Nonlinear Projection Trick in Kernel Methods

Presented @ 2013 SNU-HU Joint Workshop

Nojun Kwak
nojunk@snu.ac.kr
http://mipal.snu.ac.kr

Graduate School of Convergence Science and Technology
Seoul National University, Korea

Dec. 13, 2013

Note

This presentation is mostly based on the following paper:

- [1] Nojun Kwak, "Nonlinear Projection Trick in Kernel Methods: An Alternative to the Kernel Trick", IEEE TNNLS, vol. 24, no. 12, pp. 2113–2119, Dec. 2013.

Some of the sentences and figures in the overview are from the tutorial

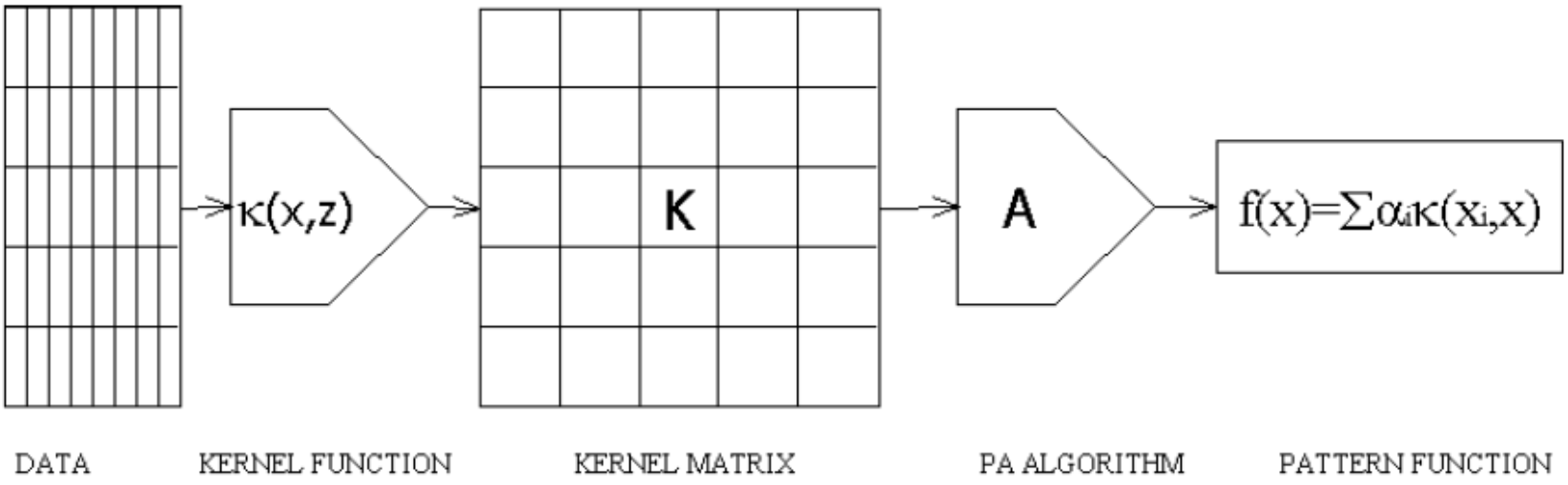
- Nelly Cristianini "Kernel methods for general pattern analysis", http://www.kernel-methods.net/tutorials/KMtalk.pdf

Outline

- Overview on Kernel Methods
  - Basic Idea
  - Kernel Trick
  - Limitation of Kernel Trick
- Nonlinear Projection Trick
  - Motivation
  - Algorithm
  - Applications
- Conclusions and future works

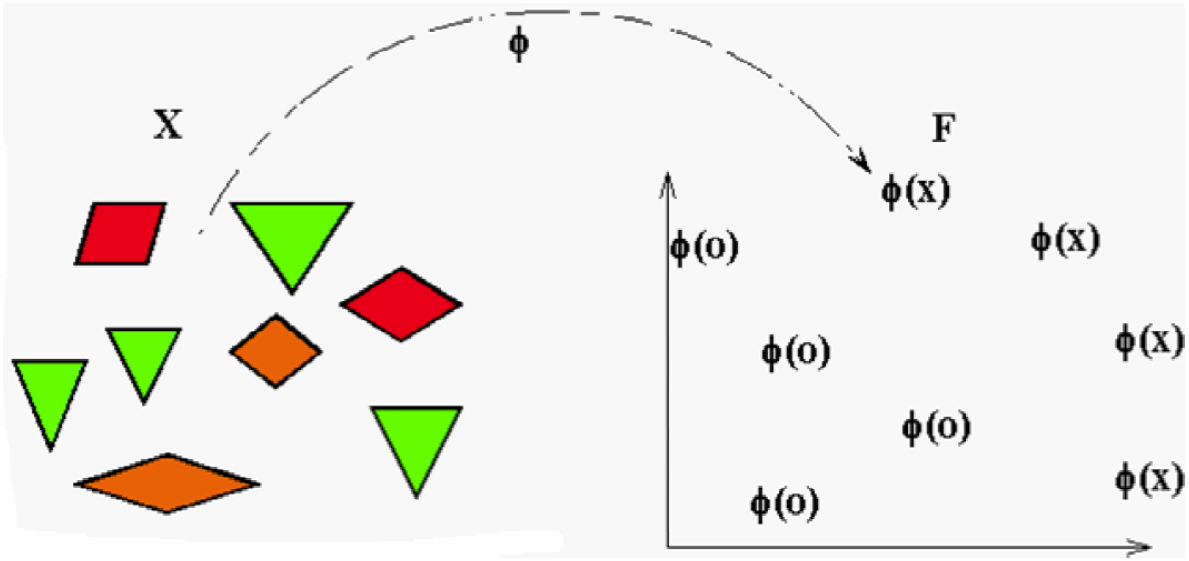
Overview on Kernel Methods

- Kernel Methods are a new class of pattern analysis algorithms which can be applied to very general types of data.
- Kernel methods offer a modular framework.
  1. In a first step, a dataset is processed into a kernel matrix.
  2. In a second step, a variety of kernel algorithms can be used to analyze the data, using only the information contained in the kernel matrix.



Basic Idea: $x \rightarrow \phi(x)$

- Kernel methods work by
  1. Map data in a high (possibly infinite) dimensional vector space.
  2. Look for (linear) relations in such a space.
- If the mapping is chosen suitably, complex relations can be simplified, and easily detected.



Basic Idea: Kernel Trick

- Much of the geometry (relative position) of the data in the embedding space is contained in all pairwise inner products.
- We can work in that space by specifying an inner product function between points in it (rather than their coordinates!!)
- In many cases, inner product in the embedding space is very cheap to compute.

