Standard Learning with Kernels

user → kernel $K$ → algorithm → $h$ → sample
Learning Kernel Framework

user → kernel family $\mathcal{K}$ → algorithm → $(K, h)$ → sample
This Part

- Early attempts
- General learning kernel formulation
  - linear, non-negative combinations
  - non-linear combinations and alternative formulations
- Alignment-based algorithms
- Ensemble combinations
Minimize Different Criteria

(Weston et al., 2000; Chapelle et al., 2002)

- **Wrapper method**: alternate a call to an SVM solver and an update of the kernel parameters.
  - solve SVM to get $\alpha^*$
  - gradient step over criterion $T$ to select kernel parameters:
    - margin criterion $T = \frac{R^2}{\rho^2}$
    - span criterion $T = \frac{1}{m} \sum_{i=1}^{m} \Theta(\alpha_i^* S_i^2 - 1)$.  

\[T = \frac{R^2}{\rho^2} \]
Reality Check

(Chapelle et al., 2002)

Selecting the width of a Gaussian kernel and the SVM parameter $C$.

**Accuracy:**

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Cross-validation</th>
<th>$R^2 / \gamma^2$</th>
<th>Span-bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>Breast cancer</td>
<td>26.04 ± 4.74</td>
<td>26.84 ± 4.71</td>
<td>25.59 ± 4.18</td>
</tr>
<tr>
<td>Diabetis</td>
<td>23.53 ± 1.73</td>
<td>23.25 ± 1.7</td>
<td>23.19 ± 1.67</td>
</tr>
<tr>
<td>Heart</td>
<td>15.95 ± 3.26</td>
<td>15.92 ± 3.18</td>
<td>16.13 ± 3.11</td>
</tr>
<tr>
<td>Thyroid</td>
<td>4.80 ± 2.19</td>
<td>4.62 ± 2.03</td>
<td>4.56 ± 1.97</td>
</tr>
<tr>
<td>Titanic</td>
<td>22.42 ± 1.02</td>
<td>22.88 ± 1.23</td>
<td>22.5 ± 0.88</td>
</tr>
</tbody>
</table>

**Speed:**

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Cross-validation</th>
<th>$R^2 / \gamma^2$</th>
<th>Span-bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>Breast cancer</td>
<td>500</td>
<td>14.2</td>
<td>7</td>
</tr>
<tr>
<td>Diabetis</td>
<td>500</td>
<td>12.2</td>
<td>9.8</td>
</tr>
<tr>
<td>Heart</td>
<td>500</td>
<td>9</td>
<td>6.2</td>
</tr>
<tr>
<td>Thyroid</td>
<td>500</td>
<td>3</td>
<td>11.6</td>
</tr>
<tr>
<td>Titanic</td>
<td>500</td>
<td>6.8</td>
<td>3.4</td>
</tr>
</tbody>
</table>
Kernel Learning & Feature Selection

- Linear kernels:
  \[ K(x_i, x_j) = \sum_{k=1}^{p} \mu_k x_i^k x_j^k, \quad \mu_k \geq 0, \quad \sum_{k=1}^{p} (\mu_k)^q \leq \Lambda \]

- Polynomial kernels:
  \[ K(x_i, x_j) = \left( 1 + \sum_{k=1}^{p} \mu_k x_i^k x_j^k \right)^d, \quad \mu_k \geq 0, \quad \sum_{k=1}^{p} (\mu_k)^q \leq \Lambda \]

- Alternate between solving SVM and gradient step
  - the margin bound: \[ R^2 / \rho^2 , \] (Weston et al., 2000)
  - the SVM dual: \[ 2\alpha^\top 1 - \alpha^\top Y^\top K_\mu Y \alpha , \] (Grandvalet & Canu, 2003).
Feature Selection: Reality Check

(Weston et al., 2000; Chapelle et al., 2002)

- Comparison with existing methods:

Figure 1: A comparison of feature selection methods on (a) a linear problem and (b) a nonlinear problem both with many irrelevant features. The $x$-axis is the number of training points, and the $y$-axis the test error as a fraction of test points.
This Part

- Early attempts
- General learning kernel formulation
  - linear, non-negative combinations
  - non-linear combinations and alternative formulations
- Alignment-based algorithms
- Ensemble combinations
Overview

LK formulations:

- (Lanckriet et al., 2004): SVM, $L_1$ regularization, general, linear, or non-negative combinations.
- (Cortes et al., 2009): KRR, $L_2$ regularization, non-negative combinations.
- (Kloft et al., 2009): SVM, $L_p$ regularization, linear, or non-negative combinations.
General LK Formulation - SVMs

**Notation:**
- $\mathcal{K}$ set of PDS kernel functions.
- $\overline{\mathcal{K}}$ kernel matrices associated to $\mathcal{K}$, assumed convex.
- $Y \in \mathbb{R}^{m \times m}$ diagonal matrix with $Y_{ii} = y_i$.

**Optimization problem:**

$$
\min_{\mathcal{K} \in \overline{\mathcal{K}}} \max_{\alpha} \ 2 \alpha^\top 1 - \alpha^\top Y^\top K Y \alpha
$$

subject to: $0 \leq \alpha \leq C \land \alpha^\top y = 0$.

- convex problem: function linear in $K$, convexity of pointwise maximum.
Consider the maximization problem:

\[
\max_{\alpha} 2 \alpha^\top 1 - \alpha^\top Y^\top KY \alpha \\
\text{subject to: } 0 \leq \alpha \leq C \land \alpha^\top y = 0.
\]

The corresponding Lagrange function is

\[
L = 2 \alpha^\top 1 - \alpha^\top Y^\top KY \alpha + 2\beta^\top \alpha - 2\gamma^\top (\alpha - C) - 2\delta \alpha^\top y.
\]

and \( \nabla_{\alpha} L = 0 \iff Y^\top KY \alpha = 1 + \beta - \gamma - \delta y. \)

Thus, \((Y^\top KY)^\dagger(1 + \beta - \gamma - \delta y)\) is one solution.

Plugging that in gives the dual problem

\[
\min_{\beta \geq 0, \gamma \geq 0, \delta} (1 + \beta - \gamma - \delta y)^\top (Y^\top KY)^\dagger (1 + \beta - \gamma - \delta y) + 2\gamma^\top C.
\]
General LK Formulation - SVMs

- The problem can now be rewritten as
  \[
  \min_{t, \beta, \gamma, \delta} t
  \text{ subject to: } t \geq (1 + \beta - \gamma - \delta y)^\top (Y^\top K Y)^\dagger (1 + \beta - \gamma - \delta y) + 2 \gamma^\top C
  \]
  \[
  (\beta \geq 0) \land (\gamma \geq 0).
  \]

- Now, by the property of the Schur complement with a singular matrix (Boyd and Vandenberghe, 2004), this is equivalent to
  \[
  \min_{t, \beta, \gamma, \delta} t
  \text{ subject to: } \begin{bmatrix}
  Y^\top K Y & 1 + \beta - \gamma - \delta y \\
  (1 + \beta - \gamma - \delta y)^\top & t - 2 \gamma^\top C
  \end{bmatrix} \succeq 0
  \]
  \[
  (\beta \geq 0) \land (\gamma \geq 0).
  \]
General LK Formulation - SVMs

Optimization problem:

\[
\min_{K \in \overline{\mathcal{K}}, t, \beta, \gamma, \delta} t \\
\text{subject to: } \begin{bmatrix} Y^T K Y & 1 + \beta - \gamma - \delta y \\ (1 + \beta - \gamma - \delta y)^\top & t - 2\gamma^\top C \end{bmatrix} \succeq 0 \\
(\beta \geq 0) \land (\gamma \geq 0).
\]

- the minimization over \( t, \beta, \gamma, \delta \) is a semi-definite program (SDP).
- if \( \overline{\mathcal{K}} = \{ K : (K \succeq 0) \land \text{Tr}[K] = 1 \} \) the full program is an SDP.
Notes

Comments on (Lanckriet et al., 2004):

- **full proof** that problem is equivalent to an SDP not given. The proof given implicitly assumes \((K + \tau I)\) invertible for \(\tau \geq 0\) which in general does not hold. In particular, for \(\tau = 0\), \(K\) is in general not invertible.

- the paper deals **exclusively** with transductive scenario. Thus, instead of minimizing over kernel functions, it minimizes over kernel matrices.

