最优化方法

在模式识别的学习和训练过程中，需要根据训练样本来确定一组与分类器模型相关的参数。学习过程往往首先定义某个准则函数，用以描述参数的“适合性”，然后寻找一组“适合性”最大的参数作为学习的结果，也就是说将模式识别的学习问题转化为针对某个准则函数的优化问题。

假设准则函数 $f(\mathbf{x})$ 针对 $\mathbf{x}$ 的每个分量都可微，下面介绍几种常用的函数优化方法。

I. 直接法求极值

根据数学分析的知识我们知道，m 维矢量 $\mathbf{x}^*$ 是函数 $f(\mathbf{x})$ 极值点的必要条件是：$\partial f / \partial x_i^* = 0$，对任意的 $1 \le i \le m$ 成立。如果把所有的偏导数写成矢量的形式，则函数 $f(\mathbf{x})$ 的极值点可以通过求解矢量方程得到：

$$\frac{\partial f(\mathbf{x})}{\partial \mathbf{x}} = \begin{pmatrix} \partial f / \partial x_1 \\ \vdots \\ \partial f / \partial x_m \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ 0 \end{pmatrix} = \mathbf{0} \qquad (1)$$

上述方程的解可能是极大值点，可能是极小值点，也可能不是极值点，具体情况还需要根据二阶导数来判断。如果我们希望求取的是 $f(\mathbf{x})$ 的最大值或最小值点，可以通过比较所有的极大值或极小值点得到。

对于简单的纯凸或纯凹函数（如二次函数），由于只存在唯一的极值点，极值点即为最大值或最小值点，因此可以直接通过求解矢量方程（1）得到 $f(\mathbf{x})$ 的优化解。

II. 梯度法

对于复杂的函数来说，直接求解方程（1）得到优化函数的极值点往往是很困难的。在这种情况下，可以考虑采用迭代的方法从某个初始值开始逐渐逼近极值点，梯度法就是一种迭代求解函数极值点的方法。

考虑多元函数 $f(\mathbf{x})$ 在点 $\mathbf{x}$ 附近的一阶泰勒级数展开式：

$$f(\mathbf{x}+\Delta\mathbf{x}) = f(\mathbf{x}) + \sum_{i=1}^{m} \frac{\partial f}{\partial x_i} \Delta x_i + r(\mathbf{x}, \Delta\mathbf{x}) \qquad (2)$$

其中 $\Delta\mathbf{x}$ 是增量矢量，Δx_i 为其第 i 维元素，$r(\mathbf{x}, \Delta\mathbf{x})$ 为展开式的余项。注意到第二项的求和式，实际上是 $f(\mathbf{x})$ 关于 $\mathbf{x}$ 的梯度矢量与 $\Delta\mathbf{x}$ 之间的内积，同时当 $\|\Delta\mathbf{x}\|$ 很小时忽略余项 $r(\mathbf{x}, \Delta\mathbf{x})$，可以得到一阶近似展开式：

$$f(\mathbf{x}+\Delta\mathbf{x}) \approx f(\mathbf{x}) + (\nabla f(\mathbf{x}))^t \Delta\mathbf{x} = f(\mathbf{x}) + \left(\frac{\partial f}{\partial \mathbf{x}}\right)^t \Delta\mathbf{x} \qquad (3)$$

如果我们要求取 $f(\mathbf{x})$ 的极小值 $\mathbf{x}^*$，可以从某个初始点 $\mathbf{x}_0$ 开始搜索，每次增加一个增量 $\Delta\mathbf{x}$。虽然不能保证 $\mathbf{x}_0 + \Delta\mathbf{x}$ 直接到达极小值点，但如果能够保证每次迭代过程中函数值逐渐减小，$f(\mathbf{x}+\Delta\mathbf{x}) < f(\mathbf{x})$，那么经过一定的迭代步数之后，就能够逐渐地接近 $\mathbf{x}^*$，这是一个

函数值逐渐下降的过程。更进一步，我们总是希望函数值下降的过程越快越好，这样可以用尽量少的迭代次数，达到对 $\mathbf{x}^*$ 更高精度的逼近，因此这种方法也被称作“最速下降法”。

