Mathematical Programming for Multiple Kernel Learning

Alex Zien

Fraunhofer FIRST.IDA, Berlin, Germany
Friedrich Miescher Laboratory, Tübingen, Germany

07. July 2009

Mathematical Programming Stream, EURO 2009
Bonn, Germany



Friedrich Miescher Laboratory
of the Max Planck Society



The Goal: Multiple Kernel Learning

- Given N data items $\mathbf{x}_n \in \mathcal{X}$ $(n = 1, \ldots, N)$,
- with associated labels $y_n \in \{-1, +1\}$ $(n = 1, \ldots, N)$,
- P "feature maps" $\Phi_p : \mathcal{X} \mapsto \mathbb{R}^{d_p}$ $(p = 1, \ldots, P)$,
- and a "regularization constant" $0 < C \in \mathbb{R}$,

efficiently solve the **Muliple Kernel Learning (MKL)** problem

$$\min_{\mathbf{w}, b, \xi, \boldsymbol{\beta}} \quad \frac{1}{2} \sum_{p=1}^{P} \beta_p \left\| \mathbf{w}_p \right\|^2 \quad + \quad C \left\langle \mathbf{1}, \xi \right\rangle$$

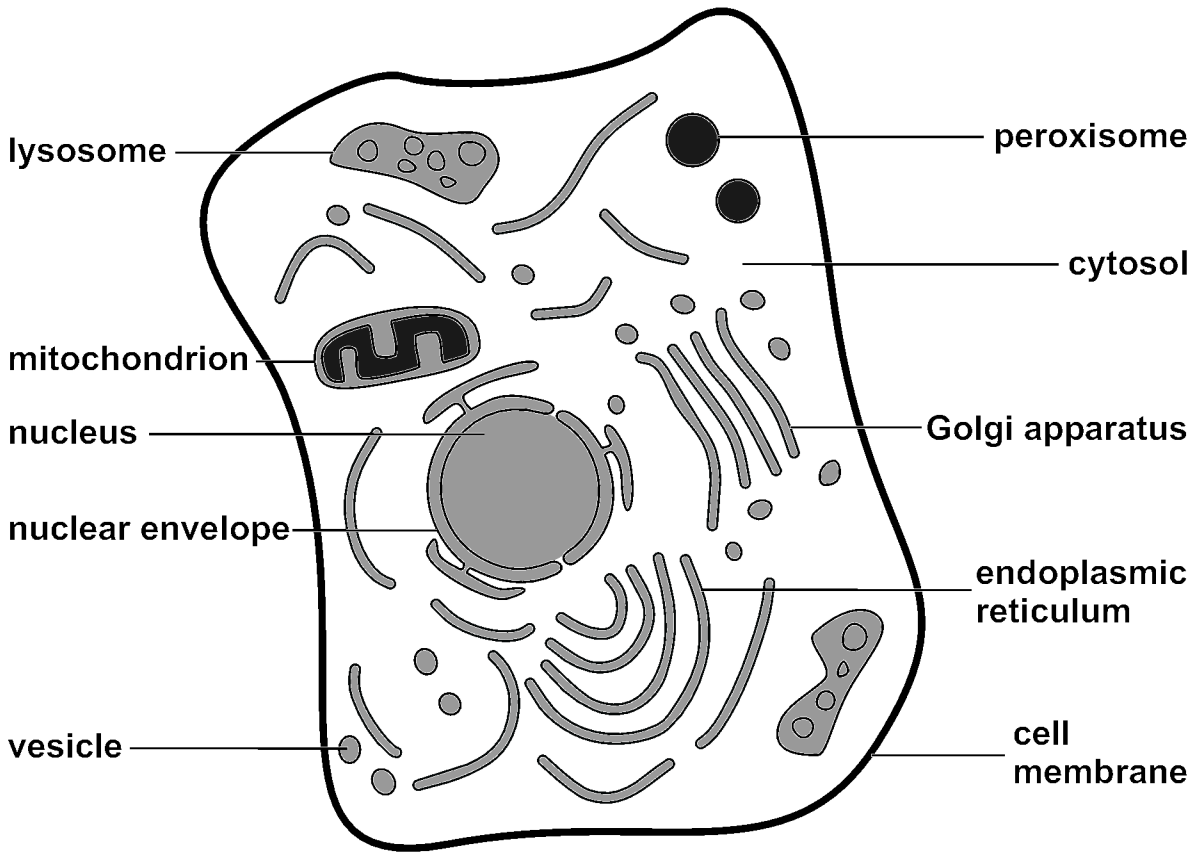
$$\text{s.t.} \quad 0 \leq \xi \in \mathbb{R}^N, \quad \forall n : y_i \left(\sum_{p=1}^{P} \beta_p \left\langle \mathbf{w}_p, \Phi_p(\mathbf{x}) \right\rangle \right) \geq 1 - \xi_n,$$

$$0 \leq \boldsymbol{\beta} \in \mathbb{R}^p, \quad \left\langle \mathbf{1}, \boldsymbol{\beta} \right\rangle \leq 1.$$

