数学工具手册

张淦淦

目录

Part I
微积分与最优化问题

1 微积分基础概念

2 矩阵代数与微积分

设 $x, \boldsymbol{x}, \boldsymbol{X}$ 分别为标量, 列向量, 矩阵,$f, \boldsymbol{f}, \boldsymbol{f}$ 分别为标量函数, 列向量函数, 矩阵函数.

$$x \in \mathcal{R}$$

$$\boldsymbol{x} = [x_1, \dots, x_m]^T \in \mathcal{R}^m$$

$$\boldsymbol{X} = [\boldsymbol{x}_1, \dots, \boldsymbol{x}_m] \in \mathcal{R}^{m\times n}$$

$$f(x), f(\boldsymbol{x}), f(\boldsymbol{X}) \in \mathcal{R}$$

$$\boldsymbol{f}(x), \boldsymbol{f}(\boldsymbol{x}), \boldsymbol{f}(\boldsymbol{X}) \in \mathcal{R}^p$$

$$\boldsymbol{F}(x), \boldsymbol{F}(\boldsymbol{x}), \boldsymbol{F}(\boldsymbol{X}) \in \mathcal{R}^{p\times q}$$

2.1 对 f 进行微分

x 对 $f(x)$ 微分, 即求导数.

$$\frac{df(x)}{dx}$$

$\boldsymbol{x}$ 对 $f(\boldsymbol{x})$ 微分, 行向量偏导算子 $D_{\boldsymbol{x}}$, 输出 $1\times m$ 的行向量, 列向量偏导算子 $\nabla_{\boldsymbol{x}}$, 输出 $m\times 1$ 的列向量.

$$D_{\boldsymbol{x}} f = \frac{\partial f}{\partial \boldsymbol{x}^T} = [\frac{\partial f}{\partial x_1}, \dots, \frac{\partial f}{\partial x_m}]$$
$$\nabla_{\boldsymbol{x}} f = \frac{\partial f}{\partial \boldsymbol{x}} = [\frac{\partial f}{\partial x_1}, \dots, \frac{\partial f}{\partial x_m}]^T$$

$\boldsymbol{X}$ 对 $f(\boldsymbol{X})$ 微分, 有两种定义, 分别为 $D_{\boldsymbol{X}} f$(Jacobian 矩阵) 和 $D_{vec(\boldsymbol{X})} f$(行偏导矩阵), 实际中,Jacobian 矩阵更有用.

$$D_{\boldsymbol{X}} f = \frac{\partial f}{\partial \boldsymbol{X}^T} = \begin{pmatrix} \frac{\partial f}{\partial x_{11}} & \cdots & \frac{\partial f}{\partial x_{m1}} \\ \vdots & \ddots & \vdots \\ \frac{\partial f}{\partial x_{1n}} & \cdots & \frac{\partial f}{\partial x_{mn}} \end{pmatrix} \in \mathcal{R}^{n\times m}$$

$$D_{vec(\boldsymbol{X})} f = \frac{\partial f(\boldsymbol{X})}{\partial vec^T(\boldsymbol{X})} = [\frac{\partial f}{\partial x_{11}}, \dots, \frac{\partial f}{\partial x_{m1}}, \dots, \frac{\partial f}{\partial x_{1n}}, \dots, \frac{\partial f}{\partial x_{mn}}]$$

$$D_{vec(\boldsymbol{X})} f = rvec(D_{\boldsymbol{X}} f) = (vec(D_{\boldsymbol{X}}^T f))^T$$

2.2 对 f 进行微分

x 对 $\boldsymbol{f}(x)$ 微分

$$\frac{\partial \boldsymbol{f}}{\partial x} = [\frac{\partial f_1}{\partial x}, \dots, \frac{\partial f_p}{\partial x}]^T$$

$\boldsymbol{x}$ 对 $\boldsymbol{f}(\boldsymbol{x})$ 微分, 已知 $\boldsymbol{x} \in \mathcal{R}^m, \boldsymbol{f} \in \mathcal{R}^p$,$m\times p$ 矩阵称为矩阵梯度, 记为 $G_{\boldsymbol{f}}$ 或 $\nabla_{\boldsymbol{f}}$; 对应的 $p\times m$ 矩阵称为 Jacobian 矩阵, 记为 $J_{\boldsymbol{f}}$

$$\begin{aligned} G_{\boldsymbol{f}} = \nabla_{\boldsymbol{f}} &= \frac{\partial \boldsymbol{f}^T}{\partial \boldsymbol{x}} = \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}} \\ &= \begin{pmatrix} \frac{\partial f_1}{\partial \boldsymbol{x}} & \cdots & \frac{\partial f_p}{\partial \boldsymbol{x}} \end{pmatrix} \\ &= \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_p}{\partial x_1} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_1}{\partial x_m} & \cdots & \frac{\partial f_p}{\partial x_m} \end{pmatrix} \in \mathcal{R}^{m\times p} \end{aligned}$$

$$J_{\boldsymbol{f}} = G_{\boldsymbol{f}}^T = \nabla_{\boldsymbol{f}}^T = \frac{\partial \boldsymbol{f}^T}{\partial \boldsymbol{x}^T} = \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}^T} \in \mathcal{R}^{p\times m}$$

$\boldsymbol{X}$ 对 $\boldsymbol{f}(\boldsymbol{X})$ 微分, 定义 $\nabla_{vec(\boldsymbol{X})}$ 为梯度算子,$\nabla_{vec(\boldsymbol{X})}\boldsymbol{f}$ 为梯度向量,$\nabla_{\boldsymbol{X}}\boldsymbol{f}$ 为梯度矩阵,$D_{\boldsymbol{x}} f$ 为 Jacobian 矩阵.

$$\nabla_{vec(\boldsymbol{X})} = \frac{\partial}{\partial vec(\boldsymbol{X})} = [\frac{\partial}{\partial x_{11}}, \dots, \frac{\partial}{\partial x_{m1}}, \dots, \frac{\partial}{\partial x_{1n}}, \dots, \frac{\partial}{\partial x_{mn}}]^T$$

$$\nabla_{vec(\boldsymbol{X})}\boldsymbol{f} = \frac{\partial \boldsymbol{f}}{\partial vec(\boldsymbol{X})} = [\frac{\partial \boldsymbol{f}}{\partial x_{11}}, \dots, \frac{\partial \boldsymbol{f}}{\partial x_{m1}}, \dots, \frac{\partial \boldsymbol{f}}{\partial x_{1n}}, \dots, \frac{\partial \boldsymbol{f}}{\partial x_{mn}}]^T$$

$$\nabla_{\boldsymbol{X}}\boldsymbol{f} = \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{X}} = \begin{pmatrix} \frac{\partial \boldsymbol{f}}{\partial x_{11}} & \cdots & \frac{\partial \boldsymbol{f}}{\partial x_{m1}} \\ \vdots & \ddots & \vdots \\ \frac{\partial \boldsymbol{f}}{\partial x_{1n}} & \cdots & \frac{\partial \boldsymbol{f}}{\partial x_{mn}} \end{pmatrix}$$

$$\nabla_{\boldsymbol{X}} f(\boldsymbol{X}) = D_{\boldsymbol{x}}^T f(\boldsymbol{X})$$

2.3 对 F 进行微分

x 对 $\boldsymbol{F}(x)$ 微分

$$\partial \boldsymbol{F} \in \mathcal{R}^{p\times q}$$

$$[\partial \boldsymbol{F}/\partial x]_{ij} = \partial \boldsymbol{F}_{ij}/\partial x$$

$\boldsymbol{x}$ 对 $\boldsymbol{F}(\boldsymbol{x})$ 微分

$$\frac{\partial \boldsymbol{F}}{\partial \boldsymbol{x}} = [\frac{\partial \boldsymbol{F}}{\partial x_1}, \dots, \frac{\partial \boldsymbol{F}}{\partial x_1}]^T$$

$\boldsymbol{X}$ 对 $\boldsymbol{F}(\boldsymbol{X})$ 微分

$$D_{\boldsymbol{X}}\boldsymbol{F} \overset{def}{=} \frac{\partial vec(\boldsymbol{F})}{\partial (vec\boldsymbol{X})^T} \in R^{pq\times mn}$$

