The Least Square Problem (LSQ) 
Methods for solving Linear LSQ 
Comments on the three methods 
Regularization techniques 
References

Methods for solving Linear Least Squares problems

Anibal Sosa

IPM for Linear Programming, 
September 2009
Outline

1. The Least Square Problem (LSQ)
   - Linear Least Square Problems

2. Methods for solving Linear LSQ
   - Normal Equations
   - QR Factorization
   - Singular Value Decomposition (SVD)

3. Comments on the three methods

4. Regularization techniques
   - Tikhonov regularization and Damped SVD
   - Tikhonov regularization order one and two
The Least Square Problem (LSQ)

- Linear Least Square Problems

Methods for solving Linear LSQ

- Normal Equations
- QR Factorization
- Singular Value Decomposition (SVD)

Comments on the three methods

Regularization techniques

- Tikhonov regularization and Damped SVD
- Tikhonov regularization order one and two
Outline

1. The Least Square Problem (LSQ)
   - Linear Least Square Problems

2. Methods for solving Linear LSQ
   - Normal Equations
   - QR Factorization
   - Singular Value Decomposition (SVD)

3. Comments on the three methods

4. Regularization techniques
   - Tikhonov regularization and Damped SVD
   - Tikhonov regularization order one and two
Outline

1. The Least Square Problem (LSQ)
   - Linear Least Square Problems

2. Methods for solving Linear LSQ
   - Normal Equations
   - QR Factorization
   - Singular Value Decomposition (SVD)

3. Comments on the three methods

4. Regularization techniques
   - Tikhonov regularization and Damped SVD
   - Tikhonov regularization order one and two
The Least Square Problem (LSQ)

- The objective function has the following special form
  \[ f(x) = \frac{1}{2} \sum_{j=1}^{m} r_j^2(x), \text{ where } r_j : \mathbb{R}^n \to \mathbb{R} \text{ are the residuals}, \text{ i.e.,} \]
  \[ \min_{x \in \mathbb{R}^n} f(x) = \min_{x \in \mathbb{R}^n} \frac{1}{2} r^T(x) r(x) = \min_{x \in \mathbb{R}^n} \frac{1}{2} ||r(x)||^2 \]

- \[ r : \mathbb{R}^n \to \mathbb{R}^m \text{ is called the residual vector, i.e., } r = \begin{bmatrix} r_1(x) \\ r_2(x) \\ \vdots \\ r_m(x) \end{bmatrix} \]

- Least square problems arise in many areas of applications
- Largest source of unconstrained optimization problems
The Least Square Problem (LSQ)

- The objective function has the following special form
  \[ f(x) = \frac{1}{2} \sum_{j=1}^{m} r_j^2(x) \], where \( r_j : \mathbb{R}^n \rightarrow \mathbb{R} \) are the residuals, i.e.,
  \[ \min_{x \in \mathbb{R}^n} f(x) = \min_{x \in \mathbb{R}^n} \frac{1}{2} r^T(x) r(x) = \min_{x \in \mathbb{R}^n} \frac{1}{2} ||r(x)||^2 \]

  \( r : \mathbb{R}^n \rightarrow \mathbb{R}^m \) is called the residual vector, i.e., \( r = [r_1(x), r_2(x), \ldots, r_m(x)] \)

- Least square problems arise in many areas of applications
- Largest source of unconstrained optimization problems
The Least Square Problem (LSQ)

- The objective function has the following special form
  \[ f(x) = \frac{1}{2} \sum_{j=1}^{m} r_j^2(x), \text{ where } r_j : \mathbb{R}^n \rightarrow \mathbb{R} \text{ are the residuals, i.e.,} \]
  \[ \min_{x \in \mathbb{R}^n} f(x) = \min_{x \in \mathbb{R}^n} \frac{1}{2} r^T(x) r(x) = \min_{x \in \mathbb{R}^n} \frac{1}{2} \| r(x) \|^2 \]

- \( r : \mathbb{R}^n \rightarrow \mathbb{R}^m \) is called the residual vector, i.e., \( r = \begin{bmatrix} r_1(x) \\ r_2(x) \\ \vdots \\ r_m(x) \end{bmatrix} \)

- Least square problems arise in many areas of applications

- Largest source of unconstrained optimization problems
Let \( \phi(x; \rho) \) be a model function that predict experimental values, for some fix parameters \( \rho \).

