SQP, SLP and Interior-Point methods for large-scale nonlinear programming

Nick Gould
Rutherford Appleton Laboratory

\[
\text{minimize } f(x) \text{ subject to } c\varepsilon(x) = 0 \text{ and } c\underline{I}(x) \geq 0
\]

IMA Workshop on Optimization in Simulation-Based Models, 9th 16th January, 2003

Joint in parts with Richard Byrd, Jorge Nocedal, Dominique Orban, Philippe Toint and Richard Waltz
NONLINEAR PROGRAMMING

minimize $f(x)$ subject to $c_{\mathcal{E}}(x) = 0, \quad c_{\mathcal{I}}(x) \geq 0$

\(x \in \mathbb{R}^n\)

\(\circ\) $f$, $c_{\mathcal{E}}$, $c_{\mathcal{I}}$ smooth (preferably $C^2$)

\(\circ\) no convexity assumptions $\implies$ content with local minimizers

\(\circ\) $n, m \overset{\text{def}}{=} |\mathcal{E}| + |\mathcal{I}|$ large, say $O(10^4) \quad O(10^6)$

\(\circ\) Jacobians, Hessians sparse and/or structured

\(\circ\) in general, constraints may be

\(\circ\) bounded on both sides: $c_{\mathcal{I}}^l \leq c_{\mathcal{I}}(x) \leq c_{\mathcal{I}}^u$

\(\circ\) simple bounds on variables: $x^l \leq x \leq x^u$

\(\circ\) linear (or linear network): $a_{\mathcal{E}}^T x = b_{\mathcal{E}}, \quad a_{\mathcal{I}}^T x \geq b_{\mathcal{I}}$

\(\circ\) nonlinear
GALAHAD

Aims:

○ build a **threadsafe fortran 90 library** of optimization modules designed to cope with a variety of commonly-occurring problems
○ in particular, produce a/some successor(s) to LANCELOT

For **GALAHAD 1.0** (April 2002), concentrated on

○ improvements to LANCELOT A \(\rightarrow\) LANCELOT B

○ two algorithms for (non-convex) quadratic programming:
  
  minimize \( q(x) = g^T x + \frac{1}{2} x^T H x \) subject to \( A_\varepsilon x = b_\varepsilon \), \( A_I x \geq b_I \), \( x \in \mathbb{R}^n \)

○ auxiliary packages for pre-solving QPs, solving trust-region subproblems, sparse linear systems, sorting, \ldots
**QPB — an interior point trust-region QP solver**

\[ \text{minimize } g^T x + \frac{1}{2} x^T H x \text{ subject to } A x \geq b \quad x \in \mathbb{R}^n \]

- uses a sequential minimization of the **barrier function**
  \[ \Phi(x, c, \mu) = g^T x + \frac{1}{2} x^T H x - \mu e^T \log c \text{ subject to } A x - c = b \]
  in a trust-region framework, keeping \( c > 0 \)

- feasibility maintained throughout \quad start near analytic centre

- primal-dual model approximately solved as an EQP via
  preconditioned projected CG \quad \text{(G., Hribar, Nocedal)}

- strong underlying convergence theory \quad \text{(Conn, G., Orban, Sartenaer, Toint)}
QPA — an active-set QP solver

\[
\begin{align*}
\text{minimize} & \quad g^T x + \frac{1}{2} x^T H x \\ x & \in \mathbb{R}^n
\end{align*}
\]

subject to \( Ax \geq b \)

\( \circ \) uses a traditional active set method with basic step computation

\[
\begin{align*}
\text{minimize} & \quad g_k^T s + \frac{1}{2} s^T H s \\ s & \in \mathbb{R}^n
\end{align*}
\]

subject to \( A_k s = 0 \)

for some subset \( A_k \) of \( A \)

\( \circ \) EQP subproblem solved via preconditioned projected CG

\( (G., H_{\text{ribar}}, Nocedal) \)

\( \circ \) \textbf{preconditioner} changes by low-rank inertia-controlled update

\( \text{Schur complement updating used} \quad (Gill, Murray, Saunders, Wright) \)

\( \circ \) actually use a single-phase penalty method

\( (Conn, Sinclair) \)

\( \circ \) many technical details

\( (G., Toint) \)
QPA -vs- QPB

- interior-point QPB usually better, and often far better, than active-set QPA when cold started