- paper has been the basis for large part of the work done in LK area.
Parameterized LK Formulation

- **Notation:**
  - \((K_{\mu})_{\mu \in \Delta}\) parameterized set of PDS kernel functions.
  - \(\Delta\) convex set, \(\mu \mapsto K_{\mu}\) concave function.
  - \(Y \in \mathbb{R}^{m \times m}\) diagonal matrix with \(Y_{ii} = y_i\).

- **Optimization problem:**

\[
\min_{\mu \in \Delta} \max_{\alpha} 2 \alpha^\top 1 - \alpha^\top Y^\top K_{\mu} Y \alpha
\]

subject to: \(0 \leq \alpha \leq C \land \alpha^\top y = 0\).

- convex problem: function convex in \(\mu\), convexity of pointwise maximum.
Linear Combinations

- $p \geq 1$ base PDS kernel functions $K_1, \ldots, K_p$.

- Kernel family:
  \[ K_{\mu} = \sum_{k=1}^{p} \mu_k K_k : \mu \in \Delta_{\text{lin}} \]
  with \[ \Delta_{\text{lin}} = \left\{ \mu \in \mathbb{R}^p : \sum_{k=1}^{p} \mu_k = 1 \land \mathbf{K}_{\mu} \succeq 0 \right\} \].

- Hypothesis sets:
  \[ H_{\text{lin}} = \left\{ h \in \mathbb{H}_K : K \in \mathcal{K}_{\text{lin}}, \| h \|_{\mathbb{H}_K} \leq 1 \right\} \].
Linear Combinations

(Lanckriet et al., 2004)

Assuming trace-normalized base kernel matrices:

\[
\text{Tr}[K_\mu] = \sum_{k=1}^{p} \mu_k \text{Tr}[K_k] = \sum_{k=1}^{p} \mu_k.
\]

Optimization problem: semi-definite program (SDP).

\[
\min_{\mu, t} \quad t
\]
\[
\text{subject to: } \begin{bmatrix} Y^\top K_\mu Y & 1 + \beta - \gamma - \delta y \\ (1 + \beta - \gamma - \delta y)^\top & t - 2\gamma^\top C \end{bmatrix} \succeq 0
\]
\[
(\beta \geq 0) \land (\gamma \geq 0)
\]
\[
(\sum_{k=1}^{p} \mu_k = 1) \land (K_\mu = \sum_{k=1}^{p} \mu_k K_k) \land (K_\mu \succeq 0).
\]
Non-Negative Combinations

- \( p \geq 1 \) base PDS kernel functions \( K_1, \ldots, K_p \).

- Kernel family:

\[
\mathcal{K}_q = \left\{ K_\mu = \sum_{k=1}^{p} \mu_k K_k : \mu \in \Delta_q \right\}
\]

with \( \Delta_q = \left\{ \mu \in \mathbb{R}^p : \|\mu\|_q \leq 1, \mu \geq 0 \right\} \).

- Hypothesis sets:

\[
H_q = \left\{ h \in \mathcal{H}_K : K \in \mathcal{K}_q, \|h\|_{\mathcal{H}_K} \leq 1 \right\}.
\]
Non-Negative Combinations

By von Neumann’s generalized minimax theorem (convexity wrt $\mu$, concavity wrt $\alpha$, $\Delta_1$ convex and compact, $\mathcal{A}$ convex and compact):

$$\min_{\mu \in \Delta_1} \max_{\alpha \in \mathcal{A}} \ 2 \alpha^\top 1 - \alpha^\top Y^\top K_{\mu} Y \alpha$$

$$= \max_{\alpha \in \mathcal{A}} \min_{\mu \in \Delta_1} \ 2 \alpha^\top 1 - \alpha^\top Y^\top K_{\mu} Y \alpha$$

$$= \max_{\alpha \in \mathcal{A}} \ 2 \alpha^\top 1 - \max_{\mu \in \Delta_1} \alpha^\top Y^\top K_{\mu} Y \alpha$$

$$= \max_{\alpha \in \mathcal{A}} \ 2 \alpha^\top 1 - \max_{k \in [1,p]} \alpha^\top Y^\top K_k Y \alpha.$$
Non-Negative Combinations

(Lanckriet et al., 2004)

Optimization problem: in view of the previous analysis, the problem can be rewritten as the following QCQP.

\[
\max_{\alpha, t} \quad 2\alpha^\top \mathbf{1} - t \\
\text{subject to: } \forall k \in [1, p], \ t \geq \alpha^\top \mathbf{Y}^\top K_k \mathbf{Y} \alpha; \\
0 \leq \alpha \leq C \land \alpha^\top \mathbf{y} = 0.
\]

• complexity (interior-point methods): \(O(pm^3)\).
N-Neg. Comb. - Primal Formulation

Optimization problem: equivalent primal.

$$\min_{w, \mu \in \Delta_q} \frac{1}{2} \sum_{k=1}^{p} \frac{\|w_k\|^2}{\mu_k} + \sum_{i=1}^{m} \max\left\{0, 1 - y_i \left(\sum_{k=1}^{p} w_k \Phi_k(x_i)\right)\right\}.$$
Rank-One Base Kernels

- Optimization problem: reduces to simple QP.

\[ \max_{\alpha} 2\alpha^\top 1 - t^2 \]

subject to: \( \forall k \in [1, p], -t \leq \alpha^\top Y^\top X_k \leq t; \)

\[ 0 \leq \alpha \leq C \land \alpha^\top y = 0. \]

- \( K_k = X_k X_k^\top \).

- application to learning sequence kernels (Cortes et al., 2008).
Solving Non-Negative Combinations

- **Wrapper methods**: interleaving a call to an SVM solver and an update of the kernel parameters.

- **Beyond wrapper methods**: methods that avoid the call to the SVM solver.

- **SMO methods**: methods that re-write the SVM solver and find the optimal kernel parameters.

- **Experimental comparison**.
Wrapper Methods

Alternate steps between solving the SVM and updating the kernel parameters using:

- SILP
- Steepest descent
- Reduced gradient
- Newton’s method
- Mirror descent
What is a Semi-Infinite Linear Program?

\[
\max_y \quad b^\top y \\
\text{subject to: } \quad a_\alpha^\top y \leq c_\alpha, \forall \alpha \in A,
\]

- where \( y, b, a_\alpha \in \mathbb{R}^m \), \( c_\alpha \in \mathbb{R} \), and \( \alpha \in A \), with \( A \) typically a compact (infinite) set.

- Efficient for large-scale problems when used with constraint generating methods.
SILP

QCQP for non-negative combinations rewritten as (changing sign in objective function):

$$\max_{\beta} \min_{\alpha} \sum_{k=1}^{p} \beta_k (\alpha^T Y^T K_k Y \alpha - 2\alpha^T 1)$$

subject to: $$(\sum_{k=1}^{p} \beta_k = 1) \land (\beta \geq 0)$$

$$(0 \leq \alpha \leq C) \land (\alpha^T y = 0)$$.
SILP - Formulation

 Optimization problem: semi-infinite linear program (SILP), e.g., LP with infinitely many constraints.

\[
\begin{align*}
\max_{\beta, \theta} \quad & \theta \\
\text{subject to:} \quad & \theta \leq \sum_{k=1}^{p} \beta_k (\alpha^\top Y^\top K_k Y \alpha - 2\alpha^\top 1) \\
& \left(\sum_{k=1}^{p} \beta_k = 1\right) \land \left(\beta \geq 0\right) \\
& \left(0 \leq \alpha \leq C\right) \land \left(\alpha^\top y = 0\right).
\end{align*}
\]
SILP - Algorithm

Algorithm: repeat following operations.

- solve LP with finite number of constraints.
- add new (most violating constraint), that is for a fixed $\beta$, find $\alpha \in A$ minimizing

$$
\sum_{k=1}^{p} \beta_k (\alpha^T Y^T K_k Y \alpha - 2\alpha^T 1) = \alpha^T Y^T K_\beta Y \alpha - 2\alpha^T 1,
$$

which coincides with solving dual SVM.

- Many other heuristics: e.g., chunking for SVM problem, removing inactive constraints for LP.
- No clear convergence rate guarantee, but handles large samples (e.g., 1M points, 20 kernels).
Reduced Gradient

- Optimization problem:
  \[
  \min_{\mu \in \Delta} \max_{\alpha} \ 2 \alpha^\top 1 - \alpha^\top Y^\top K_{\mu} Y \alpha
  \]
  subject to: \(0 \leq \alpha \leq C \land \alpha^\top y = 0\).