Usually we want to minimize the differences between the observed values \( y \in \mathbb{R}^m(\text{data}) \) and the predicted values \( \phi(x; \rho) \in \mathbb{R}^m \).

We can use LSQ setting \( r(x) = \phi(x; \rho) - y \)

\[
\min_{x \in \mathbb{R}^n} \frac{1}{2} \| \phi(x; \rho) - y \|_2^2
\]  

(1)

If \( \phi \) in (2) is nonlinear then we have a nonlinear LSQ problem

In our case \( \phi(x) = Ax \), thus we say this is a linear LSQ problem
Linear Least Square Problems

- Let $\phi(x; \rho)$ be a model function that predict experimental values, for some fix parameters $\rho$.
- Usually we want to minimize the differences between the observed values $y \in \mathbb{R}^m(\text{data})$ and the predicted values $\phi(x; \rho) \in \mathbb{R}^m$.
- We can use LSQ setting $r(x) = \phi(x; \rho) - y$

$$\min_{x \in \mathbb{R}^n} \frac{1}{2} \| \phi(x; \rho) - y \|_2^2 \quad (1)$$

- If $\phi$ in (2) is nonlinear then we have a nonlinear LSQ problem
- In our case $\phi(x) = Ax$, thus we say this is a linear LSQ problem
Let $\phi(x; \rho)$ be a model function that predict experimental values, for some fix parameters $\rho$.

Usually we want to minimize the differences between the observed values $y \in \mathbb{R}^m$(data) and the predicted values $\phi(x; \rho) \in \mathbb{R}^m$.

We can use LSQ setting $r(x) = \phi(x; \rho) - y$

$$\min_{x \in \mathbb{R}^n} \frac{1}{2} \| \phi(x; \rho) - y \|^2_2$$

(1)

If $\phi$ in (2) is nonlinear then we have a nonlinear LSQ problem

In our case $\phi(x) = Ax$, thus we say this is a linear LSQ problem
Linear Least Square Problems

Let \( \phi(x; \rho) \) be a model function that predict experimental values, for some fix parameters \( \rho \).

Usually we want to minimize the differences between the observed values \( y \in \mathbb{R}^m(\text{data}) \) and the predicted values \( \phi(x; \rho) \in \mathbb{R}^m \).

We can use LSQ setting \( r(x) = \phi(x; \rho) - y \)

\[
\min_{x \in \mathbb{R}^n} \frac{1}{2} \| \phi(x; \rho) - y \|^2_2 \tag{1}
\]

If \( \phi \) in (2) is nonlinear then we have a nonlinear LSQ problem

In our case \( \phi(x) = Ax \), thus we say this is a linear LSQ problem
Preliminaries for solving the LSQ problem

Observe that

\[ f(x) = \frac{1}{2} \|Ax - y\|^2 = \frac{1}{2} (Ax - y)^T (Ax - y) = \frac{1}{2} x^T A^T Ax - x^T A^T y + \frac{1}{2} y^T y \]

is easy to prove that

\[ \nabla f(x) = A^T (Ax - y) \quad \nabla^2 f(x) = A^T A \]

Since \( f \) is a convex function is well known that any \( x^* \) such that \( \nabla f(x^*) = 0 \) is a global minimizer of \( f \), therefore \( x^* \) satisfy the normal equations

\[ A^T Ax = A^T y \]

Next we discuss three major algorithms for solving Linear LSQ problems, assuming: i) \( m \geq n \) and ii) \( A \) is full rank
Observe that

\[ f(x) = \frac{1}{2} ||Ax - y||^2 = \frac{1}{2} (Ax - y)^T (Ax - y) = \frac{1}{2} x^T A^T Ax - x^T A^T y + \frac{1}{2} y^T y \]

is easy to prove that

\[ \nabla f(x) = A^T (Ax - y) \quad \nabla^2 f(x) = A^T A \]

Since \( f \) is a convex function is well known that any \( x^* \) such that \( \nabla f(x^*) = 0 \) is a global minimizer of \( f \), therefore \( x^* \) satisfy the normal equations

\[ A^T Ax = A^T y \]

Next we discuss three major algorithms for solving Linear LSQ problems, assuming: i) \( m \geq n \) and ii) \( A \) is full rank
Preliminaries for solving the LSQ problem