<table>
<thead>
<tr>
<th>Name</th>
<th>n</th>
<th>m</th>
<th>type</th>
<th>its</th>
<th>time</th>
</tr>
</thead>
<tbody>
<tr>
<td>QPBAND</td>
<td>500000</td>
<td>250000</td>
<td>C</td>
<td>17</td>
<td>2909</td>
</tr>
<tr>
<td>QPNBAND</td>
<td>500000</td>
<td>250000</td>
<td>NC</td>
<td>13</td>
<td>181</td>
</tr>
<tr>
<td>PORTSQP</td>
<td>100000</td>
<td>1</td>
<td>C</td>
<td>11</td>
<td>72</td>
</tr>
<tr>
<td>PORTSNQP</td>
<td>100000</td>
<td>2</td>
<td>NC</td>
<td>22</td>
<td>210</td>
</tr>
</tbody>
</table>

Compaq AlphaServer DS20
(3.5 Gbytes RAM)
QPB
C = convex
NC = nonconvex
time in CPU seconds

- when warm started (good prediction of active set known), QPA often outperforms QPB except on highly degenerate or ill-conditioned examples
GALAHAD 2.0 and the future

In May 2002 (SIOPT meeting, Toronto) I predicted

“. . . next release, GALAHAD 2.0, will include at least

○ SQP methods
  ◦ our implementation of Fletcher’s $\ell_1$QP
  ◦ our implementation of Fletcher & Leyffer’s SQP-filter approach”

I no longer am convinced of this!

What changed?
**SQP — SEQUENTIAL QUADRATIC PROGRAMMING**

\[
\text{minimize } f(x) \text{ subject to } c(x) \geq 0
\]

\[x \in \mathbb{R}^n\]

Basic SQP method:

- from current solution estimate \(x\), compute step \(s\) to

\[
\text{minimize } s^Tg(x) + \frac{1}{2}s^THs \text{ subject to } A(x)s + c(x) \geq 0
\]

\[s \in \mathbb{R}^n\]

+ (possibly) a trust region constraint \(\|s\| \leq \Delta\),

where

- \(g(x) \stackrel{\text{def}}{=} \nabla_x f(x)\)
- \(A(x) \stackrel{\text{def}}{=} \nabla_x c(x)\)
- \(H \approx \nabla_{xx}[f(x) + y^T c(x)]\) for some multipliers \(y\)

- globalize using an appropriate merit function
SQP — DRAWBACKS (I)

\[
\begin{align*}
\text{minimize } & \quad s^T g(x) + \frac{1}{2} s^T H s \\
\text{subject to } & \quad A(x) s + c(x) \geq 0
\end{align*}
\]

The SQP step computation is too coarse/expensive a calculation, especially in early iterations

- alternative cheap steps like those for KNITRO or LOQO far more cost effective ... conjugate gradients, equality constraints
- contrasts with "cheap" truncated Newton steps for unconstrained minimization

(Dembo, Steihaug)

- could truncate QP calculation ... but how in general?

(Murray, Prieto)

\[\implies \text{inefficiency}\]
SQP — DRAWBACKS (II)

\[
\begin{align*}
\text{minimize} & \quad s^T g(x) + \frac{1}{2} s^T H s \\
\text{subject to} & \quad A(x) s + c(x) \geq 0
\end{align*}
\]

In the best possible case, would like to use exact 2nd derivatives, but ...

the SQP step may be inappropriate if \( H \) is indefinite

\( \circ \) local minimizers may be uphill — bad with IP methods

\( \circ \) ultimately lower local (or global) minimizers may initially lead uphill
  if the step is a direction of negative curvature \( \text{(Goldsmith)} \)

\( \circ \) QP may be unbounded from below

\[ \implies \text{inefficiency or even catastrophe} \]
local minimizer may be uphill

→ upper bound

← lower bound

initial point

inferior local minimizer
global minimizer may be locally uphill

initial point

lower bound

global minimizer

upper bound
global minimizer may be infinity

initial point

lower bound

global minimizer
WHAT ARE THE ALTERNATIVES?

What kind of problems can we solve efficiently?

