Kernel Probabilistic Distance Clustering

Ozkan, Dilay[1] and Iyigun, Cem[1]

[1]*Department of Industrial Engineering, Middle East Technical University, Ankara, Turkey*

Consider we have an $N \times M$ data set $\mathbf{X}$, where N and M represent number of data points and features, respectively. We show data point $\mathbf{x}_i$ as M-dimensional vector, where $i = 1, ..., N$. There are T clusters, and cluster centers (prototypes) are shown by $\mathbf{c}_j$, where $j = 1, ..., T$. Then p_{ij} is the probability of data point $\mathbf{x}_i$ belonging to cluster $\mathbf{c}_j$.

1 Kernel Probabilistic Distance Clustering Algorithms

In Kernel Probabilistic Distance Clustering there are two spaces, which are *input space* (or feature space) and *kernel space*. *Input space* is the original space of the data points. On the other hand, *kernel space* represents the space where data points are mapped into via ϕ function. The dimension of the *input space* is the number of features data points have. However, *kernel space*'s dimension is determined by the ϕ function to be used. Since we will not know the ϕ function, we will not be able to detect the dimension of the *kernel space*.

1.1 Kernel Pd-clustering in Kernel Space

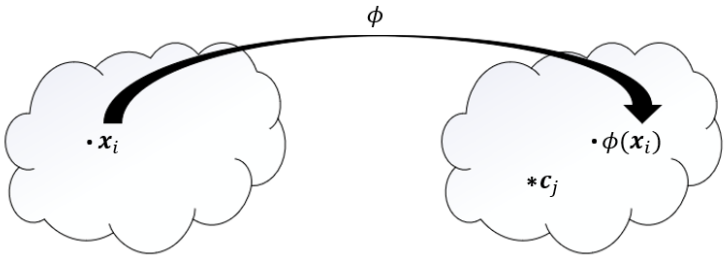

Figure 1: Kernel Pd-clustering in Input and Kernel Space

In Kernel Pd-clustering in *kernel space* (**KPd Algorithm**) centers are defined in the *kernel space* (or mapped space). In fact, center update rules are made in the *kernel space*. The data points are in the *input space*, and they are mapped into the *kernel space* via ϕ function as shown in Figure 1. Since the update rules in the algorithm are valid for all kernel functions, prior knowledge of the kernel function to be used is not required. In addition, cluster prototypes do not have an explicit representation in both *kernel* and *input spaces*.

As a distance metric Euclidean distance is considered. In general, Euclidean distance between $\mathbf{x}_i$ and $\mathbf{c}_j$, $d(\mathbf{x}_i, \mathbf{c}_j)$, is

$$d(\mathbf{x}_i, \mathbf{c}_j) = \sqrt{\sum_{m=1}^{M} (\mathbf{x}_{im} - \mathbf{c}_{jm})^2} = \|\mathbf{x}_i - \mathbf{c}_j\|.$$

If mapping of $\mathbf{x}_i$ into *kernel space* is considered, then $d(\phi(\mathbf{x}_i), \mathbf{c}_j) = \|\phi(\mathbf{x}_i) - \mathbf{c}_j\|$.

The optimization problem fo clustering in Section **??** becomes

$$\min \quad \sum_{i=1}^{N} \sum_{j=1}^{T} p_{ij}^2 \|\phi(\mathbf{x}_i) - \mathbf{c}_j\| \tag{1}$$

$$\text{s.t.} \quad \sum_{j=1}^{T} p_{ij} = 1 \qquad \forall i \tag{2}$$

$$p_{ij} \geq 0. \qquad \forall i, j \tag{3}$$

$\|\phi(\mathbf{x}_i) - \mathbf{c}_j\|$ is simply denoted by d_{ij}. The nonlinear objective function is convex in terms of p_{ij} and d_{ij}. This is because when p_{ij}'s are given and d_{ij}'s become unknown, the objective becomes convex. Similarly, given d_{ij}'s, the objective function in (1) will be a function of d_{ij}'s which is a convex function. Considering the fact that constraints are all linear, i.e. they are also convex, the problem becomes a convex optimization problem.

When $\mathbf{c}_j$'s are given, the Lagrangian becomes

$$L(\mathbf{P}, \mathbf{C}, \mathbf{\Lambda}) = \sum_{i=1}^{N} \sum_{j=1}^{T} p_{ij}^2 \|\phi(\mathbf{x}_i) - \mathbf{c}_j\| - \sum_{i=1}^{N} \lambda_i \left(\sum_{j=1}^{T} p_{ij} - 1 \right). \tag{4}$$

When we take the derivative with respect to p_{ij} and make it equal to zero, we obtain

$$2\, p_{ij} \|\phi(\mathbf{x}_i) - \mathbf{c}_j\| - \lambda_i = 0$$

$$\implies p_{ij} = \frac{\lambda_i}{2 \|\phi(\mathbf{x}_i) - \mathbf{c}_j\|}. \tag{5}$$

When we substitute (5) into (4), we get

$$\sum_{j=1}^{T} p_{ij} = 1 \implies \sum_{j=1}^{T} \frac{\lambda_i}{2\|\phi(\mathbf{x}_i) - \mathbf{c}_j\|} = 1$$

$$\implies \lambda_i = \frac{1}{\sum_{j=1}^{T} \frac{1}{2 \|\phi(\mathbf{x}_i) - \mathbf{c}_j\|}}. \tag{6}$$

If (6) is substituted into (5), we find out p_{ij} as

$$p_{ij} = \frac{\dfrac{1}{\|\phi(\mathbf{x}_i) - \mathbf{c}_j\|}}{\displaystyle\sum_{t=1}^{T} \frac{1}{\|\phi(\mathbf{x}_i) - \mathbf{c}_t\|}}. \tag{7}$$

Fixing p_{ij} and taking the derivative of (4) with respect to $\mathbf{c}_j$ gives

$$\mathbf{c}_j = \frac{\displaystyle\sum_{i=1}^{N} \frac{p_{ij}^2\, \phi(\mathbf{x}_i)}{\|\phi(\mathbf{x}_i) - \mathbf{c}_j\|}}{\displaystyle\sum_{i=1}^{N} \frac{p_{ij}^2}{\|\phi(\mathbf{x}_i) - \mathbf{c}_j\|}}. \tag{8}$$

However, the cluster centers cannot be calculated using (8) since $\phi(\mathbf{x}_i)$'s are not known explicitly. Therefore, there is no explicit representation of cluster centers. On the other side, the centers can be described in terms of the data points that belong to the corresponding clusters. That is, the centers will be the convex combination of $\phi(\mathbf{x}_i)$'s. Then (8) can be rewritten as

$$\mathbf{c}_j = \sum_{i=1}^{N} \beta_{ij}\, \phi(\mathbf{x}_i), \tag{9}$$

where

$$\beta_{ij} = \frac{\dfrac{p_{ij}^2}{\|\phi(\mathbf{x}_i) - \mathbf{c}_j\|}}{\displaystyle\sum_{i=1}^{N} \frac{p_{ij}^2}{\|\phi(\mathbf{x}_i) - \mathbf{c}_j\|}}.$$

When (9) is substituted into $\mathbf{c}_j$'s in the objective function, then the problem becomes

$$\min \quad \sum_{i=1}^{N}\sum_{j=1}^{T} p_{ij}^2 \left\|\phi(\mathbf{x}_i) - \sum_{i=1}^{N} \beta_{ij}\, \phi(\mathbf{x}_i)\right\| \tag{10}$$

$$\text{s.t.} \quad \sum_{j=1}^{T} p_{ij} = 1 \qquad \forall i \tag{11}$$

$$p_{ij} \geq 0. \qquad \forall i, j \tag{12}$$

The Lagrangian of (10) becomes

$$L(\mathbf{P}, \boldsymbol{\beta}, \boldsymbol{\Lambda}) = \sum_{i=1}^{N}\sum_{j=1}^{T} p_{ij}^2 \left\|\phi(\mathbf{x}_i) - \sum_{i=1}^{N} \beta_{ij}\, \phi(\mathbf{x}_i)\right\| - \sum_{i=1}^{N} \lambda_i \left(\sum_{j=1}^{T} p_{ij} - 1\right). \tag{13}$$

When we fix β_{ij} and take the derivative with respect to p_{ij} and make it equal to zero, we

obtain

$$p_{ij} = \frac{\dfrac{1}{\left\|\phi\left(\mathbf{x}_i\right) - \sum_{k=1}^{N} \beta_{kj}\, \phi(\mathbf{x}_k)\right\|}}{\sum_{t=1}^{T} \dfrac{1}{\left\|\phi\left(\mathbf{x}_i\right) - \sum_{k=1}^{N} \beta_{kt}\, \phi(\mathbf{x}_k)\right\|}}. \tag{14}$$

Note that p_{ij}'s are in the distance term in (13). We can write the distance term as

$$\left\|\phi\left(\mathbf{x}_i\right) - \sum_{i=1}^{N} \beta_{ij}\, \phi(\mathbf{x}_i)\right\| = \sqrt{\left\|\phi\left(\mathbf{x}_i\right) - \sum_{i=1}^{N} \beta_{ij}\, \phi(\mathbf{x}_i)\right\|^2},$$

where

$$\left\|\phi\left(\mathbf{x}_i\right) - \sum_{i=1}^{N} \beta_{ij} \phi(\mathbf{x}_i)\right\| = \sqrt{\phi\left(\mathbf{x}_i\right)\phi\left(\mathbf{x}_i\right) - 2\sum_{k=1}^{N} \beta_{kj}\, \phi\left(\mathbf{x}_i\right)\phi\left(\mathbf{x}_k\right) + \sum_{k=1}^{N}\sum_{l=1}^{N} \beta_{kj}\, \beta_{lj}\, \phi\left(\mathbf{x}_k\right)\phi\left(\mathbf{x}_l\right)}.$$

