

# An Active-Set Trust-Region Algorithm for Nonlinear Optimization

Richard Waltz

Northwestern University

with

Richard Byrd, Univ. of Colorado

Nick Gould, Rutherford Appleton Laboratory, UK

Jorge Nocedal, Northwestern Univ.

# Motivation

- ◆ Develop **active-set** approach which solves larger problems than existing active-set methods (e.g., SQP).
  - Demand for solving larger nonlinear problems
  - Active-set methods have good properties (E.g., Warm starts, good active-set and sensitivity info, more stable)

# Existing Active-set Approaches

- ◆ Sequential Linear Programming (SLP)
- ◆ Gradient Projection
- ◆ Sequential Linearly Constrained (SLC)
- ◆ Sequential Quadratic Programming (SQP)
  
- ◆ **SLP-EQP**
  - Fletcher, Sainz de la Maza (1989)
  - Chin, Fletcher (1999)
  - **SLIQUE** (Byrd, Gould, Nocedal, W., 2003)

# SQP drawbacks

- ◆ Inefficient when reduced space is large
- ◆ Must form and factor a (dense) reduced Hessian matrix
- ◆ QP subproblems often too expensive.
- ◆ How to handle indefinite case?
- ◆ Not always able to effectively make use of second derivatives.

# Advantages of SLPEQP

- ◆ Solve LPs and EQPs rather than general QPs
- ◆ EQPs solved using iterative approach (PCG)
- ◆ Do not need to form/factorize Hessian
- ◆ Not restricted to problems with a small reduced space
- ◆ Easily makes use of exact 2nd derivatives

# Problem

NLP

$$\begin{aligned} \min_x \quad & f(x) \\ \text{s.t.} \quad & h_i(x) = 0, i \in E \\ & g_i(x) \geq 0, i \in I \\ & x \in \mathcal{R}^n \end{aligned}$$

◆ Functions twice continuously differentiable

# SLPEQP Overview

0. Given:  $x$

1. Solve LP to get working set  $\mathcal{W}$ .

2. Compute a step,  $d$ , by solving an equality constrained QP using constraints in  $\mathcal{W}$ .

3. Set:  $x_T = x + d$ .

# LP subproblem

◆ Estimate active set by solving an **LP**

$$\begin{aligned} \min_d \quad & \nabla f(x)^\top d \\ \text{s.t.} \quad & h_i(x) + \nabla h_i(x)^\top d = 0, \quad i \in E \\ & g_i(x) + \nabla g_i(x)^\top d \geq 0, \quad i \in I \\ & \|d\|_\infty \leq \Delta^{\text{LP}} \end{aligned}$$

$$\begin{aligned} \mathcal{W}(x) = & \{i \in E \mid h_i(x) + \nabla h_i(x)^\top d^{\text{LP}} = 0\} \cup \\ & \{i \in I \mid g_i(x) + \nabla g_i(x)^\top d^{\text{LP}} = 0\} \end{aligned}$$

# $l_1$ LP Formulation

$$\begin{aligned} l(d) = & \nabla f(x)^T d \\ & + v \sum_{i \in E} \left| h_i(x) + \nabla h_i(x)^T d \right| \\ & + v \sum_{i \in I} \max(0, -g_i(x) - \nabla g_i(x)^T d) \end{aligned}$$

$$\begin{array}{ll} \min_d & l(d) \\ \text{s.t.} & \|d\|_\infty \leq \Delta^{\text{LP}} \end{array}$$

