

Optimization Problem

Primal Problem:

$$\begin{aligned} & \text{minimize } \frac{1}{2} \|w\|^2 \\ & \text{subject to } y_i \langle w, x_i \rangle + b - 1 \geq 0 \text{ for all } i \in \{1, 2, \dots, m\} \end{aligned}$$

Dual Problem:

Let $H_{ij} = y_i y_j \langle x_i, x_j \rangle$, then

$$\begin{aligned} & \text{maximize } -\frac{1}{2} \alpha^\top H \alpha + \sum_i \alpha_i \\ & \text{subject to } \sum_i \alpha_i y_i = 0 \text{ and } \alpha_i \geq 0 \text{ for all } i \in \{1, 2, \dots, m\}. \end{aligned}$$

Generalized Dual Problem:

$$\begin{aligned} & \text{maximize } -\frac{1}{2} \alpha^\top H \alpha + c^\top \alpha \\ & \text{subject to } A \alpha = 0 \text{ and } l \leq \alpha_i \leq u \end{aligned}$$

where A is a matrix, and l and u enforce bound constraints.

SimpleSVM

- Initialize with a *suitable* pair of points.
- **Step 1:**
 - Locate a violating point and add to the *active set*.
 - Ignore box constraints if any.
 - Solve the optimization problem for the active set.
- **Step 2:**
 - * If new solution satisfies box constraints we are done.
 - * Else remove the first box constraint violator.
- **Goto Step 2.**
- Repeat until no violators (**Goto Step 1**).

SimpleSVM is **not** a traditional active set method because it does not maintain a factorization of the null space of H (too expensive to compute).

