Regularized Active-Set and Interior Methods for Nonlinear Optimization

Philip E. Gill
University of California, San Diego

New Directions in Nonlinear Optimization

Wednesday 19th March 2014, The University of Manchester
Over the last 10 years improvements in computer hardware have resulted in a major change in how linear algebra should be done in nonlinear optimization solvers.

This change in linear algebra has resulted in the development of some new optimization methods that are designed to better exploit modern computer architectures.
Over the last 10 years improvements in computer hardware have resulted in a major change in how linear algebra should be done in nonlinear optimization solvers.

This change in linear algebra has resulted in the development of some new optimization methods that are designed to better exploit modern computer architectures.
Context

- General-purpose nonlinear solvers
  ⇒ the solvers are used in a variety of applications
- active-set methods, interior methods
- direct methods, linear equations
Classification of methods

Broadly speaking, the most “successful” general-purpose solvers are either:

- active-set methods (e.g., MINOS, SNOPT, CONOPT, . . . )

or

- interior methods (e.g., IPOPT, KNITRO, LOQO, . . . )

In many cases the problems they are applied to are “large-scale”.
Main Topics

- A “crash course” on ...
  - Active-set methods
  - Interior methods

(with apologies to the experts in the audience!)

- The impact of improvements in computer hardware
- Regularization
Active-Set Methods
It helps to start with a much simpler problem.

Optimization with equality constraints:

\[
\minimize_{x \in \mathbb{R}^n} f(x) \quad \text{subject to} \quad c(x) = 0.
\]

Under certain regularity assumptions the optimality conditions are:

\[
g(x) - J(x)^T y = 0 \\
c(x) = 0,
\]

where: \( y \) is a vector of \textit{Lagrange multipliers} \( g(x) \) is \( \nabla f(x) \), the \textit{objective gradient} \( J(x) \) is \( c'(x) \), the \textit{constraint Jacobian}. 
It helps to start with a much simpler problem.

Optimization with equality constraints:

\[
\minimize_{x \in \mathbb{R}^n} f(x) \quad \text{subject to} \quad c(x) = 0.
\]

Under certain regularity assumptions the optimality conditions are:

\[
g(x) - J(x)^T y = 0 \\
c(x) = 0,
\]

where: \( y \) is a vector of Lagrange multipliers
\( g(x) \) is \( \nabla f(x) \), the objective gradient
\( J(x) \) is \( c'(x) \), the constraint Jacobian.
An optimal \((x^*, y^*)\) solves \(F(x, y) = 0\), with

\[
F(x, y) = \begin{pmatrix} g(x) - J(x)^T y \\ -c(x) \end{pmatrix}.
\]

The Jacobian of \(F\) is

\[
\begin{pmatrix}
H(x, y) & -J(x)^T \\
-J(x) & 0
\end{pmatrix},
\]

with \(H(x, y) = \nabla^2 f(x) - \sum_{i=1}^m y_i \nabla^2 c_i(x)\), the Lagrangian Hessian.
An optimal \((x^*, y^*)\) solves \(F(x, y) = 0\), with

\[
F(x, y) = \begin{pmatrix}
g(x) - J(x)^T y \\
-c(x)
\end{pmatrix}.
\]

The Jacobian of \(F\) is

\[
\begin{pmatrix}
H(x, y) & -J(x)^T \\
-J(x) & 0
\end{pmatrix},
\]

with \(H(x, y) = \nabla^2 f(x) - \sum_{i=1}^{m} y_i \nabla^2 c_i(x)\), the \textit{Lagrangian Hessian}.
An optimal \((x^*, y^*)\) solves \(F(x, y) = 0\), with

\[
F(x, y) = \begin{pmatrix}
g(x) - J(x)^T y \\
-c(x)
\end{pmatrix}.
\]

The Jacobian of \(F\) is

\[
\begin{pmatrix}
H(x, y) & -J(x)^T \\
-J(x) & 0
\end{pmatrix},
\]

with \(H(x, y) = \nabla^2 f(x) - \sum_{i=1}^{m} y_i \nabla^2 c_i(x)\), the **Lagrangian Hessian**.
The Newton equations for the change in \((x_k, y_k)\) are:

\[
\begin{pmatrix}
H_k & -J_k^T \\
-J_k & 0
\end{pmatrix}
\begin{pmatrix}
\Delta x_k \\
\Delta y_k
\end{pmatrix}
= -
\begin{pmatrix}
g_k - J_k^T y_k \\
-c_k
\end{pmatrix}.
\]

These equations define the Newton-Lagrange method.

