UNIVERSITÄT HEIDELBERG
Zukunft. Seit 1386.

A sparse variant of qpOASES

based on a symmetric indefinite factorization and Schur complement updates

Dennis Janka[1], Christian Kirches[1], Andreas Wächter[2]

[1]Interdisciplinary Center for Scientific Computing (IWR)
Heidelberg University, Germany

[2]Industrial Engineering and Management Sciences
Northwestern University, Evanston, IL, U.S.A.

IMTEK-TEMPO Workshop on Embedded Quadratic Programming (EQP2014)
University of Freiburg, Germany

March 19, 2014

Outline

- Parametric quadratic programming
- Linear algebra tasks in qpOASES
- Structures and structure exploitation

- Sparse factorizations
- Schur complement updates
- Some bits and pieces ...

- Ongoing projects using sparse qpOASES

Parametric Quadratic Programming

Let $\mathcal{L}^k$ denote the set of one-parameter affine linear functions onto $\mathbb{R}^k$,

$$\mathcal{L}^k = \{f : [0,1] \to \mathbb{R}^k \mid \forall \tau \in (0,1): \ f(\tau) = (1-\tau)f(0) + \tau f(1)\}.$$

For $g, l^\mathrm{b}, u^\mathrm{b} \in \mathcal{L}^n$ and $l^\mathrm{A}, u^\mathrm{A} \in \mathcal{L}^m$ we are interested in solving

$$\begin{array}{cl} \min\limits_{x} & \frac{1}{2}x^T H x + g(\tau)^T x \\ \text{s.t.} & u^\mathrm{A}(\tau) \geq Ax \geq l^\mathrm{A}(\tau) \\ & u^\mathrm{b}(\tau) \geq x \geq l^\mathrm{b}(\tau) \end{array} \qquad (\mathrm{PQP}(\tau))$$

$H \in \mathbb{R}^{n\times n}$ sym. pos. def., $A \in \mathbb{R}^{m\times n}$.

Theorem: Under primal and dual nondegeneracy, the solution $x : [0,1] \to \mathbb{R}^n$ and the multipliers $y : [0,1] \to \mathbb{R}^m$ are piecewise affine linear in the parameter τ.

Determine pieces of x, y iteratively.

Critical Regions

Let $(x(0), y(0))$ be optimal for PQP(0).

Optimality conditions:

$$0 = Hx + g(0) - A^T y^A - y^b$$

$$0 \leq r^A(0) = Ax - l^A(0)$$
$$0 \leq r^b(0) = x - l^b(0)$$

$$0 \leq y_i^A \text{ if } 0 = r^A(0)$$
$$0 = y_i^A \text{ if } 0 < r^A(0)$$

$$0 \leq y_i^b \text{ if } 0 = r^b(0)$$
$$0 = y_i^b \text{ if } 0 < r^b(0)$$

Upper bounds ignored for simplicity.

Critical Regions

Let $(x(0), y(0))$ be optimal for PQP(0).

Shift of $g(0), l(0)$ towards $g(1), l(1)$:

$$0 = H(x + \tau\Delta x) + g(\tau) - A^T(y^A + \tau\Delta y^A) - (y^b + \tau\Delta y^b)$$

$$0 \le r^A(0) + \tau\Delta r^A = A(x + \tau\Delta x) - l^A(\tau)$$
$$0 \le r^b(0) + \tau\Delta r^b = x + \tau\Delta x - l^b(\tau)$$

$$0 \le y_i + \tau\Delta y_i \text{ if } 0 = r_i = \Delta r_i$$
$$0 = y_i = \Delta y_i \text{ if } 0 < r_i$$

Upper bounds ignored for simplicity.

Critical Regions

Let $(x(0), y(0))$ be optimal for PQP(0).

Compute $(\Delta x, \Delta y)$ from $\Delta g = g(1) - g(0)$, $\Delta l = l(1) - l(0)$:

$$\begin{cases} -\Delta g = H\Delta x - A^T \Delta y^A - \Delta y^b \\ \Delta x \text{ such that active } r_i \text{ remain zero} \\ \Delta y \text{ such that inactive } y_i \text{ remain zero} \end{cases}$$

$2n + m$ conditions for $2n + m$ unknowns $(\Delta x, \Delta y^b, \Delta y^A)$.