- Inner product matrix:

$$K = \begin{bmatrix} \langle x_1, x_1 \rangle & \langle x_1, x_2 \rangle & \cdot & \langle x_1, x_n \rangle \\ \langle x_2, x_1 \rangle & \langle x_2, x_2 \rangle & \cdot & \langle x_2, x_n \rangle \\ \cdot & \cdot & \cdot & \cdot \\ \langle x_n, x_1 \rangle & \langle x_n, x_2 \rangle & \cdot & \langle x_n, x_n \rangle \end{bmatrix}$$

Kernel Trick: Applicable Algorithms

- There are a lot of algorithms that can be used with inner product (or L_2-norm) information:
  - Principal component analysis (PCA)
  - Fisher discriminant (FLD or LDA)
  - Canonical correlation analysis (CCA)
  - Ridge regression
  - Support vector machines (SVM)
  - Lots more $\cdots\cdots$
- But there are still more algorithms to which kernel trick is not applicable.
  - where L_2 norm is not used in the optimization.
  - (e.g.) PCA-L1
- Motivation: *Can we find a* **direct mapping** *of the input data to the embedding feature space by kernel methods?*

Notations I

- $X \triangleq [x_1, \cdots, x_n] \in \mathbb{R}^{d \times n}$: the training data matrix
  - d : the dimension of the input space
  - n : the number of training samples
- $\Phi(X) \triangleq [\phi(x_1), \cdots, \phi(x_n)] \in \mathbb{R}^{f \times n}$: the mapped training data in the f-dimensional feature space
  - $\phi(x_i)$ is considered as a vector with respect to a countable orthornormal basis of the respective RKHS (reproducing kernel Hilbert space). $\longrightarrow$ f can be infinite.
  - $\Phi(X)$ is assumed to have zero mean, i.e., $\sum_{i=1}^{n} \phi(x_i) = 0$.
- $k(x, y) \triangleq \langle \phi(x), \phi(y) \rangle = \phi(x)^T \phi(y) \in \mathbb{R}$: a kernel function of any two inputs $x, y \in \mathbb{R}^d$.
- $K \triangleq \Phi(X)^T \Phi(X) = [k(x_i, x_j)] \in \mathbb{R}^{n \times n}$: a kernel matrix of the training data.
  - r: the rank of K
  - $r \leq n - 1 \longleftarrow \Phi(X)$ is assumed to be centered.
- $k(x) \triangleq \Phi(X)^T \phi(x) \in \mathbb{R}^n$: a kernel vector of any $x \in \mathbb{R}^d$.

Notations II

- P: an r-dimensional subspace of the feature space formed by the mapped training samples $\Phi(X)$.
- $\phi_P(x)$: the projection of $\phi(x)$ onto P. If x lies on P (e.g., one of the training samples), $\phi_P(x) = \phi(x)$.
- $\phi_w(x)$: the projection of $\phi(x)$ onto a one-dimensional vector space formed by a vector $w \in \mathbb{R}^f$, i.e., $\phi_w(x) = \langle w, \phi(x) \rangle$. In most cases, the vector w is restricted to reside in the subspace P, i.e., $w = \Phi(X)\alpha$ for some $\alpha \in \mathbb{R}^n$.

Notations III

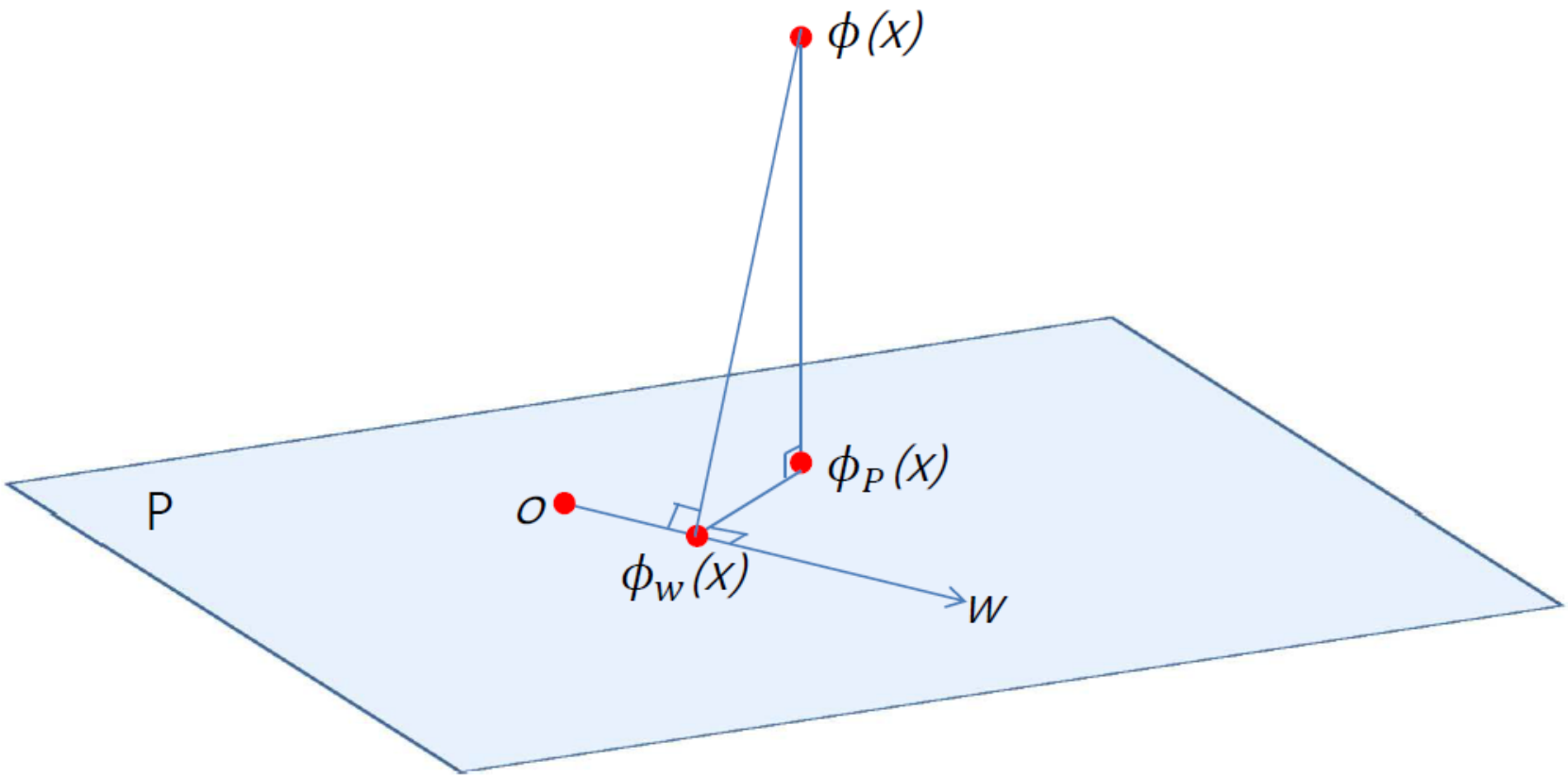


Figure : Projections in the feature space.

Bases of the Kernel Space P

Lemma

(Bases of P)

- $K = U\Lambda U^T$: *an eigenvalue decomposition of* K
  - $U = [u_1, \cdots, u_r] \in \mathbb{R}^{n \times r}$
  - $\Lambda = diag(\lambda_1, \cdots, \lambda_r) \in \mathbb{R}^{r \times r}$.
  - r: *rank of* K
- $\Longrightarrow \Pi \triangleq \Phi(X)U\Lambda^{-\frac{1}{2}} = [\pi_1, \cdots, \pi_r] \in \mathbb{R}^{f \times r}$: *orthonormal bases of* P.