Observe that

\[ f(x) = \frac{1}{2} \|Ax-y\|^2_2 = \frac{1}{2} (Ax-y)^T (Ax-y) = \frac{1}{2} x^T A^T Ax - x^T A^T y + \frac{1}{2} y^T y \]

is easy to prove that

\[ \nabla f(x) = A^T (Ax - y) \quad \nabla^2 f(x) = A^T A \]

Since \( f \) is a convex function is well known that any \( x^* \) such that \( \nabla f(x^*) = 0 \) is a global minimizer of \( f \), therefore \( x^* \) satisfy the normal equations

\[ A^T Ax = A^T y \]

Next we discuss three major algorithms for solving Linear LSQ problems, assuming: i) \( m \geq n \) and ii) \( A \) is full rank
The Least Square Problem (LSQ)
Methods for solving Linear LSQ
Comments on the three methods
Regularization techniques
References

Normal Equations

**Step 1** Compute \( A^T A \) and \( A^T y \)

**Step 2** Compute Cholesky factorization of \( A^T A > 0 \)

\[ A^T A = R^T R, \quad R \text{ is an upper triangular matrix}(R_{ii} > 0) \]

**Step 3** Perform two triangular substitutions

\[ R^T z = R^T y \implies Rx^* = z \]

Disadvantages:

- Relative error of \( x^* \approx \kappa(A)^{21} \)
- Sensitive to ill-conditioned matrices

\[ 1\kappa(A) = ||A|| ||A^{-1}|| \approx \frac{\sigma_1}{\sigma_n} = \kappa_2(A) \]
Normal Equations

**Step 1** Compute $A^T A$ and $A^T y$

**Step 2** Compute Cholesky factorization of $A^T A > 0$

$$A^T A = R^T R, \quad R \text{ is an upper triangular matrix (} R_{ii} > 0)$$

**Step 3** Perform two triangular substitutions

$$R^T z = R^T y \implies Rx^* = z$$

Disadvantages:

- Relative error of $x^* \approx \kappa(A)^{21}$
- Sensitive to ill-conditioned matrices

$$\frac{1}{\kappa(A)} = \|A\| \|A^{-1}\| \approx \frac{\sigma_1}{\sigma_n} = \kappa_2(A)$$
Normal Equations

Step 1. Compute $A^TA$ and $A^Ty$

Step 2. Compute Cholesky factorization of $A^TA > 0$

$$A^TA = R^TR, \quad R \text{ is an upper triangular matrix} (R_{ii} > 0)$$

Step 3. Perform two triangular substitutions

$$R^Tz = R^Ty \implies Rx^* = z$$

Disadvantages:

- Relative error of $x^* \approx \kappa(A)^{21}$
- Sensitive to ill-conditioned matrices

$$\kappa_2(A) = \frac{\|A\| \|A^{-1}\|}{\sigma_1} = \kappa_2(A)$$
QR Factorization

Notice that \( \| \cdot \| \) is invariant under orthogonal transformations

\[ \| Ax - y \|_2^2 = \| Q^T (Ax - y) \|_2^2 \]

where \( Q_{m \times m} \) is orthogonal

- The QR factorization is done as follows

\[
A\Pi = Q \begin{bmatrix} R \\ 0 \end{bmatrix} = [Q_1 \quad Q_2] \begin{bmatrix} R \\ 0 \end{bmatrix} = Q_1 R
\] (2)

where \( \Pi_{n \times n} \) is a permutation matrix, \( Q_1 \) is the first \( n \) columns of \( Q \) and \( R_{n \times n} \) is upper triangular with \( R_{ii} > 0 \)

- Using 2 we have

\[
\| Ax - y \|_2^2 = \left\| \begin{bmatrix} Q_1^T \\ Q_2^T \end{bmatrix} (A\Pi\Pi^T x - y) \right\|_2^2
\]
QR Factorization

Notice that $\| \cdot \|$ is invariant under orthogonal transformations

$$\|Ax - y\|_2^2 = \|Q^T (Ax - y)\|_2^2$$

where $Q_{m \times m}$ is orthogonal

- The QR factorization is done as follows

$$A\Pi = Q \begin{bmatrix} R \\ 0 \end{bmatrix} = [Q_1 \quad Q_2] \begin{bmatrix} R \\ 0 \end{bmatrix} = Q_1 R$$

where $\Pi_{n \times n}$ is a permutation matrix, $Q_1$ is the first $n$ columns of $Q$ and $R_{n \times n}$ is upper triangular with $R_{ii} > 0$