# Cauchy Point

Global convergence

$$\min_{\alpha} m(\alpha d^{\text{LP}}), \quad \alpha \in [0, 1]$$

$$m(d) = l(d) + \frac{1}{2} d^{\text{T}} H d$$

$$x^{\text{C}} = x + \alpha^{\text{LP}} d^{\text{LP}}$$

# EQP Solution

◆ Solve using **Projected Conjugate Gradient** method (GLTR)

$$\min_d \quad \nabla f(x)^T d + \frac{1}{2} d^T H d$$

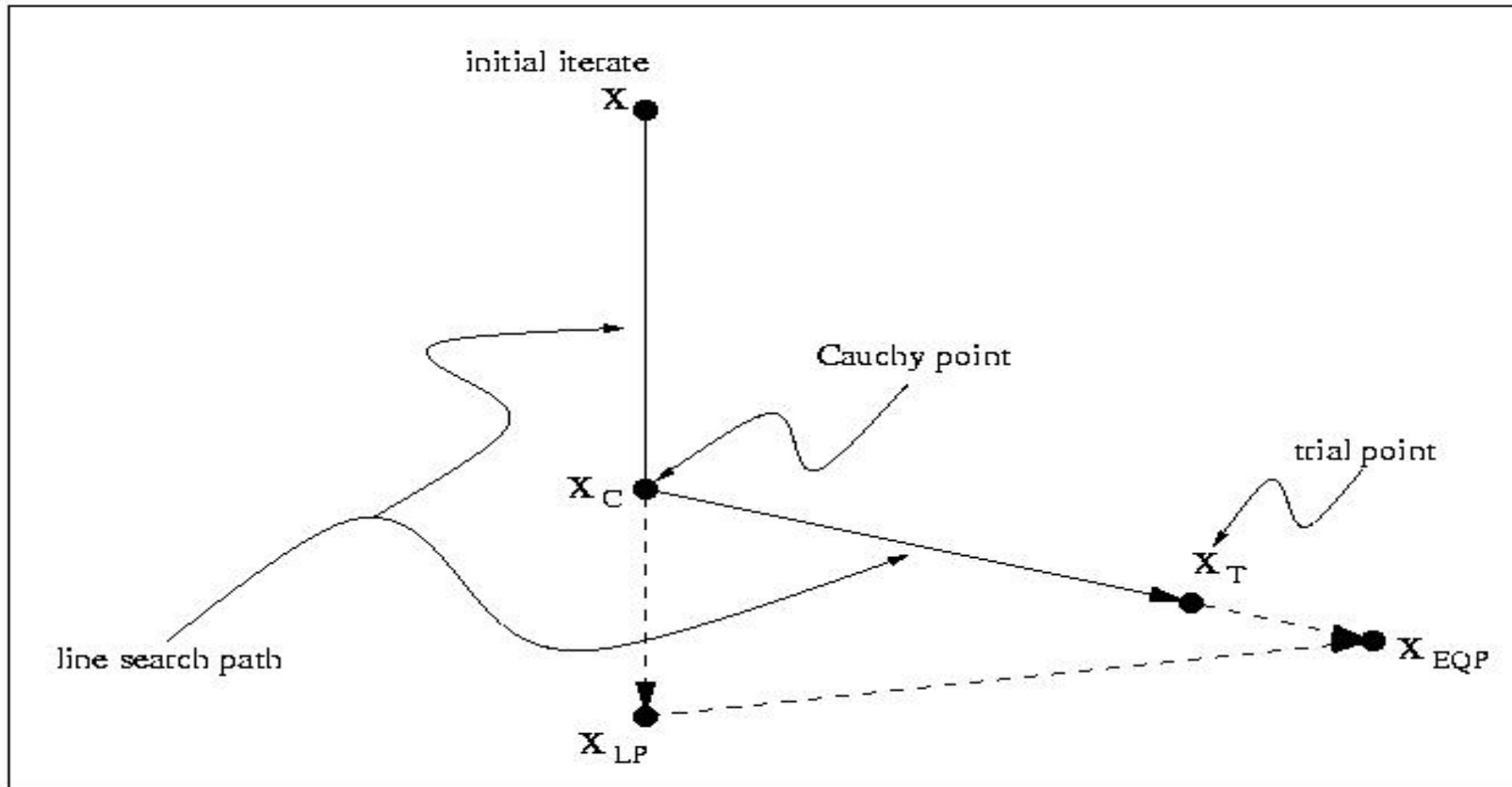
$$\text{s.t.} \quad h_i(x) + \nabla h_i(x)^T d = 0, \quad i \in \mathcal{W} \cap E$$