Application: Subcellular Localization of Proteins

Input: protein sequence

Output: target location of the protein



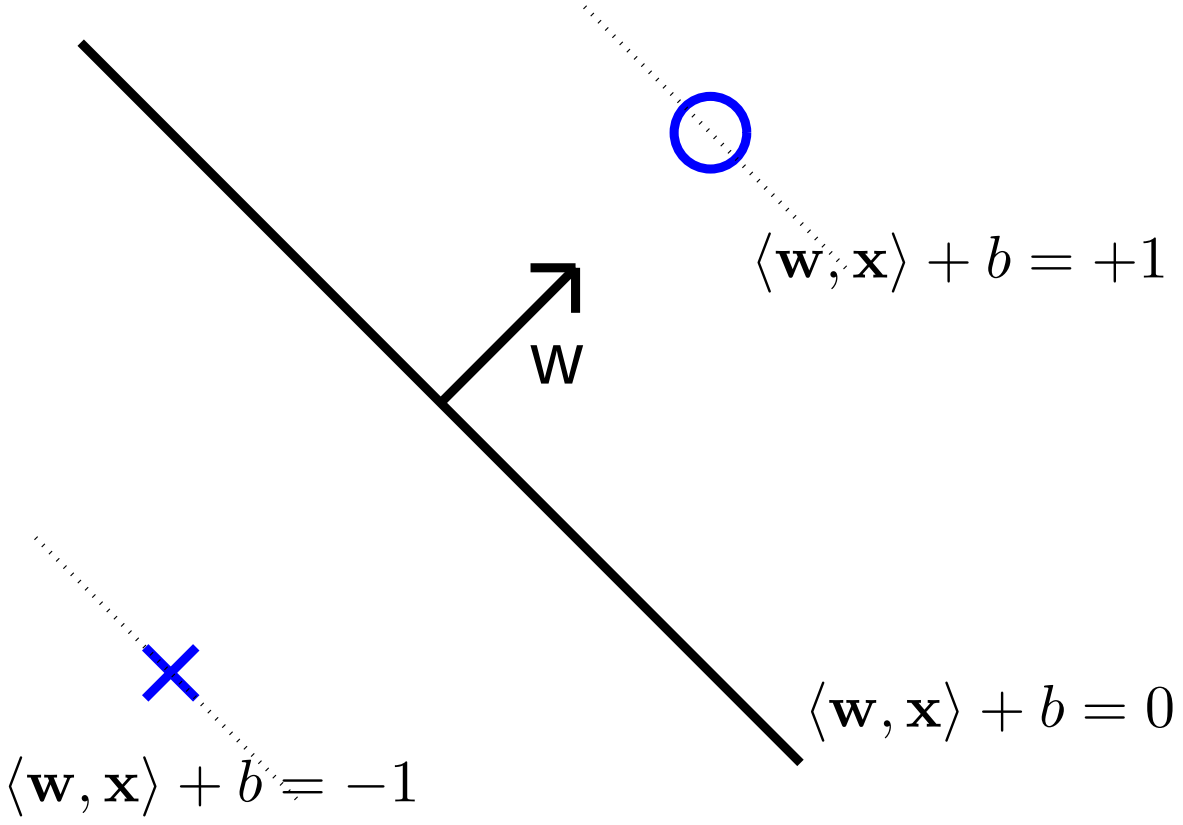
[image from "A primer on molecular biology" in "Kernel Methods in Computational Biology", MIT Press, 2004]

Outline

1. The Goal: MKL for Classification
2. Why Optimize This Problem?
   - Support Vector Machines (SVMs)
   - Non-Linearity via Kernels
   - A Large Margin MKL Model
3. Optimization for MKL
4. Application: Predicting Protein Subcellular Localization
5. Take Home Messages

Support Vector Machine (SVM): Maximum Margin

SVM:
maximum margin
classifier



$$\min_{\mathbf{w},b} \quad \underbrace{\frac{1}{2}\langle \mathbf{w}, \mathbf{w} \rangle}_{\text{regularizer}} \quad s.t. \quad \underbrace{y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1}_{\text{data fitting}}$$

Support Vector Machine: Dual Is QP

Primal:

$$\begin{aligned} \min_{\mathbf{w},b,(\xi_i)} \quad & \tfrac{1}{2}\langle \mathbf{w}, \mathbf{w} \rangle + C \textstyle\sum_i \xi_i \\ s.t. \quad & y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1 - \xi_i, \quad \xi_i \geq 0 \end{aligned}$$

at optimum: $\xi_i = \ell(\mathbf{x}_i, y_i) := \max\{1 - y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b), 0\}$

Dual:

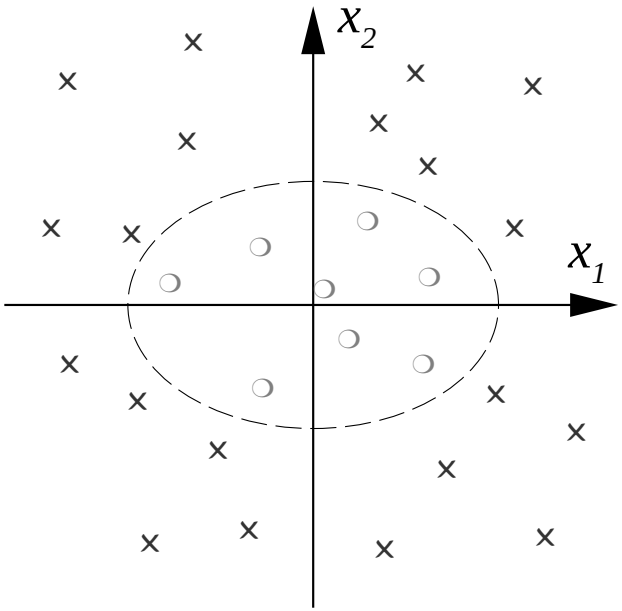
$$\begin{aligned} \min_{\alpha} \quad & \tfrac{1}{2} \textstyle\sum_i \sum_j \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle - \sum_i \alpha_i \\ s.t. \quad & \mathbf{0} \leq \alpha \leq C\mathbf{1} \\ & \mathbf{y}^\top \alpha = 0 \end{aligned}$$

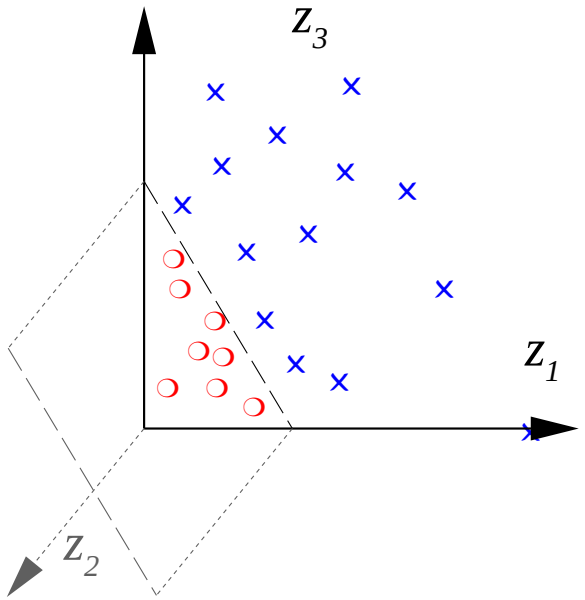
only needs pairwise dot products!