- **unconstrained problems**
  - truncated Newton, (preconditioned) conjugate gradients

- **linear programs**
  - Simplex and Interior Point methods

- **equality-constrained quadratic programs**
  - projected (preconditioned) conjugate gradients
  - can incorporate trust-region constraint using Lanczos

Suggests using any of above as subproblems
SEQUENTIAL LINEAR PROGRAMMING (SLP)

\[
\text{minimize } f(x) \text{ subject to } c_\varepsilon(x) = 0 \text{ and } c_I(x) \geq 0
\]

\[x \in \mathbb{R}^n\]

Find a correction \(\Delta x\) to solution estimate \(x\):

\[
\text{minimize } \Delta x^T \nabla_x f \text{ subject to } \nabla_x c_\varepsilon \Delta x + c_\varepsilon = 0 \text{ and } \nabla_x c_I \Delta x + c_I \geq 0
\]

**good**

- simple
- potentially solve huge problems \(n, |I| = O(10^7 \ 10^8)\)

**bad**

- slow — at best linearly convergent
- constraints may be inconsistent
- no “natural” merit function
- need good LP solver — most are “commercial”
NON-DIFFERENTIABLE PENALTY METHODS

\[
\min_{x \in \mathbb{R}^n} f(x) \text{ subject to } c_I(x) \geq 0
\]

\(\ell_1\) penalty method: “solve”

\[
\min_{x \in \mathbb{R}^n} f(x) + \rho \| \min(c_I(x), 0) \|_1
\]

for sufficiently large \(\rho\) — exact penalty function
Non-smooth problem \(\implies\) can’t use off-the-shelf method \(\implies\)

\(\text{S}\ell_1\)-LP method (Conn, Fletcher)

Find a correction \(\Delta x\) to solution estimate \(x\):

\[
\min_{\Delta x \in \mathbb{R}^n} \Delta x^T \nabla_x f + \rho \| \min(\nabla_x c_I \Delta x + c_I, 0) \|_1
\]
S\ell_1\text{-LP METHOD (cont.)}

**good**
- no inconsistency of constraints
  - natural merit function
  - globally convergent
- potentially solve huge problems \( n, |\mathcal{I}| = O(10^7 \times 10^8) \)
- \( \ell_1\)-LP can be reformulated as an LP

**bad**
- need special LP solver
  - slow at best linearly convergent
- needs a strategy for selection of parameter \( \rho \)
- \( \rho \) diverges if problem is inconsistent
\textbf{\textit{S}ℓ₁-\textit{LP-EQP}} \hspace{0.5cm} \text{(Fletcher, Sainz de la Maza)}

\begin{align*}
\text{minimize} \quad & f(x) \quad \text{subject to} \quad c_I(x) \geq 0 \\
\end{align*}

\begin{itemize}
\item find the set $\mathcal{A} \subseteq \mathcal{I}$ of \textbf{active} constraints from
\begin{align*}
\text{minimize} \quad & \Delta x^T \nabla_x f + \rho \| \min(\nabla_x c_I \Delta x + c_I, 0) \|_1 \\
\end{align*}
for which $\nabla_x c_i \Delta x + c_i = 0$ for $i \in \mathcal{A}$
\item then find a correction $\Delta x$ to “approximately”
\begin{align*}
\text{minimize} \quad & \Delta x^T \nabla_x f + \frac{1}{2} \Delta x^T H \Delta x \quad \text{subject to} \quad \nabla_x c_A \Delta x + c_A = 0 \\
\end{align*}
for some symmetric $H$,
\end{itemize}
$S\ell_1$-LP-EQP (cont.)

good  ○ LP asymptotically determines “correct” active set
        ○ equality QP (EQP) ensures superlinear asymptotic
            convergence if $H \rightarrow \nabla_{xx}[f(x) + y^Tc(x)]$
        ○ compromise between simplicity of SLP and speed of SQP

bad  ○ need good LP solver — most are “commercial”
        ○ may not identify “correct” active set fast
        ○ need efficient way to solve EQP — truncated projected
            conjugate gradients
SLIQUE, AN $\ell_1$-LP-EQP TRUST-REGION METHOD

Overview:  
(Byrd, G., Nocedal, Waltz)

Two trust regions, a “Cauchy” point & an overall trial step

- a trust-region to control the “LP” step
  \[
  \Delta x_{\text{LP}} = \arg \min_{\|\Delta x\|_\infty \leq \Delta_{\text{LP}}} \Delta x^T \nabla x f + \rho \| \min(\nabla x c_L \Delta x + c_L, 0) \|_1
  \]

- a Cauchy point $\Delta x_{\text{CP}} = \alpha_{\text{CP}} \Delta x_{\text{LP}}$ for some $\alpha_{\text{CP}} \in (0, 1]$

