A New Penalty-SQP Method

Frank E. Curtis
New York University

Informs Annual Meeting, October 2008
Outline

Background and Motivation

A New Penalty-SQP Method

Illustration of Numerical Results

Final Remarks
Sequential Quadratic Programming

We solve the nonlinear optimization problem

$$\begin{align*}
\min & \ f(x) \\
\text{s.t.} & \ b(x) \geq 0 \\
& \ c(x) = 0,
\end{align*}$$

where $f : \mathbb{R}^n \rightarrow \mathbb{R}$, $b : \mathbb{R}^n \rightarrow \mathbb{R}^l$ and $c : \mathbb{R}^n \rightarrow \mathbb{R}^m$ are smooth, via

$$\begin{align*}
\min & \ f_k + \nabla f_k^T d_k + \frac{1}{2} d_k^T (H_k + \tilde{H}_k) d_k \\
\text{s.t.} & \ b_k + \nabla b_k^T d_k \geq 0 \\
& \ c_k + \nabla c_k^T d_k = 0
\end{align*}$$

and promote convergence with the $\ell_1$ penalty function

$$\phi(x; \rho) \triangleq \rho f(x) + \sum_{i=1}^{l} \max\{0, -b^i(x)\} + \sum_{j=1}^{m} |c^j(x)|.$$
Handling infeasible subproblems

We have various options when a given subproblem is infeasible

- Feasibility restoration: replace the subproblem with

\[
\begin{align*}
\min & \quad \sum_{i=1}^l r^i + \sum_{j=1}^m (s^j + t^j) + \frac{1}{2} d_k^T \tilde{H}_k d_k \\
\text{s.t.} & \quad b_k + \nabla b_k^T d_k \geq -r \\
& \quad c_k + \nabla c_k^T d_k = -s + t
\end{align*}
\]

(... but what if we move away from optimality?)

- Constraint relaxation: replace the subproblem with

\[
\begin{align*}
\min & \quad \rho \left( f_k + \nabla f_k^T d_k + \frac{1}{2} d_k^T H_k d_k \right) + \sum_{i=1}^l r^i + \sum_{j=1}^m (s^j + t^j) + \frac{1}{2} d_k^T \tilde{H}_k d_k \\
\text{s.t.} & \quad b_k + \nabla b_k^T d_k \geq -r \\
& \quad c_k + \nabla c_k^T d_k = -s + t
\end{align*}
\]

(... but what if we move away from feasibility?)
Penalty-SQP Capabilities

<table>
<thead>
<tr>
<th>$\ell_1$QP</th>
<th>Feasible</th>
<th>Infeasible</th>
</tr>
</thead>
<tbody>
<tr>
<td>Global</td>
<td>Fast</td>
<td>Local</td>
</tr>
</tbody>
</table>
### Penalty-SQP Capabilities

<table>
<thead>
<tr>
<th>$\ell_1$QP</th>
<th>Feasible</th>
<th>Infeasible</th>
</tr>
</thead>
<tbody>
<tr>
<td>Global</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Fast Local</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Penalty-SQP Capabilities

<table>
<thead>
<tr>
<th>$S\ell_1$QP</th>
<th>Feasible</th>
<th>Infeasible</th>
</tr>
</thead>
<tbody>
<tr>
<td>Global</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Fast Local</td>
<td>✓</td>
<td></td>
</tr>
</tbody>
</table>
## Penalty-SQP Capabilities

<table>
<thead>
<tr>
<th>$\ell_1$QP</th>
<th>Feasible</th>
<th>Infeasible</th>
</tr>
</thead>
<tbody>
<tr>
<td>Global</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Fast Local</td>
<td>✓</td>
<td></td>
</tr>
</tbody>
</table>
## Penalty-SQP Capabilities

<table>
<thead>
<tr>
<th>$\ell_1$QP</th>
<th>Feasible</th>
<th>Infeasible</th>
</tr>
</thead>
<tbody>
<tr>
<td>Global</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Fast Local</td>
<td>✓</td>
<td>???</td>
</tr>
</tbody>
</table>
MINLP methods
MINLP methods

- $P_3$: $x_3 = 0$
- $P_1$: $x_1 = 0$, (infeasible)
- $P^3$: $x_3 = 1$, $x_1 = 1$
- $P^3,1$

- NLP

- ???
Summary

- There is a need for algorithms that converge quickly, regardless of whether the problem is feasible or infeasible.
- Interior-point methods are known to perform poorly in infeasible cases, but active set methods seem promising.
- Room for improvement in active set methods, too.
- Feasibility restoration techniques are an option, but we prefer a single algorithm with no heuristics.
A single algorithmic framework

