

A first-order analysis of a trust-region algorithm with inexact Jacobians

Andrea Walther
Institute of Scientific Computing
TU Dresden

8. US-Mexico Workshop on Optimization
January 11, 2007

Overview

1. Motivation
2. Trust-region algorithm with inexact Jacobians
 - (a) General observations
 - (b) Algorithm
 - (c) Convergence result
3. Numerical results
4. Conclusion and outlook

1. Motivation

Periodic adsorption processes



Joint work with Larry Biegler

Periodic adsorption processes:

- separation of components, e.g. H_2 from furnace gases
- operate at cyclic steady-state \Rightarrow CSS
- maximize overall recovery at desired purity, . . .

Periodic adsorption processes:

- separation of components, e.g. H_2 from furnace gases
- operate at cyclic steady-state \Rightarrow CSS
- maximize overall recovery at desired purity, . . .
- can be modeled by:

$$\begin{aligned}
 & \min && \phi(y(t_f)) \\
 \text{s.t.} & \dot{z}_1 &= f_1(y_1, p, t), & y_1(t_0) = y_0, t \in [t_0, t_1] \\
 & \dot{z}_i &= f_i(y_i, p, t), & y_i(t_{i-1}) = y_{i-1}(t_{i-1}), \\
 & & & t \in [t_{i-1}, t_i], i \in [2, N] \\
 & C(y_0) &= y_0 - y_N(t_N) = 0 \\
 & 0 &\geq W(y(t), y_0, p)
 \end{aligned}$$

Task: Min $f(x)$ subject to $c(x) = 0$,

with nonlinear and sufficiently smooth

- $f : \mathbb{R}^N \rightarrow \mathbb{R}$ objective,
- $c : \mathbb{R}^N \rightarrow \mathbb{R}^M$ state constraints (CSS !)
- x state variables and control variables

Task: Min $f(x)$ subject to $c(x) = 0$,

with nonlinear and sufficiently smooth

- $f : \mathbb{R}^N \rightarrow \mathbb{R}$ objective,
- $c : \mathbb{R}^N \rightarrow \mathbb{R}^M$ state constraints (CSS !)
- x state variables and control variables

Calculus-based methods:

- Consider Lagrangian $\mathcal{L}(x, \lambda) = f(x) + \lambda^T c(x)$
- Solve

$$0 = [g(x, \lambda), c(x)] \equiv [\nabla_x f(x) + \lambda^T \nabla_x c(x), c(x)]$$

Apply SQP-like method, i.e. solve KKT system

$$\nabla_{x,\lambda}^2 \mathcal{L}(x_k, \lambda_k) p_k^N = \begin{bmatrix} B(x_k, \lambda_k) & A(x_k)^T \\ A(x_k) & 0 \end{bmatrix} p_k^N = -\nabla_{x,\lambda} \mathcal{L}(x_k, \lambda_k)$$

Apply SQP-like method, i.e. solve KKT system

$$\nabla_{x,\lambda}^2 \mathcal{L}(x_k, \lambda_k) p_k^N = \begin{bmatrix} B(x_k, \lambda_k) & A(x_k)^T \\ A(x_k) & 0 \end{bmatrix} p_k^N = -\nabla_{x,\lambda} \mathcal{L}(x_k, \lambda_k)$$

Critical point: Jacobian of constraints $A(x)$

$A(x)$ dense \Rightarrow computation and factorization very expensive

- Freeze exact (factorized) Jacobian for several steps
- Use only approximation of Jacobian

Apply SQP-like method, i.e. solve KKT system

$$\nabla_{x,\lambda}^2 \mathcal{L}(x_k, \lambda_k) p_k^N = \begin{bmatrix} B(x_k, \lambda_k) & A(x_k)^T \\ A(x_k) & 0 \end{bmatrix} p_k^N = -\nabla_{x,\lambda} \mathcal{L}(x_k, \lambda_k)$$

Critical point: Jacobian of constraints $A(x)$

$A(x)$ dense \Rightarrow computation and factorization very expensive

- Freeze exact (factorized) Jacobian for several steps
- Use only approximation of Jacobian

Assumptions: One can evaluate

Apply SQP-like method, i.e. solve KKT system

$$\nabla_{x,\lambda}^2 \mathcal{L}(x_k, \lambda_k) p_k^N = \begin{bmatrix} B(x_k, \lambda_k) & A(x_k)^T \\ A(x_k) & 0 \end{bmatrix} p_k^N = -\nabla_{x,\lambda} \mathcal{L}(x_k, \lambda_k)$$

Critical point: Jacobian of constraints $A(x)$

$A(x)$ dense \Rightarrow computation and factorization very expensive

- Freeze exact (factorized) Jacobian for several steps
- Use only approximation of Jacobian

Assumptions: One can evaluate

- $A(x_k)v$
- $A(x_k)^T v$

Apply SQP-like method, i.e. solve KKT system

$$\nabla_{x,\lambda}^2 \mathcal{L}(x_k, \lambda_k) p_k^N = \begin{bmatrix} B(x_k, \lambda_k) & A(x_k)^T \\ A(x_k) & 0 \end{bmatrix} p_k^N = -\nabla_{x,\lambda} \mathcal{L}(x_k, \lambda_k)$$