- a feasibility step $\Delta x_{\text{F}} = \arg \min_{\|\Delta x\|_2 \leq \beta \Delta} \| \nabla x c_A \Delta x + c_A \|_2$, $\beta \in (0, 1]$

- an (independent) trust-region to control the “EQP” step
  \[
  \Delta x_{\text{EQP}} = \arg \min_{\|\Delta x\|_2 \leq \Delta} \Delta x^T \nabla x f + \frac{1}{2} \Delta x^T H \Delta x \\
  \text{subject to } \nabla x c_A \Delta x + c_A = \nabla x c_A \Delta x_{\text{F}} + c_A
  \]

- an overall trial step $\Delta x_{\text{T}} = \Delta x_{\text{CP}} + \alpha_{\text{T}} (\Delta x_{\text{EQP}} - \Delta x_{\text{CP}})$ for some $\alpha_{\text{T}} \in (0, 1]$
THE OVERALL TRIAL STEP

initial iterate

X

Cauchy point

X_C

line search path

X_LP

X_T

trial point

X_EQP
THE LP STEP

The **LP trust region**

- stops large steps if the LP model is unbounded from below
- must not be too large as otherwise optimally “inactive” constraints will appear in the active set
- does not have to lie within overall trust region
THE LP TRUST REGION SHOULD NOT BE TOO BIG
THE CAUCHY STEP

The Cauchy stepsize $\alpha_{cp}$

- ensures that the Cauchy point lies within the overall trust region
- ensures that the linear and quadratic model decreases are “similar”
- should not be too small as to prevent convergence

Typically $\alpha^{cp}$ is required to approximate the minimizer of

$$\alpha \Delta x_{lp}^T \nabla_x f + \frac{1}{2} \alpha^2 \Delta x_{lp}^T H \Delta x_{lp} + \rho \| \min(\alpha \nabla_x c^T \Delta x_{lp} + c^T, 0) \|_1$$

within the intersection of the LP and EQP trust-regions
THE EQP STEP

The **EQP trust region**

- stops large steps if the EQP model is unbounded from below
- must shrink if progress is impossible otherwise
  - measure progress by comparing decrease in EQP model with actual decrease in the penalty function — typical trust-region mechanism
- should not shrink to zero unnecessarily as this will prevent both global and fast local convergence
THE OVERALL STEP

The overall trial stepsize $\alpha_T$

- ensures that the EQP and quadratic model decreases are “similar”
- should ultimately be 1
STEP ACCEPTANCE

New point $x^+$ given as

$$
x^+ = \begin{cases} 
  x + \Delta x_T & \text{if } \rho \geq 1.0 \\
  x & \text{otherwise}
\end{cases}
$$

where

$$
\rho = \frac{\phi(x) - \phi(x + \Delta x_T)}{\Delta x_T^T \nabla_x f + \frac{1}{2} \Delta x_T^T H \Delta x_T + \rho \| \min(\nabla_x c_T \Delta x_T + c_T, 0) \|_1}
$$

and

$$
\phi(x) = f(x) + \rho \| \min(c_T(x), 0) \|_1
$$
TRUST-REGION RADII UPDATES

LP radius update:

$$
\Delta_{LP}^+ = \begin{cases} 
\min(\max\{1.2\|\Delta x_T\|_\infty, 1.2\|\Delta x_{CP}\|_\infty, 0.1\Delta_{LP}, 7\Delta_{LP}\}) & \text{if } \rho \geq 10^{-8} \\
\min(\max\{0.5\|\Delta x_T\|_\infty, 0.1\Delta_{LP}\}, \Delta_{LP}) & \text{otherwise}
\end{cases}
$$

Master radius update:

$$
\Delta^+ = \begin{cases} 
\max(\Delta, 7\|\Delta x_T\|_2), & \text{if } \rho \geq 0.9 \\
\max(\Delta, 2\|\Delta x_T\|_2), & \text{if } 0.3 \leq \rho < 0.9 \\
\Delta, & \text{if } 10^{-8} \leq \rho < 0.3 \\
\min(0.5\Delta, 0.5\|\Delta x_T\|_2), & \text{if } \rho < 10^{-8}
\end{cases}
$$
OTHER DETAILS

- currently use MINOS (simplex code) to solve LPs *(Murtagh & Saunders)*
- use GLTR *(GALAHAD code)* with augmented-system preconditioning to solve EQPs *(G., Lucidi, Roma, Toint)*
- hot-start LPs
- penalty parameter $\rho$ update based on how good a job current $\rho$ does in achieving “linearized feasibility” for LP
- lots of other “tricks”
- covered by more general global convergence theory *(Byrd, G., Nocedal, Waltz)*
HOW DOES THIS WORK IN PRACTICE?