$D_{\boldsymbol{X}}\boldsymbol{F} =$

$$\begin{pmatrix} \frac{\partial F_{11}}{\partial vec^T(\boldsymbol{X})} \\ \cdots \\ \frac{\partial F_{p1}}{\partial vec^T(\boldsymbol{X})} \\ \cdots \\ \frac{\partial F_{1q}}{\partial vec^T(\boldsymbol{X})} \\ \cdots \\ \frac{\partial F_{pq}}{\partial vec^T(\boldsymbol{X})} \end{pmatrix} = \begin{pmatrix} \frac{\partial F_{11}}{\partial x_{11}} & \cdots & \frac{\partial F_{11}}{\partial x_{m1}} & \cdots & \frac{\partial F_{11}}{\partial x_{1n}} & \cdots & \frac{\partial F_{11}}{\partial x_{mn}} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \frac{\partial F_{p1}}{\partial x_{11}} & \cdots & \frac{\partial F_{p1}}{\partial x_{m1}} & \cdots & \frac{\partial F_{p1}}{\partial x_{1n}} & \cdots & \frac{\partial F_{p1}}{\partial x_{mn}} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \frac{\partial F_{1q}}{\partial x_{11}} & \cdots & \frac{\partial F_{1q}}{\partial x_{m1}} & \cdots & \frac{\partial F_{1q}}{\partial x_{1n}} & \cdots & \frac{\partial F_{1q}}{\partial x_{mn}} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \frac{\partial F_{pq}}{\partial x_{11}} & \cdots & \frac{\partial F_{pq}}{\partial x_{m1}} & \cdots & \frac{\partial F_{pq}}{\partial x_{1n}} & \cdots & \frac{\partial F_{pq}}{\partial x_{mn}} \end{pmatrix}$$

$$\nabla_{\boldsymbol{X}}\boldsymbol{F} \overset{def}{=} \frac{\partial vec^T(\boldsymbol{F})}{\partial vec\boldsymbol{X}} \in R^{mn\times pq}$$

$$D_{\boldsymbol{X}}\boldsymbol{F} = (\nabla_{\boldsymbol{X}}\boldsymbol{F})^T$$

2.4 偏导数和梯度计算法则 (矩阵变元)

线性法则

$$\frac{[\partial c_1 f(\boldsymbol{X}) + c_2 g(\boldsymbol{X})]}{\partial \boldsymbol{X}} = c_1\frac{\partial f(\boldsymbol{X})}{\partial \boldsymbol{X}} + c_2\frac{\partial g(\boldsymbol{X})}{\partial \boldsymbol{X}}$$

乘积法则

$$\frac{\partial[f(\boldsymbol{X})g(\boldsymbol{X})]}{\partial \boldsymbol{X}} = g(\boldsymbol{X})\frac{\partial f(\boldsymbol{X})}{\partial \boldsymbol{X}} + f(\boldsymbol{X})\frac{\partial g(\boldsymbol{X})}{\partial \boldsymbol{X}}$$

$$\frac{\partial[f(\boldsymbol{X})g(\boldsymbol{X})h(\boldsymbol{X})]}{\partial \boldsymbol{X}} = g(\boldsymbol{X})h(\boldsymbol{X})\frac{\partial f(\boldsymbol{X})}{\partial \boldsymbol{X}} + f(\boldsymbol{X})h(\boldsymbol{X})\frac{\partial g(\boldsymbol{X})}{\partial \boldsymbol{X}} + f(\boldsymbol{X})g(\boldsymbol{X})\frac{\partial h(\boldsymbol{X})}{\partial \boldsymbol{X}}$$

商法则

$$\frac{\partial[f(\boldsymbol{X})/g(\boldsymbol{X})]}{\partial \boldsymbol{X}} = \frac{1}{g^2(\boldsymbol{X})}[g(\boldsymbol{X})\frac{\partial f(\boldsymbol{X})}{\partial \boldsymbol{X}} - f(\boldsymbol{X})\frac{\partial g(\boldsymbol{X})}{\partial \boldsymbol{X}}]$$

链式法则设 $y = f(\boldsymbol{X}), g(y)$

$$\frac{\partial g(f(\boldsymbol{X}))}{\partial \boldsymbol{X}} = \frac{dg(y)}{y}\frac{\partial f(\boldsymbol{X})}{\partial \boldsymbol{X}}$$

$$[\frac{\partial g(\boldsymbol{F}(\boldsymbol{X}))}{\partial \boldsymbol{X}}]_{ij} = \frac{\partial g(\boldsymbol{F})}{\partial x_{ij}} = \sum_{k=1}^{p}\sum_{l=1}^{q}\frac{\partial g(\boldsymbol{F})}{\partial f_{kl}}\frac{\partial f_{kl}}{\partial x_{ij}}$$

2.5 Jacobian 矩阵辨识

Jacobian 矩阵辨识主要是指利用微分来求 $f, \boldsymbol{f}, \boldsymbol{F}$ 的偏导的过程, 以实值标量函数为例说明, 实值标量函数全微分

$$\begin{aligned} df(\boldsymbol{x}) &= \frac{\partial f(\boldsymbol{x})}{\partial x_1}dx_1 + \cdots + \frac{\partial f(\boldsymbol{x})}{\partial x_m}dx_m \\ &= \begin{pmatrix} \frac{\partial f(\boldsymbol{x})}{\partial x_1} & \cdots & \frac{\partial f(\boldsymbol{x})}{\partial x_1} \end{pmatrix} \begin{pmatrix} dx_1 \\ \cdots \\ dx_m \end{pmatrix} \\ &= tr(D_{\boldsymbol{x}} f(\boldsymbol{x}) d\boldsymbol{x}) \end{aligned}$$

同理

$$df(\boldsymbol{X}) = tr(D_{\boldsymbol{X}} f(\boldsymbol{X}) d\boldsymbol{X})$$

所以, 求 $f(\boldsymbol{x})$ 偏导只要先 $df(\boldsymbol{x})$ 转换为 $tr(\boldsymbol{A}d\boldsymbol{x})$ 形式, 则可以得到 $\partial f/\partial \boldsymbol{x} = D_{\boldsymbol{x}}^T = \boldsymbol{A}^T$.

同理, 求 $f(\boldsymbol{X})$ 偏导只要先 $df(\boldsymbol{X})$ 转换为 $tr(\boldsymbol{A}d\boldsymbol{X})$ 形式, 则可以得到 $\partial f/\partial \boldsymbol{X} = D_{\boldsymbol{X}}^T = \boldsymbol{A}^T$.

应用微分求偏导时主要利用如下性质, 矩阵迹的性质和微分的性质进行变换

$$f = tr(f)$$

$$df = dtr(f) = tr(df)$$

一阶辨识表

函数类型	矩阵微分	Jacobian 矩阵
$f(x)$	$df(x) = Adx$	$A \in \mathcal{R}$
$f(\boldsymbol{x})$	$df(\boldsymbol{x}) = \boldsymbol{A}d\boldsymbol{x}$	$\boldsymbol{A} \in \mathcal{R}^{1\times m}$
$f(\boldsymbol{X})$	$df(\boldsymbol{X}) = tr(\boldsymbol{A}d\boldsymbol{x})$	$\boldsymbol{A} \in \mathcal{R}^{n\times m}$
$\boldsymbol{f}(\boldsymbol{x})$	$d\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{A}d\boldsymbol{x}$	$\boldsymbol{A} \in \mathcal{R}^{p\times m}$
$\boldsymbol{f}(\boldsymbol{X})$	$d\boldsymbol{f}(\boldsymbol{X}) = \boldsymbol{A}dvec(\boldsymbol{X})$	$\boldsymbol{A} \in \mathcal{R}^{p\times mn}$
$\boldsymbol{F}(\boldsymbol{x})$	$dvec(\boldsymbol{F}(\boldsymbol{x})) = \boldsymbol{A}d\boldsymbol{x}$	$\boldsymbol{A} \in \mathcal{R}^{pq\times m}$
$\boldsymbol{F}(\boldsymbol{X})$	$d\boldsymbol{F}(\boldsymbol{X}) = \boldsymbol{A}(d\boldsymbol{X})\boldsymbol{B}$	$\boldsymbol{B}^T \otimes \boldsymbol{A} \in \mathcal{R}^{pq\times mn}$
$\boldsymbol{F}(\boldsymbol{X})$	$d\boldsymbol{F}(\boldsymbol{X}) = \boldsymbol{C}(d\boldsymbol{X}^T)\boldsymbol{D}$	$(\boldsymbol{D}^T \otimes \boldsymbol{C})\boldsymbol{K}_{mn} \in \mathcal{R}^{pq\times mn}$

