梯度下降法

- **梯度下降法**（gradient descent）或**最速下降法**（steepest descent）：是求解*无约束最优化问题*常用方法；是迭代算法，每一步求解目标函数的*梯度向量*；
- 设$f(x)$是$\mathbb{R}^n$上具有*一阶连续偏导*的函数，无约束最优化问题是（x^*表示极小值点）：

$$\min_{x \in \mathbb{R}^n} f(x) \tag{1}$$

- **算法思路**：由于*负梯度方向*是使函数值下降最快的方向，在迭代的每一步以负梯度方向更新x的值，从而达到减少函数值的目的（极小化）；
- 由于$f(x)$一阶连续可导，在第k次迭代值为$x^{(k)}$，可将$f(x)$在$x^{(k)}$附近进行*一阶泰勒展开*:

$$f(x) = f(x^{(k)}) + g_k^T (x - x^{(k)}) \tag{2}$$

其中$g_k = g(x^{(k)}) = \nabla f(x^{(k)})$表示$f(x)$在$x^{(k)}$的梯度;

- 求出第$k+1$次的迭代值:

$$x^{(k+1)} \leftarrow x^{(k)} + \lambda_k p_k \tag{3}$$

其中p_k是搜索方向，取负梯度方向$p_k = -\nabla f(x^{(k)})$；λ_k是步长，由一维搜索确定，即λ_k使得$f(x^{(k)} + \lambda_k p_k) = \min_{\lambda \geq 0} f(x^{(k)} + \lambda p_k)$成立;

- **梯度下降算法**:
  1. **输入**目标函数$f(x)$，梯度函数$g(x) = \nabla f(x)$，计算精度ϵ;
  2. 初始化$x^{(0)} \in \mathbb{R}^n$的值，$k = 0$;
  3. 计算$f(x^{(k)})$;
  4. 计算梯度$g_k = g(x^{(k)}) = \nabla f(x^{(k)})$，当$\|g_k\| < \epsilon$时停止迭代，令$x^* = x^{(k)}$；否则令$p_k = -g_k$，求$\lambda_k$使$f(x^{(k)} + \lambda_k p_k) = \min_{\lambda \geq 0} f(x^{(k)} + \lambda p_k)$;

5. 更新$x^{(k+1)} \leftarrow x^{(k)} + \lambda_k p_k$，计算$f(x^{(k+1)})$，当$\|f(x^{(k+1)}) - f(x^{(k)})\| < \epsilon$或$\|x^{(k+1)} - x^{(k)}\| < \epsilon$时停止迭代，令$x^* = x^{(k)}$；
6. 否则$k = k + 1$，回到步骤（4）；
7. **输出**极小点x^*；

- 当目标函数是*凸函数*时，梯度下降的解是*全局最优解*；一般情况下不保证是全局最优解；
- 梯度下降法的收敛速度未必很快；