A computational experience with search directions in Interior Point Methods for Linear Programming

Marco Colombo Jacek Gondzio

School of Mathematics
The University of Edinburgh

Dundee NA 2005

Linear programming and optimality conditions

- Linear programming problem and KKT conditions

$$\begin{array}{rl} \min & c^T x \\ \text{s.t.} & Ax = b \\ & x \geq 0 \end{array} \qquad \begin{bmatrix} Ax - b \\ A^T y + s - c \\ XSe \end{bmatrix} = 0 \quad x, s \geq 0$$

- Perturb the complementarity conditions

$$XSe = \mu e$$

- Solve the perturbed KKT conditions with Newton's method

$$\begin{bmatrix} A & 0 & 0 \\ 0 & A^T & I \\ S & 0 & X \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta s \end{bmatrix} = \begin{bmatrix} b - Ax \\ c - A^T y - s \\ -XSe + \mu e \end{bmatrix}$$

Mehrotra's algorithm: Predictor direction

- Exploit linearity: solve independently for two right-hand sides

$$\begin{bmatrix} A & 0 & 0 \\ 0 & A^T & I \\ S & 0 & X \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta s \end{bmatrix} = \underbrace{\begin{bmatrix} b - Ax \\ c - A^T y - s \\ -XSe \end{bmatrix}}_{predictor} + \underbrace{\begin{bmatrix} 0 \\ 0 \\ \mu e \end{bmatrix}}_{corrector}$$

- Predictor: set $\mu = 0$ and solve for the direction Δ_a
- Evaluate the allowed stepsizes

$$\alpha_p = \max_{\alpha} : x + \alpha \Delta_a x \geq 0 \qquad \alpha_d = \max_{\alpha} : s + \alpha \Delta_a s \geq 0$$

- Predicted gap: $g_a = (x + \alpha_p \Delta_a x)^T (s + \alpha_d \Delta_a s)$
- Use the predicted gap to estimate the centering term

$$\mu = \left(\frac{g_a}{x^T s} \right)^3 \frac{x^T s}{n}$$

Mehrotra's algorithm: Corrector direction

- Error in taking a full step in the predictor

$$(X+\Delta_a X)(S+\Delta_a S)e = XSe + \underbrace{(S\Delta_a x + X\Delta_a s)}_{-XSe} + \Delta_a X \Delta_a S e$$

- Consider a second order term and solve for Δ_c

$$\begin{bmatrix} A & 0 & 0 \\ 0 & A^T & I \\ S & 0 & X \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta s \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \mu e - \Delta_a X \Delta_a S e \end{bmatrix}$$

- Find the stepsizes in the combined direction $\Delta = \Delta_a + \Delta_c$
- Usually the stepsizes are much better than those obtained by the predictor

Multiple centrality correctors

- Try to increase the stepsizes: $\tilde{\alpha} = \min(\alpha + \delta, 1)$
- Move the trial point in the neighbourhood

$$\tilde{v}_i = x_i(\tilde{\alpha}) s_i(\tilde{\alpha}) \in \mathcal{N}_s(\gamma) = \{x_i s_i : \gamma\mu \le x_i s_i \le \frac{1}{\gamma}\mu\}$$

- Define an achievable target

$$t_i = \begin{cases} 0 & \text{if } \tilde{v}_i \in [\gamma\mu, \frac{1}{\gamma}\mu] \\ \gamma\mu - \tilde{v}_i & \text{if } \tilde{v}_i < \gamma\mu \\ \frac{1}{\gamma}\mu - \tilde{v}_i & \text{if } \tilde{v}_i > \frac{1}{\gamma}\mu \end{cases} \qquad rhs = \begin{bmatrix} 0 \\ 0 \\ t \end{bmatrix}$$

