High-Precision Quadratic Programming by Iterative Refinement

Tobias Weber, Ambros Gleixner, Sebastian Sager

Institute for Mathematical Optimization
Otto-von-Guericke-University Magdeburg, Germany
Berlin, Tuesday 12th of July, 2016

Motivation

Areas of Application:

- Least squares
- Computational geometry[1]
- Support vector machines
- SQP[2]

[1] Bernd Gärtner and Sven Schönherr. An efficient, exact, and generic quadratic programming solver for geometric optimization. Proceedings of the sixteenth annual symposium on Computational geometry. ACM, 2000.

[2] Johnson, Travis C., Christian Kirches, and Andreas Wächter. An active-set method for quadratic programming based on sequential hot-starts. SIAM Journal on Optimization 25.2 (2015): 967-994.

Motivation

Areas of Application:

- Least squares
- Computational geometry[1]
- Support vector machines
- SQP[2]

Standard description:

$$\begin{aligned} \min_x \quad & \frac{1}{2} x^T Q x + c^T x \\ \text{s.t.:} \quad & Ax = b \\ & x \geq l \end{aligned}$$

[1] Bernd Gärtner and Sven Schönherr. An efficient, exact, and generic quadratic programming solver for geometric optimization. Proceedings of the sixteenth annual symposium on Computational geometry. ACM, 2000.

[2] Johnson, Travis C., Christian Kirches, and Andreas Wächter. An active-set method for quadratic programming based on sequential hot-starts. SIAM Journal on Optimization 25.2 (2015): 967-994.

First Order Optimality

The Karush-Kuhn-Tucker (KKT) conditions read:

$$
\begin{aligned}
Ax^* &= b \\
x^* &\geq l \\
Qx^* + c - A^T y^* - z^* &= 0 \\
(x^* - l)^T z^* &= 0 \\
z^* &\geq 0
\end{aligned}
$$

First Order Optimality

The Karush-Kuhn-Tucker (KKT) conditions read:

$$
\begin{aligned}
Ax^* &= b \\
x^* &\geq l \\
Qx^* + c - A^T y^* - z^* &= 0 \\
(x^* - l)^T z^* &= 0 \\
z^* &\geq 0
\end{aligned}
$$

Usually a solver provides solutions $(\tilde{x}^*, \tilde{y}^*, \tilde{z}^*)$ of the form e.g.:

$$\tilde{z}_i^* \geq -\varepsilon \quad (\varepsilon > 0)$$

Basic Solution

$$
\begin{aligned}
Qx^* + c - A^T y^* - z^* &= 0 \qquad & x^* - l, z^* &\geq 0 \\
Ax^* &= b \qquad & (x^* - l)^T z^* &= 0
\end{aligned}
$$

Basic Solution

$$
\begin{aligned}
Qx^* + c - A^T y^* - z^* &= 0 \qquad & x^* - l, z^* &\geq 0 \\
Ax^* &= b \qquad & (x^* - l)^T z^* &= 0
\end{aligned}
$$

$$
\begin{aligned}
\{1, ..., n\} &= \mathcal{B} \cup \mathcal{N} \\
\emptyset &= \mathcal{B} \cap \mathcal{N}
\end{aligned}
$$

Basic Solution

$$
\begin{aligned}
Qx^* + c - A^T y^* - z^* &= 0 \qquad\qquad & x^* - l, z^* &\geq 0 \\
Ax^* &= b & x^*_{\mathcal{N}} = l_{\mathcal{N}}, z^*_{\mathcal{B}} &= 0
\end{aligned}
$$

$$
\begin{aligned}
\{1, ..., n\} &= \mathcal{B} \cup \mathcal{N} \\
\emptyset &= \mathcal{B} \cap \mathcal{N}
\end{aligned}
$$

Basic Solution

$$Qx^* + c - A^T y^* - z^* = 0 \qquad x^* - l, z^* \geq 0$$

$$Ax^* = b \qquad x^*_{\mathcal{N}} = l_{\mathcal{N}}, z^*_{\mathcal{B}} = 0$$

$$\{1, ..., n\} = \mathcal{B} \cup \mathcal{N}$$

$$\emptyset = \mathcal{B} \cap \mathcal{N}$$

Basic solution:

$$\begin{pmatrix} -Q_{\mathcal{B}\mathcal{B}} & A^T_{\cdot\mathcal{B}} \\ A_{\cdot\mathcal{B}} & 0 \end{pmatrix} \begin{pmatrix} x_{\mathcal{B}} \\ y \end{pmatrix} = \begin{pmatrix} c_{\mathcal{B}} + Q_{\mathcal{B}\mathcal{N}} l_{\mathcal{N}} \\ b - A_{\cdot\mathcal{N}} l_{\mathcal{N}} \end{pmatrix}$$