Non-Linear Mappings

Example: All Degree 2 Monomials for a 2D Input

$$\begin{aligned} \Phi : \mathbb{R}^2 &\rightarrow \mathbb{R}^3 =: \mathcal{H} \quad (\text{“Feature Space”}) \\ (x_1, x_2) &\mapsto (z_1, z_2, z_3) := (x_1^2, \sqrt{2}\, x_1 x_2, x_2^2) \end{aligned}$$





Kernel Trick

Example: All Degree 2 Monomials for a 2D Input

$$
\begin{aligned}
\left\langle \Phi(\mathbf{x}), \Phi(\mathbf{x}') \right\rangle &= \left\langle (x_1^2, \sqrt{2}\, x_1 x_2, x_2^2), ({x'}_1^2, \sqrt{2}\, x'_1 x'_2, {x'}_2^2) \right\rangle \\
&= x_1^2 {x'}_1^2 + 2\, x_1 x_2 x'_1 x'_2 + x_2^2 {x'}_2^2 \\
&= \left(x_1 x'_1 + x_2 x'_2 \right)^2 \\
&= \left\langle \mathbf{x}, \mathbf{x}' \right\rangle^2 \\
&=: k(\mathbf{x}, \mathbf{x}')
\end{aligned}
$$

$\Rightarrow$ the dot product in $\mathcal{H}$ can be computed in $\mathbb{R}^2$

more information: "Lerning with Kernels"; Schölkopf, Smola; MIT Press]

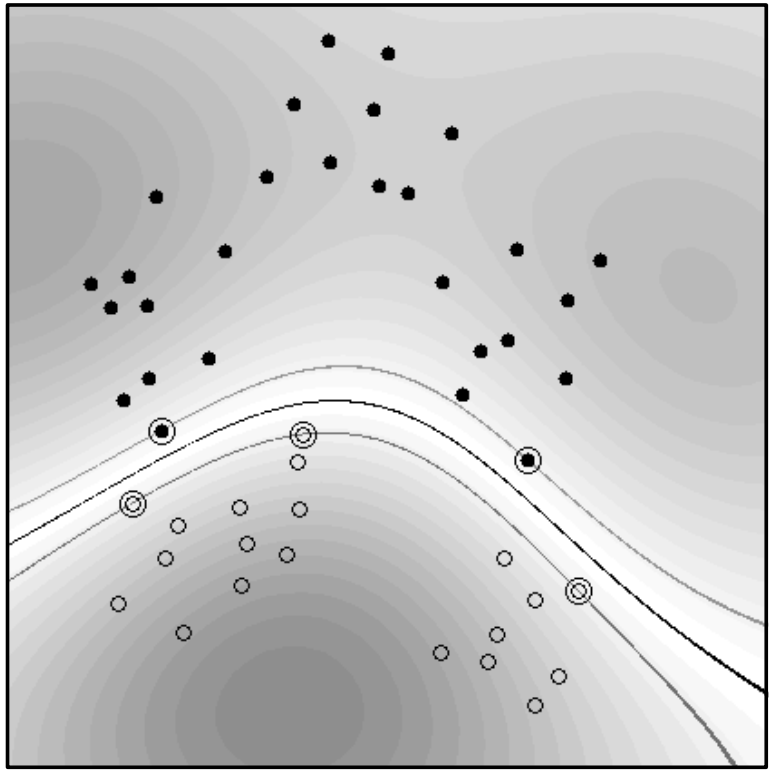
Relevant SVM Facts

- dual is QP
- **support vectors**

$\Rightarrow$ sparse solution

$\Rightarrow$ efficient optimization

- non-linear via kernels



So how to choose the best kernel function?

A Multiple Kernel Learning (MKL) Model

MKL Model: weighted linear mixture of P feature spaces

$$
\begin{aligned}
\mathcal{H}_{\boldsymbol{\gamma}} &\leftarrow \gamma_1\mathcal{H}_1 \oplus \gamma_2\mathcal{H}_2 \oplus \ldots \oplus \gamma_P\mathcal{H}_P \\
\Phi_{\boldsymbol{\gamma}}(\mathbf{x}) &\leftarrow \left(\gamma_p\Phi_p(\mathbf{x})^\top\right)^\top_{p=1,\ldots,P} \\
k_{\boldsymbol{\gamma}}(\mathbf{x},\mathbf{x}') &\leftarrow \sum_{p=1}^{P}\left\langle\gamma_p\Phi_p(\mathbf{x}),\gamma_p\Phi_p(\mathbf{x}')\right\rangle = \sum_{p=1}^{P}\gamma_p^2 k_p(\mathbf{x},\mathbf{x}') \\
\mathbf{w}_{\boldsymbol{\gamma}} &\leftarrow \left(\gamma_p\mathbf{w}_p^\top\right)^\top_{p=1,\ldots,P} \\
f_{\boldsymbol{\gamma}}(\mathbf{x}) &\leftarrow \left\langle\mathbf{w}_{\boldsymbol{\gamma}},\Phi_{\boldsymbol{\gamma}}(\mathbf{x})\right\rangle = \sum_{p=1}^{P}\gamma_p^2\left\langle\mathbf{w}_p,\Phi_p(\mathbf{x})\right\rangle
\end{aligned}
$$