The Newton-Lagrange equations are the optimality conditions for a solution \((x_k + \Delta x_k, y_k + \Delta y_k)\) of the quadratic program (QP):

\[
\begin{align*}
\text{minimize} & \quad g_k^T(x - x_k) + \frac{1}{2}(x - x_k)^T H_k (x - x_k) \\
\text{subject to} & \quad c_k + J_k(x - x_k) = 0.
\end{align*}
\]
The *Newton equations* for the change in \((x_k, y_k)\) are:

\[
\begin{pmatrix}
H_k & -J_k^T \\
-J_k & 0
\end{pmatrix}
\begin{pmatrix}
\Delta x_k \\
\Delta y_k
\end{pmatrix}
= -
\begin{pmatrix}
g_k - J_k^T y_k \\
-c_k
\end{pmatrix}.
\]

These equations define the *Newton-Lagrange method*.

The Newton-Lagrange equations are the optimality conditions for a solution \((x_k + \Delta x_k, y_k + \Delta y_k)\) of the quadratic program (QP):

\[
\begin{align*}
\text{minimize} & \quad g_k^T (x - x_k) + \frac{1}{2} (x - x_k)^T H_k (x - x_k) \\
\text{subject to} & \quad c_k + J_k (x - x_k) = 0.
\end{align*}
\]
The Newton-Lagrange equations are equivalent to the *KKT* equations:

\[
\begin{pmatrix}
H_k & J_k^T \\
J_k & 0
\end{pmatrix}
\begin{pmatrix}
\Delta x_k \\
-\Delta y_k
\end{pmatrix} = - \begin{pmatrix}
g_k - J_k^T y_k \\
c_k
\end{pmatrix}.
\]

Under certain assumptions, \(\{(x_k, y_k)\}\) converges superlinearly.

*Line-search* and *trust-region* methods force convergence with

\[
M(x_{k+1}, y_{k+1}) < M(x_k, y_k)
\]

for some *merit function* \(M(x, y)\).

E.g.,

\[
M(x, y) = f(x) + \rho \|c(x)\|_1 = f(x) + \rho \sum_{i=1}^{m} |c_i(x)|,
\]

where \(\rho\) is a positive scalar.
The Newton-Lagrange equations are equivalent to the *KKT equations*:

\[
\begin{pmatrix}
H_k & J_k^T \\ J_k & 0
\end{pmatrix}
\begin{pmatrix}
\Delta x_k \\ -\Delta y_k
\end{pmatrix}
= -\begin{pmatrix}
g_k - J_k^T y_k \\ c_k
\end{pmatrix}.
\]

Under certain assumptions, \(\{(x_k, y_k)\}\) converges *superlinearly*.

*Line-search* and *trust-region* methods force convergence with \(M(x_{k+1}, y_{k+1}) < M(x_k, y_k)\) for some *merit function* \(M(x, y)\).

e.g.,

\[
M(x, y) = f(x) + \rho \|c(x)\|_1 = f(x) + \rho \sum_{i=1}^{m} |c_i(x)|,
\]

where \(\rho\) is a positive scalar.
The Newton-Lagrange equations are equivalent to the \textit{KKT equations}:

\[
\begin{pmatrix}
H_k & J_k^T \\
J_k & 0
\end{pmatrix}
\begin{pmatrix}
\Delta x_k \\
- \Delta y_k
\end{pmatrix}
= - \begin{pmatrix}
g_k - J_k^T y_k \\
c_k
\end{pmatrix}.
\]

Under certain assumptions, \(\{(x_k, y_k)\}\) converges \textit{superlinearly}.