根据公式（3），要使得函数值下降得最快，就是要寻找一个增量矢量 $\Delta\mathbf{x}$ 使得 $\left(\nabla f(\mathbf{x})\right)^t \Delta\mathbf{x}$ 最小。注意到（3）式只是在点 $\mathbf{x}$ 附近的一阶近似，当 $\|\Delta\mathbf{x}\|$ 过大时，近似的精度会很差，因此不能直接寻找增量矢量，而是应该寻找使得函数值下降最快的方向。也就是在约束 $\|\Delta\mathbf{x}\|=1$ 的条件下，寻找使得 $\left(\nabla f(\mathbf{x})\right)^t \Delta\mathbf{x}$ 最小的增量矢量。找到最速下降的方向之后，再来确定此方向上合适的增量矢量长度。

根据 Cauchy–Schwarz 不等式，两个矢量内积的绝对值小于等于两个矢量长度的乘积，因此：

$$\left|\left(\nabla f(\mathbf{x})\right)^t \Delta\mathbf{x}\right| \le \|\nabla f(\mathbf{x})\|\|\Delta\mathbf{x}\|$$

$$\left(\nabla f(\mathbf{x})\right)^t \Delta\mathbf{x} \ge -\|\nabla f(\mathbf{x})\|\|\Delta\mathbf{x}\| = -\|\nabla f(\mathbf{x})\| \qquad (4)$$

如果令：$\Delta\mathbf{x} = -\dfrac{\nabla f(\mathbf{x})}{\|\nabla f(\mathbf{x})\|}$

$$\begin{aligned}\left(\nabla f(\mathbf{x})\right)^t \Delta\mathbf{x} &= \left(\nabla f(\mathbf{x})\right)^t \left[-\frac{\nabla f(\mathbf{x})}{\|\nabla f(\mathbf{x})\|}\right] \\ &= -\frac{\left(\nabla f(\mathbf{x})\right)^t \nabla f(\mathbf{x})}{\|\nabla f(\mathbf{x})\|} \\ &= -\frac{\|\nabla f(\mathbf{x})\|^2}{\|\nabla f(\mathbf{x})\|} = -\|\nabla f(\mathbf{x})\|\end{aligned}$$

由此我们知道，当 $\Delta\mathbf{x}$ 为负的梯度方向时，不等式（4）中的等号成立，也就是说 $\left(\nabla f(\mathbf{x})\right)^t \Delta\mathbf{x}$ 取得最小值。因此，增量矢量的方向为负的梯度方向时，函数值下降得最快。最速下降法中每一轮应该按照下式进行迭代：

$$\mathbf{x} = \mathbf{x} + \Delta\mathbf{x} = \mathbf{x} - \eta\nabla f(\mathbf{x}) \qquad (5)$$

其中参数 η 控制增量矢量的长度，在模式识别的算法中一般被称作“学习率”。与此类似，如果优化问题需要寻找的是极大值点，每次迭代中增量矢量应该沿着正的梯度方向。

梯度下降算法的过程是从一个随机的初始点 $\mathbf{x}_0$ 开始，每一轮迭代中计算当前点处的梯度矢量，然后根据公式（5）修正当前的优化点 $\mathbf{x}$。由于极值点处的梯度是 0 矢量，因此算法的收敛条件是判断当前点处梯度矢量的长度是否足够小，当达到一定的收敛精度后可以停止迭代。

梯度下降算法

初始化：$\mathbf{x}_0$，η，θ，$i=0$

do

计算当前点 $\mathbf{x}_i$ 的梯度矢量：$\nabla f(\mathbf{x})\big|_{\mathbf{x}=\mathbf{x}_i}$；

更新优化解：$\mathbf{x}_{i+1} = \mathbf{x}_i - \eta\nabla f(\mathbf{x})\big|_{\mathbf{x}=\mathbf{x}_i}$；

$i = i+1$

until $\left\|\eta\nabla f\left(\mathbf{x}\right)\big|_{\mathbf{x}=\mathbf{x}_i}\right\| < \theta$；

输出优化解 $\mathbf{x}_i$；

参数 θ 为收敛精度，值越小，输出的解越接近于极小值点。梯度下降算法的优点是：算法简单，只要能够计算任意一点的梯度矢量就可以进行迭代优化；在设置合适的学习率 η 的条件下，算法具有收敛性，能够收敛于一个极小值点。

同样，梯度下降算法也存在着自身的缺点。首先是收敛速度慢，特别是在一些梯度值较小的区域表现得尤为明显；收敛性依赖于适合的学习率 η 的设置，而与初始点 $\mathbf{x}_0$ 的选择无关，但对于一个具体问题来说还没有能够直接确定 η 的方法，一般需要进行一定的尝试；梯度法只能够保证收敛于一个极值点，而无法一次计算出所有的极值点，具体收敛于哪一个极值点决定于初始点 $\mathbf{x}_0$；同时，算法收敛的极值点不能保证是优化函数的最小值点，往往需要进行多次尝试，从得到的多个极值点中找出一个最小值点，但由于尝试的次数有限，因此也不能保证找到优化函数的最小值点。