Goal: learn mixing coefficients $\boldsymbol{\gamma} = (\gamma_p)_{p=1,\ldots,P}$ along with $\mathbf{w}, \mathbf{b}$

Large Margin MKL

Plugging it into the SVM

$$\begin{aligned} \min_{\mathbf{w},b,\xi,\boldsymbol{\gamma}} \quad & \frac{1}{2}\left\|\mathbf{w}_{\boldsymbol{\gamma}}\right\|^2 + C\sum_{i=1}^{N}\xi_i \\ \text{s.t.} \quad & \forall i : \xi_i = \ell\left(\left\langle \mathbf{w}_{\boldsymbol{\gamma}}, \Phi_{\boldsymbol{\gamma}}(\mathbf{x})\right\rangle + b, y_i\right) \end{aligned}$$

yields:

$$\begin{aligned} \min_{\mathbf{w},b,\xi,\boldsymbol{\gamma}} \quad & \frac{1}{2}\sum_{p=1}^{P}\gamma_p^2\left\|\mathbf{w}_p\right\|^2 + C\sum_{i=1}^{N}\xi_i \\ \text{s.t.} \quad & \forall i : \xi_i = \ell\left(\sum_{p=1}^{P}\gamma_p^2\left\langle \mathbf{w}_p, \Phi_p(\mathbf{x})\right\rangle, y_i\right) \end{aligned}$$

for convenience we substitute $\beta_p := \gamma_p^2$

MKL: Retain Convexity and Feasibility

Problems: (i) products $\beta_p \mathbf{w}_p$ non-convex; (ii) $\beta \to +\infty$

$$\min_{\boldsymbol{\beta},\mathbf{w},\mathbf{b},\xi} \quad \frac{1}{2}\sum_{p=1}^{P} \beta_p \left\|\mathbf{w}_p\right\|^2 + C\sum_{i=1}^{N} \xi_i$$
$$\text{s.t.} \quad \forall i : \xi_i = \ell\left(\sum_{p=1}^{P} \beta_p \left\langle \mathbf{w}_p, \Phi_p(\mathbf{x}_i)\right\rangle, y_i\right)$$

Solutions: (i) change of variables $\mathbf{v}_p := \beta_p \mathbf{w}_p$; (ii) constrain $\boldsymbol{\beta}$

$$\min_{\boldsymbol{\beta},\mathbf{v},\mathbf{b},\xi} \quad \frac{1}{2}\sum_{p=1}^{P} \frac{1}{\beta_p} \left\|\mathbf{v}_p\right\|^2 + C\sum_{i=1}^{N} \xi_i$$
$$\text{s.t.} \quad \forall i : \xi_i = \ell\left(\sum_{p=1}^{P} \left\langle \mathbf{v}_p, \Phi_p(\mathbf{x}_i)\right\rangle, y_i\right), \quad \left\|\boldsymbol{\beta}\right\|_1 \leq 1$$

[Zien, Ong; ICML 2007] [Rakotomamonjy, Bach et al.; ICML 2007]

MKL Lagrange Dual

For hinge loss, $\ell(t, y) := \max\{0, 1 - yt\}$:

$$\begin{aligned} \min_{\alpha} \quad & \gamma - \sum_i \alpha_i \\ \text{s.t.} \quad & \forall i : 0 \leq \alpha_i \leq C \\ & \sum_i y_i \alpha_i = 0 \\ & \forall p : \gamma \geq \frac{1}{2} \sum_i \sum_j y_i y_j \alpha_i \alpha_j \left\langle \Phi_p(\mathbf{x}_i), \Phi_p(\mathbf{x}_j) \right\rangle \end{aligned}$$

- quadratically constrained quadratic program (QCQP)
- for single kernel, reduces to standard SVM
- may be solved by off-the-shelf solvers (eg CPLEX)

[Zien and Ong; ICML 2007]

MKL as SILP

Semi-Infinite Linear Program (SILP):

$$
\begin{aligned}
\max_{\boldsymbol{\beta},\theta} \quad & \theta \\
\text{s.t.} \quad & \textstyle\sum_p \beta_p = 1, \quad \beta_p \geq 0 \\
& \forall \alpha \in \mathcal{S}: \;\; \theta \leq \frac{1}{2} \sum_p \beta_p \left\| \mathbf{w}_p(\alpha) \right\|^2 - \sum_i \alpha_i \\
\textit{with} \quad & \mathbf{w}_p(\alpha) := \textstyle\sum_i y_i \alpha_i \Phi_p(\mathbf{x}_i)
\end{aligned}
$$

Infinite set of constraints indexed by:

$$
\mathcal{S} \;\; := \;\; \left\{ \alpha \;\middle|\; \begin{array}{l} \forall i : 0 \leq \alpha_i \leq C \\ \textstyle\sum_i y_i \alpha_i = 0 \end{array} \right\}
$$

[Sonnenburg, Rätsch, Schäfer; NIPS 2006]

MKL Wrapper by Column Generation (1)

1. initialize LP with minimal set of constraints:

$$\sum_p \beta_p = 1, \quad \beta_p \geq 0$$

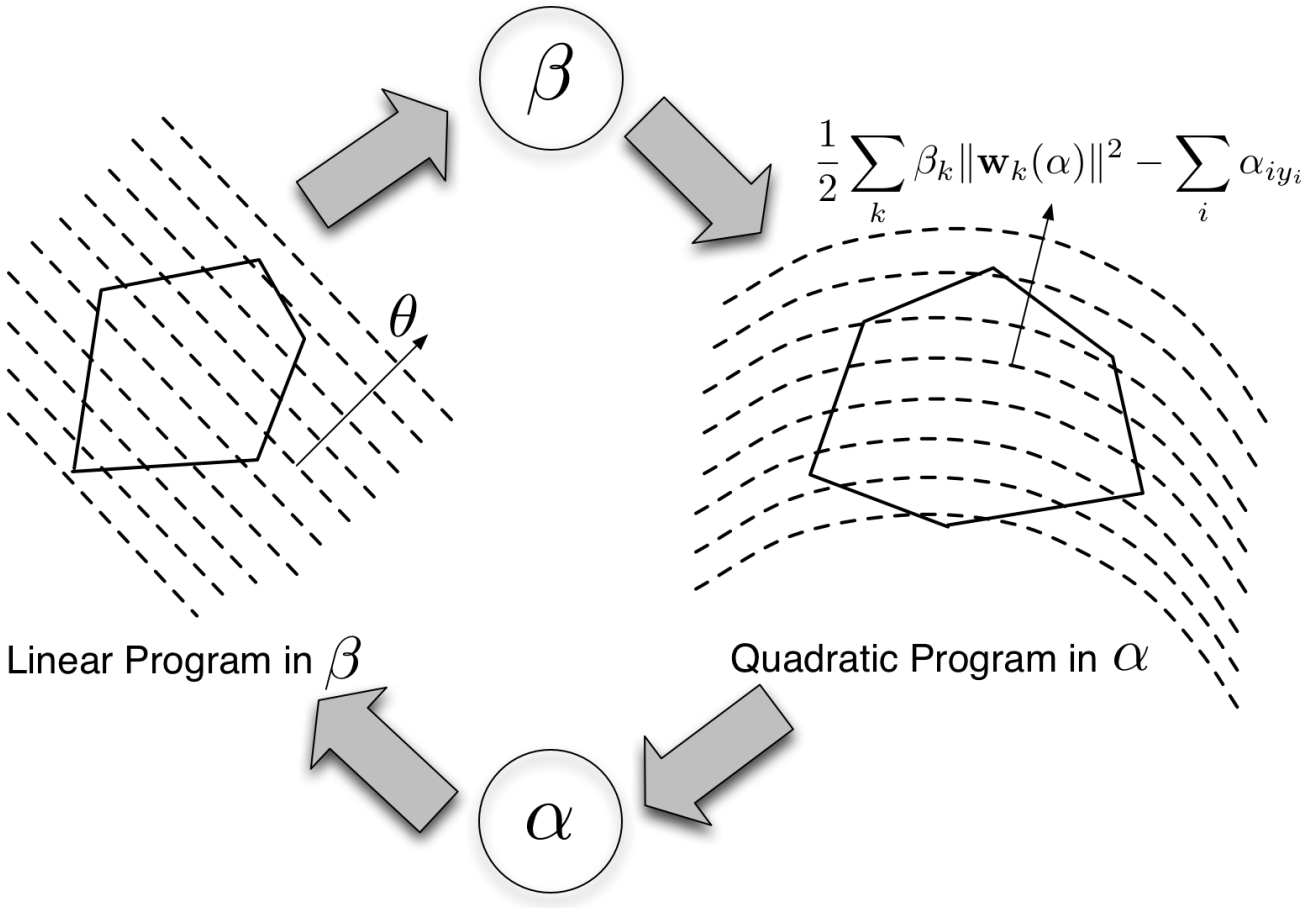
2. initialize $\boldsymbol{\beta}$ to feasible value (eg $\beta_p = 1/P$)
3. iterate:
   - for given $\boldsymbol{\beta}$, find most violated constraint:

     minimize $\frac{1}{2}\sum_p \beta_p \left\|\mathbf{w}_p(\alpha)\right\|^2 - \sum_i \alpha_i$ st $\alpha \in \mathcal{S}$

     $\Rightarrow$ solve single-kernel SVM!
   - add this constraint to LP
   - solve LP to obtain new mixing coefficients $\boldsymbol{\beta}$

$\Rightarrow$ just need wrapper around single-kernel method

MKL Wrapper by Column Generation (2)



Alternate between solving an LP for β and a QP for α.

Efficiency Comparison

The SILP-based wrapper is much more efficient than the QCQP.
Reason: it exploits efficiency of existing SVM solvers.

The same holds for projected-gradient approach.
[Rakotomamonjy, Bach et al.; ICML 2007]

Idea to gain still more efficieny:

- each constraint (gradient) requires full SVM training
- although most constraints are auxilliary anyway
  ie, they are based on suboptimal β
- could use "approximate constraints"
  ie, based on approximately trained SVMs

