Learning Kernels with Random Features

Aman Sinha & John Duchi, Stanford University
{amans, jduchi}@stanford.edu

Kaikai Zhao
2017/10/8

Introduction

- An essential element of supervised learning systems is the representation of input data.
- Kernel methods provide one approach to this problem: they implicitly transform the data to a new feature space, allowing non-linear data representations.
- This representation comes with a cost.

$$K(x_i,x_j) = \exp(-||x_i-x_j||^2/2\sigma^2), \quad \sigma \quad \text{width}$$

Introduction

- Instead of implicit representation, [1] proposes explicitly mapping the data to a low-dimensional Euclidean inner product space using a randomized feature map z:
- $R^d \rightarrow R^D$ approximates their kernel evaluation:

$$k(\mathbf{x},\mathbf{y}) = \langle \phi(\mathbf{x}), \phi(\mathbf{y}) \rangle \approx \mathbf{z}(\mathbf{x})'\mathbf{z}(\mathbf{y}).$$

Merits ???

1. Ali Rahimi, Benjamin Recht. Random Features for Large-Scale Kernel Machines. NIPS,2007

Introduction

- With the kernel trick, evaluating the machine at a test point x requires computing $f(\mathbf{x}) = \sum_{i=1}^{N} c_i k(\mathbf{x}_i, \mathbf{x})$
- which requires O(Nd) operations to compute and requires retaining much of the dataset unless the machine is very sparse.
- On the other hand, after learning a hyperplane w, a linear machine can be evaluated by simply computing

$$f(x) = \mathbf{w}'\mathbf{z}(\mathbf{x})$$

- which, with the randomized feature maps presented here, requires only O(D + d)? operations and storage.

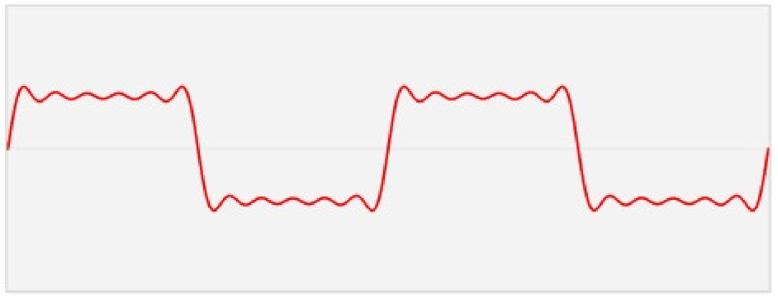
1. Ali Rahimi, Benjamin Recht. Random Features for Large-Scale Kernel Machines. NIPS,2007

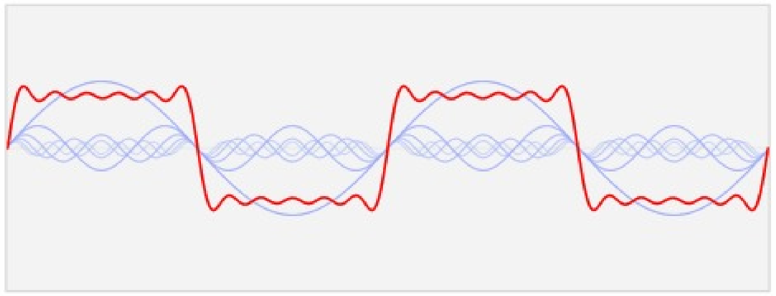
傅里叶变换公式

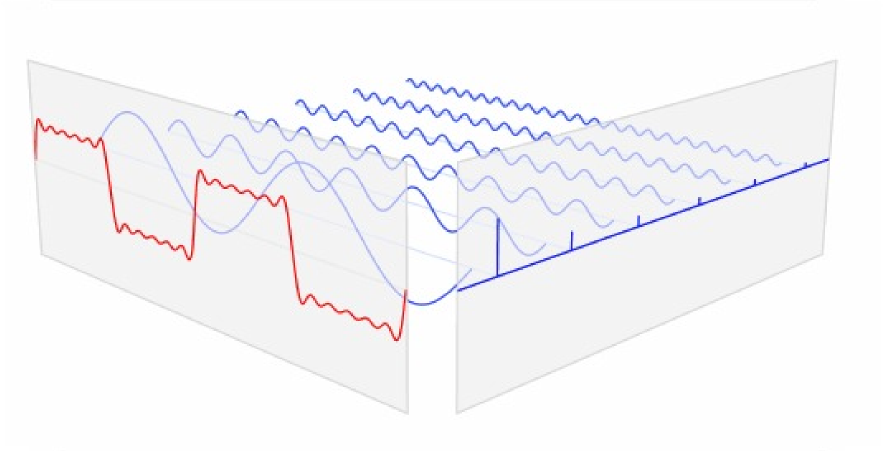
$$F(\omega) = \mathcal{F}[f(t)] = \int_{-\infty}^{\infty} f(t)e^{-iwt}dt$$