Proof.

Because K is the outer product of $\Phi(X)$ and itself, it is obvious that r, the rank of K, is the dimension of the subspace spanned by $\Phi(X)$. By utilizing the orthonormality of U $(U^TU = I_r)$ and the definition of K $(K = \Phi(X)^T\Phi(X))$, it is easy to check that $\Pi^T\Pi = I_r$ and Π becomes orthonormal bases of P. □

Coordinate of $\phi_P(x)$

Theorem

(Coordinate of $\phi_P(x)$) For any $x \in \mathbb{R}^d$,

$$\phi_P(x) = \Pi y$$

$$\text{where } y = \Lambda^{-\frac{1}{2}} U^T k(x)$$

y: the coordinate of $\phi_P(x)$ w.r.t. the bases Π

Proof.

$\phi_P(x) \in P \longrightarrow \phi_P(x) = \Pi y$ for some $y \in \mathbb{R}^r$.

$$y^\star = \underset{y}{\operatorname{argmin}} \, ||\phi(x) - \Pi y||_2^2 = \underset{y}{\operatorname{argmin}} \, y^T y - 2\phi(x)^T \Pi y$$

$$\therefore y = \Pi^T \phi(x) = \Lambda^{-\frac{1}{2}} U^T k(x).$$

□

Nonlinear Projection $x \rightarrow y$

- $\phi_P(x)$: nonlinear mapping of x onto r-dim. subspace of Π.
- Therefore, $y = \Lambda^{-\frac{1}{2}} U^T k(x)$ can be thought as a direct nonlinear projection.
- For training data X, $Y = \Lambda^{\frac{1}{2}} U^T$.
  $\because Y = \Lambda^{-\frac{1}{2}} U^T K = \Lambda^{-\frac{1}{2}} U^T U \Lambda U^T = \Lambda^{\frac{1}{2}} U^T$.
- Note $K = U \Lambda U^T = Y^T Y$
  - C.f.) Cholesky decomp. of $K = Y'^T Y'$. (Y': unique upper triangle)
  - SVD of Y': $Y' = V \Lambda^{\frac{1}{2}} U^T = VY$, V: unitary
  - Y can be interpreted as a rotation of Y'. (coordinates are not unique, but can be determined up to rotations (Euclidean invariant))

Mean of the mapped training data

Lemma

(Mean of the mapped training data) The mapped training data $Y = \Lambda^{\frac{1}{2}} U^T$ *are centered, i.e.,* $\sum_{i=1}^{n} y_i = 0$.

Proof.

From the *Theorem*, $y_i = \Lambda^{-\frac{1}{2}} U^T k(x_i)$ and it becomes $\sum_{i=1}^{n} y_i = \Lambda^{-\frac{1}{2}} U^T \sum_{i=1}^{n} k(x_i) = \Lambda^{-\frac{1}{2}} U^T \Phi(X)^T \sum_{i=1}^{n} \phi(x_i) = 0$. □

Coordinate of the residual

For any $x \in \mathbb{R}^d$

- Augmented data: $X' \triangleq [X, x] \in \mathbb{R}^{d \times (n+1)}$
- Projection: $\Phi(X') \triangleq [\Phi(X), \phi(x)] \in \mathbb{R}^{f \times (n+1)}$
- Residual: $\delta\phi_P(x) \triangleq \phi(x) - \phi_P(x)$.

Then,

- $\Phi(X')$ lies in a $(r+1)$-dimensional subspace containing P.
- Coord. of $\Phi(X)$: $[Y^T, 0]^T \in \mathbb{R}^{(r+1) \times n}$
- Coord. of $\phi(x)$: $[y^T, y_{r+1}]^T \in \mathbb{R}^{r+1}$ where $y_{r+1} = \sqrt{k(x,x) - y^T y}$.
  - $\because \phi(x) = \phi_P(x) + \delta\phi_P(x)$ and $\phi_P(x) \perp \delta\phi_P(x)$, it becomes

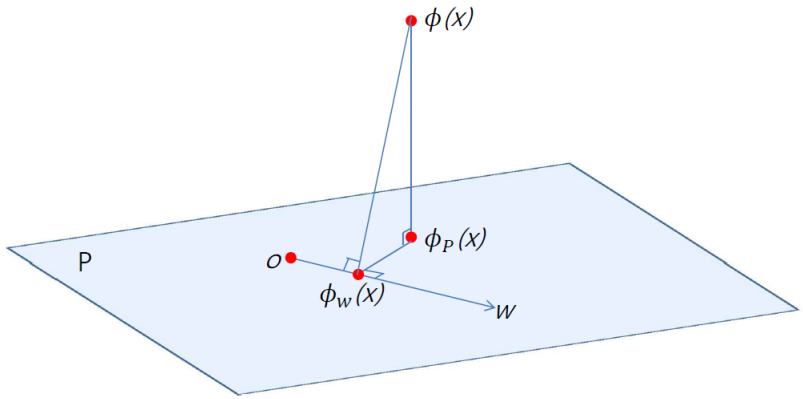
$$\begin{aligned} ||\delta\phi_P(x)||_2^2 &= ||\phi(x)||_2^2 - ||\phi_P(x)||_2^2 \\ &= k(x,x) - y^T y \end{aligned}$$

by Pythagorean trigonometric identity.

Coordinate of a vector w

Lemma

(Coordinate of a vector w) If a vector w in P can be written in the form of $w = \Phi(X)\alpha$, then it can also be written as $w = \Pi\beta$ where $\beta = Y\alpha$.



Proof.

Because Π is an orthonormal bases of P, any vector w in P can be written as $w = \Pi\Pi^T w$. Therefore,

$$\begin{aligned} w &= \Pi\Lambda^{-\frac{1}{2}}U^T\Phi(X)^T\Phi(X)\alpha = \Pi\Lambda^{-\frac{1}{2}}U^T K\alpha \\ &= \Pi Y\alpha = \Pi\beta. \end{aligned}$$

□

Note that $\beta = Y\alpha$ is the coordinate of w in P.

Coordinate of $\phi_w(x)$

Corollary

(Coordinate of $\phi_w(x)$) The projection $\phi_w(x)$ of $\phi(x)$ onto w can be obtained by $\phi_w(x) = \Pi\gamma$ where $\gamma = \frac{\beta\beta^T}{\beta^T\beta}y$.

Proof.