\textit{Line-search} and \textit{trust-region} methods force convergence with

\[M(x_{k+1}, y_{k+1}) < M(x_k, y_k)\] for some \textit{merit function} \(M(x, y)\).

\[M(x, y) = f(x) + \rho \|c(x)\|_1 = f(x) + \rho \sum_{i=1}^{m} |c_i(x)|,\]

where \(\rho\) is a positive scalar.
Inequality constraints

Optimization with a mixture of equality and inequality constraints:

$$\min_{x \in \mathbb{R}^n} f(x) \quad \text{subject to} \quad c(x) = 0, \quad x \geq 0.$$  

The QP subproblem now has inequality constraints too, i.e.,

$$\min_{x \in \mathbb{R}^n} g_k^T(x - x_k) + \frac{1}{2}(x - x_k)^T H_k (x - x_k) \quad \text{subject to} \quad c_k + J_k(x - x_k) = 0, \quad x \geq 0.$$  

$\Rightarrow$ the conventional sequential quadratic programming (SQP) method.
Active-set QP methods solve a sequence of equality-constrained problems:

\[
\begin{align*}
\text{minimize} \quad & g_k^T(x - x_k) + \frac{1}{2}(x - x_k)^TH_k(x - x_k) \\
\text{subject to} \quad & c_k + J_k(x - x_k) = 0, \quad x_N = 0.
\end{align*}
\]

“\(x_N\)” are the subset of variables “Not free to move”

“\(x_F\)” are the subset of variables “Free to move”
Each QP subproblem involves the “Newton equations” with matrix:

\[ K = \begin{pmatrix} H_F & J_F^T \\ J_F & 0 \end{pmatrix}, \]

where \( H_F \) and \( J_F \) are formed from columns associated with variables that are free to move.

As the QP iterations proceed:
- \( J_F \) has a column \textit{added}, or a column \textit{deleted}.
- \( H_F \) has a row and column \textit{added} or \textit{deleted}.
Each QP subproblem involves the “Newton equations” with matrix:

\[
K = \begin{pmatrix}
H_F & J_F^T \\
J_F & 0
\end{pmatrix},
\]

where \(H_F\) and \(J_F\) are formed from columns associated with variables that are free to move.

As the QP iterations proceed:

- \(J_F\) has a column \textit{added}, or a column \textit{deleted}.
- \(H_F\) has a row and column \textit{added} or \textit{deleted}.
Conventional active-set methods

Pros and cons

Active-set methods . . .

- identify and utilize only a *subset* of the inequalities
- are very good at exploiting a good approximate solution
- provide a “certificate of infeasibility” for infeasible problems

Active-set methods . . .

- are dependent on customized matrix updating software
- may require many QP iterations far from the solution
Conventional active-set methods

Pros and cons

Active-set methods . . .

- identify and utilize only a *subset* of the inequalities
- are very good at exploiting a good approximate solution
- provide a “certificate of infeasibility” for infeasible problems

Active-set methods . . .

- are dependent on customized matrix updating software
- may require many QP iterations far from the solution
Interior Methods
The optimality conditions for the inequality constrained case are:

\[ c(x) = 0, \quad x \geq 0, \]
\[ g(x) - J(x)^T y - z = 0, \quad z \geq 0, \]
\[ x \cdot z = 0. \]

Perturb the optimality conditions by a small scalar \( \mu \):

\[ c(x) = 0, \quad x \geq 0, \]
\[ g(x) - J(x)^T y - z = 0, \quad z \geq 0, \]
\[ x \cdot z = \mu e. \]
Interior Methods

For a sequence of decreasing $\mu$, apply Newton’s method to the equations $F(x, y, z) = 0$,

$$F(x, y, z) = \begin{pmatrix} c(x) \\ g(x) - J(x)^T y - z \\ x \cdot z - \mu e \end{pmatrix}$$

while maintaining $x > 0$ and $z > 0$. 
The Newton equations for \((\Delta x, \Delta y, \Delta z)\) have a principal block:

\[
\begin{pmatrix}
H + D & J^T \\
J & 0
\end{pmatrix}
\begin{pmatrix}
\Delta x \\
-\Delta y
\end{pmatrix}
= -
\begin{pmatrix}
g - J^T y - \mu X^{-1} e \\
c
\end{pmatrix},
\]

with \(D = ZX^{-1}\), and \(X = \text{diag}(x_1, x_2, \ldots, x_n)\), etc.

\(D = ZX^{-1}\) positive definite but \textit{inherently ill-conditioned}, i.e.,

\(D_{ii} = O(\mu)\) and \(1/D_{jj} = O(\mu)\) for some \(i\) and \(j\).

This ill-conditioning is \textit{benign} under certain assumptions.