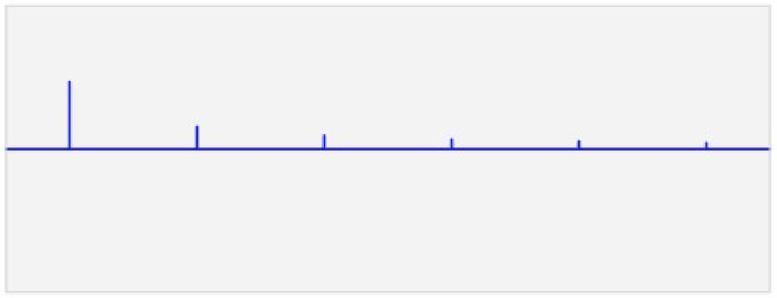
公式描述： 公式中F(ω)为f(t)的像函数，f(t)为F(ω)的像原函数。

$$f(t) = \mathcal{F}^{-1}[F(\omega)] = \frac{1}{2\pi}\int_{-\infty}^{\infty} F(\omega)e^{iwt}d\omega$$



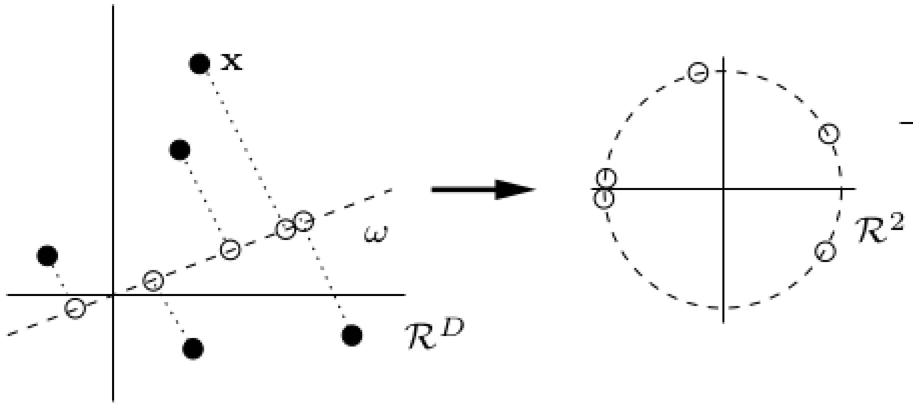






Random Fourier Features

Theorem 1 (Bochner [15]). *A continuous kernel $k(x, y) = k(x - y)$ on $\mathcal{R}^d$ is positive definite if and only if $k(\delta)$ is the Fourier transform of a non-negative measure.*

