Lecture Outline

- Least-square problems;
- Linear least-squares Problem;
- Statistical justification for Least-squares;
- Linear least-squares problem and regularisation;
- Nonlinear least-squares and Gauss-Newton method.

You should be able to . . .

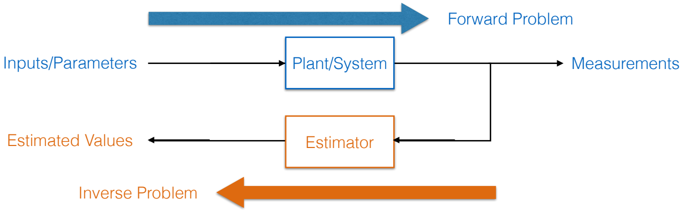
- Recognise and formulate least-square problems;
- Solve linear least-square problems;
- Identify the regualrised version of least-square problem;
- Solve nonlinear least-square problems.

Least-Square Problems

- The objective function:

$$f(x) = \frac{1}{2} \sum_{i=1}^{m} r_i^2(x)$$

- $r_i : \mathbb{R}^n \to \mathbb{R}$ is smooth and is called a *residual.*
- Standing assumption: $m \geq n$ (unless stated otherwise).
- Least squares appear in many places (largest source of unconstrained optimisation problems).
- The residuals capture the discrepancy between the model and the observed behavior of the system



Least-Square Problems

- Devise efficient, robust minimization algorithms by exploiting the special structure of the function f and its derivatives.
- Assemble the *residual vector* $r(x) = [r_1(x), \ldots, r_m(x)]^T$, thus,

$$f(x) = \frac{1}{2}\|r(x)\|_2^2$$

- The derivative of $f(x)$ can be written in terms of the $m \times n$ *Jacobian matrix* $J(x)$:

$$J(x) = \begin{bmatrix} \nabla r_1(x)^T \\ \vdots \\ \nabla r_m(x)^T \end{bmatrix}$$

Least-Square Problems

- Consequently,

$$
\begin{aligned}
\nabla f(x) &= \sum_{i=1}^{m} r_i(x) \nabla r_i(x) = J(x)^T r(x) \\
\nabla^2 f(x) &= \sum_{i=1}^{m} \nabla r_i(x) \nabla r_i(x)^T + \sum_{i=1}^{m} r_i(x) \nabla^2 r_i(x) \\
&= J(x)^T J(x) + \sum_{i=1}^{m} r_i(x) \nabla^2 r_i(x)
\end{aligned}
$$

- Often the Jacobian is easy to compute.
- Having the Jacobian gives us access to the first part of the Hessian for "free".
- The term $J(x)^T J(x)$ is often more important:
  1. Near affineness of the residuals near the solution ($\nabla^2 r_i$ is small).
  2. The residuals are relatively small (r_i is small).

Fixed-Regressor Model

- The goal is to estimate the parameters of a model, ϕ, for a system.
- The output of the system is denoted by y, the input by t, and the parameters to be estimated by x. Ideally, one expects

$$\boxed{\phi(t;x) = y(t).}$$

- It is assumed that the pair of inputs and output $(t, y(t))$ can be perfectly measured.
- Assume there are m different inputs, t_i, $i = 1, \ldots, m$. The goal then is to find the parameters x via minimising the discrepancies:

$$\boxed{f(x) = \sum_{i=1}^{m} \|\phi(t_i;x) - y(t_i)\|^2.}$$

Example from *James, Witten, Hastie, and Tibshirani, "Introduction to Statistical Learning", p. 267.*

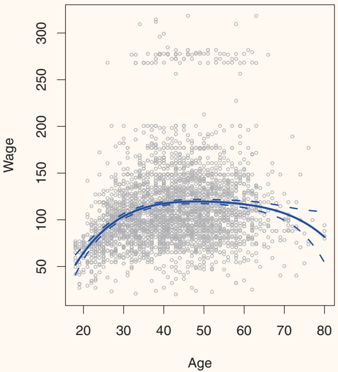
Example: Polynomial Model For Age Versus Age Data

Income and demographic information for males in the central Atlantic region of the United States. The model is assumed to be a degree 4 polynomial:

$$\phi(t;x) = x_0 + x_1 t + x_2 t^2 + x_3 t^3 + x_4 t^4$$

$$f(x) = \sum_{i=1}^{62} \|\phi(t_i;x) - y(t_i)\|^2.$$

Note that $r_i(x) = \phi(t_i;x) - y(t_i)$ is linear in unknown x.