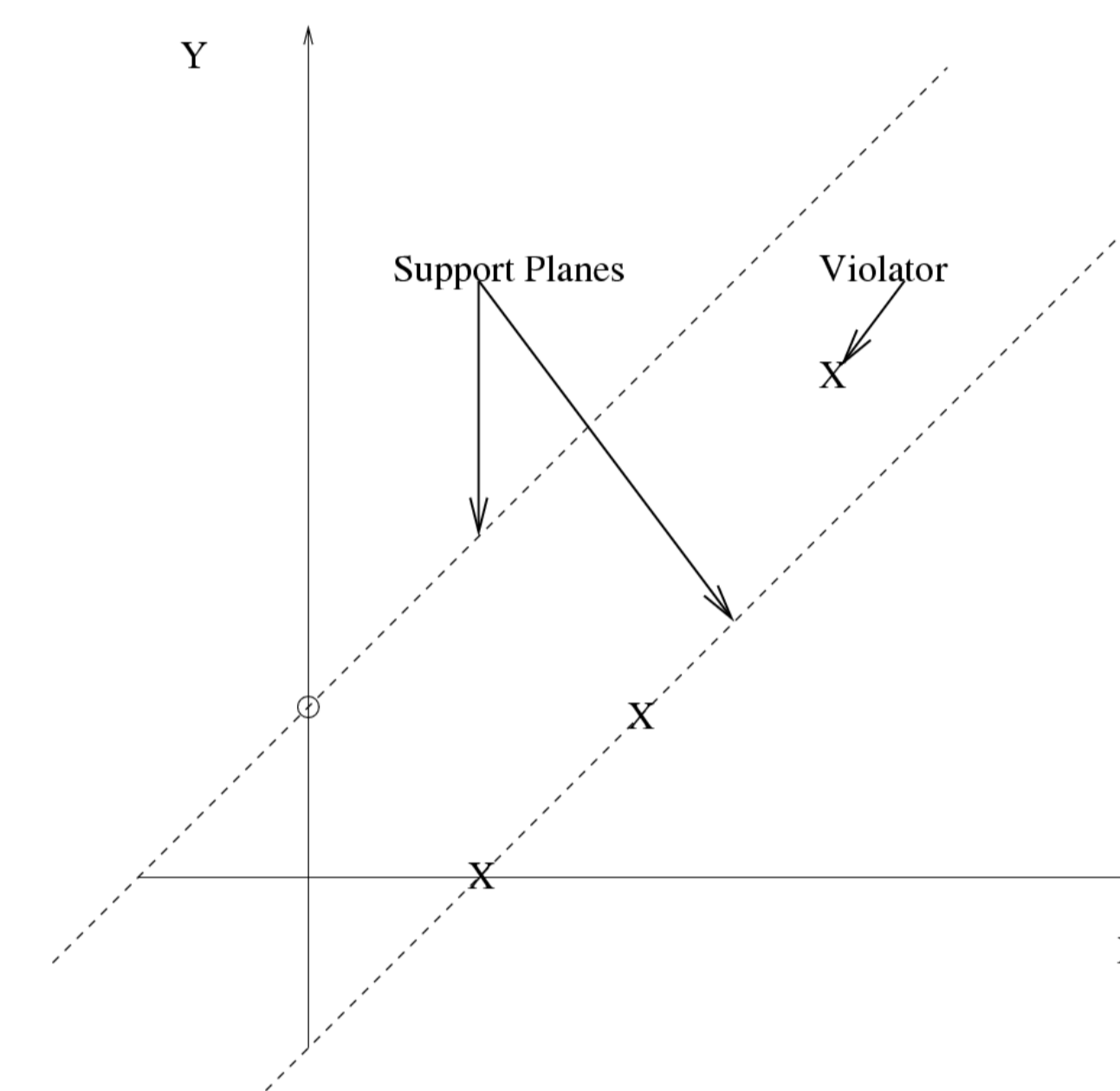
Issues

- A linear system has to be solved at every step. Can we exploit parallel CG solvers?
 - We store the kernel matrix of the active set. Two problems:
 - Storage grows as active set size increases
 - Linear solvers have problems with rank degenerate matrices
 - Need Reordering data points based on priority queue for faster access
 - Kernel cache can significantly help speedup.
 - Which points should get higher priority?
 - A SMO like cache does not work well
- Current implementation does not have a kernel cache.