- Kernel family: \(\mathcal{K} = \left\{ K_{\mu} = \sum_{k=1}^{p} \mu_k K_k : \mu \in \Delta \right\}\).

- Reduced gradient:
  
  Let \(J = 2 \alpha^\top 1 - \alpha^\top Y^\top K_{\mu} Y \alpha\)

  \[
  \nabla_{\text{red}} J_k = \frac{\partial J}{\partial \mu_k} - \frac{\partial J}{\partial \mu_m}, \ k \neq m \\
  \nabla_{\text{red}} J_m = \sum_{k \neq m} \left( \frac{\partial J}{\partial \mu_m} - \frac{\partial J}{\partial \mu_k} \right).
  \]
Reduced Gradient: SimpleMKL

(Rakotomamonjy et al., 2008)

SimpleMKL algorithm

Algorithm 1 SimpleMKL algorithm

\[
\text{set } d_m = \frac{1}{m} \text{ for } m = 1, \ldots, M
\]

\[\text{while stopping criterion not met do}\]

\[\text{compute } J(d) \text{ by using an SVM solver with } K = \sum_m d_m K_m \]

\[\text{compute } \frac{\partial J}{\partial d_m} \text{ for } m = 1, \ldots, M \text{ and descent direction } D (12).\]

\[\text{set } \mu = \arg\max_m d_m, J^\dagger = 0, d^\dagger = d, D^\dagger = D\]

\[\text{while } J^\dagger < J(d) \text{ do } \{\text{descent direction update}\}\]

\[d = d^\dagger, D = D^\dagger\]

\[v = \arg\min_{\{m|D_m < 0\}} -d_m / D_m, \gamma_{\text{max}} = -d_v / D_v\]

\[d^\dagger = d + \gamma_{\text{max}} D, D^\mu = D_{\mu} - D_{\nu}, D^\nu = 0\]

\[\text{compute } J^\dagger \text{ by using an SVM solver with } K = \sum_m d^\dagger_m K_m\]

\[\text{end while}\]

\[\text{line search along } D \text{ for } \gamma \in [0, \gamma_{\text{max}}] \{\text{calls an SVM solver for each } \gamma \text{ trial value}\}\]

\[d \leftarrow d + \gamma D\]

\[\text{end while}\]
Newton’s Method

- **Optimization problem:**
  \[
  \min_{\mu \in \Delta} F(\mu)
  \]

- **Approximate** \( F \):
  \[
  G_t(\mu) = F(\mu^t) + (\mu - \mu^t)^\top \nabla_\mu F(\mu)\big|_{\mu^t} + \frac{1}{2} (\mu - \mu^t)^\top \nabla^2 F(\mu)\big|_{\mu^t} (\mu - \mu^t).
  \]

- **Solving for** \( \mu \):
  \[
  \nabla G_t(\mu) = 0 \iff \nabla_\mu F(\mu)\big|_{\mu^t} + H(\mu^t)(\mu - \mu^t) = 0
  \]
  \[
  \iff \Delta \mu = -H^{-1}(\mu^t) \nabla F(\mu)\big|_{\mu^t}.
  \]
Newton’s Method: L_q-Norm

(Kloft et al., 2009)

- Optimization problem:

\[
\min_{\mu \in \Delta_q, w, b, \xi} \frac{1}{2} \sum_{k=1}^{p} \frac{\|w_k\|^2}{\mu_k} + C\|\xi\|_1
\]

subject to: \(\forall i \ y_i \left( \sum_{k=1}^{p} w_k^\top \psi(x)_i + b \right) \geq 1 - \xi_i, \xi \geq 0, \mu \geq 0, \|\mu\|^q_q \leq 1.\)

- Kernel family \(\mathcal{K}_q = \left\{ K_\mu = \sum_{k=1}^{p} \mu_k K_k : \mu \in \Delta_q \right\}.\)

- Lagrange function:

\[
L = \sum_{k=1}^{p} \frac{\|w_k\|^2}{\mu_k} + \delta \left( \sum_{k=1}^{p} \mu_k^q - 1 \right).
\]
Newton’s Method: $L_q$-Norm

(Kloft et al., 2009)

Computing the derivatives:

\[
\frac{\partial L}{\partial \mu_k} = -\frac{1}{2} \frac{w_k^\top w_k}{\mu_k^2} + \delta \mu_k^{q-1}
\]

\[
\frac{\partial^2 L}{\partial \mu_k^2} = \frac{w_k^\top w_k}{\mu_k^3} + (q - 1)\delta \mu_k^{q-2}
\]

Hessian diagonal:

\[
\Delta \mu_k = \frac{\frac{1}{2} \mu_k w_k^\top w_k - \delta \mu_k^{q+2}}{w_k^\top w_k + (q - 1)\delta \mu_k^{q+1}}
\]

- various techniques used to enforce non-negative parameters.
Mirror Descent

- **Optimization problem:**

\[
\min_{\mu \in \Delta} F(\mu)
\]

- **Approximate** \( F \):

\[
G_t(\mu) = F(\mu^t) + (\mu - \mu^t)^\top \nabla_{\mu} F(\mu)|_{\mu^t} + \frac{1}{s_t} B_\Omega(\mu^t \| \mu).
\]

- strictly convex function \( \Omega(\mu) \) and

\[
B_\Omega(\mu^t \| \mu) = \Omega(\mu) - \Omega(\mu^t) - (\mu - \mu^t)^\top \nabla_{\mu} \Omega|_{\mu^t}
\]

Bregman divergence defined by \( \Omega(\mu) \).
Mirror Descent

- Solving \( \nabla_{\mu} G_t(\mu) = 0 \)
  gives \( \nabla_{\mu} \Omega|_{\mu} - \nabla_{\mu} \Omega|_{\mu^t} = -s_t \nabla_{\mu} F(\mu)|_{\mu^t} \)
  and the next value of \( \mu \) given by
  \[
  \mu^{t+1} = [\nabla_{\mu} \Omega]^{-1} \left( \nabla_{\mu} \Omega|_{\mu^t} - s_t \nabla_{\mu} F(\mu)|_{\mu^t} \right).
  \]

- Examples
  - coordinate function inversion.
  \[
  \Omega(\mu) = \frac{1}{2} \|\mu\|_2^2 \Rightarrow \mu^{t+1} = \mu^t - s_t \nabla_{\mu} F(\mu)|_{\mu^t}.
  \]
  - vector of coord. \log(\mu_k).
  \[
  \Omega(\mu) = \mu^T \log(\mu) \Rightarrow \mu^{t+1} = \mu^t \exp(-s_t \nabla_{\mu} F(\mu)|_{\mu^t}).
  \]
Mirror Descent: Mixed-Norm MKL

(Nath et al., 2009)

Optimization problem:

\[
\max_{\forall j, \mu_j \in \Delta_{n_j}} \max_{\alpha \in S_m(C), \gamma \in \Delta_n} \mathbf{1}^\top \alpha - \frac{1}{2} \alpha^\top \left[ \sum_{j=1}^{n} \frac{\sum_{k=1}^{n_j} \mu_{jk} K_{jk}}{\gamma_j} \right] \alpha.
\]

Kernel family:

\[
\mathcal{K}_q = \left\{ K_{\mu} = \sum_{k=1}^{p} \sum_{l=1}^{n_k} \frac{\mu_{kl}}{\gamma_k} K_{kl} : \mu_k \in \Delta_{n_k}, \gamma \in \Delta_p \right\}
\]

\[
\Omega(\mu) = \sum_{k=1}^{p} \sum_{l=1}^{n_k} \left( \frac{\mu_{kl}}{p} + \frac{\delta}{pn_k} \right) \log \left( \frac{\mu_{kl}}{p} + \frac{\delta}{pn_k} \right).
\]
Mirror Descent: Mixed-Norm MKL

Update of kernel parameter

$$\mu_{kl}^{t+1} = \frac{\mu_{kl}^t \exp(-\rho s_t [\nabla F|u^t]_{kl})}{\sum_{l=1}^{n_k} \mu_{kl}^t \exp(-\rho s_t [\nabla F|u^t]_{kl})}$$

$$F = 2\alpha^\top 1 - \alpha^\top Y^\top K_\mu Y \alpha$$

$$K_q = \left\{ K_\mu = \sum_{k=1}^p \sum_{l=1}^{n_k} \frac{\mu_{kl}}{\gamma_k} K_{kl} : \mu_k \in \Delta_{n_k}, \gamma \in \Delta_p \right\}$$

Specific step-size gives bound on the number of iterations.
Beyond Wrappers

Avoiding call to SVM:
- Online methods, $L_q$-norm.
- Projected gradient, KRR.
Online Methods - $L_q$-Norm

(Orabona & Jie, 2011)

- Optimization problem:

$$
\min_{\tilde{w}} \quad \Omega(\tilde{w}) + \frac{1}{N} \sum_{i=1}^{N} \ell(\tilde{w}, \phi(x_i, \cdot), y_i).
$$

$$
\Omega(\tilde{w}) := \lambda/2 \|\tilde{w}\|^2_{2, \log F} + \alpha \|\tilde{w}\|_{2,1},
$$

- where $\ell$ is the hinge loss for the case of SVMs.

