Numerical Optimization Methods

5. Least Squares Methods

Dr. Hongchuan Yu

NCCA

Bournemouth University

1. Preliminary Concepts

- Linear system vs. matrix form
- Backward substitution of solving $Ax = b$
  - Given a upper triangle $A = \begin{pmatrix} a_{11} & \cdots & a_{1n} \\ & \ddots & \vdots \\ & & a_{nn} \end{pmatrix}$

$$x_k = \frac{b_k - \sum_{j=k+1}^{n} a_{kj} x_j}{a_{kk}}, k = n, \dots, 1$$

  - For a lower triangle, forward substitution!

- Gaussian elimination
  - Augmented matrix, $A' = (A|b)$

$$A'(2,:) = A'(2,:) - \frac{a_{21}}{a_{11}} A'(1,:)$$

$$A'(k,:) = A'(k,:) - \frac{a_{k1}}{a_{11}} A'(1,:)$$

$$k = 2, \dots, n$$

$$A'(3,2{:}n) = A'(3,2{:}n) - \frac{a_{32}}{a_{22}} A'(3,2{:}n)$$

$$A'(k,2{:}n) = A'(k,2{:}n) - \frac{a_{k2}}{a_{22}} A'(k,2{:}n)$$

$$k = 3, \dots, n$$

Convert *A* into an Upper triangle, ……

  - Assumption: the diagonal element a_{ii} must always be away from Zero!

- LU decomposition (square matrix)
  - Recall Gaussian elimination procedure

Eliminating each column (i.e. zero out each column),

$$M^{(1)} = \begin{pmatrix} 1 & & & \\ -l_{21} & 1 & & \\ \vdots & & \ddots & \\ -l_{n1} & 0 & \dots & 1 \end{pmatrix},$$

$$l_{21} = \frac{a_{21}}{a_{11}}, \dots, l_{n1} = \frac{a_{n1}}{a_{11}}$$

……$M^{(i)}$……

$$U = M^{(n-1)} \dots M^{(1)} A$$

$$L = {M^{(1)}}^{-1} \dots {M^{(n-1)}}^{-1}$$

$$A = LU$$

Assumption: diagonal elements of *A* away from zero!

- Apply LU factorization to $Ax = b$

  $A = LU$,

  $Ly = b$, forward substitution

  $Ux = y$, backward substitution

- Compare with Gaussian Elimination,

  A decomposition is independent of *b*!

- Problem

$$A = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \qquad \det(A) = -1$$

  Or, a_{ii} is near zero!

- Pivoting for zero out diagonal elements
  - Partial pivoting,

    Choosing q, such that,

$$\left|a_{qk}^{(k-1)}\right| = \max_{k \le i \le n} \left|a_{ik}^{(k-1)}\right|$$

$$\begin{matrix} \\ k \rightarrow \\ q \rightarrow \\ \\ \end{matrix} \begin{pmatrix} a_{11} & & \cdots & & a_{1n} \\ 0 & \ddots & & & \\ \vdots & \ddots & a_{kk} & & \vdots \\ & 0 & \vdots & \ddots & \\ 0 & \cdots & a_{nk} & \cdots & a_{nn} \end{pmatrix}$$

    Interchanging rows *k* and *q* forms a permutation!

    Cannot guarantee stability of $Ax = b$!

– Complete pivoting,

$$\left|a_{qr}{}^{(k-1)}\right| = \max_{k \le i,j \le n} \left|a_{ij}{}^{(k-1)}\right|$$

$$\begin{array}{c} \\ k \rightarrow \\ q \rightarrow \\ \\ \end{array} \begin{pmatrix} a_{11} & & \cdots & & a_{1n} \\ 0 & \ddots & & & \\ \vdots & \ddots & a_{kk} & & \vdots \\ & 0 & \vdots & \ddots & \\ 0 & \cdots & a_{nk} & \cdots & a_{nn} \end{pmatrix}$$

$$\qquad\qquad\qquad\qquad\quad k \quad r$$

Interchanging rows *k* and *q* and then columns *k* and *r*!

– Very expensive! Have to pay attention to special matrices!

- Special matrices (no interchanging!)
  - Symmetric Positive Definite matrix, $x^T Ax > 0, x \neq 0$

$$A = LU = Ldiag(u_{ii}) \begin{pmatrix} 1 & \frac{u_{12}}{u_{11}} & \cdots & \frac{u_{1n}}{u_{11}} \\ & 1 & & \frac{u_{2n}}{u_{22}} \\ & & \ddots & \vdots \\ & & & 1 \end{pmatrix}$$

$$L = \begin{pmatrix} 1 & & \cdots & \\ \frac{u_{12}}{u_{11}} & 1 & & \\ \vdots & & \ddots & \vdots \\ \frac{u_{1n}}{u_{11}} & \frac{u_{2n}}{u_{22}} & \cdots & 1 \end{pmatrix}$$

$$A = Ldiag(u_{ii})L^T$$

– Cholesky factorization

$$G = L\sqrt{diag(a_{ii})}$$
$$A = GG^T$$

– SPD matrix's properties,

- Invertible;
- $a_{ii} > 0$;
- $\max\limits_{1 \le k,j \le n} |a_{kj}| \le \max\limits_{1 \le i \le n} |a_{ii}|$;
- $(a_{ij})^2 < a_{ii}a_{jj}, i \ne j$;

– Diagonally dominate matrix,

$$a_{ii} \ge \sum_{j=1, j \ne i}^{n} |a_{ij}|$$

– Banded matrix

Tridiagonal matrix,

$$A = \begin{pmatrix} a_{11} & a_{12} & 0 & \dots & 0 \\ a_{21} & a_{22} & a_{23} & \ddots & \vdots \\ 0 & a_{32} & a_{33} & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & a_{n-1,n} \\ 0 & \dots & 0 & a_{n,n-1} & a_{nn} \end{pmatrix}$$

If $|a_{11}| > |a_{12}|$, $|a_{ii}| \geq |a_{i,i-1}| + |a_{i,i+1}|$,... $|a_{nn}| > |a_{n,n-1}|$,

apply LU to *A* directly!

- QR Decomposition

  Let $A \in R^{m \times n}, m \geq n$, be full rank,

$$A = (Q_1 \quad Q_2)_{m \times m} \begin{pmatrix} T_{upper} \\ 0 \end{pmatrix},$$

$$Q_1 \in R^{m \times n}, Q_2 \in R^{m \times (m-n)}, T_{upper} \in R^{n \times n},$$

$$Q_1{}^T Q_1 = I, Q_2{}^T Q_2 = I, Q_1{}^T Q_2 = 0$$

  This is the column orthogonal!

  – How to compute QR?

    Givens rotation,

$$G(i, j, \theta) = \begin{bmatrix} 1 & \cdots & 0 & \cdots & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & & \vdots & & \vdots \\ 0 & \cdots & c & \cdots & -s & \cdots & 0 \\ \vdots & & \vdots & \ddots & \vdots & & \vdots \\ 0 & \cdots & s & \cdots & c & \cdots & 0 \\ \vdots & & \vdots & & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & \cdots & 0 & \cdots & 1 \end{bmatrix},$$

    Applying a series of Givens rotations to *A* leads to a upper matrix T_{upper}!