We replace $\phi\left(\mathbf{x}_i\right)\phi\left(\mathbf{x}_k\right)$'s with $K(\mathbf{x}_i, \mathbf{x}_k)$'s, and we get

$$\left\|\phi\left(\mathbf{x}_i\right) - \sum_{i=1}^{N} \beta_{ij}\, \phi(\mathbf{x}_i)\right\| = \sqrt{K(\mathbf{x}_i, \mathbf{x}_i) - 2\sum_{k=1}^{N} \beta_{kj}\, K(\mathbf{x}_i, \mathbf{x}_k) + \sum_{k=1}^{N}\sum_{l=1}^{N} \beta_{kj}\, \beta_{lj}\, K(\mathbf{x}_k, \mathbf{x}_l)}. \tag{15}$$

So (15) can be calculated using kernel trick if β_{ij} values are known, so there is no need to know $\phi(\mathbf{x}_i)$. Now, we substitute (15) into (13) for the norm term, and take the derivative of (13), we get

$$\frac{\partial L}{\partial \beta_{ij}} = \sum_{h=1}^{N} \frac{1}{2}\, p_{hj}^2\, \frac{\left(-2\, K(\mathbf{x}_h, \mathbf{x}_i) + 2\sum_{k=1}^{N} \beta_{kj}\, K(\mathbf{x}_i, \mathbf{x}_k)\right)}{\left\|\phi\left(\mathbf{x}_h\right) - \sum_{l=1}^{N} \beta_{lj}\, \phi(\mathbf{x}_l)\right\|}. \tag{16}$$

When (16) is equal to 0, we obtain

$$\sum_{h=1}^{N} p_{hj}^2\, \frac{K(\mathbf{x}_h, \mathbf{x}_i)}{\left\|\phi\left(\mathbf{x}_h\right) - \sum_{l=1}^{N} \beta_{lj}\, \phi(\mathbf{x}_l)\right\|} = \sum_{h=1}^{N}\sum_{k=1}^{N} p_{hj}^2\, \frac{\beta_{kj}\, K(\mathbf{x}_k, \mathbf{x}_i)}{\left\|\phi\left(\mathbf{x}_h\right) - \sum_{l=1}^{N} \beta_{lj}\, \phi(\mathbf{x}_l)\right\|}$$

$$= \sum_{h=1}^{N}\sum_{k=1,k\neq i}^{N} p_{hj}^2\, \frac{\beta_{kj}\, K(\mathbf{x}_k, \mathbf{x}_i)}{\left\|\phi\left(\mathbf{x}_h\right) - \sum_{l=1}^{N} \beta_{lj}\, \phi(\mathbf{x}_l)\right\|} + \beta_{ij} \sum_{h=1}^{N} p_{hj}^2\, \frac{K(\mathbf{x}_i, \mathbf{x}_i)}{\left\|\phi\left(\mathbf{x}_h\right) - \sum_{l=1}^{N} \beta_{lj}\, \phi(\mathbf{x}_l)\right\|}$$

$$\implies \beta_{ij} = \frac{\displaystyle\sum_{h=1}^{N} \frac{p_{hj}^2}{\left\|\phi\left(\mathbf{x}_h\right) - \sum_{l=1}^{N} \beta_{lj}\, \phi(\mathbf{x}_l)\right\|}\left[K(\mathbf{x}_h, \mathbf{x}_i) - \sum_{k=1,k\neq i}^{N} \beta_{kj}\, K(\mathbf{x}_k, \mathbf{x}_i)\right]}{\displaystyle\sum_{h=1}^{N} p_{hj}^2\, \frac{K(\mathbf{x}_i, \mathbf{x}_i)}{\left\|\phi\left(\mathbf{x}_h\right) - \sum_{l=1}^{N} \beta_{lj}\, \phi(\mathbf{x}_l)\right\|}} \tag{17}$$

β_{ij}'s are updated as in (17). Using (14) we update p_{ij}'s. In that formula since $\mathbf{c}_j$'s are substituted with β_{ij}'s and $\phi(\mathbf{x}_i)$'s, there is still no need to know $\mathbf{c}_j$'s explicitly. The norm terms in (14) are calculated as in (15) using kernel trick.

We can define β_j as the column vector containing all β_{ij}'s for a given cluster j, then (13) is written as

$$L(\mathbf{P}, \boldsymbol{\beta}, \boldsymbol{\Lambda}) = \sum_{i=1}^{N}\sum_{j=1}^{T} p_{ij}^2 \left\|\mathbf{K}_{ii} - 2\boldsymbol{\beta}_j^T \mathbf{K}_i + \boldsymbol{\beta}_j^T \mathbf{K}\boldsymbol{\beta}_j\right\| - \sum_{i=1}^{N} \lambda_i \left(\sum_{j=1}^{T} p_{ij} - 1\right), \tag{18}$$

where $\mathbf{K}$ is the kernel matrix, $\mathbf{K}_i$ represents the i^{th} column of kernel matrix, and $\mathbf{K}_{ii}$ is the i^{th} row and i^{th} column element in that matrix. Taking the derivative of (18) with respect to $\boldsymbol{\beta}_j$ gives

$$\frac{\partial L}{\partial \boldsymbol{\beta}_j} = \sum_{i=1}^{N} p_{ij}^2 \left\|\mathbf{K}_{hh} - 2\boldsymbol{\beta}_j^T \mathbf{K}_h + \boldsymbol{\beta}_j^T \mathbf{K}\boldsymbol{\beta}_j\right\| - \sum_{i=1}^{N} \lambda_i \left(\sum_{j=1}^{T} p_{ij} - 1\right), \tag{19}$$

Making (19) equal to 0 yields

$$\boldsymbol{\beta}_j = \frac{\sum_{i=1}^{N} p_{ij}^2 \dfrac{\mathbf{K}^{-1}\mathbf{K}_i}{\left\|\mathbf{K}_{ii} - 2\boldsymbol{\beta}_j^T \mathbf{K}_i + \boldsymbol{\beta}_j^T \mathbf{K}\boldsymbol{\beta}_j\right\|}}{\sum_{i=1}^{N} \dfrac{p_{ij}^2}{\left\|\mathbf{K}_{ii} - 2\boldsymbol{\beta}_j^T \mathbf{K}_i + \boldsymbol{\beta}_j^T \mathbf{K}\boldsymbol{\beta}_j\right\|}}.$$

1.1.1 Kpd Algorithm

Kpd Algorithm can be developed by determining an update rule for β_{ij}'s and p_{ij}'s. Since the centers are not known explicitly, they are updated implicitly in the algorithm. β_{ij} in (17) can be calculated using previous values of β_{ij}'s. Thus, (17) can be written as

$$\beta_{ij}^{(r)} = \frac{K(\mathbf{x}_i, \mathbf{x}_i) - \sum_{k=1, k\neq i}^{N} \beta_{kj}^{(r-1)} K(\mathbf{x}_i, \mathbf{x}_k)}{K(\mathbf{x}_i, \mathbf{x}_i)}, \tag{20}$$

where r is the current iteration.

Using the updated β_{ij}'s and substituting them in (14), p_{ij}'s can be updated as

$$p_{ij}^{(r)} = \frac{\dfrac{1}{\left\|\phi(\mathbf{x}_i) - \sum_{k=1}^{N} \beta_{kj}^{(r)} \phi(\mathbf{x}_k)\right\|}}{\sum_{t=1}^{T} \dfrac{1}{\left\|\phi(\mathbf{x}_i) - \sum_{k=1}^{N} \beta_{kt}^{(r)} \phi(\mathbf{x}_k)\right\|}}. \tag{21}$$

The pseudocode of **Kpd Algorithm** is given in Algorithm 1 below.

Algorithm 1: KPd Algorithm

Input : data set $\mathbf{X}$, coefficient matrix $\mathbf{B}$, number of clusters T, kernel function K, stopping criterion ϵ

Output: probability matrix $\mathbf{P}$

Initialize $\mathbf{P}$ to a random probability matrix and $\mathbf{B}$ to a random coefficient matrix.

Calculate $K(\mathbf{x}_i, \mathbf{x}_l)$ for $i, l \in 1, ..., N$. *Set* $r = 0$

while $p_{ij}^{(r)} - p_{ij}^{(r-1)} > \epsilon$ **do**

  $r = r + 1$

  Update $\beta_{ij}^{(r)}$ as in (20)

  Update $p_{ij}^{(r)}$ as in (21)

end

1.2 Kernel Pd-clustering in Kernel Space with Inverse Mapping

The algorithm in the previous Section 1.1 defines the centers in the *kernel space* because the center equation cannot be calculated due to the ϕ function. However, center updates are made implicitly since the formula of the centers cannot be found using the center equation. The centers are required only when the distance between the data points and the centers are calculated. Therefore, we embedded the center equation in the distance function and using kernel trick we obtained the corresponding distances. In fact, the center updates are made in the *kernel space* because β_{ij}'s are the coefficients of the kernel centers. However, the centers can be defined in *kernel space* but updated in the *input space* so that they can be obtained explicitly.

The algorithm that will be developed in this section defines the centers in the *kernel space*. Different than the algorithm in the previous Section 1.1, the new algorithm updates the centers in the *input space*. Let $\mathbf{c}_j$'s be the centers in the *kernel space* and $\tilde{\mathbf{c}}_j$ be the image of $\mathbf{c}_j$ in *input space*. To update the centers in the *input space*, one should find the inverse mapping of them, which is $\phi^{-1}(\mathbf{c}_j)$. If ϕ is known then the inverse of $\mathbf{c}_j$ would be $\tilde{\mathbf{c}}_j$ since $\phi^{-1}(\mathbf{c}_j) = \tilde{\mathbf{c}}_j$. However, since ϕ function is not known, inverse of it cannot be calculated. Therefore, $\tilde{\mathbf{c}}_j$ can be chosen in such a way that its image, $\phi(\tilde{\mathbf{c}}_j)$, will be approximating $\mathbf{c}_j$. This algorithm will be called Kernel Pd-clustering with Inverse Mapping (**Kpd_Inv Algorithm**) and the idea of the algorithm is represented in Figure 2.