Let $w' = \frac{w}{||w||_2}$ be a unit vector. Then

$$\begin{aligned}
\phi_w(x) &= w'(w'^T\phi(x)) = w'w'^T(\phi_P(x) + \delta\phi_P(x)) \\
&= w'w'^T\phi_P(x) = w'w'^T\Pi\Lambda^{-\frac{1}{2}}U^Tk(x) \\
&= \frac{1}{\beta^T\beta}\Pi\beta\beta^T\Pi^T\Pi\Lambda^{-\frac{1}{2}}U^Tk(x) \\
&= \Pi\frac{\beta\beta^T}{\beta^T\beta}\Lambda^{-\frac{1}{2}}U^Tk(x) = \Pi\frac{\beta\beta^T}{\beta^T\beta}y = \Pi\gamma.
\end{aligned}$$

□

Centerization I

- Centerization is a necessary step in practical application of the kernel methods.
- $\Psi(X) \triangleq [\psi(x_1), \cdots, \psi(x_n)] \in \mathbb{R}^{f \times n}$: uncentered data in the feature space
- $\bar{\psi} \triangleq \frac{1}{n}\sum_{i=1}^{n} \psi(x_i) = \frac{1}{n}\Psi(X)\mathbf{1}_n \in \mathbb{R}^f$: mean of $\Psi(X)$

$$\begin{aligned}\Phi(X) &= \Psi(X) - \bar{\psi}\mathbf{1}_n^T = \Psi(X)(I_n - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^T) \\ &= \Psi(X)(I_n - E_n).\end{aligned}$$

  - $\mathbf{1}_n = [1, \cdots, 1]^T \in \mathbb{R}^n$
  - I_n: $n \times n$ identity matrix
  - $E_n \triangleq \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^T$

Centerization II

- $\kappa(a,b) = \psi(a)^T\psi(b)$: uncenterd kernel function
- $\mathcal{K} \triangleq [\kappa(x_i,x_j)] = \Psi(X)^T\Psi(X) \in \mathbb{R}^{n\times n}$: uncentered kernel matrix
- $K = \Phi(X)^T\Phi(X) = (I_n - E_n)\mathcal{K}(I_n - E_n)$: centered kernel matrix
- $\kappa(x) \triangleq [\kappa(x_1,x),\cdots,\kappa(x_n,x)]^T \in \mathbb{R}^n$: uncentered kernel vector for any $x \in \mathbb{R}^d$
- Centered kernel vector:

$$\begin{aligned} k(x) &= \Phi(X)^T\phi(x) \\ &= [\Psi(X)(I_n - E_n)]^T(\psi(x) - \bar{\psi}) \\ &= (I_n - E_n)\Psi(X)^T(\psi(x) - \frac{1}{n}\Psi(X)\mathbf{1}_n) \\ &= (I_n - E_n)[\kappa(x) - \frac{1}{n}\mathcal{K}\mathbf{1}_n]. \end{aligned}$$

Nonlinear Projection Trick I

Algorithm: Kernel method KM (Input: X, x, $\kappa(\cdot,\cdot)$, method M)

- Training phase
  1. Compute the uncentered kernel matrix $\mathcal{K}$ such that $\mathcal{K}_{ij} = \kappa(x_i, x_j)$.
  2. Compute the centered kernel K by $K = (I_n - E_n)\mathcal{K}(I_n - E_n)$.
  3. Obtain the eigenvalue decomposition of K such that $K = U\Lambda U^T$ where Λ is composed of only the nonzero eigenvalues of K and the columns of U are the corresponding unit eigenvectors of K.
  4. Compute Y, the coordinates of $\Phi(X)$, by $Y = \Lambda^{\frac{1}{2}} U^T$.
  5. Apply the method M to Y, then it is equivalent to applying the kernel method KM to X, i.e., $M(Y) \equiv KM(X)$.

Nonlinear Projection Trick II

- Test phase
  1. Compute the uncentered kernel vector $\kappa(x)$.
  2. Compute the centered kernel vector $k(x)$ by $k(x) = (I_n - E_n)[\kappa(x) - \frac{1}{n}\mathcal{K}\mathbf{1}_n]$.
  3. Obtain y, the coordinate of $\phi(x)$, by $y = \Lambda^{-\frac{1}{2}} U^T k(x)$.
  4. Apply the method M to y, then it is equivalent to applying the kernel method KM to x, i.e., $M(y) \equiv KM(x)$.

Application: KPCA I

Objective

$$w^{\star} = \underset{w}{\operatorname{argmax}} ||w^T \Phi(X)||_2^2$$
$$\text{s. t. } ||w||_2 = 1.$$

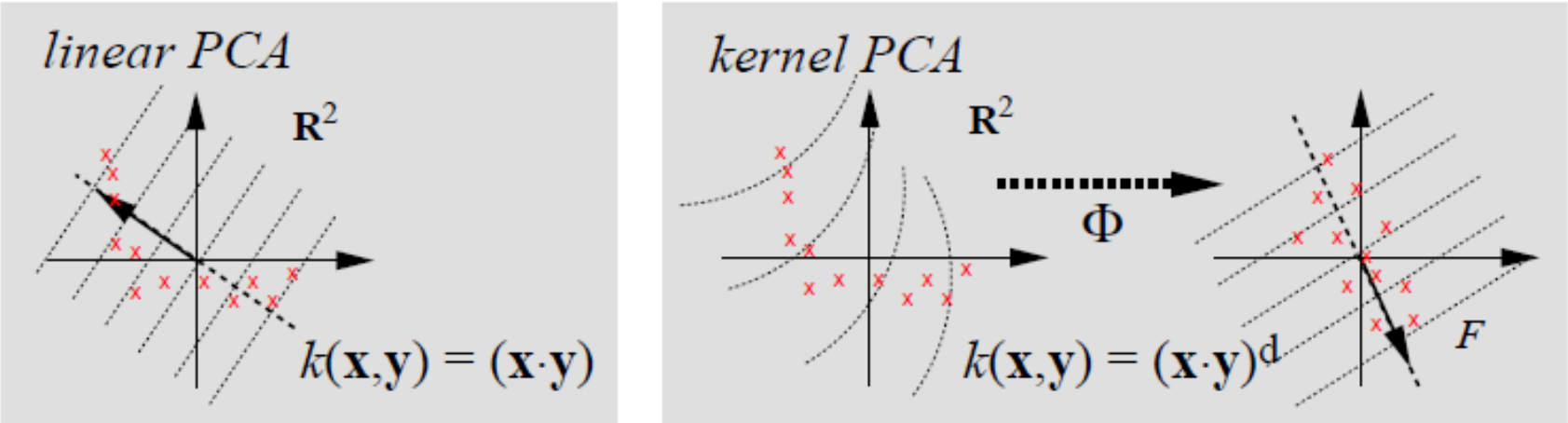


Figure : Basic idea of KPCA (from [2])

Application: KPCA II

Kernel Trick

- *Scatter matrix:*
  $S_f w_i = \lambda_i w_i$
- *Trick:* $w = \Phi(X)\alpha$
  $\rightarrow K\alpha_i = \lambda_i \alpha_i$
- *Solution:* $\alpha_i = u_i \lambda_i^{-\frac{1}{2}}$
- *Nonlinear feature:*
  $z = W^T \phi(x) = A^T k(x)$
  $= \Lambda_m^{-\frac{1}{2}} U_m^T k(x).$