- Using 2 we have

$$\|Ax - y\|_2^2 = \begin{bmatrix} Q_1^T \\ Q_2^T \end{bmatrix} (A\Pi \Pi^T x - y) \begin{bmatrix} Q_1^T \\ Q_2^T \end{bmatrix}^2$$
QR Factorization

Notice that \( || \cdot || \) is invariant under orthogonal transformations

\[
||Ax - y||_2^2 = ||Q^T (Ax - y)||_2^2
\]

where \( Q_{m \times m} \) is orthogonal

- The QR factorization is done as follows

\[
A\Pi = Q \begin{bmatrix} R \\ 0 \end{bmatrix} = [Q_1 \quad Q_2] \begin{bmatrix} R \\ 0 \end{bmatrix} = Q_1 R
\]

(2)

where \( \Pi_{n \times n} \) is a permutation matrix, \( Q_1 \) is the first \( n \) columns of \( Q \) and \( R_{n \times n} \) is upper triangular with \( R_{ii} > 0 \)

- Using 2 we have

\[
||Ax - y||_2^2 = \left| \begin{bmatrix} Q_1^T \\ Q_2^T \end{bmatrix} (A\Pi\Pi^T x - y) \right|_2^2
\]
QR Factorization (2)

\[
\begin{bmatrix}
Q_1^T \\
Q_2^T
\end{bmatrix}
\begin{pmatrix}
Q_1 & Q_2 \\
R & 0
\end{pmatrix}
\begin{pmatrix}
\Pi^T x - y
\end{pmatrix}
\]

\[
\left\| R \Pi^T x - Q_1^T y \right\|^2 + \left\| Q_2^T y \right\|^2
\]

Notice that from the last equation:

- The last term does not depend on \( x \)
- The minimum value is reached when \( R \Pi^T x - Q_1^T y = 0 \), therefore

\[
x^* = \Pi R^{-1} Q_1^T y
\]
QR Factorization (2)

\[
\begin{bmatrix}
Q_1^T \\
Q_2^T \\
\end{bmatrix}
\begin{bmatrix}
Q_1 & Q_2 \\
R & 0 \\
\end{bmatrix}
\Pi^T x - y
\]

\[
\begin{bmatrix}
R \\
0 \\
\end{bmatrix}
\Pi^T x - \begin{bmatrix}
Q_1^T \\
Q_2^T \\
\end{bmatrix} y
\]

\[
= \|R \Pi^T x - Q_1^T y\|_2^2 + \|Q_2^T y\|_2^2
\]

Notice that from the last equation:

- The last term does not depend on \( x \)
- The minimum value is reached when \( R \Pi^T x - Q_1^T y = 0 \), therefore

\[
x^* = \Pi R^{-1} Q_1^T y
\]
QR Factorization Algorithm

Step 1: Compute QR factorization of $A$
Step 2: Extract $Q_1$, identify $\Pi$ and $R$
Step 3: Perform one triangular substitution and one permutation

$$Rz = Q_1^T y \implies x^* = \Pi z$$

Advantage:
- Relative error of $x^* \approx \kappa(A)$

Disadvantage:
- Sometimes is necessary more information about data sensitivity
QR Factorization Algorithm

**Step 1** Compute QR factorization of $A$

**Step 2** Extract $Q_1$, identify $\Pi$ and $R$

**Step 3** Perform one triangular substitution and one permutation

$$Rz = Q_1^Ty \implies x^* = \Pi z$$

**Advantage:**
- Relative error of $x^* \approx \kappa(A)$

**Disadvantage:**
- Sometimes is necessary more information about data sensitivity
QR Factorization Algorithm

**Step 1** Compute QR factorization of $A$

**Step 2** Extract $Q_1$, identify $\Pi$ and $R$

**Step 3** Perform one triangular substitution and one permutation

$$Rz = Q_1^T y \implies x^* = \Pi z$$

**Advantage:**

- Relative error of $x^* \approx \kappa(A)$

**Disadvantage:**

- Sometimes is necessary more information about data sensitivity
Singular Value Decomposition (SVD)