(S. Wright '95, Forsgren, G & Shinnerl '96, M. Wright '98)
Conventional interior methods

Pros and cons

Interior methods . . .

- are easy to implement with third-party solvers
- can require very few Newton iterations

Interior methods . . .

- must factor a matrix that involves all of $J$
- are not good at exploiting a good approximate solution
- may behave poorly on infeasible problems
Conventional interior methods

Pros and cons

Interior methods . . .
- are easy to implement with third-party solvers
- can require very few Newton iterations

Interior methods . . .
- must factor a matrix that involves all of $J$
- are not good at exploiting a good approximate solution
- may behave poorly on infeasible problems
Hardware trends and their consequences

Over the last 10 years changes in computer hardware have resulted in a major change in how linear algebra is done in nonlinear solvers.

The change in linear algebra has provided the motivation for the development of new optimization methods.

These new methods attempt to combine some of the best features of SQP methods and interior methods.
Over the last 10 years changes in computer hardware have resulted in a major change in how linear algebra is done in nonlinear solvers.

The change in linear algebra has provided the motivation for the development of new optimization methods.

These new methods attempt to combine some of the best features of SQP methods and interior methods.
Hardware trends and their consequences

Over the last 10 years changes in computer hardware have resulted in a major change in how linear algebra is done in nonlinear solvers.

The change in linear algebra has provided the motivation for the development of new optimization methods.

These new methods attempt to combine some of the best features of *SQP methods* and *interior methods*.
Equations! Equations! Equations!

\[ Bx = b \quad \text{and} \quad Kv = f \]

Sparse matrices

\[ B \quad (\text{unsymmetric}), \quad K = \begin{pmatrix} H & A^T \\ A & -D \end{pmatrix} \]

The dominant approaches:

Active-set methods 1970–present
Block factorization of \( K \) + updating methods for \( B \)

Interior methods 1984–present
(i) block factorization of \( K \);
(ii) \textit{direct factorization} of \( K \);
(iii) iterative solvers for \( Kv = f \)
Equations! Equations! Equations!

\[ Bx = b \quad \quad \quad Kv = f \]

Sparse matrices

\[ B \quad (\text{unsymmetric}), \quad K = \begin{pmatrix} H & A^T \\ A & -D \end{pmatrix} \]

The dominant approaches:

Active-set methods 1970–present

Block factorization of \( K \)

+ updating methods for \( B \)

Interior methods 1984–present

(i) block factorization of \( K \);

(ii) direct factorization of \( K \);

(iii) iterative solvers for \( Kv = f \)
\[ Bx = b \quad \quad Kv = f \]

Sparse matrices

\[ B \quad (\text{unsymmetric}), \quad K = \begin{pmatrix} H & A^T \\ A & -D \end{pmatrix} \]

The dominant approaches:

Active-set methods 1970–present
- Block factorization of \( K \)
- + updating methods for \( B \)

Interior methods 1984–present
- (i) block factorization of \( K \);
- (ii) \textit{direct factorization} of \( K \);
- (iii) iterative solvers for \( Kv = f \)
# 27 years of progress

**Linear programming with MINOS**

**PILOT** 1442 rows, 3652 columns, 43220 nonzeros

<table>
<thead>
<tr>
<th>Year</th>
<th>Itns</th>
<th>Cpu secs</th>
<th>Architecture</th>
</tr>
</thead>
<tbody>
<tr>
<td>1987</td>
<td>–</td>
<td>$8.7 \times 10^4$</td>
<td>DEC Vaxstation II</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2005</td>
<td>17738</td>
<td>22.2</td>
<td>dual-core Xeon</td>
</tr>
<tr>
<td>2006</td>
<td>16865</td>
<td>9.7</td>
<td>dual-core Opteron 2.4Ghz</td>
</tr>
<tr>
<td>2007</td>
<td>16865</td>
<td>8.1</td>
<td>dual-core Opteron 3.1Ghz</td>
</tr>
<tr>
<td>2008</td>
<td>16865</td>
<td>8.7</td>
<td>quad-core Opteron 3.1Ghz</td>
</tr>
<tr>
<td>2014</td>
<td>16865</td>
<td>5.9</td>
<td>quad-core Intel I7 2.9Ghz</td>
</tr>
</tbody>
</table>
27 years of progress