Critical point: Jacobian of constraints $A(x)$

$A(x)$ dense \Rightarrow computation and factorization very expensive

- Freeze exact (factorized) Jacobian for several steps
- Use only approximation of Jacobian

Assumptions: One can evaluate

- $A(x)v \Rightarrow$ sensitivity equation, forward mode AD
- $A(x)^T v$

Apply SQP-like method, i.e. solve KKT system

$$\nabla_{x,\lambda}^2 \mathcal{L}(x_k, \lambda_k) p_k^N = \begin{bmatrix} B(x_k, \lambda_k) & A(x_k)^T \\ A(x_k) & 0 \end{bmatrix} p_k^N = -\nabla_{x,\lambda} \mathcal{L}(x_k, \lambda_k)$$

Critical point: Jacobian of constraints $A(x)$

$A(x)$ dense \Rightarrow computation and factorization very expensive

- Freeze exact (factorized) Jacobian for several steps
- Use only approximation of Jacobian

Assumptions: One can evaluate

- $A(x)v \Rightarrow$ sensitivity equation, forward mode AD
- $A(x)^T v \Rightarrow$ adjoint equation, reverse mode AD

Apply SQP-like method, i.e. solve KKT system

$$\nabla_{x,\lambda}^2 \mathcal{L}(x_k, \lambda_k) p_k^N = \begin{bmatrix} B(x_k, \lambda_k) & A(x_k)^T \\ A(x_k) & 0 \end{bmatrix} p_k^N = -\nabla_{x,\lambda} \mathcal{L}(x_k, \lambda_k)$$

Critical point: Jacobian of constraints $A(x)$

$A(x)$ dense \Rightarrow computation and factorization very expensive

- Freeze exact (factorized) Jacobian for several steps
- Use only approximation of Jacobian

Assumptions: One can evaluate

- $A(x)v \Rightarrow$ sensitivity equation, forward mode AD
- $A(x)^T v \Rightarrow$ adjoint equation, reverse mode AD

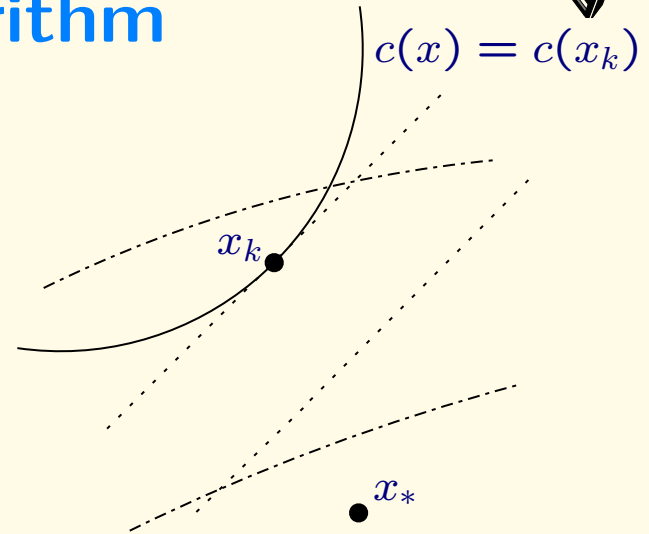
and approximates Jacobian A_k and null space Z_k with $A_k Z_k = 0$

2. Trust-region algorithm

Composite-step trust region:

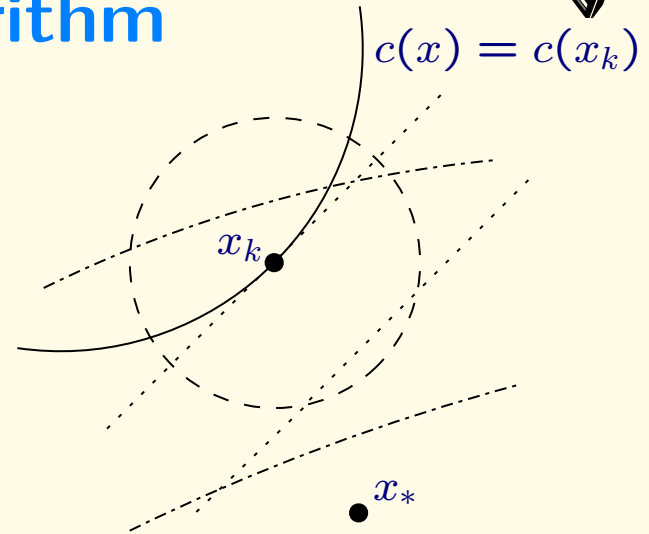
2. Trust-region algorithm

Composite-step trust region:



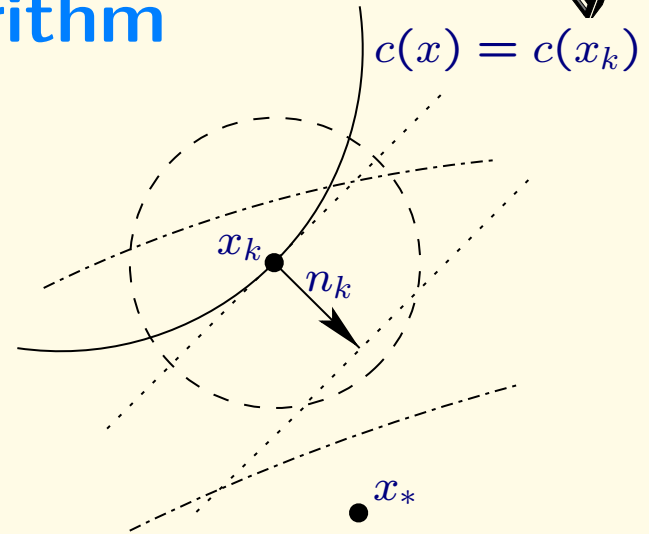
2. Trust-region algorithm

Composite-step trust region:



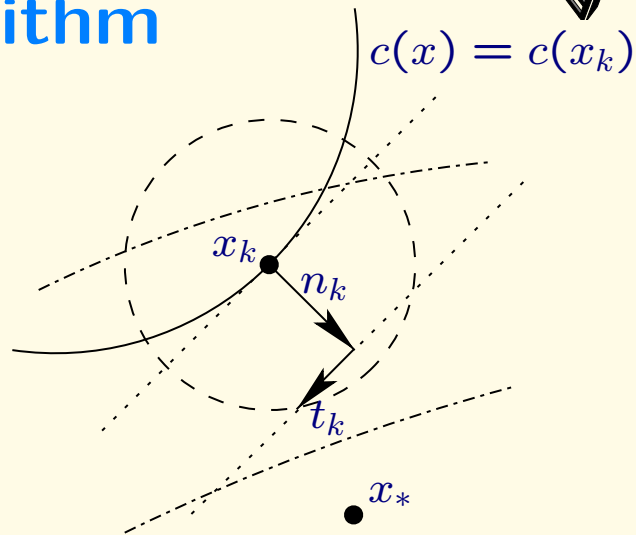
2. Trust-region algorithm

Composite-step trust region:



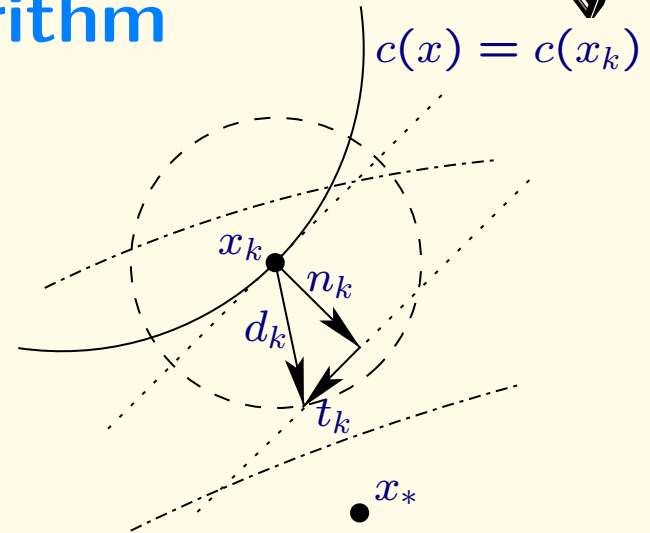
2. Trust-region algorithm

Composite-step trust region:



2. Trust-region algorithm

Composite-step trust region:

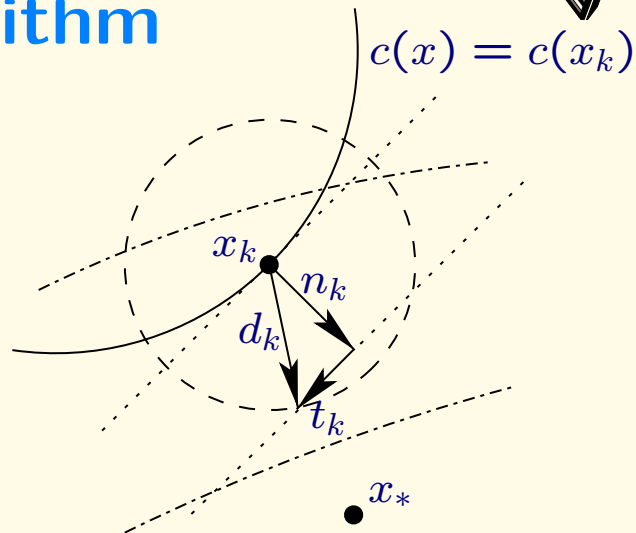


2. Trust-region algorithm

Composite-step trust region:

Normal subproblem

$$\begin{aligned} \min_{n \in \mathbb{R}^N} & \|c(x_k) + A(x_k)n\|^2 \\ \text{s.t.} & \|n\| \leq \tilde{\Delta}_k \end{aligned}$$



2. Trust-region algorithm

Composite-step trust region:

