```
%  with the constraint
%  \sum_{i = 1}^p + \sum_{j = 1}^q mu_j  = K_m
%  (without the  variable gamma) and the right-hand side c
%  u: vector of p blue points (each an n-dim vector)
%  v: vector of q red points (each an n-dim vector)
%  builds the matrix X = [-u_1 ... -u_p v1 .... v_q]
%  and the matrix Pa as 2(p+q) matrix obtained
%  by augmenting X'*X with zeros
%  K is a scale factor (K = Ks)

p = size(u,2); q = size(v,2);
% Ks = 1/(p+q);
Ks = K; Km = (p+q)*K*nu;
A = [ones(1,p) -ones(1,q) zeros(1,p+q);
     ones(1,p) ones(1,q)  zeros(1,p+q) ;
     eye(p) zeros(p,q) eye(p) zeros(p,q);
     zeros(q,p) eye(q)  zeros(q,p) eye(q) ];
c = [0; Km; Ks*ones(p+q,1)];
X = [-u v];
XX = X'*X;
Pa = [XX zeros(p+q,p+q); zeros(p+q, 2*(p+q))];
q = zeros(2*(p+q),1);
end
```

The function findpsv2 makes a vector of λ_i (and μ_j) corresponding to support vectors of type 1.

```
function [lampsv,num] = findpsv2(lambda,K,tols,tolh)
%
%   This function find the vector of
%   lambda_i's such that 0 < lambda_i < K
%   and the number of such  lambda_i.
%
% tols = 10^(-11);  %  the smaller this is, the larger the number of
                 %  points on the margin
% tolh = 10^(-9);  %
m = size(lambda,1); lampsv = zeros(m,1);
num = 0;
for i = 1:m
    if lambda(i) > tols && lambda(i) < K - tolh
       lampsv(i) = lambda(i);
       num = num + 1;
    end
```