Now choose τ maximal such that:

$$\begin{cases} -\tau \Delta r_i^A \leq r_i^A \text{ for i inactive} \\ -\tau \Delta r_i^b \leq r_i^b \text{ for i inactive} \end{cases}$$

$$\begin{cases} -\tau \Delta y_i^A \leq y_i^A \text{ for i active} \\ -\tau \Delta y_i^b \leq y_i^b \text{ for i active} \end{cases}$$

Upper bounds ignored for simplicity.

The qpOASES Algorithm

UNIVERSITÄT HEIDELBERG
Zukunft. Seit 1386.

Init: Let (x, y) solve PQP(0).

Solve: Compute $\Delta g = g(1) - g(\tau)$, $\Delta l = l(1) - l(\tau)$
Compute $(\Delta r, \Delta y)$ from $(\Delta g, \Delta l)$

Ratio: For all i inactive, let $\alpha_i = -r_i^A / \Delta r_i$
For all i active, let $\alpha_i = -y_i / \Delta y_i$
Let $i^* = \text{argmin}\{i \mid \alpha_i \geq 0\}$

Step: let $\alpha = \alpha_{i^*}$, let $\tau^+ = \tau + \alpha(1 - \tau)$,
let $r = r + \alpha \Delta r$, $y = y + \alpha \Delta y$

Term: If $\tau = 1$ then stop with (x, y) solving PQP(1).

LITest: If i^* inactive becomes active:
Test for linear dependence on actives
If needed, find j^* active that becomes inactive

NPCTest: If i^* active becomes inactive:
Test for non-positive curvature on null-space
If needed, find j^* inactive that becomes active

Loop: Go to **Solve**

Linear Algebra Tasks in qpOASES

Universität Heidelberg
Zukunft. Seit 1386.

- Working set $\mathcal{W} = (\mathcal{X}, \mathcal{A})$, (fixed vars, active cons), its complement is $\mathcal{W}^\mathrm{C} = (\mathcal{F}, \mathcal{I})$ (free vars, inactive cons)
- Computing $(\Delta x, \Delta y)$ for a working set $\mathcal{W}$ means solving

$$\underbrace{\begin{bmatrix} H_{\mathcal{X}\mathcal{X}} & H_{\mathcal{X}\mathcal{F}} & A^T_{\mathcal{A}\mathcal{X}} & I_{\mathcal{X}} \\ H_{\mathcal{F}\mathcal{X}} & H_{\mathcal{F}\mathcal{F}} & A^T_{\mathcal{A}\mathcal{F}} & \\ A_{\mathcal{A}\mathcal{X}} & A_{\mathcal{A}\mathcal{F}} & & \\ I_{\mathcal{X}} & & & \end{bmatrix}}_{=K_{\mathcal{W}}} \begin{bmatrix} \Delta x_{\mathcal{X}} \\ \Delta x_{\mathcal{F}} \\ -\Delta y^\mathrm{A}_{\mathcal{A}} \\ -\Delta y^\mathrm{b}_{\mathcal{X}} \end{bmatrix} = \begin{bmatrix} -\Delta g_{\mathcal{X}} \\ -\Delta g_{\mathcal{F}} \\ \Delta l^\mathrm{A}_{\mathcal{A}} \\ \Delta l^\mathrm{b}_{\mathcal{X}} \end{bmatrix}$$

- Per iteration, at most one member enters/leaves sets $\mathcal{W}$ and $\mathcal{W}^\mathrm{C}$
- Efficient approach: Factorize $K_{\mathcal{W}}$ once (expensive), maintain factorization through updates (cheap)

What some other active-set solvers do

UNIVERSITÄT HEIDELBERG
Zukunft. Seit 1386.

- Dense: Null-space method with updates (Gill et al., 70ies+80ies)
- Iterative: Preconditioned CG, matrix free (`QPA`, Gould et al.)
- Tailored: Exploit block structures from partial separability,
  e.g. condensing-type algorithms (Bock et al., 1984),
  Marc Steinbach's group, `qpHPSC` (K., 2010),
  `qpSchur` (Biegler et al.), `qpDUNES` (Frasch et al., next talk)