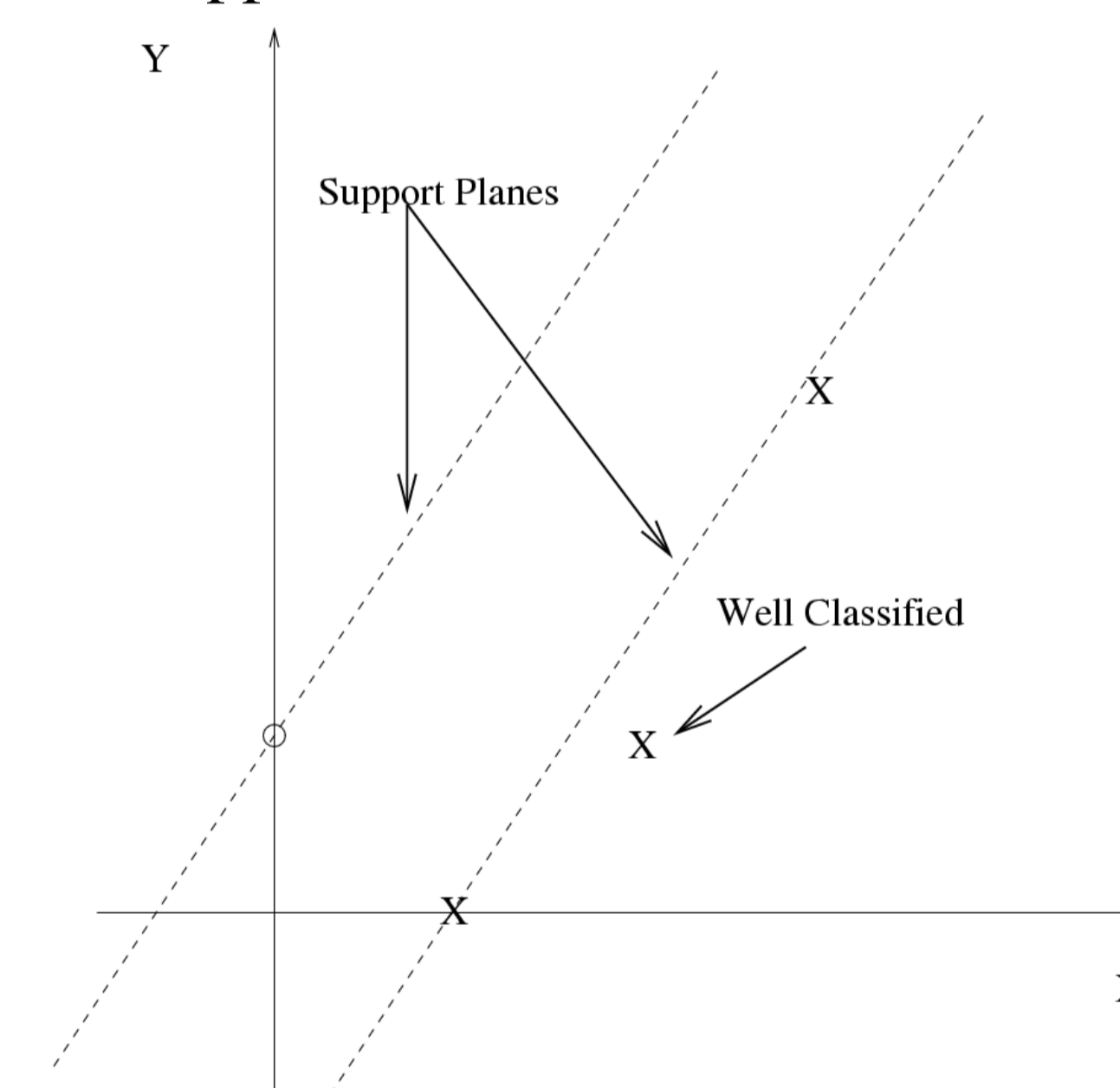
SimpleSVM - In Pictures

Without the Box Constraints

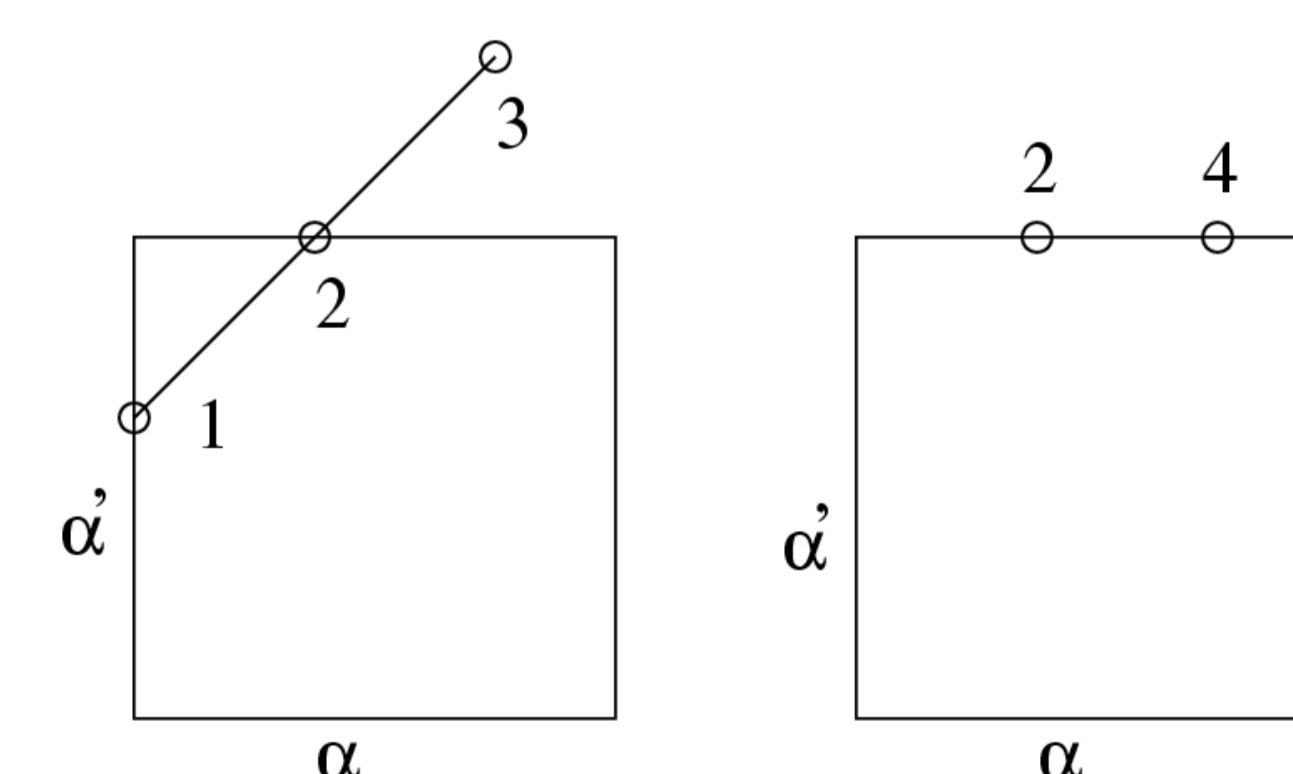
Consider a hard margin linear SVM with three support vectors and one violator:



Now the support plane has been shifted to pass through the violator. But along the way, a support vector has been dropped.



Box Constraints



- Move from current solution (point 1) by adding a new constraint α and optimize over (α, α') to reach point 3.
- Shrink solution by enforcing bound constraints (point 2).
- Now optimize over α to reach point 4 and so on.

SimplerSVM

Choosing Initial Points:

- Randomized strategies.
- Find a *good* pair with high probability.

Choosing Violators:

- Greedy version - do a sequential pass through the data.
- Smart version - order points in a priority queue.
- The priority of a point depends on:
 - The magnitude of $|y_i f(x_i) - 1|$
 - The number of times a point has been examined before
 - Whether the point was an active support vector
- Pick the highest point of the queue to examine
- Well classified points quickly settle to the bottom of the queue.