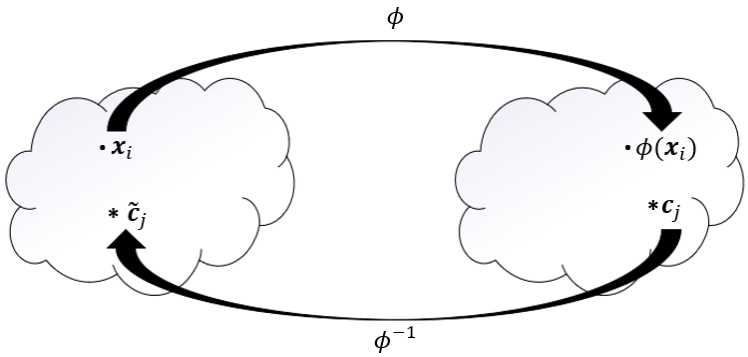

Figure 2: Kernel Pd-clustering with Inverse Mapping in Input and Kernel Space

Here, the optimization problem is the same as in Section 1.1, where the objective function

is (1) and constraints are (2) and (3). When $\mathbf{c}_j$'s are given, p_{ij}'s are found as in (7). Similarly, $\mathbf{c}_j$'s are obtained as in (8) if p_{ij}'s are given.

As mentioned above, center updates are made in the *input space*. Inverse mapping is not applicable to get the corresponding centers in the *input space*. Therefore, we will approximate $\mathbf{c}_j$ with the image of $\tilde{\mathbf{c}}_j$ on the *kernel space*, which is $\phi(\tilde{\mathbf{c}}_j)$.

To obtain the approximate centers in the *input space*, the following objective function can be considered

$$\min_{\tilde{\mathbf{c}}_j} \qquad V = \sum_{j=1}^{T} \|\phi(\tilde{\mathbf{c}}_j) - \mathbf{c}_j\|. \tag{22}$$

The objective aims to minimize the sum of distances between the *kernel space* centers and kernel images of the *input space* centers in the *input space*. $\phi(\tilde{\mathbf{c}}_j)$ refers to the mapping of $\tilde{\mathbf{c}}_j$ into the *kernel space*.

The function V in (22) can be written as

$$\sum_{j=1}^{T} \|\phi(\tilde{\mathbf{c}}_j) - \mathbf{c}_j\| = \sum_{j=1}^{T} \sqrt{K(\tilde{\mathbf{c}}_j, \tilde{\mathbf{c}}_j) - 2\,\phi(\tilde{\mathbf{c}}_j)\,\mathbf{c}_j + \mathbf{c}_j^T \mathbf{c}_j)}. \tag{23}$$

Then $\mathbf{c}_j$'s can be expressed as in (8), so we substitute them in (23). Thus, the multiplication of two ϕ functions will be obtained, and they can be written as a function of K using the kernel trick. That is, (23) becomes

$$= \sum_{j=1}^{T} \sqrt{K(\tilde{\mathbf{c}}_j, \tilde{\mathbf{c}}_j) - 2\frac{\sum_{i=1}^{N} \frac{p_{ij}^2\, K(\mathbf{x}_i, \tilde{\mathbf{c}}_j)}{\|\phi(\mathbf{x}_i) - \mathbf{c}_j\|}}{\sum_{i=1}^{N} \frac{p_{ij}^2}{\|\phi(\mathbf{x}_i) - \mathbf{c}_j\|}} + \sum_{l=1}^{N}\sum_{k=1}^{N} \frac{\frac{p_{lj}^2\, p_{kj}^2\, K(\mathbf{x}_l, \mathbf{x}_k)}{\|\phi(\mathbf{x}_l) - \mathbf{c}_j\| \|\phi(\mathbf{x}_k) - \mathbf{c}_j\|}}{\left(\sum_{i=1}^{N} \frac{p_{ij}^2}{\|\phi(\mathbf{x}_i) - \mathbf{c}_j\|}\right)\left(\sum_{i=1}^{N} \frac{p_{ij}^2}{\|\phi(\mathbf{x}_i) - \mathbf{c}_j\|}\right)}}. \tag{24}$$

To find the optimum $\tilde{\mathbf{c}}_j$, one can take the derivative of (23) with respect to $\tilde{c}_j$ and make it equal to 0. However, to take the derivative, prior information about the kernel function must be known. Below, Gaussian and Polynomial kernel functions are considered to calculate the centers.

- **Gaussian Kernel**

  When the kernel function is chosen as Gaussian kernel, (24) becomes

$$= \sum_{j=1}^{T} \sqrt{K(\tilde{\mathbf{c}}_j, \tilde{\mathbf{c}}_j) - 2\frac{\sum_{i=1}^{N} \frac{p_{ij}^2\, e^{-\|\mathbf{x}_i - \tilde{\mathbf{c}}_j\|^2/\sigma^2}}{\|\phi(\mathbf{x}_i) - \mathbf{c}_j\|}}{\sum_{i=1}^{N} \frac{p_{ij}^2}{\|\phi(\mathbf{x}_i) - \mathbf{c}_j\|}} + \sum_{l=1}^{N}\sum_{k=1}^{N} \frac{\frac{p_{lj}^2\, p_{kj}^2\, e^{-\|\mathbf{x}_l - \mathbf{x}_k\|^2/\sigma^2}}{\|\phi(\mathbf{x}_l) - \mathbf{c}_j\| \|\phi(\mathbf{x}_k) - \mathbf{c}_j\|}}{\left(\sum_{i=1}^{N} \frac{p_{ij}^2}{\|\phi(\mathbf{x}_i) - \mathbf{c}_j\|}\right)^2}}. \tag{25}$$

Since in Gaussian Kernel the distance between the object and itself is equal to 1, $K(\tilde{\mathbf{c}}_j, \tilde{\mathbf{c}}_j) = 1$ in Equation (25). Then taking the derivative of (25) with respect to $\tilde{\mathbf{c}}_j$ and equalizing it to 0 gives

$$\frac{\partial V}{\partial \tilde{\mathbf{c}}_j} = \frac{1}{2} \frac{-2}{\sum_{i=1}^{N} \frac{p_{ij}^2}{\|\phi(\mathbf{x}_i) - \mathbf{c}_j\|}} \sum_{i=1}^{N} \left(\frac{p_{ij}^2 \, e^{-\|\mathbf{x}_i - \tilde{\mathbf{c}}_j\|^2/\sigma^2}}{\|\phi(\mathbf{x}_i) - \mathbf{c}_j\|} \frac{2\,(\mathbf{x}_i - \tilde{\mathbf{c}}_j)}{\sigma^2} \right) \frac{1}{\|\phi(\tilde{\mathbf{c}}_j) - \mathbf{c}_j\|} = \mathbf{0}.$$

Then

$$\tilde{\mathbf{c}}_j = \frac{\sum_{i=1}^{N} \frac{p_{ij}^2 \, \mathbf{x}_i}{\|\phi(\mathbf{x}_i) - \mathbf{c}_j\|} e^{-\|\mathbf{x}_i - \tilde{\mathbf{c}}_j\|^2/\sigma^2}}{\sum_{i=1}^{N} \frac{p_{ij}^2}{\|\phi(\mathbf{x}_i) - \mathbf{c}_j\|} e^{-\|\mathbf{x}_i - \tilde{\mathbf{c}}_j\|^2/\sigma^2}} \tag{26}$$

for Gaussian kernel.

- **Polynomial Kernel**

  If the Polynomial kernel is considered, now (24) becomes

$$= \sum_{j=1}^{T} \sqrt{(\tilde{\mathbf{c}}_j^T \, \tilde{\mathbf{c}}_j + a)^b - 2 \frac{\sum_{i=1}^{N} \frac{p_{ij}^2 \, (\tilde{\mathbf{c}}_j^T \, \mathbf{x}_i + a)^b}{\|\phi(\mathbf{x}_i) - \mathbf{c}_j\|}}{\sum_{i=1}^{N} \frac{p_{ij}^2}{\|\phi(\mathbf{x}_i) - \mathbf{c}_j\|}} + \sum_{l=1}^{N} \sum_{k=1}^{N} \frac{\frac{p_{lj}^2 \, p_{kj}^2 \, (\mathbf{x}_l^T \, \mathbf{x}_k + a)^b}{\|\phi(\mathbf{x}_l) - \mathbf{c}_j\| \|\phi(\mathbf{x}_k) - \mathbf{c}_j\|}}{\left(\sum_{i=1}^{N} \frac{p_{ij}^2}{\|\phi(\mathbf{x}_i) - \mathbf{c}_j\|} \right) \left(\sum_{i=1}^{N} \frac{p_{ij}^2}{\|\phi(\mathbf{x}_i) - \mathbf{c}_j\|} \right)}}, \tag{27}$$

  where a and b are the predefined parameters of the Polynomial kernel.