- Use Mirror Descent algorithm to update $w$:

$$
w^{t+1} = \nabla_w \Omega^{-1}(\nabla_w \Omega|_{w^t} - s_t \nabla_w \ell(w)|_{w^t})
$$

- where $\nabla_w \ell(w)|_{w^t}$ is determined by sampling.
Projected Gradient, KRR

Kernel family:
- non-negative combinations.
- $L_q$ regularization.

Optimization problem:

\[
\min_{\mu} \max_{\alpha} \quad -\lambda \alpha^\top \alpha - \sum_{k=1}^{p} \mu_k \alpha^\top K_k \alpha + 2 \alpha^\top y \\
\text{subject to: } \mu \geq 0 \land \|\mu - \mu_0\|_q \leq \Lambda.
\]

- convex optimization: linearity in $\mu$ and convexity of pointwise maximum.
Projected Gradient, KRR

- Solving maximization problem in $\alpha$, closed-form solution $\alpha = (K_\mu + \lambda I)^{-1} y$, reduces problem to

$$\min_{\mu} \ y^T (K_\mu + \lambda I)^{-1} y$$

subject to: $\mu \geq 0 \land \|\mu - \mu_0\|_2 \leq \Lambda$.

- Convex optimization problem, one solution using projection-based gradient descent:

$$\frac{\partial F}{\partial \mu_k} = \text{Tr} \left[ \frac{\partial y^T (K_\mu + \lambda I)^{-1} y}{\partial (K_\mu + \lambda I)} \frac{\partial (K_\mu + \lambda I)}{\partial \mu_k} \right]$$

$$= - \text{Tr} \left[ (K_\mu + \lambda I)^{-1} yy^T (K_\mu + \lambda I)^{-1} \frac{\partial (K_\mu + \lambda I)}{\partial \mu_k} \right]$$

$$= - \text{Tr} \left[ (K_\mu + \lambda I)^{-1} yy^T (K_\mu + \lambda I)^{-1} K_k \right]$$

$$= - y^T (K_\mu + \lambda I)^{-1} K_k (K_\mu + \lambda I)^{-1} y = -\alpha^T K_k \alpha.$$ 

$\square$
Projected Gradient, KRR - L2 Reg.

(Cortes et al., 2009)

ProjectBasedGradientDescent\((\{(K_k)_{k\in[1,p]}\}, \mu_0)\)

1 \(\mu \leftarrow \mu_0\)
2 \(\mu' \leftarrow \infty\)
3 while \(\|\mu' - \mu\| > \epsilon\) do
4 \(\mu \leftarrow \mu'\)
5 \(\alpha \leftarrow (K\mu + \lambda I)^{-1}y\)
6 \(\mu' \leftarrow \mu + \eta (\alpha^\top K_{1}\alpha, \ldots, \alpha^\top K_{p}\alpha)^\top\)
7 for \(k \leftarrow 1\) to \(p\) do
8 \(\mu'_k \leftarrow \max(0, \mu'_k)\)
9 \(\mu' \leftarrow \mu_0 + \Lambda \frac{\mu' - \mu_0}{\|\mu' - \mu_0\|}\)
10 return \(\mu'\)
Interpolated Step, KRR - L2 Reg.

(Cortes et al., 2009)

**INTERPOLATED_ITERATIVE_ALGORITHM**\((\{K_k\}_{k \in [1,p]}, \mu_0)\)

1. \(\alpha \leftarrow \infty\)
2. \(\alpha' \leftarrow (K_{\mu_0} + \lambda I)^{-1}y\)
3. while \(\|\alpha' - \alpha\| > \epsilon\) do
4. \(\alpha \leftarrow \alpha'\)
5. \(v \leftarrow (\alpha^\top K_1 \alpha, \ldots, \alpha^\top K_p \alpha)^\top\)
6. \(\mu \leftarrow \mu_0 + \Lambda \frac{v}{\|v\|}\)
7. \(\alpha' \leftarrow \eta \alpha + (1 - \eta)(K_{\mu} + \lambda I)^{-1}y\)
8. return \(\alpha'\)

Simple and very efficient: few iterations (less than 15).
SMO Solutions

- MKL and SMO - (Bach et al., 2004)
  - Moreau-Yosida regularization to form smooth problem for $L_1$-regularization.

- MKL and SMO - (Vishwanathan et al., 2010)
  - Squared $L_q$-norm results in smooth problem in dual.
Experimental Results

- Solving the same problem.
  - only difference is the norm of the regularization.

- Compare speed for different norms.

- Compare accuracy for different norms.
SILP Algorithm

- Semi-infinite linear programming (SILP) approach for convex combinations.
- 20 base kernels, 1,000,000 training points (human splice dataset).
- Requires on-the-fly kernel computation, employs caching, chunking and parallelization.

\[\text{~28 hours}\] (Sonnenburg et al., 2006)

\[\text{precompute - precomputed kernels, no chunking, no parallelization, caching} \]
\[\text{linadd - using chunking algorithm}\]
SimpleMKL

(Rakotomamonjy et al., 2006)

- Reduced gradient method for solving $L_1$-regularized MKL.
- In regimes of small scale data, but 100’s of kernels, SimpleMKL show improvement over SILP method.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th># Kernel</th>
<th>Accuracy</th>
<th>Time (s)</th>
<th># SVM eval</th>
<th># Gradient eval</th>
</tr>
</thead>
<tbody>
<tr>
<td>SILP</td>
<td>11.6 ± 1.0</td>
<td>76.5 ± 2.3</td>
<td>224 ± 37</td>
<td>95.6 ± 13</td>
<td>95.6 ± 13</td>
</tr>
<tr>
<td>SimpleMKL</td>
<td>14.7 ± 1.4</td>
<td>76.5 ± 2.6</td>
<td>79.0 ± 13</td>
<td>314 ± 44</td>
<td>24.3 ± 4.8</td>
</tr>
<tr>
<td>Grad. Desc.</td>
<td>14.8 ± 1.4</td>
<td>75.5 ± 2.5</td>
<td>219 ± 24</td>
<td>873 ± 147</td>
<td>118 ± 8.7</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Algorithm</th>
<th># Kernel</th>
<th>Accuracy</th>
<th>Time (s)</th>
<th># SVM eval</th>
<th># Gradient eval</th>
</tr>
</thead>
<tbody>
<tr>
<td>SILP</td>
<td>33.5 ± 3.8</td>
<td>80.5 ± 5.1</td>
<td>2290 ± 864</td>
<td>903 ± 187</td>
<td>903 ± 187</td>
</tr>
<tr>
<td>SimpleMKL</td>
<td>36.7 ± 5.1</td>
<td>80.6 ± 5.1</td>
<td>163 ± 93</td>
<td>2770 ± 1560</td>
<td>115 ± 66</td>
</tr>
<tr>
<td>Grad. Desc.</td>
<td>35.7 ± 3.9</td>
<td>80.2 ± 4.7</td>
<td>469 ± 90</td>
<td>7630 ± 2600</td>
<td>836 ± 99</td>
</tr>
</tbody>
</table>

Pima $\ell = 538$ $M = 117$

Sonar $\ell = 146$ $M = 793$
Efficient $L_p$ Regularized

(Kloft et al., 2009)

- Wrapper methods for $L_p$-regularized combinations: Newton or Cutting Plane + SVM.
- Allows for efficient computation of non-sparse combinations of kernel.

~2.8 hours

MNIST dataset, Gaussian kernels

Corinna Cortes, Mehryar Mohri, Afshin Rostami - ICML 2011 Tutorial.
SMO Optimization

- SMO for $L_p$-regularization.
- Found to scale better with training size than (Kloft et al., 2009).