Our goal is to design a single optimization algorithm designed to solve

\[
(NLP) \triangleq \begin{cases} 
\min f(x) \\
\text{s.t. } b(x) \geq 0 \\
\quad c(x) = 0
\end{cases}
\]

or, if \((NLP)\) is infeasible,

\[
(FEAS) \triangleq \begin{cases} 
\min v(x) \triangleq \sum_{i=1}^{l} \max\{0, -b^i(x)\} + \sum_{j=1}^{m} |c^j(x)|
\end{cases}
\]

We combine \((NLP)\) and \((FEAS)\) to define

\[
(P) \triangleq \{\min \rho f(x) + v(x)\}
\]

where \(\rho \geq 0\) is a penalty parameter to be updated dynamically.
Our method for step computation and acceptance

We generate a step via

$$\min q_k(d_k; \rho) \triangleq \rho \left( f_k + \nabla f_k^T d_k + \frac{1}{2} d_k^T H_k d_k \right) + \sum_{i=1}^l \max \left\{ 0, -b^i_k - \nabla b^i_k^T d_k \right\} + \sum_{j=1}^m \left| c^j_k + \nabla c^j_k^T d_k \right| + \frac{1}{2} d_k^T \tilde{H}_k d_k$$

which is equivalent to the quadratic subproblem

$$\min \rho \left( f_k + \nabla f_k^T d_k + \frac{1}{2} d_k^T H_k d_k \right) + \sum_{i=1}^l r^i + \sum_{j=1}^m (s^j + t^j) + \frac{1}{2} d_k^T \tilde{H}_k d_k$$

$$\tag{Q_{\rho}}$$

s.t. $b_k + \nabla b_k^T d_k \geq -r$

$c_k + \nabla c_k^T d_k = -s + t$

and we measure progress with the exact penalty function

$$\phi(x; \rho) \triangleq \rho f(x) + v(x)$$
A New Penalty-SQP Method

A Penalty-SQP framework

Step 0. Initialize $x_0$ and set $\eta \in (0, 1)$, $\tau \in (0, 1)$ and $k \leftarrow 0$

Step 1. If $x_k$ solves ($NLP$) or ($FEAS$), then stop

Step 2. Compute a value for the penalty parameter, call it $\rho_k$

Step 3. Compute $d_k$ by solving ($Q_\rho$) with $\rho \leftarrow \rho_k$

Step 4. Let $\alpha_k$ be the first member of the sequence \{1, $\tau$, $\tau^2$, ...\} s.t.

$$\phi(x_k; \rho_k) - \phi(x_k + \alpha_k d_k; \rho_k) \geq \eta \alpha_k [q_k(0; \rho_k) - q_k(d_k; \rho_k)]$$

Step 5. Update $x_{k+1} \leftarrow x_k + \alpha_k d_k$, go to Step 1
A Penalty-SQP framework

Step 0. Initialize $x_0$ and set $\eta \in (0, 1)$, $\tau \in (0, 1)$ and $k \leftarrow 0$

Step 1. If $x_k$ solves (NLP) or (FEAS), then stop

Step 2. Compute a value for the penalty parameter, call it $\rho_k$

Step 3. Compute $d_k$ by solving $(Q_\rho)$ with $\rho \leftarrow \rho_k$

Step 4. Let $\alpha_k$ be the first member of the sequence $\{1, \tau, \tau^2, \ldots\}$ s.t.

$$\phi(x_k; \rho_k) - \phi(x_k + \alpha_k d_k; \rho_k) \geq \eta \alpha_k [q_k(0; \rho_k) - q_k(d_k; \rho_k)]$$

Step 5. Update $x_{k+1} \leftarrow x_k + \alpha_k d_k$, go to Step 1
Steering Rules

We extend the *steering rules* for setting $\rho_k^1$

1. First, we guarantee first-order progress toward minimizing the feasibility violation measure $v(x)$ during each iteration

---

Steering Rules

We extend the *steering rules* for setting $\rho_k$.

2. Second, we avoid settling on a penalty function with $\rho$ too large by ensuring that progress in $\phi$ is proportional to the largest *potential* progress in feasibility.
Penalty parameter update

a. If the current iterate is *feasible*, then the algorithm should reduce $\rho$, if necessary, to ensure linearized feasibility; i.e.,

$$l_k(d_k) \triangleq \sum_{i=1}^{l} \max \left\{ 0, -b^i_k - \nabla b^i_k \, d_k \right\} + \sum_{j=1}^{m} \left| c^j_k + \nabla c^j_k \, d \right| = 0$$