– Example:

For an underdetermined linear system, $Ax = b,\ A \in R^{m \times n}, m < n,$

$$A^T = (Q_1 \quad Q_2)\begin{pmatrix} T \\ 0 \end{pmatrix}$$

$$\left(T^T, 0\right)(Q_1 \quad Q_2)^T x = b$$

$$x' = (Q_1 \quad Q_2)\begin{pmatrix} T^{-T} \\ 0 \end{pmatrix} b = Q_1 T^{-T} b$$

$$Ax' = b$$

For nullsapce, $Ax = 0$, Q_2 forms a basis, the general solution is,

$$x = x' + Q_2 y, \qquad \forall y \in R^{n-m}$$

- SVD and Generalized SVD
  - SVD

    For $A \in R^{n\times m}$, there exist orthogonal matrices,

    $U \in R^{n\times n}$ and $V \in R^{m\times m}$,

    $U^T AV = \Sigma = diag(\sigma_1, \dots, \sigma_p)$,
  - Properties,

    (1) $Av_i = \sigma_i u_i, \quad i = 1 \dots p,$

    (2) $Rank(A) = p, p = \min(n, m)$

    $$Null(A) = span\{v_{p+1}, \dots, v_m\}$$

    $$Range(A) = span\{u_1, \dots, u_p\}$$

$$\mathbf{A} = \mathbf{U}\,\Sigma\,\mathbf{V}^*$$

$$= \begin{bmatrix} \mathbf{U}_{\mathcal{R}} & \mathbf{U}_{\mathcal{N}} \end{bmatrix} \left[\begin{array}{cccc|cc} \sigma_1 & 0 & \dots & & \dots & 0 \\ 0 & \sigma_2 & & & & \\ \vdots & & \ddots & & & \\ & & & \sigma_\rho & & \\ \hline & & & & 0 & \\ \vdots & & & & & \ddots \\ 0 & & & & & 0 \end{array}\right] \begin{bmatrix} \mathbf{V}_{\mathcal{R}}{}^* \\ \mathbf{V}_{\mathcal{N}}{}^* \end{bmatrix} = \begin{bmatrix} u_1 & \dots & u_\rho & u_{\rho+1} & \dots & u_m \end{bmatrix} \begin{bmatrix} \mathbf{S}_{\rho\times\rho} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} v_1^* \\ \vdots \\ v_\rho^* \\ v_{\rho+1}^* \\ \vdots \\ v_n^* \end{bmatrix}$$

(3) For two $A, B \in R^{n\times m}$,

$$\sigma_{max}(A+B) \le \sigma_{max}(A) + \|B\|_2$$
$$\sigma_{min}(A+B) \ge \sigma_{min}(A) - \|B\|_2$$

- Generalized SVD
  - For $A \in R^{n_1\times m}, B \in R^{n_2\times m}, n_1 > m$, assume that

$$rank\begin{pmatrix} A \\ B \end{pmatrix} = r$$

There exist orthogonal, $U_1 \in R^{n_1\times n_1}, U_2 \in R^{n_2\times n_2}$, and invertible $X \in R^{m\times m}$, such that,

$$U_1^T AX = D_A = \begin{pmatrix} I & & \\ & diag(\alpha_{p+1}, \dots, \alpha_r) & \\ & & 0 \end{pmatrix}$$

$$U_2^T BX = D_B = \begin{pmatrix} 0 & & \\ & diag(\beta_{p+1}, \dots, \beta_r) & \\ & & 0 \end{pmatrix}$$

where $p = max\{r - n_2, 0\}$.

- Numerical algebra
  - Orthogonal matrices for $Ax = b$

    $x = A^{-1}b = A^T b,$ $O(2n^2)$

    If $A = I - 2uu^T, u \in R^n,$

    $x = A^{-1}b = b - 2(u^T b)u,$ $O(4n)$

  - Permutation matrices $Ax = b$

$$A_{ij} = \begin{cases} 1, & j = \pi_i \\ 0, otherwise \end{cases}$$

$\pi = (\pi_1, \dots, \pi_n)$ is a permutation of $(1,2, \dots, n)$.

- Example,

  There are six permutations of the set $\{1,2,3\}$, namely $(1,2,3),(1,3,2),(2,1,3),(2,3,1),(3,1,2)$, and $(3,2,1)$!

  Each π_i contains only one entry with value 1 while others with 0;

$$Ax = \left(x_{\pi_1}, \dots, x_{\pi_n}\right)$$

  A is orthogonal!

$$x = A^T b$$

  Permuting the entries of *b*!

– Factor solver

Let $A = A_1 \dots A_k$, for $Ax = b$,

$$y_1 \coloneqq A_1^{-1} b$$

$$y_2 \coloneqq A_2^{-1} y_1 \dots \dots$$

$$x = A_k^{-1} y_{k-1}$$

– Block LDL^T factorization

$$A = PLDL^T P^T, D\ is\ Block\ diagonal$$

○ Solve $Ax = b$ by Gaussian elimination,

solve $Py_1 = b$;

solve $Ly_2 = y_1$, forward substitution;

solve $Dy_3 = y_2$;

solve $L^T y_4 = y_3$, backward substitution;

solve $P^T x = y_4$;

– Schur complement

Let $X = \begin{bmatrix} A & B \\ B^T & C \end{bmatrix}_{n \times n}$, $A \in R^{k \times k}$ symmetric and nonsingular, Schur complement of *A* in *X* is,

$$S = C - B^T A^{-1} B$$

If C is invertible, Schur complement of *C* in *X* is,

$$S = A - B^T C^{-1} B$$

- Inverse of *X*,

$$\begin{bmatrix} A & B \\ B^T & C \end{bmatrix}^{-1} = \begin{bmatrix} A^{-1} + A^{-1} B S^{-1} B^T A^{-1} & -A^{-1} B S^{-1} \\ -S^{-1} B^T A^{-1} & S^{-1} \end{bmatrix}$$

Positive definite $X \leftrightarrow A$ and *S* are positive definite;

If *A* is positive definite, then *X* is positive semi-definite $\leftrightarrow$ *S* is positive semi-definite!

– Block elimination

To solve $Ax = b$, let $A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}, x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, b = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$, then

$$\begin{cases} x_1 = A_{11}^{-1} b_1 - A_{11}^{-1} A_{12} x_2 \\ \left(A_{22} - A_{21} A_{11}^{-1} A_{12}\right) x_2 = b_2 - A_{21} A_{11}^{-1} b_1 \end{cases}$$

- Schur complement of 1st block A_{11},

$$S = \left(A_{22} - A_{21} A_{11}^{-1} A_{12}\right)$$

Construct sub terms, $A_{11}^{-1} A_{12}$ and $A_{11}^{-1} b_1$;

Form S and $b' = b_2 - A_{21} A_{11}^{-1} b_1$;

Solve $S x_2 = b'$;

Solve $A_{11} x_1 = b_1 - A_{12} x_2$;

- Solving underdetermined linear system
  - Given, $Ax = b, A \in R^{m \times n}, m < n, rank(A) = m$
  - Solution form,

$$\{Fz + \hat{x} | z \in R^{n-m}\}$$

where *F* is a matrix whose columns form a basis for nullspace of *A*, and $\hat{x}$ is a particular solution.

  - How to construct *F* and $\hat{x}$? (splitting method)
  - Partition *A*,

$$Ax = (A_1, A_2)\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = b$$

where $A_1 \in R^{m \times m}$ is nonsingular.

$$x_1 = {A_1}^{-1} b - {A_1}^{-1} A_2 x_2$$

– General solution,

$$x = \begin{pmatrix} -A_1^{-1}A_2 \\ I \end{pmatrix} x_2 + \begin{pmatrix} A_1^{-1}b \\ 0 \end{pmatrix}$$

then $F = \begin{pmatrix} -A_1^{-1}A_2 \\ I \end{pmatrix}$, $\hat{x} = \begin{pmatrix} A_1^{-1}b \\ 0 \end{pmatrix}$!

Challenge is HOW to partition *A*! (partition examples)

○ Apply QR factorization, $A^T = (Q_1, Q_2)\begin{pmatrix} B \\ 0 \end{pmatrix}$

Substituting factors,

$$\hat{x} = Q_1 B^{-T} b$$

$$A\hat{x} = b = B^T {Q_1}^T Q_1 B^{-T} b$$

Let $F = Q_2$, spanning nullspace,

for $x = Q_2 z$, $Ax = 0$!