Linear programming with MINOS

PILOT 1442 rows, 3652 columns, 43220 nonzeros

<table>
<thead>
<tr>
<th>Year</th>
<th>Itns</th>
<th>Cpu secs</th>
<th>Architecture</th>
</tr>
</thead>
<tbody>
<tr>
<td>1987</td>
<td>–</td>
<td>$8.7 \times 10^4$</td>
<td>DEC Vaxstation II</td>
</tr>
<tr>
<td>2005</td>
<td>17738</td>
<td>22.2</td>
<td>dual-core Xeon</td>
</tr>
<tr>
<td>2006</td>
<td>16865</td>
<td>9.7</td>
<td>dual-core Opteron 2.4Ghz</td>
</tr>
<tr>
<td>2007</td>
<td>16865</td>
<td>8.1</td>
<td>dual-core Opteron 3.1Ghz</td>
</tr>
<tr>
<td>2008</td>
<td>16865</td>
<td>8.7</td>
<td>quad-core Opteron 3.1Ghz</td>
</tr>
<tr>
<td>2014</td>
<td>16865</td>
<td>5.9</td>
<td>quad-core Intel I7 2.9Ghz</td>
</tr>
</tbody>
</table>
Multi-core architectures have revolutionized computing.

- Sparse updating methods are difficult to speed up
- ⇒ incentive to shift the emphasis from:
  
  \[\text{sparse matrix updating} \rightarrow \text{sparse matrix factorization}\]
Multi-core architectures have revolutionized computing.

- Sparse updating methods are difficult to speed up

  ⇒ incentive to shift the emphasis from:

  \[ \text{sparse matrix updating} \rightarrow \text{sparse matrix factorization} \]
Let the experts take care of it!

- Exploit state-of-the-art software from third-party solvers
- Reduce the reliance on customized software
Let the experts take care of it!

- Exploit state-of-the-art software from third-party solvers
- Reduce the reliance on customized software
Focus on methods that solve systems of the form $Kv = f$.

- *Interior methods* are already formulated to factor a fixed $K$.

- *SQP methods* can be reformulated in a similar way.
Each QP subproblem involves the “Newton equations” with matrix:

\[ K = \begin{pmatrix} H_F & J_F^T \\ J_F & 0 \end{pmatrix}, \]

where \( H_F \) and \( J_F \) are formed from columns associated with variables that are free to move.

As the QP iterations proceed:

- \( J_F \) has column \( a_s \) added, or column \( a_t \) deleted.
- \( H_F \) has a row and column added or deleted.
Updates without altering $K$

Bisschop and Meeraus 1977

Given $K$, quantities for the next iteration may be found by solving a bordered system with matrices:

\[
\begin{pmatrix}
H_F & J_F^T & h_t \\
J_F & a_t & h_{tt}
\end{pmatrix}
\]

(add column $a_t$)

\[
\begin{pmatrix}
H_F & J_F^T & e_s \\
J_F & 0 & 0
\end{pmatrix}
\]

(delete column $a_s$)
Schur complement QP method

These changes imply that an initial $K_0$ is bordered by a row and column at each step.

In general,

$$K_j v = f \equiv \begin{pmatrix} K_0 & W \\ W^T & W \end{pmatrix} \begin{pmatrix} v_1 \\ v_2 \end{pmatrix} = \begin{pmatrix} f_1 \\ f_2 \end{pmatrix}$$

1 solve with dense Schur-complement $C = -W^T K_0^{-1} W$

2 solves with $K_0$

Third-party solvers may be used to factor $K_0$

(G, Murray, Saunders & Wright 1990)
(Huynh 2007), (Maes 2010), (G & Wong 2011)
Schur complement QP method

These changes imply that an initial $K_0$ is bordered by a row and column at each step.

In general,

$$K_j v = f \equiv \begin{pmatrix} K_0 & W \\ W^T & \end{pmatrix} \begin{pmatrix} v_1 \\ v_2 \end{pmatrix} = \begin{pmatrix} f_1 \\ f_2 \end{pmatrix}$$

1 solve with dense Schur-complement $C = -W^T K_0^{-1} W$

2 solves with $K_0$

Third-party solvers may be used to factor $K_0$

(G, Murray, Saunders & Wright 1990)
(Huynh 2007), (Maes 2010), (G & Wong 2011)
Schur complement QP method

These changes imply that an initial $K_0$ is bordered by a row and column at each step.