Note that $m = 62$ and t_i is known perfectly.

Statistical Justification For Least-Squares

- Assume $r_i(x) = \phi(t_i; x) - y(t_i)$ are independent and identically distributed (IID) with a certain variance σ_i^2 and probability density function $g_i(\cdot)$.
- The *likelihood* of observing a particular set of measurements y_i, $i = 1, \ldots, m$ given the unknown parameter is actually x is

$$\begin{aligned} P(y_1, \ldots, y_m | x) &= \prod_{i=1}^{m} P(y_i | x) \\ &= \prod_{i=1}^{m} g_i(\phi(t_i; x) - y(t_i)) \end{aligned}$$

- The x that maximises this likelihood is called *the maximum likelihood estimate.*

Statistical Justification For Least-Squares

- Now assume the discrepancies follow a normal distribution, i.e.

$$g_i(r) = \frac{1}{\sqrt{2\pi\sigma_i^2}} \exp\left(-\frac{r^2}{2\sigma_i^2}\right)$$

- The likelihood function becomes

$$\boxed{P(y_1, \ldots, y_m|x) = c \exp\left(-\sum \frac{(\phi(t_i;x) - y(t_i))^2}{2\sigma_i^2}\right)}$$

- where constant $c = \prod_{i=1}^{m} (2\pi\sigma_i^2)^{-1/2}$.
- It is obvious that the likelihood is maximised if the following function is minimised:

$$\boxed{f(x) = \frac{1}{2} \sum \frac{(\phi(t_i;x) - y(t_i))^2}{\sigma_i^2}}$$

Statistical Justification For Least-Squares

- The measurements are scaled/weighted by the covariance of the noise in that measurement. It is a measure of quality.
- Alternatively,

$$\boxed{\begin{aligned} f(x) &= \frac{1}{2} r(x)^T S^{-1} r(x) \\ S &= \operatorname{diag}(\sigma_1^2, \ldots, \sigma_m^2) \end{aligned}}$$

- For the case that $\sigma_1 = \cdots = \sigma_m$ then the likelihood is maximised if the following function is minimised:

$$\boxed{\begin{aligned} f(x) &= \frac{1}{2} \sum (\phi(t_i; x) - y(t_i))^2 \\ &= \frac{1}{2} \sum r_i^2(x) \end{aligned}}$$

Linear Least-Squares Problem

- In many situations $\phi(t;x)$ is a linear function of x, e.g. polynomial fitting as seen before, or similar basis function fits.
- Then $r(x) = Jx - y$ for some matrix J and vector y.
- For cost function and its derivatives we have

$$\boxed{\begin{aligned} f(x) &= \frac{1}{2}\|Jx - y\|^2 \\ \nabla f(x) &= J^T(Jx - y), \quad \nabla^2 f(x) = J^T J \end{aligned}}$$

- Note that $f(x)$ is convex (not necessarily true for the general case.)
- Thus, any $x^\star$ that results in $\nabla f(x^\star) = 0$ is the global minimiser of $f(x)$:

$$J^T J x^\star = J^T y$$

- This is known as the *normal equations* for $f(x)$.