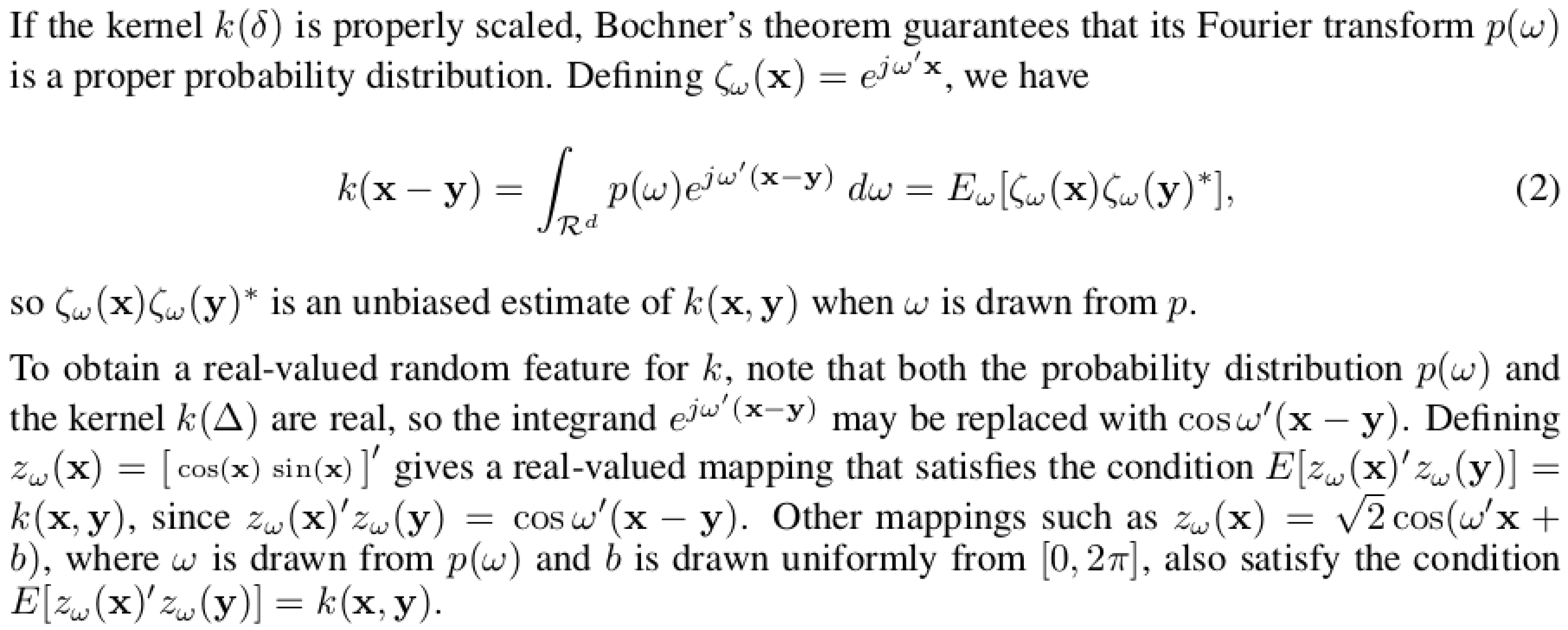


Kernel Name	$k(\Delta)$	$p(\omega)$
Gaussian	$e^{-\frac{\|\Delta\|_2^2}{2}}$	$(2\pi)^{-\frac{D}{2}} e^{-\frac{\|\omega\|_2^2}{2}}$
Laplacian	$e^{-\|\Delta\|_1}$	$\prod_d \frac{1}{\pi(1+\omega_d^2)}$
Cauchy	$\prod_d \frac{2}{1+\Delta_d^2}$	$e^{-\|\Delta\|_1}$

Figure 1: Random Fourier Features. Each component of the feature map $\mathbf{z}(\mathbf{x})$ projects $\mathbf{x}$ onto a random direction ω drawn from the Fourier transform $p(\omega)$ of $k(\Delta)$, and wraps this line onto the unit circle in $\mathcal{R}^2$. After transforming two points $\mathbf{x}$ and $\mathbf{y}$ in this way, their inner product is an unbiased estimator of $k(\mathbf{x}, \mathbf{y})$. The table lists some popular shift-invariant kernels and their Fourier transforms. To deal with non-isotropic kernels, the data may be whitened before applying one of these kernels.

1. Ali Rahimi, Benjamin Recht. Random Features for Large-Scale Kernel Machines. NIPS,2007

Random Fourier Features



If the kernel $k(\delta)$ is properly scaled, Bochner's theorem guarantees that its Fourier transform $p(\omega)$ is a proper probability distribution. Defining $\zeta_\omega(\mathbf{x}) = e^{j\omega'\mathbf{x}}$, we have