二阶辨识表

函数类型	矩阵微分	Jacobian 矩阵
$f(x)$	βdx	$\beta \in \mathcal{R}$
$f(\boldsymbol{x})$	$(d\boldsymbol{x})^T\boldsymbol{B}d\boldsymbol{x}$	$\frac{1}{2}(\boldsymbol{B} + \boldsymbol{B}^T) \in \mathcal{R}^{m\times m}$
$f(\boldsymbol{X})$	$d(vec(\boldsymbol{X}))^T\boldsymbol{B}dvec(\boldsymbol{X}))$	$\frac{1}{2}(\boldsymbol{B} + \boldsymbol{B}^T) \in \mathcal{R}^{mn\times mn}$
$\boldsymbol{f}(x)$	$\boldsymbol{b}(dx)^2$	$\boldsymbol{b} \in \mathcal{R}^{p\times 1}$
$\boldsymbol{f}(\boldsymbol{x})$	$(\boldsymbol{I}_m \otimes d\boldsymbol{x})^T\boldsymbol{B}d\boldsymbol{x}$	$\frac{1}{2}(\boldsymbol{B} + (\boldsymbol{B}')_v^T) \in \mathcal{R}^{pm\times m}$
$\boldsymbol{f}(\boldsymbol{X})$	$(\boldsymbol{I}_m \otimes dvec(\boldsymbol{X}))^T\boldsymbol{B}dvec(\boldsymbol{X})$	$\frac{1}{2}(\boldsymbol{B} + (\boldsymbol{B}')_v^T) \in \mathcal{R}^{pmn\times mn}$
$\boldsymbol{F}(x)$	$\boldsymbol{B}(dx)^2$	$vec(\boldsymbol{B}) \in \mathcal{R}^{pq\times 1}$
$\boldsymbol{F}(\boldsymbol{x})$	$d^2vec(\boldsymbol{F}) = (\boldsymbol{I}_m \otimes d\boldsymbol{x})^T\boldsymbol{B}d\boldsymbol{x}$	$\frac{1}{2}(\boldsymbol{B} + (\boldsymbol{B}')_v^T) \in \mathcal{R}^{pmq\times m}$
$\boldsymbol{F}(\boldsymbol{X})$	$d^2vec(\boldsymbol{F}) = (\boldsymbol{I}_m \otimes dvec(\boldsymbol{X}))^T\boldsymbol{B}dvec(\boldsymbol{X})$	$\frac{1}{2}(\boldsymbol{B} + (\boldsymbol{B}')_v^T) \in \mathcal{R}^{pmqn\times mn}$

3 张量代数与微积分

3.1 爱因斯坦求和约定

Rule 1在一个表达式中出现 1 次的 index 称为 free index, 出现 2 次的 index 称为 dummy index.

Rule 2若等式左边表达式中 dummy index 不在等式右边出现, 则对 dummy index 求和. $a_ix_i := \sum_i a_ix_i$, $a_{ij}b_j := \sum_j a_{ij}b_i$

Rule 3若等式右边的 free index 不在等式左边出现, 则对该 free index 求和. 如:

$$c_i = a_{ij}b_j = \sum_j a_{ij}b_j, \quad c = a_{ij}b_j = \sum_{i,j} a_{ij}b_j$$

Rule 4若等式右边的 dummy index 在等式左边出现, 则该 dummy index 不求和, 如 $c_{ij} = a_ib_{ij}$

两个常用的符号

- Kronecker-delta 符号

$$\delta = \begin{cases} 1, & i = j \\ 0, & i \neq j \end{cases}$$

- Levi-Civita 符号

$$\epsilon_{a_1a_2\ldots a_n} = \begin{cases} 1, & (a_1, \ldots, a_n) \text{ 是偶排列} \\ -1, & (a_1, a_2, \ldots, a_n) \text{ 是奇排列} \\ 0, & \text{otherwise} \end{cases}$$

3.2 张量求导法则 (Einstein Notaion)

线性法则设自变量 x 的 index set 为 s_4, 若有 $C_{s_3}(x) = a_{s_1'}A_{s_1}(x) + a_{s_2'}B_{s_2}(x)$, 则

$$(\frac{\partial C(x)}{\partial x})_{s_3 s_4} = a_{s_1'}(\frac{\partial A(x)}{\partial x})_{s_1 s_4} + b_{s_2'}(\frac{\partial B(x)}{\partial x})_{s_2 s_4}$$

乘积法则设自变量 x 的 index set 为 s_4, 若有 $C_{s_3}(x) = A_{s_1}(x)B_{s_2}(x)$, 则

$$(\frac{\partial C(x)}{\partial x})_{s_3 s_4} = B(x)_{s_2}(\frac{\partial A(x)}{\partial x})_{s_1 s_4} + A(x)_{s_1}(\frac{\partial B(x)}{\partial x})_{s_2 s_4}$$

由乘积法则可以继续推出, 若有 $C_{s_3}(x) = [D_{s_5}(x)E_{s_6}(x)]B_{s_2}(x)$

$$\begin{aligned}(\frac{\partial C(x)}{\partial x})_{s_3 s_4} =& B(x)_{s_2}[E_{s_6}(x)(\frac{\partial D(x)}{\partial x})_{s_5 s_4} + D_{s_5}(x)(\frac{\partial E(x)}{\partial x})_{s_6 s_4}] \\ &+ D_{s_5}(x)E_{s_6}(x)(\frac{\partial B(x)}{\partial x})_{s_2 s_4} \\ =& B(x)_{s_2}E_{s_6}(x)(\frac{\partial D(x)}{\partial x})_{s_5 s_4} + B(x)_{s_2}D_{s_5}(x)(\frac{\partial E(x)}{\partial x})_{s_6 s_4} \\ &+ D_{s_5}(x)E_{s_6}(x)(\frac{\partial B(x)}{\partial x})_{s_2 s_4}\end{aligned}$$

商法则设自变量 x 的 index set 为 s_4, 若有 $C_{s_3}(x) = \frac{A_{s_1}(x)}{B_{s_2}(x)}$, 则

$$(\frac{\partial C(x)}{\partial x})_{s_3 s_4} = \frac{B(x)_{s_2}(\frac{\partial A(x)}{\partial x})_{s_1 s_4} - A(x)_{s_1}(\frac{\partial B(x)}{\partial x})_{s_2 s_4}}{B_{s2}^2}$$

链式法则设自变量 x 的 index set 为 s_3, 设映射 f 为 unary function, 其定义域 index set 为 s_1, 值域 index set 为 s_2, 若有 $C(x)_{s_2} = f(A_{s_1}(x))$, 则

$$(\frac{\partial C(x)}{\partial x})_{s_2 s_3} = (\frac{\partial f(A)}{\partial x})_{s_2 s_1}(\frac{\partial A(x)}{\partial x})_{s_1 s_3}$$

设自变量 x 的 index set 为 s_2, 设映射 f 为 unary elementwise function, 其定义域和值域 index set 为 s_1, 若有 $C(x)_{s_1} = f(A_{s_1}(x))$, 则

$$(\frac{\partial C(x)}{\partial x})_{s_1 s_2} = (\frac{\partial f(A)}{\partial x})_{s_1}(\frac{\partial A(x)}{\partial x})_{s_1 s_2}$$

3.3 简单示例

二阶张量求逆对张量 G 求逆, 可以定义为

$$det(M) = \frac{1}{n!}\epsilon_{i_1 i_2 \dots i_n}\epsilon_{j_1 j_2 \dots j_n}G_{i_1 j_1}G_{i_2 j_2}\dots G_{i_n j_n}$$

$$adj(M)_{ij} = \frac{1}{(n-1)!}\epsilon_{i i_2 \dots i_n}\epsilon_{j j_2 \dots j_n}G_{i_2 j_2}\dots G_{i_n j_n}$$

$$inv(G)_{ij} = \frac{adj(G)_{ij}}{det(G)}$$

二阶张量逆的求导

$$(\frac{inv(G)}{\partial G})_{ijkl} = \{det(G)(\frac{\partial adj(G)}{\partial G})_{ijkl} + adj_{ij}(\frac{det(G)}{G})_{kl}\}/det(G)^2$$

$$\begin{aligned}(\frac{\partial adj(G)}{\partial G})_{xipq} =& \partial\{\frac{1}{(3-1)!}\epsilon_{ijk}\epsilon_{xyz}G_{jy}G_{kz}\}/\partial G_{pq} \\ =& \frac{1}{(3-1)!}\epsilon_{ijk}\epsilon_{xyz}(G(v)_{kz}\delta_{jp}\delta_{yq} + G(v)_{jy}\delta_{kp}\delta_{zq})\end{aligned}$$

$$\begin{aligned}(\frac{\partial det(G)}{\partial G})_{pq} =& \partial\{\frac{1}{3!}\epsilon_{ijk}\epsilon_{xyz}G_{ix}G_{jy}G_{kz}\}/\partial G_{pq} \\ =& \frac{1}{3!}\epsilon_{ijk}\epsilon_{xyz}(G_{jy}G_{kz}\delta_{ip}\delta_{xq} \\ &+ G_{ix}G_{kz}\delta_{jp}\delta_{yq} + G_{ix}G_{jy}\delta_{kp}\delta_{zq})\end{aligned}$$

4 凸优化

凸优化问题往往没有一个解析表达式, 存在很多有效的算法求解凸优化问题, 如内点法等, 内点法可以在多项式时间内给定精度求解这些凸优化问题, 判断某个问题是否属于凸优化问题或识别哪些可以转换为凸优化问题是关键.