III. 牛顿法

采用梯度法对一个复杂的函数进行优化，迭代的收敛速度往往很慢。这主要是由于，梯度法是利用一阶泰勒级数展开式在 $\mathbf{x}$ 附近用一个线性函数来近似优化 $f\left(\mathbf{x}\right)$，当 $f\left(\mathbf{x}\right)$ 是一个复杂的非线性函数时，近似的精度很低。二阶技术就是用二次函数来近似优化函数，降低近似误差，从而达到提高优化迭代效率的目的。下面，首先来看一下函数的二阶泰勒级数展开式：

$$f\left(\mathbf{x}+\Delta\mathbf{x}\right) \approx f\left(\mathbf{x}\right)+\left(\frac{\partial f}{\partial \mathbf{x}}\right)^t \Delta\mathbf{x}+\frac{1}{2}\Delta\mathbf{x}^t\mathbf{H}\Delta\mathbf{x} \qquad (6)$$

其中 $\mathbf{H}$ 是 f 的二阶导数海森矩阵：

$$\mathbf{H}=\begin{bmatrix} \frac{\partial^2 f}{\partial x_1 \partial x_1} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_m} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2 \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_2 \partial x_m} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_m \partial x_1} & \frac{\partial^2 f}{\partial x_m \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_m \partial x_m} \end{bmatrix}$$

为了寻找使得 $f\left(\mathbf{x}+\Delta\mathbf{x}\right)$ 最小的权值增量 $\Delta\mathbf{x}$，（6）式对 $\Delta\mathbf{x}$ 微分求取极值点，同时考虑到 $\mathbf{H}$ 是对称矩阵：

$$\frac{\partial f}{\partial \mathbf{x}}+\mathbf{H}\Delta\mathbf{x}=0\ ,\quad \Delta\mathbf{x}=-\mathbf{H}^{-1}\left(\frac{\partial f}{\partial \mathbf{x}}\right) \qquad (7)$$

这样，我们就得到了在二阶泰勒级数近似条件下的最优权值增量，此方法一般被称为牛顿法。

牛顿法虽然形式上很简单，但在实际问题中往往无法直接应用。首先，当矢量 $\mathbf{x}$ 的维数 m 很大时，$m\times m$ 的海森矩阵 $\mathbf{H}$ 无论是计算、存储还是求逆的复杂度都很高；更严重的问题是函数 f 并不是一个二次函数，而牛顿法是建立在二阶近似基础之上的，这就导致了直接使用牛顿法并不能保证算法的收敛。牛顿法虽然无法直接使用，但受此启发人们提出了多种近

似的二阶优化技术。

IV. 拟牛顿法

在牛顿法中存在着计算函数 $f(\mathbf{x})$ 的二阶导数矩阵 $\mathbf{H}$ 的困难，拟牛顿法利用函数的一阶导数（梯度）来近似递推矩阵 $\mathbf{H}$ 。

类似于公式（6），函数在 $\mathbf{x}_{k+1}$ 附近的二阶泰勒级数展开为：

$$f(\mathbf{x}) \approx f(\mathbf{x}_{k+1}) + \mathbf{g}_{k+1}^t(\mathbf{x}-\mathbf{x}_{k+1}) + \frac{1}{2}(\mathbf{x}-\mathbf{x}_{k+1})^t \mathbf{H}_{k+1}(\mathbf{x}-\mathbf{x}_{k+1}) \tag{8}$$

其中 $\mathbf{g}_{k+1} = (\partial f/\partial \mathbf{x})_{\mathbf{x}=\mathbf{x}_{k+1}}$ 是函数 $f(\mathbf{x})$ 在 $\mathbf{x}_{k+1}$ 处的梯度，$\mathbf{H}_{k+1}$ 是函数在 $\mathbf{x}_{k+1}$ 处的海森矩阵。

（8）式两边对 $\mathbf{x}$ 求导：

$$\mathbf{g}(\mathbf{x}) \approx \mathbf{g}_{k+1} + \mathbf{H}_{k+1}(\mathbf{x}-\mathbf{x}_{k+1}) \tag{9}$$