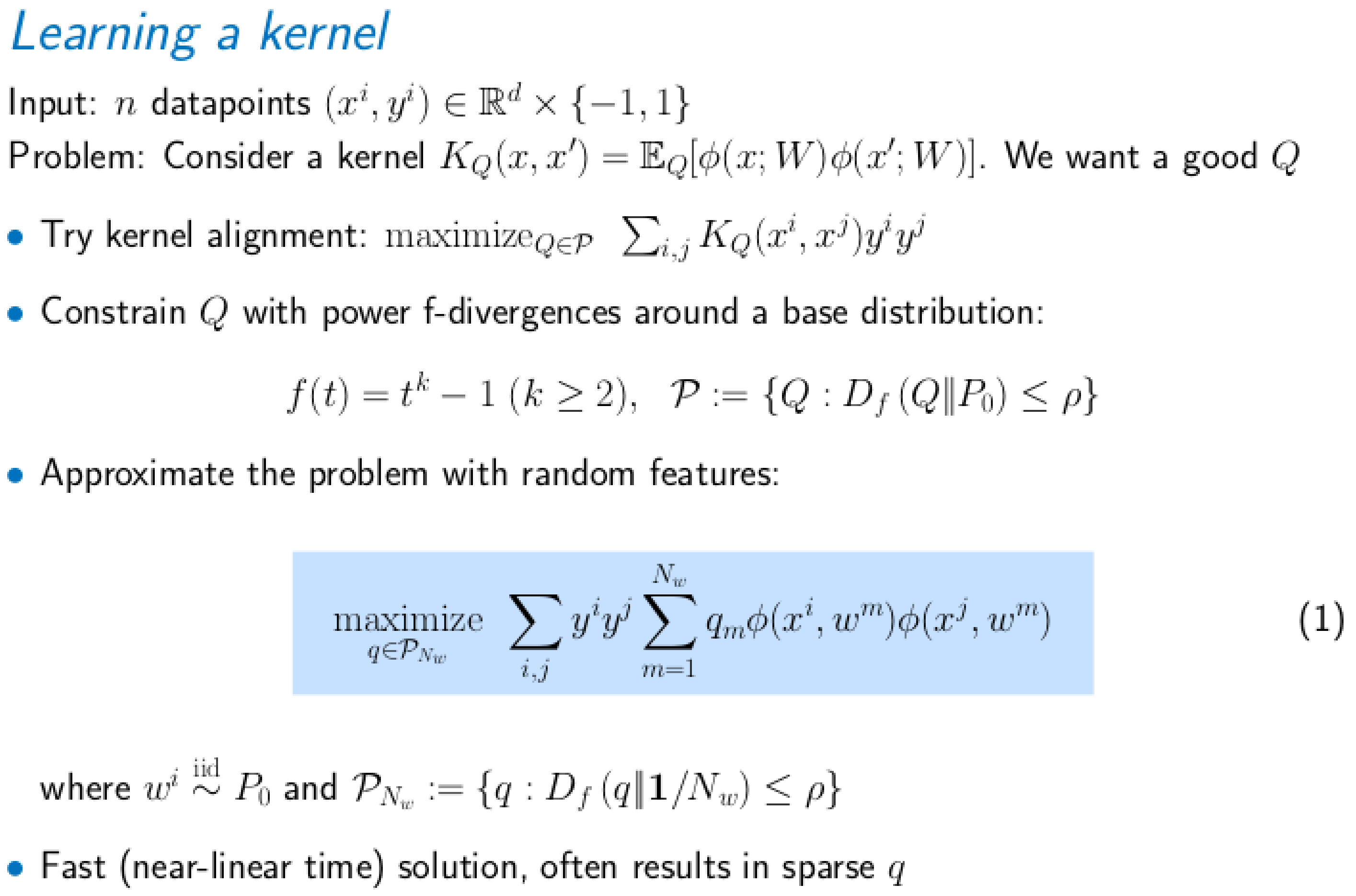
$$k(\mathbf{x}-\mathbf{y}) = \int_{\mathcal{R}^d} p(\omega) e^{j\omega'(\mathbf{x}-\mathbf{y})}\, d\omega = E_\omega[\zeta_\omega(\mathbf{x})\zeta_\omega(\mathbf{y})^*], \tag{2}$$

so $\zeta_\omega(\mathbf{x})\zeta_\omega(\mathbf{y})^*$ is an unbiased estimate of $k(\mathbf{x},\mathbf{y})$ when ω is drawn from p.