General form, $x = \hat{x} + Q_2 z$!

- Apply LU factorization,

$$A^T = LU$$

where, $L \in R^{n \times m}$ a unit lower triangle (applying row pivoting Alg.), $U \in R^{m \times m}$ upper triangle.

Substituting factors,

Let $L = \begin{pmatrix} L_1 \\ L_2 \end{pmatrix}$,

$$F = \begin{pmatrix} -{L_1}^{-T} {L_2}^T \\ I \end{pmatrix}, \qquad \hat{x} = \begin{pmatrix} {L_1}^{-T} Ub \\ 0 \end{pmatrix}$$

where $L_1 \in R^{m \times m}$ unit lower triangle, $L_2 \in R^{(n-m) \times m}$

- Solving ill-conditioned problem
  - Some singular values of *A* tend to zero!

    Let $A = U\Sigma V^T$, Tolerable condition number,

    $$k(A) = \frac{\lambda_1}{\lambda_r}, r \leq \min(m, n)$$

    Let the singular values below λ_r be 0!

    $$A_r = U \begin{pmatrix} \Sigma_r & 0 \\ 0 & 0 \end{pmatrix} V^T$$

  - Solution *x* of the smallest norm,

    $$x = V \begin{pmatrix} \Sigma_r^{-1} & 0 \\ 0 & 0 \end{pmatrix} U^T b$$

    The Particular solution is some pseudoinverse solution!

- Iterative methods for $Ax = b$
  - Select a nonsingular matrix, *M*, and partition *A* as, $A = M - N$, and $M^{-1}A = I - M^{-1}N$
  - Define iterative scheme, initial guess, $x^{(0)}$,

$$Mx^{(k+1)} = b + Nx^{(k)}, k = 0,1, \ldots$$
$$x^{(k+1)} = M^{-1}b + M^{-1}Nx^{(k)}$$

  - Iteration matrix,

    $M^{-1}N$ and vector $M^{-1}b$ are precomputed!

  - Problem,

    how to select *M*?/or how to partition *A*?

    (partition methods)

- Jacobian Iteration

Let $M = diag(A) = D$ and $N = -(L + U)$,

$$x^{(k+1)} = D^{-1}\left(b - (L + U)x^{(k)}\right)$$

$$x_i{}^{(k+1)} = \frac{1}{a_{ii}}\left(b_i - \sum_{j=1}^{i-1} a_{ij}x_j{}^{(k)} - \sum_{j=i+1}^{n} a_{ij}x_j{}^{(k)}\right)$$

Note that, computing variables, $x_i{}^{(k+1)}$, in parallel

If *A* is diagonal dominant, for any $x^{(0)}$, it can converges!

Convergence is very slow!

- Gauss-Seidel iteration,

  To speed up convergence,

  Let $M = D + L$, and $N = -U$,

  Iteration scheme,

$$(D + L)x^{(k+1)} = b - Ux^{(k)}$$

$$x^{(k+1)} = D^{-1}(b - Lx^{(k+1)} - Ux^{(k)})$$

$$x_i{}^{(k+1)} = \frac{1}{a_{ii}}\left(b_i - \sum_{j=1}^{i-1} a_{ij}x_j{}^{(k+1)} - \sum_{j=i+1}^{n} a_{ij}x_j{}^{(k)} \right)$$

  If *A* is SPD or diagonal dominant,

  It can converge for any $x^{(0)}$.

- Successive over-relaxation (SOR)

  To further accelerate the convergence,

  Let $M = \frac{1}{w}D + L$ and $N = \left(\frac{1}{w} - 1\right)D - U$

  Iteration scheme,

  $$\left(\frac{1}{w}D + L\right)x^{(k+1)} = \left(\left(\frac{1}{w} - 1\right)D - U\right)x^{(k)} + b$$

  $$x^{(k+1)} = (1 - w)x^{(k)} + w\left(D^{-1}\left(b - Lx^{(k+1)} - Ux^{(k)}\right)\right)$$

  The 2nd term is the Gauss-Seidel iteration!

$$x_i^{(k+1)} = (1-w)x_i^{(k)} + \frac{w}{a_{ii}}\left(b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij}x_j^{(k)}\right)$$

$$0 < w < 2!$$

Appropriate *w* will effectively accelerate the convergence!

- Summary
  - Gaussian Elimination
  - LU factorization (pivoting, Cholesky factorization)
  - Iterative methods (Jacobi's method, Gaussian-Seidel, SOR)

2. Linear Least Squares Methods

- Least squares problem
  - Standard approach to the approximate solution of overdetermined systems;
  - Application: Data fitting!
  - Linear least squares problems,

$$\sum_{j=1}^{n} x_{ij}\beta_j = y_i, i = 1, \dots, m, m > n$$

Or, matrix form, $X\beta = Y, X \in R^{m\times n}, Y \in R^n,$

Quadratic optimization form,

$$\beta^* = arg\min_{\beta} F(\beta) = \|Y - X\beta\|^2$$

– Normal equation,

$$\frac{\partial F}{\partial \beta} = 0$$

$$(X^T X)\beta = X^T Y$$

where, *X* is full rank!

There exists Closed-Form Solution!

– Another method for full rank *X*,

$$\min_{\beta} \|Y - X\beta\|^2$$

$$= \left\|UDV^T\beta - Y\right\|^2 = \left\|DV^T\beta - U^T Y\right\|^2$$

Let $\beta' = V^T\beta$ and $Y' = U^T Y$,

$$\beta' = argmin\|D\beta' - Y'\|$$

$$\beta = V\beta'$$

- When *X* is not full rank,

$$X = (U_1, U_2)\begin{pmatrix} D \\ 0 \end{pmatrix} V^T$$

$$\min_{\beta} \|Y - X\beta\|^2$$

$$= \left\|U^T(Y - X\beta)\right\|^2$$

$$= \left\|DV^T\beta - {U_1}^T Y\right\|^2 + \left\|{U_2}^T Y\right\|^2$$

  - Recall the solution form of full rank *X*,

$$\begin{cases} \beta^* = VD^{-1}{U_1}^T Y \\ \beta^* = \sum_{i=1}^{n} \dfrac{{u_i}^T Y v_i}{\sigma_i} \end{cases}$$

– When $\sigma_{i,\ldots,n} = 0$, general solution,

$$\beta = \sum_{\sigma_i \neq 0} \frac{{u_i}^T Y v_i}{\sigma_i} + \sum_{\sigma_i = 0} \alpha_i u_i$$

where, arbitrary factors α_i.

- The particular solution with the smallest norm,

$$\beta^* = \sum_{\sigma_i \neq 0} \frac{{u_i}^T Y v_i}{\sigma_i}$$

- Another method for rank-deficient system (Pseudo-inverse)

  For a deficient rank system $Ax = b$, let $A = UDV^T$. Define pseudo-inverse of *D* as,

$$d_{ii}^{+} = \begin{cases} 0, & if\ d_{ii} = 0 \\ d_{ii}^{-1}, & otherwise \end{cases}$$

  - The pseudo-inverse of *A*, $A^{+} = VD^{+}U^{T}$, i.e.

$$x = A^{+}b$$

- Pseudoinverse of a full rank system $Ax = b$
  - Case 1, $A \in R^{m \times n}, m > n, Rank(A) = n$

$$A^{+} = \left(A^{T}A\right)^{-1}A^{T}$$

  - Case 2, $n > m, Rank(A) = m$

$$A^{+} = A^{T}\left(AA^{T}\right)^{-1}$$

$$A\left(A^{T}\left(AA^{T}\right)^{-1}b\right) = b \rightarrow x = A^{T}\left(AA^{T}\right)^{-1}b$$

- Weighted Least Squares
  - Add a diagonal matrix *W*,

$$\beta^* = arg\min_{\beta} S(\beta) = (Y - X\beta)^T W (Y - X\beta)$$

$$= \sum_{i=1}^{n} w_i \left(y_i - X_i^T \beta\right)^2$$

  - Normal equation,

$$\left(X^T W X\right)\beta = X^T W Y$$

  - Solution,

$$\beta^* = \left(X^T W X\right)^{-1} X^T W Y$$

– Define weight *W*,

○ Example of fitting scattered points,

$$y_i = \beta_0 + \beta_1 x_i + \cdots + \beta_m x_i^m + \varepsilon_i$$

Define *X* as a polynomial basis vector,

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^m \\ 1 & x_2 & x_2^2 & \cdots & x_2^m \\ 1 & x_3 & x_3^2 & \cdots & x_3^m \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & x_n^2 & \cdots & x_n^m \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \vdots \\ \beta_m \end{bmatrix} + \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \varepsilon_3 \\ \vdots \\ \varepsilon_n \end{bmatrix},$$

Define *W* by a Gaussian, $W = diag\left(e^{-\frac{\|x_i - \bar{x}\|^2}{h^2}}\right)$

(polynomial regression!)