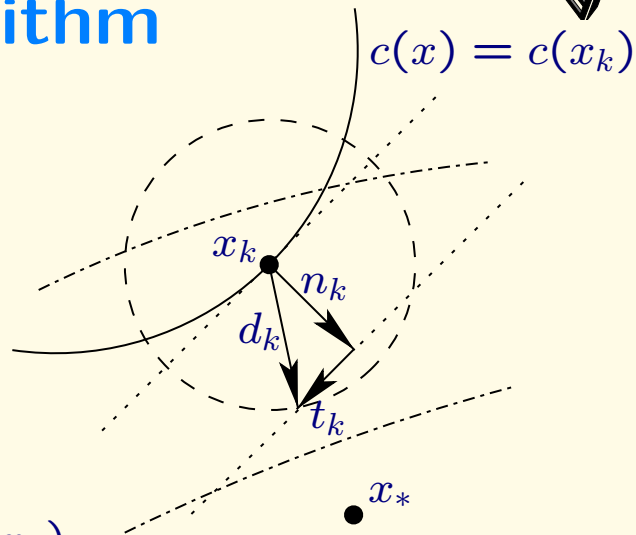
Normal subproblem

$$\begin{aligned} \min_{n \in \mathbb{R}^N} & \|c(x_k) + A(x_k)n\|^2 \\ \text{s.t.} & \|n\| \leq \tilde{\Delta}_k \end{aligned}$$

Cauchy step: $n_k^C = -\alpha_k^C A(x_k)^T c(x_k)$

where α_k^C solution of

$$\min_{\alpha \geq 0} \|c(x_k) - \alpha A(x_k) A(x_k)^T c(x_k)\| \quad \text{s.t.} \quad \|\alpha A(x_k)^T c(x_k)\| \leq \tilde{\Delta}_k$$



2. Trust-region algorithm

Composite-step trust region:

Normal subproblem

$$\begin{aligned} \min_{n \in \mathbb{R}^N} & \|c(x_k) + A(x_k)n\|^2 \\ \text{s.t.} & \|n\| \leq \tilde{\Delta}_k \end{aligned}$$

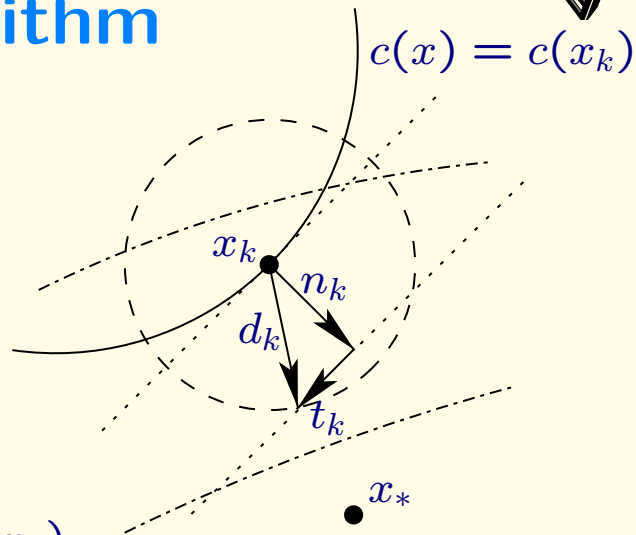
Cauchy step: $n_k^C = -\alpha_k^C A(x_k)^T c(x_k)$

where α_k^C solution of

$$\min_{\alpha \geq 0} \|c(x_k) - \alpha A(x_k) A(x_k)^T c(x_k)\| \quad \text{s.t.} \quad \|\alpha A(x_k)^T c(x_k)\| \leq \tilde{\Delta}_k$$

⇒ Computation of **exact** Cauchy step possible!!

Ensures that *Normal Cauchy Decrease Condition* holds



Tangential subproblem

$$\begin{aligned} \min_{p \in \mathbb{R}^{N-M}} & (\nabla f(x_k) + B_k n_k)^T Z_k p + \frac{1}{2} p^T Z_k^T B_k Z_k p \\ \text{s.t.} & \|Z_k p\| \leq \hat{\Delta}_k \end{aligned}$$

Tangential subproblem

$$\begin{aligned} \min_{p \in \mathbb{R}^{N-M}} & (\nabla f(x_k) + B_k n_k)^T Z_k p + \frac{1}{2} p^T Z_k^T B_k Z_k p \\ \text{s.t.} & \|Z_k p\| \leq \hat{\Delta}_k \end{aligned}$$

Cauchy step t_k^C :

$$p_k^C = Z_k^T (\nabla f(x_k) + B_k n_k) \quad \Rightarrow$$

$$t_k^C = -\theta_k^C Z_k p_k^C$$

where θ_k^C solution of

$$\begin{aligned} \min_{\theta \geq 0} & (\nabla f(x_k) + B_k n_k)^T \theta Z_k p_k^C + \frac{\theta^2}{2} (p_k^C)^T Z_k^T B_k Z_k p_k^C \\ \text{s.t.} & \|\theta Z_k p_k^C\| \leq \hat{\Delta}_k \end{aligned}$$

Tangential subproblem

$$\begin{aligned} \min_{p \in \mathbb{R}^{N-M}} & (\nabla f(x_k) + B_k n_k)^T Z_k p + \frac{1}{2} p^T Z_k^T B_k Z_k p \\ \text{s.t.} & \|Z_k p\| \leq \hat{\Delta}_k \end{aligned}$$