CUTEr test set problem sizes and characteristics

<table>
<thead>
<tr>
<th>Problem class</th>
<th>Problem size</th>
<th># of problems</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>QP</td>
</tr>
<tr>
<td>Very Small</td>
<td>$1 \leq n + m &lt; 100$</td>
<td>30</td>
</tr>
<tr>
<td>Small</td>
<td>$100 \leq n + m &lt; 1000$</td>
<td>24</td>
</tr>
<tr>
<td>Medium</td>
<td>$1000 \leq n + m &lt; 10000$</td>
<td>27</td>
</tr>
<tr>
<td>Large</td>
<td>$10000 \leq n + m$</td>
<td>36</td>
</tr>
<tr>
<td>Total</td>
<td>all</td>
<td>117</td>
</tr>
</tbody>
</table>

BC = Bound Constrained, GC = Generally Constrained

Compare with:

- KNITRO  interior-point, CG-based, mature
- SNOPT  SQP, 1st derivative, mature
**ROBUSTNESS** By problem class and problem size:

<table>
<thead>
<tr>
<th>Problem class</th>
<th>Sample size</th>
<th>Slique # Opt</th>
<th>Slique % Opt</th>
<th>Knitro # Opt</th>
<th>Knitro % Opt</th>
<th>Snopt # Opt</th>
<th>Snopt % Opt</th>
</tr>
</thead>
<tbody>
<tr>
<td>QP</td>
<td>117</td>
<td>89</td>
<td>76.1</td>
<td>113</td>
<td>96.6</td>
<td>99</td>
<td>84.6</td>
</tr>
<tr>
<td>BC</td>
<td>107</td>
<td>96</td>
<td>89.7</td>
<td>98</td>
<td>91.6</td>
<td>71</td>
<td>66.3</td>
</tr>
<tr>
<td>GC</td>
<td>336</td>
<td>253</td>
<td>75.3</td>
<td>295</td>
<td>87.8</td>
<td>285</td>
<td>84.8</td>
</tr>
<tr>
<td>Total</td>
<td>560</td>
<td>438</td>
<td>78.2</td>
<td>506</td>
<td>90.4</td>
<td>455</td>
<td>81.3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Problem class</th>
<th>Sample size</th>
<th>Slique # Opt</th>
<th>Slique % Opt</th>
<th>Knitro # Opt</th>
<th>Knitro % Opt</th>
<th>Snopt # Opt</th>
<th>Snopt % Opt</th>
</tr>
</thead>
<tbody>
<tr>
<td>Very Small</td>
<td>266</td>
<td>247</td>
<td>92.9</td>
<td>251</td>
<td>94.4</td>
<td>250</td>
<td>94.0</td>
</tr>
<tr>
<td>Small</td>
<td>76</td>
<td>58</td>
<td>76.3</td>
<td>67</td>
<td>88.2</td>
<td>70</td>
<td>92.1</td>
</tr>
<tr>
<td>Medium</td>
<td>118</td>
<td>86</td>
<td>72.9</td>
<td>103</td>
<td>87.3</td>
<td>83</td>
<td>70.3</td>
</tr>
<tr>
<td>Large</td>
<td>100</td>
<td>47</td>
<td>47.0</td>
<td>85</td>
<td>85.0</td>
<td>52</td>
<td>52.0</td>
</tr>
<tr>
<td>Total</td>
<td>560</td>
<td>438</td>
<td>78.2</td>
<td>506</td>
<td>90.4</td>
<td>455</td>
<td>81.3</td>
</tr>
</tbody>
</table>
PERFORMANCE PROFILE - CPU medium/large BCs

log2–scaled CPU Performance Profile. BC Problems in CUTEr.

- SLIQUE
- KNITRO
- SNOPT
PERFORMANCE PROFILE - CPU medium/large QPs

log2–scaled CPU Performance Profile. QP Problems in CUTEr.

