

# From large margin classification to $\psi$ -learning

Xiaotong Shen

Department of Statistics  
The Ohio State University

Joint with G. Tseng, X. Zhang, W. Wong, Harvard  
Univ.

Joint with S. Liu, Ohio State Univ.

# Formulation

## 1. Binary classification

(1) input space  $S$ , (2) index (output) space  $O = \{+1, -1\}$ , (3) decision function  $f$ , (4) a training sample  $Z$ .

## 2. Learning

(1) Construct  $f(x) : \mathcal{R}^d \rightarrow R^1$  via  $Z = (X_i, Y_i)_{i=1}^n$  of input/output pairs, i.i.d.  $\sim$  unknown distribution  $P(x, y)$ .

(2) Classifier (Generalizer):  $Sign(f)$ .

## 3. Accuracy

Generalization error:  $Err(f) =$

$$P(Y f(X) \leq 0) = \frac{1}{2} E(1 - Sign(Y f(X))).$$

# Large Margin Classifiers

## 1. Margins

(1) Margin for an instance  $(x_i, y_i)$  wrt a decision function  $f$  is  $\gamma_i = y_i f(x_i)$ .

(2) Separation margin:  $J(f)$ .

## 2. Penalized margin-based cost function:

$$C \sum_{i=1}^n l(Y_i f(X_i)) + J(f),$$

$C > 0$  : penalization coef;  $J(f)$ .

## 3. Examples

(1) SVM: (Hinge loss)

$l(v) = (1 - v)_+^q = \psi_{svm}(v)$ , c.f., Burge (1988).

(2)  $\varepsilon$ -SVM: (Scholkopf et al., 2001).

(3) IVM: (Logistic)  $l(v) = \log(1 + \exp(-v))$ ,  
(Hastie & Zhu, 2002).

(4)  $\psi$ -learning:  $l(v) = \psi(v)$ , (Shen et al.,  
2002).

(5) NN: (sigmoid loss)  $l(v) = 1 - \tanh(cv)$ .

## Motivation for $\psi$ -learning

### 1. Goal

Minimize generalization error.

### 2. Implementation: penalization.

$$\frac{1}{2}\|w\|^2 + C \sum_{i=1}^n (1 - \text{Sign}(y_i f(x_i))).$$

### 3. Scaling problem

Only for correctly specified instances.

### 4. Idea

Drive correctly specified instances away from decision boundaries to fix scaling:

$1 - \text{Sign} \rightarrow \psi$  (non-increasing),

$$\begin{aligned} U \geq \psi(x) &> 0 && \text{if } x \in (0, \tau] \\ \psi(x) &= (1 - \text{Sign}(x)) && \text{if otherwise,} \end{aligned} \tag{1}$$

# $\psi$ -learning

## 1. Linear

- Decision functions:  $f(x) = w \cdot \tilde{x}$ , where  $w = (w_1, \dots, w_d, b)$  and  $\tilde{x} = (x, 1)$ .
- Find  $w$  to minimize

$$s(w) = \frac{1}{2} \|w\|^2 + C \sum_{i=1}^n \psi(y_i f(x_i)). \quad (2)$$

- Tuning parameter:  $C > 0$ .