In general,

$$K_j v = f \equiv \begin{pmatrix} K_0 & W \\ W^T & 0 \end{pmatrix} \begin{pmatrix} v_1 \\ v_2 \end{pmatrix} = \begin{pmatrix} f_1 \\ f_2 \end{pmatrix}$$

1 solve with dense Schur-complement $C = -W^T K_0^{-1} W$

2 solves with $K_0$

Third-party solvers may be used to factor $K_0$

(G, Murray, Saunders & Wright 1990)
(Huynh 2007), (Maes 2010), (G & Wong 2011)
Regularization
The Need for Regularization

- Symmetric KKT systems from optimization are hard to solve!
  - Gould, Hu & Scott (2007) report that no solver worked on all 61 indefinite systems tested!

- On a degenerate optimization problem, the matrices
  \[
  \begin{pmatrix}
  H_F & J_F^T \\
  J_F & 0
  \end{pmatrix}
  \text{ and } \begin{pmatrix}
  H + ZX^{-1} & J^T \\
  J & 0
  \end{pmatrix}
  \]
  are ill-conditioned or singular.

  The local convergence rate is no longer superlinear.
The earliest *regularized methods* perturb the KKT system. i.e., fix some small $\mu$ and solve the perturbed equations:

$$
\begin{pmatrix}
H_F & J_F^T \\
J_F & -\mu I
\end{pmatrix}
\begin{pmatrix}
\Delta x_F \\
-\Delta y_k
\end{pmatrix}
= -
\begin{pmatrix}
[g_k - J_k^T y_k]_F \\
c_k
\end{pmatrix}.
$$

(Saunders '96, Altman & Gondzio '99)

This is called *dual regularization*.

Primal regularization (not discussed here) would be applied to $H_F$. 
Again, it helps to start with a simpler problem.

We derive regularized methods for equality constraints:

$$\min_{x \in \mathbb{R}^n} f(x) \quad \text{subject to} \quad c(x) = 0.$$
Recall that the optimality conditions are:

\[ g(x) - J(x)^T y = 0 \]
\[ c(x) = 0. \]

For a given \( \mu \ll 1 \), and an estimate \( y^E \) of the optimal multipliers, consider the perturbed conditions:

\[ g(x) - J(x)^T y = 0 \]
\[ c(x) + \mu(y - y^E) = 0. \]

An optimal \( (x^*, y^*) \) solves \( F(x, y) = 0 \), with

\[ F(x, y) = \begin{pmatrix} g(x) - J(x)^T y \\ - (c(x) + \mu(y - y^E)) \end{pmatrix}. \]
Recall that the optimality conditions are:

\[ g(x) - J(x)^T y = 0 \]
\[ c(x) = 0. \]

For a given \( \mu \ll 1 \), and an estimate \( y^E \) of the optimal multipliers, consider the perturbed conditions:

\[ g(x) - J(x)^T y = 0 \]
\[ c(x) + \mu(y - y^E) = 0. \]

An optimal \((x^*, y^*)\) solves \( F(x, y) = 0 \), with

\[ F(x, y) = \begin{pmatrix} g(x) - J(x)^T y \\ -(c(x) + \mu(y - y^E)) \end{pmatrix}. \]
In this case, the Newton equations for the change in \((x_k, y_k)\) are:

\[
\begin{pmatrix}
H_k & J_k^T \\
J_k & -\mu I
\end{pmatrix}
\begin{pmatrix}
\Delta x_k \\
-\Delta y_k
\end{pmatrix}
= -
\begin{pmatrix}
g_k - J_k^T y_k \\
c_k + \mu (y_k - y^E)
\end{pmatrix},
\]

which are the optimality conditions for a solution \((x_k + \Delta x_k, y_k + \Delta y_k)\) of the QP:

\[
\begin{align*}
\text{minimize} & \quad g_k^T (x - x_k) + \frac{1}{2} (x - x_k)^T H_k (x - x_k) + \frac{1}{2} \mu \|y\|^2 \\
\text{subject to} & \quad c_k + \mu (y - y^E) + J_k (x - x_k) = 0.
\end{align*}
\]

