Solving minimax problems with feasible sequential quadratic programming
Feasible sequential quadratic programming (FSQP) refers to a class of sequential quadratic programming methods.

Engineering applications: number of variables is not large; evaluations of objective or constraint functions and of the gradients are time consuming.

Advantages:

1) Generating feasible iterates.
2) Reducing the amount of computation.
3) Enjoying the same global and fast local convergence properties.
Background

- The constrained mini-max problem

Minimize \( \max_{i \in I} \{ f_i(x) \} \) s.t. \( x \in X \)

\( X \) is the set of points \( x \in \mathbb{R}^n \) satisfying

\[
\begin{align*}
bl & \leq x \leq bu \\
g_j(x) & \leq 0, \quad j = 1, \ldots, n_i \\
g_j(x) & \equiv \langle c_{j-n_i}, x \rangle - d_{j-n_i} \leq 0, \quad j = n_i + 1, \ldots, t_i \\
h_j(x) & = 0, \quad j = 1, \ldots, n_e \\
h_j(x) & \equiv \langle a_{j-n_e}, x \rangle - b_{j-n_e} = 0, \quad j = n_e + 1, \ldots, t_e
\end{align*}
\]

* Bounds
* Nonlinear inequality
* Linear inequality
* Nonlinear equality
* Linear equality
Algorithm - FSQP

* Step 1. Initialization
* Step 2. A search arc
* Step 3. Arc search
* Step 4. Updates
Initialization \((x, H, p, k)\)

A feasible point \(x_0\)

**Initial guess** \(x_{00}\)

<table>
<thead>
<tr>
<th>Infeasible for linear constraints</th>
<th>Strictly convex quadratic program</th>
</tr>
</thead>
<tbody>
<tr>
<td>Infeasible for the nonlinear inequality constraints</td>
<td>Armijo-type line search</td>
</tr>
</tbody>
</table>

Initial Hessian matrix \(H_0 = \text{the identity matrix},\)

\[ p_{0,j} = \varepsilon_2 \text{ for } j = 1, \ldots, n_e \]

Positive penalty parameters

\[ f_m(x, p) = \max_{i \in I} \{ f_i(x) \} - \sum_{j=1}^{n_e} p_j h_j(x) \]

Iteration index \(k = 0\)
Computation of a search arc

1. Compute $d_k^0$

2. Compute $d_k^1$

3. Set $d_k = (1 - \rho_k) d_k^0 + \rho_k d_k^1$

4. Compute $\tilde{d}_k$

Stop: if $\| d_k^0 \| \leq \varepsilon$ and $\sum_{j=1}^{n_e} | h_j(x_k) | \leq \varepsilon_e$
Computation of a search arc

1. Compute $d^0_k$, solution of $QP(x_k, H_k, p_k)$

$$\min_{d^0} \frac{1}{2} \langle d^0, H_k d^0 \rangle + f'(x_k, d^0, p_k)$$

s.t.

$$bl \leq x_k + d^0 \leq bu$$

$$g_j(x_k) + \langle \nabla g_j(x_k), d^0 \rangle \leq 0, \quad j = 1, \ldots, t_i$$

$$h_j(x_k) + \langle \nabla h_j(x_k), d^0 \rangle \leq 0, \quad j = 1, \ldots, n_e$$

$$\langle a_j, x_k + d^0 \rangle = b_j \quad j = 1, \ldots, t_e - n_e$$

$$f'(x, d, p) = \max_{i \in I^f} \{ f_i(x) \} + \langle \nabla f_i(x), d \rangle - f_{i^*}(x) - \sum_{j=1}^{n_e} p_j \langle \nabla h_j(x), d \rangle$$

$$\tilde{f}_i(x) = \max_{i \in I^f} \{ f_i(x) \}$$

3. Set $d_k = (1 - \rho_k)d^0_k + \rho_k d^1_k$

$$\rho_k = \frac{\| d^0_k \|^\kappa}{\| d^0_k \|^\kappa + \nu_k}$$

$$\nu_k = \max(0.5, \| d^1_k \|^{\tau_1})$$

2. Compute $d^1_k$, solution of $QP(x_k, d^0_k, p_k)$

$$\min_{d^1 \in \mathbb{R}^n, p \in \mathbb{R}} \frac{\eta}{2} \langle d^0_k - d^1, d^0_k - d^1 \rangle + \gamma$$

s.t.

$$bl \leq x_k + d^1 \leq bu$$

$$f'(x_k, d^1, p_k) \leq \gamma$$

$$g_j(x_k) + \langle \nabla g_j(x_k), d^1 \rangle \leq \gamma \quad j = 1, \ldots, n_i$$

$$\langle c_j, x_k + d^1 \rangle \leq d_j \quad j = 1, \ldots, t_i - n_i$$

$$h_j(x_k) + \langle \nabla h_j(x_k), d^1 \rangle \leq \gamma \quad j = 1, \ldots, n_e$$

$$\langle a_j, x_k + d^1 \rangle = b_j \quad j = 1, \ldots, t_e - n_e$$

4. Compute $\tilde{d}_k$
Quadratic programming

Strictly convex quadratic programming:

- Unique global minimum
- Matrix $C$ need to be positive definite

\[
\text{Minimize } \frac{1}{2} X' CX + D' X \\
\text{Subject to } AX \leq B \\
X \geq 0
\]

$m = \text{number of constraints}$;
\n$n = \text{number of variables}$

\[
X_{n \times 1} = [x_1, x_2, \ldots, x_n]'
\]

$A_{m \times n}; B_{m \times 1}; C_{n \times n}; D_{n \times 1}$

Extended Wolfe’s simplex method

- No derivative

Quadratic programming

Lagrangian function: $L(x, \lambda) = \frac{1}{2} X'CX + D'X + \lambda(AX - B)$

Karush-Kuhn-Tucker conditions:

$$\begin{align*}
\frac{\partial L}{\partial \lambda} &= AX - B \leq 0 \\
\frac{\partial L}{\partial X} &= X'C + D' + \lambda A \geq 0; CX + D + A'\lambda' \geq 0 \\
\lambda \frac{\partial L}{\partial \lambda} &= \lambda(AX - B) = 0 \\
X'\frac{\partial L}{\partial X} &= X'(CX + D + A'\lambda') = 0
\end{align*}$$

$$\begin{align*}
AX + \nu &= B \\
CX + A'\lambda' - \mu + s &= -D \\
\lambda \nu &= 0 \\
\mu'X &= 0 \\
X \geq 0, \lambda \geq 0, \mu \geq 0, \nu \geq 0, s \geq 0
\end{align*}$$

$\nu$ = slack variables; $\mu$ = surplus variables
$s$ = artificial variables

$(\lambda, \nu)$ and $(\mu', X)$ = complementary slack variables
Linear programming

Quadratic programming:  \[ \Rightarrow \]  Conditioned linear programming:

Minimize \( \frac{1}{2} X'CX + D'X \)

Subject to \( AX \leq B \)

\( X \geq 0 \)

Minimize \( M' \cdot s \)

Subject to \( AX + \nu = B \)

\( CX + A'\lambda' - \mu + s = -D \)

\( X, \lambda, \mu, \nu, s \geq 0 \)

\((\lambda, \nu)\) and \((\mu, X)\) are complementary slack variables.

Simplex tableau:

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>n</th>
<th>m</th>
<th>m</th>
<th>n</th>
<th>m</th>
<th>n</th>
</tr>
</thead>
<tbody>
<tr>
<td>rhs</td>
<td></td>
<td>X</td>
<td>\lambda'</td>
<td>\mu</td>
<td>\nu</td>
<td></td>
<td>s</td>
</tr>
<tr>
<td>Z</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>\nu</td>
<td></td>
<td>B</td>
<td>A</td>
<td></td>
<td>I</td>
<td></td>
<td></td>
</tr>
<tr>
<td>s</td>
<td>- D</td>
<td>C</td>
<td>A'</td>
<td>- I</td>
<td>I</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\( \nu \) and \( s \) are basis variables
Test example

Minimize \( f(x) = x_1^2 + 2x_2^2 - 2x_1x_2 - 4x_1 \)

Subject to

\[
\begin{align*}
2x_1 + x_2 & \leq 6.0 \\
x_1 - 4x_2 & \leq 0.0 \\
x_1, x_2 & \geq 0.0
\end{align*}
\]

\[
X = \begin{bmatrix}
x_1 \\
x_2
\end{bmatrix}, \quad A = \begin{bmatrix}
2 & 1 \\
1 & -4
\end{bmatrix}, \quad B = \begin{bmatrix}
6 \\
0
\end{bmatrix},
\]

\[
C = \begin{bmatrix}
2 & -2 \\
-2 & 4
\end{bmatrix}, \quad D = \begin{bmatrix}
-4 \\
0
\end{bmatrix}
\]

\( m = 2 \) number of constraints; \( n = 2 \) number of variables

Optimal solution

\( x_1^* = 2.462, \quad x_2^* = 1.077, \quad f(x^*) = -6.769 \)
Arc search

\[ \delta_k = f'(x_k, d_k, p_k) \quad \text{If } n_i + n_e \neq 0 \text{ and } \delta_k = -\langle d_k^0, H_k d_k^0 \rangle. \]

Compute \( t_k \), the first number \( t \) in the sequence \( \{1, \beta, \beta^2, \ldots\} \) satisfying

\[
f_m(x_k + td_k + t^2 \tilde{d}_k, p_k) \leq f_m(x_k, p_k) + \alpha t \delta_k,
\]

\[ g_j(x_k + td_k + t^2 \tilde{d}_k) \leq 0, \quad j = 1, \ldots, n_i. \]

\[
\langle c_{j-n_i}, x_k + td_k + t^2 \tilde{d}_k \rangle \leq d_{j-n_i}, \quad \forall j > n_i \text{ and } j \notin I_k^g(d_k)
\]

\[ h_j(x_k + td_k + t^2 \tilde{d}_k) \leq 0, \quad j = 1, \ldots, n_e. \]
Updates \((x, H, p, k)\)

\[ x_{k+1} = x_k + t_k d_k + t_k^2 \tilde{d}_k \]

