$$0 \le \alpha_i \le C, \forall i$$

从而最终我们的问题变为：

$$\min_{\vec{\alpha}} \Psi(\vec{\alpha}) = \min_{\vec{\alpha}} \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} y_i y_j K(\vec{x}_i, \vec{x}_j) \alpha_i \alpha_j - \sum_{i=1}^{N} \alpha_i$$
$$0 \le \alpha_i \le C, \forall i,$$
$$\sum_{i=1}^{N} y_i \alpha_i = 0.$$

继而，根据KKT条件可以得出其中α_i取值的意义为：

$$\alpha_i = 0 \Leftrightarrow y_i u_i \ge 1,$$
$$0 < \alpha_i < C \Leftrightarrow y_i u_i = 1,$$
$$\alpha_i = C \Leftrightarrow y_i u_i \le 1.$$

这里的α_i还是拉格朗日乘子(问题通过拉格朗日乘法数来求解)

1. 对于第1种情况，表明α_i是正常分类，在边界内部（我们知道正确分类的点yi*f(xi)>=0）；
2. 对于第2种情况，表明了α_i是支持向量，在边界上；
3. 对于第3种情况，表明了α_i是在两条边界之间；

而最优解需要满足KKT条件，即上述3个条件都得满足，以下几种情况出现将会出现不满足：

- $y_i u_i$<=1但是α_i<C则是不满足的,而原本α_i=C
- $y_i u_i$>=1但是α_i>0则是不满足的而原本α_i=0

- $y_i u_i$=1但是α_i=0或者α_i=C则表明不满足的，而原本应该是0<α_i<C

所以要找出不满足KKT条件的这些ai，并更新这些ai，但这些ai又受到另外一个约束，即

注：别忘了2.1.1节中，L对a、b求偏导，得到：

$$\frac{\partial \mathcal{L}}{\partial w} = 0 \Rightarrow w = \sum_{i=1}^{n} \alpha_i y_i x_i$$

$$\frac{\partial \mathcal{L}}{\partial b} = 0 \Rightarrow \sum_{i=1}^{n} \alpha_i y_i = 0$$

因此，我们通过另一个方法，即同时更新ai和aj，要求满足以下等式：

$$a_i^{new} y_i + a_j^{new} y_j = a_i^{old} y_i + a_j^{old} y_j = \text{常数}$$

就能保证和为0的约束。

利用yiai+yjaj=常数，消去ai，可得到一个关于单变量aj的一个凸二次规划问题，不考虑其约束0<=aj<=C,可以得其解为：

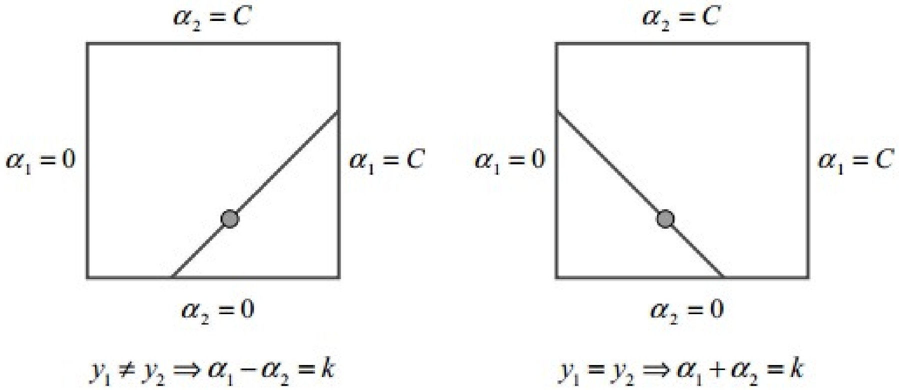
$$\alpha_j^{new} = \alpha_j + \frac{y_j (E_i - E_j)}{\eta}$$

这里$E_i = u_i - y_i$，$\eta = K(\vec{x}_1, \vec{x}_1) + K(\vec{x}_2, \vec{x}_2) - 2K(\vec{x}_1, \vec{x}_2)$，$\alpha_j$表示旧值。

然后考虑约束0<=aj<=C可得到a的解析解为：

$$\alpha_j^{new,clipped} = \begin{cases} H & \text{if } \alpha_j^{new} \geq H \\ \alpha_j^{new} & \text{if } L < \alpha_j^{new} < H \\ L & \text{if } \alpha_j^{new} \leq L \end{cases}$$