π(τ)

0 1 2 3 4 5 6 7 8 9 10

τ

SLIQUE

KNITRO

SNOPT
PERFORMANCE PROFILE - CPU medium/large GCs

log2–scaled CPU Performance Profile. GC Problems in CUTEr.

- SLIQUE
- KNITRO
- SNOPT
AN INTERIOR-POINT ALTERNATIVE

A non-differentiable penalty-barrier method

\[
\begin{align*}
\tag{1}
\text{minimize} & \quad f(x) \quad \text{subject to} \quad c_\mathcal{E}(x) = 0 \quad \text{and} \quad c_\mathcal{I}(x) \geq 0 \\
& \quad x \in \mathbb{R}^n
\end{align*}
\]

Aim to solve problem by minimizing the non-differentiable penalty \( f^n \)

\[
\phi(x, \nu) = f(x) + \nu \sum_{i \in \mathcal{E}} |c_i(x)| + \nu \sum_{i \in \mathcal{I}} \max(-c_i(x), 0)
\]

for some sufficiently large \( \nu \)

Can reformulate this as a smooth problem:

- replace the terms \(|c_i(x)|\) and \(\max(-c_i(x), 0)\) by equivalent smooth terms
NON-SMOOTH TERMS

equality constraints: write contribution $\nu|c_i(x)|$ as

$$\nu[r_i + s_i], \text{ where } c_i(x) = r_i - s_i \text{ and } (r_i, s_i) \geq 0,$$

or alternatively as

$$\nu[c_i(x) + 2s_i], \text{ where } c_i(x) + s_i \geq 0 \text{ and } s_i \geq 0$$

inequality constraints: write contribution $\nu \max(-c_i(x), 0)$ as

$$\nu s_i, \text{ where } c_i(x) = r_i - s_i \text{ and } (r_i, s_i) \geq 0$$

or alternatively as

$$\nu s_i, \text{ where } c_i(x) + s_i \geq 0 \text{ and } s_i \geq 0$$

$\Rightarrow$
A SMOOTH REFORMULATION

Thus the minimization of $\phi$ may be expressed as

$$
\begin{align*}
\minimize_{x,s} & \quad f(x) + \nu \sum_{i \in \mathcal{E}} [c_i(x) + 2s_i] + \nu \sum_{i \in \mathcal{I}} s_i \\
\text{subject to} & \quad c_i(x) + s_i \geq 0 \quad \text{and} \quad s_i \geq 0 \quad \text{for all} \quad i \in \mathcal{E} \cup \mathcal{I}
\end{align*}
$$

involving “surplus” variables $s$.

- can use IP methods to solve this inequality-constrained problem
- finding an initial interior point is trivial
- may sometimes be better to replace $\nu |c_i(x)|$ term by
  
  $$
  \nu [2r_i - c_i(x)], \quad \text{where} \quad r_i - c_i(x) \geq 0 \quad \text{and} \quad r_i \geq 0
  $$

  especially if initially $c_i(x) < 0$

- if ever $c_i(x) > 0$, can simply remove $s_i$

(Mayne & Polak, Tits, Wächter, Bakhtiari, Urban & Lawrence)
WHY IS THIS PROMISING?

- general constrained problem reduced to smooth unconstrained problem simply involving barrier terms
- linear algebra well understood for such problems
- Newton-like subproblem easy to truncate using (e.g.) conjugate gradients
- to improve performance, better to use primal-dual rather than primal Newton model
- can take direct account of (for example) linear constraints & simple bounds on variables ("phase-1" procedure)
- global and local convergence theory established
BARRIER FUNCTION AND ITS DERIVATIVES

(logarithmic) barrier function: \( \Psi_{\mu,\nu}(x, s) = \)
\[
f(x) + \nu e^T \varepsilon [c\varepsilon(x) + 2s\varepsilon] + \nu e^T s - \mu e^T \log(c(x) + s) - \mu e^T \log s
\]
\[
\nabla_v \Psi_{\mu,\nu}(x, s) = \begin{pmatrix} g(x) - J^T(x) y(x, s) \\ \nu e - y(x, s) - u(s) \end{pmatrix}
\]
\[
\nabla_{vv} \Psi_{\mu,\nu}(x, s) = \begin{pmatrix} H(x, y(x, s)) + \mu J^T(x)(C(x) + S)^{-2} J(x) & \mu J^T(x)(C(x) + S)^{-2} \\ \mu(C(x) + S)^{-2} J(x) & \mu(C(x) + S)^{-2} + \mu S^{-2} \end{pmatrix}
\]