4.1 基本定义

凸集:C 是凸集 $\Leftrightarrow \boldsymbol{x}_1, \boldsymbol{x}_2 \in C$, 对于 $0 \le \theta \le 1$, 有

$$\theta\boldsymbol{x_1} + (1-\theta)\boldsymbol{x}_2 \in C$$

凸集例子:

保凸运算:

凸函数定义:若函数 c 是凸函数, 则 $dom(c)$ 是凸集, 对于 $x, y \in dom(c)$ 和任意 $0 \le \theta \le 1$, 有

$$c(\theta x + (1-\theta)y) \le \theta c(x) + (1-\theta)c(y)$$

仿射函数:

$$c(\boldsymbol{x}) = \boldsymbol{A}\boldsymbol{x} - \boldsymbol{b} \text{ 或} c(\boldsymbol{x}) = \boldsymbol{a}^T\boldsymbol{x} - b$$

凸函数一阶条件:c 是凸函数 $\Leftrightarrow$ c 可微 (∇c 在 $dom(c)$ 内处处存在),$dom(c)$ 是凸集, 对于 $x, y \in dom(c)$

$$c(\boldsymbol{y}) \ge c(\boldsymbol{x}) + \nabla c(\boldsymbol{x})^T(\boldsymbol{y} - \boldsymbol{x})$$

凸函数二阶条件:c 是凸函数 $\Leftrightarrow$ c 二阶可微,$dom(c)$ 是凸集, 对于 $x \in dom(c)$

$$\nabla^2 c(\boldsymbol{x}) \ge 0$$

凸函数例子:

保凸运算:

4.2 凸优化问题

优化问题:

$$\begin{aligned}\min_{\boldsymbol{x}\in\mathcal{R}^n} \quad & f(\boldsymbol{x}) \\ s.t. \quad & c_i(\boldsymbol{x}) \le 0, \quad i \in \mathcal{I} \\ & c_i(\boldsymbol{x}) = 0, \quad i \in \mathcal{E}\end{aligned}$$

$\boldsymbol{x}$ 是优化变量,$f: R^n \to R$ 是目标函数.$\mathcal{I}, \mathcal{E}$ 是有界指示数据集合.

凸优化问题:目标函数和不等式约束函数是凸函数, 等式约束函数是仿射函数.

$$\begin{aligned} \underset{\boldsymbol{x}\in\mathcal{R}^n}{min} \quad & f(\boldsymbol{x}) \\ s.t. \quad & c_i(\boldsymbol{x}) \le 0, \quad i \in \mathcal{I} \\ & \boldsymbol{a_i}^T\boldsymbol{x} - b_i = 0, \quad i \in \mathcal{E} \end{aligned}$$

凸优化问题的可行集是凸的:可行集 $\mathcal{D}$ 是一个凸集,$|\mathcal{I}|$ 个下水平集 $\{\boldsymbol{x}|c_i(\boldsymbol{x}) \le 0\}$, 以及 $|\mathcal{E}|$ 个超平面 $\{\boldsymbol{x}|\boldsymbol{a_i}^T\boldsymbol{x} - b_i = 0\}$ 的交集.

全局最优解与局部最优解:凸优化问题任意局部最优解也是全局最优解

最优性准则 (有约束):$\boldsymbol{x}$ 是最优解, 当且仅当 $x \in \mathcal{D}$ 且

$$\nabla f(\boldsymbol{x})^T(\boldsymbol{y} - \boldsymbol{x}) \ge 0, \forall \boldsymbol{y} \in \mathcal{D}$$

最优性准则 (无约束):

$$\nabla f(\boldsymbol{x}) = 0$$

4.2.1 等价凸问题

消除等式约束:凸问题的等式约束可以写为 $\boldsymbol{Ax} = \boldsymbol{b}$, 则 $\boldsymbol{x} = \boldsymbol{Fz} + \boldsymbol{x}_0$, 其中 $\boldsymbol{x}_0$ 是一个特解,$\boldsymbol{F}$ 是域为 $\boldsymbol{A}$ 的零空间矩阵. 原优化问题变为

$$\begin{aligned} \underset{\boldsymbol{z}\in\mathcal{R}^n}{min} \quad & f(\boldsymbol{Fz} + \boldsymbol{x}_0) \\ s.t. \quad & c_i(\boldsymbol{Fz} + \boldsymbol{x}_0) \le 0, \quad i \in \mathcal{I} \end{aligned}$$

理论上好处是可以集中精力于不含等式约束的凸优化问题, 但很多时候会变得更难以理解和求解, 因此最好保留等式约束.

引入等式约束:引入新变量和线性等式约束.

松弛变量:如果不等式约束 c_i 是仿射的, 则引入变量 s_i, 将不等式约束 $c_i(\boldsymbol{x}) \ge 0$ 变为 $c_i(\boldsymbol{x}) + s_i = 0$.

上境图问题形式:如果目标函数 f 是线性的, 则可改写为

$$\begin{aligned} min\, t s.t. \quad & f(\boldsymbol{x}) - t \le 0 \\ & c_i(\boldsymbol{x}) \le 0, \quad i \in \mathcal{I} \\ & \boldsymbol{a_i}^T\boldsymbol{x} - b_i = 0, \quad i \in \mathcal{E} \end{aligned}$$

优化部分变量:

$$\underset{x,y}{inf}\, f(x,y) = \underset{x}{inf}\, \widehat{f}(x)$$

其中,$\widehat{f}(x) = \underset{y}{inf}\, f(x,y)$, 即总可以通过先优化一部分变量在优化另一部分变量达到优化一个函数的目的.

4.2.2 常见凸优化问题形式

4.2.3 对偶

Lagrange 函数:针对一般形式优化问题

$$L(x,\lambda,v) = f(x) + \sum_{i\in\mathcal{I}} \lambda c_i(x) + \sum_{i\in\mathcal{E}} v_i c_i(x)$$

Lagrange 对偶函数:

$$g(\lambda,v) = \underset{x\in\mathcal{D}}{inf}\, L(x,\lambda,v)$$

$$dom(g) = \{(\lambda,v)|g(\lambda,v) > -\infty\}$$

Lagrange 对偶函数性质:设 Lagrange 对偶问题最优值为 d^*, 原问题最优值为 p^*.$\widehat{x}$ 表示可行解.

$$g(\lambda,v) = \underset{x\in\mathcal{D}}{inf}\, L(x,\lambda,v) \le q^* \le L(\widehat{x},\lambda,v) \le f(\widehat{x})$$

对偶可行:

$$\lambda \ge 0 \text{ 且} (\lambda,v) \in dom(g)$$

Lagrange 对偶问题: Lagrange 函数能得到的最好下界是什么?

$$\begin{aligned} & max\, g(\lambda,v) \\ & s.t. \lambda \ge 0 \end{aligned}$$

对偶问题总是凸问题.

最优对偶间隙:$p^* - d^*$

弱对偶性:$d^* \le p^*$

强对偶性:$d^* = p^*$

强对偶性一般不成立, 凸问题强对偶性通常 (但不总是) 成立. 很多研究成果给出了强对偶性成立的条件 (除凸性外), 称为约束准则, 一个简单的约束准则是 Slater 条件.

Slate 条件:$\exists x \in relint(\mathcal{D})$ 使得

$$c_i(\boldsymbol{x}) < 0, i \in \mathcal{I}, \boldsymbol{Ax} = \boldsymbol{b}$$

当 Slate 条件成立且原问题为凸问题时, 强对偶性成立.