Linear Least-Squares Problem

- Let's assume $m \geq n$ and J is full column rank.
- Three ways to solve the system of equations $J^T J x^\star = J^T y$:
    1. Cholesky Factorisation
    2. QR Factorisation
    3. Singular Value Decomposition (SVD)
- The Cholesky-based algorithm is particularly useful when $m \gg n$ and it is practical to store $J^T J$ but not J itself or when J is sparse.
- This approach must be modified when J is rank-deficient or ill conditioned to allow pivoting of the diagonal elements of $J^T J$.
- The QR approach avoids squaring of the condition number and hence may be more numerically robust.
- The SVD approach is the most robust and reliable of all, and potentially the most expensive. It also provides the sensitivity of the solution to perturbations in y or J.

Linear Least-Squares Problem: Cholesky Factorisation

- One obvious approach is to solve $J^T J x^\star = J^T y$.
  1. compute $J^T J$ and $J^T y$;
  2. compute the Cholesky factorisation of the symmetric matrix $J^T J$;
  3. perform two triangular substitutions with the Cholesky factors to recover the solution $x^\star$.
- Chelosky factorisation is possible for $m \geq n$, and $\text{Rank}(J) = n$:

$$J^T J = C^T C$$

- $C \in \mathbb{R}^{n \times n}$ and upper triangular.
- Solve the triangular systems $C^T \zeta = b$ where $b = J^T y$ and $Cx = \zeta$ to find $x^\star$.
- The relative error in the computed solution of a problem is usually proportional to the condition number, this method depends on the $\kappa(J^T J)$ which is the square of $\kappa(J)$.
- When J is ill conditioned, the Cholesky factorisation process may break down.

Linear Least-Squares Problem: QR Factorisation

- Note, for orthogonal $Q \in \mathbb{R}^{m \times m}$:

$$\boxed{\|Jx - y\| = \|Q^T(Jx - y)\|}$$

- Perform a QR factorisation on J with column pivoting:

$$J \overbrace{\Pi}^{\substack{\text{permutation}\\ \text{matrix and}\\ \text{orthogonal}}} = \overbrace{Q}^{\text{orthogonal}} \begin{bmatrix} \overbrace{R}^{\substack{\text{upper triangular}\\ \text{positive diagonals}}} \\ 0 \end{bmatrix} = \overbrace{Q_1}^{\substack{\text{the first } n\\ \text{columns of } Q}} R$$

- $\Pi \in \mathbb{R}^{n \times n}$, $Q = \begin{bmatrix} Q_1 & Q_2 \end{bmatrix} \in \mathbb{R}^{m \times m}$, $R \in \mathbb{R}^{n \times n}$.

Linear Least-Squares Problem: QR Factorisation

- From $\|Jx - y\| = \|Q^T(Jx - y)\|$ and the factorisation:

$$\|Jx - y\|^2 = \|R(\Pi^T x) - Q_1^T y\|^2 + \|Q_2^T y\|^2$$

- The second summand is independent of x.
- $\|Jx - y\|$ is minimised by

$$\boxed{x^\star = \Pi R^{-1} Q_1^T y}$$

- First solve $R\zeta = Q_1^T y$ then permute ζ to obtain x: $x^\star = \Pi\zeta$.
- The relative error in the final computed solution $x^\star$ is usually proportional to $\kappa(J)$, not its square.
- For greater robustness or more information about the sensitivity of the solution to perturbations in the data (J or y), SVD is used.

Linear Least-Squares Problem: Singular-value Decomposition (SVD)

- The SVD of J:

$$J = \overbrace{U}^{\text{orthogonal}} \begin{bmatrix} \overbrace{S}^{\substack{\text{diagonal}\\ \text{with}\\ \text{positive}\\ \text{elements}}} \\ 0 \end{bmatrix} \overbrace{V^T}^{\text{orthogonal}} = \overbrace{U_1}^{\substack{\text{the first } n\\ \text{columns of } U}} S V^T$$

- $U = \begin{bmatrix} U_1 & U_2 \end{bmatrix} \in \mathbb{R}^{m \times m}$, $S \in \mathbb{R}^{n \times n} = \operatorname{diag}(\sigma_1, \ldots, \sigma_n)$ where $\sigma_1 \geq \cdots \geq \sigma_n > 0$, $V \in \mathbb{R}^{n \times n}$.
- Note that $J^T J = V S^2 V^T$ so the columns of V are the eigenvectors of $J^T J$ with eigenvalues σ_i^2, $i = 1, \ldots, n$.