What some other active-set solvers do

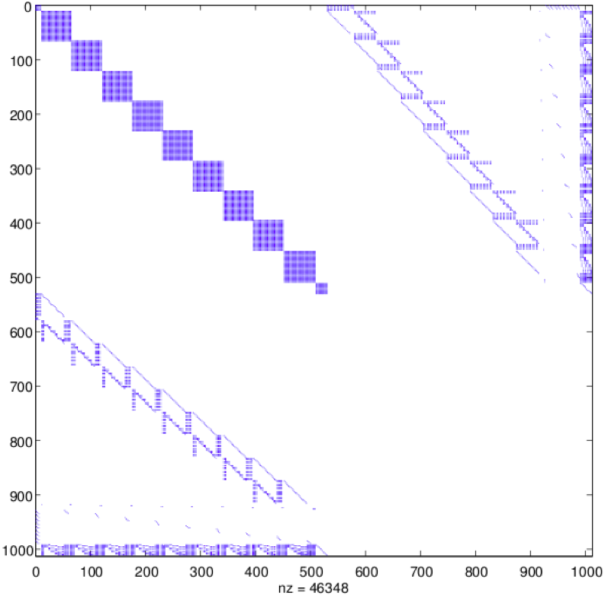
Universität Heidelberg
Zukunft. Seit 1386.

- Dense: Null-space method with updates (Gill et al., 70ies+80ies)
- Iterative: Preconditioned CG, matrix free (QPA, Gould et al.)
- Tailored: Exploit block structures from partial separability, e.g. condensing-type algorithms (Bock et al., 1984), Marc Steinbach's group, qpHPSC (K., 2010), qpSchur (Biegler et al.), qpDUNES (Frasch et al., next talk)

General-purpose:

- CPLEX, GuRoBi: Solve LP with complementarities using Wolfe's method (50ies !), can use sparse techniques for revised simplex
- bqpd: LIU and Fletcher-Matthews updates (Fletcher et al., 90ies)
- QPBLU(R), SQIC: sparse LDL^T and Schur complement updates (Saunders ≥2008, Gill, Wong, 2013)
- Andreas approached me with a similar implementation for qpOASES in late 2012

Typical Matrices from Control Problems



Sparse Factorizations for K

Possibilities:

- Sparse $K = LDL^T$
- Sparse $K = LU$
- Sparse $LK = U$

Codes:

- HSL MA27/57
- MA32/42, PARADISO, SuperLU, UMFPACK, WSMP

Advantages:

- Pivoting for numerical stability and structural sparsity
- Detection of rank deficiency
- More efficient memory accesses
- Vectorization opportunities, BLAS-3
- Parallelization opportunities: multi-core

Schur Complement Idea

$Kd = r$, have factorization of K.

Active set changes append border to KKT system matrix K:

$$\begin{bmatrix} K & M \\ M^T & N \end{bmatrix} \begin{bmatrix} d \\ p \end{bmatrix} = \begin{bmatrix} r \\ q \end{bmatrix}$$

$$\Longrightarrow \quad \begin{bmatrix} K & M \\ 0 & N - M^T K^{-1} M \end{bmatrix} \begin{bmatrix} d \\ p \end{bmatrix} = \begin{bmatrix} r \\ q - M^T K^{-1} r \end{bmatrix}$$

$$\Longrightarrow \quad \begin{bmatrix} K & M \\ 0 & S \end{bmatrix} \begin{bmatrix} d \\ p \end{bmatrix} = \begin{bmatrix} r \\ q - M^T K^{-1} r \end{bmatrix}$$

Two tasks:

- Need to explain how M, N, q are formed
- Need to maintain factorization of Schur complement S

For active-set QP: Gill, Murray, Saunders, Wright, 1987.

Add/Remove a Bound

Add a bound $i \in \mathcal{F}$ to $\mathcal{X}$: Express $\Delta x_i = \Delta l_i^{\mathrm{b}}$

$$\begin{bmatrix} H_{\mathcal{F}\mathcal{F}} & A_{\mathcal{A}\mathcal{F}}^T & e_i \\ A_{\mathcal{A}\mathcal{F}} & 0 & 0 \\ e_i^T & 0 & 0 \end{bmatrix} \begin{bmatrix} \Delta x_{\mathcal{F}} \\ -\Delta y_{\mathcal{A}} \\ -\Delta y_i \end{bmatrix} = \begin{bmatrix} -\Delta \tilde{g}_{\mathcal{F}} \\ \Delta l_{\mathcal{A}} \\ \Delta b_i \end{bmatrix}$$

$\Delta \tilde{g}_{\mathcal{F}}$ is $\Delta g_{\mathcal{F}}$ with i-th component set to zero.