  The derivative of (27) with respect to $\tilde{\mathbf{c}}_j$ is

$$\frac{\partial V}{\partial \tilde{\mathbf{c}}_j} = \frac{1}{2} \left(2\,b\,\tilde{\mathbf{c}}_j (\tilde{\mathbf{c}}_j^T \, \tilde{\mathbf{c}}_j + a)^{b-1} - 2 \frac{\sum_{i=1}^{N} \frac{p_{ij}^2 \, b \, (\tilde{\mathbf{c}}_j^T \, \mathbf{x}_i + a)^{b-1} \, \mathbf{x}_i}{\|\phi(\mathbf{x}_i) - \mathbf{c}_j\|}}{\sum_{i=1}^{N} \frac{p_{ij}^2}{\|\phi(\mathbf{x}_i) - \mathbf{c}_j\|}} \right) \frac{1}{\|\phi(\tilde{\mathbf{c}}_j) - \mathbf{c}_j\|} = \mathbf{0}.$$

  Then

$$\tilde{\mathbf{c}}_j = \frac{\sum_{i=1}^{N} \left(\frac{p_{ij}^2 \, (\tilde{\mathbf{c}}_j^T \, \mathbf{x}_i + a)^{b-1}}{\|\phi(\mathbf{x}_i) - \mathbf{c}_j\|} \right) \mathbf{x}_i}{(\tilde{\mathbf{c}}_j^T \, \tilde{\mathbf{c}}_j + a)^{b-1} \sum_{i=1}^{N} \frac{p_{ij}^2}{\|\phi(\mathbf{x}_i) - \mathbf{c}_j\|}} \tag{28}$$

  for Polynomial kernel.

1.2.1 Center Update

$\tilde{\mathbf{c}}_j$ values can be calculated using (26) and (28) but these equations contain $\mathbf{c}_j$'s in the denominator terms. Therefore, $\mathbf{c}_j$'s can be approximated with the image of $\tilde{\mathbf{c}}_j$'s from the previous

iteration, so we obtain

$$\tilde{\mathbf{c}}_j^{(r)} = \frac{\sum_{i=1}^{N} \frac{(p_{ij}^{(r-1)})^2 \, \mathbf{x}_i}{\sqrt{2 - 2\,K\left(\mathbf{x}_i, \tilde{\mathbf{c}}_j^{(r-1)}\right)}} e^{-\|\mathbf{x}_i - \tilde{\mathbf{c}}_j^{(r-1)}\|^2/\sigma^2}}{\sum_{i=1}^{N} \frac{(p_{ij}^{(r-1)})^2}{\sqrt{2 - 2\,K\left(\mathbf{x}_i, \tilde{\mathbf{c}}_j^{(r-1)}\right)}} e^{-\|\mathbf{x}_i - \tilde{\mathbf{c}}_j^{(r-1)}\|^2/\sigma^2}} \tag{29}$$

for Gaussian kernel. When function K is replaced with Gaussian, (29) becomes

$$\tilde{\mathbf{c}}_j^{(r)} = \frac{\sum_{i=1}^{N} \frac{(p_{ij}^{(r-1)})^2 \, \mathbf{x}_i}{\sqrt{2 - 2\,e^{-\|\mathbf{x}_i - \tilde{\mathbf{c}}_j^{(r-1)}\|^2/\sigma^2}}} e^{-\|\mathbf{x}_i - \tilde{\mathbf{c}}_j^{(r-1)}\|^2/\sigma^2}}{\sum_{i=1}^{N} \frac{(p_{ij}^{(r-1)})^2}{\sqrt{2 - 2\,e^{-\|\mathbf{x}_i - \tilde{\mathbf{c}}_j^{(r-1)}\|^2/\sigma^2}}} e^{-\|\mathbf{x}_i - \tilde{\mathbf{c}}_j^{(r-1)}\|^2/\sigma^2}}. \tag{30}$$

For Polynomial kernel, $\mathbf{c}_j$'s are approximated and (28) is obtained as

$$\tilde{\mathbf{c}}_j^{(r)} = \frac{\sum_{i=1}^{N} \frac{(p_{ij}^{(r-1)})^2 \, ((\tilde{\mathbf{c}}_j^{(r-1)})^T \, \mathbf{x}_i + a)^{b-1}}{\sqrt{K(\mathbf{x}_i, \mathbf{x}_i) - 2K(\mathbf{x}_i, \tilde{\mathbf{c}}_j^{(r-1)}) + K(\tilde{\mathbf{c}}_j^{(r-1)}, \tilde{\mathbf{c}}_j^{(r-1)})}} \mathbf{x}_i}{(\tilde{\mathbf{c}}_j^{(r-1)})^T \, \tilde{\mathbf{c}}_j^{(r-1)} + a)^{b-1} \sum_{i=1}^{N} \frac{(p_{ij}^{(r-1)})^2}{\sqrt{K(\mathbf{x}_i, \mathbf{x}_i) - 2K(\mathbf{x}_i, \tilde{\mathbf{c}}_j^{(r-1)}) + K(\tilde{\mathbf{c}}_j^{(r-1)}, \tilde{\mathbf{c}}_j^{(r-1)})}}}. \tag{31}$$

Writing Polynomial kernel function explicitly in (31) would give

$$\tilde{\mathbf{c}}_j^{(r)} = \frac{\sum_{i=1}^{N} \frac{(p_{ij}^{(r-1)})^2 \, b \, ((\tilde{\mathbf{c}}_j^{(r-1)})^T \, \mathbf{x}_i + a)^{b-1}}{\sqrt{(\mathbf{x}_i^T \, \mathbf{x}_i + a)^b - 2\left(\mathbf{x}_i^T \phi(\tilde{\mathbf{c}}_j^{(r-1)}) + a\right)^b + \left(\phi(\tilde{\mathbf{c}}_j^{(r-1)})^T \, \phi(\tilde{\mathbf{c}}_j^{(r-1)}) + a\right)^b}} \mathbf{x}_i}{(\tilde{\mathbf{c}}_j^{(r-1)})^T \, \tilde{\mathbf{c}}_j^{(r-1)} + a)^{b-1} \sum_{i=1}^{N} \frac{(p_{ij}^{(r-1)})^2}{\sqrt{(\mathbf{x}_i^T \, \mathbf{x}_i + a)^b - 2\left(\mathbf{x}_i^T \phi(\tilde{\mathbf{c}}_j^{(r-1)}) + a\right)^b + \left(\phi(\tilde{\mathbf{c}}_j^{(r-1)})^T \, \phi(\tilde{\mathbf{c}}_j^{(r-1)}) + a\right)^b}}}. \tag{32}$$

The centers in any iteration will be updated as (30) for Gaussian kernel and (32) for Polynomial kernel.

1.2.2 Probability Update

After $\tilde{\mathbf{c}}_j$'s are updated, their images in the *kernel space*, $\phi(\tilde{\mathbf{c}}_j)$, will be used to approximate p_{ij}'s and (7) becomes

$$p_{ij}^{(r)} = \frac{\frac{1}{\|\phi\left(\mathbf{x}_i\right) - \phi(\tilde{\mathbf{c}}_j^{(r)})\|}}{\sum_{t=1}^{T} \frac{1}{\|\phi\left(\mathbf{x}_i\right) - \phi(\tilde{\mathbf{c}}_t^{(r)})\|}}. \tag{33}$$

The distances in the denominators of (33) can be written as a function of K, and probability update becomes

$$p_{ij}^{(r)} = \frac{\dfrac{1}{\sqrt{K(\mathbf{x}_i,\mathbf{x}_i) - 2K(\mathbf{x}_i,\tilde{\mathbf{c}}_j^{(r)}) + K(\tilde{\mathbf{c}}_j^{(r)},\tilde{\mathbf{c}}_j^{(r)})}}}{\sum_{t=1}^{T} \dfrac{1}{\sqrt{K(\mathbf{x}_i,\mathbf{x}_i) - 2K(\mathbf{x}_i,\tilde{\mathbf{c}}_j^{(r)}) + K(\tilde{\mathbf{c}}_j^{(r)},\tilde{\mathbf{c}}_j^{(r)})}}}. \tag{34}$$

1.2.3 Kpd_Inv Algorithm

When we implement the ideas above, $\tilde{\mathbf{c}}_j$'s are updated first. Contrary to **KPd Algorithm**, $\tilde{\mathbf{c}}_j$'s are explicitly known since they are defined in the *input space*. $\mathbf{c}_j$'s will be replaced with $\phi(\tilde{\mathbf{c}}_j)$'s, and the distance between the data points and the centers in the *kernel space* will be calculated using $\phi(\tilde{\mathbf{c}}_j)$.

Algorithm 2 provides the pseudocode of **KPd_Inv Algorithm**.

Algorithm 2: Kpd_Inv Algorithm

Input : data set $\mathbf{X}$, number of clusters T, kernel function K, stopping criterion ϵ
Output: probability matrix $\mathbf{P}$

Initialize $\mathbf{P}$ as a random probability matrix.
Calculate $K(\mathbf{x}_i,\mathbf{x}_l)$ for $i,l \in 1,...,N$.
Set $r = 0$
while $p_{ij}^{(r)} - p_{ij}^{(r-1)} > \epsilon$ **do**
    $r = r + 1$
    Update $\tilde{\mathbf{c}}_j^{(r)}$ as in (30) if kernel function is Gaussian, and update as in (32) if kernel is chosen as polynomial
    Calculate $K(\mathbf{x}_i,\tilde{\mathbf{c}}_j^{(r)})$ for $i \in 1,...,N, j \in 1,...,T$
    Update $p_{ij}^{(r)}$ as in (34)
end

1.3 Kernel Pd-clustering in Feature (Input) Space

Kernel Pd-clustering in Feature Space Algorithm (**KPd_F Algorithm**) defines and updates the centers in the *input space*. Therefore, $\mathbf{c}_j$ represents the centers in the *input space* and $\tilde{\mathbf{c}}_j$ is that of *kernel space*. To calculate the distance between each data point and their corresponding clusters in *kernel space*, both data points and the cluster centers are required to be mapped into *kernel space* beforehand as shown in Figure 3.