Linear Least-Squares Problem: SVD

- From $\|Jx - y\| = \|U^T(Jx - y)\|$ and the factorisation:
$$\|Jx - y\|^2 = \|S(V^T x) - U_1^T y\|^2 + \|U_2^T y\|^2$$
- Again the second summand is independent of x.
- $\|Jx - y\|$ is minimised by
$$\boxed{x^\star = V S^{-1} U_1^T y}$$
- Let u_i and v_i be the the i-th column of U and V respectively:
$$\boxed{x^\star = \sum_{i=1}^{n} \frac{u_i^T y}{\sigma_i} v_i}$$
- For small σ_i, $x^\star$ is particularly sensitive to perturbations in y or J that affect $u_i^T y$.
- This is important especially when $\kappa(S) \gg 1$.

Linear Least-Squares Problem: SVD

- When $\text{Rank}(J) = n$ but $\kappa(S) \gg 1$, the last few singular values $\sigma_n, \sigma_{n-1}, \ldots$ are small relative to σ_1.
- So an approximate solution that is less sensitive to perturbations than the true solution can be obtained by omitting these terms from the summation.
- When J is rank deficient some σ_i are exactly zero, then:

$$\boxed{x^\star = \sum_{i \in \{j | \sigma_j \neq 0\}} \frac{u_i^T y}{\sigma_i} v_i + \sum_{i \in \{j | \sigma_j = 0\}} \tau_i v_i}$$

- Often, the solution with smallest norm is the most desirable, and we obtain it by setting $\tau_i = 0$.
- For very large problem one can apply iterative techniques to solve the normal equations.

Linear Least-Squares Problem and Moore-Penrose Pseudoinverse

- For the case where $J^T J$ is invertible, the solution is the form:

$$x^\star = \overbrace{(J^T J)^{-1} J^T}^{J^\dagger} y$$

- $J^\dagger$ is called the Moore-Penrose pseudoinverse.

Definition (Moore-Penrose Pseudoinverse): *Let $J = USV^T$ be the singular value decomposition (SVD) of $J \in \mathbb{R}^{m \times n}$. Then, the Moore-Penrose Pseudoinverse, $J^\dagger$, is $J^\dagger = VS^\dagger U^T$ where $S^\dagger$ is obtained by replacing the nonzero entries of the diagonal of matrix S with their inverse and transposing it.*

$$JJ^\dagger J = J,\ J^\dagger J J^\dagger = J^\dagger,\ (JJ^\dagger)^T = JJ^\dagger,\ (J^\dagger J)^T = J^\dagger J$$

Linear Least-Squares Problem and Regularisation

- For the case where J^TJ is invertible, $J^\dagger = (J^TJ)^{-1}J^T$.
- For the case where JJ^T is invertible, $J^\dagger = J^T(JJ^T)^{-1}$.
- The problems where neither J^TJ or JJ^T are invertible are called ill-posed least square problems.
- Regularisation methods are a way to fix this problem.
- **Tikhonov Reqularisation**[i]**:**

$$\min \quad \|Jx - y\|^2 + \frac{1}{\gamma}\|\Gamma x\|^2$$

- Γ is a proper scaling matrix (full-rank and often identity)

$$x^\star = (J^TJ + \frac{1}{\gamma}\Gamma^T\Gamma)^{-1}J^Ty$$

- Note that as $\gamma \to \infty$, $(J^TJ + \frac{1}{\gamma}\Gamma^T\Gamma)^{-1}J^T \to J^\dagger$.

[i] Ridge reqularisation in learning literature.

