A FAST ALGORITHM FOR LEAST SQUARES PROBLEMS
Inner-Outer Iteration Method for Least Squares Problems

Keishi Morikuni, The Graduate University for Advanced Studies
Ken Hayami, National Institute of Informatics, The Graduate University for Advanced Studies
Shin'ichi Oishi, The University of Tokyo

WHAT FOR?
- Survey
- Control
- Signal processing
- Statistical processing
- Image processing
- Optimization
- etc.

HOW?
- Development of solvers for solving least squares problems
- Theoretical analysis of the property of the solvers
- Verification through numerical experiments

WHAT DO WE UNDERSTAND?
Property of the solutions of least squares problems

\[ \min_{x \in \mathbb{R}^n} \| b - Ax \|_2 \]

where \( A \in \mathbb{R}^{m \times n} \) is "large", i.e., find \( x \) which minimizes \( \| b - Ax \|_2 \)

\[ \min \left\| \begin{bmatrix} b \ - A \end{bmatrix} \right\|_2 \quad \text{or} \quad \min \left\| \begin{bmatrix} b - A \end{bmatrix} \right\|_2 \quad \text{nonzero element} \]

HOW IS IT RELATED TO US?
For example...
- To determine the 3-D structure of protein molecules
- To determine the inner constitution of the earth from the analysis of seismic waves

WHY DO WE STUDY?
- To develop faster, more accurate, and more efficient methods for solving least squares problems
- To solve larger and larger, more and more ill-conditioned problems
- Theoretical understanding of the properties of the solvers

ITERATIVE METHOD
Iterative methods have advantages for large, sparse problems
The process is described as
1. Choose the initial solution \( x_0 \)
2. For \( j = 1, 2, \ldots, \) Do
3. Update \( x_j \) with an algorithm
4. EndDo

We should choose or have to develop better algorithms to achieve faster, more robust, and more accurate convergence
Krylov subspace iterative methods for linear systems use

\[ K_m(A, r_0) = \text{span} \{ r_0, Ar_0, \ldots, A^{m-1}r_0 \} \quad \text{for } A \in \mathbb{R}^{n \times n} \text{ and initial residual } r_0 = b - Ax_0 \]
- The generalized minimal residual (GMRES) method minimizes \( \| b - Ax \|_2 \) over the subspace
- \( \text{span} \{ v_1, v_2, \ldots, v_k \} = K_m(A, r_0) \), where \( (v_{j+1}, v_i) = 0 \) for \( i = 1, 2, \ldots, j \) and \( (v_{j+1}, v_{j+1}) = 1 \)
- The conjugate gradient method for least squares problems (CGLS) applies the CG method to \( A^T A x = A^T b \)

NUMERICAL EXPERIMENTS

Table 1: Information of the matrices

<table>
<thead>
<tr>
<th>Problem</th>
<th>m</th>
<th>n</th>
<th>nnz</th>
<th>dens [%]</th>
<th>rank</th>
<th>( \kappa(A) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maragal.6</td>
<td>21251</td>
<td>10144</td>
<td>537694</td>
<td>0.25</td>
<td>8331</td>
<td>2.91 \times 10^9</td>
</tr>
<tr>
<td>Maragal.8</td>
<td>60845</td>
<td>33693</td>
<td>1308415</td>
<td>0.06</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

Figure 1: Convergence history of the relative residuals for Maragal.6 and .8
Iterative methods need good preconditioners to achieve a better convergence and less storage requirement

- Robust Incomplete Factorization by Benzi and Tuma (2003)
- Preconditioners such as
  - Generalized Approximate Inverse by Cui and Hayami (2009)
  - Incomplete Givens Orthogonalization by Yin and Hayami (2009)