Chunking Approach

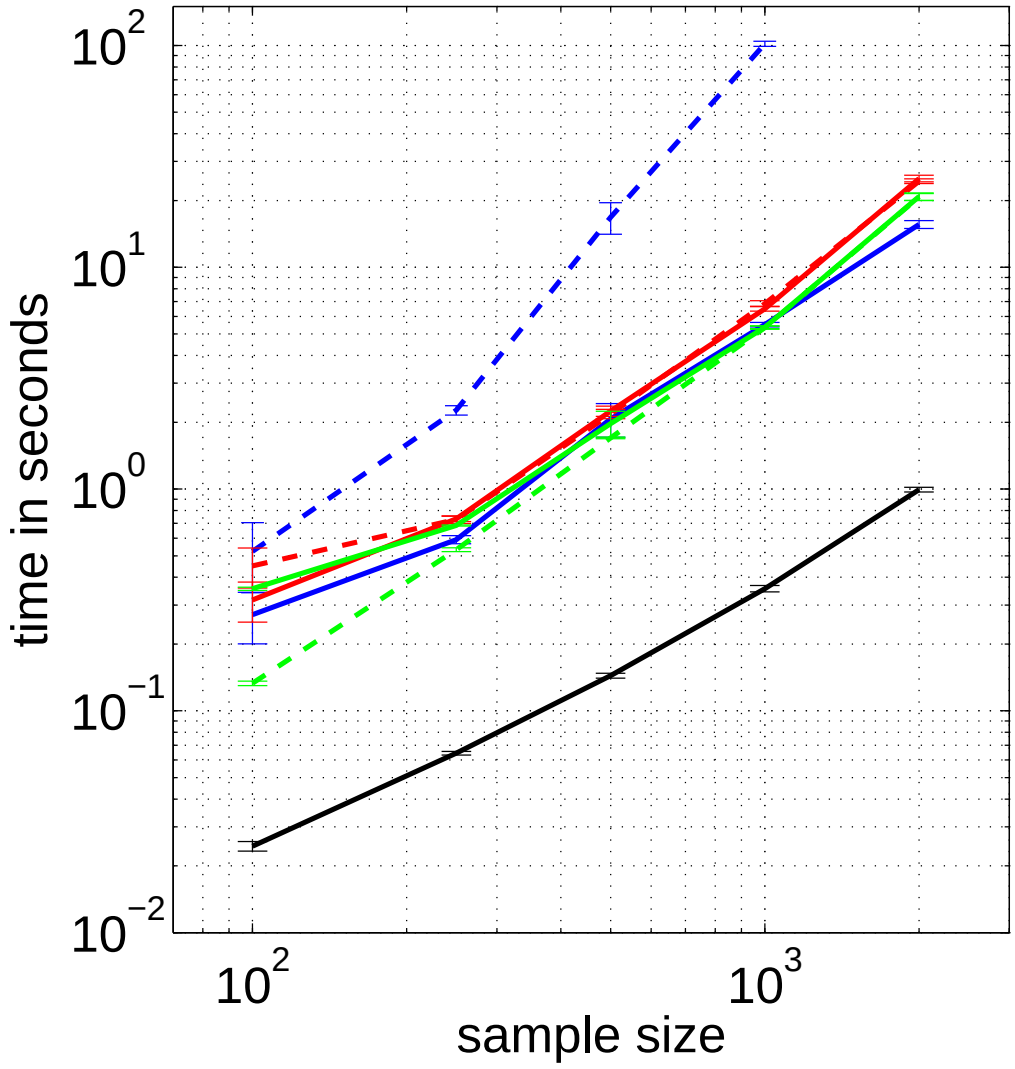
Idea: interleave β-update with SVM chunking

- SVM chunking: repeatedly optimize subproblem
  - select subset of α variables ("chunk")
  - optimize on this subset
  - clever subset selection $\Rightarrow$ efficiency
  - [Joachims; "Making Large–Scale SVM Learning Practical"]
- after each chunk, generate constraint and update β
  [Sonnenburg, Rätsch, Schäfer; NIPS 2006]
- thereafter, SVM optimization can simply continue
  (similar to a "hot start")

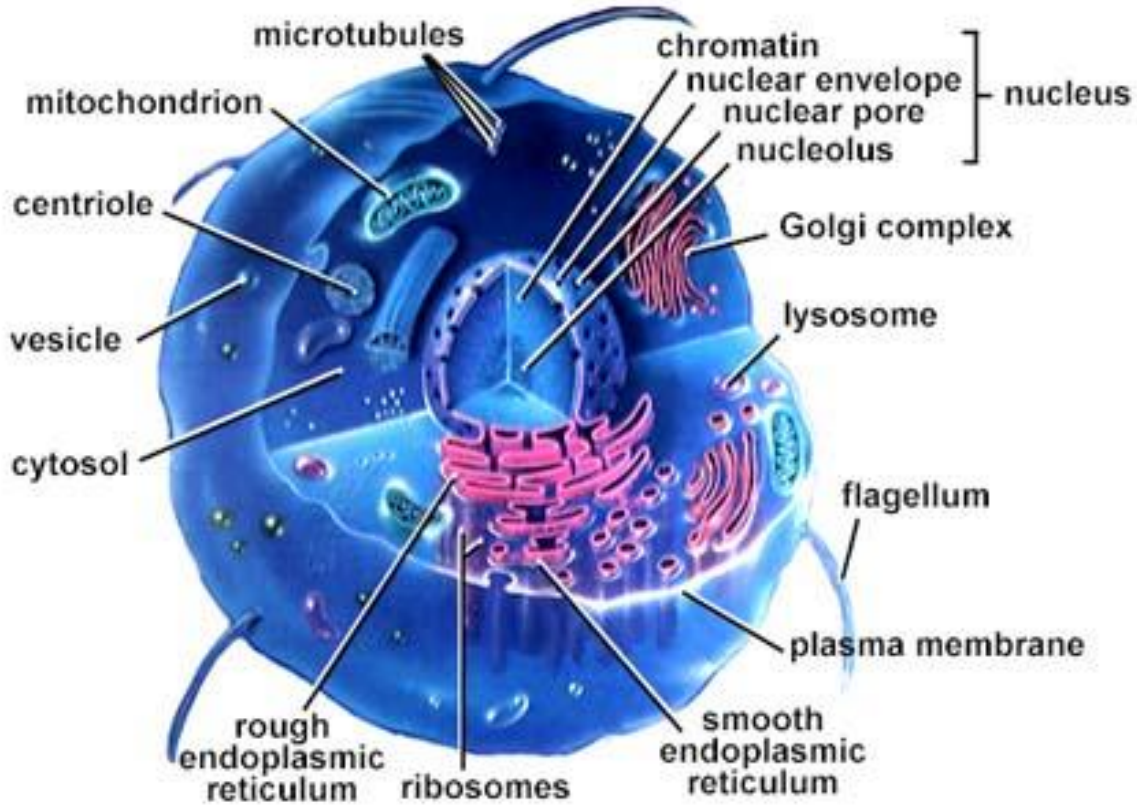
[implemented in `Shogun`, `http://www.shogun-toolbox.org/`]

Efficiency Comparison

- dashed blue: wrapper
- full black: SVM (single kernel)
- others: chunking MKL



Blue Picture of a Cell



Input: protein sequence

Output: target location of the protein

[image taken from the internet]

List of 69 Kernels

64 = 4*16 Motif kernels

- 4 subsequences (all, last 15, first 15, first 60)
- $16 = 2^{5-1}$ patterns of length 5 (•,○,○,○,○)