Karush-Kuhn-Tucker(KKT) 条件:

$$\begin{aligned} & c_i(x*) \le 0, \quad i \in \mathcal{I} \\ & c_i(x*) = 0, \quad i \in \mathcal{E} \\ & \lambda_i^* \ge 0, \quad i \in \mathcal{I} \\ & \lambda_i^* c_i(x^*) = 0, \quad i \in \mathcal{I} \cup \mathcal{E} \\ & \nabla f(x^*) + \sum_{i\in\mathcal{I}} \lambda_i^* \nabla c_i(x^*) + \sum_{i\in\mathcal{E}} v_i^* \nabla c_i(x^*) = 0 \end{aligned}$$

非凸问题的 KKT 条件:对于目标函数和约束函数可微的任意优化问题, 如果强对偶性成立 (非凸问题一般不成立), 那么任何一对原问题最优解和对偶问题最优解 x, λ, v 必满足 KKT 条件.

凸问题的 KKT 条件:当原问题是凸问题, 强对偶性成立, 满足 KKT 条件的 x, λ, v 是原, 对偶问题的最优解.

5 动态最优化

5.1 变分法

5.2 最优控制理论

5.3 动态规划

Part II

概率

6 概率形式化描述

7 概率图

8 MCMC 与变分推断

Part III

数值算法

9 最优化理论数值算法

9.1 整数优化与组合优化

9.2 线性规划

9.2.1 单纯形法

9.2.2 内点法

9.3 连续无约束非线性最优化

9.3.1 基础

9.3.2 无约束优化问题最优条件

必要条件:如果 $\boldsymbol{x}^*$ 是最优解, 那么 $\nabla f(\boldsymbol{x}^*)$ 并且 $\nabla^2 f(\boldsymbol{x}^*)$ 是半正定.

充分条件:对于任意点 $\boldsymbol{x}^*$, 如果 $\nabla f(\boldsymbol{x}^*)$ 并且 $\nabla^2 f(\boldsymbol{x}^*)$ 是正定. 那么 $\boldsymbol{x}^*$ 是 f 的一个 strong local minimizer.

线搜索方法

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha_k \boldsymbol{p}_k$$

其中 α_k 是 step length, 线搜索方法是否有效取决于方向 $\boldsymbol{p}_k$ 和步长 α 的选取.

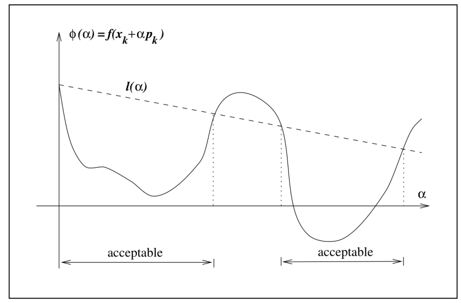
$\boldsymbol{p}_k$ 的选取,

- 一种是 steepest descent direction:$-\nabla f_k$
- 一种是 Newton direction:$-(\nabla^2 f_k)^{-1}\nabla f_k$

α_k 的选取

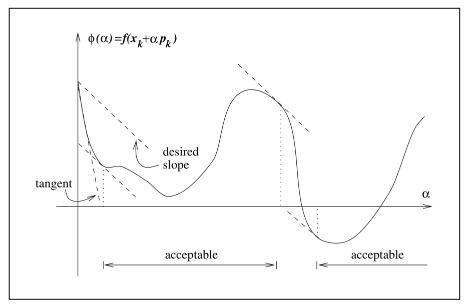
- sufficient decrease condtion:

$$f(\boldsymbol{x}_k + \alpha \boldsymbol{p}_k) \leq f(\boldsymbol{x}_k) + c_1 \alpha \nabla f_k^T \boldsymbol{p}_k \qquad c_1 \in (0,1)$$



- curvature condition:

$$\nabla f(\boldsymbol{x}_k + \alpha_k \boldsymbol{p}_k)^T \boldsymbol{p}_k \geq c_2 \nabla f_k^T \boldsymbol{p}_k \qquad c_2 \in (c_1, 1)$$



sufficient decrease condition 与 curvature conditon 合起来就是 Wolfe conditions. 另外,strong Wolfe conditions 是

$$f(\boldsymbol{x}_k + \alpha \boldsymbol{p}_k) \leq f(\boldsymbol{x}_k) + c_1 \alpha \nabla f_k^T \boldsymbol{p}_k \qquad c_1 \in (0,1)$$
$$|\nabla f(\boldsymbol{x}_k + \alpha_k \boldsymbol{p}_k)^T \boldsymbol{p}_k| \leq c_2 |\nabla f_k^T \boldsymbol{p}_k| \qquad c_2 \in (c_1, 1)$$

回溯线搜索

Algorithm 3.1 (Backtracking Line Search).
Choose $\bar{\alpha} > 0$, $\rho \in (0,1)$, $c \in (0,1)$; Set $\alpha \leftarrow \bar{\alpha}$;
repeat until $f(x_k + \alpha p_k) \leq f(x_k) + c\alpha \nabla f_k^T p_k$
  $\alpha \leftarrow \rho\alpha$;
end (repeat)
Terminate with $\alpha_k = \alpha$.

非常适用于 Newton 法, 但是没那么适用于拟牛顿法和 CG 法.

α_0 初值选取, 对于牛顿和拟牛顿,$\alpha_0 = 1$, 对于 steepest descent 与 CG. 有两种流行的选择方式:

- 选择 α_0, 令 $\alpha_0 \nabla f_k^T \boldsymbol{p}_k = \alpha_{k-1} \nabla f_{k-1}^T \boldsymbol{p}_{k-1}$

$$\alpha_0 = \alpha_{k-1} \frac{\nabla f_{k-1}^T \boldsymbol{p}_{k-1}}{\nabla f_k^T \boldsymbol{p}_k}$$

- $$\alpha_0 = \frac{2(f_k - f_{k-1})}{\phi'(0)}$$

line search 算法 ，以下算法返回 α_*(满足 strong Wolfe condition)

Algorithm 3.5 (Line Search Algorithm).

Set $\alpha_0 \leftarrow 0$, choose $\alpha_{\max} > 0$ and $\alpha_1 \in (0, \alpha_{\max})$;
$i \leftarrow 1$;
repeat
  Evaluate $\phi(\alpha_i)$;
  if $\phi(\alpha_i) > \phi(0) + c_1\alpha_i\phi'(0)$ or $[\phi(\alpha_i) \ge \phi(\alpha_{i-1})$ and $i > 1]$
    $\alpha_* \leftarrow$**zoom**(α_{i-1}, α_i) and **stop**;
  Evaluate $\phi'(\alpha_i)$;
  if $|\phi'(\alpha_i)| \le -c_2\phi'(0)$
    set $\alpha_* \leftarrow \alpha_i$ and **stop**;
  if $\phi'(\alpha_i) \ge 0$
    set $\alpha_* \leftarrow$**zoom**(α_i, α_{i-1}) and **stop**;
  Choose $\alpha_{i+1} \in (\alpha_i, \alpha_{\max})$;
  $i \leftarrow i + 1$;
end (repeat)

Algorithm 3.6 (zoom).

repeat
  Interpolate (using quadratic, cubic, or bisection) to find
    a trial step length α_j between α_{lo} and α_{hi};
  Evaluate $\phi(\alpha_j)$;
  if $\phi(\alpha_j) > \phi(0) + c_1\alpha_j\phi'(0)$ or $\phi(\alpha_j) \ge \phi(\alpha_{\text{lo}})$
    $\alpha_{\text{hi}} \leftarrow \alpha_j$;
  else
    Evaluate $\phi'(\alpha_j)$;
    if $|\phi'(\alpha_j)| \le -c_2\phi'(0)$
      Set $\alpha_* \leftarrow \alpha_j$ and **stop**;
    if $\phi'(\alpha_j)(\alpha_{\text{hi}} - \alpha_{\text{lo}}) \ge 0$
      $\alpha_{\text{hi}} \leftarrow \alpha_{\text{lo}}$;
    $\alpha_{\text{lo}} \leftarrow \alpha_j$;
end (repeat)