Nonlinear Least-Squares: The Gauss-Newton Method

- In essence a modified Newtons method with line search.
- Instead of finding the search direction via $\nabla^2 f_k p = -\nabla f_k$, the following is solved:

$$\boxed{J_k^T J_k p_k^{GN} = -J_k^T r_k}$$

- Using the approximation $\nabla^2 f_k \approx J_k^T J_k$, frees us from computing individual Hessians, $\nabla^2 r_i$, $i = 1, \ldots, m$.
- If J_k is computed earlier when calculating $\nabla f_k = J_k^T r_k$[ii], the approximation does not require any more derivation.
- In many situations the first term, $J^T J$, dominates the second term in the Hessian of f (at least close to $x^\star$)
- The convergence rate of Gauss-Newton is similar to that of Newton's method.

[ii] **Notation abuse!** r_k corresponds to the value of r at step k, not the k-th entry of r. Index k always is used to denote the step number.

Nonlinear Least-Squares: The Gauss-Newton Method

- Whenever $\text{Rank}(J_k) = n$ and $\nabla f_k \neq 0$, p_k^{GN} is a descent direction (thus suitable for line-search):

$$(p_k^{GN})^T \nabla f_k = (p_k^{GN})^T J_k^T r_k = -(p_k^{GN})^T J_k^T J_k p_k^{GN}$$
$$= -\|J_k p_k^{GN}\|^2 \leq 0$$

- The inequality is strict unless $J_k p_k^{GN} = 0$ in which case x_k is a stationary point:

$$J_k^T r_k = \nabla f_k = 0.$$

- p_k^{GN} is the solution to the following linear least-squares problem:

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

- The search direction p_k^{GN} can be found by applying linear least-squares algorithms to this subproblem.

Nonlinear Least-Squares: The Gauss-Newton Method

- If QR or SVD are used to solve $\min_p \quad \frac{1}{2}\|J_k p + r_k\|^2$, the Hessian approximation $J_k^T J_k$ does not need to be computed explicitly.
- If $m \gg n$, it may be unwise to store J explicitly.
- The search direction p_k^{GN} can be found by applying linear least-squares algorithms to this subproblem. Instead save r_i and ∇r_i, and compute $J_k^T J_k$ and the gradient vector $J_k^T r_k$:

$$J_k^T J_k = \sum_{i=1}^{m} (\nabla r_j)_k (\nabla r_j)_k^T, \quad J_k^T r_k = \sum_{i=1}^{m} (r_j)_k (\nabla r_j)_k$$

- The equation for p_k^{GN} is obtained from a linear model for the the vector function $r(x_k + p) \approx r_k + J_k p$, i.e. by minimising

$$f(x_k + p) = \frac{1}{2}\|r(x_k + p)\|^2 \approx \frac{1}{2}\|r_k + J_k p\|^2$$

Nonlinear Least-Squares: The Gauss-Newton Method

Implementations of the Gauss-Newton method usually perform a line search in the direction p_k^{GN}, requiring the step length to satisfy conditions like the Armijo and Wolfe conditions.

Theorem (Convergence of Gauss-newton Method): *Suppose each residual function r_i is Lipschitz continuously differentiable in a neighborhood $\mathcal{N}$ of the bounded subevel set $\mathcal{L} = \{x | f(x) \leq f(x_0)\}$ where x_0 is the starting point for the algorithm, and that the Jacobians $J(x)$ satisfy the uniform full-rank condition on $\mathcal{N}$, i.e. $\exists \gamma > 0$ such that $\|J(x)z\| \geq \gamma \|z\|, \forall x \in \mathcal{N}$. Then if the iterates x_k are generated by the Gauss-Newton method with step lengths α_k that satisfy Wolfe conditions, we have*

$$\lim_{k \to \infty} J_k^T r_k = 0.$$

Nonlinear Least-Squares: The Gauss-Newton Method

- $\exists L, \beta > 0$ such that $\forall x, \bar{x} \in \mathcal{N}$ and $i = 1, \ldots, m$:

$$|r_i(x)| \leq \beta, \quad \|\nabla r_i(x)\| \leq \beta$$

$$|r_i(x) - r_i(\bar{x})| \leq L\|x - \bar{x}\|, \quad \|\nabla r_i(x) - \nabla r_i(\bar{x})\| \leq L\|x - \bar{x}\|$$