Cauchy step t_k^C :

$$\begin{aligned} p_k^C &= Z_k^T (\nabla f(x_k) + B_k n_k) \quad \Rightarrow \\ t_k^C &= -\theta_k^C Z_k p_k^C \end{aligned}$$

where θ_k^C solution of

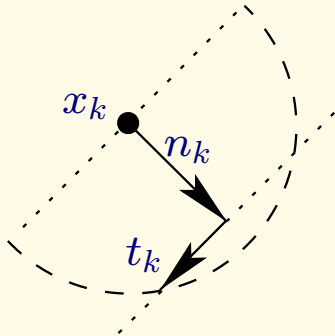
$$\begin{aligned} \min_{\theta \geq 0} & (\nabla f(x_k) + B_k n_k)^T \theta Z_k p_k^C + \frac{\theta^2}{2} (p_k^C)^T Z_k^T B_k Z_k p_k^C \\ \text{s.t.} & \|\theta Z_k p_k^C\| \leq \hat{\Delta}_k \end{aligned}$$

\Rightarrow Cauchy step for **inexact** subproblem available, but

Inexactness:

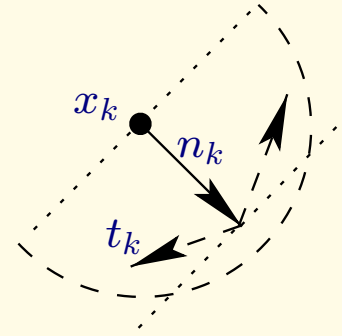
$Z_k = \text{exact null space}$

$$A(x_k)(n_k + t_k) = A(x_k)n_k$$



$Z_k = \text{inexact null space}$

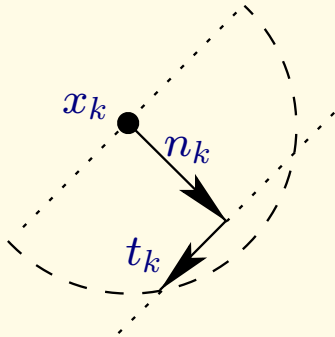
$$A(x_k)(n_k + t_k) = A(x_k)n_k + ?$$



Inexactness:

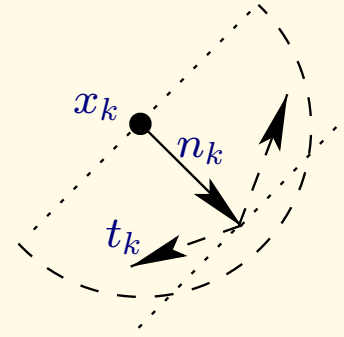
$Z_k =$ exact null space

$$A(x_k)(n_k + t_k) = A(x_k)n_k$$



$Z_k =$ inexact null space

$$A(x_k)(n_k + t_k) = A(x_k)n_k + ?$$

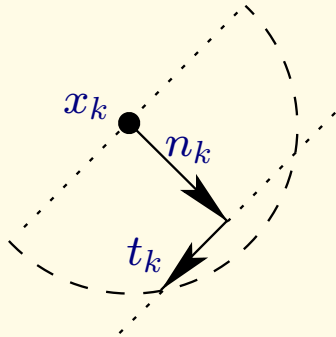


$$\begin{aligned} \text{pred}_k(d_k) &= \text{hpred}(t_k) + \mu_k \text{vpred}(n_k) + \chi_k + \text{err}_k(d_k, \mu_k) \\ &= \text{ipred}_k(d_k) + \text{err}_k(d_k, \mu_k) \end{aligned}$$

Inexactness:

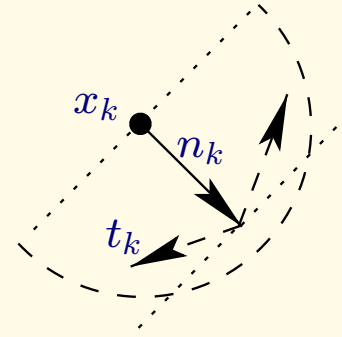
$Z_k =$ exact null space

$$A(x_k)(n_k + t_k) = A(x_k)n_k$$



$Z_k =$ inexact null space

$$A(x_k)(n_k + t_k) = A(x_k)n_k + ?$$



$$\begin{aligned} \text{pred}_k(d_k) &= \text{hpred}(t_k) + \mu_k \text{vpred}(n_k) + \chi_k + \text{err}_k(d_k, \mu_k) \\ &= \text{ipred}_k(d_k) + \text{err}_k(d_k, \mu_k) \end{aligned}$$

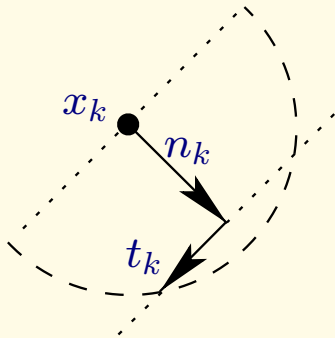
Need bound on deviation: For $\nu > 0$ $\|A(x_k)t_k\| \leq \nu \Delta_k \Rightarrow$

$$\begin{aligned} |\text{err}_k(d_k, \mu_k)| &= \mu_k \left| \|c(x_k) + A(x_k)n_k\| - \|c(x_k) + A(x_k)d_k\| \right| \\ &\leq \mu_k \|A(x_k)t_k\| \leq \mu_k \nu \Delta_k \end{aligned}$$