将 $\mathbf{x}=\mathbf{x}_k$ 代入，为了表示方便，令 $\mathbf{s}_k = \mathbf{x}_{k+1}-\mathbf{x}_k$， $\mathbf{y}_k = \mathbf{g}_{k+1}-\mathbf{g}_k$。如果 $\mathbf{H}_{k+1}$ 可逆，则可得到拟牛顿方程：

$$\mathbf{H}_{k+1}\mathbf{s}_k = \mathbf{y}_k \tag{10}$$

拟牛顿法的关键是要用上一步的海森矩阵 $\mathbf{H}_k$ 来递推当前的 $\mathbf{H}_{k+1}$，因此需要对 $\mathbf{H}_k$ 进行校正。假设 $\mathbf{H}_{k+1}$ 可以通过对 $\mathbf{H}_k$ 的秩二校正得到，即：

$$\mathbf{H}_{k+1} = \mathbf{H}_k + a\mathbf{u}\mathbf{u}^t + b\mathbf{v}\mathbf{v}^t \tag{11}$$

将（11）式代入拟牛顿方程（10），则有：

$$\mathbf{H}_{k+1}\mathbf{s}_k = \mathbf{H}_k\mathbf{s}_k + a\mathbf{u}\mathbf{u}^t\mathbf{s}_k + b\mathbf{v}\mathbf{v}^t\mathbf{s}_k = \mathbf{y}_k \tag{12}$$

满足（12）式的 $\mathbf{u}$ 、$\mathbf{v}$ 和 a 、b 是不唯一的，可以选择：$\mathbf{u}=\mathbf{y}_k$， $\mathbf{v}=\mathbf{H}_k\mathbf{s}_k$，代入（12）式：

$$\mathbf{v} + a\mathbf{u}\mathbf{u}^t\mathbf{s}_k + b\mathbf{v}\mathbf{v}^t\mathbf{s}_k - \mathbf{u} = (a\mathbf{u}^t\mathbf{s}_k - 1)\mathbf{u} + (b\mathbf{v}^t\mathbf{s}_k + 1)\mathbf{v} = \mathbf{0}$$

当 $\mathbf{u}$ 和 $\mathbf{v}$ 不为 0 矢量时有：

$$a\mathbf{u}^t\mathbf{s}_k = 1, \quad b\mathbf{v}^t\mathbf{s}_k = -1$$

因此：

$$a = \frac{1}{\mathbf{u}^t\mathbf{s}_k} = \frac{1}{\mathbf{y}_k^t\mathbf{s}_k}, \quad b = -\frac{1}{\mathbf{v}^t\mathbf{s}_k} = -\frac{1}{\mathbf{s}_k^t\mathbf{H}_k\mathbf{s}_k} \tag{13}$$

代入（11）式，得到递推公式：

$$\mathbf{H}_{k+1} = \mathbf{H}_k + \frac{\mathbf{y}_k\mathbf{y}_k^t}{\mathbf{y}_k^t\mathbf{s}_k} - \frac{\mathbf{H}_k\mathbf{s}_k\mathbf{s}_k^t\mathbf{H}_k}{\mathbf{s}_k^t\mathbf{H}_k\mathbf{s}_k} \tag{14}$$

这样，我们就可以由上一步的海森矩阵 $\mathbf{H}_k$ 、位置差 $\mathbf{s}_k = \mathbf{x}_{k+1}-\mathbf{x}_k$ 以及梯度差 $\mathbf{y}_k = \mathbf{g}_{k+1}-\mathbf{g}_k$ 来近似地递推计算当前的海森矩阵 $\mathbf{H}_{k+1}$，而不需要计算函数 $f(\mathbf{x})$ 的二阶导数。在牛顿法中需要计算的是海森矩阵的逆矩阵（见公式（7）），而由（14）式迭代得到的近似矩阵存在奇异阵的可能性，同时逆矩阵的计算也相对比较复杂，因此常用的方法是直接递推海森矩阵的逆矩阵。由（14）式进一步的推导可以得到逆矩阵的递推公式：

$$\mathbf{H}_{k+1}^{-1} = \left(\mathbf{I} - \frac{\mathbf{s}_k\mathbf{y}_k^t}{\mathbf{s}_k^t\mathbf{y}_k}\right)\mathbf{H}_k^{-1}\left(\mathbf{I} - \frac{\mathbf{y}_k\mathbf{s}_k^t}{\mathbf{s}_k^t\mathbf{y}_k}\right) + \frac{\mathbf{s}_k\mathbf{s}_k^t}{\mathbf{s}_k^t\mathbf{y}_k} \tag{15}$$

