第四章 数值计算

机器学习中的数值计算：

- 通过迭代过程更新解的估计值来解决数学问题的算法
- 不是通过解析过程推导出公式来提供正确解的方法

常见操作：

- 优化（找到最小化或最大化函数值的参数）
- 线性方程组的求解

对数字计算机来说实数无法在有限内存下精确表示，因此仅仅是计算涉及实数的函数也是困难的。

第四章 数值计算

上溢和下溢

下溢：

- 接近零的数被四舍五入为零
- 不许多函数在其参数为零而不是一个很小的正数时才会表现出质的不同（被零除或者去零的对数）

上溢：

- 大量级的数被近似成∞或者$-\infty$时发生上溢
- 进一步的运算通常会导致这些无限值变为非数字

第四章 数值计算

上溢和下溢

softmax函数：

$$\mathrm{softmax}(\boldsymbol{x})_i = \frac{\exp(x_i)}{\sum_{j=1}^{n} \exp(x_j)}$$

假设所有的x_i都是是极小的负数（下溢）或者极大的正数（上溢）都会导致函数结果未定义

解决：

使用$\mathrm{softmax}(\boldsymbol{z})$，其中$\boldsymbol{z} = \boldsymbol{x} - max_i x_i$

第四章 数值计算

病态条件

条件数表征函数相对于输入的微小变化 而变化的快慢程度。

对于$f(\boldsymbol{x}) = \boldsymbol{A}^{-1}\boldsymbol{x}$，当$\boldsymbol{A} \in \mathbb{R}^{n*n}$具有特征值分解时，条件数为：