- The number of correctors allowed depends on

$$\frac{\textit{Factorization effort}}{\textit{Backsolve effort}}$$

- A corrector is accepted if $\hat{\alpha} \ge \alpha + \rho\delta$

Practical pitfalls

- The stepsizes in the affine-scaling direction can be very short, especially if the point is badly centered
- Mehrotra's corrector is computed on the basis of full step in affine scaling direction
- Sometimes the magnitude of the corrector is much larger than the magnitude of the predictor (Cartis, 2005)
- Short steps in the combined direction may be produced

Weighting the corrector direction

- Cartis (2005) suggests weighting the corrector by α^2, based on a quadratic approximation of a local path from the current point to a target on the central path (PDSOC)
- Generalize Mehrotra's scheme

$$\Delta^{\omega} = \Delta_a + \omega \Delta_c$$

- Salahi, Peng and Terlaky (2005) propose $\omega = \alpha$ and safeguards based on centering
- Computational study on the weight of the corrector

Finding the best weight

- Find the corrector direction Δ_c

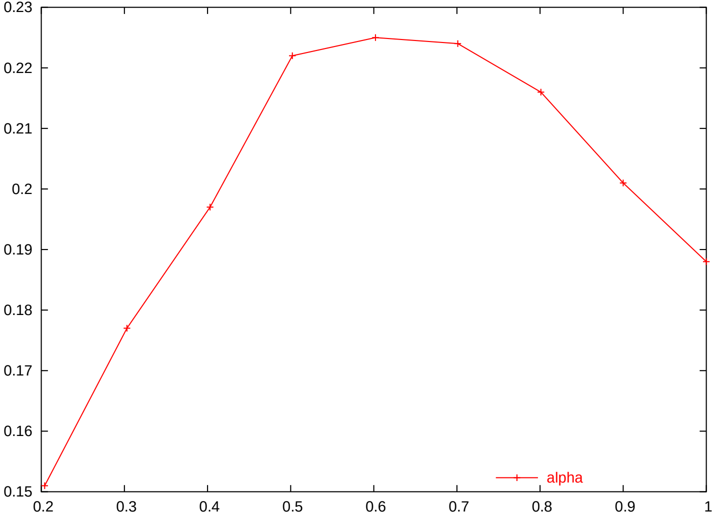
$$\begin{bmatrix} A & 0 & 0 \\ 0 & A^T & I \\ S & 0 & X \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta s \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \mu e - \Delta_a X \Delta_a S e \end{bmatrix}$$

- Set $\omega \in [\alpha_p \alpha_d, 1]$ and compute

$$\Delta^\omega = \Delta_a + \omega \Delta_c$$

- Do a linesearch to find the optimal $\hat{\omega}$ that maximises the product of the stepsizes $\alpha_p^\omega \alpha_d^\omega$ in Δ^ω

Omega and alpha



Computational experience

- Results from a collection of 100 problems (Netlib and Kennington)
- Initial comparison between PCx code (Czyzyk, Mehrotra, Wright) and HOPDM code (Gondzio)
- Different linear algebra in PCx and HOPDM accounts for different choices concerning multiple centrality correctors
- Difference in termination criteria: implemented in HOPDM the criteria used in PCx
- Analysis done on number of iterations and number of backsolves

Computational results I

	PCx	HOPDM	HOPDM-ω	Change
Iterations	2086	1808	1749	-3%
Backsolves	5542	5547	5789	+5%
Backsolves/iter.	2.66	3.07	3.31	+8%

- Decrease in iteration count but increase in backsolves
- We are accepting more multiple centrality correctors than normally we would do
- Ask for more stepsize increase in multiple centrality correctors

Computational results II

- Use more aggressive centrality correctors

$$\hat{\delta} = 3\delta$$

	HOPDM+	HOPDM-ω+	Change	Total
Iterations	1765	1613	-8%	-10%
Backsolves	5132	5286	+3%	-4%
Backsolves/iter.	2.91	3.28	+13%	+7%

Future work

- Clarify the usefulness of ω
- Are there heuristics to (approximately) localise $\hat{\omega}$?
- Choice of settings in multiple centrality correctors