The Huller combines desirable properties:

- **It converges** to the **Hard Margin SVM** solution (with bias),
- A **single pass** over the training set matches the error rates of a SVM, using a fraction of the time and memory.

3- **Parametrization**

Solve
\[
\min_{\alpha} ||X_p - X_N||^2
\]
subject to
\[
X_p = \sum_{i \in P} \alpha_i x_i, \quad \sum_{i \in P} \alpha_i = 1, \quad \alpha_i \geq 0
\]
\[
X_N = \sum_{j \in N} \alpha_j x_j, \quad \sum_{j \in N} \alpha_j = 1, \quad \alpha_j \geq 0
\]

Then, the optimal hyperplane is:
\[
\hat{y}(x) = (X_p - X_N) x + \left(\frac{X_N X_N - X_p X_p}{2}\right)
\]

5- **Tricks**

- **Remove support vectors efficiently.**
  When \( x_i \) is already a support vector (\( \alpha_i > 0 \)), allow \( \lambda \) to become slightly negative.

  - **Compute \( \lambda \) very quickly.**
    Cache values of \( X_p X_p, X_N X_N, \) and \( X_p X_N \).
    Update cached values when \( X_p \) or \( X_N \) changes.

    Iteration time is proportional to number of SVs.

6- **The Huller**

initialize \( \alpha, X_p, X_p, X_N, X_N \), and \( X_p X_N \)
repeat for \( t = 1 \ldots T \)
- pick a random example \( x_t \) that is not a support vector.
- perform the single example update.
- pick a random support vector \( x'_t \).
- perform the single example update.

1st update: example \( x_t \) can become support vector.
  (when \( \lambda > 0 \))

2nd update: support vector \( x'_t \) can be removed.
  (when \( \lambda \) reaches its negative lower bound.)

7- **MNIST Experiments**

- **Simple online learning algorithm.**
- **Converges to** **Hard Margin SVM** solution.
- **Matches** HM-SVM accuracy after a single epoch.
- **Uses less memory and runs quickly.**

Continuation:

Bordes, Ertekin, Weston, Bottou: Fast Kernel Classifiers with Online and Active Learning, to appear in JMLR (handles noisy data, runs faster, reduces number of support vectors with active learning.)