

# 1 Solving Nonlinear Equations [by Root Finding $y = 0$ ]

Root Multiplicity,  $m$ :  $0 = f(\bar{x}) = f'(\bar{x}) = \dots = f^{(m-1)}(\bar{x})$  (Simple Root:  $m = 1$ )

$k$ -th Iteration Error:  $e_k = x_k - \bar{x}$       Convergence Rate,  $r$ :  $\lim_{k \rightarrow \infty} \frac{\|e_{k+1}\|}{\|e_k\|^r} = C$  ( $0 < C < 1$  if  $r = 1$ )

## 1.1 One Dimension/Equation skipped a lot

Interval Bisection (Finding  $y = 0$ ):  $[f(a) < 0], [f(b) > 0], [f \text{ is cont.}] \Rightarrow \exists m \text{ s.t. } f(m) = 0$

Fixed-Point Iteration (Finding  $y = x$ ):  $\text{cont. } f(x) = 0 \Rightarrow \text{Find } g(x) = x \rightarrow x_{k+1} = g(x_k)$

$\sim$  Banach-Fixed Point Theorem (there are many FP theorems)

- $g$  is Contractive (over a domain):  $\text{dist}(g(x), g(y)) \leq q \cdot \text{dist}(x, y) \quad q \in [0, 1)$
- $e_{k+1} = [x_{k+1} - \bar{x}] = [g(x_k) - g(\bar{x})] = g'(\xi_k)(x_k - \bar{x}) = g'(\xi_k)e_k$
- $\forall |g'(\xi_k)| < G < 1 \Rightarrow (|e_{k+1}| \leq G|e_k| \leq \dots \leq G^k|e_0|) \Rightarrow \lim_{k \rightarrow \infty} e_k = 0 \quad (G = \max g' \text{ over domain})$
- $\lim_{k \rightarrow \infty} |g'(\xi_k)| = \boxed{\begin{matrix} (0 < |g'(\bar{x})| < 1) \\ \text{(one contractive condition)} \end{matrix}} = C \quad (r = 1)$
- $\boxed{g'(\bar{x}) = 0} \Rightarrow [g(x_k) - g(\bar{x})] = \frac{g''(\xi_k)}{2}(x_k - \bar{x})^2 \Rightarrow \boxed{\left| \frac{g''(\bar{x})}{2} \right| = C} \quad (r = 2 \text{ if } \bar{x} \text{ is an } m = 2 \text{ root of } g)$

Newton's Method (Finding  $y = 0$ ):

$$f(\bar{x}) = 0 = f(x_k + h_k) \approx f(x_k) + f'(x_k)h_k \Rightarrow \boxed{x_{k+1} = x_k + h_k = x_k - \frac{f(x_k)}{f'(x_k)}}$$

- $\boxed{g(x) \equiv x - \frac{f(x)}{f'(x)}} \Rightarrow g(\bar{x}) = \bar{x}, \boxed{g'(\bar{x}) = \frac{f(\bar{x})f''(\bar{x})}{f'(\bar{x})^2} = 0}, \boxed{r = 2} \quad (\text{if } \bar{x} \text{ is a simple root of } f)$
- $\bar{x} \text{ is an } m > 1 \text{ root of } f \Rightarrow \boxed{r = 1, C = 1 - 1/m} \quad (\text{proof not given})$

Secant Method/Linear Interpolation (Finding  $y = 0$ ):

$$f'(x_k) \approx \frac{f(x_k) - f(x_{k-1})}{x_k - x_{k-1}} \quad \text{Approx. } f'(x_k) \text{ with a secant line's slope} \Rightarrow \boxed{x_{k+1} = x_k + h_k = x_k - \frac{x_k - x_{k-1}}{f(x_k) - f(x_{k-1})} f(x_k)}$$

- $\boxed{r = r_+ \approx 1.618} : r_+^2 - r_+ - 1 = 0 \quad (\text{proof hard})$
- Lower cost of iter. offsets the larger number of iter. compared to Newton's Method with derivatives

Inverse Parabolic Interpolation: Use 3 pts to approx. an inverse [sideways] parabola