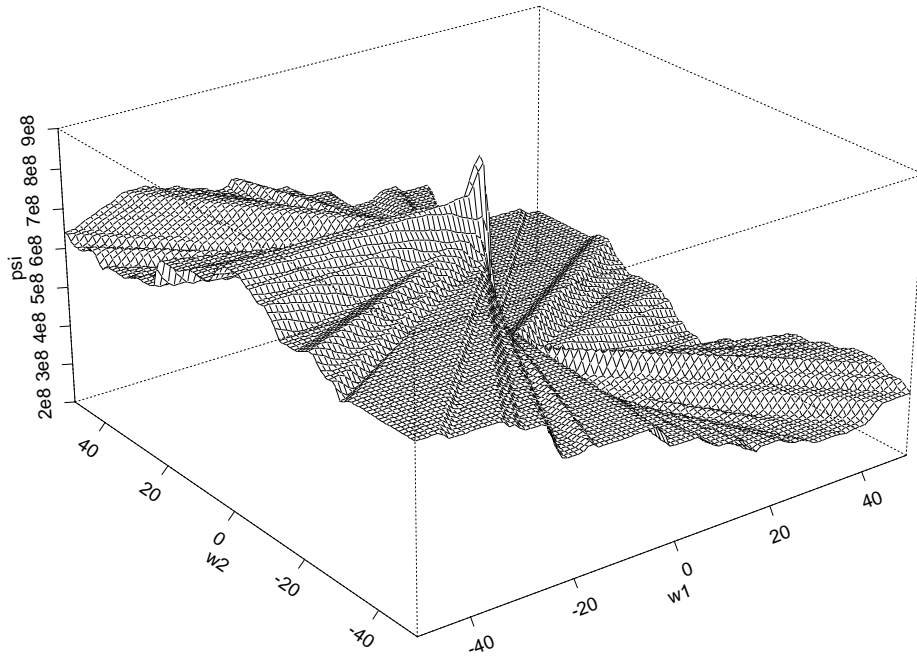
## 2. Nonlinear

- Decision functions:  $f(x) = g(x) + b$  with  $g(x) = \sum_{i=1}^n w_i K(x_i, x)$ .
- Kernel:  $K$  (satisfy some assumptions).
- Find  $\{w = (w_1, \dots, w_n), b\}$  to minimize

$$\frac{1}{2} \|g\|_K^2 + C \sum_{i=1}^n \psi(y_i f(x_i)).$$

# Deterministic Nonconvex Minimization

1. Difficult and unexplored in Computer Sciences and Statistics.
2. D.C. programming (Global minimization)  
Key: D.C. decomposition (Diff Convex func).
  - DCA (An and Tao, 97).
  - Outer approx (Blanquero & Carrizosa, 00).



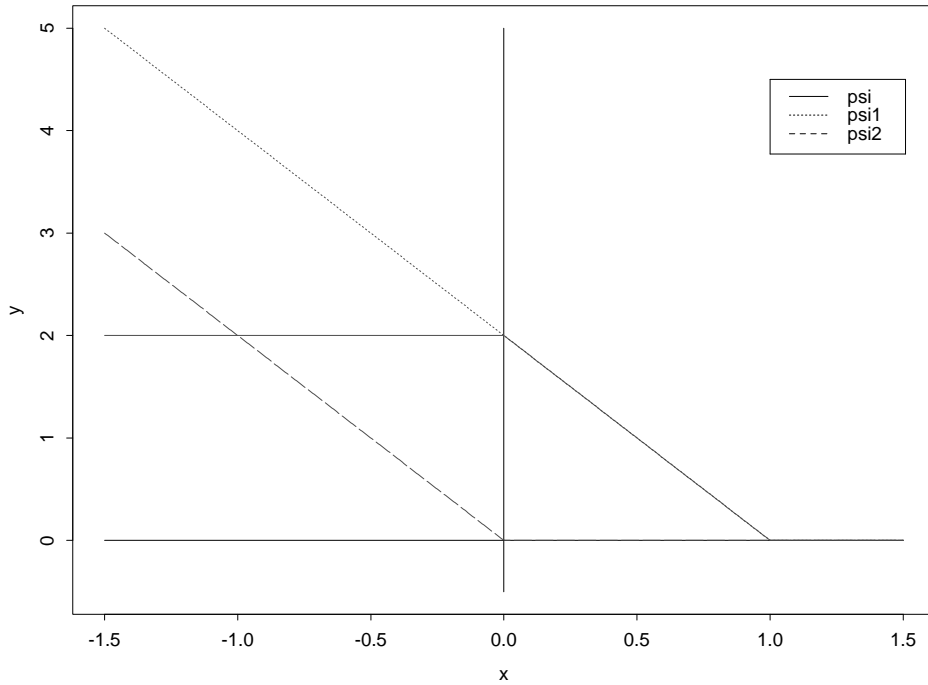
**Figure 2.** Perspective plot for  $s(w)$  as a function  $w = (w_1, w_2)$  and  $b = 0.25$  for one example with  $n = 50$ ,  $C = 10^7$  and 20% flipping.