- Moving Least Squares
  - Basic idea, each point p is fitted by weighted Least squares!
  - Notations,

    X is defined by Polynomial base $P(p) = (1, x, y, \dots)^T$,

    e.g. $P^*(p_i) = \left(1, x_i, x_i^2, \dots\right)^T, i = 1, \dots, n, p_i = (x_i)$ (1D),

    $P^*(p_i) = \left(1, x_i, y_i, x_i^2, x_i y_i, y_i^2, \dots\right)^T, p_i = (x_i, y_i)$ (2D),

    $P^*(p_i) = \left(1, x_i, y_i, z_i, x_i^2, x_i y_i, x_i z_i, y_i^2, y_i z_i, z_i^2, \dots\right)^T, p_i = (x_i, y_i, z_i)$ (3D),

    Let weight at point p be $W(p - p_i) = diag\left(w_i(p - p_i)\right)$, and $w_i(.)$ may be a gaussian!

    Let p_i be particular points (/control points).

    - Step 1: Polynomial approximation if β is given,

      $$f(p) = P(p)\beta$$

    - Step 2: Determine Surface if β is unknown,

      $$f(p_i) = P^*(p_i)\beta, i = 1, \dots, n$$

      e.g. implicit function $f(p)$ and vector β

– Moving least squares problem,

Define the functional F on a point *p*,

$$\min_{\beta} F = \frac{1}{2}\sum_{i=1}^{n} w_i (p - p_i)(P^*(p_i)\beta - f_i)^2$$

Matrix form,

$$F = \frac{1}{2}(P^*\beta - f^*)^T_{1\times n} W(p - p_i)(P^*\beta - f^*)_{n\times 1}, i = 1, \dots n$$

n control points!

- 1st order derivative,

$$P^{*T} W P^* \beta = P^{*T} W f^*$$

- The local solution on *p*,

$$\beta^* = \left(P^{*T} W P^*\right)^{-1} P^{*T} W f^*$$

– Implementation

- Step 1: Approximation solution on some point *p*,

$$f(p) = P(p)\beta^*$$
$$= P(p)\left(P^{*T}W(p-p_i)P^*\right)^{-1}P^{*T}W(p-p_i)f^*$$
$$i = 1 \ldots n$$

Note that $P(p)$ is different from P^{*T}, since P^{*T} only contains the control points p_i. Similarly $f(p)$ is different from f^*!

- Step 2: MLS Shape function on any point *p*,

$$\varphi(p) = P(p)\left(P^{*T}W(p-p_i)P^*\right)^{-1}P^{*T}W(p-p_i)f^*$$
$$i = 1 \ldots n$$

Note that, (1) $\varphi(p)$ are unknown in closed form! (2) *W* of $\varphi(p)$ depends on $(p-p_i), i = 1 \ldots n$!

Therefore, for a new point *p*, we have to re-compute $\varphi(p)$ accordingly, i.e. updating β^* for every new point *p*!

3. Constrained Least Squares

- Least squares solution of homogeneous Eqs.
  - Find *x* minimizing $\|Ax\|$ subject to $\|x\| = 1$
    - Take SVD, $\left\|UDV^T x\right\| = \left\|DV^T x\right\|$

    and

    $$\|x\| = \left\|V^T x\right\|$$

    Thus, $\min_x \|Ax\|$ is equivalent to,

    $$\min_y \|Dy\|, \qquad s.t. \|y\| = 1, \qquad y = V^T x$$

    There exists a solution $y = (0, \dots, 0, 1)^T$, which implies that $x = Vy$ and the last column of *V* is the least squares solution of $Ax = 0$. (it can be proven by the eigenvalue decomposition of $A^T A$)

- Solving $\min_x \|Ax\|^2$
  - Recall Rayleigh quotient, $\min_x \frac{x^T A^T A x}{x^T x}$, if $\|x\| \neq 1$,
  - SVD of A, $A = U\Sigma V^T$, $\min_x x^T V \Sigma^2 V^T x$
  - When the smallest eigenvalue is zero, it reaches the minimal at x as the eigenvector corresponding to the smallest eigenvalue.

- Remark
  - This is a standard form, which most constrained least square problems can be converted into!
  - The basic idea is to solve a few of typical least square problems as standard forms and then convert many least square problems into some appropriate standard forms through matrix techniques!

- Inequality constraints,
  - Solve $x = argmin\|Ax - b\|^2,\ s.t.\|x\| \leq a$
    - Take SVD to *A*, $A = U\Sigma V^T$,

      $$\left\|U\Sigma V^T x - b\right\| = \left\|\Sigma\hat{x} - \hat{b}\right\|$$

      where, $\hat{x} = V^T x$ and $\hat{b} = U^T b, r = rank(A)$
    - If $\sum_{i=1}^{r}\left(\frac{\hat{b}_i}{\sigma_i}\right)^2 > a^2$,

      find ρ^* such that $\sum_{i=1}^{r}\left(\frac{\sigma_i\hat{b}_i}{{\sigma_i}^2+\rho^*}\right)^2 = a^2$,

      update, $x = (v_i)_{i=1:r}\left(\frac{\sigma_i\hat{b}_i}{{\sigma_i}^2+\rho^*}\right)_{i=1:r}^T$
    - If $\sum_{i=1}^{r}\left(\frac{\hat{b}_i}{\sigma_i}\right)^2 \leq a^2$,

      update $x = (v_i)_{i=1:r}\left(\frac{\hat{b}_i}{\sigma_i}\right)_{i=1:r}^T$

– Find x, $\min_x \|Ax - b\|$ subject to $\|Bx - d\| \leq a$

- Let $A \in R^{n_1 \times m}, B \in R^{n_2 \times m}$, take Generalized SVD to $\begin{pmatrix} A \\ B \end{pmatrix}$

  Convert to,

  $$\min_y \left\| D_A y - \hat{b} \right\|, s.t. \left\| D_B y - \hat{d} \right\| \leq a$$

  where $y = X^{-1}x, \hat{b} = U_1^T b, \hat{d} = U_2^T d$

- Apply Lagrange multipliers,

  $$\min_{\lambda, y} \left\| D_A y - \hat{b} \right\|^2 + \lambda \left(\left\| D_B y - \hat{d} \right\|^2 - a^2 \right)$$

- Take gradient on it w.r.t. y,

  $$\left({D_A}^T D_A + \lambda {D_B}^T D_B \right) y = {D_A}^T \hat{b} + \lambda {D_B}^T \hat{d}$$

- Solution,

$$y_i(\lambda) = \begin{cases} \dfrac{\alpha_i \hat{b}_i + \lambda \beta_i \hat{d}_i}{{\alpha_i}^2 + \lambda {\beta_i}^2}, i = 1 \dots p \\ \dfrac{\hat{b}_i}{\alpha_i}, \qquad i = p+1, \dots m \end{cases}$$

Lagrange parameter λ,

$$\lambda = argmin \left\| D_B y(\lambda) - \hat{d} \right\|^2 - a^2$$

Note that it is 1-dimensioanl search!