Nonlinear Projection Trick

- *Nonlinear projection:*
  $Y = \Lambda^{\frac{1}{2}} U^T$
- *Scatter matrix:*
  $S_Y = YY^T = \Lambda$
  $\rightarrow e_i$: *eigenvector*
- *Nonlinear feature:*
  $z_i = e_i^T y =$
  $e_i^T \Lambda^{-\frac{1}{2}} U^T k(x) =$
  $\lambda_i^{-\frac{1}{2}} u_i^T k(x).$
  $\rightarrow z = \Lambda_m^{-\frac{1}{2}} U_m^T k(x).$

Application: KSVM I

Training data: $\{(x_i, c_i)\}_{i=1}^n$ where $x_i \in \mathbb{R}^d$ and $c_i \in \{-1, 1\}$

Objective

$$(w^\star, b^\star) = \underset{(w,b)}{\operatorname{argmin}} \frac{1}{2}||w||_2^2$$

subject to $\quad c_i(w^T \phi(x_i) + b) \geq 1 \quad \forall i = 1, \cdots, n$

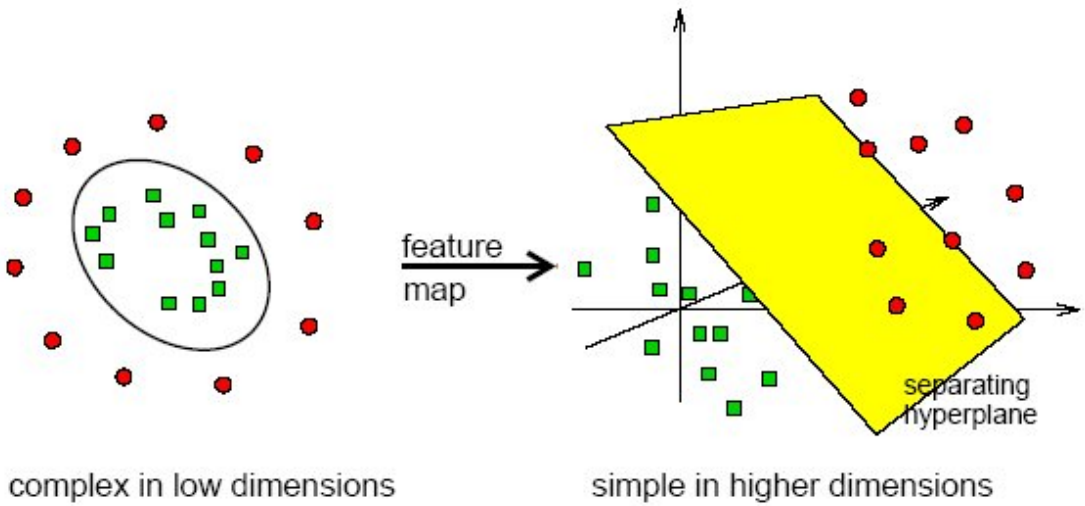


Figure : Basic idea of KSVM

Kernel Trick

- *Dual form by Lagrange multipliers* $\{\alpha_i\}_{i=1}^n$

$$\alpha^\star = \underset{\alpha}{\operatorname{argmax}} \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i,j=1}^n \alpha_i \alpha_j c_i c_j k(x_i, x_j)$$
$$\text{subject to } \sum_{i=1}^n \alpha_i c_i = 0 \text{ and } \alpha_i \geq 0 \quad \forall i = 1, \cdots, n \tag{1}$$

- *Once* α_i*'s are found,* $w = \sum_{i=1}^n \alpha_i c_i \phi(x_i)$ *and* b *can be computed so that it meets the KKT condition* $\alpha_i[c_i(w^T\phi(x_i) + b) - 1] = 0$ *for all* $i = 1, \cdots, n$.
- *Classification of* x*:* $\text{sgn}(w^T\phi(x) + b)$.
- *Need not have to find an explicit expression for* w
  $\because w^T\phi(x) = \sum_{i=1}^n \alpha_i c_i k(x_i, x)$

Application: KSVM III

Nonlinear Projection Trick

- *Nonlinear projection:* $Y = \Lambda^{\frac{1}{2}} U^T = [y_1, \cdots, y_n]$
- *Then a linear SVM is solved for* $\{y_i, c_i\}_{i=1}^n$.
- *Primal problem:*

$$(v^\star, d^\star) = \underset{(v,d)}{\operatorname{argmin}} \frac{1}{2}||v||_2^2$$

$$\text{subject to} \quad c_i(v^T y_i + d) \geq 1 \quad \forall i = 1, \cdots, n$$

- *Dual problem:*

$$\beta^\star = \underset{\beta}{\operatorname{argmax}} \sum_{i=1}^n \beta_i - \frac{1}{2} \sum_{i,j=1}^n \beta_i \beta_j c_i c_j y_i^T y_j \tag{2}$$

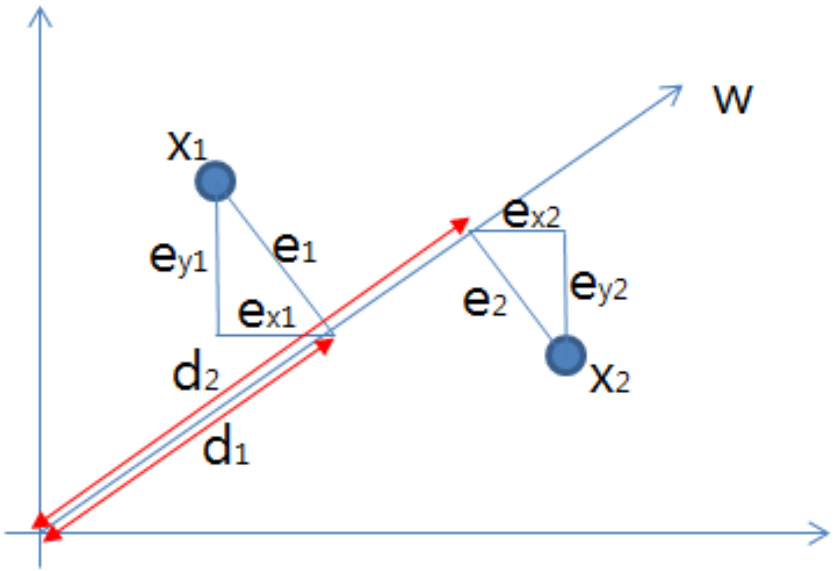
$$\text{subject to} \sum_{i=1}^n \beta_i c_i = 0 \text{ and } \beta_i \geq 0 \quad \forall i = 1, \cdots, n$$

- *(1) and (2) are exactly the same* $\because k(x_i, x_j) = y_i^T y_j$

Other applications

- The above examples (KPCA and KSVM) use L_2 norm in the optimization.
- In this case, we have shown that KT and NPT are equivalent.
- However, when other norm is used in the optimization, KT cannot be used because dot product is not used.
- An example will follow - PCA-L1.

Formulation of PCA-L1 [3] I



- Motivation:
  - Previous methods (L1-PCA, R1-PCA) minimizes reconstruction error (E, 1st interpretation).
  - Instead of solving minimization problem, **maximize the dispersion of projection** (**D, 2nd interpretation**).