## D.C. Decomposition

$$s(w) = s_1(w) - s_2(w), \quad (3)$$

$$s_1(w) : \frac{1}{2} \|w\|^2 + C \sum_{i=1}^n \psi_1(y_i f(\tilde{x}_i)),$$

$$s_2(w) : -C \sum_{i=1}^n \psi_2(y_i f(\tilde{x}_i)).$$



**Figure 3.** Plot of functions  $\psi$ ,  $\psi_1$ , and  $\psi_2$ , where  $\psi = \psi_1 - \psi_2$  is a difference convex decom of  $\psi$ .

# DCA

## 1. Idea

- $s_2 \rightarrow s_2(w^*) + \langle w - w^*, \nabla s_2(w^*) \rangle$ , affine minorization,  $\nabla$ : subgradient.
- Solve a seq of convex subproblems:  
 $\min_w (s_1(w) - \langle w, \nabla s_2(w^{(k)}) \rangle)$  by QP.

Linear:

$$\max_{\alpha} \sum_{i=1}^n \alpha_i [1 - y_i \langle V_1^{(k)}, x_i \rangle] - \frac{1}{2} \sum_{i,j=1}^n \alpha_i \alpha_j y_i y_j \langle x_i, x_j \rangle,$$

subj to  $\sum_{i=1}^n \alpha_i y_i = -V_2^{(k)}$ ,  $2C \geq \alpha_i \geq 0$ . Sol  
 $\alpha_i^* \rightarrow (w_1^*, \dots, w_d^*) = V_1^{(k)} + \sum_{i=1}^n \alpha_i^* y_i x_i$ ;  $b$  :  
 $\langle w^*, \tilde{x}_i \rangle = 1$  for  $i$  with  $2C > \alpha_i > 0$ .

## 2. Algorithm 1: (Linear & Nonlinear)

**Step 1:** (Initial)  $w^{(0)}$ .

**Step 2:** (Iteration) At iteration  $k$ , compute  $w^{(k+1)}$  by solving the quadratic program.

**Step 3:** (Stopping) if  $\|w^{(k+1)} - w^{(k)}\| \leq \epsilon$ .

Sol:  $w^s = \operatorname{argmin}_{i \leq k+1} w^{(i)}$ .

## DCA, Continued

1. Theorem: (Convergence of **Algorithm 1**)  
 $s(w^{(k)})$  is nonincreasing,  
 $\lim_{k \rightarrow \infty} s(w^{(k)}) \geq \min_w s(w)$ , and  
 $\lim_{k \rightarrow \infty} \|w^{(k+1)} - w^{(k)}\| = 0$ . Moreover,  
Algorithm 1 terminates finitely.
2. Convergence  
in 4-7 steps. Complexity  $\rightarrow$  QP = quadratic programming.
3. Support vectors and SVM
  - Support vectors: the instances defining the separating surface, i.e.,  $\alpha_i > 0$ ;  
 $i = 1, \dots, n$ .
  - $s_1$ : equi SVM cost function.  $s_2$ : corrects the bias due to imposed convexity on  $s_1$ .

# Outer Approximation

## 1. Idea

- $s_1 \rightarrow \max_{1 \leq j \leq k} s_1(w^{(j)}) + \langle w, \nabla s_1(w^{(j)}) \rangle$ .  
affine minorizations.
- Solve a sequence of concave problems:  
 $\min_w L^{(k)}(w)$ ,  $L^{(k)} = -s_2 + t$  with  
 $t \geq \max_{1 \leq j \leq k} \langle w, \nabla s_1(w^{(j)}) \rangle$ .