- Equality Constraints
  - Find *x* minimizing $\|Ax\|$ subject to $\|x\| = 1$ and $Cx = 0$.
    - Assume $rank(C) = r$, $C = UDV^T$
    - Note that the row space of *C* is generated by the first *r* rows of V^T.
    - Orthogonal complement of row space, $C^{\perp}$, is the null space of C, i.e. the last columns of V corresponding to 0-eigenvalue

      $$CC^{\perp} = 0$$

      For *x* of constraint $Cx = 0$, (involving constraint into cost Fun!)

      $$x = C^{\perp}y \text{ and } \|x\| = \left\|C^{\perp}y\right\| = \|y\|$$

    - Convert to, $\min_y \left\|AC^{\perp}y\right\|$ subject to $\|y\| = 1$

      (comparing it with a standard form!)

- Find *x* minimizing $\|Ax\|$ subject to $\|x\| = 1$ and $x = G\hat{x}$ (*x* and $\hat{x}$ are unknown)
  - Assume $rank(G) = r$, $G = UDV^T$
  - Note that the row space of G^T is generated by the first *r* rows of U^T.
  - Orthogonal complement of row space, $G^{\perp}$, is *the null space of* G^T, i.e.

$$G^T G^{\perp} = 0$$

  - For *x* of ${G^{\perp}}^T x = 0$, *x* is the solution of $x = G\hat{x}$, i.e. nullspace! Only consider nullspace here!
  - Thus convert constraint $x = G\hat{x}$ to ${G^{\perp}}^T x = 0$, i.e.

$$\min_x \|Ax\|, \qquad s.t. {G^{\perp}}^T x = 0 \; and \; \|x\| = 1$$

(this is the previous problem!)

  - If $\hat{x}$ is required too, $\hat{x} = G^{+}x$.

– Find x, $\min_x \|Ax - b\|, s.t. Bx = d$

where $B \in R^{p \times n}, A \in R^{m \times n}, d \in R^p, rank(B) = p$, A and B full rank;

- Apply QR factorization to B, $Q^T B^T = \begin{pmatrix} T \\ 0 \end{pmatrix}, Q \in R^{n \times n}, T \in R^{p \times p}$

  Let $AQ = (A_1, A_2), A_1 \in R^{m \times p}, A_2 \in R^{m \times (n-p)}, Q^T x = \begin{pmatrix} y \\ z \end{pmatrix}, y \in R^p, z \in R^{(n-p)}$;

- Then,

  $Ax = A_1 y + A_2 z$ and $T^T y = d$

- Solution,

  To solve y by backward substitution.

$$z = arg \min_z \|A_2 z - (b - A_1 y)\|^2$$

$$x = Q \begin{pmatrix} y \\ z \end{pmatrix}$$

– Find *x*, minimizing $\|Ax\|$ subject to $\|Cx\| = 1$

- Let $C = UDV^T$, $\|Cx\| = 1$ is equivalent to,

$$\|DV^T x\| = 1$$

Convert to, $\min_y \|AVy\| \; s.t. \; \|Dy\| = 1, y = V^T x$

- Let $rank(D) = r$, and $A' = A\left(V_{:,1:r}, V_{:,r+1:n}\right) = (A'_1, A'_2), D = \begin{pmatrix} D_{1:r,:} \\ 0 \end{pmatrix}, y = \left(V_{:,1:r}, V_{:,r+1:n}\right)^T x = \begin{pmatrix} y_1 \\ y_2 \end{pmatrix}$

$\|A'y\|$ is equivalent to $\|A'_1 y_1 + A'_2 y_2\|$ partitioning A' based on rank, i.e. $A'y = (A'_1, A'_2)\begin{pmatrix} y_1 \\ y_2 \end{pmatrix}$

The constraint is changed as, $\|D_{1:r} y_1\| = 1$

- Substituting $y_2 = -{A'_2}^+ A'_1 y_1$ to $\|A'_1 y_1 + A'_2 y_2\|$,

$$\min_{x'} \left\| (I - A'_2 {A'_2}^+) A'_1 D_1^{-1} x' \right\|, s.t. \|x'\| = 1$$

where, $x' = D_1 y_1$, (the previous problem!)

4. Nonlinear Least Squares Methods

- General least squares problems,
  - Residual form (real function),

$$\min_{x \in R^n} F(x) = \frac{1}{2} \sum_{i=1}^{m} {r_i}^2(x), m \geq n$$

where $r_i(x(t_i)) = y_i - f(x(t_i))$ (e.g. data fitting). Or,

$$\min_{x \in R^n} \frac{1}{2} \|r(x)\|_2^2, r \in R^m$$

  - 1st order derivative,

$$\nabla F(x) = \sum_{i=1}^{m} r_i(x) \nabla r_i(x) = J(x)^T r(x)$$

where, Jacobian $J(x) = \left(\frac{\partial r_i}{\partial x_j} \right)_{\substack{i=1,\ldots,m \\ j=1,\ldots,n}}$

– Apply 1st order Taylor expansion to $r(x)$ around optimal solution x^*,

$$\min_x \|r(x^*) + J(x^*)(x - x^*)\|_2$$

This is to solve linear system,

$$\min_x \|J^* x - b^*\|^2$$

Computing pseudoinverse requires $J^{*T} J^*$ is full rank!

– Assumption of iteration scheme,

The sequence of Jacobian $\{J_k\}$ is always full rank!

Rising Question, J_k may be rank deficient!

– Hessian,

$$\nabla^2 F(x)$$

$$= \sum_{j=1}^{m} \nabla r_j(x) \nabla r_j(x)^T + \sum_{j=1}^{m} r_j(x) \nabla^2 r_j(x)$$

$$= J(x)^T J(x) + \sum_{j=1}^{m} r_j(x) \nabla^2 r_j(x)$$

Note that $\nabla J(x) = \nabla^2 r(x)$

The 1st part dominates the others in Hessian;

The 2nd part is highly nonlinear (usually cancelled)!

– Challenging issue is how to approximate Hessian!

- Gaussian-Newton method
  - Recall Newton method
    - Jacobin $J(x)$ is derivative of residual error,

$$J(x)_{ij} = \frac{\partial r_i(x)}{\partial x_j}$$

    - Gradient is derivative of $F(x)$,

$$\nabla F(x)_j = \frac{\partial f(x)}{\partial x_j} = \sum_{i=1}^{n} r_i(x) \frac{\partial r_i(x)}{\partial x_j}$$

    - 1st order derivative (necessary condition)

$$\nabla F(x) = J(x)^T r(x) = 0$$

– If $F(x)$ is C^2 continuous, Newton's Method updates x by,

$$x^{(k+1)} = x^{(k)} - \left(\nabla^2 F\left(x^{(k)}\right)\right)^{-1} \nabla F\left(x^{(k)}\right)$$

$$= x^{(k)} - \left(J_k{}^T J_k + H_k\right)^{-1} J_k{}^T r_k$$

Searching direction $p^{(k)} = x^{(k+1)} - x^{(k)}$, satisfying,

$$(J_k{}^T J_k + H_k) p^{(k)} = -J_k{}^T r_k$$

H_k is the 2nd term of Hessian;

– If H_k is neglected,

This is required by Gauss-Newton method!