Iterative Refinement

Consider $Ax = b$ with approximative solution $\tilde{x}^0$, in each iteration:

Iterative Refinement

Consider $Ax = b$ with approximative solution $\tilde{x}^0$, in each iteration:

- Calculate the error
  $\tilde{r} = b - A\tilde{x}^i$
- Solve $A\tilde{d} = \tilde{r}$
- Correct solution
  $\tilde{x}^{i+1} = \tilde{x}^i + \tilde{d}$

Iterative Refinement

Consider $Ax = b$ with approximative solution $\tilde{x}^0$, in each iteration:

- Calculate the error
  $\tilde{r} = b - A\tilde{x}^i$
- Solve $A\tilde{d} = \tilde{r}$
- Correct solution
  $\tilde{x}^{i+1} = \tilde{x}^i + \tilde{d}$

- Calculate the error
  $\tilde{r} = b - A\tilde{x}^i$
- Solve $C\tilde{d} = \tilde{r}$ ($C \approx A$)
- Correct solution
  $\tilde{x}^{i+1} = \tilde{x}^i + \tilde{d}$

Iterative Refinement

Consider $Ax = b$ with approximative solution $\tilde{x}^0$, in each iteration:

- Calculate the error $\tilde{r} = b - A\tilde{x}^i$
- Solve $A\tilde{d} = \tilde{r}$
- Correct solution $\tilde{x}^{i+1} = \tilde{x}^i + \tilde{d}$

Call this exact refinement?

- Calculate the error $\tilde{r} = b - A\tilde{x}^i$
- Solve $C\tilde{d} = \tilde{r}$ ($C \approx A$)
- Correct solution $\tilde{x}^{i+1} = \tilde{x}^i + \tilde{d}$

Call this inexact refinement?

Scaled Iterative Refinement

What to do for higher precision? Scale the error:

Scaled Iterative Refinement

What to do for higher precision? Scale the error:

- Calculate $r = (b - Ax^i)\Delta, \quad \Delta >> 1$ (exactly)
- Solve $A\tilde{d} = r$ (approximately)
- Correct solution $x^{i+1} = x^i + \tilde{d}/\Delta$ (exactly)

Scaled Iterative Refinement

What to do for higher precision? Scale the error:

- Calculate $r = (b - Ax^i)\Delta, \;\; \Delta >> 1$ (exactly)
- Solve $A\tilde{d} = r$ (approximately)
- Correct solution $x^{i+1} = x^i + \tilde{d}/\Delta$ (exactly)

Error:

$$Ax^{i+1} - b = A(x^i + \tilde{d}/\Delta) - b = \frac{\Delta Ax^i + A\tilde{d} - \Delta b}{\Delta} = \frac{A\tilde{d} - r}{\Delta}$$

Refine Basic Solutions

For some approximate point (x^*, y^*) we refine:

$$\begin{pmatrix} -Q & A^T \\ A & 0 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} c \\ b \end{pmatrix}$$

Refine Basic Solutions

For some approximate point (x^*, y^*) we refine:

$$\begin{pmatrix} -Q & A^T \\ A & 0 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} c \\ b \end{pmatrix}$$

and get:

$$\begin{pmatrix} -\frac{\Delta_P}{\Delta_D} Q & A^T \\ A & 0 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} \hat{c}\Delta_D \\ \hat{b}\Delta_P \end{pmatrix} = \begin{pmatrix} (Qx^* + c - A^T y^*)\Delta_D \\ (b - Ax^*)\Delta_P \end{pmatrix}$$

QP Refinement

Theorem (Gleixner, 2015)

Suppose we are given a QP in form

$$\min\{\frac{1}{2}x^TQx + c^Tx \mid Ax = b, x \geq l\}, \tag{1}$$

then for $x^* \in \mathbb{R}^n$, $y^* \in \mathbb{R}^m$, and scaling factors $\Delta_P, \Delta_D > 0$, consider the transformed problem

$$\min\{\frac{\Delta_P}{2\Delta_D}x^TQx + \Delta_D\hat{c}^Tx \mid Ax = \Delta_P\hat{b}, x \geq \Delta_P\hat{l}\}, \tag{2}$$

where $\hat{c} = Qx^* + c - A^Ty^*$, $\hat{b} = b - Ax^*$, and $\hat{l} = l - x^*$. Then for any $\hat{x} \in \mathbb{R}^n$, $\hat{y} \in \mathbb{R}^m$ the following holds:

- *$\hat{x}$ ($\hat{y}$) is primal (dual) feasible for (2) within an absolute tolerance $\varepsilon_P > 0$ ($\varepsilon_D > 0$) iff $x^* + \frac{\hat{x}}{\Delta_P}$ ($y^* + \frac{\hat{y}}{\Delta_D}$) is primal (dual) feasible for (1) within ε_P/Δ_P (ε_D/Δ_D).*
- *$\hat{x}$, $\hat{y}$ satisfy complementary slackness for (2) within an absolute tolerance $\varepsilon_S > 0$ iff $x^* + \frac{\hat{x}}{\Delta_P}$, $y^* + \frac{\hat{y}}{\Delta_D}$ satisfy complementary slackness for (1) within $\varepsilon_S/(\Delta_P\Delta_D)$.*

The Algorithm

$$\text{QP}\begin{cases} \min & x^T Q x + \quad c^T x \\ \text{s. t.} & \qquad Ax = \quad b \\ & \qquad\; x \geq \quad 0 \end{cases}$$

The Algorithm

$$x^* \mapsto \hat{x}$$

$$y^* \mapsto \hat{y}$$

$$\text{QP}\left\{\begin{array}{rrl} \min & x^TQx + & c^Tx \\ \text{s. t.} & Ax = & b \\ & x \geq & 0 \end{array}\right.$$

The Algorithm

$$x^* \mapsto \hat{x} \qquad y^* \mapsto \hat{y}$$

$$\text{QP}\left\{\begin{array}{rrl} \min & x^TQx + & c^Tx \\ \text{s. t.} & Ax = & b \\ & x \geq & 0 \end{array}\right. \qquad \left.\begin{array}{r} \hat{c} = Qx^* + c - A^Ty^* \\ \hat{b} = b - Ax^* \\ \hat{l} = x^* \end{array}\right\}\text{Ref.}$$

The Algorithm

$$
\text{QP}\left\{\begin{array}{lrl} \min & x^TQx + & c^Tx \\ \text{s. t.} & Ax = & b \\ & x \geq & 0 \end{array}\right. \xrightarrow[y^* \mapsto \hat{y}]{x^* \mapsto \hat{x}} \left.\begin{array}{r} \hat{c} = Qx^* + c - A^Ty^* \\ \hat{b} = b - Ax^* \\ \hat{l} = x^* \end{array}\right\}\text{Ref.}
$$

$$
\text{Ref.} \xrightarrow[\hat{l},\ \Delta]{\hat{c},\ \hat{b}} \text{QP}
$$

The Algorithm

$$
\begin{tikzcd}
\text{QP}\left\{\begin{array}{ll}\min & x^TQx+\Delta\hat{c}^Tx\\ \text{s. t.} & Ax=\Delta\hat{b}\\ & x\ge\Delta\hat{l}\end{array}\right. \arrow[r, bend left, "x^*\mapsto\hat{x}", "y^*\mapsto\hat{y}"'] & \left.\begin{array}{r}\hat{c}=Qx^*+c-A^Ty^*\\ \hat{b}=b-Ax^*\\ \hat{l}=x^*\end{array}\right\}\text{Ref.} \arrow[l, bend left, "\hat{c},\ \hat{b}"', "\hat{l},\ \Delta"]
\end{tikzcd}
$$

The Algorithm

$$x^* \mapsto x^* + \hat{x}/\Delta$$

$$y^* \mapsto y^* + \hat{y}/\Delta$$

$$\text{QP}\left\{\begin{array}{lr} \min & x^T Q x + \Delta \hat{c}^T x \\ \text{s. t.} & Ax = \Delta \hat{b} \\ & x \geq \Delta \hat{l} \end{array}\right. \qquad \left.\begin{array}{r} \hat{c} = Qx^* + c - A^T y^* \\ \hat{b} = b - Ax^* \\ \hat{l} = x^* \end{array}\right\}\text{Ref.}$$

$$\hat{c},\ \hat{b}$$

$$\hat{l},\ \Delta$$

The Implementation

- Extension of SoPlex LP refinment framework[3]
- Use of qpOASES to solve QPs[4]
- GMP for rational calculations (operator overloading)
- Templates for different precision oparations

[3] Gleixner, Ambros M., Daniel E. Steffy, and Kati Wolter. Improving the accuracy of linear programming solvers with iterative refinement. Proceedings of the 37th International Symposium on Symbolic and Algebraic Computation. ACM, 2012.