3 BLAST similarity kernels

1. linear kernel on E-values
2. Gaussian kernel on E-values, width 1000
3. Gaussian kernel on log E-values, width 1e5

2 phylogenetic kernels

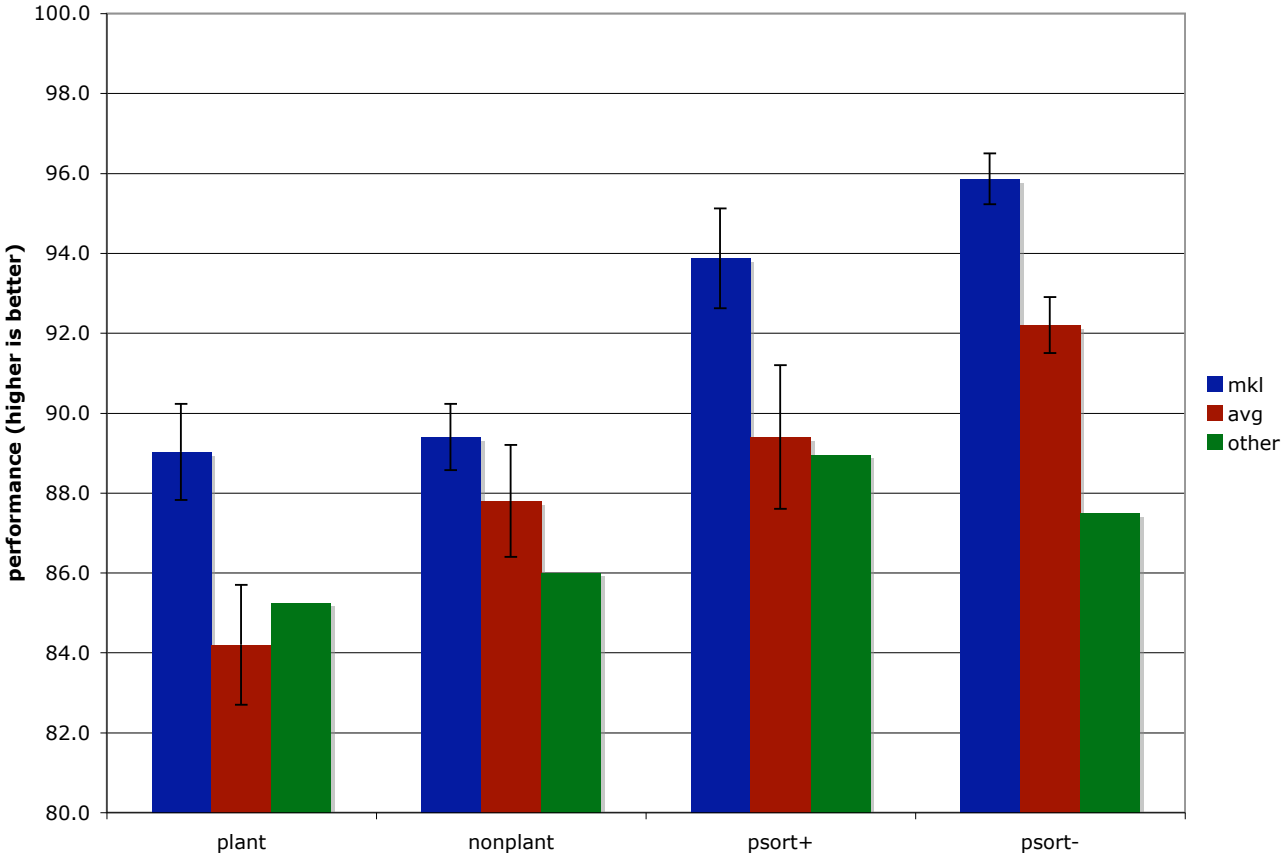
1. linear kernel
2. Gaussian kernel, width 300

[all described in: C. S. Ong, A. Zien; WABI 2008]

Better Than Previous Work

MCC [%] (plant, nonplant)

F1 [%] (psort+, psort-)



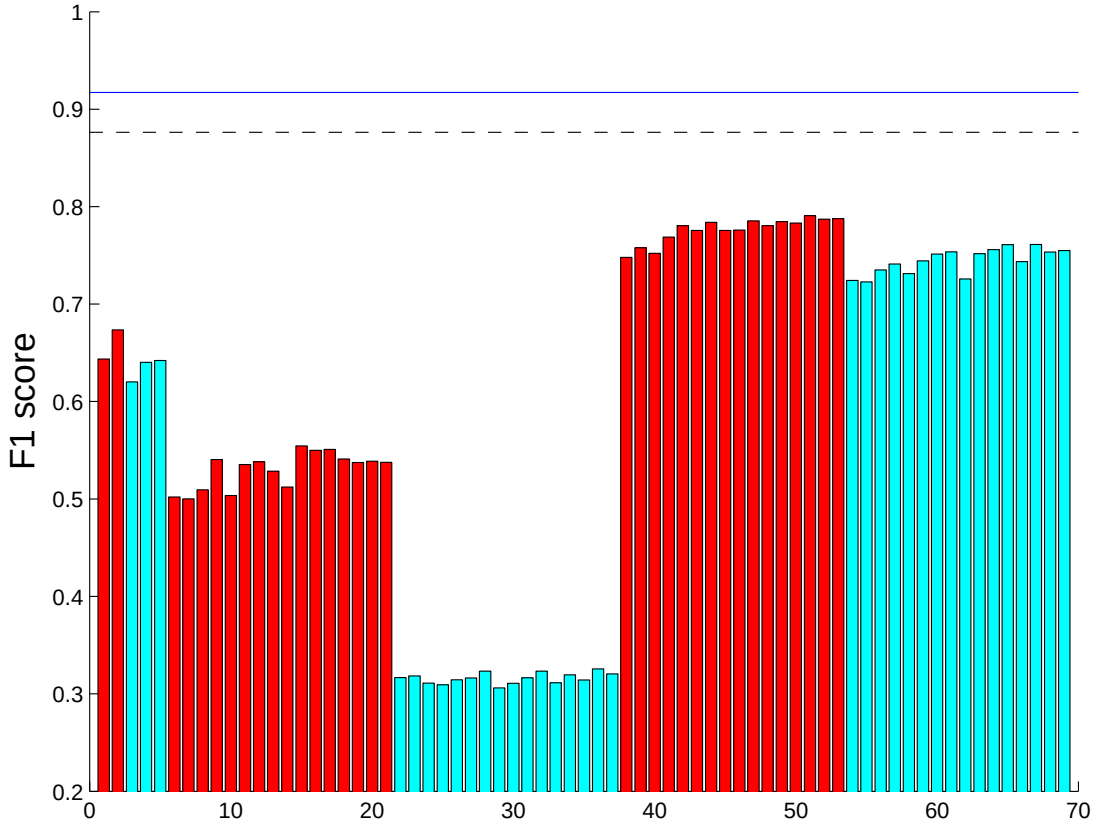
MCMKL unweighted sum of kernels TargetLoc / PSORTb v2.0