where \( v = (x, s) \)
\[
y\varepsilon(x, s) = \mu(C\varepsilon(x) + S\varepsilon)^{-1} e\varepsilon - \nu e\varepsilon
\]
\[
y\varepsilon(x, s) = \mu(C\varepsilon(x) + S\varepsilon)^{-1} e\varepsilon
\]
\[
u(s) = \mu S^{-1} e
\]
\[
J(x) = \nabla_x c(x) \quad \text{and} \quad H(x, y) = \nabla_{xx} f(x) - \sum_i y_i \nabla_{xx} c_i(x)
\]
\[
C(x) = \text{diag } c(x) \quad \text{(etc)}
\]
BASIC SEARCH DIRECTION SUBPROBLEM

Primal-dual Hessian approximation: \( \nabla_{vv} \Psi_{\mu,\nu}^{pd}(x, s) = \)
\[
\begin{pmatrix}
H(x, y^{pd}) + J^T(x)Y^{pd}(C(x) + S)^{-1}J(x) & J^T(x)Y^{pd}(C(x) + S)^{-1}

Y^{pd}(C(x) + S)^{-1}J(x) & Y^{pd}(C(x) + S)^{-1} + U^{pd}S^{-1}
\end{pmatrix}
\]

Find search direction \( \Delta v = (\Delta x, \Delta s) \) to (approximately)
\[
\text{minimize} \quad \Delta v^T \nabla_v \Psi_{\mu,\nu}(x, s) + \frac{1}{2} \Delta v^T \nabla_{vv} \Psi_{\mu,\nu}(x, s) \Delta v \quad \text{s.t.} \quad \| \Delta v \|_B \leq \Delta
\]

\( \circ \) \( B \) positive-definite approximation of \( \nabla_{vv} \Psi_{\mu,\nu}^{pd}(x, s) \)

\( \circ \) \( B \) replaces \( H(x, y^{pd}) \) by suitable \( P \), e.g.

\( \circ \) \( P = 0 \)

\( \circ \) \( P = I \)

\( \circ \) \( P = H(x, y^{pd}) \) (!!!)
PRECONDITIONING CONJUGATE GRADIENTS

\[
\begin{align*}
\text{minimize } & \Delta v^T \nabla_v \Psi_{\mu,\nu}(x, s) + \frac{1}{2} \Delta v^T \nabla_{vv} \Psi_{\mu,\nu}^{PD}(x, s) \Delta v \\
\text{s.t. } & \|\Delta v\|_B \leq \Delta
\end{align*}
\]

Use preconditioned conjugate gradients \hspace{1cm} \text{basic preconditioning step}

\[
\begin{pmatrix}
P + J^T Y(C + S)^{-1} J & J^T Y(C + S)^{-1} \\
Y(C + S)^{-1} J & Y(C + S)^{-1} + US^{-1}
\end{pmatrix}
\begin{pmatrix}
\Delta x \\
\Delta s
\end{pmatrix}
= \begin{pmatrix}
r_x \\
r_s
\end{pmatrix}
\]

Possibly too dense \implies \text{define } w = Y(C + S)^{-1}(J \Delta x + \Delta s) \implies

\[
\begin{pmatrix}
P & 0 & J^T \\
0 & US^{-1} & I \\
J & I & -Y^{-1}(C + S)
\end{pmatrix}
\begin{pmatrix}
\Delta x \\
\Delta s \\
w
\end{pmatrix}
= \begin{pmatrix}
r_x \\
r_s \\
0
\end{pmatrix}
\]

\(P\) suitable \iff \text{above matrix has precisely rank } J - \text{ve eigenvalues}
OUTSTANDING ISSUES

- penalty parameter updates
- which side should we penalize equality constraints?
- is it better to remove surplus variables $s_i$ as soon as possible?
- is primal-dual Hessian better “globally”?
- choice of $P$ in preconditioner?
CONCLUSIONS

○ SQP methods may be too expensive in general

○ cheaper alternatives using LP & unconstrained minimization subproblems worth pursuing

○ SLIQUE “promising” but needs improvements — IP LP??

○ non-differentiable penalty-barrier method SUPERB(?) under development

○ structure & decomposition likely crucial to make further progress

○ there is already very good, publicly available software for solving linear & nonlinear optimization problems