ECE276A: Sensing & Estimation in Robotics
Lecture 3: Unconstrained Optimization

Instructor:

Nikolay Atanasov: natanasov@ucsd.edu

Teaching Assistants:

Qiaojun Feng: qif007@eng.ucsd.edu

Arash Asgharivaskasi: aasghari@eng.ucsd.edu

Thai Duong: tduong@eng.ucsd.edu

Yiran Xu: y5xu@eng.ucsd.edu

UC San Diego

JACOBS SCHOOL OF ENGINEERING

Electrical and Computer Engineering

Vectors

- A **vector** $\mathbf{x} \in \mathbb{R}^d$ with d dimensions is a collection of scalars $x_i \in \mathbb{R}$ for $i = 1, \ldots, d$ organized is a column:

$$\mathbf{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_d \end{bmatrix} \qquad \mathbf{x}^\top = \begin{bmatrix} x_1 & \cdots & x_d \end{bmatrix}$$

- A **norm** on a vector space V over a field F is a function $\|\cdot\| : V \to \mathbb{R}$ such that for all $a \in F$ and all $\mathbf{x}, \mathbf{y} \in V$:
  - $\|a\mathbf{x}\| = |a|\|\mathbf{x}\|$ (absolute homogeneity)
  - $\|\mathbf{x} + \mathbf{y}\| \leq \|\mathbf{x}\| + \|\mathbf{y}\|$ (triangle inequality)
  - $\|\mathbf{x}\| \geq 0$ (non-negativity)
  - $\|\mathbf{x}\| = 0$ iff $\mathbf{x} = 0$ (definiteness)
- The **Euclidean norm** of a vector $\mathbf{x} \in \mathbb{R}^d$ is $\|\mathbf{x}\|_2 := \sqrt{\mathbf{x}^\top \mathbf{x}}$ and satisfies:
  - $\max_{1 \leq i \leq d} |x_i| \leq \|\mathbf{x}\|_2 \leq \sqrt{d} \max_{1 \leq i \leq d} |x_i|$
  - $|\mathbf{x}^\top \mathbf{y}| \leq \|\mathbf{x}\|_2 \|\mathbf{y}\|_2$ (Cauchy-Schwarz Inequality)

Matrices

- A **matrix** $A \in \mathbb{R}^{m \times n}$ is a rectangular array of scalars $A_{ij} \in \mathbb{R}$ for $i = 1, \ldots, m$ and $j = 1, \ldots, n$
- The entries of the **transpose** $A^\top \in \mathbb{R}^{n \times m}$ of a matrix $A \in \mathbb{R}^{m \times n}$ are $A^\top_{ij} = A_{ji}$. The transpose satisfies: $(AB)^\top = B^\top A^\top$
- The **trace** of a matrix $A \in \mathbb{R}^{n \times n}$ is the sum of its diagonal entries:

$$\text{tr}(A) := \sum_{i=1}^{n} A_{ii} \qquad\qquad \text{tr}(ABC) = \text{tr}(BCA) = \text{tr}(CAB)$$

- The **determinant** of a matrix $A \in \mathbb{R}^{n \times n}$ is:

$$\det(A) := \sum_{j=1}^{n} A_{ij} \mathbf{cof}_{ij}(A) \qquad\qquad \det(AB) = \det(A)\det(B) = \det(BA)$$

  where $\mathbf{cof}_{ij}(A)$ is the **cofactor** of the entry A_{ij} and is equal to $(-1)^{i+j}$ times the determinant of the $(n-1) \times (n-1)$ submatrix that results when the i^{th}-row and j^{th}-col of A are removed. This recursive definition uses the fact that the determinant of a scalar is the scalar itself.

Matrix Inverse

- The **adjugate** is the transpose of the cofactor matrix:

$$\textbf{adj}(A) := \textbf{cof}(A)^\top$$

- The **inverse** A^{-1} of A exists iff $\det(A) \neq 0$ and satisfies:

$$A^{-1} = \frac{\textbf{adj}(A)}{\det(A)} \qquad\qquad (AB)^{-1} = B^{-1}A^{-1}$$

- If $A \in \mathbb{R}^{n \times n}$ and $\mathbf{q} \in \mathbb{C}^n$ is a nonzero vector such that:

$$A\mathbf{q} = \lambda \mathbf{q}$$

  then $\mathbf{q}$ is an **eigenvector** corresponding to the **eigenvalue** $\lambda \in \mathbb{C}$.