## 1.2 $m$ Dimensions/System of Equations stuff skipped

Newton's Method (Solving  $\vec{y} = 0$ ):

$$\{J_f(\vec{x})\}_{ij} = \frac{\partial f_i(\vec{x})}{\partial x_j} : \quad J_f(\vec{x}_k)\vec{h}_k = -\vec{f}(\vec{x}_k) \Rightarrow \vec{x}_{k+1} = \vec{x}_k + \vec{h}_k = \vec{x}_k - J_f(\vec{x}_k)^{-1}\vec{f}(\vec{x}_k)$$

- $$\vec{g}(\vec{x}) \equiv \vec{x} - J_f(\vec{x})^{-1}\vec{f}(\vec{x}) \Rightarrow \begin{aligned} J_g(\vec{x}) &= \cancel{I - J_f(\vec{x})^{-1}J_f(\vec{x})} + \sum_{i=1}^n H_i(\vec{x})f_i(\vec{x}) & H_i &= \text{component} \\ &\text{(if } J_f(\vec{x}) \text{ is nonsingular)} & & \text{matrix of the} \\ & & & \text{tensor, } D_x J_f(\vec{x}) \end{aligned}$$

$$= \mathcal{O} \Rightarrow \boxed{r=2} \quad (\text{uh... idk})$$
- $LU$  fact. of the Jacobian costs  $\mathcal{O}(n^3)$

Broyden's [Secant Updating] Method (Solving  $\vec{y} = 0$ ):

$$B_k\vec{h}_k = -\vec{f}(\vec{x}_k) \Rightarrow \vec{x}_{k+1} = \vec{x}_k + \vec{h}_k, \quad B_{k+1} = B_k + \frac{f(\vec{x}_{k+1})h_k^T}{h_k^T h_k} \quad (\text{cost is } \mathcal{O}(n^3))$$

- $B_{k+1}(\vec{x}_{k+1} - \vec{x}_k) = B_{k+1}\vec{h}_k = f(\vec{x}_{k+1}) - f(\vec{x}_k)$
- $B_k$  factorization is updated to factorization of  $B_{k+1}$  at cost  $\mathcal{O}(n^2)$  instead of directly from the above eq.
- Lower cost of iter. offsets the larger number of iter. compared to Newton's Method with derivatives

## 2 Optimizing [By Finding $\min f(\vec{x}) = f(\vec{x})$ ]

### 2.1 Function Shape and Convexity

Coercive:  $\lim_{x \rightarrow \pm\infty} f(x) = \infty$       Unimodal:  $\begin{matrix} a \leq \bar{x} \leq b \\ x_1 < x_2 \end{matrix} : \begin{matrix} x_2 < \bar{x} \rightarrow f(x_1) > f(x_2) \\ \bar{x} < x_1 \rightarrow f(x_1) < f(x_2) \end{matrix}$

$\exists$  global min  $f$  if

- cont.  $f$  on a closed and bounded set
- cont.  $f$  is coercive on a closed, unbounded set
- cont.  $f$  on a set and has a nonempty, closed, and bounded sublevel set
- domain set is unbounded: cont.  $f$  is coercive  $\Leftrightarrow$  all sublevel sets are bounded

$f$  is convex [on a convex set] :

- any sublevel set is convex
- any local min. is a global min

$f$  is strictly convex [on a convex set] :

- any local min. is a unique global min.
- if set is unbounded:  $f$  is coercive  $\Leftrightarrow f$  has a unique global min.

## 2.2 Derivative Tests (Gradient, Jacobian, Hessian) and Lagrangians

Req. :  $\boxed{\text{cont. } f(\bar{x}) = \min f, \text{ cont. } \vec{\nabla} f(\bar{x}), \text{ cont. } H_f(\bar{x})}$

Taylor's Theorem: 
$$\begin{aligned} f(\bar{x} + \vec{s}) - f(\bar{x}) &= \vec{\nabla} f(\bar{x} + \alpha_1 \vec{s}) \cdot \vec{s} = \vec{\nabla} f(\bar{x}) \cdot \vec{s} + \frac{1}{2} \langle \vec{s} | H_f(\bar{x} + \alpha_2 \vec{s}) | \vec{s} \rangle \geq 0 \\ f(\bar{x} + s\hat{u}) - f(\bar{x}) &= \vec{\nabla} f(\bar{x} + \alpha_1 s\hat{u}) \cdot s\hat{u} = \vec{\nabla} f(\bar{x}) \cdot \vec{s} + \frac{s^2}{2} \langle \hat{u} | H_f(\bar{x} + \alpha_2 \vec{s}) | \hat{u} \rangle \end{aligned}$$