## 2. Algorithm 2:

**Step 1:** (Initial) Set  $\bar{s} = s(w_0)$  and  $w^s = w^{(0)}$ .

**Step 2:** (Vertex enumeration & discrete minimization). At iteration  $k$ , set  $\underline{s} = w^{(k+1)} = \operatorname{argmin}_w L^{(j)}(w)$ ,  $w^{(k+1)} \rightarrow$ : (1) seek new vertices for  $t \geq s_1(w^{(k)}) + \langle w, \nabla s_1(w^{(k)}) \rangle$  via vertex enumeration, (2) find  $\min (w^{(k+1)}, t^{(k+1)})$  among new  $(w, t)$ -vertices.

**Step 3:** (Redundant) Drop inactive constraints.

**Step 4:** (Stopping) If  $\bar{s} - \underline{s} \leq \varepsilon$ , stop &  $w^s = w^{(k+1)}$ , otherwise, continue. If  $s(w^{(k+1)}) < \bar{s}$ , then set  $\bar{s} = s(w^{(k+1)})$ .

## Outer Approximation, Continued

1. Theorem: (Convergence of **Algorithm 2**)  
The sequence  $w^{(k)}$  converges to global optima, i.e.,  $\lim_{k \rightarrow \infty} s(w^{(k+1)}) = \min_w s(w)$ , and  $|s(w^s) - \min_w s(w)| \leq \varepsilon$  when stop.
2. Complexity  
 $O(dN^{d/2+3})$ ,  $N$  : # active constraints.
3. Convergence  
in 200 steps but guarantee globality of solutions.
4. Comparison  
**Algorithm 1**: Good for large  $d$ ; may not be global.  
**Algorithm 2**: Good for small  $d$  and large  $n$ ; global.

# Theory for $\psi$ -Learning

## Formulation

1. Candidate classification sets:

$\mathcal{G}(\mathcal{F}) = \{x \in S : f(x) \geq 0, f \in \mathcal{F}\}$ , defined by  $\mathcal{F}$ , candidate decision functions.

2. Ideal performance:  $Err(f^*) = \min_f Err(f)$ ,  
 $f^*$ : Bayes decision,  $\bar{f} = Sign(f^*)$ : Bayes classifier.

3. Actual performance:  $Err(\hat{f})$ .

4. Comparison: (Actual-Ideal)

- $e(\hat{f}, f^*) = Err(\hat{f}) - Err(f^*) \geq 0$ .
- $e(\hat{f}, f^*) \leq e_\psi(f, \bar{f}) = \frac{1}{2} E\psi(Y f(X)) - E\psi(Y \bar{f}(X))$ .

# Assumptions

1. **Assumption A:** (Approximation) For  $s_n \rightarrow 0$  (pos seq) as  $n \rightarrow \infty$ , there exists  $f_0 \in \mathcal{F}$  such that  $e_\psi(f_0, \bar{f}) \leq s_n$ .  
Equivalently,  $\inf_{\{f \in \mathcal{F}\}} e_\psi(f, \bar{f}) \leq s_n$ .
2. **Assumption B:** (Boundary behavior) There exist some constants  $0 < \alpha \leq +\infty$  and  $c_1 > 0$  such that  $P(x \in S : |f^*(x)| \leq \delta) \leq c_1 \delta^\alpha$  for any small  $\delta \geq 0$ .
3. **Assumption C:** (Metric entropy) Equation:

$$\sup_{\{k \geq 1\}} \phi(\varepsilon_n, k) \leq c_2 n^{1/2}, \quad (4)$$

where  $\phi(\varepsilon_n, k) = \int_{c_4 L}^{c_3^{1/2} L^{\alpha/2(\alpha+1)}} H^{1/2}(u^2/2, \mathcal{G}(k)) du / L$  and  $L = L(\varepsilon_n, C, k) = \min(\varepsilon_n^2 + (Cn)^{-1} J_0(k/2 - 1), 1)$ ,  
 $J_0 = \max(J(f_0), 1)$ ,  $J(f) = \frac{1}{2} \|w\|^2$  or  $\frac{1}{2} \|g\|_K^2$ .

4. **Assumption D:**  $\psi$  satisfies (1).