- A real matrix can have complex eigenvalues and eigenvectors, which appear in conjugate pairs. The n eigenvalues of $A \in \mathbb{R}^{n \times n}$ are precisely the n roots of the **characteristic polynomial** of A:

$$p(\lambda) := \det(\lambda I - A)$$

Positive Semidefinite Matrices

- The roots of a polynomial are continuous functions of its coefficients and hence the eigenvalues of a matrix are continuous functions of its entries.

$$\text{tr}(A) := \sum_{i=1}^{n} \lambda_i \qquad \det(A) := \prod_{i=1}^{n} \lambda_i$$

- The product $\mathbf{x}^\top Q\mathbf{x}$ for $Q \in \mathbb{R}^{n\times n}$ and $\mathbf{x} \in \mathbb{R}^n$ is called a **quadratic form** and Q can be assumed **symmetric**, $Q = Q^\top$, because:

$$\frac{1}{2}\mathbf{x}^\top(Q + Q^\top)\mathbf{x} = \mathbf{x}^\top Q\mathbf{x}, \qquad \forall \mathbf{x} \in \mathbb{R}^n$$

- A symmetric matrix $Q \in \mathbb{R}^{n\times n}$ is **positive semidefinite** if $\mathbf{x}^\top Q\mathbf{x} \geq 0$ for all $\mathbf{x} \in \mathbb{R}^n$.

- A symmetric matrix $Q \in \mathbb{R}^{n\times n}$ is **positive definite** if it is positive semidefinite and if $\mathbf{x}^\top Q\mathbf{x} = 0$ implies $\mathbf{x} = 0$

- All eigenvalues of a symmetric matrix are **real**. Hence, all eigenvalues of a positive semidefinite matrix are non-negative and all eigenvalues of a positive definite matrix are positive.

Schur Complement

- The **Schur complement** of block D of $M = \begin{bmatrix} A & B \\ C & D \end{bmatrix}$ is $S_D = A - BD^{-1}C$
- Let $M = \begin{bmatrix} A & B \\ B^\top & D \end{bmatrix}$ be symmetric. Then:
  - $M \succ 0 \Leftrightarrow A \succ 0, S_A = D - B^\top A^{-1} B \succ 0$
  - $M \succ 0 \Leftrightarrow D \succ 0, S_D = A - BD^{-1} B^\top \succ 0$
  - $M \succeq 0 \Leftrightarrow A \succeq 0, S_A \succeq 0, (I - AA^g)B = 0$, where A^g is the generalized inverse of A
  - $M \succeq 0 \Leftrightarrow D \succeq 0, S_D \succeq 0, (I - DD^g)B^\top = 0$, where D^g is the generalized inverse of D

Matrix Inversion Lemma

- **Square completion**:

$$\frac{1}{2}x^\top Ax + b^\top x + c = \frac{1}{2}\left(x + A^{-1}b\right)^\top A\left(x + A^{-1}b\right) + c - \frac{1}{2}b^\top A^{-1}b$$

- **Woodbury matrix identity**:

$$(A + BDC)^{-1} = A^{-1} - A^{-1}B\left(CA^{-1}B + D^{-1}\right)^{-1}CA^{-1}$$

- **Block matrix inversion**:

$$\begin{aligned}
\begin{bmatrix} A & B \\ C & D \end{bmatrix}^{-1} &= \begin{bmatrix} I & 0 \\ D^{-1}C & I \end{bmatrix}^{-1} \begin{bmatrix} A - BD^{-1}C & 0 \\ 0 & D \end{bmatrix}^{-1} \begin{bmatrix} I & BD^{-1} \\ 0 & I \end{bmatrix}^{-1} \\
&= \begin{bmatrix} I & 0 \\ -D^{-1}C & I \end{bmatrix} \begin{bmatrix} \left(A - BD^{-1}C\right)^{-1} & 0 \\ 0 & D^{-1} \end{bmatrix} \begin{bmatrix} I & -BD^{-1} \\ 0 & I \end{bmatrix} \\
&= \begin{bmatrix} \left(A - BD^{-1}C\right)^{-1} & -\left(A - BD^{-1}C\right)^{-1}BD^{-1} \\ -D^{-1}C\left(A - BD^{-1}C\right)^{-1} & D^{-1} + D^{-1}C\left(A - BD^{-1}C\right)^{-1}BD^{-1} \end{bmatrix}
\end{aligned}$$

Derivatives (numerator layout)