b. Otherwise, $\rho$ should be set so that

$$l_k(0) - l_k(d_k) \geq \epsilon_1 (l_k(0) - l_k(\bar{d}_k))$$
$$q_k(0; \rho_k) - q_k(d_k; \rho_k) \geq \epsilon_2 (q_k(0; 0) - q_k(\bar{d}_k; 0))$$

where the *reference direction* is given by

$$\bar{d}_k \triangleq \arg \min (Q_\rho) \text{ for } \rho = 0$$
Penalty parameter update, extended

a. If the current iterate is feasible, then the algorithm should reduce $\rho$, if necessary, to ensure linearized feasibility; i.e.,

$$l_k(d_k) \triangleq \sum_{i=1}^{l} \max \left\{ 0, -b_i^i - \nabla b_i^i d_k \right\} + \sum_{j=1}^{m} |c_j^j + \nabla c_j^j d| = 0$$

b. Otherwise, $\rho$ should be set so that

$$l_k(0) - l_k(d_k) \geq \epsilon_1 (l_k(0) - l_k(\bar{d}_k))$$

$$q_k(0; \rho_k) - q_k(d_k; \rho_k) \geq \epsilon_2 (q_k(0; 0) - q_k(\bar{d}_k; 0))$$

c. Finally, if $\rho$ is decreased we ensure

$$\rho_k \leq \epsilon_3 \|KKT\; error \; for \; (FEAS)\|^2$$
Strategy for fast convergence

Hitting a moving target:

\[ x_k \rightarrow x_\rho \rightarrow \hat{x} \]

where

\[ x_k \triangleq k\text{th iterate of the algorithm} \]
\[ x_\rho \triangleq \text{solution of penalty problem (}P\text{)} \]
\[ \hat{x} \triangleq \text{infeasible stationary point of (}NLP\text{), solution of (}FEAS\text{)} \]

It has been shown\(^2\) that, for some \(C, C' > 0\),

\[
\| x_{k+1} - \hat{x} \| \leq \| x_{k+1} - x_\rho \| + \| x_\rho - \hat{x} \| \\
\leq C \| x_k - x_\rho \|^2 + O(\rho) \\
\leq C' \| x_k - \hat{x} \|^2 + O(\rho),
\]

so convergence is quadratic if \(\rho \propto \| x_k - \hat{x} \|^2\)

\(^2\)Byrd, Curtis, Nocedal (2008)
Strategy for fast convergence

Hitting a moving target:

\[ x_k \rightarrow x_\rho \rightarrow \hat{x} \]

where

\( x_k \triangleq \) kth iterate of the algorithm
\( x_\rho \triangleq \) solution of penalty problem \((P)\)
\( \hat{x} \triangleq \) infeasible stationary point of \((NLP)\), solution of \((FEAS)\)
Strategy for fast convergence

Hitting a moving target:

\[ x_k \rightarrow x_\rho \rightarrow \hat{x} \]

where

\[ x_k \triangleq \text{kth iterate of the algorithm} \]
\[ x_\rho \triangleq \text{solution of penalty problem (P)} \]
\[ \hat{x} \triangleq \text{infeasible stationary point of (NLP), solution of (FEAS)} \]
An infeasible problem with a strict minimizer of infeasibility

\[
\begin{align*}
\min & \quad x_1 + x_2 \\
\text{s.t.} & \quad x_2 - x_1^2 - 1 \geq 0 \\
& \quad 0.3(1 - e^{x_2}) \geq 0
\end{align*}
\]