- $\lim_{s \rightarrow 0} \left( \frac{f(\bar{x} + \vec{s}) - f(\bar{x})}{s} = \vec{\nabla} f(\bar{x} + \alpha_1 s\hat{u}) \cdot \hat{u} \right) \Rightarrow \left( \vec{\nabla} f(\bar{x}) \cdot \hat{u} \geq 0 \rightarrow \boxed{\vec{\nabla} f(\bar{x}) \cdot \vec{s} \geq 0} \right), \quad \boxed{\text{Cauchy-Schwarz} \rightarrow \max \vec{\nabla} f(\bar{x}) \cdot \hat{u} \text{ if } \vec{u} = \vec{\nabla} f(\bar{x})}$
- $\boxed{\vec{u} = \mp \vec{\nabla} f(\bar{x})} \Rightarrow \lim_{s \rightarrow 0} \left( \frac{f(\bar{x} + \vec{s}) - f(\bar{x})}{s} = \mp \frac{\vec{\nabla} f(\bar{x} + \alpha_1 s\hat{u}) \cdot \vec{\nabla} f(\bar{x})}{\|\vec{\nabla} f(\bar{x})\|} \right) = \mp \|\vec{\nabla} f(\bar{x})\| \leq 0 \quad \boxed{\text{if } \pm \vec{\nabla} f(\bar{x}) \neq 0, \text{ its dir. is an ascent/descent.}}$
- $\lim_{s \rightarrow 0} \left( \frac{f(\bar{x} + \vec{s}) - f(\bar{x}) + f(\bar{x} - \vec{s}) - f(\bar{x})}{s^2} = \frac{\langle \hat{u} | H_f(\bar{x} + \alpha_2 \vec{s}) + H_f(\bar{x} - \alpha_3 \vec{s}) | \hat{u} \rangle}{2} \right) = \langle \hat{u} | H_f(\bar{x}) | \hat{u} \rangle \Rightarrow \boxed{\langle \vec{s} | H_f(\bar{x}) | \vec{s} \rangle \geq 0}$

### 2.2.1 Unconstrained Optimization Conditions

- $\boxed{f(\bar{x}) = \min f} \Leftrightarrow \left( \begin{array}{l} \vec{\nabla} f(\bar{x}) \cdot \vec{s} \geq 0, \vec{\nabla} f(\bar{x}) \cdot -\vec{s} \geq 0 \\ \Rightarrow \boxed{\vec{\nabla} f(\bar{x}) = 0} \end{array} \right), \quad \vec{u} = -\vec{\nabla} f(\bar{x}) \Rightarrow \boxed{\vec{\nabla} f(\bar{x}) = 0}, \quad \boxed{\begin{array}{l} \text{(for strict convexity)} \\ \langle \vec{s} | H_f(\bar{x}) | \vec{s} \rangle > 0 \end{array}} \right)$

Optimization  $f: \mathbb{R}^n \rightarrow \mathbb{R} \quad \boxed{\min f(\vec{x}) = y}$

$$\boxed{\mathcal{L}(\vec{x}) = f(\vec{x})}, \quad \boxed{\nabla \mathcal{L}(\bar{x}) = 0}, \quad \boxed{H_{\mathcal{L}} = \nabla_{xx} \mathcal{L}: \langle s | H_{\mathcal{L}}(\bar{x}) | s \rangle > 0} \Rightarrow \boxed{y = f(\bar{x})}$$

### 2.2.2 Constrained Optimization Conditions

- $\boxed{\vec{s} = \text{feasible direction}} \Leftrightarrow \left( \boxed{\vec{\nabla} f(\bar{x}) \cdot \vec{s} \geq 0}, \boxed{\langle \vec{s} | H_f(\bar{x}) | \vec{s} \rangle \geq 0} \right)$