**Theorem**

If $A_{m \times n}$ is real then there exist orthogonal matrices

$$U = [u_1 \ldots u_m] \in \mathbb{R}^{m \times m} \text{ and } V = [v_1 \ldots v_n] \in \mathbb{R}^{n \times n}$$

such that $A = U \Sigma V^T$, where

$\Sigma = \text{diag}(\sigma_1, \ldots, \sigma_p) \in \mathbb{R}^{m \times n}$, $p = \min\{m, n\}$ and $\sigma_1 \geq \sigma_2 \ldots \geq \sigma_p \geq 0$

In our case $\sigma_1 \geq \sigma_2 \ldots \geq \sigma_n > 0$ since $A$ is full rank and $m \gg n$ thus

$$A = U \begin{bmatrix} \Sigma_1 \\ 0 \end{bmatrix} V^T = [U_1 \quad U_2] \begin{bmatrix} \Sigma_1 \\ 0 \end{bmatrix} V^T = U_1 \Sigma_1 V^T \quad (3)$$

where $U_1$ has the first $n$ columns of $U$ and $\Sigma_1 = \text{diag}(\sigma_1, \ldots, \sigma_n)$. 
**Theorem**

*If* $A_{m \times n}$ *is real then there exist orthogonal matrices*

$$U = [u_1 \ldots u_m] \in \mathbb{R}^{m \times m} \text{ and } V = [v_1 \ldots v_n] \in \mathbb{R}^{n \times n}$$

*such that* $A = U \Sigma V^T$, *where*

$$\Sigma = \text{diag}(\sigma_1, \ldots, \sigma_p) \in \mathbb{R}^{m \times n}, \ p = \min\{m, n\} \text{ and } \sigma_1 \geq \sigma_2 \ldots \geq \sigma_p \geq 0$$

*In our case* $\sigma_1 \geq \sigma_2 \ldots \geq \sigma_n > 0$ *since* $A$ *is full rank and* $m \gg n$ *thus*

$$A = U \begin{bmatrix} \Sigma_1 \\ 0 \end{bmatrix} V^T = [U_1 \ U_2] \begin{bmatrix} \Sigma_1 \\ 0 \end{bmatrix} V^T = U_1 \Sigma_1 V^T \quad (3)$$

*where* $U_1$ *has the first* $n$ *columns of* $U$ *and* $\Sigma_1 = \text{diag}(\sigma_1, \ldots, \sigma_n)$.
The thin SVD

- Using (3) and similar ideas from QR

\[
\|Ax - y\|^2 = \left\| \begin{bmatrix} \Sigma_1 & 0 \\ 0 & 0 \end{bmatrix} \left( V^T x \right) - \begin{bmatrix} U_1^T \\ U_2^T \end{bmatrix} y \right\|^2
\]

\[
= \| \Sigma_1 \left( V^T x \right) - U_1^T y \|^2 + \| U_2^T y \|^2
\]

Again from the last equation:

- The last term does not depend on \( x \)
- The minimum value is reached when \( \Sigma_1 \left( V^T x \right) - U_1^T y = 0 \), therefore

\[
x^* = V \Sigma^{-1} U_1^T y
\]

or equivalently

\[
x^* = \sum_{i=1}^{n} \left( \frac{u_i^T y}{\sigma_i} \right) v_i \quad (4)
\]
The thin SVD

- Using (3) and similar ideas from QR

\[
\|Ax - y\|_2^2 = \left\| \begin{bmatrix} \Sigma_1 \\ 0 \end{bmatrix} (V^T x) - \begin{bmatrix} U_1^T \\ U_2^T \end{bmatrix} y \right\|_2^2 = \|\Sigma_1 (V^T x) - U_1^T y\|_2^2 + \|U_2^T y\|_2^2
\]

Again from the last equation:

- The last term does not depend on \( x \)
- The minimum value is reached when \( \Sigma (V^T x) - U_1^T y = 0 \), therefore

\[
x^* = V \Sigma^{-1} U_1^T y
\]

or equivalently

\[
x^* = \sum_{i=1}^{n} \left( \frac{u_i^T y}{\sigma_i} \right) v_i
\]
Equation (4) gives useful information about $x^*$ sensitivity

- Small changes in $A$ or $y$ can induce large changes in $x^*$ if $\sigma_i$ is small
- $A$ is rank deficient when $\frac{\sigma_n}{\sigma_1} \ll 1$. ($\sigma_n$ is the distance from $A$ to the set of singular matrices)

$x^*$ calculated as in (4) has the smallest 2-norm of all minimizers

Advantage:

- Most robust and reliable

Disadvantage:

- Most expensive
Equation (4) gives useful information about $x^*$ sensitivity
- Small changes in $A$ or $y$ can induce large changes in $x^*$ if $\sigma_i$ is small
- $A$ is rank deficient when $\frac{\sigma_n}{\sigma_1} \ll 1$. ($\sigma_n$ is the distance from $A$ to the set of singular matrices)
- $x^*$ calculated as in (4) has the smallest 2-norm of all minimizers

Advantage:
- Most robust and reliable

Disadvantage:
- Most expensive
The Normal Equation vs QR vs SVD

- The Cholesky-based algorithm is practical if $m \gg n$ (it is easier to store $A^T A$), even if $A$ is sparse.

- The QR algorithm avoids squaring $\kappa(A)$.

- When $A$ is rank-deficient, some $\sigma_i \approx 0$ thus any vector

  $$x^* = \sum_{\sigma_i \neq 0} \left( \frac{u_i^T y}{\sigma_i} \right) v_i + \sum_{\sigma_i = 0} \tau v_i$$

  is also a minimizer of $\|Ax - y\|$, for $\tau$ such that $\sigma_i \geq \tau_i$. Thus setting $\tau_i = 0$ we get the minimum norm solution.

**Remark:** For very large problems, it is recommended to use iterative methods such as Conjugate Gradient.

---

2 This is a type of filter by doing truncation.
Normal Eq. vs QR vs SVD

- The Cholesky-based algorithm is practical if $m \gg n$ (is easier store $A^T A$), even if $A$ is sparse

- The QR algorithm avoid squaring $\kappa(A)$

- When $A$ is rank-deficient, some $\sigma_i \approx 0$ thus any vector

\[
\chi^* = \sum_{\sigma_i \neq 0} \left( \frac{u_i^T y}{\sigma_i} \right) v_i + \sum_{\sigma_i = 0} \tau v_i
\]

is also a minimizer of $\|Ax - y\|$, for $\tau$ such that $\sigma_i \geq \tau_i$. Thus setting $\tau_i = 0$ we get the minimum norm solution\(^2\)

Remark: For very large problems is recommended to use iterative methods as *Conjugate Gradient*

\(^2\)This is a type of filter by doing truncation
The Cholesky-based algorithm is practical if $m \gg n$ (is easier store $A^T A$), even if $A$ is sparse.

The QR algorithm avoid squaring $\kappa(A)$.

When $A$ is rank-deficient, some $\sigma_i \approx 0$ thus any vector

$$x^* = \sum_{\sigma_i \neq 0} \left( \frac{u_i^T y}{\sigma_i} \right) v_i + \sum_{\sigma_i = 0} \tau v_i$$

is also a minimizer of $\|Ax - y\|$, for $\tau$ such that $\sigma_i \geq \tau_i$. Thus setting $\tau_i = 0$ we get the minimum norm solution.

Remark: For very large problems is recommended to use iterative methods as Conjugate Gradient.

\[\text{This is a type of filter by doing truncation}\]
Normal Eq. vs QR vs SVD

- The Cholesky-based algorithm is practical if \( m \gg n \) (is easier store \( A^T A \)), even if \( A \) is sparse.
- The QR algorithm avoids squaring \( \kappa(A) \).
- When \( A \) is rank-deficient, some \( \sigma_i \approx 0 \) thus any vector

\[
\chi^* = \sum_{\sigma_i \neq 0} \left( \frac{u_i^T y}{\sigma_i} \right) v_i + \sum_{\sigma_i = 0} \tau v_i
\]

is also a minimizer of \( \|Ax - y\| \), for \( \tau \) such that \( \sigma_i \geq \tau \). Thus setting \( \tau_i = 0 \) we get the minimum norm solution \(^2\)

**Remark:** For very large problems is recommended to use iterative methods as *Conjugate Gradient*.

\(^2\)This is a type of filter by doing truncation.
Tikhonov regularization\(^a\)

\(^a\)Ridge regression

- Most commonly used method for ill-posed problems
- The ill-conditioned problem 1 is posed as

\[
\min \frac{1}{2} \|Ax - y\|_2^2 + \frac{1}{2} \alpha^2 \|x\|_2^2 \tag{5}
\]

for some suitable regularization parameter \(\alpha > 0\)
- This improves the problem condition, even if \(A\) is rank-deficient, shifting the small singular values

\[
(A^T A + \alpha I_n) x = \underbrace{A^T Ax + \alpha x}_{\lambda x} = (\lambda + \alpha) x
\]

for any eigenvalue \(\lambda\) and eigenvector \(x\) of \(A^T A\)
The Least Square Problem (LSQ)
Methods for solving Linear LSQ
Comments on the three methods
Regularization techniques
References