Inexactness:

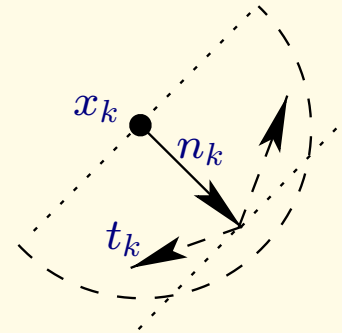
$Z_k = \text{exact null space}$

$$A(x_k)(n_k + t_k) = A(x_k)n_k$$



$Z_k = \text{inexact null space}$

$$A(x_k)(n_k + t_k) = A(x_k)n_k + ?$$



$$\begin{aligned} \text{pred}_k(d_k) &= \text{hpred}(t_k) + \mu_k \text{vpred}(n_k) + \chi_k + \text{err}_k(d_k, \mu_k) \\ &= \text{ipred}_k(d_k) + \text{err}_k(d_k, \mu_k) \end{aligned}$$

Alternative:

$$-\text{err}_k(d_k, \mu_k) \leq \eta \text{ipred}_k(d_k), \quad \eta \in (0, 1)$$

$$\Rightarrow \text{pred}_k(d_k) > 0 !!$$

Lagrange Multipliers

Exact:

$$\lambda_{k+1} = (A(x_{k+1})A(x_{k+1})^T)^{-1}A(x_{k+1})\nabla f(x_{k+1})$$

Lagrange Multipliers

Exact:

$$\lambda_{k+1} = (A(x_{k+1})A(x_{k+1})^T)^{-1}A(x_{k+1})\nabla f(x_{k+1})$$

Inexact:

$$\lambda_{k+1} = (A_{k+1}A_{k+1}^T)^{-1}A_{k+1}\nabla f(x_{k+1})$$

Lagrange Multipliers

Exact:

$$\lambda_{k+1} = (A(x_{k+1})A(x_{k+1})^T)^{-1}A(x_{k+1})\nabla f(x_{k+1})$$

Inexact:

$$\lambda_{k+1} = (A_{k+1}A_{k+1}^T)^{-1}A_{k+1}\nabla f(x_{k+1})$$

Once more: Bound on inexactness of Z_{k+1} with $\omega \in (0, \frac{1}{2})$

$$\|Z_{k+1}^T A(x_{k+1})^T \lambda_{k+1}\| \leq \omega \|Z_{k+1}^T \nabla f(x_{k+1})\|$$

to ensure global convergence to first-order critical points

Algorithm

Start: Init $x_0, \lambda_0, \mu_{-1} > 0, A_0, Z_0, \Delta_0, \eta, \rho \in (0, 1), \omega \in (0, \frac{1}{2}), \nu > 0$
for $k = 0, 1, \dots$

Algorithm

Start: Init $x_0, \lambda_0, \mu_{-1} > 0, A_0, Z_0, \Delta_0, \eta, \rho \in (0, 1), \omega \in (0, \frac{1}{2}), \nu > 0$

for $k = 0, 1, \dots$

1. Compute total step $d_k = n_k + t_k$ as described before
2. Compute $\mu_k \geq \mu_{k-1}$ such that $\text{ipred}_k(d_k) \geq \rho \mu_k \nu \text{pred}_k(n_k)$
3. If $\text{ared}_k(d_k) < \eta \text{ipred}_k(d_k)$ decrease Δ_k , go to 1.
4. Set $x_{k+1} = x_k + d_k$ and update Δ_k .
5. Compute new A_{k+1}, Z_{k+1} , and Lagrange multipliers λ_{k+1}
6. If $-Z(x_{k+1})^T \nabla f(x_{k+1}) = 0, c(x_{k+1}) = 0$, stop
else $k = k + 1$, go to 1.

Algorithm

Start: Init $x_0, \lambda_0, \mu_{-1} > 0, A_0, Z_0, \Delta_0, \eta, \rho \in (0, 1), \omega \in (0, \frac{1}{2}), \nu > 0$

for $k = 0, 1, \dots$

1. Compute total step $d_k = n_k + t_k$ as described before
2. Compute $\mu_k \geq \mu_{k-1}$ such that $\text{ipred}_k(d_k) \geq \rho \mu_k \nu \text{pred}_k(n_k)$
3. If $-\text{err}_k(d_k) < \tilde{\eta} \text{ipred}_k(d_k)$, improve A_k and go to 1.
4. If $\text{ared}_k(d_k) < \eta \text{ipred}_k(d_k)$ decrease Δ_k , go to 1.
5. Set $x_{k+1} = x_k + d_k$ and update Δ_k .
6. Compute new A_{k+1}, Z_{k+1} , and Lagrange multipliers λ_{k+1} such that $\|Z_{k+1}^T A(x_{k+1})^T \lambda_{k+1}\| \leq \omega \|Z_{k+1}^T \nabla f(x_{k+1})\|$.
7. If $-Z_{k+1}^T \nabla f(x_{k+1}) = 0, c(x_{k+1}) = 0$, go to 6
else $k = k + 1$, go to 1.