In this case, the QP subproblem involves both \(x\) and \(y\).
In this case, the Newton equations for the change in \((x_k, y_k)\) are:

\[
\begin{pmatrix} H_k & J_k^T \\ J_k & -\mu I \end{pmatrix} \begin{pmatrix} \Delta x_k \\ -\Delta y_k \end{pmatrix} = - \begin{pmatrix} g_k - J_k^T y_k \\ c_k + \mu (y_k - y^E) \end{pmatrix},
\]

which are the optimality conditions for a solution \((x_k + \Delta x_k, y_k + \Delta y_k)\) of the QP:

\[
\begin{array}{l}
\text{minimize } \quad g_k^T(x - x_k) + \frac{1}{2}(x - x_k)^T H_k (x - x_k) + \frac{1}{2} \mu \|y\|^2 \\
\text{subject to } \quad c_k + \mu (y - y^E) + J_k (x - x_k) = 0.
\end{array}
\]

In this case, the QP subproblem involves both \(x\) and \(y\).
For small $\mu$, or $y^E \approx y^*$, the equations
\[
\begin{pmatrix}
H_k & J_k^T \\
J_k & -\mu I
\end{pmatrix}
\begin{pmatrix}
\Delta x_k \\
-\Delta y_k
\end{pmatrix}
= -
\begin{pmatrix}
g_k - J_k^T y_k \\
c_k + \mu(y_k - y^E)
\end{pmatrix}.
\]
are a perturbation of the Newton-Lagrange equations:
\[
\begin{pmatrix}
H_k & J_k^T \\
J_k & 0
\end{pmatrix}
\begin{pmatrix}
\Delta x_k \\
-\Delta y_k
\end{pmatrix}
= -
\begin{pmatrix}
g_k - J_k^T y_k \\
c_k
\end{pmatrix}.
\]

The parameter $\mu$ regularizes the Newton-Lagrange equations.

It is not necessary for $J_k$ to have full rank.
For small $\mu$, or $y^E \approx y^*$, the equations

$$
\begin{pmatrix}
H_k & J_k^T \\
J_k & -\mu I
\end{pmatrix}
\begin{pmatrix}
\Delta x_k \\
-\Delta y_k
\end{pmatrix}
= -
\begin{pmatrix}
g_k - J_k^T y_k \\
c_k + \mu(y_k - y^E)
\end{pmatrix}.
$$

are a perturbation of the Newton-Lagrange equations:

$$
\begin{pmatrix}
H_k & J_k^T \\
J_k & 0
\end{pmatrix}
\begin{pmatrix}
\Delta x_k \\
-\Delta y_k
\end{pmatrix}
= -
\begin{pmatrix}
g_k - J_k^T y_k \\
c_k
\end{pmatrix}
$$

The parameter $\mu$ regularizes the Newton-Lagrange equations.

*It is not necessary for $J_k$ to have full rank.*
minimize \( \begin{aligned} & x, y & \end{aligned} \)  \( g_k^T(x - x_k) + \frac{1}{2}(x - x_k)^T H_k(x - x_k) + \frac{1}{2} \mu \|y\|^2 \)

subject to \( \begin{aligned} & c_k + \mu(y - y^E) + J_k(x - x_k) = 0. \end{aligned} \)

Substituting the constraint into the objective gives

\[
\begin{aligned}
\min_{x, y} & \quad g_k^T(x - x_k) + \frac{1}{2}(x - x_k)^T H_k(x - x_k) \\
& \quad - (c_k + J_k(x - x_k))^T y^E + \frac{1}{2\mu} \|c_k + J_k(x - x_k)\|^2 \\
& \quad + \frac{1}{2\mu} \|c_k + J_k(x - x_k) + \mu(y - y^E)\|^2.
\end{aligned}
\]

This is an \textit{unconstrained} quadratic function of \( x \) and \( y \).
This is a **local quadratic model** for the augmented Lagrangian:

\[
M(x, y; \mu, y^E) = f(x) - c(x)^T y^E + \frac{1}{2\mu} \|c(x)\|^2 \\
+ \frac{1}{2\mu} \|c(x) + \mu(y - y^E)\|^2.
\]

\[\Rightarrow\] \(M(x, y; \mu, y^E)\) is a “natural” merit function for the perturbed method of Newton-Lagrange.

\[\Rightarrow\] \(M(x, y; \mu, y^E)\) is a “natural” merit function for the perturbed SQP method.
Regularized SQP Methods