$$g_i(x) + \nabla g_i(x)^T d = 0, \quad i \in \mathcal{W} \cap I$$

$$\|d\|_2 \leq \Delta$$

# Trial Step

$$x_T = x_C + \alpha_2(x_{EQP} - x_C)$$



# Good News

1. SLIQUE seems to do a good job of identifying the active-set quickly
2. SLIQUE can solve many problems which have lots of degrees of freedom (DOF)

DOF	#	SLIQUE	KNITRO
2000+	57	42	48

Problems with reduced space > 2000 at the solution.

# Bad News

- ◆ SLIQUE inefficient on large-scale problems

Size	#	SLIQUE	KNITRO
VS	266	251 / 94%	251 / 94%
S	76	52 / 68%	67 / 88%
M	118	90 / 76%	103 / 87%
L	100	63 / 63%	85 / 85%
<b>TOT</b>	<b>560</b>	<b>456</b>	<b>506</b>

**VS:**  $n + m < 100$     **M:**  $1000 \leq n + m < 10000$

**S:**  $100 \leq n + m < 1000$     **L:**  $10000 \leq n + m$

# Why?

## SLIQUE timing Statistics

Size	%LP	%EQP	%Fact	%Eval
VS	10	16	5	48
S	36	16	13	19
M	43	33	8	8
L	49	35	5	6
<b>TOT</b>	<b>41</b>	<b>30</b>	<b>7</b>	<b>13</b>

- ◆ LP too expensive for large-scale problems.
- ◆ Relative LP cost grows as problem size grows

# Why is LP so costly?

Warms starts inefficient:

1. Typically many LP trust-region constraints active (even near the solution).
2. Near solution active problem constraints stabilize, but active TR constraints may not!

# Why is LP so costly?

## Statistics:

1. On average about **45% of total LP iterations** are to sort out trust-region constraints
  - Terminate LP early
2. On average active-set does not change over last **27% of outer iterations**
  - Skip LP as we approach solution

# Approximate LP

## ◆ LP problem

$$\begin{array}{ll} \min_d & l(d) \\ \text{s.t.} & \|d\|_{\infty} \leq \Delta^{\text{LP}} \end{array}$$