(Vishwanathan et al., 2009)

![Graphs showing comparison between SMO-MKL and Shogun for different kernels and training points.](image)

- ~0.3h for 166 points
- ~6h for 50 kernels
Stochastic Gradient Descent

(Orabona et al., 2010 & 2011)

- OBSCURE and UFO-MKL for $L_p$-regularization.
- Primal formulation allows for general loss functions, e.g. multi-class classification.

![Graph showing classification rate vs. training time](image)
**L₁-Regularized Combinations**

(Lanckriet et al., 2004)

- Learn with sparse linear combinations of kernels.
- Combining kernels can help performance, but do simple uniform combinations suffice?

<table>
<thead>
<tr>
<th>Dataset</th>
<th>$\mu_{1,+}$</th>
<th>$\mu_{2,+}$</th>
<th>$\mu_{3,+}$</th>
<th>$\mu_{4,+}$</th>
<th>$\mu_{5,+}$</th>
<th>TSA</th>
<th>SM2,C</th>
<th>TSA best c/v</th>
<th>RBF</th>
</tr>
</thead>
<tbody>
<tr>
<td>Breast Cancer</td>
<td>0</td>
<td>0</td>
<td>3.24</td>
<td>0.94</td>
<td>0.82</td>
<td>97.1%</td>
<td></td>
<td>96.8%</td>
<td></td>
</tr>
<tr>
<td>Ionosphere</td>
<td>0.85</td>
<td>0.85</td>
<td>2.63</td>
<td>0.68</td>
<td>0</td>
<td>94.5%</td>
<td></td>
<td>94.2%</td>
<td></td>
</tr>
<tr>
<td>Heart</td>
<td>0</td>
<td>3.89</td>
<td>0.06</td>
<td>1.05</td>
<td>0</td>
<td>84.1%</td>
<td></td>
<td>83.2%</td>
<td></td>
</tr>
<tr>
<td>Sonar</td>
<td>0</td>
<td>3.93</td>
<td>1.07</td>
<td>0</td>
<td>0</td>
<td>84.8%</td>
<td></td>
<td>84.2%</td>
<td></td>
</tr>
<tr>
<td>2-norm</td>
<td>0.49</td>
<td>0.49</td>
<td>0</td>
<td>3.51</td>
<td>0</td>
<td>96.5%</td>
<td></td>
<td>97.2%</td>
<td></td>
</tr>
</tbody>
</table>
L$_1$-Regularized Combinations  
(Lanckriet et al., Bioinformatics 2004)

- Yeast protein classification, 7 domain specific kernels, 2318 samples.
Multi-Class L₁-Regularized

(Zien & Ong., 2007; Ong & Zien, 2008)

- Predict subcellular localization (TargetP dataset), 5 classes, 69 base kernels.
- Multi-class SVM with L₁-regularization.

MC-MKL

unif
L₂-Regularized Combinations

(Cortes et al., 2009)

- Dense combinations are beneficial when using many kernels.
- Combining kernels based on single features, can be viewed as principled feature weighting.
L_p-Regularized Combinations

(Sonnenburg et al., Bioinformatics 2006; Kloft et al., 2009)

- Non-sparse combination are found to be more effective (in terms of AUC) for transcription start site (TSS) recognition.

- 5 kernels, up to 60,000 training examples.
This Part

- Early attempts
- General learning kernel formulation
  - linear, non-negative combinations
  - non-linear combinations and alternative formulations
- Alignment-based algorithms
- Ensemble combinations
Non-Linear Combinations and Alternative Formulations

- Gaussian and polynomial kernels
  - DC-Programming algorithm (Argyriou et al., 2005)
  - Generalized MKL (Varma & Babu, 2009)
  - Polynomial kernels - KRR (Cortes et al., 2009)
- Hierarchical kernels (Bach, 2008)
- Hyperkernels (Ong et al., 2005)
- Radius-based kernel learning (Gai et al., 2010)
Gaussian and Polynomial Kernels

(Weston et al, 2000; Argyriou et al., 2005; Varma and Babu, 2009)

Optimize over a continuously parameterized set of

Gaussians: \( K_\mu(x_i, x_j) = \prod_{k=1}^{p} \exp \left( -\mu_k (x_{ik} - x_{jk})^2 \right) \)

Polynomials: \( K_{\mu,d}(x_i, x_j) = (1 + \sum_{k=1}^{p} \mu_k x_{ik} x_{jk})^d \)

Wrapper method:

- (Argyriou et al., 2005): squared loss, DC (difference of convex functions) to find new parameters.
GMKL: Reality Check

(Varma and Babu, 2009)

- **Feature selection**: train MKL and rank features according to weights. Retrain with top-k weights. Compare to other feature selection algorithms:

<table>
<thead>
<tr>
<th>Ionosphere: N = 246, M = 34, Uniform MKL = 89.9 ± 2.5</th>
<th>Uniform GMKL = 94.6 ± 2.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nd</td>
<td>AdaBoost</td>
</tr>
<tr>
<td>5</td>
<td>75.2 ± 6.9</td>
</tr>
<tr>
<td>10</td>
<td>-</td>
</tr>
<tr>
<td>15</td>
<td>-</td>
</tr>
<tr>
<td>20</td>
<td>-</td>
</tr>
<tr>
<td>25</td>
<td>-</td>
</tr>
<tr>
<td>30</td>
<td>-</td>
</tr>
<tr>
<td>34</td>
<td>-</td>
</tr>
</tbody>
</table>

MKL + $l_1$-reg: $K_{\mu}(x_i, x_j) = \sum_{k=1}^{p} \mu_k \exp \left(-\gamma_k (x_{ik} - x_{jk})^2\right)$

GMKL + $l_1$-reg: $K_{\mu}(x_i, x_j) = \prod_{k=1}^{p} \exp \left(-\mu_k (x_{ik} - x_{jk})^2\right)$

Unknown how $\gamma_k$ is chosen...
GMKL: Reality Check

(Varma and Babu, 2009)

Accuracy:

<table>
<thead>
<tr>
<th>Database</th>
<th>SimpleMKL</th>
<th>GMKL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sonar</td>
<td>80.6 ± 5.1 (793)</td>
<td>82.3 ± 4.8 (60)</td>
</tr>
<tr>
<td>Wpbc</td>
<td>76.7 ± 1.2 (442)</td>
<td>79.0 ± 3.5 (34)</td>
</tr>
<tr>
<td>Ionosphere</td>
<td>91.5 ± 2.5 (442)</td>
<td>93.0 ± 2.1 (34)</td>
</tr>
<tr>
<td>Liver</td>
<td>65.9 ± 2.3 (091)</td>
<td>72.7 ± 4.0 (06)</td>
</tr>
<tr>
<td>Pima</td>
<td>76.5 ± 2.6 (117)</td>
<td>77.2 ± 2.1 (08)</td>
</tr>
</tbody>
</table>

HKL + $l_1$-reg: $K_{\mu,4}(x_i, x_j) = \prod_{k=1}^{p} \left(1 + \mu_k x_{ik} x_{jk}\right)^4$

GMKL + $l_1$-reg: $K_{\mu,2}(x_i, x_j) = \left(1 + \sum_{k=1}^{p} \mu_k x_{ik} x_{jk}\right)^2$
Polynomial Kernels - KRR

(Cortes et al., 2010)

- $p \geq 1$ base PDS kernel functions $K_1, \ldots, K_p$.

- Kernel family: polynomial degree $d \geq 2$.

\[
K_q = \left\{ K_\mu = \left( \sum_{k=1}^{p} \mu_k \right)^d : \mu \in \Delta_q \right\}
\]

with $\Delta_q = \left\{ \mu \in \mathbb{R}^p : \|\mu\|_q \leq 1, \mu \geq 0 \right\}$.

- Hypothesis sets:

\[
H_q = \left\{ h \in \mathbb{H}_K : K \in K_q, \|h\|_{\mathbb{H}_K} \leq 1 \right\}.
\]
Polynomial Kernels - KRR

- **Optimization problem:** case $d=2$.

\[
\min_{\mu} \max_{\alpha} \quad -\lambda \alpha^\top \alpha - \sum_{k,l=1}^{p} \mu_k \mu_k \alpha^\top (K_k \circ K_l) \alpha + 2\alpha^\top y
\]
subject to: $\mu \geq 0 \land \|\mu - \mu_0\|_q \leq \Lambda$.