- **Derivatives by scalar**

$$\frac{d\mathbf{y}}{dx} = \begin{bmatrix} \frac{dy_1}{dx} \\ \vdots \\ \frac{dy_m}{dx} \end{bmatrix} \in \mathbb{R}^{m\times 1} \qquad \frac{dY}{dx} = \begin{bmatrix} \frac{dY_{11}}{dx} & \cdots & \frac{dY_{1n}}{dx} \\ \vdots & \ddots & \vdots \\ \frac{d\mathbf{Y}_{m1}}{dx} & \cdots & \frac{d\mathbf{Y}_{mn}}{dx} \end{bmatrix} \in \mathbb{R}^{m\times n}$$

- **Derivatives by vector**

$$\frac{dy}{d\mathbf{x}} = \underbrace{\begin{bmatrix} \frac{dy}{dx_1} & \cdots & \frac{dy}{dx_p} \end{bmatrix}}_{[\nabla_{\mathbf{x}} y]^\top \text{ (gradient transpose)}} \in \mathbb{R}^{1\times p} \qquad \frac{d\mathbf{y}}{d\mathbf{x}} = \underbrace{\begin{bmatrix} \frac{dy_1}{dx_1} & \cdots & \frac{dy_1}{dx_p} \\ \vdots & \ddots & \vdots \\ \frac{dy_m}{dx_1} & \cdots & \frac{dy_m}{dx_p} \end{bmatrix}}_{\text{Jacobian}} \in \mathbb{R}^{m\times p}$$

- **Derivatives by matrix**

$$\frac{dy}{dX} = \begin{bmatrix} \frac{dy}{dX_{11}} & \cdots & \frac{dy}{dX_{p1}} \\ \vdots & \ddots & \vdots \\ \frac{dy}{dX_{1q}} & \cdots & \frac{dy}{dX_{pq}} \end{bmatrix} \in \mathbb{R}^{q\times p}$$

Matrix Derivatives Example

- $\frac{d}{dX_{ij}} X = \mathbf{e}_i \mathbf{e}_j^\top$
- $\frac{d}{d\mathbf{x}} A\mathbf{x} = A$
- $\frac{d}{d\mathbf{x}} \mathbf{x}^\top A\mathbf{x} = \mathbf{x}^\top (A + A^\top)$
- $\frac{d}{dx} M^{-1}(x) = -M^{-1}(x) \frac{dM(x)}{dx} M^{-1}(x)$
- $\frac{d}{dX} \operatorname{tr}(AX^{-1}B) = -X^{-1}BAX^{-1}$
- $\frac{d}{dX} \log\det X = X^{-1}$

Matrix Derivatives Example

- $\frac{d}{d\mathbf{x}}A\mathbf{x} = \begin{bmatrix} \frac{d}{dx_1}\sum_{j=1}^n A_{1j}x_j & \cdots & \frac{d}{dx_n}\sum_{j=1}^n A_{1j}x_j \\ \vdots & \ddots & \vdots \\ \frac{d}{dx_1}\sum_{j=1}^n A_{mj}x_j & \cdots & \frac{d}{dx_n}\sum_{j=1}^n A_{mj}x_j \end{bmatrix} = \begin{bmatrix} A_{11} & \cdots & A_{1n} \\ \vdots & \ddots & \vdots \\ A_{m1} & \cdots & A_{mn} \end{bmatrix}$
- $\frac{d}{d\mathbf{x}}\mathbf{x}^\top A\mathbf{x} = \mathbf{x}^\top A^\top \frac{d\mathbf{x}}{d\mathbf{x}} + \mathbf{x}^\top \frac{dA\mathbf{x}}{d\mathbf{x}} = \mathbf{x}^\top(A^\top + A)$
- $M(x)M^{-1}(x) = I \quad \Rightarrow \quad 0 = \left[\frac{d}{dx}M(x)\right]M^{-1}(x) + M(x)\left[\frac{d}{dx}M^{-1}(x)\right]$
- $$\begin{aligned} \frac{d}{dX_{ij}}\operatorname{tr}(AX^{-1}B) &= \operatorname{tr}(A\frac{d}{dX_{ij}}X^{-1}B) = -\operatorname{tr}(AX^{-1}\mathbf{e}_i\mathbf{e}_j^\top X^{-1}B) \\ &= -\mathbf{e}_j^\top X^{-1}BAX^{-1}\mathbf{e}_i = -\mathbf{e}_i^\top\left(X^{-1}BAX^{-1}\right)^\top\mathbf{e}_j \end{aligned}$$
- $$\begin{aligned} \frac{d}{dX_{ij}}\log\det X &= \frac{1}{\det(X)}\frac{d}{dX_{ij}}\sum_{k=1}^n X_{ik}\mathbf{cof}_{ik}(X) \\ &= \frac{1}{\det(X)}\mathbf{cof}_{ij}(X) = \frac{1}{\det(X)}\mathbf{adj}_{ji}(X) = \mathbf{e}_i^\top X^{-T}\mathbf{e}_j \end{aligned}$$