Implementation:

- $\mu_M$ is used to define the merit function $M(x, y ; \mu_M, y^E)$.
- $\mu \ll 1$ is a fixed regularization parameter.
- $\mu$ is used to define the QP subproblem ($\mu \ll \mu_M$).
- The QP equations are

$$
\begin{pmatrix}
H_k & J_k^T \\
J_k & -\mu I
\end{pmatrix}
\begin{pmatrix}
\Delta x_k \\
-\Delta y_k
\end{pmatrix}
= -\begin{pmatrix}
g_k - J_k^T y_k \\
c_k + \mu(y_k - y^E)
\end{pmatrix}.
$$

- $y^E = y_{k+1}$ if a certain proximity measure is satisfied.

The regularization is now an integral part of the method.

(G & Robinson, 2013)
In the inequality constrained case, the QP subproblem is:

\[
\begin{align*}
\text{minimize} & \quad g_k^T(x - x_k) + \frac{1}{2}(x - x_k)^T H_k (x - x_k) + \frac{1}{2} \mu \|y\|^2 \\
\text{subject to} & \quad c_k + \mu(y_k - y^E) + J_k(x - x_k) = 0, \quad x \geq 0.
\end{align*}
\]

A point \((x, y, z)\) satisfying the optimality conditions solves the bound-constrained problem

\[
\begin{align*}
\text{minimize} & \quad M(x, y; \mu, y^E) \quad \text{subject to} \quad x \geq 0.
\end{align*}
\]
Regularized SQP Methods

Properties:

- Global convergence under weak assumptions.
  
  augmented Lagrangian method “kicks in” as a “fall-back.”

- With $y^E = y_{k+1}$, the iterates $(x_k, y_k)$ converge at a Q-superlinear rate.

(G, Kungurtsev & Robinson, 2014)
Regularized Interior Methods

Similar ideas may be applied to interior methods.

Recall that the optimality conditions are

\[
\begin{align*}
    c(x) &= 0, \quad x \geq 0, \\
    g(x) - J(x)^T y - z &= 0, \quad z \geq 0, \\
    x \cdot z &= 0.
\end{align*}
\]

Given estimates \((y^E, z^E)\) of the optimal \((y^*, z^*)\), consider

\[
\begin{align*}
    c(x) + \mu(y - y^E) &= 0, \quad x + \mu e > 0, \\
    g(x) - J(x)^T y - z &= 0, \quad z > 0, \\
    x \cdot z &= \mu(z^E - z).
\end{align*}
\]
This leads to a regularized system with matrix:

\[
\begin{pmatrix}
H_k + ZX\mu^{-1} & J_k^T \\
J_k & -\mu I
\end{pmatrix}
\begin{pmatrix}
\Delta x_k \\
\Delta y_k
\end{pmatrix}
= - \begin{pmatrix}
g_k - J_k^T y_k - \mu X\mu^{-1} z^E \\
c_k + \mu (y_k - y^E)
\end{pmatrix}.
\]

where \( X\mu = X + \mu I \).
The corresponding *unconstrained* merit function is then:

\[
M(x, z, y; \mu, y^E, z^E) = f(x) - c(x)^T y^E + \frac{1}{2\mu} \|c(x)\|^2 \\
+ \frac{1}{2\mu} \|c(x) + \mu(y - y^E)\|^2 \\
- \sum_{i=1}^{n} \mu z_i^E \ln \left( x_i + \mu \right) \\
- \sum_{i=1}^{n} \mu z_i^E \ln \left( (x_i + \mu)z_i \right) \\
- \sum_{i=1}^{n} \left( \mu(z_i^E - z_i) - x_i z_i \right).
\]

This combines an augmented Lagrangian and shifted barrier.

As in the SQP case, two parameters $\mu$ and $\mu_M$ are used, $\mu \ll \mu_M$. 
Summary

Recent improvements in computer hardware have necessitated a major change in how linear algebra should be done in nonlinear optimization solvers.

In particular, there has been a shift towards the development of algorithms that can be implemented using third-party linear solvers.

This shift has required the formulation of new optimization methods that use dual regularization to resolve the numerical and theoretical difficulties associated with ill-posed or degenerate problems.
Thanks for listening!
One last thought:

Two "Math for Dummies" at $16.99 each. That'll be $50.
References