To obtain a real-valued random feature for k, note that both the probability distribution $p(\omega)$ and the kernel $k(\Delta)$ are real, so the integrand $e^{j\omega'(\mathbf{x}-\mathbf{y})}$ may be replaced with $\cos\omega'(\mathbf{x}-\mathbf{y})$. Defining $z_\omega(\mathbf{x}) = [\,\cos(\mathbf{x})\ \sin(\mathbf{x})\,]'$ gives a real-valued mapping that satisfies the condition $E[z_\omega(\mathbf{x})' z_\omega(\mathbf{y})] = k(\mathbf{x},\mathbf{y})$, since $z_\omega(\mathbf{x})' z_\omega(\mathbf{y}) = \cos\omega'(\mathbf{x}-\mathbf{y})$. Other mappings such as $z_\omega(\mathbf{x}) = \sqrt{2}\cos(\omega'\mathbf{x} + b)$, where ω is drawn from $p(\omega)$ and b is drawn uniformly from $[0, 2\pi]$, also satisfy the condition $E[z_\omega(\mathbf{x})' z_\omega(\mathbf{y})] = k(\mathbf{x},\mathbf{y})$.

e^(-jwt) = cos(wt) - jsin(wt)   e^(jwt) = cos(wt) + jsin(wt)

Problem Setup and Approach



Learning a kernel

Input: n datapoints $(x^i, y^i) \in \mathbb{R}^d \times \{-1, 1\}$

Problem: Consider a kernel $K_Q(x, x') = \mathbb{E}_Q[\phi(x; W)\phi(x'; W)]$. We want a good Q

- Try kernel alignment: $\text{maximize}_{Q \in \mathcal{P}} \;\sum_{i,j} K_Q(x^i, x^j) y^i y^j$
- Constrain Q with power f-divergences around a base distribution:

$$f(t) = t^k - 1 \;(k \geq 2), \quad \mathcal{P} := \{Q : D_f(Q \| P_0) \leq \rho\}$$

- Approximate the problem with random features:

$$\underset{q \in \mathcal{P}_{N_w}}{\text{maximize}} \;\sum_{i,j} y^i y^j \sum_{m=1}^{N_w} q_m \phi(x^i, w^m) \phi(x^j, w^m) \qquad (1)$$

where $w^i \overset{\text{iid}}{\sim} P_0$ and $\mathcal{P}_{N_w} := \{q : D_f(q \| \mathbf{1}/N_w) \leq \rho\}$

- Fast (near-linear time) solution, often results in sparse q

- At a high level, we take a feature mapping, find a distribution that aligns this mapping with the labels **y**, and draw random features from the learned distribution; we then use these features in a standard supervised learning approach.

Efficiently solving problem

$$\Phi = [\phi^1 \ \cdots \ \phi^n] \in \mathbb{R}^{N_w \times n} \qquad \phi^i = [\phi(x^i, w^1) \ \cdots \ \phi(x^i, w^{N_w})]^T \in \mathbb{R}^{N_w}$$

- is the randomized feature representation for x^i and $w^m \sim P_0$, we can rewrite the optimization objective as

$$\sum_{i,j} y^i y^j \sum_{m=1}^{N_w} q_m \phi(x^i, w^m)\phi(x^j, w^m) = \sum_{m=1}^{N_w} q_m \left(\sum_{i=1}^{n} y^i \phi(x^i, w^m) \right)^2 = q^T \left((\Phi y) \odot (\Phi y) \right)$$

$$D_f\left(Q \| P_0\right) \leq \rho$$

Lagrangian L: $\mathcal{L}(q, \lambda) = q^T \left((\Phi y) \odot (\Phi y) \right) - \lambda \left(D_f\left(q \| \mathbf{1}/N_w\right) - \rho \right)$

Lagrange dual function: $g(\lambda) = \sup_{q \in \Delta} \mathcal{L}(q, \lambda)$

Probability simplex: $\Delta := \{ q \in \mathbb{R}_+^{N_w} : q^T \mathbf{1} = 1 \}$

Efficiently solving problem

- Minimizing g(λ) yields the solution to problem
- This is a convex optimization problem in one dimension so we can use bisection
- Each iteration is maximizing L(q, λ) with respect to q for a given λ
- For f(t) = $t^k - 1$, we define $v := (\Phi y) \odot (\Phi y)$
- solve

$$\underset{q \in \Delta}{\text{maximize}}\ q^T v - \lambda \frac{1}{N_w} \sum_{m=1}^{N_w} (N_w q_m)^k.$$

- KKT conditions

$$q_m = \left[v_m / \lambda N_w^{k-1} + \tau\right]_+^{\frac{1}{k-1}} \qquad \sum_m q_m = 1.$$

$$\underset{q\in\Delta}{\text{maximize}}\; q^T v - \lambda \frac{1}{N_w}\sum_{m=1}^{N_w}(N_w q_m)^k. \tag{8}$$

$$q_m = \left[v_m/\lambda N_w^{k-1} + \tau\right]_+^{\frac{1}{k-1}} \qquad \textstyle\sum_m q_m = 1.$$