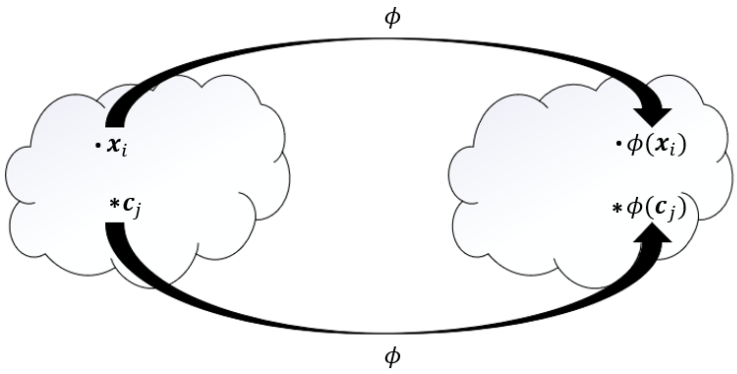

Figure 3: Kernel Pd-clustering in Feature Space in Input and Kernel Space

Then the optimization problem becomes

$$\begin{aligned} \min \quad & \sum_{i=1}^{N}\sum_{j=1}^{T} p_{ij}^2 \left\|\phi\left(\mathbf{x}_i\right) - \phi\left(\mathbf{c}_j\right)\right\| \\ \text{s.t.} \quad & \sum_{j=1}^{T} p_{ij} = 1 \qquad \forall i \\ & p_{ij} \geq 0\,. \qquad \forall i, j \end{aligned}$$

where $\mathbf{c}_j$ represents cluster centers in feature space. Let $\mathbf{\Phi_c}$ be a $T \times 1$ vector containing the images of *input space* centers. That is, $\mathbf{\Phi_c} = [\phi(\mathbf{c}_1)...\phi(\mathbf{c}_T)]^T$. Then the Lagrangian becomes

$$L(\mathbf{P}, \mathbf{\Phi_c}, \mathbf{\Lambda}) = \sum_{i=1}^{N}\sum_{j=1}^{T} p_{ij}^2 \left\|\phi\left(\mathbf{x}_i\right) - \phi\left(\mathbf{c}_j\right)\right\| - \sum_{i=1}^{N} \lambda_i \left(\sum_{j=1}^{T} p_{ij} - 1\right). \tag{35}$$

Taking the derivative with respect to p_{ij} we obtain

$$p_{ij} = \frac{\dfrac{1}{\left\|\phi\left(\mathbf{x}_i\right) - \phi\left(\mathbf{c}_t\right)\right\|}}{\displaystyle\sum_{t=1}^{T} \frac{1}{\left\|\phi(\mathbf{x}_i) - \phi\left(\mathbf{c}_t\right)\right\|}}. \tag{36}$$

Given p_{ij}'s, if we take the derivative of (35) with respect to $\phi(\mathbf{c}_j)$, we obtain

$$\phi(\mathbf{c}_j) = \frac{\displaystyle\sum_{i=1}^{N} \frac{p_{ij}^2\, \phi(\mathbf{x}_i)}{\left\|\phi(\mathbf{x}_i) - \mathbf{c}_j\right\|}}{\displaystyle\sum_{i=1}^{N} \frac{p_{ij}^2}{\left\|\phi(\mathbf{x}_i) - \mathbf{c}_j\right\|}}. \tag{37}$$

In (37), $\phi(\mathbf{c}_j)$ depends on $\phi(\mathbf{x}_i)$'s, which cannot be derived. Therefore, the derivative of (35) with respect to $\mathbf{c}_j$ can calculated only when the Kernel function is known, since $\mathbf{c}_j$'s are given as a function of ϕ. We consider two Kernel functions which are Gaussian and Polynomial Kernels.

- **Gaussian Kernel**

  We observe $\|\phi(\mathbf{x}_i) - \phi(\mathbf{c}_j)\|$ can be written as

$$\sqrt{K(\mathbf{x}_i, \mathbf{x}_i) - 2K(\mathbf{x}_i, \mathbf{c}_j) + K(\mathbf{c}_j, \mathbf{c}_j)} = \sqrt{2 - 2K(\mathbf{x}_i, \mathbf{c}_j)}. \tag{38}$$

  for Gaussian Kernel. If we substitute the distance term in (35) with (38), then L becomes the function of p_{ij}'s and $\mathbf{c}_j$'s. To take the gradient of L, we need to know kernel function. In the case of Gaussian kernel, Lagrangian becomes

$$L(\mathbf{P}, \mathbf{C}, \mathbf{\Lambda}) = \sum_{i=1}^{N}\sum_{j=1}^{T} p_{ij}^2 \sqrt{2 - 2K(\mathbf{x}_i, \mathbf{c}_j)} - \sum_{i=1}^{N} \lambda_i \left(\sum_{j=1}^{T} p_{ij} - 1 \right). \tag{39}$$

  Then for the given p_{ij}'s the derivative of (39) with respect to $\mathbf{c}_j$ is

$$\frac{\partial L}{\partial \mathbf{c}_j} = \frac{1}{2} \sum_{i=1}^{N} p_{ij}^2 \frac{1}{\sqrt{2 - 2K(\mathbf{x}_i, \mathbf{c}_j)}} (-2)\, e^{-\|\mathbf{x}_i - \mathbf{c}_j\|^2/\sigma^2} \frac{2(\mathbf{x}_i - \mathbf{c}_j)}{\sigma^2} = 0$$

$$\implies \sum_{i=1}^{N} p_{ij}^2 \frac{e^{-\|\mathbf{x}_i - \mathbf{c}_j\|^2/\sigma^2}}{\sqrt{2 - 2K(\mathbf{x}_i, \mathbf{c}_j)}} (\mathbf{x}_i - \mathbf{c}_j) = 0.$$

  Therefore, we obtain $\mathbf{c}_j$ as

$$\mathbf{c}_j = \frac{\sum_{i=1}^{N} \left(p_{ij}^2 \frac{e^{-\|\mathbf{x}_i - \mathbf{c}_j\|^2/\sigma^2}}{\sqrt{2 - 2K(\mathbf{x}_i, \mathbf{c}_j)}} \right) \mathbf{x}_i}{\sum_{i=1}^{N} p_{ij}^2 \frac{e^{-\|\mathbf{x}_i - \mathbf{c}_j\|^2/\sigma^2}}{\sqrt{2 - 2K(\mathbf{x}_i, \mathbf{c}_j)}}}. \tag{40}$$

- **Polynomial Kernel**

  When the Kernel function is chosen as Polynomial, $\|\phi(x_i) - \phi(c_j)\|$ is calculated as

$$\|\phi(\mathbf{x}_i) - \phi(\mathbf{c}_j)\| = \sqrt{(\mathbf{x}_i^T \mathbf{x}_i + a)^b - 2(\mathbf{c}_j^T \mathbf{x}_i + a)^b + (\mathbf{c}_j^T \mathbf{c}_j + a)^b}.$$

  Lagrangian becomes

$$L(\mathbf{P}, \mathbf{C}, \mathbf{\Lambda}) = \sum_{i=1}^{N}\sum_{j=1}^{T} p_{ij}^2 \sqrt{(\mathbf{x}_i^T \mathbf{x}_i + a)^b - 2(\mathbf{c}_j^T \mathbf{x}_i + a)^b + (\mathbf{c}_j^T \mathbf{c}_j + a)^b} - \sum_{i=1}^{N} \lambda_i \left(\sum_{j=1}^{T} p_{ij} - 1 \right). \tag{41}$$

  Then for given p_{ij}'s, the derivative of (41) with respect to $\mathbf{c}_j$ becomes

$$\frac{\partial L}{\partial \mathbf{c}_j} = \frac{1}{2} \sum_{i=1}^{N} p_{ij}^2 \frac{1}{\|\phi(\mathbf{x}_i) - \phi(\mathbf{c}_j)\|} \left(-2\, b\, \mathbf{x}_i\, (\mathbf{c}_j^T\, \mathbf{x}_i + a)^{b-1} + 2\, b\, \mathbf{c}_j\, (\mathbf{c}_j^T\, \mathbf{c}_j + a)^{b-1} \right) = 0$$

$$\implies \sum_{i=1}^{N} \frac{p_{ij}^2}{\|\phi(\mathbf{x}_i) - \phi(\mathbf{c}_j)\|} \left(\mathbf{c}_j\, (\mathbf{c}_j^T\, \mathbf{c}_j + a)^{b-1} - \mathbf{x}_i\, (\mathbf{c}_j^T\, \mathbf{x}_i + a)^{b-1} \right) = 0\,,$$

  which gives

$$\mathbf{c}_j = \frac{\sum_{i=1}^{N} \frac{p_{ij}^2}{\|\phi(\mathbf{x}_i) - \phi(\mathbf{c}_j)\|} (\mathbf{c}_j^T\, \mathbf{x}_i + a)^{b-1}\, \mathbf{x}_i}{(\mathbf{c}_j^T\, \mathbf{c}_j + a)^{b-1} \sum_{i=1}^{N} \frac{p_{ij}^2}{\|\phi(\mathbf{x}_i) - \phi(\mathbf{c}_j)\|}}. \tag{42}$$

1.3.1 Center Update

Since the denominator terms in center equations for both Gaussian and Polynomial kernels have $\mathbf{c}_j$'s, previous center values should be used when they are updated. Thus, (40) is rearranged as

$$\mathbf{c}_j^{(r)} = \frac{\sum_{i=1}^{N} \left((p_{ij}^{(r-1)})^2 \frac{e^{-\|\mathbf{x}_i - \mathbf{c}_j^{(r-1)}\|^2/\sigma^2}}{\sqrt{2 - 2K(\mathbf{x}_i, \mathbf{c}_j^{(r-1)})}} \right) \mathbf{x}_i}{\sum_{i=1}^{N} (p_{ij}^{(r-1)})^2 \frac{e^{-\|\mathbf{x}_i - \mathbf{c}_j^{(r-1)}\|^2/\sigma^2}}{\sqrt{2 - 2K(\mathbf{x}_i, \mathbf{c}_j^{(r-1)})}}}. \tag{43}$$