– The kernel of GN is its Normal Equation,

$$J_k{}^T J_k p^{(k)} = -J_k{}^T r_k$$

which yields the GN iterative scheme as,

$$x^{(k+1)} = x^{(k)} - \left(J_k{}^T J_k\right)^{-1} J_k{}^T r_k$$

– When J_k is full rank, Normal Equation is regarded as a linear LS problem, i.e. "$Ax = b$"!

$$\min_{p^{(k)}} \left\| J_k p^{(k)} + r_k \right\|^2$$

This means that it is unnecessary to compute Hessian approximation $J_k{}^T J_k$. (There exist QR/SVD algorithms for this linear LS problem!)

Is it true a linear LS solution is a stable solution? (note that GN ignores the 2nd term of Hessian! We need to evaluate the influence of the 2nd term!)

– Question: GN iterative scheme convergences?

  - This is a descent step with direction $p^{(k)}$. Consider 1st order Tayler expansion,

$$F(x) = F\left(x^{(k)}\right) + \nabla F\left(x^{(k)}\right)^T p^{(k)}$$

which is expected to $F(x) - F\left(x^{(k)}\right) \leq 0$. So,

$$p^{(k)^T} \nabla F\left(x^{(k)}\right) = p^{(k)^T} J_k{}^T r_k = -p^{(k)^T} J_k{}^T J_k p^{(k)}$$
$$= -\left\| J_k p^{(k)} \right\|^2 \leq 0$$

This means, reasonable solution $x^{(k)}$ leads to

$$J_k p^{(k)} = 0$$

further leads to $J_k{}^T r_k = \nabla F\left(x^{(k)}\right) = 0$;

- Convergence,

$$x^{(k)} + p^{(k)} - x^*$$
$$= x^{(k)} - x^* - \left(J^T J\right)^{-1} \nabla F\left(x^{(k)}\right)$$
$$= \left(J^T J\right)^{-1} \left(J^T J\left(x^{(k)}\right)\left(x^{(k)} - x^*\right) - \left(\nabla F\left(x^{(k)}\right) - \nabla F(x^*)\right)\right)$$

Note $\nabla F\left(x^{(k)}\right) = {J_k}^T r_k, \nabla F(x^*) = 0.$

Recall the definition of Hessian Matrix,

$$\nabla F\left(x^{(k)}\right) - \nabla F(x^*)$$
$$= \int_0^1 J\left(x^* + t\left(x^{(k)} - x^*\right)\right)^T J\left(x^* + t\left(x^{(k)} - x^*\right)\right)\left(x^{(k)} - x^*\right)dt$$
$$+ \int_0^1 H\left(x^* + t\left(x^{(k)} - x^*\right)\right)\left(x^{(k)} - x^*\right)dt$$

Submitting it into iterative scheme yields……

- Norm form,

$$\left\|x^{(k)}+p^{(k)}-x^{*}\right\|$$

$$\leq \int_{0}^{1}\left\|\left(J^{T}J\right)^{-1}H\left(x^{*}+t\left(x^{(k)}-x^{*}\right)\right)\right\|\left\|x^{(k)}-x^{*}\right\|dt+O\left(\left\|x^{(k)}-x^{*}\right\|^{2}\right)$$

$$\approx \left\|\left(J^{T}J(x^{*})\right)^{-1}H(x^{*})\right\|\left\|x^{(k)}-x^{*}\right\|+O\left(\left\|x^{(k)}-x^{*}\right\|^{2}\right)$$

It is desirable, $\left\|\left(J^{T}J(x^{*})\right)^{-1}H(x^{*})\right\|<1$!!!

(i.e. if satisfying it, GN works well; otherwise to be failed! This is the assumption of GN!)

In practice, we don't know $H(x^{*})$!

Recall the linear system, $\min_{p^{(k)}}\left\|J_{k}p^{(k)}+r_{k}\right\|^{2}$

We have to determine in WHICH scenario the linear LS solution is stable!!!

- To solve Linear LS problem,

$$\min_{p^{(k)}} \left\| J_k p^{(k)} + r_k \right\|^2$$

  - Case 1: overdetermined problem

    Let Jacobian $J \in R^{m \times n}$, direction $p \in R^n$, and $m > n$; and x^* be a minimizer, to solve *p* by,

$$J_k p^{(k)} = r_*$$

    i.e. $p^{(k)} = x^* - x^{(k)}$, r_* residual error w.r.t. optimal solution: $y - y^*$!

- Consider $J_k^T J_k p^{(k)} = J_k^T r_*$

  Change it as,

$$J_k^T J_k p^{(k)} = \left(J_k^T - J_*^T\right) r_*$$

  Because $J_*^T r_* = 0$, (this is the 1st order necessary condition!)

$$J_k^T r_* = \left(J_* + \nabla J_*(x - x^*)\right)^T r_* + O\left(\|\varepsilon\|^2\right)$$
$$= (x - x^*)^T \nabla J_*^T r_* + O\left(\|\varepsilon\|^2\right)$$

  Recall,

$$\nabla^2 F(x) = J(x)^T J(x) + \sum_{j=1}^{m} r_j(x) \nabla^2 r_j(x)$$

  We have,

$$\nabla J_*^T r_* = \nabla^2 F(x^*) - J_*^T J_*$$

  Further lead to

$$J_k^T r_* = (x - x^*)^T \left(\nabla^2 F(x^*) - J_*^T J_*\right) + O\left(\|\varepsilon\|^2\right)$$

- Conclude the estimation upper bound

(1) If $\|r_*\|$ is large,

$$\left\|J_k^T J_k p^{(k)}\right\| = \left\|J_k^T r_*\right\| \leq \left\|\left(\nabla^2 F(x^*) - J_*^T J_*\right)\right\| \|r_*\| + O\left(\|\varepsilon\|^2\right)$$

This says that even for a large residual problem, it can converge fast if it is not very nonlinear!

(2) If $\|x - x^*\|$ is large (i.e. distant point *x*)

$$\left\|J_k^T J_k p^{(k)}\right\| = \left\|J_k^T r_*\right\|$$
$$\leq \left\|\left(\nabla^2 F(x^*) - J_*^T J_*\right)\right\| \|x - x^*\| + O\left(\|\varepsilon\|^2\right)$$

This says that even for initial *x* far from x^* or large residual problem (but very nonlinear), there is no reason to expect GN to work well because iteration cannot decrease the upper bound!

- Case 2: Underdetermined problem ($m < n$)
  - Applying SVD to J_k yields pseudoinverse J_k^+,

$$p^{(k)} = J_k^+ r_k$$

The pseudoinverse solution is a particular solution but not an unique solution,

The general solution is,

$$p = J_k^+ r_k + Fz$$

i.e. there is no unique solution.

Note that Gauss-Newton cannot guarantee to converge to an unique solution since it is underdetermined!

- Damped Gauss-Newton
  - Recall Armijo Rule

    Sufficient decrease: for $F\left(x^k\right)$, actual decrease is equal to the predicated decrease up to a scale.

    Search on a line from x^k in direction of locally decreasing $F(x)$,

    Start with $m = 0$ then increase m until sufficient decrease is achieved, i.e. $\lambda = \alpha^m, \alpha \in (0,1)$

    This is called “backtracking”/”pullbacks”,

    For each m, a new function evaluation is required!

– Recall Gauss-Newton iterative scheme, if J_k is full rank, the search direction $p^{(k)}$,

$$p^{(k)} = x^{k+1} - x^k = -(J_k^T J_k)^{-1} J_k^T r_k$$

The search direction is a descend direction!