Formulation of PCA-L1 [3] II

- Problem formulation

$$W^* = \underset{W}{\operatorname{argmax}} D_1(W) \text{ subject to } WW^T = I_m \tag{3}$$

$$D_1(W) = \sum_{i=1}^{n} ||W^T \boldsymbol{x}_i||_1 = \sum_{i=1}^{n} \sum_{k=1}^{m} |\boldsymbol{w}_k^T \boldsymbol{x}_i| \tag{4}$$

- Pros and Cons of (3)
  - (3) is invariant to rotations.
  - As R1-PCA, the solution depends on m.
- Smaller problem: $m = 1$

$$\boldsymbol{w}^* = \underset{\boldsymbol{w}}{\operatorname{argmax}} ||\boldsymbol{w}^T X||_1 = \underset{\boldsymbol{w}}{\operatorname{argmax}} \sum_{i=1}^{n} |\boldsymbol{w}^T \boldsymbol{x}_i| \tag{5}$$

$$\text{subject to} \quad ||\boldsymbol{w}||_2 = 1.$$

Algorithm: PCA-L1

1. Initialization: Pick any $\boldsymbol{w}(0)$. Set $\boldsymbol{w}(0) \leftarrow \boldsymbol{w}(0)/||\boldsymbol{w}(0)||_2$ and $t = 0$.

Algorithm: PCA-L1

1. Initialization: Pick any $\boldsymbol{w}(0)$. Set $\boldsymbol{w}(0) \leftarrow \boldsymbol{w}(0)/||\boldsymbol{w}(0)||_2$ and $t = 0$.
2. Polarity check: For all $i \in \{1, \cdots, n\}$, if $\boldsymbol{w}^T(t)\boldsymbol{x}_i < 0$, $p_i(t) = -1$, otherwise $p_i(t) = 1$.

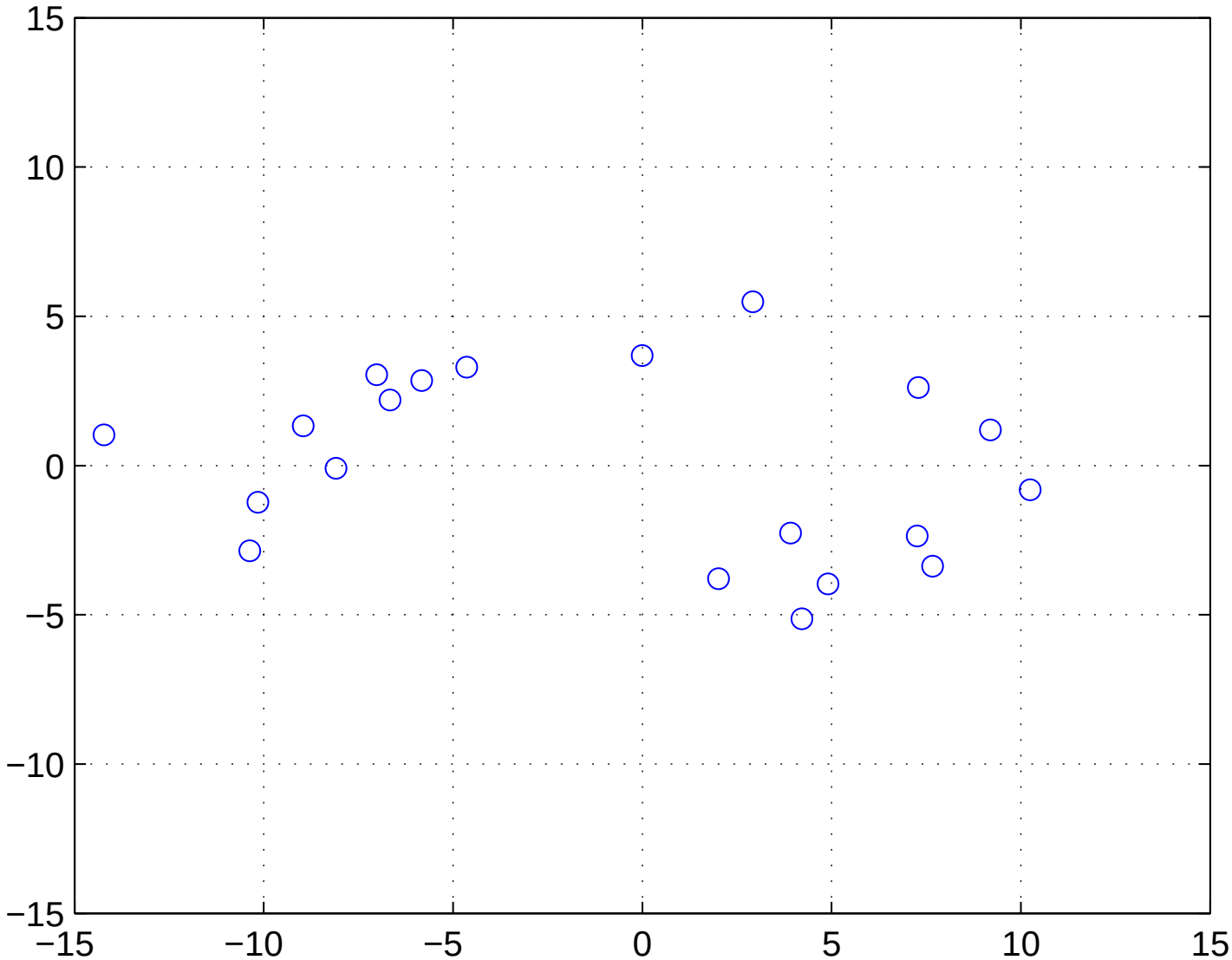
Algorithm: PCA-L1

1. Initialization: Pick any $\boldsymbol{w}(0)$. Set $\boldsymbol{w}(0) \leftarrow \boldsymbol{w}(0)/||\boldsymbol{w}(0)||_2$ and $t = 0$.
2. Polarity check: For all $i \in \{1, \cdots, n\}$, if $\boldsymbol{w}^T(t)\boldsymbol{x}_i < 0$, $p_i(t) = -1$, otherwise $p_i(t) = 1$.
3. Flipping and maximization: Set $t \leftarrow t + 1$ and $\boldsymbol{w}(t) = \sum_{i=1}^{n} p_i(t-1)\boldsymbol{x}_i$. Set $\boldsymbol{w}(t) \leftarrow \boldsymbol{w}(t)/||\boldsymbol{w}(t)||_2$.