BFGS formula to compute \(H_{k+1} = H_{k+1}^T > 0\).

\[ p_{k+1, j} = \begin{cases} 
  p_{k,j} & p_{k,j} + \bar{\mu}_j \geq \varepsilon_1 \\
  \max\{\varepsilon_1 - \bar{\mu}_j, \delta p_{k,j}\} & \text{otherwise}
\end{cases} \]

Solve the unconstrained quadratic problem in \(\bar{\mu}\)

\[
\min_{\mu \in \mathbb{R}^{n_{\mu}}} \left\| \sum_{j=1}^{n_f} \zeta_{k,j} \nabla f_j(x_{k+1}) + \xi_k + \sum_{j=1}^{t_f} \lambda_{k,j} \nabla g_j(x_{k+1}) + \sum_{j=1}^{t_z} \mu_{k,j} \nabla h_j(x_{k+1}) + \sum_{j=1}^{n_k} \bar{\mu}_j \nabla h_j(x_{k+1}) \right\|^2
\]

\[ k = k + 1 \]
Updates

**BFGS formula with Powell’s modification:**

\[ H_{k+1} \] is the Hessian of the Lagrangian function \( f_m(x_k, p_k) \)

\[ \eta_{k+1} = x_{k+1} - x_k \]

\[ \gamma_{k+1} = \nabla_x f_m(x_{k+1}, p_k) - \nabla_x f_m(x_k, p_k) \]

\[ \xi_{k+1} = \theta_{k+1} \cdot \gamma_{k+1} + (1 - \theta_{k+1}) \cdot H_k \delta_{k+1} \]

\[ \theta_{k+1} = \begin{cases} 1, & \eta_{k+1}^T \gamma_{k+1} \geq 0.2 \eta_{k+1}^T H_k \eta_{k+1} \\ \frac{0.8 \eta_{k+1}^T H_k \eta_{k+1}}{\eta_{k+1}^T H_k \eta_{k+1} - \eta_{k+1}^T \gamma_{k+1}}, & \text{otherwise} \end{cases} \]

\[ H_{k+1} = H_k - \frac{H_k \eta_{k+1} \eta_{k+1}^T H_k}{\eta_{k+1}^T H_k \eta_{k+1}} + \frac{\xi_{k+1} \xi_{k+1}^T}{\eta_{k+1}^T \xi_{k+1} \eta_{k+1}} \]
## Project Schedule

<table>
<thead>
<tr>
<th>Date</th>
<th>Tasks</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>October</strong></td>
<td>• Literature review;</td>
</tr>
<tr>
<td></td>
<td>• Specify the implementation module details;</td>
</tr>
<tr>
<td></td>
<td>• Structure the implementation;</td>
</tr>
<tr>
<td><strong>November</strong></td>
<td>• Develop the quadratic programming module;</td>
</tr>
<tr>
<td></td>
<td>• Unconstrained quadratic program;</td>
</tr>
<tr>
<td></td>
<td>• Strictly convex quadratic program;</td>
</tr>
<tr>
<td></td>
<td>• Validate the quadratic programming module;</td>
</tr>
<tr>
<td><strong>December</strong></td>
<td>• Develop the Gradient and Hessian matrix calculation module;</td>
</tr>
<tr>
<td></td>
<td>• Validate the Gradient and Hessian matrix calculation module;</td>
</tr>
<tr>
<td></td>
<td>• Midterm project report and presentation;</td>
</tr>
<tr>
<td>Month</td>
<td>Tasks</td>
</tr>
<tr>
<td>---------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>January</td>
<td>- Develop Armijo line search module;</td>
</tr>
<tr>
<td></td>
<td>- Validate Armijo line search module;</td>
</tr>
<tr>
<td>February</td>
<td>- Develop the feasible initial point module;</td>
</tr>
<tr>
<td></td>
<td>- Validate the feasible initial point module;</td>
</tr>
<tr>
<td></td>
<td>- Integrate the program;</td>
</tr>
<tr>
<td>March</td>
<td>- Debug and document the program;</td>
</tr>
<tr>
<td></td>
<td>- Validate and test the program with case application;</td>
</tr>
<tr>
<td>April</td>
<td>- Add arch search variable $\tilde{d}$ in;</td>
</tr>
<tr>
<td></td>
<td>- Compare calculation efficiency of line search with arch search methods;</td>
</tr>
<tr>
<td>May</td>
<td>- Develop the user interface if time available;</td>
</tr>
<tr>
<td></td>
<td>- Final project report and presentation;</td>
</tr>
</tbody>
</table>
Bibliography