$$\max_{i,j} \left| \frac{\lambda_i}{\lambda_j} \right|.$$

- 最大特征值与最小特征值的模之比
- 该数很大时，矩阵求逆对输入的误差特别敏感
- 病态条件的矩阵也会放大预先存在的误差

第四章 数值计算

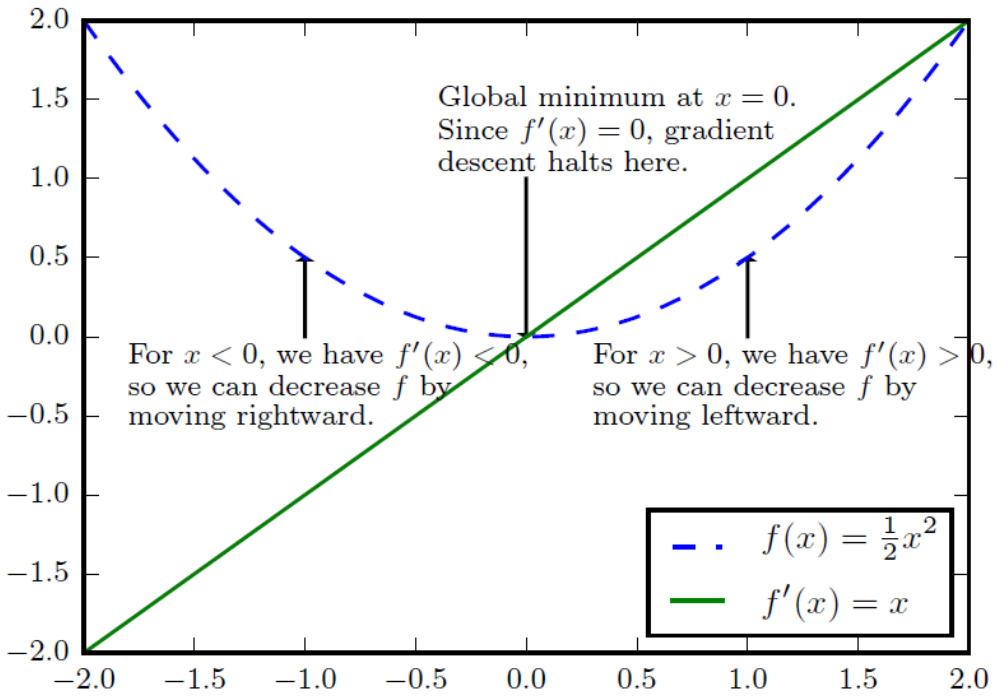
基于梯度的优化方法

目标：最小化$f(\boldsymbol{x})$：

- 目标函数、准则（ criterion ）、代价函数、损失函数、误差函数
- 记：$x^* = arg\min f(\boldsymbol{x})$

方法：梯度下降：

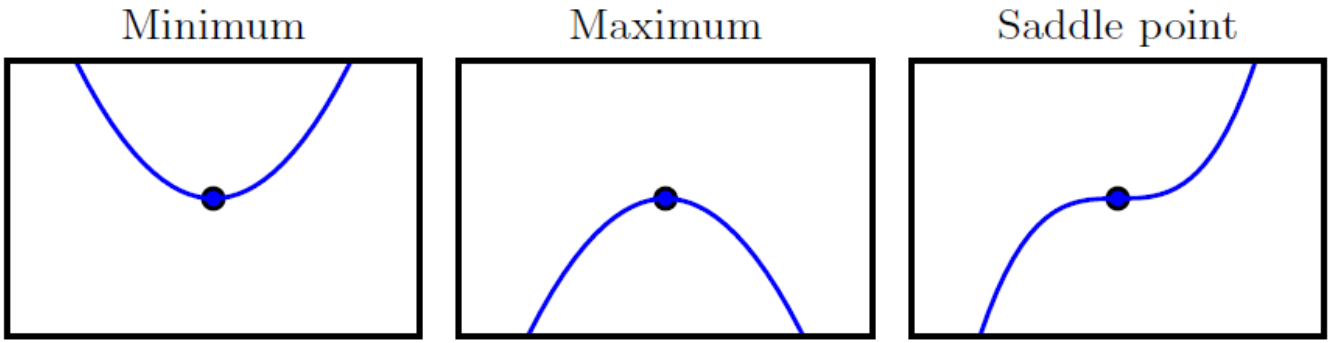
- 将$\boldsymbol{x}$往导数的反方向移动一小步来减小$f(\boldsymbol{x})$
- 线性方程组的求解



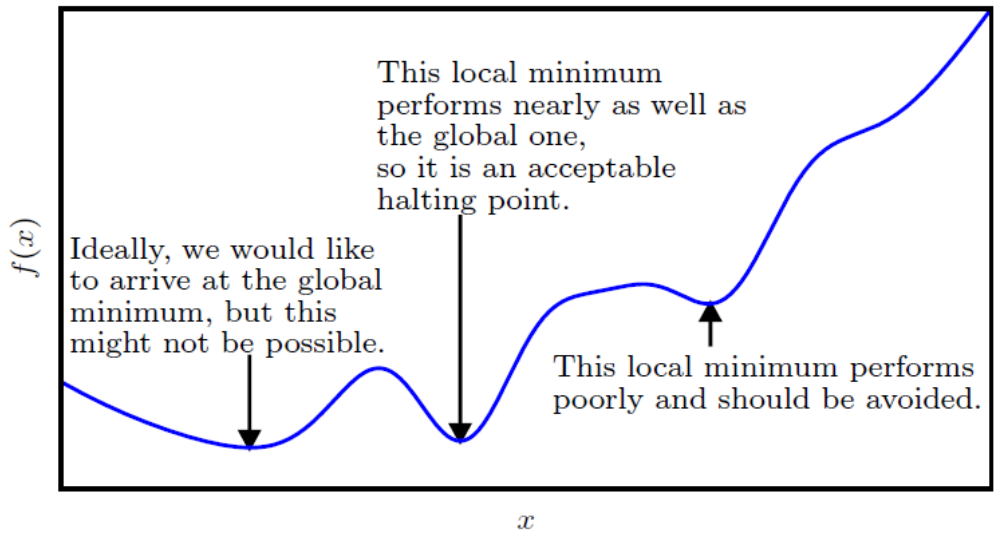
第四章 数值计算

基于梯度的优化方法

临界点：



深度学习背景下通常寻找使f非常小的点，但不一定最小



第四章 数值计算

基于梯度的优化方法

偏导数和梯度：

- f 的导数是包含所有偏导数的向量，记为$\nabla_{\boldsymbol{x}} f(\boldsymbol{x})$
- 梯度的第i个元素是f关于x_i的偏导数
- 临界点是梯度中所有元素都为零的点