If K function is replaced with Gaussian kernel, then (43) becomes

$$\mathbf{c}_j^{(r)} = \frac{\sum_{i=1}^{N} \left((p_{ij}^{(r-1)})^2 \frac{e^{-\|\mathbf{x}_i - \mathbf{c}_j^{(r-1)}\|^2/\sigma^2}}{\sqrt{2 - 2\,e^{-\|\mathbf{x}_i - \mathbf{c}_j^{(r-1)}\|^2/\sigma^2}}} \right) \mathbf{x}_i}{\sum_{i=1}^{N} (p_{ij}^{(r-1)})^2 \frac{e^{-\|\mathbf{x}_i - \mathbf{c}_j^{(r-1)}\|^2/\sigma^2}}{\sqrt{2 - 2\,e^{-\|\mathbf{x}_i - \mathbf{c}_j^{(r-1)}\|^2/\sigma^2}}}}. \tag{44}$$

In the case of Polynomial kernel, (42) can be rewritten as

$$\mathbf{c}_j^{(r)} = \frac{\sum_{i=1}^{N} \frac{(p_{ij}^{(r-1)})^2 \left((\tilde{\mathbf{c}}_j^{(r-1)})^T \mathbf{x}_i + a\right)^{b-1}}{\sqrt{K(\mathbf{x}_i, \mathbf{x}_i) - 2K(\mathbf{x}_i, \tilde{\mathbf{c}}_j^{(r-1)}) + K(\tilde{\mathbf{c}}_j^{(r-1)}, \tilde{\mathbf{c}}_j^{(r-1)})}} \mathbf{x}_i}{(\tilde{\mathbf{c}}_j^{(r-1)})^T \tilde{\mathbf{c}}_j^{(r-1)} + a)^{b-1} \sum_{i=1}^{N} \frac{(p_{ij}^{(r-1)})^2}{\sqrt{K(\mathbf{x}_i, \mathbf{x}_i) - 2K(\mathbf{x}_i, \tilde{\mathbf{c}}_j^{(r-1)}) + K(\tilde{\mathbf{c}}_j^{(r-1)}, \tilde{\mathbf{c}}_j^{(r-1)})}}},$$

which gives

$$\mathbf{c}_j^{(r)} = \frac{\sum_{i=1}^{N} \frac{(p_{ij}^{(r-1)})^2\, b\, \left((\tilde{\mathbf{c}}_j^{(r-1)})^T \mathbf{x}_i + a\right)^{b-1}}{\sqrt{\left(\mathbf{x}_i^T \mathbf{x}_i + a\right)^b - 2\left(\mathbf{x}_i^T \phi(\tilde{\mathbf{c}}_j^{(r-1)}) + a\right)^b + \left(\phi(\tilde{\mathbf{c}}_j^{(r-1)})^T \phi(\tilde{\mathbf{c}}_j^{(r-1)}) + a\right)^b}} \mathbf{x}_i}{(\tilde{\mathbf{c}}_j^{(r-1)})^T \tilde{\mathbf{c}}_j^{(r-1)} + a)^{b-1} \sum_{i=1}^{N} \frac{(p_{ij}^{(r-1)})^2}{\sqrt{\left(\mathbf{x}_i^T \mathbf{x}_i + a\right)^b - 2\left(\mathbf{x}_i^T \phi(\tilde{\mathbf{c}}_j^{(r-1)}) + a\right)^b + \left(\phi(\tilde{\mathbf{c}}_j^{(r-1)})^T \phi(\tilde{\mathbf{c}}_j^{(r-1)}) + a\right)^b}}}. \tag{45}$$

1.3.2 Probability Update

Upon completing the center updates in the *input space*, p_{ij}'s are updated using $\mathbf{c}_j^{(r)}$'s. Then (36) becomes

$$p_{ij}^{(r)} = \frac{\frac{1}{\left\|\phi(\mathbf{x}_i) - \phi\left(\mathbf{c}_t^{(r)}\right)\right\|}}{\sum_{t=1}^{T} \frac{1}{\left\|\phi(\mathbf{x}_i) - \phi\left(\mathbf{c}_t^{(r)}\right)\right\|}}. \tag{46}$$

Distance terms in the denominator terrms of the probability update is a function of K, and it refers to

$$p_{ij}^{(r)} = \frac{\frac{1}{\sqrt{K(\mathbf{x}_i,\mathbf{x}_i) - 2K(\mathbf{x}_i,\mathbf{c}_j^{(r)}) + K(\mathbf{c}_j^{(r)},\mathbf{c}_j^{(r)})}}}{\sum_{t=1}^{T} \frac{1}{\sqrt{K(\mathbf{x}_i,\mathbf{x}_i) - 2K(\mathbf{x}_i,\mathbf{c}_j^{(r)}) + K(\mathbf{c}_j^{(r)},\mathbf{c}_j^{(r)})}}}. \tag{47}$$

1.3.3 Kpd_F Algorithm

This algorithm defines and updates the centers in the *input space*. Since they are introduced in the *input space*, they are known explicitly. Afterwards, probabilities are updated accordingly. Pseudocode of **KPd_F Algorithm** is given in Algorithm 3.

Algorithm 3: Kpd_F Algorithm

Input : data set $\mathbf{X}$, number of clusters T, kernel function K, stopping criterion ϵ
Output: probability matrix $\mathbf{P}$

Initialize $\mathbf{P}$ as a random probability matrix.
Calculate $K(\mathbf{x}_i, \mathbf{x}_l)$ for $i, l \in 1, ..., N$.
Set $r = 0$
while $p_{ij}^{(r)} - p_{ij}^{(r-1)} > \epsilon$ **do**
    $r = r + 1$
    Update $\mathbf{c}_j^{(r)}$ as in (44) if kernel function is Gaussian, and update as in (45) if kernel is chosen as Polynomial
    Calculate $K(\mathbf{x}_i, \mathbf{c}_j^{(r)})$ for $i \in 1, ..., N, j \in 1, ..., T$
    Update $p_{ij}^{(r)}$ as in (47)
end

2 Kernel Mahalanobis Pd-clustering

We consider Euclidean norm in the algorithms explained in Section 1. In this section, we study the statistical distance (i.e., Mahalanobis distance) in the kernel algorithms, where the correlation between the data features is considered. First, we introduce Kernel Mahalanobis distance. Later, Kernel probabilistic distance clustering with Mahalanobis distance will be introduced.

2.1 Kernel Mahalanobis Distance

Consider we have an $n \times m$ data set $\mathbf{X}$, where n shows the number of data points and m is that of features in the *input space*. Therefore, each data point $\mathbf{x}_i$ is an m-dimensional column vector, $i = 1, ..., n$. Function ϕ represents the mapping from original space to Hilbert Space $\mathscr{H}$

(or *kernel space*), i.e. $\phi : X \to \mathscr{H}$. In this mapping, the dimension of the vectors is changed to s. Therefore, $\mathbf{x}_i$ becomes $s \times 1$ vector.

Let $\mathbf{\Phi}$ contains the mapping of each data point $\mathbf{x}_i$. Then $\mathbf{\Phi}$ is an $s \times n$ matrix shown as $\mathbf{\Phi} = [\phi(\mathbf{x}_1)...\phi(\mathbf{x}_n)]_{s\times n}$. The mean of $\phi(\mathbf{x}_i)$'s are calculated as

$$\boldsymbol{\phi}_\mu = \frac{1}{n}\sum_{i=1}^{n}\phi(\mathbf{x}_i) = \frac{1}{n}\mathbf{\Phi}\mathbf{1}_n\,, \tag{48}$$

where $\mathbf{1}_n$ is a column vector of 1's. If data point $\phi(\mathbf{x}_i)$ is centered with $\boldsymbol{\phi}_\mu$, it is denoted by

$$\tilde{\phi}(\mathbf{x}_i) = \phi(\mathbf{x}_i) - \boldsymbol{\phi}_\mu\,. \tag{49}$$

Using (48), we can write (49) as

$$\tilde{\phi}(\mathbf{x}_i) = \phi(\mathbf{x}_i) - \frac{1}{n}\mathbf{\Phi}\mathbf{1}_n\,. \tag{50}$$

Let $\tilde{\mathbf{\Phi}}$ represent the matrix of centered data points $\mathbf{x}_i$, $i = 1, ..., n$. Then

$$\tilde{\mathbf{\Phi}} = \left[\tilde{\phi}(\mathbf{x}_1)...\tilde{\phi}(\mathbf{x}_n)\right] = \mathbf{\Phi} - \boldsymbol{\phi}_\mu\mathbf{1}_n^T\,. \tag{51}$$

We can substitute (48) in (51) for $\boldsymbol{\phi}_\mu$and obtain

$$\tilde{\mathbf{\Phi}} = \mathbf{\Phi} - \frac{1}{n}\mathbf{\Phi}\mathbf{1}_n\mathbf{1}_n^T = \mathbf{\Phi}\left[\mathbf{I}_{n\times n} - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^T\right]\,. \tag{52}$$

Then (52) can be written as

$$\tilde{\mathbf{\Phi}} = \mathbf{\Phi}\mathbf{H}\,, \tag{53}$$

where $\mathbf{H}$ is an $n \times n$ centering matrix shown as

$$\mathbf{H} = \mathbf{I}_{n\times n} - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^T\,.$$

$\mathbf{H}$ matrix has some properties. The transpose and square of $\mathbf{H}$ is equal to itself. That is, $\mathbf{H} = \mathbf{H}^T = \mathbf{H}^2$.