<table>
<thead>
<tr>
<th>$k$</th>
<th>$f_k$</th>
<th>$v_k$</th>
<th>$E_k(\rho_{k-1})$</th>
<th>$E_k(0)$</th>
<th>$\rho_k$</th>
<th>$|d_k|$</th>
<th>$\alpha_k$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>5.00e+00</td>
<td>9.92e+00</td>
<td>8.84e+00</td>
<td>7.51e+00</td>
<td>1.00e+00</td>
<td>1.71e+00</td>
<td>1.00e+00</td>
</tr>
<tr>
<td>1</td>
<td>2.66e+00</td>
<td>2.82e+00</td>
<td>3.51e+00</td>
<td>2.23e+00</td>
<td>1.00e-02</td>
<td>1.16e+00</td>
<td>1.00e+00</td>
</tr>
<tr>
<td>2</td>
<td>1.04e+00</td>
<td>1.06e+00</td>
<td>9.14e-01</td>
<td>9.07e-01</td>
<td>1.00e-02</td>
<td>1.13e+00</td>
<td>5.00e-01</td>
</tr>
<tr>
<td>3</td>
<td>3.20e-01</td>
<td>8.23e-01</td>
<td>4.55e-01</td>
<td>4.53e-01</td>
<td>1.00e-02</td>
<td>7.76e-01</td>
<td>1.00e+00</td>
</tr>
<tr>
<td>4</td>
<td>6.74e-01</td>
<td>6.09e-01</td>
<td>4.33e-01</td>
<td>4.38e-01</td>
<td>1.00e-02</td>
<td>4.66e-01</td>
<td>1.00e+00</td>
</tr>
<tr>
<td>5</td>
<td>9.72e-01</td>
<td>5.81e-01</td>
<td>2.84e-01</td>
<td>2.73e-01</td>
<td>1.00e-02</td>
<td>2.29e-01</td>
<td>1.00e+00</td>
</tr>
<tr>
<td>6</td>
<td>9.39e-01</td>
<td>5.22e-01</td>
<td>4.52e-02</td>
<td>5.49e-02</td>
<td>1.00e-03</td>
<td>4.24e-02</td>
<td>1.00e+00</td>
</tr>
<tr>
<td>7</td>
<td>9.99e-01</td>
<td>5.16e-01</td>
<td>1.50e-03</td>
<td>9.19e-04</td>
<td>8.45e-07</td>
<td>9.42e-04</td>
<td>1.00e+00</td>
</tr>
<tr>
<td>8</td>
<td>1.00e+00</td>
<td>5.15e-01</td>
<td>4.79e-08</td>
<td>8.93e-07</td>
<td>---------</td>
<td>---------</td>
<td>---------</td>
</tr>
</tbody>
</table>
## Problem batch as a MINLP

| $k$ | $f_k$       | $v_k$       | $E_k(\rho_{k-1})$ | $E_k(0)$  | $\rho_k$ | $||d_k||$ | $\alpha_k$ |
|-----|-------------|-------------|-------------------|-----------|----------|----------|------------|
| 7   | +6.28e+04  | 1.89e+00   | 4.16e-02          | 1.65e+00 | 1.00e-05 | 1.50e+00 | 1.00e+00  |
| 8   | +9.33e+04  | 2.88e-01   | 2.50e-01          | 3.44e-01 | 1.00e-05 | 9.55e-01 | 1.00e+00  |
| 9   | +1.03e+05  | 2.07e-02   | 1.86e-02          | 2.44e-02 | 1.00e-05 | 2.17e-01 | 1.00e+00  |
| 10  | +1.04e+05  | 4.24e-04   | 3.59e-04          | 4.99e-04 | 2.49e-07 | 1.58e-03 | 1.00e+00  |
| 11  | +1.04e+05  | 3.76e-08   | 5.06e-07          | ---------| ---------| ---------| 1.00e+00  |

Adding the constraint $t/|1| \geq 5$ creates an infeasible problem:

| $k$ | $f_k$       | $v_k$       | $E_k(\rho_{k-1})$ | $E_k(0)$  | $\rho_k$ | $||d_k||$ | $\alpha_k$ |
|-----|-------------|-------------|-------------------|-----------|----------|----------|------------|
| 11  | +2.27e+05  | 3.09e+00   | 3.92e-02          | 3.41e-02 | 1.00e-06 | 3.68e-01 | 1.00e+00  |
| 12  | +2.27e+05  | 3.07e+00   | 3.87e-04          | 4.32e-02 | 1.00e-07 | 7.74e-01 | 1.00e+00  |
| 13  | +2.66e+05  | 3.04e+00   | 6.76e-03          | 6.79e-03 | 1.00e-07 | 2.24e-01 | 1.00e+00  |
| 14  | +2.66e+05  | 3.04e+00   | 1.19e-05          | 1.20e-05 | 1.44e-10 | 1.30e-01 | 1.00e+00  |
| 15  | +2.66e+05  | 3.04e+00   | 7.82e-09          | 6.02e-09 | ---------| ---------| 1.00e+00  |
Summary

- We have argued for the need of optimization algorithms that converge quickly, even on infeasible problems.
- We have presented a penalty-SQP approach that transitions smoothly between solving an optimization problem and the corresponding feasibility problem.
- The novel components of the algorithm are:
  1. the quadratic model $q_k$ (it is quadratic even for $\rho = 0$)
  2. an appropriate update for $\rho$ to ensure superlinear convergence on infeasible problem instances
- We have illustrated that the algorithm performs well on both feasible and infeasible problem instances.
Penalty-SQP Capabilities Now

<table>
<thead>
<tr>
<th>$\ell_1$QP</th>
<th>Feasible</th>
<th>Infeasible</th>
</tr>
</thead>
<tbody>
<tr>
<td>Global</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Fast Local</td>
<td>✓</td>
<td>✓✓✓</td>
</tr>
</tbody>
</table>

Questions?

Thanks!