f在$\boldsymbol{u}$方向上的方向导数：

- $f(\boldsymbol{x}+\alpha \boldsymbol{u})$关于$\alpha$的导数（在$\alpha=0$时取得）

优化：

$$\frac{\partial}{\partial \alpha} f(\boldsymbol{x}+\alpha \boldsymbol{u})=\boldsymbol{u}^{\top} \nabla_{x} f(\boldsymbol{x})$$

$$\min_{u, \boldsymbol{u}^{\top} u=1} \boldsymbol{u}^{\top} \nabla_{x} f(\boldsymbol{x})=\min_{u, \boldsymbol{u}^{\top} u=1}\|\boldsymbol{u}\|_{2}\left\|\nabla_{x} f(\boldsymbol{x})\right\|_{2} \cos \theta \Rightarrow \min_{u} \cos \theta$$

这在$\boldsymbol{u}$与梯度方向相反时取得最小，这个方向优化成为**最速下降法**

$$\boldsymbol{x}^{\prime}=\boldsymbol{x}-\epsilon \nabla_{x} f(\boldsymbol{x})$$

ϵ是学习率，一个确定步长大小的正标量

第四章 数值计算

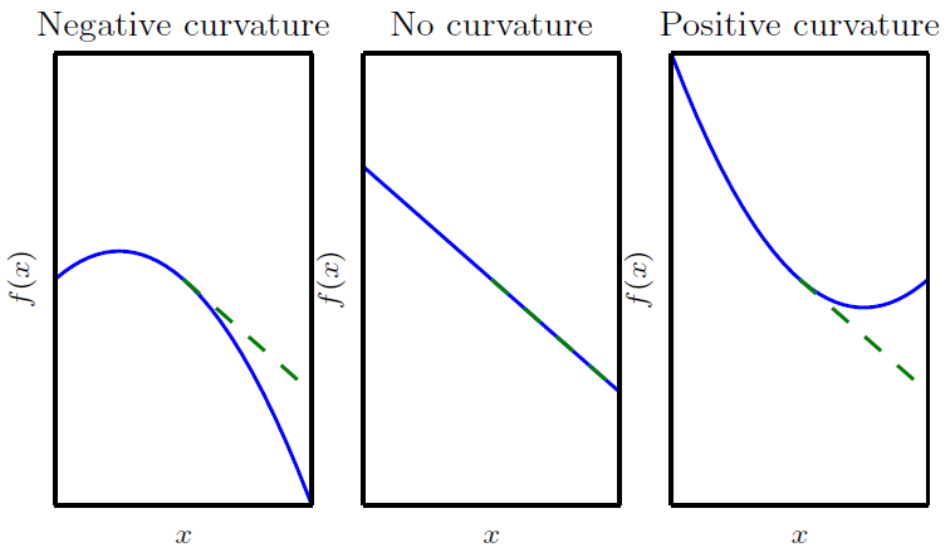
基于梯度的优化方法

Jacobian矩阵

- 一阶偏导数以一定方式排列成的矩阵
- $\boldsymbol{J} \in \mathbb{R}^{\boldsymbol{n*m}}$定义为$J_{i,j} = \frac{\partial}{\partial x_j} f(x)_i$

曲率：

- 二阶导数是对曲率的衡量



第四章 数值计算

基于梯度的优化方法

Hessian矩阵

- 二阶偏导数以一定方式排列成的矩阵
- $\boldsymbol{H}(f)(\boldsymbol{x})$定义为$\boldsymbol{H}(f)(\boldsymbol{x})_{i,j} = \frac{\partial^2}{\partial x_i \partial x_j} f(\boldsymbol{x})$
- Hessian矩阵等价于梯度的Jacobian矩阵

$$\frac{\partial^2}{\partial x_i \partial x_j} f(\boldsymbol{x}) = \frac{\partial^2}{\partial x_j \partial x_i} f(\boldsymbol{x})$$

Hessian 矩阵是实对称的，所以f在特定的$\boldsymbol{d}$方向上的二阶导数可以写作$\boldsymbol{d}^T \boldsymbol{H} \boldsymbol{d}$