[4] Ferreau, Hans Joachim, et al. qpOASES: A parametric active-set algorithm for quadratic programming. Mathematical Programming Computation 6.4 (2014): 327-363.

The Implementation

- Extension of SoPlex LP refinment framework[3]
- Use of qpOASES to solve QPs[4]
- GMP for rational calculations (operator overloading)
- Templates for different precision oparations
- Rational (and floating point) reading and writing of QPS
- Adaptive scaling and solving
- Sparse solving with qpOASES

[3] Gleixner, Ambros M., Daniel E. Steffy, and Kati Wolter. Improving the accuracy of linear programming solvers with iterative refinement. Proceedings of the 37th International Symposium on Symbolic and Algebraic Computation. ACM, 2012.

[4] Ferreau, Hans Joachim, et al. qpOASES: A parametric active-set algorithm for quadratic programming. Mathematical Programming Computation 6.4 (2014): 327-363.

Results

Solver	Time	Iter	Ref	Fails	Tol. reached	Incons.
qpOASES*	0.06	97	0	0	70	3

Results for Maros Mészáros subset: Problems with less than 1000 variables and constraints (73). *Tolerance $\approx 10^{-9}$, **Tolerance 10^{-100}

Results

Solver	Time	Iter	Ref	Fails	Tol. reached	Incons.
qpOASES*	0.06	97	0	0	70	3
QPIR**	0.19	104	10	11	59	3

Results for Maros Mészáros subset: Problems with less than 1000 variables and constraints (73). *Tolerance $\approx 10^{-9}$, **Tolerance 10^{-100}

Results

Solver	Time	Iter	Ref	Fails	Tol. reached	Incons.
qpOASES*	0.06	97	0	0	70	3
QPIR**	0.19	104	10	11	59	3
QPIR SPAR**	-	-	-	25	45	3

Results for Maros Mészáros subset: Problems with less than 1000 variables and constraints (73). *Tolerance $\approx 10^{-9}$, **Tolerance 10^{-100}

Results

Solver	Time	Iter	Ref	Fails	Tol. reached	Incons.
qpOASES*	0.06	97	0	0	70	3
QPIR**	0.19	104	10	11	59	3
QPIR SPAR**	-	-	-	25	45	3
QPIR RELX**	0.14	26	8	3	67	3

Results for Maros Mészáros subset: Problems with less than 1000 variables and constraints (73). *Tolerance $\approx 10^{-9}$, **Tolerance 10^{-100}

Relaxed settings: MPC + NZCTests + DriftCorrection + Ramping; Reliable + numRefinmentSteps 10

Future Work

- Solve basic system in rational precision
- Handle infeasible and unbounded QPs
- Compare with existing solver for geometric optimization[5]

[5] Bernd Gärtner and Sven Schönherr. An efficient, exact, and generic quadratic programming solver for geometric optimization. Proceedings of the sixteenth annual symposium on Computational geometry. ACM, 2000.

The Algorithm

Solve $\min\{\frac{1}{2}x^TQx + c^Tx \| Ax = b, x \geq 0\}$ approximately and get basis $\mathcal{B}$ and x^1, y^1 returned as optimal. Define maximal scaling per iteration α and tolerances $\varepsilon_P, \varepsilon_D$. For $k = 1, 2, ...k_{max}$ do:

- $\hat{b} = b - Ax^k$
- $\delta_P = \max\{\|\hat{b}\|_\infty, \max_i -x_i^k\}$
- $\hat{c} = Qx^k + c - A^Ty^k$
- $\delta_D = \max\{\|\hat{c}_\mathcal{B}\|_\infty, \max_{i \notin \mathcal{B}} -\hat{c}_i\}$
- $\Delta = \min\{1/\delta_P, 1/\delta_D, \alpha\Delta\}$
- If $\delta_P \leq \varepsilon_P$ and $\delta_D \leq \varepsilon_D$: break
- Else:
  - Solve $\min\{\frac{1}{2}x^TQx + \Delta\hat{c}^Tx \| Ax = \Delta\hat{b}, x \geq -\Delta x^k\}$ (Hotstart!)
  - Get $\mathcal{B}$, $\tilde{x}^*$, and $\tilde{y}^*$
  - Update $x^{k+1} = x^k + \frac{\tilde{x}^*}{\Delta}$ and $y^{k+1} = y^k + \frac{\tilde{y}^*}{\Delta}$
  - $x_i^{k+1} = 0 \; \forall \; i \notin \mathcal{B}$