- Consequently, $\exists \bar{\beta} > 0$ such that $\|J(x)^T\| = \|J(x)\| \leq \bar{\beta}$ for all $x \in \mathcal{L}$.
- The gradient $\nabla f = \sum_{i=1}^{m} r_i \nabla r_i$ is Lipschitz continuous (via results concerning Lipschitz continuity of products and sums).
- Thus, the hypotheses of the **Zoutendijk's Result** are satisfied.
- Now we check the angle θ_k between the search direction p_k^{GN} and $-\nabla f_k$ is uniformly bounded away from $\pi/2$.

Nonlinear Least-Squares: The Gauss-Newton Method

$$
\begin{aligned}
\cos\theta_k &= -\frac{\nabla f_k^T p_k^{GN}}{\|\nabla f_k^T\| \|p_k^{GN}\|} = \frac{\|J_k p_k^{GN}\|^2}{\|p_k^{GN} J_k^T J_k p_k^{GN}\|} \\
&\geq \frac{\gamma^2 \|p_k^{GN}\|^2}{\bar{\beta}^2 \|p_k^{GN}\|^2} = \frac{\gamma^2}{\bar{\beta}^2} > 0.
\end{aligned}
$$

- From the Zoutendijks Result $\nabla f_k \to 0$. $\square$
- If $\text{rank}(J_k) < n$ the subprobelm still can be solved. However there is no guaranteed that θ is uniformly bounded away from $\pi/2$ and one cannot guarantee convergence.
- Convergence rate can be found similar to that of Newton's method.

$$
\begin{aligned}
x_k + p_k^{GN} - x^\star &= x_k - x^\star - [J_k^T J_k]^{-1} \nabla f_k \\
&= [J_k^T J_k]^{-1} \left[[J_k^T J_k](x_k - x^\star) + (\nabla f^\star - \nabla f_k) \right]
\end{aligned}
$$

Nonlinear Least-Squares: The Gauss-Newton Method

- Remember $\nabla^2 f(x) = J(x)^T J(x) + \sum_{i=1}^m r_i(x)\nabla^2 r_i(x)$. Let $M(x) = J(x)^T J(x)$ ($M(x_k) = J_k^T J_k$) and $H(x) = \sum_{i=1}^m r_i(x)\nabla^2 r_i(x)$. Then

$$\begin{aligned}\nabla f_k - \nabla f^\star = &\int_0^1 M(x^\star + \tau(x_k - x^\star))(x_k - x^\star)d\tau \\ &+ \int_0^1 H(x^\star + \tau(x_k - x^\star))(x_k - x^\star)d\tau\end{aligned}$$

- Assuming Lipschitz continuity of H near $x^\star$ yields:

$$\begin{aligned}&\|x_k + p_k^{GN} - x^\star\| \\ &\leq \int_0^1 \|M(x_k)^{-1}H(x^\star + \tau(x_k - x^\star))\|\|x_k - x^\star\|d\tau \\ &\qquad + O(\|x_k - x^\star\|^2) \\ &\approx M(x^\star)^{-1}H(x^\star)\|\|x_k - x^\star\| + O(\|x_k - x^\star\|^2)\end{aligned}$$

Nonlinear Least-Squares: The Gauss-Newton Method

- If $\|[J^T(x^\star)J(x^\star)]^{-1}H(x^\star)\| \ll 1$ the convergence is rapid and when $H(x^\star) = 0$ the convergence is quadratic.
- When n and m are both large and the Jacobian $J(x)$ is sparse, the cost of computing steps exactly by factoring either J_k or $J_k^T J_k$ at each iteration may become quite expensive relative to the cost of function and gradient evaluations.
- Inexact variants of the Gauss-Newton algorithm that are analogous to the inexact Newton algorithms discussed earlier can be used.
- Simply replace the Hessian $\nabla^2 f(x_k)$ in those methods by its approximation $J_k^T J_k$.
- The positive semidefiniteness of this approximation simplifies the resulting algorithms in several places.