Solving the Linear Problem:

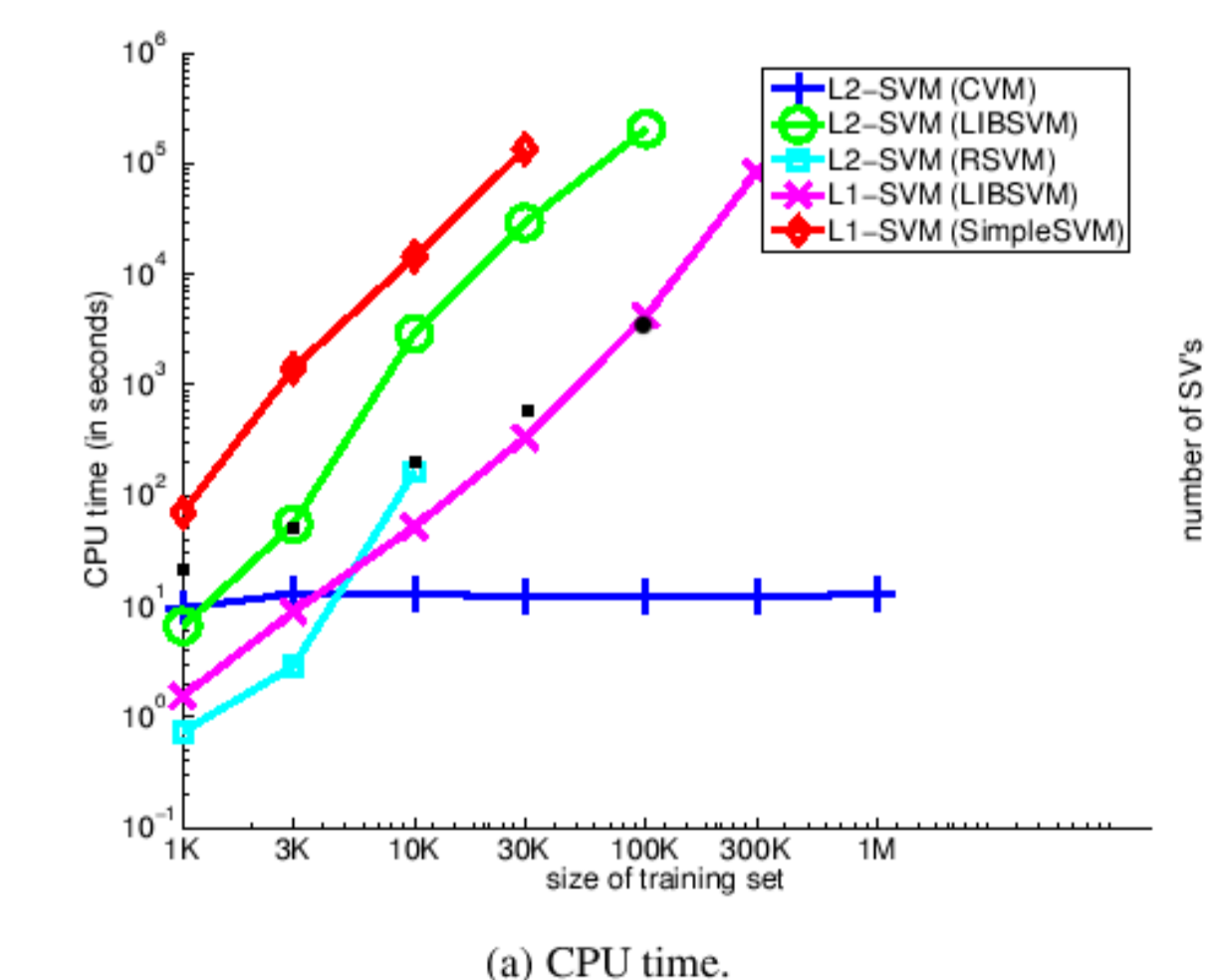
- Need to solve a linear problem ($Ax = b$) for each iteration.
- Kernel matrices are generally rank-degenerate with fast decaying eigenvalues.
- Exploit low rank using a CG solver: number of steps = *effective rank* of the matrix.
- Efficient kernel matrix times vector computation can speed up linear solvers.

Convergence:

- Few sweeps thru the dataset suffice.
- Speed of convergence: Linear.

Very Preliminary Experiments

We tested it out on the checkers dataset. The black dots indicates the performance of SimplerSVM vs various other algorithms (graph source: Tsang, Kwok, Cheung 2005).



Open source code and more results and comparison results coming soon: <http://lineal.developer.nicta.com.au>

Acknowledgements

Originally SimpleSVM was developed in collaboration with M Narasimha Murty, Gaelle Loosli, and Stéphane Canu. Simon Burton helped with the implementation.