（14）和（15）一般称为 BFGS 拟牛顿法递推公式。

BFGS 拟牛顿算法

初始化：$\mathbf{x}_0$，$\mathbf{H}_0=\mathbf{I}$，θ，$k=0$

计算 $\mathbf{x}_0$ 处的梯度矢量 $\mathbf{g}_0=\nabla f(\mathbf{x})\big|_{\mathbf{x}=\mathbf{x}_0}$；

do

计算优化方向：$\mathbf{d}_k=-\mathbf{H}_k^{-1}\mathbf{g}_k$；

沿方向 $\mathbf{d}_k$ 进行 1 维搜索，使得 $\mathbf{x}_{k+1}=\mathbf{x}_k+\alpha_k\mathbf{d}_k$ 为此方向上的极小值点，$\alpha_k>0$；

计算 $\mathbf{x}_{k+1}$ 处的梯度矢量：$g_{k+1}=\nabla f(\mathbf{x})\big|_{\mathbf{x}=\mathbf{x}_{k+1}}$；

由公式（15）递推计算 $\mathbf{H}_{k+1}^{-1}$；

$k=k+1$

until $\|\mathbf{g}_{k+1}\|<\theta$；

输出优化解 $\mathbf{x}_{k+1}$；

V. 共轭梯度法

拟牛顿法解决了二阶导数矩阵 $\mathbf{H}$ 的计算问题，但当矢量 $\mathbf{x}$ 的维数 m 较大时，需要的存储量和矩阵乘法的计算量仍然较大。共轭梯度法也是一种近似的二阶方法，只需计算一阶导数，不需要计算和存储二阶导数矩阵。

对于 $m\times m$ 矩阵 $\mathbf{H}$ 来说，如果对于任意的两个矢量 $\mathbf{d}_i,\mathbf{d}_j$ 满足 $\mathbf{d}_i^t\mathbf{H}\mathbf{d}_j=0$，$i\neq j$，$i,j=0,1,\cdots,m-1$，则称 $\mathbf{d}_0,\cdots,\mathbf{d}_{m-1}$ 是关于矩阵 $\mathbf{H}$ 的一组共轭矢量。显然，当 $\mathbf{H}$ 为单位矩阵 $\mathbf{I}$ 时，这组共轭矢量之间是相互正交的。

下面来看一下优化函数 $f(\mathbf{x})$ 为二次正定函数的情况：$f(\mathbf{x})=\mathbf{x}^t\mathbf{H}\mathbf{x}+\mathbf{u}^t\mathbf{x}+c$，其中 $\mathbf{H}$ 为 $n\times n$ 的正定对称矩阵。我们先来证明这样一个事实：如果采用共轭方向算法从任意的起始点 $\mathbf{x}_0$ 出发，每一轮迭代都是沿着矩阵 $\mathbf{H}$ 的一组共轭矢量方向 $\mathbf{d}_0,\cdots,\mathbf{d}_{m-1}$ 做 1 维搜索，找到在一个共轭方向上的最小值点，那么只需要 m 轮迭代就可以找到函数 $f(\mathbf{x})$ 的最小值点。

共轭方向算法：

1、初始化起始点 $\mathbf{x}_0$，一组关于矩阵 $\mathbf{H}$ 的共轭矢量 $\mathbf{d}_0,\cdots,\mathbf{d}_{m-1}$，$k=0$；

2、计算 α_k 和 $\mathbf{x}_{k+1}$，使得：

$$f(\mathbf{x}_k+\alpha_k\mathbf{d}_k)=\min_{\alpha} f(\mathbf{x}_k+\alpha\mathbf{d}_k)$$

$$\mathbf{x}_{k+1}=\mathbf{x}_k+\alpha_k\mathbf{d}_k$$

3、转到 2，直到 $k=m-1$ 为止。

上述事实可以由如下定理的证明得到：

【定理 1】对于正定二次优化函数 $f(\mathbf{x})$，如果按照共轭方向进行搜索，至多经过 m 步精确的线性搜索可以终止；并且每一个 $\mathbf{x}_{i+1}$ 都是在 $\mathbf{x}_0$ 和方向 $\mathbf{d}_0,\cdots,\mathbf{d}_i$ 所张成的线性流形 $\left\{\mathbf{x}\middle|\mathbf{x}=\mathbf{x}_0+\sum_{j=0}^{i}\alpha_j\mathbf{d}_j\right\}$ 中的极值点。

证明：令 $\mathbf{g}_i$ 为第 i 步的梯度，即：$\mathbf{g}_i=\nabla f(\mathbf{x})\big|_{\mathbf{x}=\mathbf{x}_i}$。上述定理实际上只需证明，对 $\forall j\le i$，