Theorem: Well-posedness,
i.e. Δ_k cannot shrink to zero if x_k is not a stationary point

Proof: Exploiting

$$\|A(x_k)t_k\| \leq \nu\Delta_k \quad \text{or} \quad -\text{err}_k(d_k) < \tilde{\eta} \text{ipred}_k(d_k) \quad (1)$$

yields

$$|\text{ipred}_k(d_k) - \text{ared}_k(d_k)| \leq \gamma(1 + \mu_k)\Delta_k^2 \quad (2)$$

The remaining proof follows Byrd, Gilbert, Nocedal [2000].
For details see W. [2005]

Theorem: Well-posedness,
i.e. Δ_k cannot shrink to zero if x_k is not a stationary point

Proof: Exploiting

$$\|A(x_k)t_k\| \leq \nu\Delta_k \quad \text{or} \quad -\text{err}_k(d_k) < \tilde{\eta} \text{ipred}_k(d_k) \quad (1)$$

yields

$$|\text{ipred}_k(d_k) - \text{ared}_k(d_k)| \leq \gamma(1 + \mu_k)\Delta_k^2 \quad (2)$$

The remaining proof follows Byrd, Gilbert, Nocedal [2000].
For details see W. [2005]

Theorem: Feasibility of all limit points, i.e. $\lim_{k \rightarrow \infty} c(x_k) = 0$.

Proof: Based on (2), for details see W. [2005].

Requirements on inexactness: Up to now only (1) needed !!

Theorem: All limit points are first-order critical, i.e.

$$\lim_{k \rightarrow \infty} p_k = \lim_{k \rightarrow \infty} (\nabla f(x_k) + A(x_k)^T \lambda_k) = 0$$

Proof: Using $\varrho = \omega / (1 - \omega) \in (0, 1)$ and

$$\|Z_{k+1}^T A(x_{k+1})^T \lambda_{k+1}\| \leq \omega \|Z_{k+1}^T \nabla f(x_{k+1})\| \quad (3)$$

$$\begin{aligned} \Rightarrow \|Z_k^T p_k\| &= \|Z_k^T (\nabla f(x_k) + A(x_k)^T \lambda_k)\| \\ &\geq (1 - \omega) \|Z_k^T \nabla f(x_k)\| \geq \frac{1 - \omega}{\omega} \|Z_k^T A(x_k)^T \lambda_k\|. \end{aligned}$$

$$\begin{aligned} \Rightarrow \|p_k^C\| &= \|-Z_k^T (\nabla f(x_k) + B_k n_k)\| \\ &\geq \gamma_2 \|p_k\| - \varrho \gamma_2 \|p_k\| - \gamma_3 \|n_k\| = (1 - \varrho) \gamma_2 \|p_k\| - \gamma_3 \|n_k\| \end{aligned}$$

Theorem: All limit points are first-order critical, i.e.

$$\lim_{k \rightarrow \infty} p_k = \lim_{k \rightarrow \infty} (\nabla f(x_k) + A(x_k)^T \lambda_k) = 0$$

Proof: Using $\varrho = \omega / (1 - \omega) \in (0, 1)$ and

$$\|Z_{k+1}^T A(x_{k+1})^T \lambda_{k+1}\| \leq \omega \|Z_{k+1}^T \nabla f(x_{k+1})\| \quad (3)$$

$$\begin{aligned} \Rightarrow \|Z_k^T p_k\| &= \|Z_k^T (\nabla f(x_k) + A(x_k)^T \lambda_k)\| \\ &\geq (1 - \omega) \|Z_k^T \nabla f(x_k)\| \geq \frac{1 - \omega}{\omega} \|Z_k^T A(x_k)^T \lambda_k\|. \end{aligned}$$

$$\begin{aligned} \Rightarrow \|p_k^C\| &= \|-Z_k^T (\nabla f(x_k) + B_k n_k)\| \\ &\geq \gamma_2 \|p_k\| - \varrho \gamma_2 \|p_k\| - \gamma_3 \|n_k\| = (1 - \varrho) \gamma_2 \|p_k\| - \gamma_3 \|n_k\| \end{aligned}$$

Remaining proof similar to Byrd, Gilbert, Nocedal [2000], for details see W. [2005]

Requirements on inexactness: Here only (3) needed !!