- **Closed-form solution** $\alpha = (K_\mu + \lambda I)^{-1} y$ leads to:

\[
\min_{\mu} F(\mu) = y^\top \left( \sum_{k,l=1}^{p} \mu_k \mu_k K_k \circ K_l + \lambda I \right)^{-1} y
\]
subject to: $\mu \geq 0 \land \|\mu - \mu_0\|_q \leq \Lambda$. 

Function Properties

- Two properties:
  - decreasing.
  - no interior stationary points
    ➞ optimal solution at the boundary.

- Convex regions exist under certain conditions.
Pol. Kernels - KRR: Reality Check

(Cortes et al., 2010)

- Sentiment dataset (Blitzer et al.).

- Polynomial kernels with $d=2$, and $L_1$ and $L_2$ regularization. Baseline is a uniformly weighted quadratic kernel.
Hierarchical Kernel Learning (Bach, 2008)

Example:

- Sub kernel:
  \[ K_{i,j}(x_i, x'_i) = \binom{q}{j}(1 + x_i x'_i)^j, \quad i \in [1, p], \quad j \in [0, q] \]

- Full kernel:
  \[ K(x, x') = \prod_{i=1}^{p}(1 + x_i x'_i)^q \]

- Convex optimization problem under some assumptions, complexity polynomial in the number of kernels selected, sparsity through \( L_1 \) regularization and hierarchical selection criteria.
### HKL: Reality Check

(Bach, 2008)

Regression:
Normalized Mean Squared Error \times 100

<table>
<thead>
<tr>
<th>dataset</th>
<th>( n )</th>
<th>( p )</th>
<th>( k )</th>
<th>( #(V) )</th>
<th>L2</th>
<th>MKL</th>
<th>HKL</th>
</tr>
</thead>
<tbody>
<tr>
<td>abalone</td>
<td>4177</td>
<td>10</td>
<td>pol4</td>
<td>( \approx 10^7 )</td>
<td>44.2±1.3</td>
<td>44.5±1.1</td>
<td><strong>43.3±1.0</strong></td>
</tr>
<tr>
<td>abalone</td>
<td>4177</td>
<td>10</td>
<td>rbf</td>
<td>( \approx 10^{10} )</td>
<td>43.0±0.9</td>
<td>43.7±1.0</td>
<td>43.0±1.1</td>
</tr>
<tr>
<td>bank-32fh</td>
<td>8192</td>
<td>32</td>
<td>pol4</td>
<td>( \approx 10^{22} )</td>
<td>40.1±0.7</td>
<td><strong>38.7±0.7</strong></td>
<td>38.9±0.7</td>
</tr>
<tr>
<td>bank-32fh</td>
<td>8192</td>
<td>32</td>
<td>rbf</td>
<td>( \approx 10^{31} )</td>
<td>39.0±0.7</td>
<td>38.4±0.7</td>
<td><strong>38.4±0.7</strong></td>
</tr>
<tr>
<td>bank-32fm</td>
<td>8192</td>
<td>32</td>
<td>pol4</td>
<td>( \approx 10^{22} )</td>
<td>6.0±0.1</td>
<td>6.1±0.3</td>
<td>5.1±0.1</td>
</tr>
<tr>
<td>bank-32fm</td>
<td>8192</td>
<td>32</td>
<td>rbf</td>
<td>( \approx 10^{31} )</td>
<td>5.7±0.2</td>
<td>5.9±0.2</td>
<td><strong>4.6±0.2</strong></td>
</tr>
<tr>
<td>bank-32nh</td>
<td>8192</td>
<td>32</td>
<td>pol4</td>
<td>( \approx 10^{22} )</td>
<td>44.3±1.2</td>
<td>46.0±1.2</td>
<td><strong>43.6±1.1</strong></td>
</tr>
<tr>
<td>bank-32nh</td>
<td>8192</td>
<td>32</td>
<td>rbf</td>
<td>( \approx 10^{31} )</td>
<td>44.3±1.2</td>
<td>46.1±1.1</td>
<td><strong>43.5±1.0</strong></td>
</tr>
<tr>
<td>bank-32nm</td>
<td>8192</td>
<td>32</td>
<td>pol4</td>
<td>( \approx 10^{22} )</td>
<td>17.2±0.6</td>
<td>21.0±0.7</td>
<td><strong>16.8±0.6</strong></td>
</tr>
<tr>
<td>bank-32nm</td>
<td>8192</td>
<td>32</td>
<td>rbf</td>
<td>( \approx 10^{31} )</td>
<td>16.9±0.6</td>
<td>20.9±0.7</td>
<td><strong>16.4±0.6</strong></td>
</tr>
<tr>
<td>boston</td>
<td>506</td>
<td>13</td>
<td>pol4</td>
<td>( \approx 10^9 )</td>
<td><strong>17.1±3.6</strong></td>
<td>22.2±2.2</td>
<td>18.1±3.8</td>
</tr>
<tr>
<td>boston</td>
<td>506</td>
<td>13</td>
<td>rbf</td>
<td>( \approx 10^{12} )</td>
<td><strong>16.4±4.0</strong></td>
<td>20.7±2.1</td>
<td>17.1±4.7</td>
</tr>
<tr>
<td>pumadyn-32fh</td>
<td>8192</td>
<td>32</td>
<td>pol4</td>
<td>( \approx 10^{22} )</td>
<td>57.3±0.7</td>
<td><strong>56.4±0.7</strong></td>
<td>56.4±0.8</td>
</tr>
<tr>
<td>pumadyn-32fh</td>
<td>8192</td>
<td>32</td>
<td>rbf</td>
<td>( \approx 10^{31} )</td>
<td>57.7±0.6</td>
<td>56.5±0.8</td>
<td><strong>55.7±0.7</strong></td>
</tr>
<tr>
<td>pumadyn-32fm</td>
<td>8192</td>
<td>32</td>
<td>pol4</td>
<td>( \approx 10^{22} )</td>
<td>6.9±0.1</td>
<td>7.0±0.1</td>
<td>3.1±0.0</td>
</tr>
<tr>
<td>pumadyn-32fm</td>
<td>8192</td>
<td>32</td>
<td>rbf</td>
<td>( \approx 10^{31} )</td>
<td>5.0±0.1</td>
<td>7.1±0.1</td>
<td><strong>3.4±0.0</strong></td>
</tr>
<tr>
<td>pumadyn-32nh</td>
<td>8192</td>
<td>32</td>
<td>pol4</td>
<td>( \approx 10^{22} )</td>
<td>84.2±1.3</td>
<td>83.6±1.3</td>
<td><strong>36.7±0.4</strong></td>
</tr>
<tr>
<td>pumadyn-32nh</td>
<td>8192</td>
<td>32</td>
<td>rbf</td>
<td>( \approx 10^{31} )</td>
<td>56.5±1.1</td>
<td>83.7±1.3</td>
<td><strong>35.5±0.5</strong></td>
</tr>
<tr>
<td>pumadyn-32nm</td>
<td>8192</td>
<td>32</td>
<td>pol4</td>
<td>( \approx 10^{22} )</td>
<td><strong>60.1±1.9</strong></td>
<td>77.5±0.9</td>
<td><strong>5.5±0.1</strong></td>
</tr>
<tr>
<td>pumadyn-32nm</td>
<td>8192</td>
<td>32</td>
<td>rbf</td>
<td>( \approx 10^{31} )</td>
<td><strong>15.7±0.4</strong></td>
<td>77.6±0.9</td>
<td><strong>7.2±0.1</strong></td>
</tr>
</tbody>
</table>
Hyperkernels

Kernels over kernels, $K$

Representer theorem: $m^2$ Lagrange multipliers:

$$K(x, x') = \sum_{i,j=1}^{m} \beta_{i,j} K(((x_i, x_j), (x, x'))) \quad \forall x, x' \in X, \quad \beta_{i,j} \geq 0$$

Hyperkernel example:

$$K\left((x, x'), (x'', x''')\right) = \prod_{j=1}^{d} \frac{1 - \lambda}{1 - \lambda \exp \left( - \sigma_j ((x_j - x'_j)^2 + (x''_j - x'''_j)^2) \right)}$$

For fixed $\sigma_j$ SDP problem similar to Lanckriet, SeDuMi.
Hyperkernels: Reality Check

(Ong et al, 2005)

\[
K\left((x, x'), (x'', x''')\right) = \prod_{j=1}^{d} \frac{1 - \lambda}{1 - \lambda \exp \left(-\sigma_j \left((x_j - x'_j)^2 + (x''_j - x'''_j)^2\right)\right)}
\]

\(\sigma_j\) is fixed.
Radius-based Kernel Learning, RKL

(Gai et al, 2010)

- Slack term $O(\sqrt{R^2/\rho^2})$.