9.3.3 牛顿法

9.3.4 拟牛顿法

把目标函数在 $\boldsymbol{x}_k$ 附近近似成 quadratic 模型,$\boldsymbol{B}_k$ 是 $n \times n$ 对称正定矩阵

$$m_k(\boldsymbol{p}) = f_k + \nabla f_k^T\boldsymbol{p} + \frac{1}{2}\boldsymbol{p}^T\boldsymbol{B}_k\boldsymbol{p}$$

$$\boldsymbol{p}_k = \underset{\boldsymbol{p}_k}{argmin}\, m_k(\boldsymbol{p}_k)$$

$$\boldsymbol{p}_k = -\boldsymbol{B}_k^{-1}\nabla f_k$$

$$\nabla m_{k+1}(-\alpha_k\boldsymbol{p}_k) = \nabla f_{k+1} - \alpha_k\boldsymbol{B}_{k+1}\boldsymbol{p}_k = \nabla m_k(0) = \nabla f_k$$

$$\boldsymbol{B}_{k+1}\alpha_k\boldsymbol{p}_k = \triangle f_{k+1} - \nabla f_k$$

重新表述为

$$s_k = x_{k+1} - x_k = \alpha_k\boldsymbol{p}_k \qquad y_k = \nabla f_{k+1} - \triangle f_k$$

$$\boldsymbol{B}_{k+1}s_k = y_k$$

DFP 方法如何找到 $\boldsymbol{B}_{k+1}$

$$\underset{\boldsymbol{B}}{min}\,||\boldsymbol{B} - \boldsymbol{B}_k||$$

$$s.t.\boldsymbol{B} = \boldsymbol{B}^T, \quad \boldsymbol{B}s_k = y_k$$

可以得到唯一解,DFP 方法

$$\boldsymbol{B}_{k+1} = (\boldsymbol{I} - \boldsymbol{\rho}_k\boldsymbol{y}_k\boldsymbol{s}_k^T)\boldsymbol{B}_K(\boldsymbol{I} - \boldsymbol{\rho}_k\boldsymbol{s}_k\boldsymbol{y}_k^T) + \boldsymbol{\rho}_k\boldsymbol{y}_k\boldsymbol{y}_k^T$$

$$\boldsymbol{\rho}_k = \frac{1}{\boldsymbol{y}_k^T\boldsymbol{s}_k}$$

在更新 $\boldsymbol{p}$ 用的是 $\boldsymbol{H}_k = \boldsymbol{B}_k^{-1}$,

$$\boldsymbol{H}_{k+1} = \boldsymbol{H}_k - \frac{\boldsymbol{H}_k\boldsymbol{y}_k\boldsymbol{y}_k^T\boldsymbol{H})_k}{\boldsymbol{y}_k^T\boldsymbol{H}_k\boldsymbol{y}_k} + \frac{\boldsymbol{s}_k\boldsymbol{s}_{\boldsymbol{k}}^T}{\boldsymbol{y}_k^T\boldsymbol{s}_k}$$

BFGS 方法

$$\underset{\boldsymbol{H}}{min}\,||\boldsymbol{H} - \boldsymbol{H}_k||$$

$$s.t.\boldsymbol{H} = \boldsymbol{H}^T, \quad \boldsymbol{H}\boldsymbol{y}_k = \boldsymbol{s}_k$$

$$\boldsymbol{H}_{k+1} = (\boldsymbol{I} - \boldsymbol{\rho}_k\boldsymbol{s}_k\boldsymbol{y}_k^T)\boldsymbol{H}_k(\boldsymbol{I} - \boldsymbol{\rho}_k\boldsymbol{y}_k\boldsymbol{s}_k^T) + \boldsymbol{\rho}_k\boldsymbol{s}_k\boldsymbol{s}_k^T$$

Algorithm 6.1 (BFGS Method).

Given starting point x_0, convergence tolerance $\epsilon > 0$,
  inverse Hessian approximation H_0;
$k \leftarrow 0$;
while $\|\nabla f_k\| > \epsilon$;
  Compute search direction

$$p_k = -H_k\nabla f_k;$$

  Set $x_{k+1} = x_k + \alpha_k p_k$ where α_k is computed from a line search
    procedure to satisfy the Wolfe conditions (3.6);
  Define $s_k = x_{k+1} - x_k$ and $y_k = \nabla f_{k+1} - \nabla f_k$;
  Compute H_{k+1} by means of (6.17);
  $k \leftarrow k + 1$;
end (while)

SR1 方法SR1 不保证更新的矩阵能保持正定性, 有好的数值结果.

9.3.5 共轭梯度下降法

9.4 连续有约束非线性最优化

9.4.1 基础

$$\mathcal{A}(\boldsymbol{x}) = \mathcal{E} \cup \{i \in \mathcal{I} | c_i(\boldsymbol{x} = 0\}$$

$\mathcal{A}(\boldsymbol{x})$ 是 active set, 如果 $c_i(\boldsymbol{x}) = 0$, 则是 active 的, 如果 $c_i(\boldsymbol{x}) > 0$ 则称为 inactive 的.

如果 active set constraint 的梯度, 即 $\{\nabla c_i(\boldsymbol{x}), i \in \mathcal{A}(\boldsymbol{x})\}$ 是线性无关的, 则 LICQ 成立.

一阶最优条件 (KKT 条件)设 Lagrangian function 为

$$\mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}) = f(\boldsymbol{x}) - \sum_{i \in \mathcal{E} \cup \mathcal{I}} \lambda_i c_i(\boldsymbol{x})$$

如果 $\boldsymbol{x}^*$ 是 local solution,f, c_i 都是连续可微的且 LICQ 在 $\boldsymbol{x}^*$ 成立,$\boldsymbol{\lambda}^*$ 是 lagrange multiplier. 则

$$c_i(x*) \le 0, \quad i \in \mathcal{I}$$

$$c_i(x*) = 0, \quad i \in \mathcal{E}$$

$$\lambda_i^* \ge 0, \quad i \in \mathcal{I}$$

$$\lambda_i^* c_i(x^*) = 0, \quad i \in \mathcal{I} \cup \mathcal{E}$$

$$\nabla f(x^*) + \sum_{i \in \mathcal{I}} \lambda_i^* \nabla c_i(x^*) + \sum_{i \in \mathcal{E}} v_i^* \nabla c_i(x^*) = 0$$

9.4.2 增强拉格朗日算法

$$\lambda_i^{k+1} = \lambda_i^k - \mu_k c_i(x_k), \quad \text{for all } i \in \mathcal{E}. \tag{17.39}$$

Framework 17.3 (Augmented Lagrangian Method-Equality Constraints).
Given $\mu_0 > 0$, tolerance $\tau_0 > 0$, starting points x_0^s and λ^0;
for $k = 0, 1, 2, \ldots$
 Find an approximate minimizer x_k of $\mathcal{L}_A(\cdot, \lambda^k; \mu_k)$, starting at x_k^s,
  and terminating when $\|\nabla_x \mathcal{L}_A(x_k, \lambda^k; \mu_k)\| \le \tau_k$;
 if a convergence test for (17.1) is satisfied
  stop with approximate solution x_k;
 end (if)
 Update Lagrange multipliers using (17.39) to obtain λ^{k+1};
 Choose new penalty parameter $\mu_{k+1} \ge \mu_k$;
 Set starting point for the next iteration to $x_{k+1}^s = x_k$;
 Select tolerance τ_{k+1};
end (for)

$$\min_x \mathcal{L}_A(x, \lambda; \mu) \quad \text{subject to } l \le x \le u. \tag{17.50}$$

Algorithm 17.4 (Bound-Constrained Lagrangian Method).
Choose an initial point x_0 and initial multipliers λ^0;
Choose convergence tolerances η_* and ω_*;
Set $\mu_0 = 10$, $\omega_0 = 1/\mu_0$, and $\eta_0 = 1/\mu_0^{0.1}$;
for $k = 0, 1, 2, \ldots$
 Find an approximate solution x_k of the subproblem (17.50) such that

$$\left\| x_k - P\left(x_k - \nabla_x \mathcal{L}_A(x_k, \lambda^k; \mu_k), l, u\right)\right\| \le \omega_k;$$

 if $\|c(x_k)\| \le \eta_k$
  (* test for convergence *)
  if $\|c(x_k)\| \le \eta_*$ and $\left\| x_k - P\left(x_k - \nabla_x \mathcal{L}_A(x_k, \lambda^k; \mu_k), l, u\right)\right\| \le \omega_*$
   stop with approximate solution x_k;
  end (if)
  (* update multipliers, tighten tolerances *)
  $\lambda^{k+1} = \lambda^k - \mu_k c(x_k)$;
  $\mu_{k+1} = \mu_k$;
  $\eta_{k+1} = \eta_k / \mu_{k+1}^{0.9}$;
  $\omega_{k+1} = \omega_k / \mu_{k+1}$;
 else
  (* increase penalty parameter, tighten tolerances *)
  $\lambda^{k+1} = \lambda^k$;
  $\mu_{k+1} = 100\mu_k$;
  $\eta_{k+1} = 1/\mu_{k+1}^{0.1}$;
  $\omega_{k+1} = 1/\mu_{k+1}$;
 end (if)
end (for)