Remove a bound $i \in \mathcal{X}$ to $\mathcal{F}$: x_i now free, system needs to grow

$$\begin{bmatrix} H_{\mathcal{F}\mathcal{F}} & A_{\mathcal{A}\mathcal{F}}^T & H_{\mathcal{F}i} \\ A_{\mathcal{A}\mathcal{F}} & 0 & A_{\mathcal{A}i} \\ H_{i\mathcal{F}} & A_{\mathcal{A}i}^T & H_{ii} \end{bmatrix} \begin{bmatrix} \Delta x_{\mathcal{F}} \\ -\Delta y_{\mathcal{A}} \\ \Delta x_i \end{bmatrix} = \begin{bmatrix} -\Delta \tilde{g}_{\mathcal{F}} \\ \Delta l_{\mathcal{A}} \\ -\Delta g_i \end{bmatrix}$$

Add/Remove a Constraint

Add a constraint $i \in \mathcal{I}$ to $\mathcal{A}$: Express $A_i \Delta x = \Delta l_i^A$

$$\begin{bmatrix} H_{\mathcal{F}\mathcal{F}} & A_{\mathcal{A}\mathcal{F}}^T & A_{i\mathcal{F}}^T \\ A_{\mathcal{A}\mathcal{F}} & 0 & 0 \\ A_{i\mathcal{F}} & 0 & 0 \end{bmatrix} \begin{bmatrix} \Delta x_{\mathcal{F}} \\ -\Delta y_{\mathcal{A}} \\ -\Delta y_i \end{bmatrix} = \begin{bmatrix} -\Delta g_{\mathcal{F}} \\ \Delta l_{\mathcal{A}} \\ \Delta b_i \end{bmatrix}$$

Remove a constraint $i \in \mathcal{A}$ to $\mathcal{I}$: Add a free constraint slack

$$\begin{bmatrix} H_{\mathcal{F}\mathcal{F}} & A_{\mathcal{A}\mathcal{F}}^T & 0 \\ A_{\mathcal{A}\mathcal{F}} & 0 & e_i \\ 0 & e_i^T & 0 \end{bmatrix} \begin{bmatrix} \Delta x_{\mathcal{F}} \\ -\Delta y_{\mathcal{A}} \\ s \end{bmatrix} = \begin{bmatrix} -\Delta \tilde{g}_{\mathcal{F}} \\ \Delta \tilde{l}_{\mathcal{A}} \\ 0 \end{bmatrix}$$

$\Delta \tilde{l}_{\mathcal{A}}$ is $\Delta l_{\mathcal{A}}$ with i-th component set to zero.

Update of Schur Complement

Assume

$$\bar{N} = \begin{bmatrix} N & n_1 \\ n_1^T & n_2 \end{bmatrix}, \quad \bar{M} = \begin{bmatrix} M & m \end{bmatrix}$$

Then the new Schur complement matrix $\bar{S}$ is

$$\bar{S} = \begin{bmatrix} M^T \\ m^T \end{bmatrix} S \begin{bmatrix} M & m \end{bmatrix} - \begin{bmatrix} N & n_1 \\ n_1^T & n_2 \end{bmatrix} = \begin{bmatrix} M^T K^{-1} M & M^T K^{-1} m \\ m^T K^{-1} M & m^T K^{-1} m \end{bmatrix} - \begin{bmatrix} N & n_1 \\ n_1^T & n_2 \end{bmatrix}$$

$$= \begin{bmatrix} S & M^T K^{-1} m - n_1 \\ m^T K^{-1} M - n_1^T & m^T K^{-1} m - n_2 \end{bmatrix}$$

Rank-1 updates to a factorization of S, e.g. dense QR.

Tradeoffs

Stability: Schur complement S may quickly become ill-conditioned.

Sparse solvers typically provide a condition number estimate obtained from the diagonal pivots. `qpOASES` refacctorizes if this exceeds a threshold `condMax`; we can go pretty high, e.g. $\texttt{condMax} = 10^{14}$

Tradeoffs

Stability: Schur complement S may quickly become ill-conditioned.

Sparse solvers typically provide a condition number estimate obtained from the diagonal pivots. `qpOASES` refacctorizes if this exceeds a threshold `condMax`; we can go pretty high, e.g. $\texttt{condMax} = 10^{14}$

Speed: Schur complement S grows in every iteration.

`qpOASES` refectorizes every `nSmax`, e.g. $\texttt{nSmax} = 100$, iterations $\Longrightarrow$ obtain new factors for new $K_{\mathcal{W}}$ and start with an empty border.

There is a tradeoff between working with the dense QR factor of S and computing a sparse factor of $K_{\mathcal{W}}$.