The covariance operator in the Hilbert Space, shown as $\mathbf{C} : \mathscr{H} \to \mathscr{H}$, operates on $\phi(\mathbf{x}) \in \mathscr{H}$ as

$$\mathbf{C}\phi(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n}\left(\phi(\mathbf{x}_i) - \boldsymbol{\phi}_\mu\right)\langle\phi(\mathbf{x}_i) - \boldsymbol{\phi}_\mu, \phi(\mathbf{x}_i)\rangle. \tag{54}$$

We know that $\left(\phi(\mathbf{x}_i) - \boldsymbol{\phi}_\mu\right)$ is equal to $\tilde{\phi}(\mathbf{x}_i)$, so (54) can be written as

$$\mathbf{C}\phi(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n}\tilde{\phi}(\mathbf{x}_i)\tilde{\phi}(\mathbf{x}_i)^T\phi(\mathbf{x}) = \frac{1}{n}\tilde{\mathbf{\Phi}}\tilde{\mathbf{\Phi}}^T\phi(\mathbf{x})\,. \tag{55}$$

Using (53), $\mathbf{C}$ becomes

$$\mathbf{C} = \frac{1}{n}\tilde{\mathbf{\Phi}}\tilde{\mathbf{\Phi}}^T = \frac{1}{n}\mathbf{\Phi}\mathbf{H}\mathbf{H}^T\mathbf{\Phi}^T = \frac{1}{n}\mathbf{\Phi}\mathbf{H}\mathbf{H}\mathbf{\Phi}^T = \frac{1}{n}\mathbf{\Phi}\mathbf{H}\mathbf{\Phi}^T\,.$$

Remember that $\tilde{\mathbf{\Phi}}$ is the matrix of centered data points in *kernel space*. Then using (53), centered Kernel matrix (i.e., the matrix containing the inner products of all centered kernels of data points) is

$$\widetilde{\mathbf{K}} = \tilde{\mathbf{\Phi}}^T \tilde{\mathbf{\Phi}} = \mathbf{H}\mathbf{\Phi}^T\mathbf{\Phi}\mathbf{H} = \mathbf{H}\mathbf{K}\mathbf{H}\,,$$

where

$$K = \mathbf{\Phi}^T\mathbf{\Phi}\,. \tag{56}$$

For instance, the entry in i^{th} row and l^{th} column in $\widetilde{\mathbf{K}}$ gives the inner product of centered kernels of data points $\mathbf{x}_i$ and $\mathbf{x}_l$.

Let $\bar{\mathbf{k}}_x$ be a column vector whose i^{th} element represents the inner product of $\mathbf{x}$ and $\mathbf{x}_i$ in the *kernel space*. That is,

$$\bar{\mathbf{k}}_{\mathbf{x}} = [k(\mathbf{x}_1, \mathbf{x}), ..., k(\mathbf{x}_n, \mathbf{x})]^T = \mathbf{\Phi}^T\phi(\mathbf{x})\,. \tag{57}$$

Then the inner product of $\tilde{\boldsymbol{\phi}}(\mathbf{x})$ with other centered kernel data points $\tilde{\boldsymbol{\phi}}(\mathbf{x}_i)$'s is

$$\tilde{\mathbf{k}}_{\mathbf{x}} = \tilde{\mathbf{\Phi}}^T\tilde{\phi}(\mathbf{x})\,. \tag{58}$$

When (53) is substituted into (58), we obtain

$$\tilde{\mathbf{k}}_{\mathbf{x}} = (\mathbf{\Phi}\mathbf{H})^T\tilde{\phi}(\mathbf{x}) = \mathbf{H}^T\mathbf{\Phi}^T\tilde{\phi}(\mathbf{x}) = \mathbf{H}\left(\mathbf{\Phi}^T\tilde{\phi}(\mathbf{x})\right)\,. \tag{59}$$

Using $\boldsymbol{\phi}(\mathbf{x})$ in (49), (59) becomes

$$\tilde{\mathbf{k}}_{\mathbf{x}} = \mathbf{H}\left(\mathbf{\Phi}^T\left(\boldsymbol{\phi}(\mathbf{x}_i) - \boldsymbol{\phi}_\mu\right)\right) = \mathbf{H}\left(\mathbf{\Phi}^T\boldsymbol{\phi}(\mathbf{x}_i) - \mathbf{\Phi}^T\boldsymbol{\phi}_\mu\right)\,. \tag{60}$$

We know from (57) that $\mathbf{\Phi}^T\boldsymbol{\phi}(\mathbf{x}_i)$ in (60) is equal to $\bar{\mathbf{k}}_{\mathbf{x}}$. Moreover, by substituting (48) for $\boldsymbol{\phi}_\mu$ into (60) we get

$$\tilde{\mathbf{k}}_{\mathbf{x}} = \mathbf{H}\left(\bar{\mathbf{k}}_{\mathbf{x}} - \frac{1}{n}\mathbf{\Phi}^T\mathbf{\Phi}\mathbf{1}_n\right)\,. \tag{61}$$

Using (56) for $\mathbf{\Phi}^T\mathbf{\Phi}$, (61) becomes

$$\tilde{\mathbf{k}}_{\mathbf{x}} = \mathbf{H}\left(\bar{\mathbf{k}}_{\mathbf{x}} - \frac{1}{n}\mathbf{K}\mathbf{1}_n\right)\,. \tag{62}$$

The inner product of kernel of $\mathbf{x}$ with itself is shown as $k(\mathbf{x}, \mathbf{x})$ or $k_{\mathbf{xx}}$. That is,

$$k_{\mathbf{xx}} = \boldsymbol{\phi}(\mathbf{x})^T\boldsymbol{\phi}(\mathbf{x})\,. \tag{63}$$

When $k_{\mathbf{xx}}$ is for the centered data points, we obtain

$$\tilde{k}_{\mathbf{xx}} = \tilde{\phi}(\mathbf{x})^T\tilde{\phi}(\mathbf{x})\,. \tag{64}$$

Using (50) for $\tilde{\phi}(\mathbf{x})$ in (64) we get

$$\tilde{k}_{\mathbf{xx}} = \left(\phi(\mathbf{x}) - \frac{1}{n}\mathbf{\Phi}\mathbf{1}_n\right)^T\left(\phi(\mathbf{x}) - \frac{1}{n}\mathbf{\Phi}\mathbf{1}_n\right) = \phi(\mathbf{x})^T\phi(\mathbf{x}) - \frac{2}{n}\mathbf{1}_n^T\mathbf{\Phi}^T\phi(\mathbf{x}) + \frac{1}{n^2}\left(\mathbf{\Phi}\mathbf{1}_n\right)^T\left(\mathbf{\Phi}\mathbf{1}_n\right)\,. \tag{65}$$

Substituting (63) for $\boldsymbol{\phi}(\mathbf{x})^T\boldsymbol{\phi}(\mathbf{x})$, (57) for $\mathbf{\Phi}^T\phi(\mathbf{x})$, and (56) for $\mathbf{\Phi}^T\mathbf{\Phi}$ gives

$$\tilde{k}_{\mathbf{xx}} = k_{\mathbf{xx}} - \frac{2}{n}\mathbf{1}_n^T\bar{\mathbf{k}}_{\mathbf{x}} + \frac{1}{n^2}\mathbf{1}_n^T\mathbf{K}\mathbf{1}_n\,. \tag{66}$$

2.2 Kernel Mahalanobis Distance for Invertible Covariance

The kernelized Mahalanobis distance is

$$d_{IC}^2(\mathbf{x}) = d_{IC}^2(\phi(\mathbf{x}); \{\boldsymbol{\phi}_\mu, \mathbf{C}\}) = (\phi(\mathbf{x}) - \boldsymbol{\phi}_\mu)^T \mathbf{C}^{-1} (\phi(\mathbf{x}) - \boldsymbol{\phi}_\mu) \,. \tag{67}$$

Therefore, the covariance matrix must be invertible. It restricts the dimension of $\mathscr{H}$ to a finite dimension, which is s and $s < n$. $\tilde{\mathbf{\Phi}}$ has a singular value decomposition, which is

$$\tilde{\mathbf{\Phi}} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T. \tag{68}$$

Note that $\mathbf{U} \in \mathbb{R}^{s\times s}$, $\mathbf{V} \in \mathbb{R}^{n\times n}$, and $\mathbf{\Sigma} \in \mathbb{R}^{s\times n}$, where $\mathbf{U}$ and $\mathbf{V}$ contain the eigenvectors and the eigenvalues are in the diagonals of $\mathbf{\Sigma}$ matrix. Then the covariance matrix can be written as

$$\mathbf{C} = \frac{1}{n}\tilde{\mathbf{\Phi}}\tilde{\mathbf{\Phi}}^T \,. \tag{69}$$

Using (68), the covariance in (69) can be rewritten as

$$\mathbf{C} = \frac{1}{n}\mathbf{U}\mathbf{\Sigma}\mathbf{V}^T \left(\mathbf{U}\mathbf{\Sigma}\mathbf{V}^T\right)^T = \frac{1}{n}\mathbf{U}\mathbf{\Sigma}\mathbf{V}^T\mathbf{V}\mathbf{\Sigma}^T\mathbf{U}^T \,.$$

By using the orthogonality of $\mathbf{U}$ and $\mathbf{V}$ matrices, we know that $\mathbf{U}^T\mathbf{U} = \mathbf{I}$ and $\mathbf{V}^T\mathbf{V} = \mathbf{I}$. Then we obtain

$$\mathbf{C} = \frac{1}{n}\mathbf{U}\mathbf{\Sigma}\mathbf{\Sigma}^T\mathbf{U}^T$$

and

$$\frac{1}{n}\mathbf{C}^{-1} = \mathbf{U}\left(\mathbf{\Sigma}\mathbf{\Sigma}^T\right)^{-1}\mathbf{U}^T \,. \tag{70}$$

If both sides of (70) are multiplied with $\tilde{\mathbf{\Phi}}$, and (68) is substituted for $\tilde{\mathbf{\Phi}}$ on the right-hand side of the equation, we obtain

$$\frac{1}{n}\mathbf{C}^{-1}\tilde{\mathbf{\Phi}} = \mathbf{U}\left(\mathbf{\Sigma}\mathbf{\Sigma}^T\right)^{-1}\mathbf{\Sigma}\mathbf{V}^T \,. \tag{71}$$