$\mathbf{g}_{i+1}^t\mathbf{d}_j=0$ 即可。因为 $\mathbf{g}_{i+1}$ 正交于 $\mathbf{d}_0,\cdots,\mathbf{d}_i$，则 $\mathbf{g}_{i+1}$ 正交于它们所张成的线性流形，$\mathbf{x}_{i+1}=\mathbf{x}_0+\sum_{j=0}^{i}\alpha_j\mathbf{d}_j$ 包含在此线性流形中，因此在此线性流形中 $f(\mathbf{x})$ 的梯度为 0，即 $\mathbf{x}_{i+1}$ 为在线性流形上的极值点。当 $i+1=m$ 时，$\mathbf{d}_0,\cdots,\mathbf{d}_{m-1}$ 所张成的线性流形即为整个 m 维空间 R^m，只有当 $\mathbf{g}_m=\mathbf{0}$ 时，才有 $\mathbf{g}_m^t\mathbf{d}_j=0$ 成立，因此 $\mathbf{x}_m$ 为极值点。

梯度 $\mathbf{g}=\nabla f(\mathbf{x})=\mathbf{H}\mathbf{x}+\mathbf{u}$，因此两次迭代之间梯度的差值矢量为：

$$\mathbf{g}_{k+1}-\mathbf{g}_k=\mathbf{H}(\mathbf{x}_{k+1}-\mathbf{x}_k)=\alpha_k\mathbf{H}\mathbf{d}_k \tag{16}$$

对于 $\forall j<i$：

$$\begin{aligned}\mathbf{g}_{i+1}^t\mathbf{d}_j&=\mathbf{g}_{i+1}^t\mathbf{d}_j-\mathbf{g}_i^t\mathbf{d}_j+\mathbf{g}_i^t\mathbf{d}_j-\mathbf{g}_{i-1}^t\mathbf{d}_j+\mathbf{g}_{i-1}^t\mathbf{d}_j-\cdots+\mathbf{g}_{j+1}^t\mathbf{d}_j\\&=\mathbf{g}_{j+1}^t\mathbf{d}_j+\sum_{k=j+1}^{i}(\mathbf{g}_{k+1}-\mathbf{g}_k)^t\mathbf{d}_j\\&=\mathbf{g}_{j+1}^t\mathbf{d}_j+\sum_{k=j+1}^{i}\alpha_k\mathbf{d}_k^t\mathbf{H}\mathbf{d}_j\end{aligned}$$

$\mathbf{x}_{j+1}$ 是沿着 $\mathbf{d}_j$ 方向搜索的极值点，因此 $\mathbf{g}_{j+1}^t\mathbf{d}_j=0$，而 $\mathbf{d}_0,\cdots,\mathbf{d}_i$ 互为共轭，所以有 $\sum_{k=j+1}^{i}\alpha_k\mathbf{d}_k^t\mathbf{H}\mathbf{d}_j=0$，因此：

$$\mathbf{g}_{i+1}^t\mathbf{d}_j=0$$

上述定理得证。■

由此可以看出，当优化函数 $f(\mathbf{x})$ 是二次函数时，共轭方向法只需经过 m 轮迭代就可以收敛于函数的最小值点。如果函数 $f(\mathbf{x})$ 不是二次函数，我们仍然可以在每一轮迭代中沿着海森矩阵 $\mathbf{H}$ 的共轭方向搜索到极小值点，只不过不能保证算法经过 m 轮迭代收敛，一般需要更多的迭代次数。

使用这种方法还需要解决的一个问题是，如何得到关于优化函数 $f(\mathbf{x})$ 海森矩阵 $\mathbf{H}$ 的一组共轭方向矢量，而不需要计算出矩阵 $\mathbf{H}$。实际上任意给定一个初始的方向 $\mathbf{d}_0$ 就可以确定一组关于矩阵 $\mathbf{H}$ 的共轭方向矢量，在共轭梯度法中一般选择初始点 $\mathbf{x}_0$ 处的负梯度方向作为初始方向：

$$\mathbf{d}_0=-\mathbf{g}_0=-\left(\frac{\partial f}{\partial\mathbf{x}}\right)_{\mathbf{x}=\mathbf{x}_0} \tag{17}$$