3. Numerical results

Implementation:

- C/C++ program
- AD-tool ADOL-C to compute $A(x)v$, $A(x)^T v$, $B(x)v$
- TR1 update to approximate $A(x)$, $Z(x)$

3. Numerical results

Implementation:

- C/C++ program
- AD-tool ADOL-C to compute $A(x)v$, $A(x)^T v$, $B(x)v$
- TR1 update to approximate $A(x)$, $Z(x)$

Test Problems:

- CUTER collection
- small SMB problem

Results for small CUTEr problems

problem	iter. count		evaluation of full $A(x_k)$			
	exact	inexact	exact	err	Lagr	sum+1
bt5	45	45	39	1	0	2
bt6	12	14	13	9	2	12
bt7	230	231	219	31	9	41
bt8	9	9	10	0	0	1
hs7	11	11	12	0	0	1
hs8	5	65	6	0	0	1
hs9	3	3	4	0	0	1
hs26	14	14	15	0	0	1
hs27	13	13	14	1	1	3
hs40	12	14	8	2	1	4
hs42	27	27	22	0	3	4
hs77	12	12	10	7	2	10
hs78	8	19	6	7	0	8
hs79	4	4	5	3	0	4

Dimension of larger problems

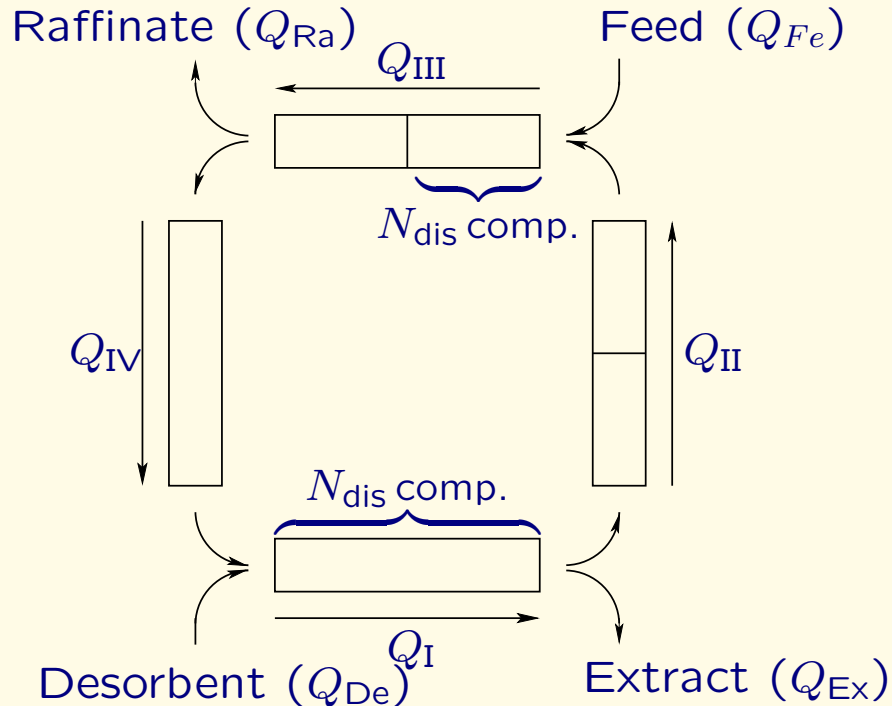
Problem	n	m	objective	constraints	$f(x_*)$
genhs28	300	298	nonlinear	linear	$3.31481 * 10^1$
orthregc	1005	500	nonlinear	nonlinear	$1.87906 * 10^1$
dtoc1nd	2998	2000	nonlinear	nonlinear	$2.37703 * 10^1$
dtoc2	2998	2000	nonlinear	nonlinear	$4.9723 * 10^{-1}$
dtoc3	2999	1998	nonlinear	linear	$2.3518 * 10^2$
dtoc4	2999	1998	nonlinear	nonlinear	$2.8748 * 10^0$
dtoc5	1999	999	nonlinear	nonlinear	$1.5274 * 10^0$
dtoc6	2001	1001	nonlinear	nonlinear	$1.7176 * 10^4$
eigena2	110	55	nonlinear	nonlinear	$0.0000 * 10^0$
eigenc2	462	231	nonlinear	nonlinear	$0.0000 * 10^0$
broydnbd	2000	2000	—	nonlinear	—

Results for larger CUTer problems

problem	iter. count		evaluation of full $A(x_k)$			
	exact	inexact	exact	err	Lagr	sum+1
genhs28	3	3	4	0	0	1
orthregc	94	186	93	36	35	72
dtoc1nd	7	8	8	0	4	5
dtoc2	6	153	7	8	0	9
dtoc3	5	5	6	0	0	1
dtoc4	4	5	5	0	4	5
dtoc5	6	9	7	0	3	4
dtoc6	106	104	97	39	15	55
eigena2	1	1	2	0	0	1
eigenc2	11	33	12	2	1	4
broydnbd	7	22	8	0	0	1

Small simulated moving bed system

Biegler, Diehl, W.



Objective: Reach desired state

Results for simulated moving bed problem

N_{dis}	n	m	iter. count		evaluation of full $A(x_k)$			
			exact	inexact	exact	err	Lagr	sum+1
10	125	120	5	5	6	0	4	5
15	185	180	5	6	6	0	3	4
20	245	240	10	7	11	1	3	5
30	365	360	6	6	7	1	2	4
40	485	480	5	6	6	0	4	5
60	725	720	5	7	6	2	2	5

4. Conclusion and outlook

- Trust-region method without exact Jacobian for equality-constrained optimization
- Global convergence imposes two conditions on inexactness
- Resulting inequalities can be easily verified by evaluating matrix-vector products
- TR1 update or frozen Jacobian for approximating $A(x)$
- Future plans:
 - Improvement of algorithm, e.g. second order correction
 - Extension to inequalities