## ◆ Truncate LP solution when

$$l(0) - l(d) \geq \eta [l(0) - l(d^{\text{LP}})]$$

# Approximate LP

Condition

$$l(0) - l(d) \geq \eta [l(0) - l(d^{\text{LP}})]$$

is satisfied when

$$l(0) - l(d) \geq \frac{\eta}{(1-\eta)} \text{gap}(d)$$

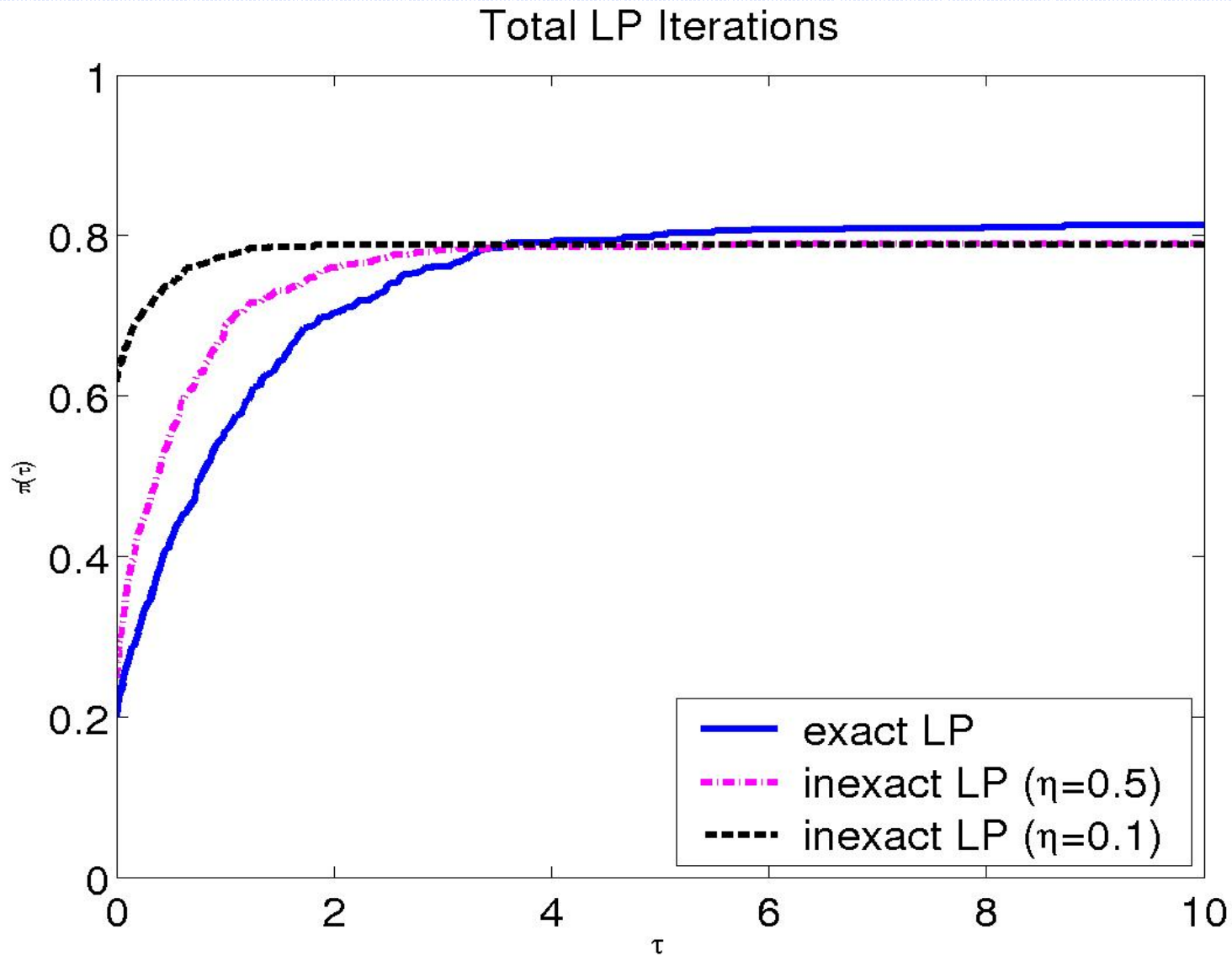
$$\text{gap}(d) = \sum_{j \in S} |rc_j(2\Delta^{\text{LP}})|$$