A small typo in the form of q_m
A missing k in the denominator

Algorithm 1 Kernel optimization with $f(t) = t^k - 1$ as divergence

```
INPUT: distribution $P_0$ on $\mathcal{W}$, sample $\{(x^i, y^i)\}_{i=1}^n$, $N_w \in \mathbb{N}$, feature function $\phi$, $\epsilon > 0$
OUTPUT: $q \in \mathbb{R}^{N_w}$ that is an $\epsilon$-suboptimal solution to $(\boxed{4})$.
SETUP: Draw $N_w$ samples $w^m \overset{\text{iid}}{\sim} P_0$, build feature matrix $\Phi$, compute $v := (\Phi y) \odot (\Phi y)$.
Set $\lambda_u \leftarrow \infty$,  $\lambda_l \leftarrow 0$,  $\lambda_s \leftarrow 1$
while $\lambda_u = \infty$
  $q \leftarrow \text{argmax}_{q\in\Delta}\, \mathcal{L}(q, \lambda_s)$        // (solution to problem $(\boxed{8})$)
  if $D_f(q \| \mathbf{1}/N_w) < \rho$  then  $\lambda_u \leftarrow \lambda_s$  else  $\lambda_s \leftarrow 2\lambda_s$
while $\lambda_u - \lambda_l > \epsilon\lambda_s$
  $\lambda \leftarrow (\lambda_u + \lambda_l)/2$
  $q \leftarrow \text{argmax}_{q\in\Delta}\, \mathcal{L}(q, \lambda)$        // (solution to problem $(\boxed{8})$)
  if $D_f(q \| \mathbf{1}/N_w) < \rho$  then  $\lambda_u \leftarrow \lambda$  else  $\lambda_l \leftarrow \lambda$
```

However, it is not the case in the MATLAB implementation, which is confusing.

$$\chi^2(P|Q) = \sum_{i=1}^{n}\frac{(p_i - q_i)^2}{q_i} \approx \sum_{i=1}^{n}\frac{p_i^2}{q_i} - 1$$

project_onto_simplex

$$\underset{\mathbf{w}}{\text{minimize}} \ \frac{1}{2}\|\mathbf{w}-\mathbf{v}\|_2^2 \quad \text{s.t.} \quad \sum_{i=1}^{n} w_i \ = \ z \ , \ \ w_i \geq 0$$

When $z = 1$ the above is projection onto the probabilistic simplex. The Lagrangian of the problem in Eq. (3) is

$$\mathcal{L}(\mathbf{w}, \boldsymbol{\zeta}) \ = \ \frac{1}{2}\|\mathbf{w}-\mathbf{v}\|^2 + \theta\left(\sum_{i=1}^{n} w_i - z\right) - \boldsymbol{\zeta}\cdot\mathbf{w} \ \ ,$$

where $\theta \in \mathbb{R}$ is a Lagrange multiplier and $\boldsymbol{\zeta} \in \mathbb{R}_+^n$ is a vector of non-negative Lagrange multipliers. Differentiating with respect to w_i and comparing to zero gives the optimality condition, $\frac{d\mathcal{L}}{dw_i} = w_i - v_i + \theta - \zeta_i = 0$. The complementary slackness KKT condition implies that whenever $w_i > 0$ we must have that $\zeta_i = 0$. Thus, if $w_i > 0$ we get that

$$w_i \ = \ v_i - \theta + \zeta_i \ = \ v_i - \theta \ . \qquad (4)$$

```
INPUT: A vector $\mathbf{v} \in \mathbb{R}^n$ and a scalar $z > 0$
  Sort $\mathbf{v}$ into $\boldsymbol{\mu}$ : $\mu_1 \geq \mu_2 \geq \ldots \geq \mu_p$
  Find $\rho = \max\left\{ j \in [n] : \mu_j - \frac{1}{j}\left(\sum_{r=1}^{j} \mu_r - z\right) >$
  Define $\theta = \frac{1}{\rho}\left(\sum_{i=1}^{\rho} \mu_i - z\right)$
OUTPUT: $\mathbf{w}$ s.t. $w_i = \max\{v_i - \theta \ , \ 0\}$
```

Figure 1. Algorithm for projection onto the simplex.