Recall Newton Method for comparison,

$$x = x^k - \left(\nabla^2 F(x^k)\right)^{-1} \nabla F(x^k)$$

– Applying Armijo Rule to GN yields Damped GN

$$p^{(k)} \leftarrow \alpha^m p^{(k)}, \alpha \in (0,1)$$

For each *m*, the sufficient decrease is required, which results in the iterative scheme as,

$$x^{(k+1)} = x^{(k)} - \alpha^m (J_k^T J_k)^{-1} J_k^T r_k$$

– Example

i	1	2	3	4	5	6	7
{x}	0.038	0.194	0.425	0.626	1.253	2.500	3.740
{y}	0.050	0.127	0.094	0.2122	0.2729	0.2665	0.3317

$$r_i = y_i - \frac{\beta_1 x_i}{\beta_2 + x_i} \qquad i = 1, \ldots, 7$$

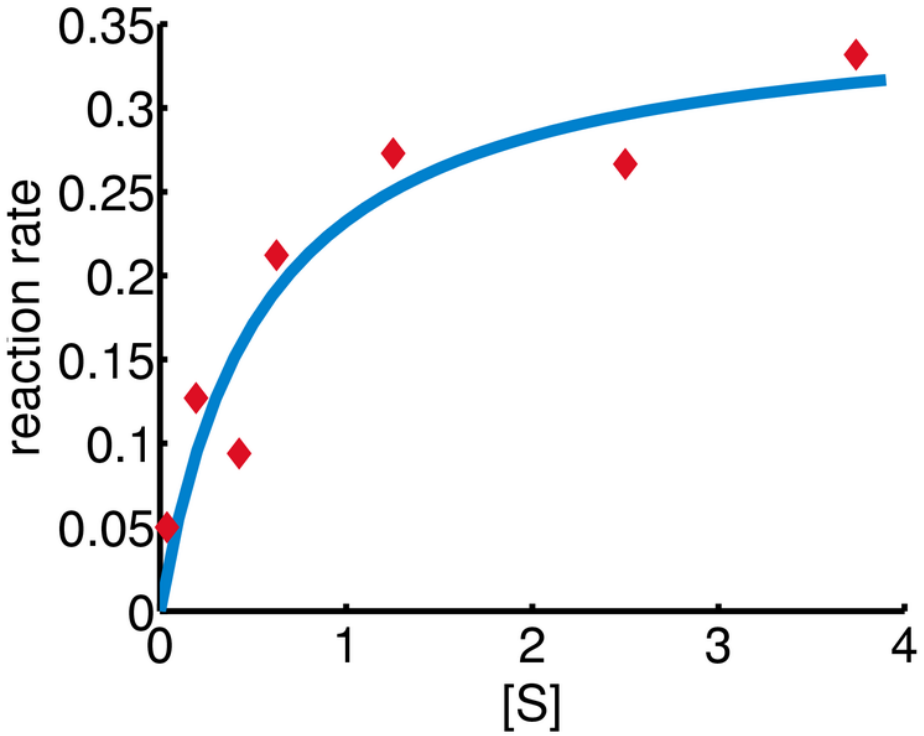
Initial guess of variables $\beta_1 = 0.9$ and $\beta_2 = 0.2$,

1st order derivative, $J(x)_{ij} = \left(\frac{\partial r_i}{\partial \beta_j}\right)_{7\times 2} =$

$$\left(-\frac{x_i}{\beta_2 + x_i}, \frac{\beta_1 x_i}{(\beta_2 + x_i)^2}\right)$$

– Plot result

5 iterations, convergence to $\beta^* = (0.362, 0.556)$



- Levenberg-Marquardt method
  - Jacobian, $J(x)$, may be not full rank!

    The same idea, remove the 2nd order term of Hessian;

    Recall Gauss-Newton's method, compute the searching direction *p*,

$$J_k{}^T J_k p^{(k)} = -J_k{}^T r_k$$

  - Basis idea: within a small region, Hessian is SPD!

$$\min_p \frac{1}{2} \|J_k p + r_k\|^2$$

    where, $\|p\| \leq \gamma$, i.e. strategy changing linear search to trust region!!!

– Add constraints to nonlinear LS problem,

$$\min_p \frac{1}{2}\|J_k p + r_k\|^2 + \alpha_k \|p\|^2$$

where α_k is Lagrange multiplier for constraint.

– Question,

If *p* is outside region γ_k, J_k may be not full rank!

- Modify the searching direction,

$$\left(J_k^T J_k + \rho I\right) p^{(k)} = -J_k^T r_k$$

in case of any ill-conditioned Jacobian.

- Formulate it as a least squares problem,

$$\min_p \frac{1}{2}\left\|\begin{pmatrix} J \\ \sqrt{\rho}I \end{pmatrix} p + \begin{pmatrix} r \\ 0 \end{pmatrix}\right\|^2$$

  Then, update $p^{(k)}$ and further update $x^{(k)}$……

  Note: bound constraint is changed to parameter ρ!

- Rising Questions,

  If ρ is given, direction p can be updated by…

  Basic idea is to try any possible ρ……

  How to determine ρ?

  If direction $p^{(k)}$ is given, how to find ρ?

  Note that ρ_k may vary from 0 to a large step size.

  However, initial value, $\rho_0 = \left\|J_0^T J_0\right\|$

- Fast computation of solving $\begin{pmatrix} J \\ \sqrt{\rho} I \end{pmatrix} p = \begin{pmatrix} r \\ 0 \end{pmatrix}$,

  If applying QR decomposition,

$$\begin{pmatrix} J \\ \sqrt{\rho} I \end{pmatrix} = Q_\rho \begin{pmatrix} U_\rho \\ 0 \end{pmatrix}$$

  where, $U_\rho^T U_\rho = J^T J + \rho I$;

  Re-formulate Equation as,

$$Q_\rho \begin{pmatrix} U_\rho \\ 0 \end{pmatrix} p = \begin{pmatrix} r \\ 0 \end{pmatrix}$$

$$U_\rho p = Q_\rho^T r$$

  A proper ρ makes U_ρ full-rank!

  Thus, we need the closed forms of Q_ρ and U_ρ!

- Speeding up QR computation!

  Applying QR factoring to Jacobian yields

$$J = Q\begin{pmatrix} U \\ 0 \end{pmatrix}$$

$$\begin{pmatrix} J \\ \sqrt{\rho} I \end{pmatrix} = \begin{pmatrix} Q & \\ & I \end{pmatrix}\begin{pmatrix} U \\ 0 \\ \sqrt{\rho} I \end{pmatrix}$$

  Then, taking Givens Rotation $\bar{Q}$, such that,

$$\bar{Q}\begin{pmatrix} U \\ 0 \\ \sqrt{\rho} I \end{pmatrix} = \begin{pmatrix} U_\rho \\ 0 \\ 0 \end{pmatrix}$$

  Define, $Q_\rho = \begin{pmatrix} Q & \\ & I \end{pmatrix}\bar{Q}^T$!

  Each updating Q_ρ for new attempt ρ only needs to re-compute $\bar{Q}$!

  U_ρ has been available when computing $\bar{Q}$!

- Compute direction p by $U_\rho p = Q_\rho^T r$

- Summary
  - Newton method

$$F(x) = F(x^k) + \nabla F(x^k)^T (x - x^k) + \frac{1}{2}(x - x^k)^T \nabla^2 F(x^k)(x - x^k)$$

  - Gauss-Newton method,

$$F(x) = F(x^k) + \nabla F(x^k)^T (x - x^k) + \frac{1}{2}(x - x^k)^T J_k{}^T J_k (x - x^k)$$

  - Damped Gauss-Newton method,

$$F(x) = F(x^k) + \nabla F(x^k)^T (x - x^k) + \frac{1}{2\alpha^m}(x - x^k)^T J_k{}^T J_k (x - x^k)$$

  - Levenberg-Marquardt Alg.

$$F(x) = F(x^k) + \nabla F(x^k)^T (x - x^k) + \frac{1}{2}(x - x^k)^T (J_k{}^T J_k + \rho I)(x - x^k)$$

5. Orthogonal Distance Regression

- Total least squares problem
  - Consider a general noisy system,

$$\min_{x,E} \|(A+E)x - b\|_2^2$$

where, E and $r = (A+E)x - b$ both are perturbations!