for least squares problems make improvements in convergence

**Algorithm 1** PCGMLS method using inner iterations

1. Choose \( x_0 \) and compute \( r_0 = b - A x_0 \)
2. Roughly solve \( A^T A z = A^T r_0 \) to obtain \( z \approx z_0 = B r_0 \)
3. By using an inner iteration
4. For \( j = 0, 1, \ldots \), Do
5. \( q_j = A r_j \)
6. \( \alpha_j = \gamma_j/q_j \)
7. \( s_j = r_j + \alpha_j q_j \)
8. If \( \|A^T(b-A x_{j+1})\|_2 / \|A^T r_0\|_2 < \epsilon \), then stop
9. \( r_{j+1} = r_j - \alpha_j q_j \)
10. \( s_{j+1} = A^T r_{j+1} \)
11. Roughly solve \( A^T A z = A^T r_{j+1} \) to obtain \( z \approx z_{j+1} = B r_{j+1} \)
12. By using an inner iteration
13. \( y_{j+1} = (s_{j+1} - z_{j+1}) \)
14. \( \beta_j = \gamma_j / \gamma_{j+1} \)
15. \( p_{j+1} = s_{j+1} + \beta_j y_{j+1} \)
16. EndDo

**Algorithm 2** RA-GMRES(k) method using inner iterations

1. Choose \( x_0 \) and compute \( r_0 = b - A x_0 \)
2. Roughly solve \( A^T A z = A^T r_0 \) to obtain \( z \approx z_0 = B r_0 \)
3. By using an inner iteration
4. For \( j = 1, 2, \ldots, k \), Do
5. Roughly solve \( A^T A z = A^T r_j \) to obtain \( z = w_j = B r_j \)
6. By using an inner iteration
7. \( y_j = (s_j - z_j) \)
8. \( v_j = w_j - z_j \)
9. EndDo
10. \( h_{j+1} = \|w_j\|_2 \)
11. \( u_j = w_j / h_{j+1} \)
12. Find \( y \in R^k \) that minimizes \( \min_{y \in R^k} \|\beta e_1 - R_j y\|_2 = \min_{y \in R^k} \|B r_j\|_2 \)
13. \( x_j = x_0 + [v_1, v_2, \ldots, v_k] y \)
14. If \( \|A^T(b-A x_j)\|_2 / \|A^T r_0\|_2 < \epsilon \), then stop
15. EndDo
16. \( x_0 = x_k \) and go to 2

**Algorithm 3** Cimmino-NR method

1. Choose \( x^{(0)} \in R^n \)
2. \( r^{(0)} = b - A x^{(0)} \)
3. For \( k = 0, 1, \ldots \), Do
4. For \( i = 1, 2, \ldots, n \), Do
5. \( d_{i}(k) = \chi(r_{i}(k), a_{i}) \)
6. EndDo
7. \( x^{(k+1)} = x^{(k)} + d^{(k)} \)
8. \( r^{(k+1)} = r^{(k)} - A d^{(k)} \)
9. EndDo

**Algorithm 4** NR-SOR method

1. Choose \( x^{(0)} \in R^n \)
2. \( r = b - A x^{(0)} \)
3. For \( k = 0, 1, 2, \ldots \), Do
4. For \( i = 1, 2, \ldots, n \), Do
5. \( d_{i}(k) = \chi(r_{i}(k), a_{i}) \)
6. \( x^{(k+1)} = x^{(k)} + d^{(k)} \)
7. \( r = r - d^{(k)} a_{i} \)
8. EndDo
9. EndDo

**Table 1: Rank deficient cases for \( \epsilon = 10^{-6} \)**

<table>
<thead>
<tr>
<th>Method</th>
<th>PCGMLS</th>
<th>RA-GMRES(k)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Freq.</td>
<td>diag.</td>
<td>RIF</td>
</tr>
<tr>
<td>Maragal.K</td>
<td>11,645</td>
<td>13,287</td>
</tr>
<tr>
<td>Maragal.L</td>
<td>62,44</td>
<td>58,03(10)</td>
</tr>
</tbody>
</table>

**Figure 2:** Number of iterations and computation time for Maragal.8 with various parameters

**WHAT WILL THIS RESEARCH ACCOMPLISH IN THE NEXT 10 YEARS?**

- The property of the solvers for least squares problems will be sufficiently revealed.
- They will be widely used with nothing to worry about breakdown and slow convergence.
- Massively parallel computers will employ them to solve dramatically large problems.
- With the growing need for solving difficult and complicated problems, our solvers will help science, engineering, and industry.