# Approximate LP Results

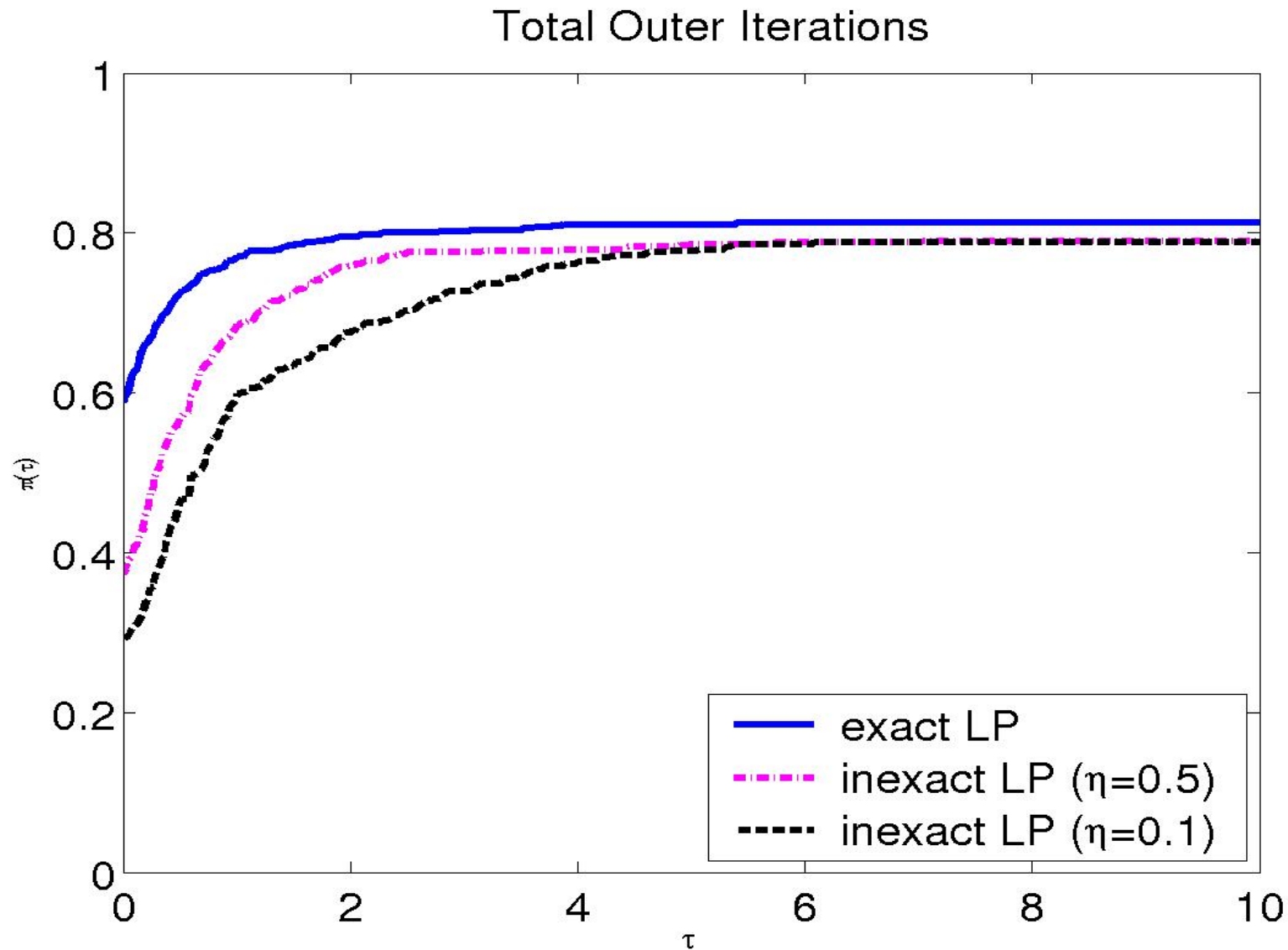
◆ Tested on 568 CUTER problems

Type	#	Ratio LP iters (inexact/exact)	Ratio Outer iters (inexact/exact)
BC	110	0.33	4.74
QP	119	0.81	1.85
GC	339	0.81	2.52
<b>TOT</b>	<b>568</b>	<b>0.70</b>	<b>2.87</b>

# Total LP iterations



# Total outer iterations



# Skip LP

- ◆ Skip LP solve when the active set seems to have stabilized.
- ◆ Move into SEQP mode.
- ◆ How do you know when active-set has stabilized?
- ◆ How to recover if wrong active-set?

# Summary/Conclusion

- ◆ SLIQUE robust overall
- ◆ Efficient for small/medium problems
- ◆ Better able to handle problems with large-reduced space
- ◆ Less efficient and robust for large-scale problems (vs. IP) because of warm start inefficiencies...
- ◆ But progress being made on using approximate LP solutions

PART 2: R. Byrd, Wed. 9:30 (306/34)