Unconstrained Optimization

- Many problems we encounter in this course, lead to an optimization problem of the form:

$$\min_{\mathbf{x}} f(\mathbf{x})$$

Descent Direction Theorem

Suppose f is differentiable at $\bar{\mathbf{x}}$. If $\exists\ \delta\mathbf{x}$ such that $\nabla f(\bar{\mathbf{x}})^\top \delta\mathbf{x} < 0$, then $\exists\ \epsilon > 0$ such that $f(\bar{\mathbf{x}} + \alpha\delta\mathbf{x}) < f(\bar{\mathbf{x}})$ for all $\alpha \in (0, \epsilon)$.

- The vector $\delta\mathbf{x}$ is called a **descent direction**
- The theorem states that if a descent direction exists at $\bar{\mathbf{x}}$, then it is possible to move to a new point that has a lower f value.
- **Steepest descent direction**: $\delta\mathbf{x} := -\frac{\nabla f(\bar{\mathbf{x}})}{\|\nabla f(\bar{\mathbf{x}})\|}$
- Based on this theorem, we can derive conditions for determining the optimality of $\bar{\mathbf{x}}$

Optimality Conditions

First-order Necessary Condition

Suppose f is differentiable at $\bar{\mathbf{x}}$. If $\bar{\mathbf{x}}$ is a local minimizer, then $\nabla J(\bar{\mathbf{x}}) = 0$.

Second-order Necessary Condition

Suppose f is twice-differentiable at $\bar{\mathbf{x}}$. If $\bar{\mathbf{x}}$ is a local minimizer, then $\nabla f(\bar{\mathbf{x}}) = 0$ and $\nabla^2 f(\bar{\mathbf{x}}) \succeq 0$.

Second-order Sufficient Condition

Suppose f is twice-differentiable at $\bar{\mathbf{x}}$. If $\nabla f(\bar{\mathbf{x}}) = 0$ and $\nabla^2 f(\bar{\mathbf{x}}) \succ 0$, then $\bar{\mathbf{x}}$ is a local minimizer.

Necessary and Sufficient Condition

Suppose f is differentiable at $\bar{\mathbf{x}}$. If f is **convex**, then $\bar{\mathbf{x}}$ is a global minimizer **if and only if** $\nabla f(\bar{\mathbf{x}}) = 0$.

Descent Optimization Methods

- Convex unconstrained optimization: just need to solve the equation $\nabla f(\mathbf{x}) = 0$ to determine a global minimizer $\mathbf{x}^*$
- Even if f is not convex, we can obtain a critical point by solving $\nabla f(\mathbf{x}) = 0$
- However, $\nabla f(\mathbf{x}) = 0$ might not be easy to solve explicitly
- **Descent methods**: iterative methods for unconstrained optimization. Given an initial guess $\mathbf{x}^{(k)}$, take a step of size $\alpha^{(k)} > 0$ along a certain direction $\delta\mathbf{x}^{(k)}$:

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha^{(k)}\delta\mathbf{x}^{(k)}$$

- Different methods differ in the way $\delta\mathbf{x}^{(k)}$ and $\alpha^{(k)}$ are chosen but
  - $\delta\mathbf{x}^{(k)}$ should be a descent direction: $\nabla f(\mathbf{x}^{(k)})^\top\delta\mathbf{x}^{(k)} < 0$ for all $\mathbf{x}^{(k)} \neq \mathbf{x}^*$
  - $\alpha^{(k)}$ needs to ensure sufficient decrease in f to guarantee convergence:

$$\alpha^{(k),*} \in \underset{\alpha>0}{\arg\min}\, f(\mathbf{x}^{(k)} + \alpha\delta\mathbf{x}^{(k)})$$

  Usually $\alpha^{(k)}$ is obtained via inexact **line search** methods

Gradient Descent (First-Order Method)

- **Idea**: $-\nabla f(\mathbf{x}^{(k)})$ points in the direction of steepest local descent
- **Gradient descent**: let $\delta \mathbf{x}^{(k)} := -\nabla f(\mathbf{x}^{(k)})$ and iterate:

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - \alpha^{(k)} \nabla f(\mathbf{x}^{(k)})$$