## Theory of $\psi$ -Learning, Continued

1. Theorem: (Accuracy of  $\psi$ -learning  $\hat{f}$ ) For a constant  $c_5 > 0$ ,

$$P\left(e(\hat{f}, \bar{f}) \geq \delta_n^2\right) \leq 3.5 \exp\left(-c_5 n(nC)^{-\frac{\alpha+2}{\alpha+1}} J_0^{\frac{\alpha+2}{\alpha+1}}\right),$$

provided that  $Cn \geq 2\delta_n^{-2} J_0$ , where

$\delta_n^2 = \min(\max(\varepsilon_n^2, 2s_n), 1)$ , and  $\varepsilon_n > 0$

satisfying (4).

2. Corollary

$$|e(\hat{f}, f^*)| = O_p(\delta_n^2), \quad E|e(\hat{f}, f^*)| = O(\delta_n^2),$$

provided that  $n^{-\frac{1}{\alpha+1}} (C^{-1} J_0)^{\frac{\alpha+2}{\alpha+1}}$  is bounded away from zero.

# Theoretical Examples

## Linear

1. Class:  $\mathcal{F} = \{f(x) = wx + b : w \in \mathcal{R}^2\}$ .
2. Input:  $S = \{(x_1, x_2) : x_1^2 + x_2^2 \leq 1\}$ .
3. True decision (Bayes)  $f^*$ : vertical line.
4.  $e(\hat{f}, f^*) = O_p(n^{-1}); Ee(\hat{f}, f^*) = O(n^{-1})$   
for sufficiently large  $C > 0$ .
5. Rate is optimal, c.f., Blumer (1989).

## Nonlinear ( $p$ th degree of smoothness)

1.  $\mathcal{F} : \{f = g + b : g(x) = \sum_{i=1}^n w_i K(x, x_i)\}$ ,  
where  $f$  has at most  $M$  of real roots.  
 $J(f) : \int_0^1 [\frac{dg}{dx}]^2 dx < \infty$ ,  $K$ : Spline kernel.
2. Input:  $S = [0, 1]$ .
3. True decision  $f^* \in \mathcal{F}$ .
4.  $e(\hat{f}, f^*) = O_p(\frac{\log n}{n}); Ee(\hat{f}, f^*) = O(\frac{\log n}{n})$ ,  
for  $C \sim \varepsilon_n^2 n^{-1} \sim 1/\log n$ .
5. Surprisingly fast rate: nearly optimal.

## Simulated Examples

### 1. Purpose

- Performance:  $\psi$ -learning vs SVM.
- Gain an insight: Choice of  $l$ .

### 2. Two-D linear classification

- Decision functions:  $f(x) = \sum_{i=1}^2 w_i x_i + b$ .
- Training samples:
  - (a)  $\{(X_{i1}, X_{i2})\}_{i=1}^n \rightarrow$  uniform over  $\{(x_1, x_2) : x_1^2 + x_2^2 \leq 1\}$ .
  - (b)  $Y_i = 1$  if  $X_{i1} \geq 0$  and  $-1$  otherwise.
  - (c) Randomly selected labels  $\{Y_i\}_{i=1}^n \rightarrow$  flipped.