Comparing it with an usual noisy system,

$$\min_{x} \|Ax - b\|_2^2$$

Basic idea is to consider noises both from observation and from system!

  - For nonlinear systems,

$$r_i = \varphi(x, t_i + \delta_i) - b_i, \qquad i = 1, \dots, m$$

Note that there are 2 kinds of variables here, x, δ while r is an implicit variable.

- Orthogonal distance regression,
  - Basic idea, minimizing two norms (residual r and input perturbation δ!)

$$\min_{x,\delta_i,r_i} \frac{1}{2}\sum_{i=1}^{m} {w_i}^2 {r_i}^2 + {d_i}^2 {\delta_i}^2$$

where w_i, d_i are given.

Rewrite it as, (eliminating residual-implicit variable *r*!)

$$\min_{x,\delta_i} \frac{1}{2}\sum_{i=1}^{2m} s_i^2(x,\delta)$$

where,

$$s_i(x,\delta) = \begin{cases} w_i\big(b_i - \varphi(x, t_i + \delta_i)\big), & i = 1,\dots,m \\ d_{i-m}\delta_{i-m}, & i = m+1,\dots,2m \end{cases}$$

– Take 1st order derivative to s_i,

$$\frac{\partial s_i}{\partial \delta_j} = \frac{\partial(\varphi(t_i + \delta_i, x) - b_i)}{\partial \delta_j} = 0, \qquad i, j = 1, \dots, m, i \neq j$$

$$\frac{\partial s_i}{\partial x_j} = 0, \qquad j = 1, \dots, n, i = 1, \dots, m$$

$$\frac{\partial s_{m+i}}{\partial \delta_j} = \begin{cases} d_i, & if\ i = j \\ 0, & otherwise \end{cases}$$

– Partition the Jacobian as,

$$J(x, \delta) = \begin{pmatrix} A & V \\ 0 & D \end{pmatrix}$$

where, *A* is the partial derivatives w.r.t *x*; *V* and *D* are the partial derivatives w.r.t δ and are diagonal matrices with size *m* by *m*.

– Normal equation,

$$\begin{pmatrix} A^T A + \rho I & A^T V \\ VA & V^2 + D^2 + \rho I \end{pmatrix} \begin{pmatrix} p_x \\ p_\delta \end{pmatrix} = - \begin{pmatrix} A^T r_x \\ V r_x + D r_\delta \end{pmatrix}$$

where vectors $p_{x,\delta}$ contain *x* and δ!

– This is solved by standard L-M Alg.

$$\min_{p_x, p_\delta, \rho} \frac{1}{2} \left\| \begin{pmatrix} J \\ \sqrt{\rho} I \end{pmatrix} \begin{pmatrix} p_x \\ p_\delta \end{pmatrix} + \begin{pmatrix} r_x \\ r_\delta \\ 0 \end{pmatrix} \right\|^2$$

6. Coursework

- Reference,
  - Image deformation using moving least squares, at http://faculty.cs.tamu.edu/schaefer/research/mls.pdf
  - Codes, at http://www.mathworks.co.uk/matlabcentral/fileexchange/12249-moving-least-squares

- Image deformation,
  - Given a set of control point pairs, do a specified affine transformation on a domain according to the control point pairs;
  - For an arbitrary point *x*, define the affine F_x as,

$$\min_{F_x} \sum_i w_i(x) \|F_x(p_i) - q_i\|^2$$

where, control point pairs, (p_i, q_i); $F_x(p_i) \rightarrow q_i$!

Let weight $w_i(x) = \frac{1}{\|p_i - x\|^2}$;

– Affine transform,

- $F_x(p) = Mp + T \rightarrow q$
- Firstly solve *T* by,

$$T = arg\min_{T} \sum_{i} w_i \langle Mp_i + T - q_i, Mp_i + T - q_i \rangle$$

Yield, $T(x) = q^*(x) - Mp^*(x)$,

where, $q^* = \sum_i w_i q_i / \sum_i w_i$ and $p^* = \sum_i w_i p_i / \sum_i w_i$

– Rewrite Affine Transform,

- $F_x(p) = M(p - p^*) + q^*$
- Then solve *M* by,

$$\min_M \sum_i w_i \|M(p_i - p^*) - (q_i - q^*)\|^2$$

- The 1st order derivative w.r.t. *M*,

$$(p_i - p^*) w_i \left(\left(p_i^T - p^{*T} \right) M^T - \left(q_i^T - q^{*T} \right) \right) = 0, i = 1..n$$

- Local solution at point *x*,

$$M = \left(\sum_i (p_i - p^*)^T w_i (p_i - p^*) \right)^{-1} \sum_j w_j \, (p_j - p^*)^T (q_j - q^*)$$

- Mapping *x* by Affine *M*,

$$x' = M(x - p^*) + q^*$$

Moreover, compute the pointwise affine *M* for updating!

Note that, many info of *M* can be pre-computed!

There exist non-uniform scaling and shear…

– Consider similarity deformation,

- Constrain, $M^T M = sI$!
- Let $M^T = (M_1, M_2)_{2\times 2}$,
- $M_2 = M_1^{\perp}$
- $M_1^{\perp} p = M_1 p^{\perp}$
- $p^T M^T = p^T \left(M_1, M_1^{\perp}\right) = \begin{pmatrix} p^T \\ p^{\perp} \end{pmatrix} M_1$
- $\min_{M_1} \sum_i w_i \left\| \begin{pmatrix} (p_i - p^*)^T \\ -(p_i - p^*)^{\perp} \end{pmatrix} M_1 - (q_i - q^*) \right\|^2$

- Solution,

$$M_1 = \left(\begin{pmatrix} (p_i - p^*)^T \\ -(p_i - p^*)^\perp \end{pmatrix}^T \begin{pmatrix} (p_i - p^*)^T \\ -(p_i - p^*)^\perp \end{pmatrix} \right)^{-1} \begin{pmatrix} (p_i - p^*)^T \\ -(p_i - p^*)^\perp \end{pmatrix}^T (q_i - q^*)$$

$$M_2 = \left(\begin{pmatrix} (p_i - p^*)^T \\ -(p_i - p^*)^\perp \end{pmatrix}^T \begin{pmatrix} (p_i - p^*)^T \\ -(p_i - p^*)^\perp \end{pmatrix} \right)^{-1} \begin{pmatrix} (p_i - p^*)^T \\ -(p_i - p^*)^\perp \end{pmatrix}^T (-q_i + q^*)^\perp$$

- Local solution at point *x*,

$$M = \frac{1}{\mu} \sum_i w_i \begin{pmatrix} (p_i - p^*)^T \\ -(p_i - p^*)^\perp \end{pmatrix} \left((q_i - q^*), -(q_i - q^*)^\perp \right)$$

where, $\mu = \sum_i w_i \, (p_i - p^*)^T (p_i - p^*)$

- Mapping *x* by similarity *M*,

$$x' = M(x - p^*) + q^*$$

Note that, many info of *M* can be pre-computed, but *M* should be computed point to point!

– Rigid deformation

- $\min_{s,R} \sum_i w_i \|sR(p_i - p^*) - (q_i - q^*)\|^2$

s.t. $R^T R = I, s = 1;$

- Local solution at point *x*,
- $M = \frac{1}{\mu}\sum_i w_i \begin{pmatrix} (p_i - p^*)^T \\ -(p_i - p^*)^\perp \end{pmatrix} \left((q_i - q^*), -(q_i - q^*)^\perp\right)$
- It is the same as the solution of similarity deformation!
- Factor μ,

$$\mu = \sqrt{\left(\sum_i w_i \, (q_i - q^*)^T (p_i - p^*)\right)^2 + \left(\sum_i w_i \, (q_i - q^*)^T (p_i - p^*)^\perp\right)^2}$$

Note that, we cannot pre-compute it.