而第 $k+1$ 步的共轭方向矢量 $\mathbf{d}_{k+1}$，由第 $k+1$ 步的负梯度方向 $-\mathbf{g}_{k+1}$ 与第 k 步的共轭方向矢量 $\mathbf{d}_k$ 的线性组合得到：

$$\mathbf{d}_{k+1}=-\mathbf{g}_{k+1}+\beta_k\mathbf{d}_k \tag{18}$$

组合系数 β_k 的确定有多种方式，常用的包括：

Crowder-Wolfe 公式：$\beta_k=\dfrac{\mathbf{g}_{k+1}^t(\mathbf{g}_{k+1}-\mathbf{g}_k)}{\mathbf{d}_k^t(\mathbf{g}_{k+1}-\mathbf{g}_k)}$ （19）

Fletcher-Reeves 公式：$\beta_k=\dfrac{\mathbf{g}_{k+1}^t\mathbf{g}_{k+1}}{\mathbf{g}_k^t\mathbf{g}_k}$ （20）

Polak-Ribiere-Polyak 公式： $\beta_k = \dfrac{\mathbf{g}_{k+1}^t(\mathbf{g}_{k+1}-\mathbf{g}_k)}{\mathbf{g}_k^t\mathbf{g}_k}$ （21）

共轭梯度算法

初始化： $\mathbf{x}_0$ ， θ ， $k=0$

计算 $\mathbf{x}_0$ 处的负梯度矢量作为初始的搜索方向： $\mathbf{d}_0 = -\mathbf{g}_0 = -\nabla f(\mathbf{x})\big|_{\mathbf{x}=\mathbf{x}_0}$ ；

do

沿着共轭方向 $\mathbf{d}_k$ 搜索局部最小值点，即求解优化问题；

$$f(\mathbf{x}_k+\alpha_k\mathbf{d}_k) = \min_{\alpha} f(\mathbf{x}_k+\alpha\mathbf{d}_k)$$

$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k\mathbf{d}_k$

计算 $\mathbf{x}_{k+1}$ 处的梯度矢量： $g_{k+1} = \nabla f(\mathbf{x})\big|_{\mathbf{x}=\mathbf{x}_{k+1}}$ ；

由公式（18）以及（19）、（20）、（21）中的一个计算下一个搜索方向 $\mathbf{d}_{k+1}$ ；

$k = k+1$

until $\|\mathbf{g}_{k+1}\| < \theta$ ；

输出优化解 $\mathbf{x}_{k+1}$ ；

VI. 约束优化

前面介绍的几种方法，解决的都是直接针对函数 $f(\mathbf{x})$ 的优化问题，解矢量 $\mathbf{x}^*$ 可以是 R^m 空间中的任意矢量，一般称为“无约束优化”。在模式识别中，经常还会遇到另外一类“约束优化”问题，要求解矢量满足一定的约束条件。约束优化问题的一般形式可以表示为：

$$\min_{\mathbf{x}\in R^m} f(\mathbf{x}) \qquad (22)$$

约束：

$$c_i(\mathbf{x}) = 0, \quad i = 1,\cdots,l$$
$$c_i(\mathbf{x}) \le 0, \quad i = l+1,\cdots,k$$

其中前 l 个称为等式约束，后 $k-l$ 个称为不等式约束。对于约束优化问题的严格证明比较复杂，需要的数学知识超出了本书的范畴，下面简单介绍两种正文中需要用到的约束问题求解方法。

线性等式约束优化问题

在这类问题中只包含等式约束，并且函数 $c_i(\mathbf{x})$ 均为线性函数。先看一个二维矢量和一个线性约束的简单例子：

$$\min_{\mathbf{x}\in R^2} f(\mathbf{x}) = x_1^2 + x_2^2 \qquad (23)$$

约束：

$$\mathbf{a}^t\mathbf{x} = b$$

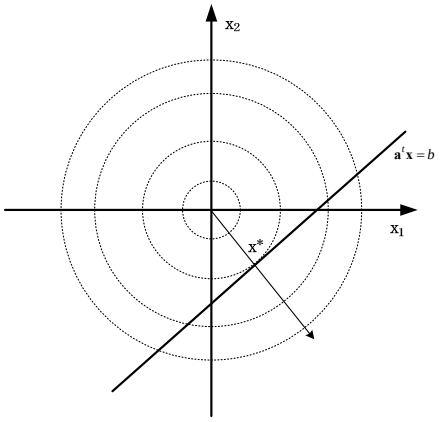

图 1 极值点处优化函数的梯度与线性约束的权值矢量共线

图 1 中虚线表示的是优化函数 $f(\mathbf{x})$ 的等值线，而实线是满足约束条件的点所在的直线，优化问题的解 $\mathbf{x}^*$ 应该在这条直线上。显然这条直线上函数值最小的点处于直线与 $f(\mathbf{x})$ 等值线相切的位置，即在 $\mathbf{x}^*$ 处优化函数的梯度矢量 $\nabla f(\mathbf{x}^*)$ 与直线正交，与直线的权值矢量 $\mathbf{a}$ 共线，因此 $\mathbf{x}^*$ 是如下方程组的解：

$$\begin{cases} \nabla f(\mathbf{x}) = \dfrac{\partial f(\mathbf{x})}{\partial \mathbf{x}} = \lambda \mathbf{a} \\ \mathbf{a}^t \mathbf{x} = b \end{cases} \tag{24}$$