John Duchi, Shai Shalev-Shwartz. Efficient Projections onto the L1-Ball for Learning in High Dimensions. ICML,2008

% min. x' * v

- % s.t. norm(x - u, 2)^2 <= rho
- %   sum(x) == 1, x >= 0.

- % A partial dual to the problem is given by the Lagrangian
- %
- % L(x, lambda) = (lambda/2) * (norm(x - u, 2)^2 - rho) + x' * v
- % subject to   sum(x) == 1, x >= 0.
- %

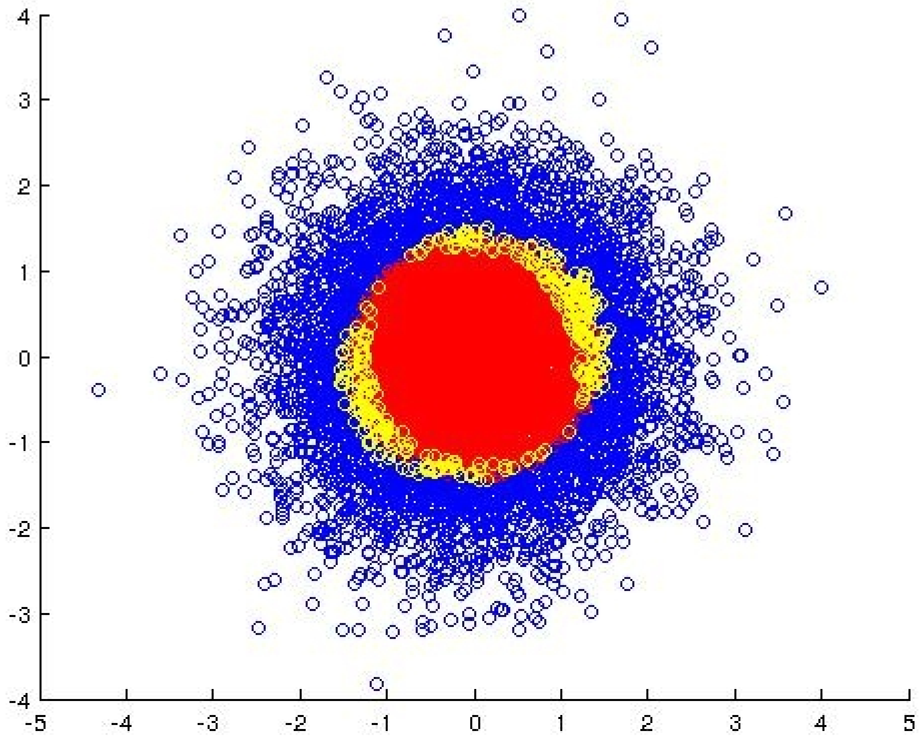
Consistency and generalization performance guarantees

$$\underset{q \in \mathcal{P}_{N_w}}{\text{maximize}} \quad \sum_{i,j} y^i y^j \sum_{m=1}^{N_w} q_m \phi(x^i, w^m) \phi(x^j, w^m). \tag{4}$$

- Although the procedure (4) is a discrete approximation to a heuristic kernel alignment problem,we can provide guarantees on its consistency as well as the generalization performance of our subsequent model trained with the optimized kernel.
- Three lemmas and two theorems(proofs in the supplement)

Empirical evaluations

- synthetic data $x^i \overset{\text{iid}}{\sim} \mathsf{N}(0, I)$
- Labels $y^i = \text{sign}(\|x\|_2 - \sqrt{d})$. $x \in \mathbb{R}^d$.
- Gaussian kernel corresponds to $\phi(x, (w, v)) = \cos((x, 1)^T (w, v))$ $(W, V) \sim \mathsf{N}(0, I) \times \mathsf{Uni}(0, 2\pi)$
- Training set n=10^4 , test set 10^3
- The Gaussian kernel is ill-suited for this task, as the Euclidean distance used in this kernel does not capture the underlying structure of the classes.
- Random features Nw = 2*10^4