- For fixed kernel, the radius is constant, but for combinations of kernels it varies.

- **Primal:**
  
  $R^2(K) = \min_{y,c} y, \text{ s.t. } y \geq \|\phi_K(x_i) - c\|^2$

- **Dual:**
  
  $R^2(K) = \max_{\beta_i} \sum_{i=1}^m \beta_i K(x_i, x_i) - \sum_{i,j=1}^m \beta_i \beta_j K(x_i, x_j), \text{ s.t. } \beta_i \geq 0, \sum_{i=1}^m \beta_i = 1$

- RKL optimization

  $$\min_{\theta} g(\theta),$$

  where $g(\theta) = \left\{ \max_{\alpha_i} \sum_i \alpha_i - \frac{1}{2r^2(\theta)} \sum_{i,j} \alpha_i \alpha_j y_i y_j K_{i,j}(\theta), \text{ s.t. } \sum_i \alpha_i y_i = 0, 0 \leq \alpha_i \leq C \right\}$,

  where $r^2(\theta) = \left\{ \max_{\beta_i} \sum_i \beta_i K_{i,i}(\theta) - \sum_{i,j} \beta_i K_{i,j}(\theta) \beta_j, \text{ s.t. } \sum_i \beta_i = 1, \beta_i \geq 0 \right\}$. 
### RKL: Reality Check

(Gai et al, 2010)

<table>
<thead>
<tr>
<th>Index</th>
<th>1 Unif</th>
<th>2 MKL $L_1$</th>
<th>3 KL-C $L_1$</th>
<th>4 Ours $L_1$</th>
<th>5 MKL $L_2$</th>
<th>6 KL-C $L_2$</th>
<th>7 Ours $L_2$</th>
<th>8 Ours No</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ionosphere</td>
<td>94.0(1.4) 20</td>
<td>92.9(1.6) 3.8</td>
<td>86.0(1.9) 4.0</td>
<td>95.7(0.9) 2.8</td>
<td>94.3(1.5) 20</td>
<td>84.4(1.6) 18</td>
<td>95.7(0.9) 3.0</td>
<td>95.7(0.9) 3.0</td>
</tr>
<tr>
<td>Splice</td>
<td>51.7(0.1) 20</td>
<td>79.5(1.9) 1.0</td>
<td>80.5(1.9) 2.8</td>
<td>86.5(2.4) 3.2</td>
<td>82.0(2.2) 20</td>
<td>74.0(2.6) 14</td>
<td>86.5(2.4) 2.2</td>
<td>86.3(2.5) 3.2</td>
</tr>
<tr>
<td>Liver</td>
<td>58.0(0.0) 20</td>
<td>59.1(1.4) 4.2</td>
<td>62.9(3.5) 4.0</td>
<td>64.1(4.2) 3.6</td>
<td>67.0(3.8) 20</td>
<td>64.1(3.9) 11</td>
<td>64.1(4.2) 8.0</td>
<td>64.3(4.3) 6.6</td>
</tr>
<tr>
<td>Fourclass</td>
<td>81.2(1.9) 20</td>
<td>97.7(1.2) 7.0</td>
<td>94.0(1.2) 2.0</td>
<td>100 (0.0) 1.0</td>
<td>97.3(1.6) 20</td>
<td>94.0(1.3) 17</td>
<td>100 (0.0) 1.0</td>
<td>100 (0.0) 1.6</td>
</tr>
<tr>
<td>Heart</td>
<td>83.7(6.1) 20</td>
<td>84.1(5.7) 7.4</td>
<td>83.3(5.9) 1.8</td>
<td>84.1(5.7) 5.2</td>
<td>83.7(5.8) 20</td>
<td>83.3(5.1) 19</td>
<td>84.4(5.9) 5.4</td>
<td>84.8(5.0) 5.8</td>
</tr>
<tr>
<td>Germannum</td>
<td>70.0(0.0) 20</td>
<td>70.0(0.0) 7.2</td>
<td>71.9(1.8) 9.8</td>
<td>73.7(1.6) 4.8</td>
<td>71.5(0.8) 20</td>
<td>71.6(2.1) 13</td>
<td>73.9(1.2) 6.0</td>
<td>73.9(1.8) 5.8</td>
</tr>
<tr>
<td>Musk1</td>
<td>61.4(2.9) 20</td>
<td>85.5(2.9) 1.6</td>
<td>73.9(2.9) 2.0</td>
<td>93.3(2.3) 4.0</td>
<td>87.4(3.0) 20</td>
<td>61.9(3.1) 19</td>
<td>93.5(2.2) 3.8</td>
<td>93.3(2.3) 3.8</td>
</tr>
<tr>
<td>Wdbc</td>
<td>94.4(1.8) 20</td>
<td>97.0(1.8) 1.2</td>
<td>97.4(2.3) 4.6</td>
<td>97.4(1.6) 6.2</td>
<td>96.8(1.6) 20</td>
<td>97.4(2.0) 11</td>
<td>97.6(1.9) 5.8</td>
<td>97.6(1.9) 5.8</td>
</tr>
<tr>
<td>Wpbc</td>
<td>76.5(2.9) 20</td>
<td>76.5(2.9) 7.2</td>
<td>52.2(5.9) 9.6</td>
<td>76.5(2.9) 17</td>
<td>75.9(1.8) 20</td>
<td>51.0(6.6) 17</td>
<td>76.5(2.9) 15</td>
<td>76.5(2.9) 15</td>
</tr>
<tr>
<td>Sonar</td>
<td>76.5(1.8) 20</td>
<td>82.3(5.6) 2.6</td>
<td>80.8(5.8) 7.4</td>
<td>86.0(2.6) 2.6</td>
<td>85.2(2.9) 20</td>
<td>80.2(5.9) 11</td>
<td>86.0(2.6) 2.6</td>
<td>86.0(3.3) 3.0</td>
</tr>
<tr>
<td>Coloncancer</td>
<td>67.2(11) 20</td>
<td>82.6(8.5) 13</td>
<td>74.5(4.4) 11</td>
<td>84.2(4.2) 7.2</td>
<td>76.5(9.0) 20</td>
<td>76.0(3.6) 15</td>
<td>84.2(4.2) 5.6</td>
<td>84.2(4.2) 7.6</td>
</tr>
</tbody>
</table>

‘KL-C’ is (Chapelle et al. 2002).

Norm reg.: $L_1 \sim \sum_i \beta_i = 1$, $L_2 \sim \sum_i \beta_i^2 = 1$, or unconstrained (No).

Change of reg. changes relationship to radius.
This Part

- Early attempts
- General learning kernel formulation
  - linear, non-negative combinations
  - non-linear combinations and alternative formulations
- Alignment-based algorithms
- Ensemble combinations
Centered Alignment-Based LK

(Cortes et al., 2010)

- Two stages:
  - Kernel Selection \( \rightarrow K \rightarrow \text{Learning Algorithm} \rightarrow h \)
  - \( S_1 \)
  - \( S_2 \)

- Outperforms uniform baseline and previous algorithms.