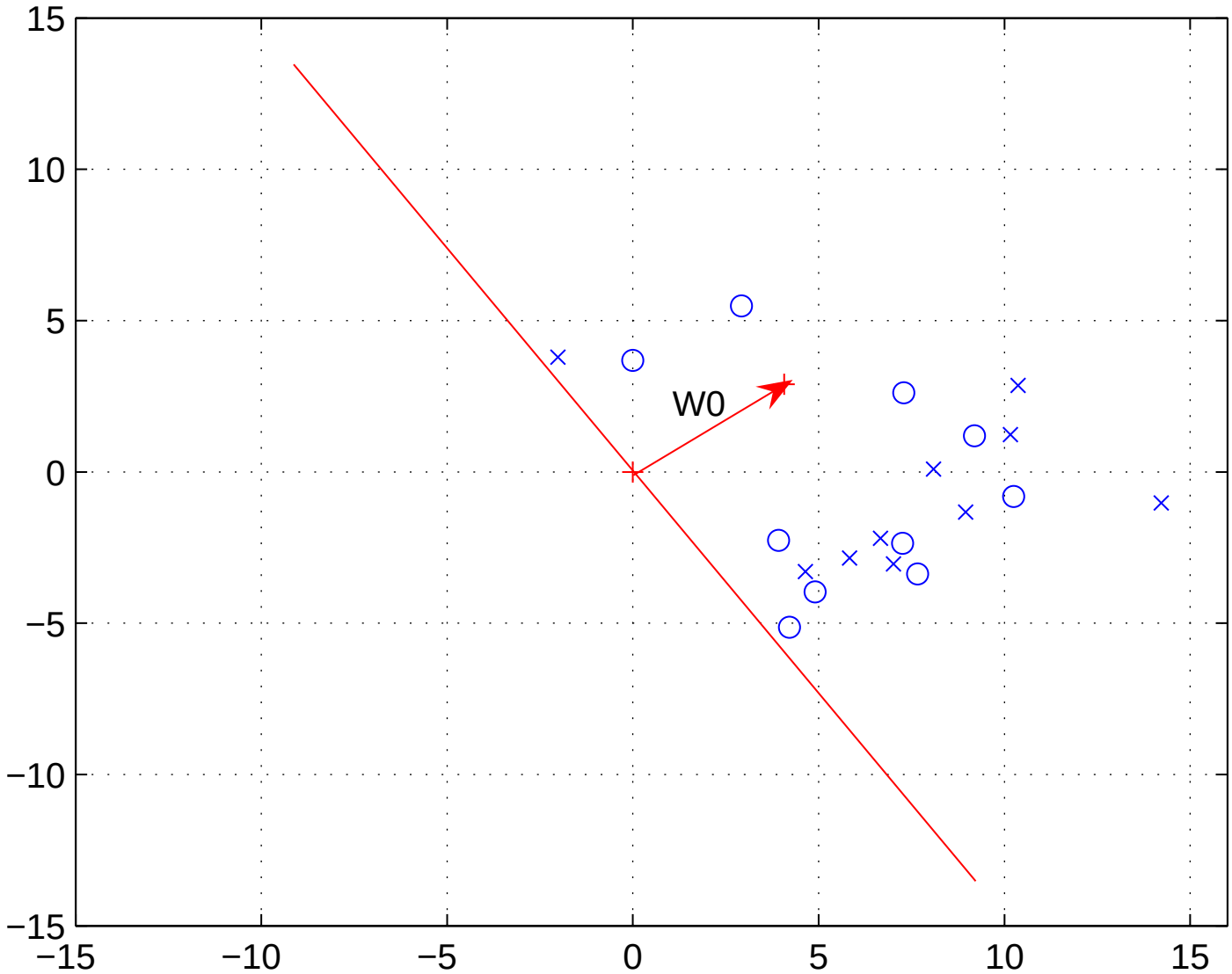
Algorithm: PCA-L1

1. Initialization: Pick any $\boldsymbol{w}(0)$. Set $\boldsymbol{w}(0) \leftarrow \boldsymbol{w}(0)/||\boldsymbol{w}(0)||_2$ and $t = 0$.
2. Polarity check: For all $i \in \{1, \cdots, n\}$, if $\boldsymbol{w}^T(t)\boldsymbol{x}_i < 0$, $p_i(t) = -1$, otherwise $p_i(t) = 1$.
3. Flipping and maximization: Set $t \leftarrow t + 1$ and $\boldsymbol{w}(t) = \sum_{i=1}^{n} p_i(t-1)\boldsymbol{x}_i$. Set $\boldsymbol{w}(t) \leftarrow \boldsymbol{w}(t)/||\boldsymbol{w}(t)||_2$.
4. Convergence check:
   a. If $\boldsymbol{w}(t) \neq \boldsymbol{w}(t-1)$, go to Step 2.
   b. Else if there exists i such that $\boldsymbol{w}^T(t)\boldsymbol{x}_i = 0$, set $\boldsymbol{w}(t) \leftarrow (\boldsymbol{w}(t) + \Delta\boldsymbol{w})/||\boldsymbol{w}(t) + \Delta\boldsymbol{w}||_2$ and go to Step 2. Here, $\Delta\boldsymbol{w}$ is a small nonzero random vector.
   c. Otherwise, set $\boldsymbol{w}^\star = \boldsymbol{w}(t)$ and stop.

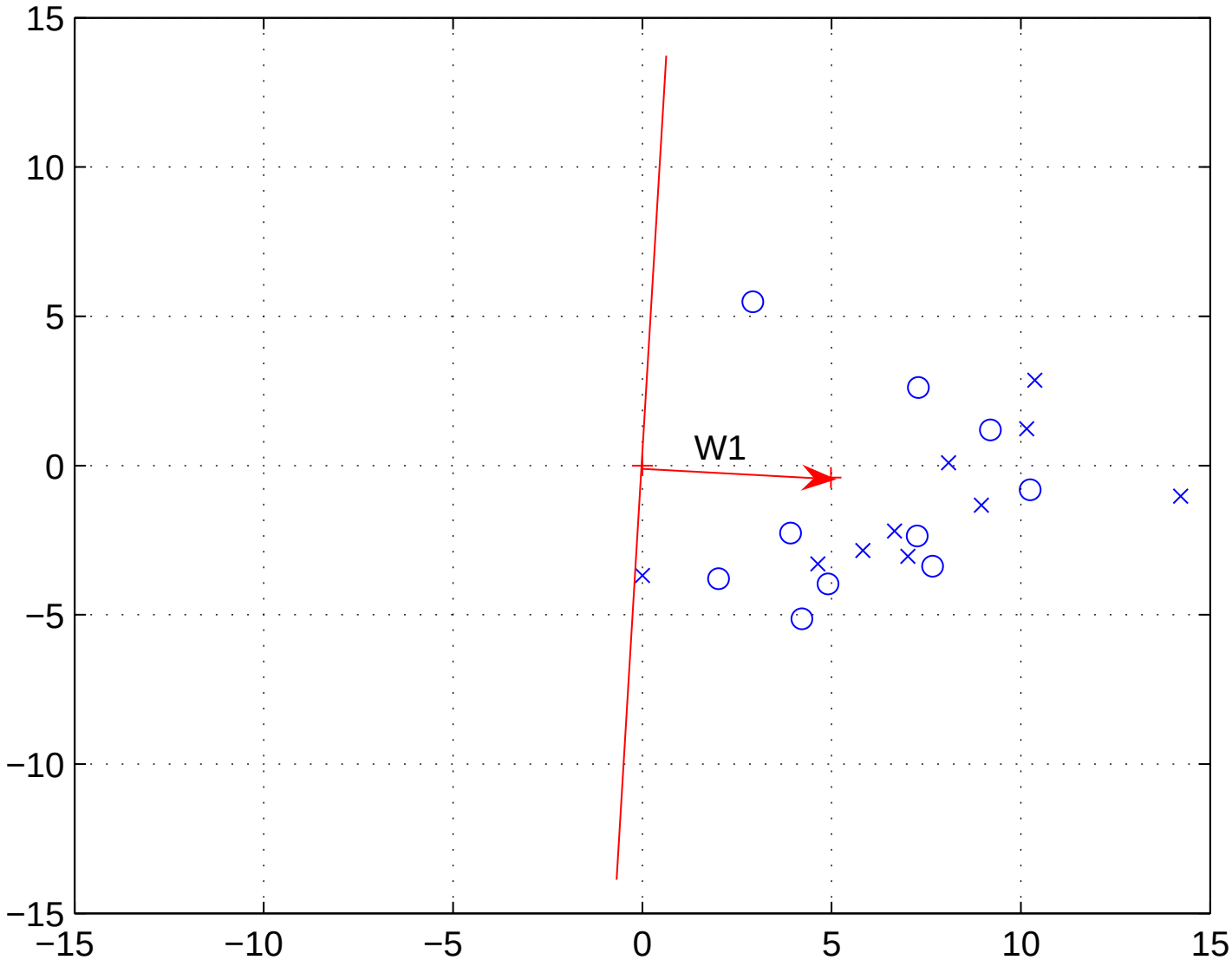
Simple Example (2D case)