Optimization  $\begin{array}{l} f: \mathbb{R}^n \rightarrow \mathbb{R} \\ g: \mathbb{R}^n \rightarrow \mathbb{R}^m \\ h: \mathbb{R}^n \rightarrow \mathbb{R}^p \end{array} \quad \min f(\vec{x}) = y \quad \text{w/} \quad \left( \begin{array}{l} \vec{g}(\vec{x}) = 0 \\ \vec{h}(\vec{x}) \leq 0 \end{array} \right) \quad \begin{array}{l} \text{active: } h_i(\bar{x}) = 0 \\ \text{inactive: } h_i(\bar{x}) < 0 \rightarrow \bar{\mu}_i = 0 \end{array} \quad \text{(see KKT)}$

$$\begin{aligned} \mathcal{L}(\bar{x}, \bar{\lambda}, \bar{\mu}) &= f(\bar{x}) + \bar{\lambda} \cdot \vec{g}(\bar{x}) + \bar{\mu} \cdot \vec{h}(\bar{x}) \\ &= f + \sum_i^m \lambda_i g_i + \sum_i^p \cancel{\mu_i h_i} \quad \text{(KKT) if } \bar{x} = \bar{x} \end{aligned}, \quad \nabla \mathcal{L}(\bar{x}, \bar{\lambda}, \bar{\mu}) = \begin{pmatrix} \nabla_x \mathcal{L} = 0 \\ \nabla_{\lambda} \mathcal{L} = 0 \\ \nabla_{\mu} \mathcal{L} \leq 0 \end{pmatrix} = \begin{pmatrix} \nabla f(\bar{x}) + J_g^T(\bar{x}) \bar{\lambda} + J_h^T(\bar{x}) \bar{\mu} \\ \vec{g}(\bar{x}) \\ \vec{h}(\bar{x}) \end{pmatrix}$$

$$H_{\mathcal{L}}(\bar{x}, \bar{\lambda}, \bar{\mu}) = \begin{pmatrix} \nabla_{xx} \mathcal{L} & \nabla_{x\lambda} \mathcal{L} & \nabla_{x\mu} \mathcal{L} \\ \nabla_{\lambda x} \mathcal{L} & \nabla_{\lambda\lambda} \mathcal{L} & \nabla_{\lambda\mu} \mathcal{L} \\ \nabla_{\mu x} \mathcal{L} & \nabla_{\mu\lambda} \mathcal{L} & \nabla_{\mu\mu} \mathcal{L} \end{pmatrix} = \begin{pmatrix} \nabla_{xx} \mathcal{L} & J_g^T & J_h^T \\ J_g & 0 & 0 \\ J_h & 0 & 0 \end{pmatrix}, \quad \boxed{\nabla_{xx} \mathcal{L}(\bar{x}, \bar{\lambda}, \bar{\mu}) = H_f + \sum_i^m \bar{\lambda}_i H_{g_i} + \sum_i^{\text{act} \leq p} \bar{\mu}_i H_{h_i}}$$

(can't be pos. def.)

- Assume  $m \leq n$  (not overdetermined)

- $y = f(\bar{x}) : \nabla \mathcal{L}(\bar{x}, \bar{\lambda}, \bar{\mu}) \dots$  ,  $p = 0 : Z^T (\nabla_{xx} \mathcal{L}) Z > 0$  col. of  $Z$  = basis of  $\text{null}(J_g)$

- Assume  $h_i$  don't contradict each other? Assume full rank( $J_{h_{\text{act}}}$ )

- $y = f(\bar{x}) : \nabla \mathcal{L}(\bar{x}, \bar{\lambda}, \bar{\mu}) \dots$  ,  $p > 0$ , Karush-Kuhn-Tucker (KKT) :  $\bar{\mu}_i \geq 0$ ,  $\bar{\mu}_i h_i(\bar{x}) = 0$  (2nd deriv. cond. not given)

## 2.3 Unconstrained One Dimension/Independent Variable

[Interval] Golden-Section Search (if Unimodal):  $\tau^2 = 1 - \tau = .382$  ,  $r = 1$  ,  $C = \tau$

$$[a < x_1 < x_2 < b] : \begin{cases} f(x_1) > f(x_2) \rightarrow [x_1 < x_2 < x_1 + \tau(b - x_1) < b] \\ f(x_1) \leq f(x_2) \rightarrow [a < a + (1 - \tau)(x_2 - a) < x_1 < x_2] \end{cases}$$

Newton's Method:  $f(\bar{x}) = f(x + h) \approx f(x) + f'(x)h + \frac{1}{2}f''(x)h^2 = g(h)$

$$g(\frac{-b}{2a}) = \min g \text{ (or max)} \Rightarrow x_{k+1} = x_k + h_k = x_k - \frac{b}{2a} = x_k - \frac{f'(x)}{f''(x)} , r = 2$$