Tikhonov regularization

Most commonly used method for ill-posed problems

The ill-conditioned problem 1 is posed as

$$\min \frac{1}{2} ||Ax - y||_2^2 + \frac{1}{2} \alpha^2 ||x||_2^2$$  \hspace{1cm} (5)

for some suitable regularization parameter $\alpha > 0$

This improves the problem condition, even if $A$ is rank-deficient, shifting the small singular values

$$(A^T A + \alpha I_n) x = A^T Ax + \alpha x = (\lambda + \alpha) x$$

for any eigenvalue $\lambda$ and eigenvector $x$ of $A^T A$
Tikhonov regularization

Most commonly used method for ill-posed problems

The ill-conditioned problem 1 is posed as

$$\min \frac{1}{2} ||Ax - y||_2^2 + \frac{1}{2} \alpha^2 ||x||_2^2$$

(5)

for some suitable regularization parameter $\alpha > 0$

This improves the problem condition, even if $A$ is rank-deficient, shifting the small singular values

$$(A^T A + \alpha I_n) x = A^T Ax + \alpha x = (\lambda + \alpha) x$$

for any eigenvalue $\lambda$ and eigenvector $x$ of $A^T A$
A little algebra shows that the minimum solution of (5) is given by the nonsingular system

\[(A^T A + \alpha^2 I_n) x = A^T y\]

and from (4) we can show that

\[x^* = \sum_{i=1}^{n} f_i \left( \frac{u_i^T y}{\sigma_i} \right) v_i\]

where \( f_i = \frac{\sigma_i^2}{\sigma_i^2 + \alpha^2} \) are known as filter factors\(^3\)

The impact of a small \( \alpha \) in the filter factors is:

- None for large \( \sigma_i (\alpha \ll \sigma_i) \), i.e. \( \frac{\sigma_i^2}{\sigma_i^2 + \alpha^2} \approx 1 \)
- Reduce the magnification of \( \frac{1}{\sigma_i} \) since \( \frac{\sigma_i^2}{\sigma_i^2 + \alpha^2} \approx \frac{\sigma_i^2}{\alpha^2} \ll 1 \)

A “good” choice of \( \alpha \) may provide enough numerical stability to expect a good approximate solution.

\(^3\)In signal processing are known as Wiener filters.
A little algebra shows that the minimum solution of (5) is given by the nonsingular system

\[(A^T A + \alpha^2 I_n) x = A^T y\]

and from (4) we can show that

\[x^* = \sum_{i=1}^{n} f_i \left( \frac{u_i^T y}{\sigma_i} \right) v_i\]

where \(f_i = \frac{\sigma_i^2}{\sigma_i^2 + \alpha^2}\) are known as filter factors\(^3\)

- The impact of a small \(\alpha\) in the filter factors is:
  - None for large \(\sigma_i(\alpha \ll \sigma_i)\), i.e. \(\frac{\sigma_i^2}{\sigma_i^2 + \alpha^2} \approx 1\)
  - Reduce the magnification of \(\frac{1}{\sigma_i}\) since \(\frac{\sigma_i^2}{\sigma_i^2 + \alpha^2} \approx \frac{\sigma_i^2}{\alpha^2} \ll 1\)
  - A “good” choice of \(\alpha\) may provide enough numerical stability to expect a good approximate solution

\(^3\)In signal processing are known as Wiener filters.
A little algebra shows that the minimum solution of (5) is given by the nonsingular system

\[(A^T A + \alpha^2 I_n) x = A^T y\]

and from (4) we can show that

\[x^* = \sum_{i=1}^{n} f_i \left( \frac{u_i^T y}{\sigma_i} \right) v_i\]

where \(f_i = \frac{\sigma_i^2}{\sigma_i^2 + \alpha^2}\) are known as filter factors

The impact of a small \(\alpha\) in the filter factors is:

- None for large \(\sigma_i(\alpha \ll \sigma_i)\), i.e. \(\frac{\sigma_i^2}{\sigma_i^2 + \alpha^2} \approx 1\)
- Reduce the magnification of \(\frac{1}{\sigma_i}\) since \(\frac{\sigma_i^2}{\sigma_i^2 + \alpha^2} \approx \frac{\sigma_i^2}{\alpha^2} \ll 1\)

A “good” choice of \(\alpha\) may provide enough numerical stability to expect a good approximate solution.
Damping the large components in magnitude may not inhibit undesirable behavior of the singular values.