如果我们构造一个函数：

$$L(\mathbf{x}, \lambda) = f(\mathbf{x}) - \lambda(\mathbf{a}^t \mathbf{x} - b) \tag{25}$$

那么就会发现：

$$\frac{\partial L(\mathbf{x}, \lambda)}{\partial \mathbf{x}} = \frac{\partial f(\mathbf{x})}{\partial \mathbf{x}} - \lambda \mathbf{a} = 0, \quad \frac{\partial L(\mathbf{x}, \lambda)}{\partial \lambda} = -\mathbf{a}^t \mathbf{x} + b = 0 \tag{26}$$

刚好得到了（24）式的方程组。换句话说就是，（23）式的约束优化问题可以转化为求解函数 $L(\mathbf{x}, \lambda)$ 的无约束优化极值问题。函数 $L(\mathbf{x}, \lambda)$ 称为 Lagrange 函数，而 λ 称为 Lagrange 系数。

对于 m 维矢量的优化，以及多个线性等式约束的问题也有同样的结果。将所有的线性约束写成矩阵形式：

$$\min_{\mathbf{x} \in R^m} f(\mathbf{x}) \tag{27}$$

约束：

$$\mathbf{A}\mathbf{x} = \mathbf{b}$$

可以证明，上述约束优化问题可以通过构造 Lagrange 函数转化为无约束问题求解：

$$L(\mathbf{x}, \boldsymbol{\lambda}) = f(\mathbf{x}) - \boldsymbol{\lambda}^t (\mathbf{A}\mathbf{x} - \mathbf{b}) = f(\mathbf{x}) - \sum_{i=1}^{l} \lambda_i (\mathbf{a}_i^t \mathbf{x} - b_i) \tag{28}$$

Lagrange 函数的极值点满足方程：

$$\begin{cases} \dfrac{\partial L(\mathbf{x}, \boldsymbol{\lambda})}{\partial \mathbf{x}} = \dfrac{\partial f(\mathbf{x})}{\partial \mathbf{x}} - \mathbf{A}^t \boldsymbol{\lambda} = \mathbf{0} \\ \dfrac{\partial L(\mathbf{x}, \boldsymbol{\lambda})}{\partial \boldsymbol{\lambda}} = \mathbf{A}\mathbf{x} - \mathbf{b} = \mathbf{0} \end{cases} \tag{29}$$

不等式约束优化问题

不等式约束优化问题的一般形式是：

$$\min_{\mathbf{x}\in R^m} f(\mathbf{x}) \tag{30}$$

约束：

$$c_i(\mathbf{x}) \le 0, \quad i = 1,\cdots,k$$

类似于等式约束优化，引入 Lagrange 函数：

$$L(\mathbf{x},\boldsymbol{\lambda}) = f(\mathbf{x}) - \sum_{i=1}^{k} \lambda_i c_i(\mathbf{x}) \tag{31}$$

可以证明优化问题（30）的最优解 $\mathbf{x}^*$，满足如下一组必要条件，称为 Karush-Kuhn-Tucker（KKT）条件：

1. $\partial L(\mathbf{x}^*,\boldsymbol{\lambda}) / \partial \mathbf{x} = \mathbf{0}$
2. $\lambda_i \ge 0, \qquad i = 1,2,\cdots,k$
3. $\lambda_i c_i(\mathbf{x}^*) = 0, \quad i = 1,2,\cdots,k$

因此，不等式约束优化问题可以通过求解上述三式得到极值点。观察条件 2 和 3 可以得出这样一个关于约束不等式的结论：当某一个不等式在极值点 $\mathbf{x}^*$ 上是以 $c_i(\mathbf{x}) < 0$ 的方式得到满足时，对应的 Lagrange 系数 $\lambda_i = 0$；当以 $c_i(\mathbf{x}) = 0$ 的形式满足时，$\lambda_i > 0$。