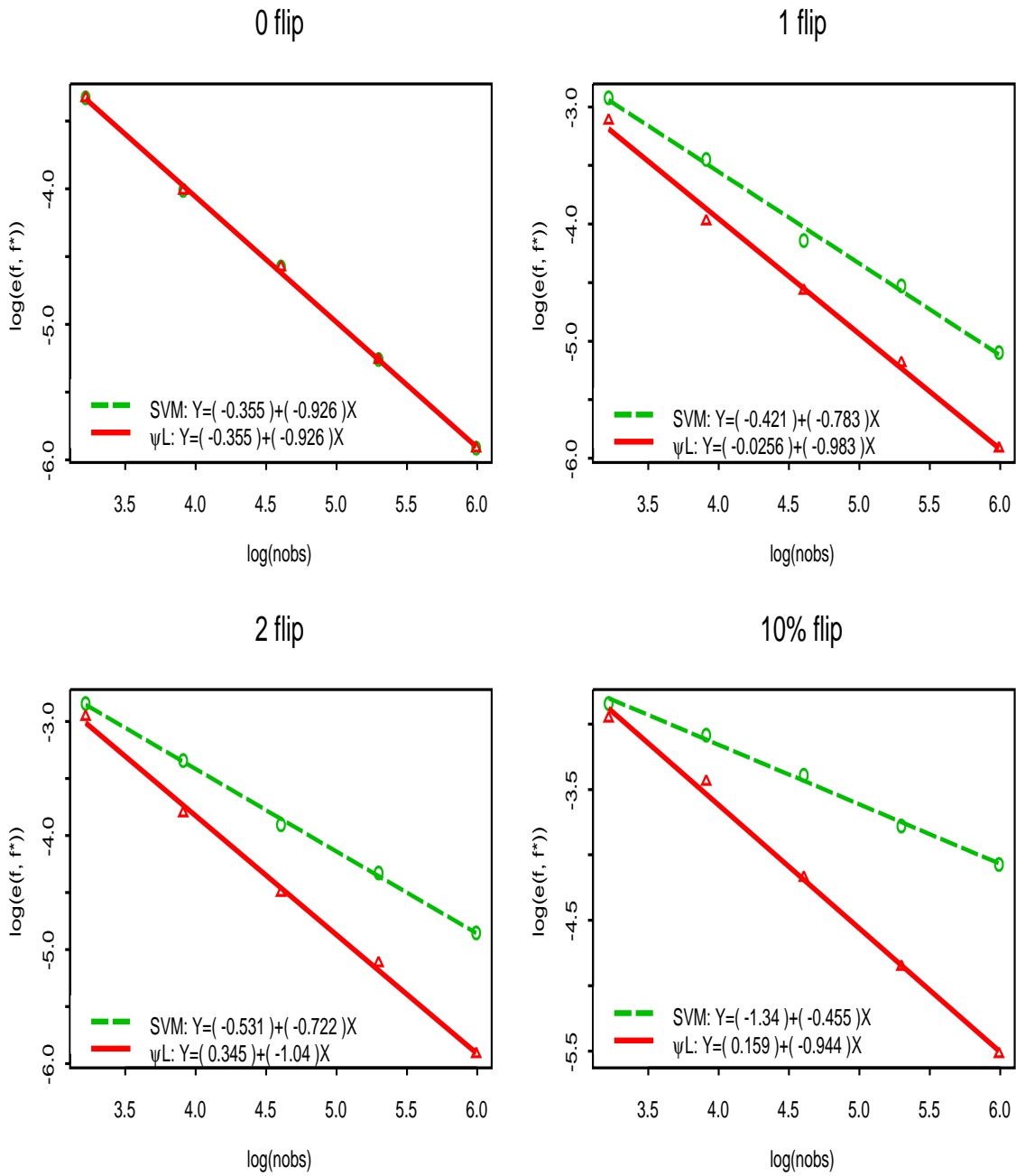


Figure 1: Plot of  $\log(e(\hat{f}, f^*))$  as a function of  $\log n$  for  $\psi$ -learning and SVM. Here  $C = 10^7$ .