- $\boldsymbol{d}$是特征向量时，二阶导是对应特征值
- 其他方向，二阶导是所有特征值的加权平均

第四章 数值计算

基于梯度的优化方法

二阶导数：

- 预期一个梯度下降步骤能表现得多好

$$f(\boldsymbol{x}) \approx f(\boldsymbol{x}^{(0)}) + (\boldsymbol{x} - \boldsymbol{x}^{(0)})^\top \boldsymbol{g} + \frac{1}{2}(\boldsymbol{x} - \boldsymbol{x}^{(0)})^\top \boldsymbol{H}(\boldsymbol{x} - \boldsymbol{x}^{(0)})$$

- $\boldsymbol{g}$是梯度， $\boldsymbol{H}$是$x^{(0)}$点的Hessian
- 学习率为ϵ，则
- $f(\boldsymbol{x}^{(0)} - \epsilon \boldsymbol{g}) \approx f(\boldsymbol{x}^{(0)}) - \epsilon \boldsymbol{g}^\top \boldsymbol{g} + \frac{1}{2}\epsilon^2 \boldsymbol{g}^\top \boldsymbol{H} \boldsymbol{g}.$
- 函数的原始值、函数斜率导致的预期改善、函数曲率导致的校正
- $\boldsymbol{g}^T \boldsymbol{H} \boldsymbol{g}$为零或负时，增加$\epsilon$将永远使$f$下降
- $\boldsymbol{g}^T \boldsymbol{H} \boldsymbol{g}$为正时，$\epsilon^* = \frac{\boldsymbol{g}^\top \boldsymbol{g}}{\boldsymbol{g}^\top \boldsymbol{H} \boldsymbol{g}}$
- 最坏情况是$\boldsymbol{g}$与$\boldsymbol{H}$最大特征值对应的特征向量对齐，此时最优步长为$\frac{1}{\lambda_{\max}}$，此时Hessian 的特征值决定学习率的量级

第四章 数值计算

基于梯度的优化方法

二阶导数：

- 确定一个临界点类型
- 一维

  $f'(x) = 0$ 且 $f''(x) > 0$ 时 $\Rightarrow$ $\boldsymbol{x}$ 是一个局部极小点

  $f'(x) = 0$ 且 $f''(x) < 0$ 时 $\Rightarrow$ $\boldsymbol{x}$ 是一个局部极大点

- 多维

  当 Hessian 是正定的 $\Rightarrow$ 则该临界点是局部极小点

  当 Hessian 是负定的 $\Rightarrow$ 这个点就是局部极大点

第四章 数值计算

基于梯度的优化方法

用Hessian 矩阵的信息来指导梯度下降：牛顿法：

$$f(\boldsymbol{x}) \approx f(\boldsymbol{x}^{(0)}) + (\boldsymbol{x} - \boldsymbol{x}^{(0)})^{\top} \nabla_{\boldsymbol{x}} f(\boldsymbol{x}^{(0)}) + \frac{1}{2} (\boldsymbol{x} - \boldsymbol{x}^{(0)})^{\top} \boldsymbol{H}(f)(\boldsymbol{x}^{(0)})(\boldsymbol{x} - \boldsymbol{x}^{(0)})$$

求导并令其一阶导为零，计算临界点：

$$\boldsymbol{x}^{*} = \boldsymbol{x}^{(0)} - \boldsymbol{H}(f)(\boldsymbol{x}^{(0)})^{-1} \nabla_{\boldsymbol{x}} f(\boldsymbol{x}^{(0)})$$

- f是正定二次函数时，可以直接计算最优解
- f在局部近似为正定二次，可以多次迭代寻求最优解

第四章 数值计算

约束优化

x在某些集合$\mathbb{S}$中找到$f(x)$最大值或者最小值：

- 集合$\mathbb{S}$内的点x被称为可行点

Karush–Kuhn–Tucker（KKT）方法：

- 针对约束优化非常通用的解决方案
- 必要条件
  - 广义Lagrange 函数梯度为零
  - 所有关于x和KKT 乘子的约束都满足
  - 不等式约束显示的“互补松弛性”：$\alpha \odot h(x) = 0$

