found the local minimum and the Cauchy point == the first local minimizer (it is different a bit than trust-region, plus you are not guaranteed to find the global minimum anyways.)

Subspace Minimization

- Active set of Cauchy Point

$$\mathcal{A}(x^c) = \{i \mid x_i^c = l_i \text{ or } x_i^c = u_i\}.$$

- Solve subspace minimization problem

$$\begin{aligned} \min_x \; & q(x) = \tfrac{1}{2} x^T G x + x^T c \\ \text{subject to} \quad & x_i = x_i^c, \; i \in \mathcal{A}(x^c), \\ & l_i \le x_i \le u_i, \; i \notin \mathcal{A}(x^c). \end{aligned}$$

- No need to solve exactly. For example truncated CG with termination if one inactive variable reaches bound.

Gradient Projection for QP

Algorithm 16.5 (Gradient Projection Method for QP).

Compute a feasible starting point x^0;

for $k = 0, 1, 2, \ldots$

if x^k satisfies the KKT conditions for (16.68)

stop with solution $x^* = x^k$;

Set $x = x^k$ and find the Cauchy point x^c;

Find an approximate solution x^+ of (16.74) such that $q(x^+) \leq q(x^c)$

and x^+ is feasible;

$x^{k+1} \leftarrow x^+$;

end (for)

Or, equivalently, if projection does not advance from 0.

Observations – Gradient Projection

- Note that the Projection – Active set solve loop must be iterated to optimality.
- What is the proper stopping criteria? How do we verify the KKT?
- Idea: When projection does not progress (x is a fixed point of the gradient projection)!
- That is, on each component, either the gradient is 0, or the x component is at the right bound!