Orthogonal Distance Regression

- When we first visited regression we assumed that t variable in the model $\phi(t; x)$ can be measured exactly.
- But this might not be the case (often errors in the input t are much smaller than observations)
- Models that take these errors into account are known in the statistics literature as *errors-in-variables models*
- The resulting optimization problems are referred to as *total least squares* in the case of a linear model, see Golub and Van Loan, or as *orthogonal distance regression* in the nonlinear case.
- Let's introduce perturbations δ_i for each t_i.
- The least-squares problem for positive weights w_i and d_i becomes:

$$\min_{x,\delta} \quad \frac{1}{2} \sum_{i=1}^{m} w_i (y_i - \phi(t_i + \delta_i; x))^2 + d_i \delta_i^2$$

Orthogonal Distance Regression

$$\min_{x,\delta} \quad \frac{1}{2}\sum_{i=1}^{m} w_i |y_i - \phi(t_i+\delta_i;x)|^2 + d_i\delta_i^2 = \frac{1}{2}\sum_{i=1}^{2m} r_i^2(x,\delta)$$

$$\delta = [\delta_1, \ldots, \delta_m]$$

$$r_i(x,\delta) = \begin{cases} \sqrt{w_i}(y_i - \phi(t_i+\delta_i;x)) & i = 1,\ldots,m \\ \sqrt{d_{i-m}}\delta_{i-m} & i = m+1,\ldots,2m \end{cases}$$

- This problem is a standard least-squares problem with $2m$ residuals and $m+n$ unknowns.
- A naive implementation of the existing methods might be quite expensive.
- However, the Jacobian has a nice structure.

Orthogonal Distance Regression

$$\frac{\partial r_i}{\partial \delta_j} = \frac{\partial [y_i - \phi(t_i + \delta_i; x]}{\partial \delta_j} = 0, \quad i \neq j$$

$$\frac{\partial r_i}{\partial x_j} = 0, \quad i = m+1, \ldots, 2m, \ j = 1, \ldots, n$$

$$\frac{\partial r_{m+i}}{\partial \delta_j} = \begin{cases} d_i & i = j \\ 0 & i \neq j \end{cases}$$

$$\boxed{J(x, \delta) = \begin{bmatrix} \hat{J} & V \\ 0 & D \end{bmatrix}}$$

- $V, D \in \mathbb{R}^{m \times m}$ are diagonal and $\hat{J} \in \mathbb{R}^{m \times n}$ is the matrix of partial derivatives of $\phi(t_i + \delta_i; x)$ with respect to x.
- This strucutre can be used to solve for p^{GN}.

Orthogonal Distance Regression

- This strucutre can be used to solve for p^{GN}:

$$\boxed{\begin{bmatrix} \hat{J}^T\hat{J} & \hat{J}^T V \\ V\hat{J} & V^2 + D^2 \end{bmatrix} \begin{bmatrix} p_x^{GN} \\ p_\delta^{GN} \end{bmatrix} = \begin{bmatrix} \hat{J}^T r_1 \\ V r_1 + D r_2 \end{bmatrix}}$$

$$p^{GN} = \begin{bmatrix} p_x^{GN} \\ p_\delta^{GN} \end{bmatrix}, \quad r = \begin{bmatrix} r_1 \\ r_2 \end{bmatrix}$$

- The lower right submatrix $V^2 + D^2$; it is easy to eliminate p_δ^{GN} from this system and obtain a smaller $n \times n$ system to be solved for p_x^{GN}.
- The total cost of finding a step is only marginally greater than for the $m \times n$ problem arising from the standard least-squares model.