第四章 数值计算

约束优化

广义Lagrange 函数：

- 集合$\mathbb{S}$的约束：等式约束$g^{(i)}(\boldsymbol{x})$和不等式约束$h^{(j)}(\boldsymbol{x})$

$$\mathbb{S} = \{\boldsymbol{x} | \forall i, g^{(i)}(\boldsymbol{x}) = 0 \text{ and} \forall j, h^{(j)}(\boldsymbol{x}) \leq 0\}$$

- KKT 乘子λ_i和α_j

$$L(x, \boldsymbol{\lambda}, \boldsymbol{\alpha}) = f(\boldsymbol{x}) + \sum_i \lambda_i g^{(i)}(\boldsymbol{x}) + \sum_j \alpha_j h^{(j)}(\boldsymbol{x})$$

- 优化无约束的广义Lagrangian（$\mathbb{S}$内有可行点且$f \neq \infty$）

$$\min_{\boldsymbol{x} \in \mathbb{S}} f(\boldsymbol{x}) \leftrightarrow \min_{\boldsymbol{x}} \max_{\boldsymbol{\lambda}} \max_{\boldsymbol{\alpha}, \boldsymbol{\alpha} \geq 0} L(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\alpha})$$

- 约束最大化：构造$-f(x)$ 的广义Lagrange 函数

$$\min_{\boldsymbol{x}} \max_{\boldsymbol{\lambda}} \max_{\boldsymbol{\alpha}, \boldsymbol{\alpha} \geq 0} -f(\boldsymbol{x}) + \sum_i \lambda_i g^{(i)}(\boldsymbol{x}) + \sum_j \alpha_j h^{(j)}(\boldsymbol{x})$$

$$\Rightarrow \max_{\boldsymbol{x}} \min_{\boldsymbol{\lambda}} \min_{\boldsymbol{\alpha}, \boldsymbol{\alpha} \geq 0} f(\boldsymbol{x}) + \sum_i \lambda_i g^{(i)}(\boldsymbol{x}) - \sum_j \alpha_j h^{(j)}(\boldsymbol{x})$$

第四章 数值计算

实例：线性最小二乘

目标函数：

$$f(x) = \frac{1}{2}\|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|_2^2$$

计算梯度：

$$\nabla_{\boldsymbol{x}} f(x) = \boldsymbol{A}^\top(\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}) = \boldsymbol{A}^\top\boldsymbol{A}\boldsymbol{x} - \boldsymbol{A}^\top\boldsymbol{b}.$$

算法：

算法 4.1 从任意点 $\boldsymbol{x}$ 开始，使用梯度下降关于 $\boldsymbol{x}$ 最小化 $f(x) = \frac{1}{2}||\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}||_2^2$ 的算法。

将步长 (ϵ) 和容差 (δ) 设为小的正数。

while $||\boldsymbol{A}^\top\boldsymbol{A}\boldsymbol{x} - \boldsymbol{A}^\top\boldsymbol{b}||_2 > \delta$ **do**

$\boldsymbol{x} \leftarrow \boldsymbol{x} - \epsilon\left(\boldsymbol{A}^\top\boldsymbol{A}\boldsymbol{x} - \boldsymbol{A}^\top\boldsymbol{b}\right)$

end while

第四章 数值计算

实例：线性最小二乘

目标函数（牛顿法）：

$$f(x) = \frac{1}{2}\|Ax - b\|_2^2 \qquad x^\top x \le 1$$

引入Lagrangian：

$$L(x, \lambda) = f(x) + \lambda(x^\top x - 1)$$

目标：$\min_{x} \max_{\lambda, \lambda \ge 0} L(x, \lambda)$

微分，且λ的选择必须使结果服从约束，于是，

$$\frac{\partial}{\partial x} L(x, \lambda) = A^\top A x - A^\top b + 2\lambda x = 0$$

$$\frac{\partial}{\partial \lambda} L(x, \lambda) = x^\top x - 1 = 0$$

$$\Rightarrow \quad x = (A^\top A + 2\lambda I)^{-1} A^\top b$$