9.4.3 SQP

SQP 算法引出考虑含等式约束优化问题

$$min\ f(\boldsymbol{x})$$
$$s.t.\ \boldsymbol{c}(\boldsymbol{x}) = 0$$

牛顿法视角, 即用牛顿法解该问题 KKT 条件, 由 KKT 条件, 设

$$\boldsymbol{F}(\boldsymbol{x}, \boldsymbol{\lambda}) = \begin{pmatrix} \nabla f(\boldsymbol{x}) - \boldsymbol{A}(\boldsymbol{x})^T\boldsymbol{\lambda} \\ \boldsymbol{c}(\boldsymbol{x}) \end{pmatrix} = 0$$
$$\boldsymbol{A}(\boldsymbol{x})^T = [\nabla c_1(\boldsymbol{x}), \ldots, \nabla c_m(\boldsymbol{x})]$$

$\boldsymbol{F}(\boldsymbol{x}, \boldsymbol{\lambda})$ 的 Jacobian 矩阵

$$\boldsymbol{J_F} = \begin{pmatrix} \nabla^2_{\boldsymbol{xx}}\boldsymbol{\mathcal{L}}(\boldsymbol{x}, \boldsymbol{\lambda}) & -\boldsymbol{A}(\boldsymbol{x})^T \\ \boldsymbol{A}(\boldsymbol{x})^T & \boldsymbol{0} \end{pmatrix}$$

$$\nabla^2_{\boldsymbol{xx}}\boldsymbol{\mathcal{L}}(\boldsymbol{x}, \boldsymbol{\lambda}) = \nabla_{\boldsymbol{x}}(\nabla f(\boldsymbol{x}) - \boldsymbol{A}(\boldsymbol{x})^T\boldsymbol{\lambda})$$

牛顿法更新 $(\boldsymbol{x}_k, \boldsymbol{\lambda}_k)$

$$\begin{pmatrix} \boldsymbol{x}_{k+1} \\ \boldsymbol{\lambda}_{k+1} \end{pmatrix} = \begin{pmatrix} \boldsymbol{x}_k \\ \boldsymbol{\lambda}_k \end{pmatrix} + \begin{pmatrix} \boldsymbol{p}_k \\ \boldsymbol{u}_k \end{pmatrix}$$
$$\boldsymbol{J_F}\begin{pmatrix} \boldsymbol{p}_k \\ \boldsymbol{p}_\lambda \end{pmatrix} = -\boldsymbol{F}(\boldsymbol{x}, \boldsymbol{\lambda}) \tag{1}$$

SQP 框架下视角, 设 QP 问题

$$\min_{\boldsymbol{p}} f_k + \nabla f_k^T \boldsymbol{p}\frac{1}{2} + \boldsymbol{p}^T\nabla^2_{\boldsymbol{xx}}\boldsymbol{\mathcal{L}}\boldsymbol{p}$$
$$s.t.\ \boldsymbol{A}_k\boldsymbol{p} + \boldsymbol{c}_k = 0$$

该问题 KKT 条件

$$\nabla^2_{\boldsymbol{xx}}\boldsymbol{\mathcal{L}}\boldsymbol{p}_k + \nabla f_k - \boldsymbol{A}_k^T\boldsymbol{u}_k = 0$$
$$\boldsymbol{A}_k\boldsymbol{p}_k + \boldsymbol{c}_k = 0$$

QP 问题是凸优化问题, 即通过求该 QP 问题得到的解 $(\boldsymbol{p}_k, \boldsymbol{u}_k)$, 必然满足 KKT 条件, 同时也是 (1) 的解

SQP 算法一般问题用 SQP 算法求解, 如下优化问题

$$min\ f(\boldsymbol{x})$$
$$s.t.\ \boldsymbol{c}_i(\boldsymbol{x}) = 0,\ i \in \mathcal{E}$$
$$\boldsymbol{c}_i(\boldsymbol{x}) \ge 0,\ i \in \mathcal{I}$$

可以通过求解如下一系列 QP 子问题实现

$$\begin{aligned} &\min_{\boldsymbol{p}}\ f_k + \nabla f_k^T\boldsymbol{p} + \frac{1}{2}\boldsymbol{p}^T\nabla^2_{\boldsymbol{xx}}\boldsymbol{\mathcal{L}}\boldsymbol{p} \\ &s.t. \nabla\boldsymbol{c}_i(\boldsymbol{x}_k)^T\boldsymbol{p} + \boldsymbol{c}_i(\boldsymbol{x}_k) = 0,\ i \in \mathcal{E} \\ &\nabla\boldsymbol{c}_i(\boldsymbol{x}_k)^T\boldsymbol{p} + \boldsymbol{c}_i(\boldsymbol{x}_k) \ge 0,\ i \in \mathcal{I} \end{aligned} \tag{18.11}$$

新的迭代点为 $(\boldsymbol{x}_k + \boldsymbol{p}_k, \boldsymbol{\lambda}_{k+1})$,$\boldsymbol{p}_k$ 和 $\boldsymbol{\lambda}_{k+1}$ 是 (18.11) 的解和 Lagrange multiplier.

Algorithm 18.3 (Line Search SQP Algorithm).
Choose parameters $\eta \in (0, 0.5)$, $\tau \in (0, 1)$, and an initial pair (x_0, λ_0);
Evaluate $f_0, \nabla f_0, c_0, A_0$;
If a quasi-Newton approximation is used, choose an initial $n \times n$ symmetric positive definite Hessian approximation B_0, otherwise compute $\nabla^2_{xx}\mathcal{L}_0$;
repeat until a convergence test is satisfied
 Compute p_k by solving (18.11); let $\hat{\lambda}$ be the corresponding multiplier;
 Set $p_\lambda \leftarrow \hat{\lambda} - \lambda_k$;
 Choose μ_k to satisfy (18.36) with $\sigma = 1$;
 Set $\alpha_k \leftarrow 1$;
 while $\phi_1(x_k + \alpha_k p_k; \mu_k) > \phi_1(x_k; \mu_k) + \eta\alpha_k D_1(\phi(x_k; \mu_k)p_k)$
  Reset $\alpha_k \leftarrow \tau_\alpha\alpha_k$ for some $\tau_\alpha \in (0, \tau]$;
 end (while)
 Set $x_{k+1} \leftarrow x_k + \alpha_k p_k$ and $\lambda_{k+1} \leftarrow \lambda_k + \alpha_k p_\lambda$;
 Evaluate $f_{k+1}, \nabla f_{k+1}, c_{k+1}, A_{k+1}$, (and possibly $\nabla^2_{xx}\mathcal{L}_{k+1}$);
 If a quasi-Newton approximation is used, set
  $s_k \leftarrow \alpha_k p_k$ and $y_k \leftarrow \nabla_x\mathcal{L}(x_{k+1}, \lambda_{k+1}) - \nabla_x\mathcal{L}(x_k, \lambda_{k+1})$,
 and obtain B_{k+1} by updating B_k using a quasi-Newton formula;
end (repeat)

$$\mu \ge \frac{\nabla f_k^T p_k + (\sigma/2)p_k^T\nabla^2_{xx}\mathcal{L}_k p_k}{(1-\rho)||c_k||_1}$$
$$\rho \in (0, 1)$$

$$\phi_1(x;\mu) = f(x) + \mu||c(x)||_1$$

$D(\phi_1(x_k;\mu;p_k))$ 表示 ϕ_1 在 p_k 方向的梯度

$$D(\phi_1(x_k;\mu;p_k)) = \triangledown f_k^T p_k - \mu||c_k||_1$$

用 BFGS 拟合 Hessian 矩阵

$$H_{k+1} = (I - p_k s_k y_k^T) H_k (I - p_k y_k s_k^T) + p_k s_k s_k^T$$

9.4.4 内点法

$$\min_{x,s} \ f(x) \tag{19.1a}$$
$$\text{subject to} \quad c_E(x) = 0, \tag{19.1b}$$
$$c_I(x) - s = 0, \tag{19.1c}$$
$$s \geq 0. \tag{19.1d}$$