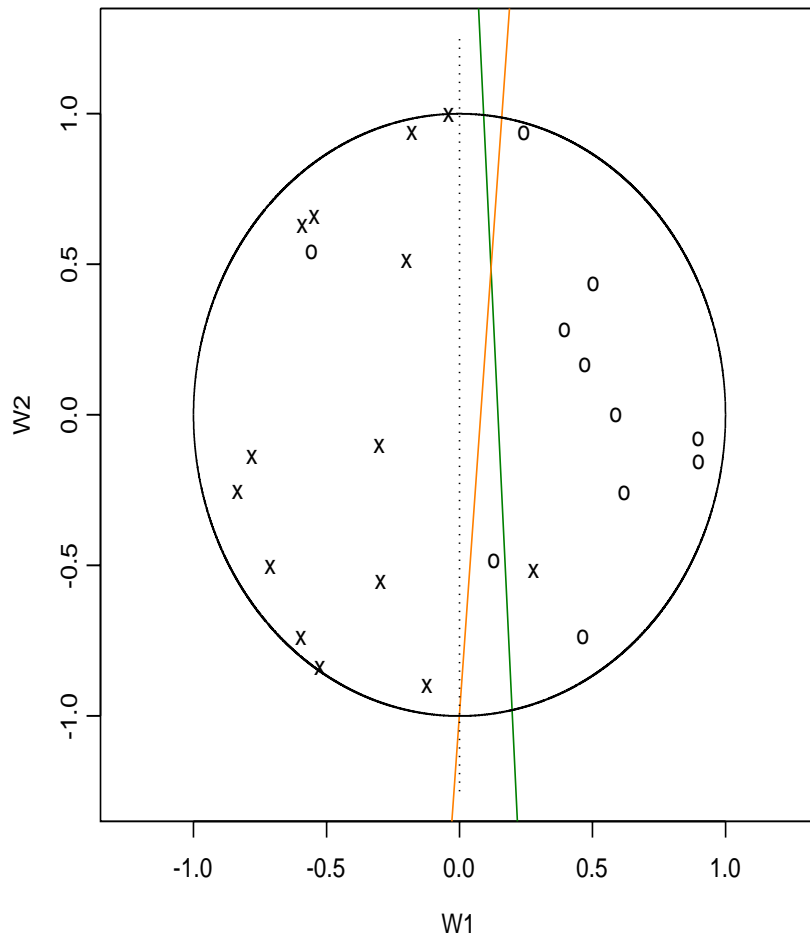


Figure 2: Plot of  $\psi$ -learning, SVM, and the true decision functions, represented by the purple, orange and vertical lines, respectively. The training sample size is 25.

## Performance of Algorithms

Table 1: Globality of Algorithm 1, checked in percent of agreement over 100 simulation replications.

Flip	C=1	Ave # iter	C=10 <sup>3</sup>	Ave # iter
0%	79%	2.43	96%	1.20
10%	77%	2.82	85%	2.07
20%	74%	2.85	84%	2.10

$\varepsilon = 0.01$ .

1. Algorithm 1: Converge within 3 steps on average.
2. Algorithm 2: Converge within 200 steps on average.

## Benchmark: SVM vs $\psi$ -learning

1. Cancer classification: Wisconsin Breast Cancer Data (WBCD).
2. Data: Each sample was assigned to a 9-dimensional vector of diagnostic characteristics, with each component being in the interval 1 to 10, with 1 corresponding to a normal state and 10 to a most abnormal state.
3. Diagnostic characteristics: 1) Clump Thickness, 2) Uniformity of Cell Size, 3) Uniformity of Cell Shape, 4) Marginal Adhesion, 5) ...
4. Examination: biopsy on a sample tissue.
5. Goal: Discriminate between benign and malignant samples.

## Benchmark Examples, Continued

1. For each example, randomly divide the dataset into two halves, for testing and training. The performance is averaged over 100 pairs of randomly selected training and testing datasets.
2. Search optimal  $C$  over grid points in  $[10^{-3}, 10^3]$ .
3. Testing errors: SVM:  $2.65 \pm 0.06\%$ ,  $\psi$ :  $2.38 \pm 0.06\%$ .  
#Support Vectors: SVM: 41.06,  $\psi$ : 29.80.

### Observations:

1. Testing correctness:  $\psi$ -learning better for most cases.
2. On average,  $\psi$ -learning reduces # support vectors of SVM. The percent of reduction, however, varies.

## Take Away Messages

1. Further developments.
  - (A) Applications: (1) Cancer gene classification (?), (2) Object tracking (Liu, Zheng, Shen, 2002), ....
  - (B) Choice of  $l$ : New learning methodology.
  - (C) Algorithms: Nonconvex optimization.
  - (D) Model selection: Adaptive choice of tuning parameter(s)  $C$  (Data perturbation).
  - (E) Multicategorical problems: (Liu & Shen, 2003).
  - (F) Choice of kernel and basis selection.
  - (G) More ....