Strong regularization is needed, penalizing rapid changes of \( x_i \) (4)

\[
\min \frac{1}{2} \| Ax - y \|_2^2 + \frac{1}{2} \alpha^2 \sum_{i=2}^{n-1} (x_i - x_{i-1})^2
\]

Again this expression is minimized by the solution of

\[
(A^T A + \alpha^2 B_1^T B_1) x = A^T y
\]

where

\[
B_1 = \begin{bmatrix}
1 & -1 & 0 & 0 & 0 \\
0 & 1 & -1 & 0 & 0 \\
\vdots & \vdots & \ddots & \ddots & \ddots \\
\vdots & \vdots & \ddots & \ddots & -1 \\
0 & 0 & 0 & 0 & 1 \\
\end{bmatrix}_{(n-1) \times n}
\]
Damping the large components in magnitude may not inhibit undesirable behavior of the singular values.

Strong regularization is needed, penalizing rapid changes of $x_i$ (4)

$$\min \frac{1}{2} ||Ax - y||^2_2 + \frac{1}{2} \alpha^2 \sum_{i=2}^{n-1} (x_i - x_{i-1})^2$$

Again this expression is minimized by the solution of

$$(A^T A + \alpha^2 B_1^T B_1) x = A^T y$$

where

$$B_1 = \begin{bmatrix} 1 & -1 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & \vdots & \ddots & -1 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \ (n-1) \times n$$
Damping the large components in magnitude may not inhibit undesirable behavior of the singular values.

Strong regularization is needed, penalizing rapid changes of $x_i$ (4)

$$\min \frac{1}{2} ||Ax - y||^2_2 + \frac{1}{2} \alpha^2 \sum_{i=2}^{n-1} (x_i - x_{i-1})^2$$

Again this expression is minimized by the solution of

$$(A^T A + \alpha^2 B_1^T B_1) x = A^T y$$

where

$$B_1 = \begin{bmatrix}
1 & -1 & 0 & 0 & 0 & 0 \\
0 & 1 & -1 & 0 & 0 & 0 \\
\vdots & \vdots & 1 & \vdots & \ddots & \vdots \\
\vdots & \vdots & \vdots & \vdots & \ddots & -1 \\
0 & 0 & 0 & 0 & 0 & 1
\end{bmatrix}_{(n-1) \times n}$$
Tikhonov regularization order two

An even stronger regularization is

$$\min \frac{1}{2} \|Ax - y\|^2 + \frac{1}{2} \alpha^2 \sum_{i=2}^{n-1} (x_{i+1} - 2x_i + x_{i-1})^2$$

Again this expression is minimized by the solution of

$$(A^T A + \alpha^2 B_2^T B_2) x = A^T y$$

where

$$B_2 = \begin{bmatrix}
-2 & 1 & 0 & 0 & \cdots \\
1 & -2 & 1 & 0 & \cdots \\
\vdots & 1 & -2 & 1 & \vdots \\
\vdots & \vdots & \ddots & \ddots & \vdots \\
0 & 0 & 0 & 1 & -2
\end{bmatrix}_{(n-2) \times n}$$
Tikhonov regularization order two

- An even stronger regularization is

\[
\min \frac{1}{2} ||Ax - y||^2 + \frac{1}{2} \alpha^2 \sum_{i=2}^{n-1} (x_{i+1} - 2x_i + x_{i-1})^2
\]

- Again this expression is minimized by the solution of

\[
(A^T A + \alpha^2 B_2^T B_2) x = A^T y
\]

where

\[
B_2 = \begin{bmatrix}
-2 & 1 & 0 & 0 & \cdots \\
1 & -2 & 1 & 0 & \cdots \\
\vdots & 1 & -2 & 1 & \vdots \\
\vdots & \vdots & \ddots & \ddots & \vdots \\
0 & 0 & 0 & 1 & -2
\end{bmatrix}_{(n-2) \times n}
\]
References