Successive Linear Interpolation [Secant Method]: Not useful, since line usually has no minimum

Successive Parabolic Interpolation: Use 3 pts to approx. a parabola w/  $r = 1.324$  (not guaranteed)

## 2.4 Unconstrained $m$ -Dimensions/Independent Variables

Steepest [Gradient] Descent/Line Search (go down  $-\nabla f(\vec{x}_k)$ ):

$$\phi(\alpha) = f(\vec{x} - \alpha \vec{\nabla} f(\vec{x})) , \phi(\alpha_k) = \min \phi \Rightarrow \vec{x}_{k+1} = \vec{x}_k - \alpha_k \vec{\nabla} f(\vec{x}_k) , r = 1 , C_{\text{varies}}$$

- $\vec{\nabla} f(\vec{x}_k) \cdot \vec{\nabla} f(\vec{x}_{k+1}) = 0 \Rightarrow$  Path will zig-zag to the min. (not too efficient)

Newton's Method:  $f(\bar{x}) = f(\vec{x} + \vec{h}) \approx f(\vec{x}) + \vec{\nabla} f(\vec{x}) \cdot \vec{h} + \frac{1}{2} \langle \vec{h} | H_f(\vec{x}) | \vec{h} \rangle$

$$H_f(\vec{x}_k) \vec{h}_k = -\vec{\nabla} f(\vec{x}_k) \Rightarrow \vec{x}_{k+1} = \vec{x}_k + \vec{h}_k , r = 2$$

BFGS [Secant Updating] Method:  $B_k \vec{h}_k = -\vec{\nabla} f(\vec{x}_k) , \vec{y}_k = \vec{\nabla} f(x_{k+1}) - \vec{\nabla} f(x_k)$

$$\Rightarrow \vec{x}_{k+1} = \vec{x}_k + \vec{h}_k , B_{k+1} = B_k + \frac{|y_k\rangle\langle y_k|}{\langle y_k | h_k \rangle} - \frac{B_k |h_k\rangle\langle h_k| B_k}{\langle h_k | B_k | H_k \rangle} \text{ (cost is } \mathcal{O}(n^3))$$

- Preserves symmetry and pos. def.
- $B_k$  factorization is updated to factorization of  $B_{k+1}$  at cost  $\mathcal{O}(n^2)$  instead of directly from the above eq.
- Lower cost of iter. offsets the larger number of iter. compared to Newton's Method with derivatives

Conjugate Gradient [Line Search] :

$$\boxed{\vec{h}_{k+1} = \vec{\nabla} f(\vec{x}_{k+1}) - \frac{\vec{\nabla} f(\vec{x}_{k+1}) \cdot \vec{\nabla} f(\vec{x}_{k+1})}{\vec{\nabla} f(\vec{x}_k) \cdot \vec{\nabla} f(\vec{x}_k)} \vec{h}_k} \quad (\text{Fletcher and Reeves}) \Rightarrow \boxed{\vec{x}_{k+1} = \vec{x}_k - \alpha_k \vec{h}_k}$$

- Seq. of conj. (where  $(a, b) = \langle a | H_f | b \rangle$ ) search directions implicitly accumulates info. about  $H_f$ .
- Better for nonlin. to use 
$$\boxed{\vec{h}_{k+1} = \vec{\nabla} f(\vec{x}_{k+1}) - \frac{\vec{\nabla} f(\vec{x}_{k+1}) \cdot \vec{\nabla} f(\vec{x}_{k+1}) - \vec{\nabla} f(\vec{x}_k) \cdot \vec{\nabla} f(\vec{x}_{k+1})}{\vec{\nabla} f(\vec{x}_k) \cdot \vec{\nabla} f(\vec{x}_k)} \vec{h}_k}$$
 (Polak and Ribiere)
- Restart algorithm after  $n$  iter. using last point as the new initial; a quadratic func. finishes after at most  $n$  iter.