- A good choice for $\alpha^{(k)}$ is $\frac{1}{L}$, where $L > 0$ is the Lipschitz constant of $\nabla f(\mathbf{x})$:

$$\|\nabla f(\mathbf{x}) - \nabla f(\mathbf{x}')\| \leq L \|\mathbf{x} - \mathbf{x}'\| \qquad \forall \mathbf{x}, \mathbf{x}' \in \mathbf{dom}(f)$$

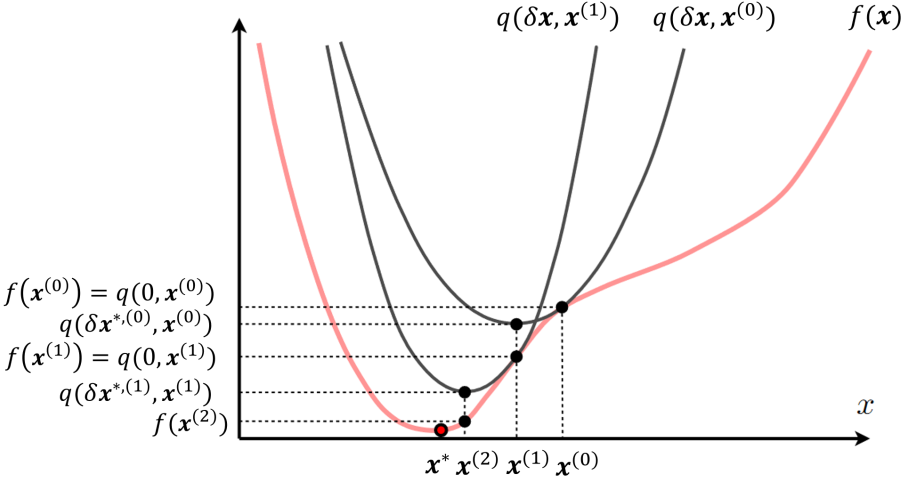
Newton's Method (Second-Order Method)

- **Newton's method**: iteratively approximates f by a quadratic function
- Since $\delta\mathbf{x}$ is a 'small' change to the initial guess $\mathbf{x}^{(k)}$, we can approximate f using a Taylor-series expansion:

$$f(\mathbf{x}^{(k)}+\delta\mathbf{x}) \approx f(\mathbf{x}^{(k)}) + \underbrace{\left(\frac{\partial f(\mathbf{x})}{\partial \mathbf{x}}\bigg|_{\mathbf{x}=\mathbf{x}^{(k)}}\right)}_{\text{Gradient Transpose}} \delta\mathbf{x} + \frac{1}{2}\delta\mathbf{x}^\top \underbrace{\left(\frac{\partial^2 f(\mathbf{x})}{\partial \mathbf{x}\partial \mathbf{x}^\top}\bigg|_{\mathbf{x}=\mathbf{x}^{(k)}}\right)}_{\text{Hessian}} \delta\mathbf{x}$$

- The symmetric Hessian matrix $\nabla^2 f(\mathbf{x}^{(k)})$ needs to be positive-definite for this method to work.

Newton's Method (Second-Order Method)



Newton's Method (Second-Order Method)

- Find $\delta\mathbf{x}$ that minimizes the quadratic approximation to $f(\mathbf{x}^{(k)} + \delta\mathbf{x})$
- Since this is an unconstrained optimization problem, $\delta\mathbf{x}^*$ can be determined by setting the derivative with respect to $\delta\mathbf{x}$ to zero:

$$\frac{\partial f(\mathbf{x}^{(k)} + \delta\mathbf{x})}{\partial \delta\mathbf{x}} \approx \left(\left.\frac{\partial f(\mathbf{x})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}^{(k)}} \right) + \delta\mathbf{x}^\top \left(\left.\frac{\partial^2 f(\mathbf{x})}{\partial \mathbf{x}\partial \mathbf{x}^\top}\right|_{\mathbf{x}=\mathbf{x}^{(k)}} \right)$$

$$\Rightarrow \quad \left(\left.\frac{\partial^2 f(\mathbf{x})}{\partial \mathbf{x}\partial \mathbf{x}^\top}\right|_{\mathbf{x}=\mathbf{x}^{(k)}} \right) \delta\mathbf{x} = - \left(\left.\frac{\partial f(\mathbf{x})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}^{(k)}} \right)^\top$$

- The above is a linear system of equations and can be solved when the Hessian is invertible, i.e., $\nabla^2 f(\mathbf{x}^{(k)}) \succ 0$:

$$\delta\mathbf{x}^* = - \left[\nabla^2 f(\mathbf{x}^{(k)})\right]^{-1} \nabla f(\mathbf{x}^{(k)})$$