Better Than Single Kernels and Than Average Kernel

——, MKL

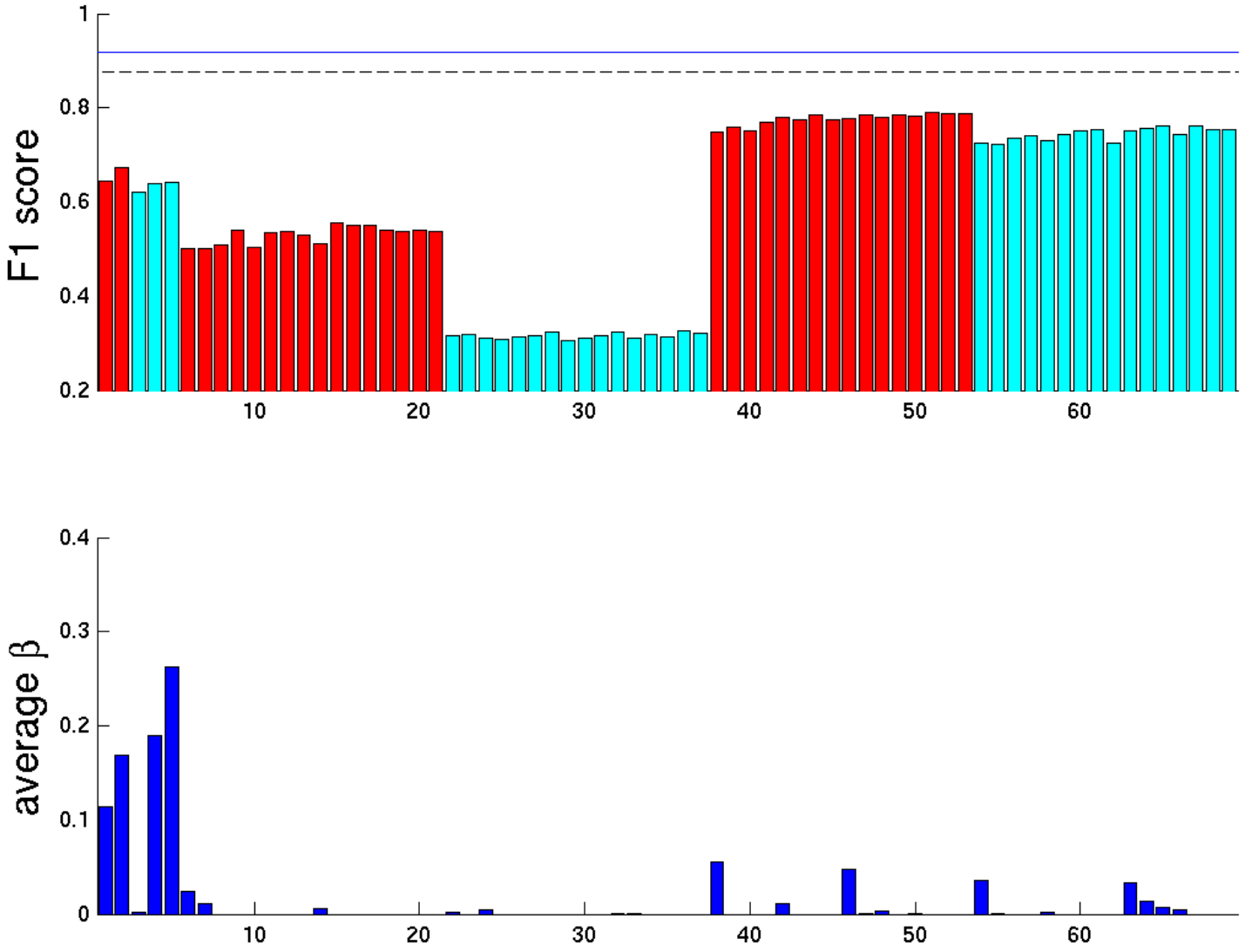
- - - -, sum with uniform weight

bars, single kernel



1. pylogenetic profiles
2. BLAST similarities
3. motifs, complete sequence
4. motifs, last 15 AAs
5. motifs, first 15 AAs
6. motifs, first 60 AAs

Weights $\not\propto$ Single-Kernel Performances



What You Should Take Home From This Talk

- SVMs are QPs, but require special solvers for efficiency
- MKL amounts to a QCQP
- ... but formulating it as SILP is more efficient,
  because it reduces to several (highly optimized) SVMs
- ... but it can be solved even more efficiently
  by *interleaving* SVM training with the outer optimization
- yes, we need the speed:
  faster $\Rightarrow$ more training data $\Rightarrow$ higher accuracy

Questions?