- Centered alignment is key: different from notion used by (Cristiannini et al., 2001).
Centered Alignment

(Cortes et al., 2010)

Definition:

\[ \rho(K, K') = \frac{\mathbb{E}[K_c K'_c]}{\sqrt{\mathbb{E}[K_c^2] \mathbb{E}[K'_c^2]}}, \]

with \( K_c(x, x') = (\Phi(x) - \mathbb{E}_x[\Phi])^\top (\Phi(x') - \mathbb{E}_{x'}[\Phi]). \)

Idea: choose \( K \in \mathcal{K} \) maximizing alignment with the labeling kernel (target kernel):

\[ K_Y(x, x') = f(x) f(x'). \]
Centering crucial for correlation with error. See also (Meila et al., 2003; Pothin & Richard, 2008).

correlation: -0.95  0.45

correlation: -0.96  -0.86

Centered  Un-centered
Notes

(Cortes et al., 2010) comment on (Cristianini, Shawe-Taylor, Elisseef, Kandola, 2001) and related papers by the same authors:

• alignment definition does not correlate well with performance.
• thus, poor empirical performance.
• main proof of the paper about the existence of good classifiers is incorrect.
• concentration bound not directly on quantities of interest.
Existence of Good Predictor

Theorem: let $h^*$ be the hypothesis defined for all $x$ by

$$h^*(x) = \frac{E_{x'}[y'K_c(x, x')]}{\sqrt{E[K_c^2]}},$$

and assume normalized labels: $E[y^2] = 1$. Then,

$$\text{error}(h^*) = E_x[(h^*(x) - y)^2] \leq 2(1 - \rho(K, K_Y)).$$
Proof

\[ E_x[h^*^2(x)] = E_x \left[ \frac{E_{x'}[y' K_c(x, x')]^2}{E[K_c^2]} \right] \]

\[ \leq E_x \left[ \frac{E_{x'}[y'^2]E_{x'}[K_c^2(x, x')]}{E[K_c^2]} \right] \]

\[ = \frac{E_{x, x'}[K_c^2(x, x')]}{E[K_c^2]} = 1. \]

Thus,

\[ E[(y - h^*(x))^2] = E_x[h^*(x)^2] + E_x[y^2] - 2E_x[y h^*(x)] \]

\[ \leq 1 + 1 - 2\rho(K, K_Y). \]

But, alignment between kernel functions unavailable!
Empirical Centered Alignment

Definition:

\[ \hat{\rho}(K, K') = \frac{\langle K_c, K'_c \rangle_F}{\|K_c\|_F \|K'_c\|_F}. \]

Concentration bound: with probability at least \(1 - \delta\),

\[ |\rho(K, K') - \hat{\rho}(K, K')| \leq 6\beta \left[ \frac{3}{m} + \sqrt{\frac{2 \log \frac{6}{\delta}}{m}} \right], \]

with \( \beta = \max(R^4/E[K_c^2], R^4/E[K'_c^2]) \).
Algorithm

Empirical alignment maximization:

$$\mu^* = \arg\max_{\mu \in \Delta_1} \hat{\rho}(K_\mu, yy^\top) = \arg\max_{\mu \in \Delta_1} \frac{\langle K_{\mu c}, yy^\top \rangle_F}{\| K_{\mu c} \|_F}$$

with $K_\mu = \sum_{k=1}^{p} \mu_k K_k$.

Reduces to simple QP: $\mu^* = \frac{v^*}{\| v^* \|}$,

$$v^* = \arg\min_{v \succeq 0} v^\top M v - 2 v^\top a,$$

$$a = \left( \langle K_{1c}, yy^\top \rangle_F, \ldots, \langle K_{pc}, yy^\top \rangle_F \right)^\top, \quad M_{kl} = \langle K_{kc}, K_{lc} \rangle_F.$$
Alternative Algorithm

Based on independent base kernel alignments:

\[ K_{\mu} \propto \sum_{k=1}^{p} \hat{\rho}(K_k, K_Y)K_k. \]

- Easily scales to very large numbers of kernels.
Centered Alignment: Reality Check

(Cortes et al., 2010)

Gaussian base kernels with varying bandwidth.

Corinna Cortes, Mehryar Mohri, Afshin Rostami - ICML 2011 Tutorial.
Centered Alignment: Reality Check

(Cortes et al., 2010)

4,000 rank-1 base kernels.

BOOKS

ELECTRONICS

KITCHEN

uniform 1-stage simple align

uniform 1-stage simple align

uniform 1-stage simple align

RMSE

RMSE

RMSE

Alignment

Alignment

Alignment
Centered Alignment-Based LK

Properties:
- outperforms uniform combination.
- based on new definition of centered alignment.
- effective in classification and regression.
- proof of existence of good predictors.
- concentration bound for centered alignment.
- stability-based generalization bound.
- algorithm reduced to a simple QP.

Question: better criterion for first stage?
This Part

- Early attempts
- General learning kernel formulation
  - linear, non-negative combinations
  - non-linear combinations and alternative formulations
- Alignment-based algorithms
- Ensemble combinations
Ensemble Combinations

(Gehler & Nowozin, 2009; Cortes et al., 2011)

- Two stages:

  \[ K_1 \rightarrow \text{Learning Alg.} \rightarrow h_1 \]

  \[ \vdots \]

  \[ K_p \rightarrow \text{Learning Alg.} \rightarrow h_p \]

  \[ \text{Lin. Comb.} \rightarrow h \]

- Standard Learning algorithm in first stage.

- Second stage linearly combines predictions from the first stage, 
  \[ h(x) = \sum_{i=1}^{p} \mu_i h_i(x). \]
Ensemble Hypothesis Class

- $L_q$ regularized ensemble:

$$\mathcal{E}_p^q = \left\{ \sum_{k=1}^{p} \mu_k h_k : \|h_k\|_{W_k} \leq \Lambda_k, k \in [1, p], \mu \in \Delta_q \right\}.$$  

- Note, difference in regularization.

- How do learning kernel (LK) and ensemble kernel (EK) methods compare?
  - Hypothesis complexity.
  - Empirical performance.
Let $\eta_0 = 23/22$ and $\Lambda_* = \max_{k \in [1,p]} \Lambda_k$ furthermore assume, $\forall k \in [1,p], \forall x \in \mathcal{X}$ $K_k(x,x) \leq R^2$, then

$$\hat{R}_S(\mathcal{E}_p^1) \leq \sqrt{\frac{\eta_0 e \left[\log p\right]\Lambda_*^2 R^2}{m}}$$

Same as LK!

$$\hat{R}_S(\mathcal{E}_p^q) \leq \sqrt{\frac{\eta_0 r p^{\frac{2}{r}} \Lambda_*^2 R^2}{m}}$$

Differs by $p^{1/(2r)}$ factor.
Ensemble Comb.: Reality Check

(Gortes et al., 2011)

Gaussian base kernels

**Protien Fold**
- simp. align
- align
- I2 ensemble

**Spambase**
- simp. align
- align
- I2 ensemble

**Ionosphere**
- simp. align
- align
- I2 ensemble

**Kinematics**
- simp. align
- align
- I2 ensemble

stage 1: SVM
stage 2: \(L_2\)-reg SVM

stage 1: KRR
stage 2: KRR

Corinna Cortes, Mehryar Mohri, Afshin Rostami - ICML 2011 Tutorial.
One-Stage Ensemble

- Minimize the error of the ensemble hypothesis:

\[
\min_{\mu \in \Delta_q} \min_{h \in \mathcal{H}_\mu} \sum_{k=1}^p \lambda_k \| h_k \|^2_{K_k} + \sum_{i=1}^m L \left( \sum_{k=1}^p \mu_k h_k(x_i), y_i \right)
\]

- For \( q=1 \) optimization reduces to two-stage problem.

- In general, not practical due to cross-validation needed over \( \lambda_k \) for all \( k \).
Multi-Class LPBoost

(Gehler & Nowozin, 2009) use a multi-class LPBoost based second stage optimization:

$$\min_{\mu, \xi, \rho} \mu - \rho + \frac{1}{\nu N} \sum_{i=1}^{m} \xi_i$$

s.t. \[ \sum_{k=1}^{p} \mu_k h_{k, y_i}(x_i) - \arg\max_{y_j \neq y_i} \sum_{k=1}^{p} \mu_k h_{k, y_j}(x_i) + \xi_i \geq \rho \]

\[ \sum_{k=1}^{p} \mu_k = 1, \mu_k \geq 0 \]
Multi-Class LPBoost

A more complex formulation allows for separate weights for each class:

$$\begin{align*}
\min_{\mu, \xi, \rho} & \quad -\rho + \frac{1}{\nu N} \sum_{i=1}^{m} \xi_i \\
\text{s.t.} & \quad \sum_{k=1}^{p} \mu_k y_i h_{k,y_i}(x_i) - \arg\max_{y_j \neq y_i} \sum_{k=1}^{p} \mu_k h_{k,y_j}(x_i) + \xi_i \geq \rho \\
& \quad \forall c \in [1, C], \quad \sum_{k=1}^{p} \mu_k^c = 1, \mu_k^c \geq 0
\end{align*}$$
LP-B and LP-\(\beta\): Reality Check (Gehler and Nowozin, 2009)

- Two-stage algorithm, combine classifiers trained on individual kernels (39 kernels).
This Part

- Early attempts
- General learning kernel formulation
  - linear, non-negative combinations
  - non-linear combinations and alternative formulations
- Alignment-based algorithms
- Ensemble combinations

DONE