### 2.4.1 Nonlinear Least Squares, $\{ \min \|\vec{r}(\vec{x})\|^2 : \vec{f}(\vec{a}, \vec{x}) + \vec{r}(\vec{x}) = \vec{b} \}$

Linear Least Squares	Nonlinear Least Squares
$\begin{pmatrix} \vdots \\ -\vec{a}_i \\ \vdots \end{pmatrix} \begin{pmatrix}   \\ \vec{x} \\   \end{pmatrix} + \begin{pmatrix}   \\ \vec{r} \\   \end{pmatrix} = \begin{pmatrix}   \\ \vec{b} \\   \end{pmatrix} \Rightarrow \begin{pmatrix}   \\ \vec{f}(\vec{a}, \vec{x})_i \\   \end{pmatrix} + \begin{pmatrix}   \\ \vec{r} \\   \end{pmatrix} = \begin{pmatrix}   \\ \vec{b} \\   \end{pmatrix}$	

$\boxed{\phi(\vec{x}) \equiv \frac{1}{2} \vec{r} \cdot \vec{r}}, \quad \boxed{-\vec{\nabla} \phi(\vec{x}) = -J_r^T \vec{r}}$	<p>Newton's Method</p> $\boxed{H_\phi(\vec{x}_k) \vec{h}_k = -\vec{\nabla} \phi(\vec{x}_k)} \Rightarrow \boxed{\vec{x}_{k+1} = \vec{x}_k + \vec{h}_k}$ <p>(usually expensive to compute)</p>	
$\boxed{H_\phi(\vec{x}) = J_r^T J_r + \sum_i H_{r_i} \vec{r}_i}$	:	

Gauss-Newton Method: If  $\vec{r}$  is small  $\Rightarrow H_\phi \approx J_r^T J_r \Rightarrow \boxed{J_r^T (J_r \vec{h}_k) = -J_r^T \vec{r}(\vec{x}_k)}$  System of Normal Equations

Levenberg-Marquardt Method (Gauss-Newton + Line Search):

$\boxed{(J_r^T J_r + \mu_k I) \vec{h}_k = -J_r^T \vec{r}(\vec{x}_k) \Rightarrow \vec{x}_{k+1} = \vec{x} + \vec{h}_k}$ $\Rightarrow \boxed{\begin{pmatrix} J_r^T(\vec{x}) & \sqrt{\mu_k} I \end{pmatrix} \begin{pmatrix} J_r(\vec{x}) \\ \sqrt{\mu_k} I \end{pmatrix} \vec{h}_k = \begin{pmatrix} J_r^T(\vec{x}) & \sqrt{\mu_k} I \end{pmatrix} \begin{pmatrix} -\vec{r}(\vec{x}_k) \\ 0 \end{pmatrix}}$	<p><u>Regularization</u></p> <ul style="list-style-type: none"> <li>• Replacing <math>H_{r_i} \vec{r}_i</math> terms with a scalar mult. of <math>I</math>.</li> <li>• Shifting the Gauss-Newton Hessian to make it pos. def (or boosting its rank).</li> </ul>
---	---

## 2.5 Constrained $m$ -Dimensions/Independent Variables

<p>Newton's Method</p> $\boxed{H_{\mathcal{L}} \vec{h}_k = -\vec{\nabla} \mathcal{L}}$	<p>KKT Matrix (Eq. Constr)</p> $\begin{pmatrix} \nabla_{xx} \mathcal{L} & J_g^T \\ J_g & 0 \end{pmatrix} \begin{pmatrix} \vec{s}_k \\ \vec{\delta}_k \end{pmatrix} = - \begin{pmatrix} \nabla f(\vec{x}) + J_g^T(\vec{x}) \bar{\lambda} \\ \vec{g}(\vec{x}) \end{pmatrix} \Rightarrow$ $\boxed{\begin{pmatrix} B & J^T \\ J & 0 \end{pmatrix} \begin{pmatrix} \vec{s} \\ \delta \end{pmatrix} = - \begin{pmatrix} w \\ g \end{pmatrix}}$	<p>[Sequential] Quadratic Programming (SQP) Problem</p> $\min_s \left( \vec{s}_k \cdot \vec{\nabla}_x \mathcal{L} + \frac{1}{2} \langle \vec{s}_k   \vec{\nabla}_{xx} \mathcal{L}   \vec{s}_k \rangle \right)$ <p>s.t. <math>J_g(\vec{x}_k) \vec{s}_k + \vec{g}(\vec{x}_k) = 0</math></p>
--	--	---