由上式子 KKT 条件

$$\nabla f(x) - A_E{}^T(x) y - A_I{}^T(x) z = 0, \tag{19.2a}$$
$$Sz - \mu e = 0, \tag{19.2b}$$
$$c_E(x) = 0, \tag{19.2c}$$
$$c_I(x) - s = 0, \tag{19.2d}$$

with $\mu = 0$, together with

$$s \geq 0, \qquad z \geq 0. \tag{19.3}$$

将 KKT 条件转换为非线性系统, 用 Newton 法求解

$$\begin{bmatrix} \nabla^2_{xx}\mathcal{L} & 0 & -A_E{}^T(x) & -A_I{}^T(x) \\ 0 & Z & 0 & S \\ A_E(x) & 0 & 0 & 0 \\ A_I(x) & -I & 0 & 0 \end{bmatrix} \begin{bmatrix} p_x \\ p_s \\ p_y \\ p_z \end{bmatrix} = - \begin{bmatrix} \nabla f(x) - A_E{}^T(x)y - A_I{}^T(x)z \\ Sz - \mu e \\ c_E(x) \\ c_I(x) - s \end{bmatrix}, \tag{19.6}$$

$$\mathcal{L}(x, s, y, z) = f(x) - y^T c_E(x) - z^T (c_I(x) - s). \tag{19.7}$$

当求出 $\boldsymbol{p} = (p_x, p_s, p_y, p_z)$ 后, 求 (x^+, s^+, y^+, z^+),$\tau \in (0,1)$, 一般取 $\tau = 0.995$

$$x^+ = x + \alpha_s^{\max} p_x, \quad s^+ = s + \alpha_s^{\max} p_s, \tag{19.8a}$$
$$y^+ = y + \alpha_z^{\max} p_y, \quad z^+ = z + \alpha_z^{\max} p_z, \tag{19.8b}$$

where

$$\alpha_s^{\max} = \max\{\alpha \in (0,1] : s + \alpha p_s \geq (1-\tau)s\}, \tag{19.9a}$$
$$\alpha_z^{\max} = \max\{\alpha \in (0,1] : z + \alpha p_z \geq (1-\tau)z\}, \tag{19.9b}$$

以下给出一种基础的内点法形式, 如果想要扩展到非凸非线性情况, 需要改进.

$$E(x, s, y, z; \mu) = \max\left\{\|\nabla f(x) - A_E(x)^T y - A_I(x)^T z\|, \|Sz - \mu e\|, \|c_E(x)\|, \|c_I(x) - s\|\right\}, \tag{19.10}$$

Algorithm 19.1 (Basic Interior-Point Algorithm).

Choose x_0 and $s_0 > 0$, and compute initial values for the multipliers y_0 and $z_0 > 0$. Select an initial barrier parameter $\mu_0 > 0$ and parameters $\sigma, \tau \in (0,1)$. Set $k \leftarrow 0$.

repeat until a stopping test for the nonlinear program (19.1) is satisfied
  repeat until $E(x_k, s_k, y_k, z_k; \mu_k) \leq \mu_k$
    Solve (19.6) to obtain the search direction $p = (p_x, p_s, p_y, p_z)$;
    Compute $\alpha_s^{\max}, \alpha_z^{\max}$ using (19.9);
    Compute $(x_{k+1}, s_{k+1}, y_{k+1}, z_{k+1})$ using (19.8);
    Set $\mu_{k+1} \leftarrow \mu_k$ and $k \leftarrow k+1$;
  end
  Choose $\mu_k \in (0, \sigma\mu_k)$;
end

改进基础内点法, 首先 (19.6) 可以被重写为对称矩阵形式

$$\begin{bmatrix} \nabla^2_{xx}\mathcal{L} & 0 & A_E{}^T(x) & A_I{}^T(x) \\ 0 & \Sigma & 0 & -I \\ A_E(x) & 0 & 0 & 0 \\ A_I(x) & -I & 0 & 0 \end{bmatrix} \begin{bmatrix} p_x \\ p_s \\ -p_y \\ -p_z \end{bmatrix} = - \begin{bmatrix} \nabla f(x) - A_E{}^T(x)y - A_I{}^T(x)z \\ z - \mu S^{-1}e \\ c_E(x) \\ c_I(x) - s \end{bmatrix}, \tag{19.12}$$

where

$$\Sigma = S^{-1}Z. \tag{19.13}$$

We must also guard against singularity of the primal-dual matrix caused by the rank deficiency of A_E (the matrix $[A_I \ -I]$ always has full rank). We do so by including a regularization parameter $\gamma \geq 0$, in addition to the modification term δI, and work with the modified primal-dual matrix

$$\begin{bmatrix} \nabla^2_{xx}\mathcal{L} + \delta I & 0 & A_E(x)^T & A_I(x)^T \\ 0 & \Sigma & 0 & -I \\ A_E(x) & 0 & -\gamma I & 0 \\ A_I(x) & -I & 0 & 0 \end{bmatrix}. \tag{19.25}$$

A procedure for selecting γ and δ is given in Algorithm B.1 in Appendix B. It is invoked at every iteration of the interior-point method to enforce the inertia condition (19.24) and to guarantee nonsingularity. Other matrix modifications to ensure positive definiteness have been discussed in Chapter 3 in the context of unconstrained minimization.

步长参数选择, 定义 merit 函数

$$\phi_\nu(x, s) = f(x) - \mu \sum_{i=1}^m \log s_i + \nu\|c_E(x)\| + \nu\|c_I(x) - s\|, \tag{19.26}$$

$$\alpha_s \in (0, \alpha_s^{\max}], \qquad \alpha_z \in (0, \alpha_z^{\max}], \tag{19.27}$$

providing sufficient decrease of the merit function or ensuring acceptability by the filter. The new iterate is then defined as

$$x^+ = x + \alpha_s p_x, \quad s^+ = s + \alpha_s p_s, \tag{19.28a}$$
$$y^+ = y + \alpha_z p_y, \quad z^+ = z + \alpha_z p_z. \tag{19.28b}$$

当 $\boldsymbol{p}$ 已解算, 最大步长通过 (19.9) 获得, 此时通过 backtracking line search 算步长.

(19.12) 中有 Lagrangian 的二阶导, 需要用 quasi-Newton 方式逼近 $\boldsymbol{B}$, 可以采用 BFGS,limited-memory BFGS,SR1 方式. 此处,$(\triangle x, \triangle l)$ 取代 (s, y).

$$\Delta l = \nabla_x \mathcal{L}(x^+, s^+, y^+, z^+) - \nabla_x \mathcal{L}(x, s^+, y^+, z^+),$$
$$\Delta x = x^+ - x.$$
$$B = \xi I + WMW^T, \tag{19.29}$$

第二种表达形式

$$\min_{x,s} \ f(x) \tag{19.1a}$$
$$\text{subject to} \quad c_E(x) = 0, \tag{19.1b}$$
$$c_I(x) - s = 0, \tag{19.1c}$$
$$s \geq 0. \tag{19.1d}$$

9.4.5 ADMM 算法

将 ML 问题视作优化问题, 其目标函数往往可以写成 loss function+regulation 形式,ADMM 解决 regulation 为 L1 范数的时的凸优化问题.

10 稠密矩阵数值算法

10.1 矩阵分解

10.1.1 SVD 分解

10.1.2 LU 分解

10.1.3 QR 分解

10.1.4 Cholesky 分解

10.2 Linear Systems

$\boldsymbol{Ax} = \boldsymbol{b}$

10.2.1 Direct 方法

LU 分解 Gaussian 消元过程将 A 分解成下三角矩阵 L 和上三角矩阵 U, 即 LU 分解.

$$\boldsymbol{A} = \boldsymbol{LU}$$

$$\boldsymbol{Ly} = \boldsymbol{b}$$

$$\boldsymbol{Ux} = \boldsymbol{y}$$

Cholesky 分解如果 $\boldsymbol{A}$ 是对称正定矩阵 (symmetric positive definite, SPD), 即 $\boldsymbol{A} = \boldsymbol{A}^T$, 且 $\boldsymbol{x}^T\boldsymbol{Ax}$,$\forall \boldsymbol{x} \neq 0$.

10.2.2 Iterative 方法

10.3 Least Squres Systems

10.3.1 Direct 方法

10.3.2 Iterative 方法

10.4 Eigenvalue

11 稀疏矩阵数值算法

12 偏微分方程数值解

12.1 有限差分法

12.2 有限元法

13 插值与拟合