- **Newton's method**:

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - \alpha^{(k)} \left[\nabla^2 f(\mathbf{x}^{(k)})\right]^{-1} \nabla f(\mathbf{x}^{(k)})$$

Newton's Method (Comments)

- Newton's method, like any other descent method, converges only to a **local** minimum
- **Damped Newton phase**: when the iterates are "far away" from the optimal point, the function value is decreased sublinearly, i.e., the step sizes $\alpha^{(k)}$ are small
- **Quadratic convergence phase**: when the iterates are "sufficiently close" to the optimum, full Newton steps are taken, i.e. $\alpha^{(k)} = 1$, and the function value converges quadratically to the optimum
- A **disadvantage** of Newton's method is the need to form the Hessian, which can be numerically ill-conditioned or very computationally expensive in high dimensional problems

Gauss-Newton's Method

- **Gauss-Newton** is an approximation to Newton's method that avoids computing the Hessian. It is applicable when the objective function has the following quadratic form:

$$f(\mathbf{x}) = \frac{1}{2}\mathbf{e}(\mathbf{x})^\top \mathbf{e}(\mathbf{x}) \qquad \mathbf{e}(\mathbf{x}) \in \mathbb{R}^m$$

- The Jacobian and Hessian matrices are:

$$\text{Jacobian:} \qquad \left.\frac{\partial f(\mathbf{x})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}^{(k)}} = \mathbf{e}(\mathbf{x}^{(k)})^\top \left(\left.\frac{\partial \mathbf{e}(\mathbf{x})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}^{(k)}} \right)$$

$$\text{Hessian:} \qquad \left.\frac{\partial^2 f(\mathbf{x})}{\partial \mathbf{x}\partial \mathbf{x}^\top}\right|_{\mathbf{x}=\mathbf{x}^{(k)}} = \left(\left.\frac{\partial \mathbf{e}(\mathbf{x})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}^{(k)}} \right)^\top \left(\left.\frac{\partial \mathbf{e}(\mathbf{x})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}^{(k)}} \right) + \sum_{i=1}^{m} e_i(\mathbf{x}^{(k)}) \left(\left.\frac{\partial^2 e_i(\mathbf{x})}{\partial \mathbf{x}\partial \mathbf{x}^\top}\right|_{\mathbf{x}=\mathbf{x}^{(k)}} \right)$$

Gauss-Newton's Method

- Near the minimum of f, the second term in the Hessian is small relative to the first and the Hessian can be approximated according to:

$$\left.\frac{\partial^2 f(\mathbf{x})}{\partial \mathbf{x} \partial \mathbf{x}^\top}\right|_{\mathbf{x}=\mathbf{x}^{(k)}} \approx \left(\left.\frac{\partial \mathbf{e}(\mathbf{x})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}^{(k)}} \right)^\top \left(\left.\frac{\partial \mathbf{e}(\mathbf{x})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}^{(k)}} \right)$$

- The above does not involve any second derivatives and leads to the system:

$$\left(\left.\frac{\partial \mathbf{e}(\mathbf{x})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}^{(k)}} \right)^\top \left(\left.\frac{\partial \mathbf{e}(\mathbf{x})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}^{(k)}} \right) \delta\mathbf{x} = - \left(\left.\frac{\partial \mathbf{e}(\mathbf{x})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}^{(k)}} \right)^\top \mathbf{e}(\mathbf{x}^{(k)})$$

- **Gauss-Newton's method**:

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha^{(k)} \delta\mathbf{x}$$

Gauss-Newton's Method (Alternative Derivation)

- Another way to think about the Gauss-Newton method is to start with a Taylor expansion of $\mathbf{e}(\mathbf{x})$ instead of $f(\mathbf{x})$:

$$\mathbf{e}(\mathbf{x}^{(k)} + \delta\mathbf{x}) \approx \mathbf{e}(\mathbf{x}^{(k)}) + \left(\left.\frac{\partial \mathbf{e}(\mathbf{x})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}^{(k)}} \right) \delta\mathbf{x}$$

- Substituting into f leads to:

$$f(\mathbf{x}^{(k)} + \delta\mathbf{x}) \approx \frac{1}{2} \left(\mathbf{e}(\mathbf{x}^{(k)}) + \left(\left.\frac{\partial \mathbf{e}(\mathbf{x})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}^{(k)}} \right) \delta\mathbf{x} \right)^\top \left(\mathbf{e}(\mathbf{x}^{(k)}) + \left(\left.\frac{\partial \mathbf{e}(\mathbf{x})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}^{(k)}} \right) \delta\mathbf{x} \right)$$