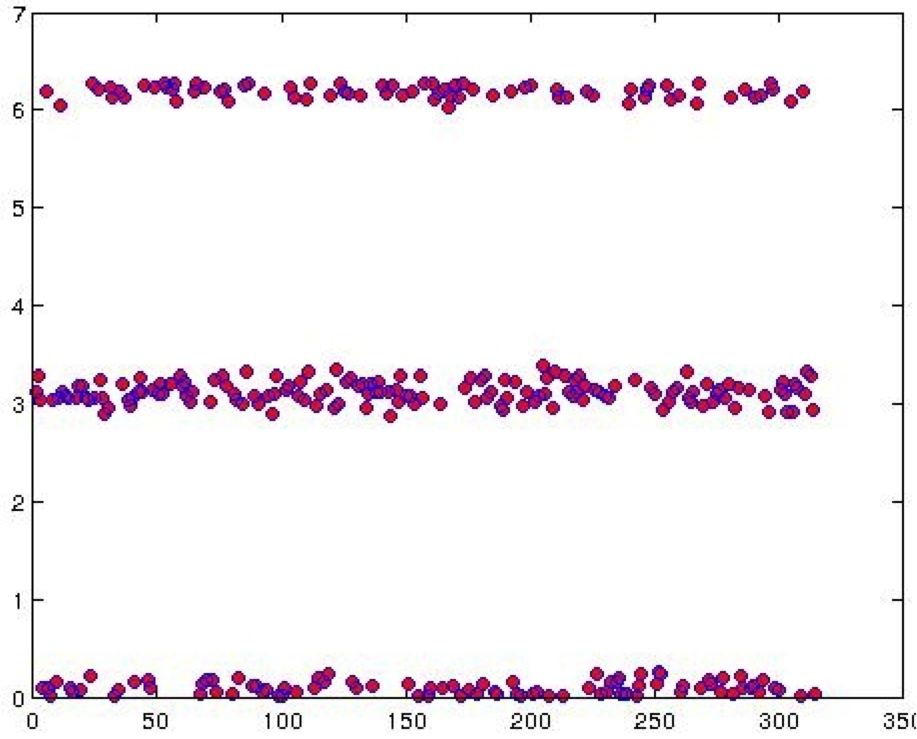
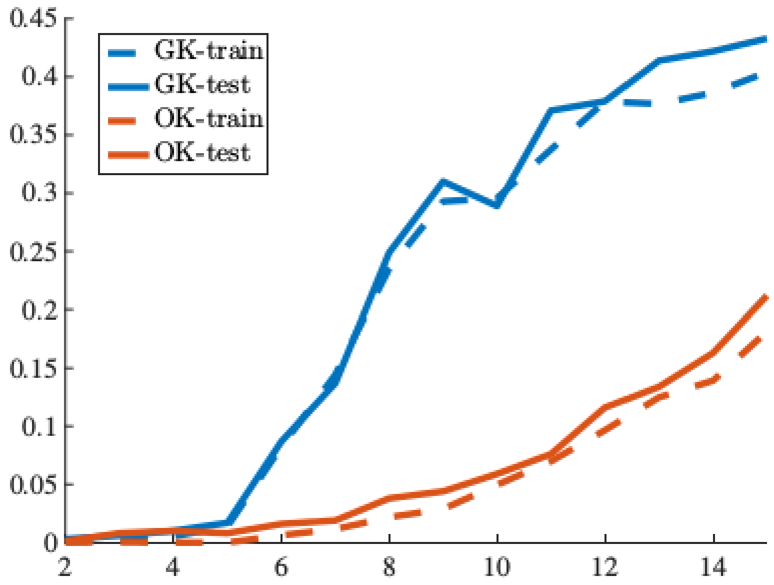


Figure 1 shows the results of the experiments for $d \in \{2, \ldots, 15\}$. Figure 1(a) illustrates the output of the optimization when $d = 2$. The selected kernel features w^m lie near $(1, 1)$ and $(-1, -1)$; the offsets v^m are near 0 and π, giving the feature $\phi(\cdot, w, v)$ a parity flip. Thus, the kernel computes similarity between datapoints via neighborhoods of $(1, 1)$ and $(-1, -1)$ close to the classification boundary. In higher dimensions, this generalizes to neighborhoods of pairs of opposing points along the surface of the d-sphere; these features provide a coarse approximation to vector magnitude.


GK-train
GK-test
OK-train
OK-test
0.45
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
2
4
6
8
10
12
14

(b) Error vs. d