Guesses of the Working Set

Advantage of pivoting factorization: Can easily cope with rank-deficient A and wild guesses of $\mathcal{W}_0$

Opens up possibility to use crash strategies to obtain good initial guesses.

Guesses of the Working Set

Advantage of pivoting factorization: Can easily cope with rank-deficient A and wild guesses of $\mathcal{W}_0$

Opens up possibility to use crash strategies to obtain good initial guesses.

Assume $\mathcal{W}_0$ such that $K_{\mathcal{W}_0}$ is singular because $A_{\mathcal{AF}}$ has linearly dependent rows.

Then, a pivoting factorization will yield P, Q such that

$$P^T K_{\mathcal{W}_0} Q = \left[\begin{array}{c|c} K_{\bar{\mathcal{W}}_0} & I_{\mathcal{B}} \end{array}\right].$$

$\bar{\mathcal{W}}_0 \subset \mathcal{W}_0$ is a maximal linear independent subset of $\mathcal{W}_0$. The unassigned column pivots $\mathcal{B}$ are a minimal set of simple bounds to augment $\bar{\mathcal{W}}_0$.

Inertia Control

Can use `qpOASES` to find second-order critical points of indefinite QPs.

If $K_{\mathcal{W}_0}$ had correct inertia, then $In(S) = (\sigma_-, \sigma_+, 0)$ where σ_- and σ_+ are the numbers of constraints removed and added.

Can monitor inertia of S to monitor inertia of $K_{\mathcal{W}}$: Only need to look at sign changes of determinant.

Inertia Control

Can use `qpOASES` to find second-order critical points of indefinite QPs.

If $K_{\mathcal{W}_0}$ had correct inertia, then $In(S) = (\sigma_-, \sigma_+, 0)$ where σ_- and σ_+ are the numbers of constraints removed and added.

Can monitor inertia of S to monitor inertia of $K_{\mathcal{W}}$: Only need to look at sign changes of determinant.

In case of incorrect inertia of S, we can flip the side of the removed bound/constraint and add it again.

In case of incorrect inertia of $K_{\mathcal{W}_0}$ itself, a pivoting factorization will yield P, Q such that

$$P^T K_{\mathcal{W}_0} Q = \left[\begin{array}{c|c} K_{\bar{\mathcal{W}}_0} & I_{\mathcal{B}} \end{array} \right].$$

where the unassigned column pivots $\mathcal{B}$ are a minimal set of simple bounds to augment $\bar{\mathcal{W}}_0$ such that correct inertia is obtained.

Things to to

- Internal representation of working set slightly involved to work with
- Apparently slows down some sparse vector operations
- `qpOASES` cold startup currently somewhat sluggish. Crashing strategies for guessing the initial working set will cure this.

  But remember that `qpOASES`' strength is solving sequences of problems!
- Thorough comparisons with dense qpOASES and sparse QP codes such as `bqpd` and `SQIC`

  (QP Benchmark Collection will be very helpful)

Sparse qpOASES in ongoing projects

- Moving to **COIN-OR**:

  A new release of qpOASES including the sparse variant will join the COIN-OR initiative.
  `http://www.coin-or.org`

- **iQP**: T.C. Johnson, C. Kirches, A. Wächter. An Active-Set Quadratic Programming Method Based On Sequential Hot-Starts.
  `http://www.optimization-online.org/DB_HTML/2013/10/4084.html`
  (submitted to SIAM J. Opt.)

- Optimum Experimental Design for dynamic processes:
  Sparse Block-SR1 SQP code for NLPs coming from dynamic OED problems.
  Part of Dennis' thesis. Paper to be submitted to Comp. Optim. Appl. or Math. Prog. Comput.

Sparse qpOASES in ongoing projects

- Moving to **COIN-OR**:

  A new release of qpOASES … parse variant will join the C…
  `http://www.coi…`

- **iQP**: T.C. Joh… Set Quadratic Programming …
  `http://www.opt… 4084.html`
  (submitted to SIA…

- Optimum Experim… ses:
  Sparse Block-SR1 S… rom dynamic OED problems.
  Part of Dennis' thesis. Po… itted to Comp. Optim. Appl. or Math. Prog. Comput.

Thank you very much for your attention!

Christian Kirches

Interdisciplinary Center for Scientific Computing (IWR)
Heidelberg University

mail Im Neuenheimer Feld 368
69120 Heidelberg, Germany
phone +49 6221 54-8812
e-mail christian.kirches@iwr.uni-heidelberg.de