- Minimizing this with respect to $\delta\mathbf{x}$ leads to the same system as before:

$$\left(\left.\frac{\partial \mathbf{e}(\mathbf{x})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}^{(k)}} \right)^\top \left(\left.\frac{\partial \mathbf{e}(\mathbf{x})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}^{(k)}} \right) \delta\mathbf{x} = -\left(\left.\frac{\partial \mathbf{e}(\mathbf{x})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}^{(k)}} \right)^\top \mathbf{e}(\mathbf{x}^{(k)})$$

Levenberg-Marquardt's Method

- The **Levenberg-Marquardt** modification to the Gauss-Newton method uses a positive diagonal matrix D to condition the Hessian approximation:

$$\left(\left(\left.\frac{\partial \mathbf{e}(\mathbf{x})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}^{(k)}} \right)^{\top} \left(\left.\frac{\partial \mathbf{e}(\mathbf{x})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}^{(k)}} \right) + \lambda D \right) \delta\mathbf{x} = - \left(\left.\frac{\partial \mathbf{e}(\mathbf{x})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}^{(k)}} \right)^{\top} \mathbf{e}(\mathbf{x}^{(k)})$$

- When $\lambda \geq 0$ is large, the descent vector $\delta\mathbf{x}$ corresponds to a very small step in the direction of steepest descent. This helps when the Hessian approximation is poor or poorly conditioned by providing a meaningful direction.

Levenberg-Marquardt's Method (Summary)

- An iterative optimization approach for the unconstrained problem:

$$\min_{\mathbf{x}} f(\mathbf{x}) := \frac{1}{2}\sum_j \mathbf{e}_j(\mathbf{x})^\top \mathbf{e}_j(\mathbf{x}) \qquad \mathbf{e}_j(\mathbf{x}) \in \mathbb{R}^{m_j},\ \mathbf{x} \in \mathbb{R}^n$$

- Given an initial guess $\mathbf{x}^{(k)}$, determine a descent direction $\delta\mathbf{x}$ by solving:

$$\left(\sum_j J_j(\mathbf{x}^{(k)})^\top J_j(\mathbf{x}^{(k)}) + \lambda D\right)\delta\mathbf{x} = -\left(\sum_j J_j(\mathbf{x}^{(k)})^\top \mathbf{e}_j(\mathbf{x}^{(k)})\right)$$

  where $J_j(\mathbf{x}) := \frac{\partial \mathbf{e}_j(\mathbf{x})}{\partial \mathbf{x}} \in \mathbb{R}^{m_j \times n}$, $\lambda \geq 0$, $D \in \mathbb{R}^{n\times n}$ is a positive diagonal matrix, e.g., $D = \mathbf{diag}\left(\sum_j J_j(\mathbf{x}^{(k)})^\top J_j(\mathbf{x}^{(k)})\right)$

- Obtain an updated estimate according to:

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha^{(k)}\delta\mathbf{x}$$

Unconstrained Optimization Example

- Let $f(\mathbf{x}) := \frac{1}{2}\sum_{j=1}^{n} \|A_j\mathbf{x} + b_j\|_2^2$ for $\mathbf{x} \in \mathbb{R}^d$ and assume $\sum_{j=1}^{n} A_j^\top A_j \succ 0$
- Solve the unconstrained optimization problem $\min_{\mathbf{x}} f(\mathbf{x})$ using:
  - The necessary and sufficient optimality condition for convex function f
  - Gradient descent
  - Newton's method
  - Gauss-Newton's method
- We will need $\nabla f(\mathbf{x})$ and $\nabla^2 f(\mathbf{x})$:

$$\frac{df(\mathbf{x})}{d\mathbf{x}} = \frac{1}{2}\sum_{j=1}^{n} \frac{d}{d\mathbf{x}}\|A_j\mathbf{x} + b_j\|_2^2 = \sum_{j=1}^{n} (A_j\mathbf{x} + b_j)^\top A_j$$

$$\nabla f(\mathbf{x}) = \frac{df(\mathbf{x})}{d\mathbf{x}}^\top = \left(\sum_{j=1}^{n} A_j^\top A_j\right)\mathbf{x} + \left(\sum_{j=1}^{n} A_j^\top b_j\right)$$

$$\nabla^2 f(\mathbf{x}) = \frac{d}{d\mathbf{x}}\nabla f(\mathbf{x}) = \sum_{j=1}^{n} A_j^\top A_j \succ 0$$