把SMO中对于两个参数求解过程看成线性规划来理解来理解的话，那么下图所表达的便是约束条件 $a_i^{new}y_i + a_j^{new}y_j = a_i^{old}y_i + a_j^{old}y_j = 常数$ ：



根据yi和yj同号或异号，可得出两个拉格朗日乘子α的上下界分别为：

$$\begin{cases} L = \max(0, \alpha_j - \alpha_i), H = \max(C, C + \alpha_j - \alpha_i) & if \quad y_i \neq y_j \\ L = \max(0, \alpha_j + \alpha_i - C), H = \max(C, \alpha_j - \alpha_i) & if \quad y_i = y_j \end{cases}$$

对于α_i，有 $\alpha_i^{new} = \alpha_i + y_i y_j(\alpha_j - \alpha_j^{new,clipped})$ 。

那么如何求得ai和aj呢？

- 对于ai，即第一个乘子，可以通过刚刚说的那3种不满足KKT的条件来找；
- 而对于第二个乘子aj可以找满足条件： $\max|E_i - E_j|$ 求得。

而b的更新则是：

$$b_1 = b - E_i - y_i(a_i - a_i^{old})k(x_i, x_i) - y_j(a_j - a_j^{old})k(x_i, x_j)$$

$$b_2 = b - E_j - y_i(a_i - a_i^{old})k(x_i, x_i) - y_j(a_j - a_j^{old})k(x_j, x_j)$$

在满足下述条件：

$$
b := \begin{cases} b_1 & \text{if } 0 < \alpha_i < C \\ b_2 & \text{if } 0 < \alpha_j < C \\ (b_1 + b_2)/2 & \text{otherwise} \end{cases}
$$

下更新b，且每次更新完两个乘子的优化后，都需要再重新计算b，及对应的Ei值。

最后更新所有ai，y和b，这样模型就出来了，从而即可求出咱们开头提出的分类函数

$$
f(x) = \sum_{j=1}^{n} \alpha_j y_j k(x_j, x) + b
$$

此外，这里也有一篇类似的文章，大家可以参考下。

3.5.2、SMO算法的步骤

这样，SMO的主要步骤如下：

Repeat till convergence {

1. Select some pair α_i and α_j to update next (using a heuristic that tries to pick the two that will allow us to make the biggest progress towards the global maximum).
2. Reoptimize $W(\alpha)$ with respect to α_i and α_j, while holding all the other α_k's $(k \neq i, j)$ fixed.

}

意思是，

1. 第一步选取一对α_i和α_j，选取方法使用启发式方法；
2. 第二步，固定除α_i和α_j之外的其他参数，确定W极值条件下的α_i，α_j由α_i表示。

假定在某一次迭代中，需要更新x_1，x_2对应的拉格朗日乘子α_1，α_2，那么这个小规模的二次规划问题写为：

$$
L_s = \max_{\alpha}\{(\alpha_1 + \alpha_2) + \sum_{i=3}^{n} \alpha_i - \frac{1}{2}\left|\left|\alpha_1 y_1 \phi(\boldsymbol{x_1}) + \alpha_1 y_1 \phi(\boldsymbol{x_1}) + \sum_{i=3}^{n} \alpha_i y_i \phi(\boldsymbol{x_i})\right|\right|^2\}
$$

$$
s.t. \alpha_1 y_1 + \alpha_2 y_2 = -\sum_{i=3}^{n} \alpha_i y_i, 0 < \alpha_i < C, \forall i
$$

那么在每次迭代中，如何更新乘子呢？引用这里的两张PPT说明下：

更新拉格朗日乘子α_1, α_2

– 步骤1：计算上下界L和H

- $L = \max(0, \alpha_2^{old} - \alpha_1^{old}), H = \min(C, C + \alpha_2^{old} - \alpha_1^{old}), if\ y_1 \neq y_2$
- $L = \max(0, \alpha_2^{old} + \alpha_1^{old} - C), H = \min(C, \alpha_2^{old} + \alpha_1^{old}), if\ y_1 = y_2$

– 步骤2：计算L_s的二阶导数

- $\eta = 2\phi(\boldsymbol{x_1})^t\phi(\boldsymbol{x_2}) - \phi(\boldsymbol{x_1})^t\phi(\boldsymbol{x_1}) - \phi(\boldsymbol{x_2})^t\phi(\boldsymbol{x_2})$