Direct Solution: KKT Matrix is sym. and sparse  $\rightarrow$  solve for  $\vec{h}_k$  using sym. indef. factorization w/ some pivoting

(Column-Space)

Range-Space Method:  $Bs = -w - J^T \delta$  ,  $Js = -g \rightarrow JB^{-1}(-w - J^T \delta) = -g$   
 $\rightarrow (JB^{-1}J^T)\delta = g - JB^{-1}w$

- Solve for  $\delta$ , then for  $s$ .
- Forming  $(JB^{-1}J^T)_{m \times m}$  leads to issues similar to forming  $A^T A$  (loss of info. and degrades conditioning).
- $B$  must be nonsingular and  $J$  full rank.
- Useful if  $m$  is small.

Null-Space Method:  $J^T = (Q_{\parallel} \ Q_{\perp}) \begin{pmatrix} R \\ 0 \end{pmatrix} \quad (Q_{\parallel} \in \mathbb{R}^{n \times m}) \Rightarrow \begin{cases} JQ_{\parallel} = R^T \\ JQ_{\perp} = 0 \end{cases}$

Find  $u_{\parallel}$  :  $Js \equiv (JQ_{\parallel}u_{\parallel} + \cancel{JQ_{\perp}u_{\perp}}) = \boxed{R^T u_{\parallel} = -g}$

Find  $u_{\perp}$  :  $Q_{\perp}^T(Bs + J^T \delta = -w) \rightarrow (Q_{\perp}^T BQ_{\parallel})u_{\parallel} + (Q_{\perp}^T BQ_{\perp})u_{\perp} = -Q_{\perp}^T w - \cancel{(JQ_{\perp})^T} \delta$   
 $\boxed{(Q_{\perp}^T BQ_{\perp})u_{\perp} = -Q_{\perp}^T w - (Q_{\perp}^T BQ_{\parallel})u_{\parallel}}$

Find  $\delta$  :  $Q_{\parallel}^T(J^T \delta = -w - Bs) \rightarrow \boxed{R\delta = -Q_{\parallel}^T w - Q_{\parallel}^T B(Q_{\parallel}u_{\parallel} - Q_{\perp}u_{\perp})}$

- Near a min.,  $(Q_{\perp}^T BQ_{\perp})$  can be Cholesky factored.
- Avoids issues with loss of info. and degraded conditioning.
- $J$  must be full rank and  $R$  nonsingular.
- Useful if  $m$  is large, so  $n - m$  is small.

Decent Initial  $\vec{\lambda}_0$  Guess Given an  $\vec{x}_0$ :  $J_g^T(\vec{x}_0)\vec{\lambda}_0 + \vec{r} = -\vec{\nabla} f(\vec{x}_0) \quad (\text{Linear Least Sq.})$

Penalty Func. Method

$$\lim_{\rho \rightarrow \infty} \vec{x}_{\rho} = \bar{x}$$

(“Under approp. conds.”)

One Simple Function  
(Ill-conditioned  $\rho \gg 1$ ) :  $\min_{\vec{x}} \phi_{\rho}(\vec{x}) = f(\vec{x}) + \frac{1}{2}\rho \|g(\vec{x})\|^2$

Augmented Lagrangian  
(Less Ill-conditioned) :  $\min_{\vec{x}} \mathcal{L}_{\rho}(\vec{x}) = f(\vec{x}) + \vec{\lambda}_0 \cdot \vec{g}(\vec{x}) + \frac{1}{2}\rho \|g(\vec{x})\|^2$

Barrier Func. Method

$$\lim_{\rho \rightarrow 0} \vec{x}_{\rho} = \bar{x}$$

(“Under approp. conds.”)

Inverse :  $\min_{\vec{x}} \phi_{\rho}(\vec{x}) = f(\vec{x}) - \rho \sum_i^p \frac{1}{h_i(\vec{x})}$

Logarithmic :  $\min_{\vec{x}} \phi_{\rho}(\vec{x}) = f(\vec{x}) - \rho \sum_i^p \log(-h_i(\vec{x}))$

(For Ineq. Constr.)

- Along with line search and trust region (not explained), a merit func. - using perhaps a penalty func. - can be used to make an algorithm more robust.
- An active set strategy (not explained) can be used with an SQP method for ineq.-constr. problems.
- A penalty method penalizes points that violates constraints, but doesn't avoid them. Barrier methods do.