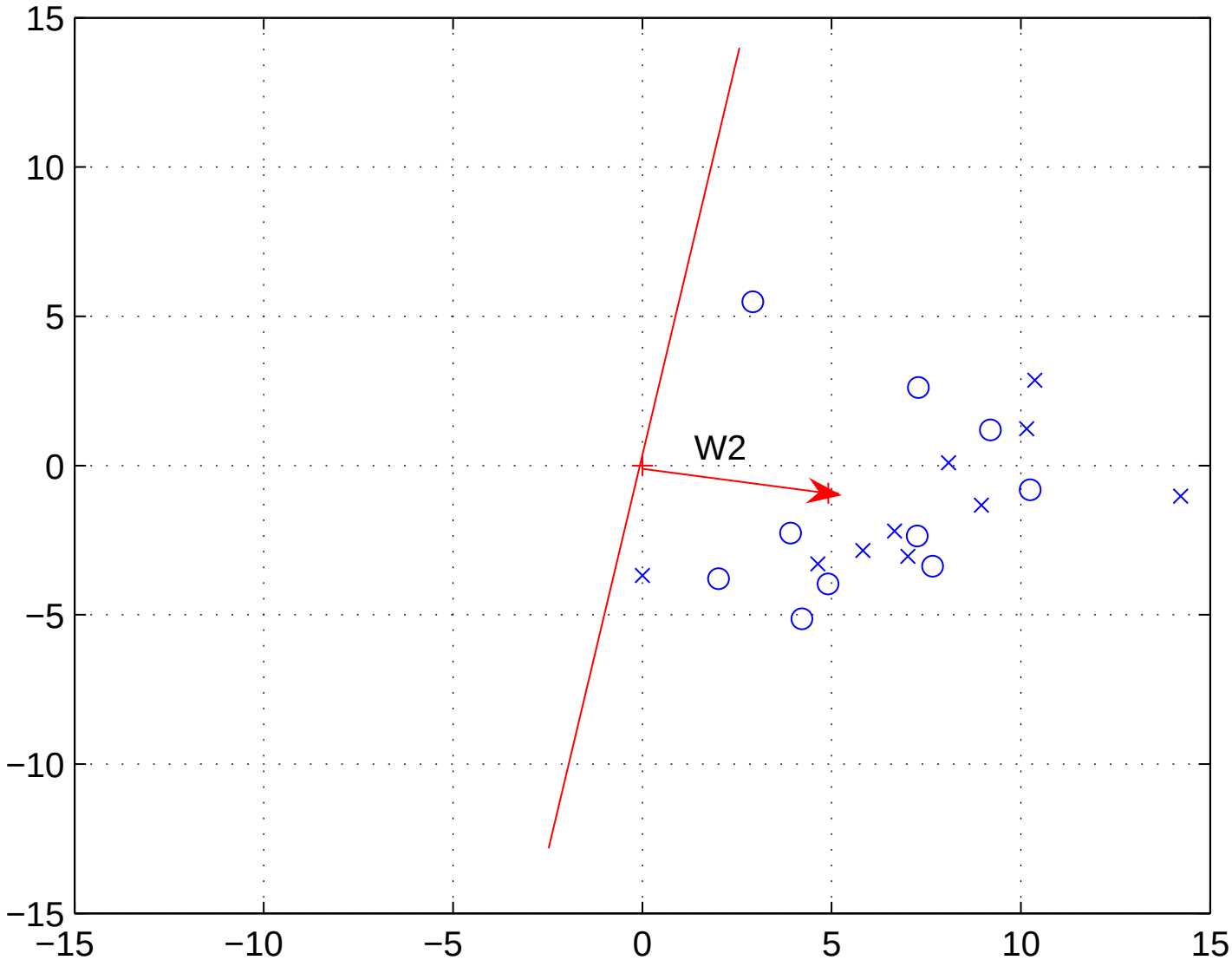
Simple Example (2D case)



Simple Example (2D case)



Simple Example (2D case)



Application: KPCA-L1

Objective

$$w^\star = \underset{w}{\operatorname{argmax}} ||w^T \Phi(X)||_1 = \underset{w}{\operatorname{argmax}} \sum_{i=1}^{n} |w^T \phi(x_i)| \quad (6)$$

$$\text{subject to } ||w||_2 = 1.$$

- Kernel trick is not applicable to KPCA-L1.
- The PCA-L1 algorithm can directly be applied to Y to obtain the KPCA-L1. (NPT)

Conclusions & Future Works

- NPT was proposed as an alternative to the KT.
- The two are equivalent.
- NPT is intuitive and easy to implement.
- Eigenvalue decomposition (or singular value decompotion) of Kernel matrix plays an essential role in NPT.
- NPT widens the applicability of Kernel methods to any problems that can be done in the input space. (e.g. gradient search, L1 optimization, $\cdots$)

References

[1] N. Kwak,
"Nonlinear Projection Trick in Kernel Methods: An Alternative to the Kernel Trick,"
IEEE TNNLS, vol. 24, no. 12, pp. 2113–2119, Dec. 2013.

[2] B. Schölkopf1, A. Smola and K.R. Müller,
Kernel Principal Component Analysis
ADVANCES IN KERNEL METHODS - SUPPORT VECTOR LEARNING, MIT Press pp. 327–352, 1999.

[3] N. Kwak,
"Principal component analysis based on L1 norm maximization,"
IEEE TPAMI, vol. 30, no. 9, pp. 1672–1680, Sep. 2008.