Necessary and Sufficient Optimality Condition

- Solve $\nabla f(\mathbf{x}) = 0$ for $\mathbf{x}$:

$$0 = \nabla f(\mathbf{x}) = \left(\sum_{j=1}^{n} A_j^\top A_j\right)\mathbf{x} + \left(\sum_{j=1}^{n} A_j^\top b_j\right)$$

$$\mathbf{x} = -\left(\sum_{j=1}^{n} A_j^\top A_j\right)^{-1}\left(\sum_{j=1}^{n} A_j^\top b_j\right)$$

- The solution above is unique since we assumed that $\sum_{j=1}^{n} A_j^\top A_j \succ 0$

Gradient Descent

- Start with an initial guess $\mathbf{x}^{(0)} = \mathbf{0}$
- At iteration k, gradient descent uses the descent direction $\delta\mathbf{x}^{(k)} = -\nabla f(\mathbf{x}^{(k)})$
- Given arbitary $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^d$, determine the Lipschitz constant of $\nabla f(\mathbf{x})$:

$$\|\nabla f(\mathbf{x}_1) - \nabla f(\mathbf{x}_2)\| = \left\| \left(\sum_{j=1}^{n} A_j^\top A_j \right) (\mathbf{x}_1 - \mathbf{x}_2) \right\| \leq \underbrace{\left\| \sum_{j=1}^{n} A_j^\top A_j \right\|}_{L} \|\mathbf{x}_1 - \mathbf{x}_2\|$$

- Choose step size $\alpha^{(k)} = \frac{1}{L}$ and iterate:

$$\begin{aligned} \mathbf{x}^{(k+1)} &= \mathbf{x}^{(k)} + \alpha^{(k)} \delta\mathbf{x}^{(k)} \\ &= \mathbf{x}^{(k)} - \frac{1}{L} \left(\sum_{j=1}^{n} A_j^\top A_j \right) \mathbf{x}^{(k)} - \frac{1}{L} \left(\sum_{j=1}^{n} A_j^\top b_j \right) \end{aligned}$$

Newton's Method

- Start with an initial guess $\mathbf{x}^{(0)} = \mathbf{0}$
- At iteration k, Newton's method uses the descent direction:

$$\begin{aligned}\delta\mathbf{x}^{(k)} &= -\left[\nabla^2 f(\mathbf{x}^{(k)})\right]^{-1} \nabla f(\mathbf{x}^{(k)}) \\ &= -\mathbf{x}^{(k)} - \left(\sum_{j=1}^{n} A_j^\top A_j\right)^{-1} \left(\sum_{j=1}^{n} A_j^\top b_j\right)\end{aligned}$$

  and updates the solution estimate via:

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \delta\mathbf{x}^{(k)} = -\left(\sum_{j=1}^{n} A_j^\top A_j\right)^{-1} \left(\sum_{j=1}^{n} A_j^\top b_j\right)$$

- Note that for this problem, Newton's method converges in one iteration!

Gauss-Newton's Method

- $f(\mathbf{x})$ is of the form $\frac{1}{2}\sum_{j=1}^{n} \mathbf{e}_j(\mathbf{x})^\top \mathbf{e}_j(\mathbf{x})$ for $\mathbf{e}_j(\mathbf{x}) := A_j\mathbf{x} + b_j$
- The Jacobian of $\mathbf{e}_j(\mathbf{x})$ is $J_j(\mathbf{x}) = A_j$
- Start with an initial guess $\mathbf{x}^{(0)} = \mathbf{0}$
- At iteration k, Gauss-Newton's method uses the descent direction:

$$\begin{aligned}
\delta\mathbf{x}^{(k)} &= -\left(\sum_{j=1}^{n} J_j(\mathbf{x}^{(k)})^\top J_j(\mathbf{x}^{(k)})\right)^{-1} \left(\sum_{j=1}^{n} J_j(\mathbf{x}^{(k)})^\top \mathbf{e}_j(\mathbf{x}^{(k)})\right) \\
&= -\left(\sum_{j=1}^{n} A_j^\top A_j\right)^{-1} \left(\sum_{j=1}^{n} A_j^\top (A_j\mathbf{x}^{(k)} + b_j)\right) \\
&= -\mathbf{x}^{(k)} - \left(\sum_{j=1}^{n} A_j^\top A_j\right)^{-1} \left(\sum_{j=1}^{n} A_j^\top b_j\right)
\end{aligned}$$

- If $\alpha^{(k)} = 1$, in this problem, Gauss-Newton's method behaves exactly like Newton's method and coverges in one iteration!