– 步骤3：更新L_s

$$\alpha_2^{new} = \alpha_2^{old} - \frac{y_2(e_1 - e_2)}{\eta}$$

$$e_i = g^{old}(\boldsymbol{x_i}) - y_i$$

– 步骤4：计算变量α_2

$$\alpha^{temp} = \begin{cases} H, & \text{if}\alpha_2^{new} \geq H \\ \alpha_2^{new}, & \text{if } L \leq \alpha_2^{new} \leq H \\ L, & \text{if } \alpha_2^{new} \leq L \end{cases}$$

– 步骤5：更新α_1

$$\alpha_1^{new} = \alpha_1^{old} + y_1 y_2(\alpha_2^{old} - \alpha^{temp})$$

知道了如何更新乘子，那么选取哪些乘子进行更新呢？具体选择方法有以下两个步骤：

1. 步骤1：先“扫描”所有乘子，把第一个违反KKT条件的作为更新对象，令为a2；
2. 步骤2：在所有不违反KKT条件的乘子中，选择使|E1 −E2|最大的a1（注：别忘了，其中$E_i = u_i - y_i$，而$u = \vec{w} \cdot \vec{x} - b$，求出来的E代表函数ui对输入xi的预测值与真实输出类标记yi之差）。

值得一提的是，每次更新完两个乘子的优化后，都需要再重新计算b，及对应的Ei值。

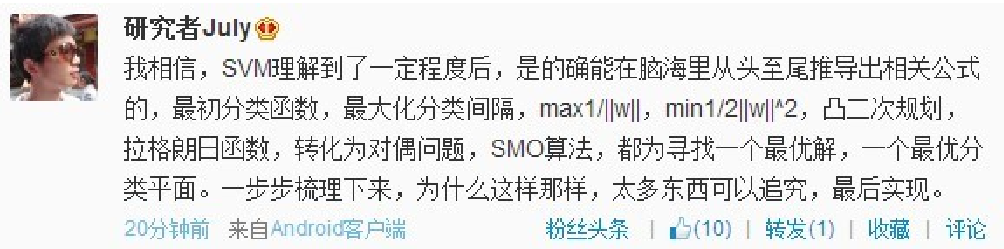
与此同时，乘子的选择务必遵循两个原则：

- 使乘子能满足KKT条件
- 对一个满足KKT条件的乘子进行更新，应能最大限度增大目标函数的值（类似于梯度下降）

综上，SMO算法的基本思想是将Vapnik在1982年提出的Chunking方法推到极致，SMO算法每次迭代只选出两个分量ai和aj进行调整，其它分量则保
在得到解ai和aj之后，再用ai和aj改进其它分量。与通常的分解算法比较，尽管它可能需要更多的迭代次数，但每次迭代的计算量比较小，所以该算法
快速收敛性，且不需要存储核矩阵，也没有矩阵运算。

3.5.3、SMO算法的实现

行文至此，我相信，SVM理解到了一定程度后，是的确能在脑海里从头至尾推导出相关公式的，最初分类函数，最大化分类间隔，max1/||w||，mi
二次规划，拉格朗日函数，转化为对偶问题，SMO算法，都为寻找一个最优解，一个最优分类平面。一步步梳理下来，为什么这样那样，太多东西可
实现。如下图所示：



至于下文中将阐述的核函数则为是为了更好的处理非线性可分的情况，而松弛变量则是为了纠正或约束少量“不安分”或脱离集体不好归类的因子。

台湾的林智仁教授写了一个封装SVM算法的libsvm库，大家可以看看，此外这里还有一份libsvm的注释文档。

除了在这篇论文《fast training of support vector machines using sequential minimal optimization》中platt给出了SMO算法的逻辑代码之外，这里也有一份SMO的实现代码，大家可以看下。

其余更多请参看文末参考文献和推荐阅读中的条目6《支持向量机--算法、理论和扩展》和条目11《统计学习方法》的相关章节，或跳至下文3.4节。

3.6、SVM的应用

或许我们已经听到过，SVM在很多诸如文本分类，图像分类，生物序列分析和生物数据挖掘，手写字符识别等领域有很多的应用，但或许你并没强烈的意识到，SVM可以成功应用的领域远远超出现在已经在开发应用了的领域。