Note that $\widetilde{\mathbf{K}} = \tilde{\mathbf{\Phi}}^T\tilde{\mathbf{\Phi}}$. Following the SVD, it is equal to

$$\widetilde{\mathbf{K}} = \mathbf{V}\mathbf{\Sigma}^T\mathbf{U}^T\mathbf{U}\mathbf{\Sigma}\mathbf{V}^T = \mathbf{V}\mathbf{\Sigma}^T\mathbf{\Sigma}\mathbf{V}^T \,.$$

If $\tilde{\mathbf{\Phi}}$ is invertible, then $\mathbf{\Sigma}^{-1}$ exists. If $\tilde{\mathbf{\Phi}}$ is not invertible, we can find the pseudoinverse of $\mathbf{\Sigma}$, denoted by $\mathbf{\Sigma}^\dagger$. In this case, we assume that $\tilde{\mathbf{\Phi}}$ is not invertible, therefore $\mathbf{\Sigma}$ is singular and pseudoinverse of it should be calculated. $\mathbf{\Sigma}^\dagger$ is found by taking the reciprocal of the diagomal elements, i.e., eigenvalues, and then taking the transpose of the matrix. Since $\mathbf{\Sigma}$ is singular, $\mathbf{\Sigma}^T\mathbf{\Sigma}$ is also non-invertible. Therefore, we find the pseudo-inverse of $\widetilde{\mathbf{K}}$, shown as $\widetilde{\mathbf{K}}^\dagger$, as

$$\widetilde{\mathbf{K}}^\dagger = \mathbf{V}\left(\mathbf{\Sigma}^T\mathbf{\Sigma}\right)^\dagger\mathbf{V}^T \,.$$

By multiplying both sides with $\tilde{\mathbf{\Phi}}$ from the left, we get

$$\tilde{\mathbf{\Phi}}\widetilde{\mathbf{K}}^\dagger = \tilde{\mathbf{\Phi}}\mathbf{V}\left(\mathbf{\Sigma}^T\mathbf{\Sigma}\right)^\dagger\mathbf{V}^T \,. \tag{72}$$

When we substitute (68) into $\tilde{\mathbf{\Phi}}$ on the right-hand side of (72), we obtain

$$\tilde{\mathbf{\Phi}}\widetilde{\mathbf{K}}^{\dagger} = \mathbf{U}\mathbf{\Sigma}\left(\mathbf{\Sigma}^T\mathbf{\Sigma}\right)^{\dagger}\mathbf{V}^T . \tag{73}$$

Note that (71) and (73) are equal to each other. Therefore, we obtain

$$\tilde{\mathbf{\Phi}}\widetilde{\mathbf{K}}^{\dagger} = \frac{1}{n}\mathbf{C}^{-1}\tilde{\mathbf{\Phi}} . \tag{74}$$

We know that $\mathbf{C} = \frac{1}{n}\tilde{\mathbf{\Phi}}\tilde{\mathbf{\Phi}}^T$ from (69). Therefore, using (50) $\mathbf{C}\,\tilde{\phi}(\mathbf{x})$ can be written as

$$\mathbf{C}\,\tilde{\phi}(\mathbf{x}) = \frac{1}{n}\tilde{\mathbf{\Phi}}\tilde{\mathbf{\Phi}}^T\left(\phi(\mathbf{x}) - \frac{1}{n}\mathbf{\Phi}\mathbf{1}_n\right). \tag{75}$$

By substituting the transpose of (53) into $\tilde{\mathbf{\Phi}}^T$ in (75), we get

$$\mathbf{C}\,\tilde{\phi}(\mathbf{x}) = \frac{1}{n}\tilde{\mathbf{\Phi}}\mathbf{H}\mathbf{\Phi}^T\left(\phi(\mathbf{x}) - \frac{1}{n}\mathbf{\Phi}\mathbf{1}_n\right) = \frac{1}{n}\tilde{\mathbf{\Phi}}\mathbf{H}\left(\mathbf{\Phi}^T\phi(\mathbf{x}) - \frac{1}{n}\mathbf{\Phi}^T\mathbf{\Phi}\mathbf{1}_n\right). \tag{76}$$

Note that $\mathbf{\Phi}^T\phi(\mathbf{x})$ is equal to $\bar{\mathbf{k}}_x$ and $\mathbf{\Phi}^T\mathbf{\Phi}$ is $\mathbf{K}$. Then (76) will be

$$\mathbf{C}\,\tilde{\phi}(\mathbf{x}) = \frac{1}{n}\tilde{\mathbf{\Phi}}\mathbf{H}\left(\bar{\mathbf{k}}_{\mathbf{x}} - \frac{1}{n}\mathbf{K}\mathbf{1}_n\right), \tag{77}$$

and using (62), it leads to

$$\mathbf{C}\,\tilde{\phi}(\mathbf{x}) = \frac{1}{n}\tilde{\mathbf{\Phi}}\tilde{\mathbf{k}}_{\mathbf{x}} . \tag{78}$$

We know that C is invertible. Using (78), we obtain

$$\tilde{\phi}(\mathbf{x}) = \frac{1}{n}\mathbf{C}^{-1}\tilde{\mathbf{\Phi}}\tilde{\mathbf{k}}_{\mathbf{x}}. \tag{79}$$

Substitute (74) into (79) and get

$$\tilde{\phi}(\mathbf{x}) = \tilde{\mathbf{\Phi}}\,\widetilde{\mathbf{K}}^{\dagger}\,\tilde{\mathbf{k}}_{\mathbf{x}} . \tag{80}$$

Therefore, kernelized Mahalanobis distance for invertible covariance becomes

$$d^2_{IC}(\mathbf{x}) = d^2_{IC}(\phi(\mathbf{x}); \{\boldsymbol{\phi}_\mu, \mathbf{C}\}) = \tilde{\phi}(\mathbf{x})^T\mathbf{C}^{-1}\tilde{\phi}(\mathbf{x}) = \tilde{\phi}(\mathbf{x})^T\mathbf{C}^{-1}\tilde{\mathbf{\Phi}}\,\widetilde{\mathbf{K}}^{\dagger}\,\tilde{\mathbf{k}}_{\mathbf{x}}. \tag{81}$$

Note that when $\mathbf{C}^{-1}\tilde{\mathbf{\Phi}}$ is obtained from (74) and substituted into (81), the distance function becomes

$$d^2_{IC}(\mathbf{x}) = d^2_{IC}(\phi(\mathbf{x}); \{\boldsymbol{\phi}_\mu, \mathbf{C}\}) = n\,\tilde{\phi}(\mathbf{x})^T\tilde{\mathbf{\Phi}}\widetilde{\mathbf{K}}^{\dagger}\widetilde{\mathbf{K}}^{\dagger}\tilde{\mathbf{k}}_{\mathbf{x}}$$

$$= \tilde{\phi}(\mathbf{x})^T\mathbf{C}^{-1}\tilde{\phi}(\mathbf{x}) = n\,\tilde{\mathbf{k}}_{\mathbf{x}}^T\left(\widetilde{\mathbf{K}}^{\dagger}\right)^2\tilde{\mathbf{k}}_{\mathbf{x}} . \tag{82}$$

2.2.1 Mahalanobis Distance between Two Data Points

Please note that (82) refers to the distance between $\phi(\tilde{\mathbf{x}})$ and the mean of the data set. However, we can find the distance between two data points which are centralized in *kernel space*, say $\tilde{\phi}(\mathbf{x})$ and $\tilde{\phi}(\mathbf{y})$. Mahalanobis distance function can be written as

$$d^2_{IC}(\mathbf{x},\mathbf{y}) = d^2_{IC}(\phi(\mathbf{x}),\phi(\mathbf{y});\{\boldsymbol{\phi}_\mu,\mathbf{C}\}) = [\tilde{\phi}(\mathbf{x})-\tilde{\phi}(\mathbf{y})]^T\,\mathbf{C}^{-1}\,[\tilde{\phi}(\mathbf{x})-\tilde{\phi}(\mathbf{y})]\,. \tag{83}$$

Following (80) the difference between $\tilde{\phi}(\mathbf{x})$ and $\tilde{\phi}(\mathbf{y})$ is

$$\tilde{\phi}(\mathbf{x})-\tilde{\phi}(\mathbf{y}) = \tilde{\boldsymbol{\Phi}}\,\widetilde{\mathbf{K}}^{\dagger}[\tilde{\mathbf{k}}_{\mathbf{x}}-\tilde{\mathbf{k}}_{\mathbf{y}}]\,. \tag{84}$$

When (84) is substituted into (83), Mahalanobis distance becomes

$$d^2_{IC}(\mathbf{x},\mathbf{y}) = [\tilde{\phi}(\mathbf{x})-\tilde{\phi}(\mathbf{y})]^T\,\mathbf{C}^{-1}\,\tilde{\boldsymbol{\Phi}}\,\widetilde{\mathbf{K}}^{\dagger}[\tilde{\mathbf{k}}_{\mathbf{x}}-\tilde{\mathbf{k}}_{\mathbf{y}}]\,. \tag{85}$$

For $\mathbf{C}^{-1}\,\tilde{\boldsymbol{\Phi}}$ in (85), (74) is substituted and we get

$$d^2_{IC}(\mathbf{x},\mathbf{y}) = n\,[\tilde{\phi}(\mathbf{x})-\tilde{\phi}(\mathbf{y})]^T\,\tilde{\boldsymbol{\Phi}}\,(\widetilde{\mathbf{K}}^{\dagger})^2[\tilde{\mathbf{k}}_{\mathbf{x}}-\tilde{\mathbf{k}}_{\mathbf{y}}] \tag{86}$$

Following (58), $\tilde{\phi}(\mathbf{x})^T\,\tilde{\boldsymbol{\Phi}}$ and $\tilde{\phi}(\mathbf{y})^T\,\tilde{\boldsymbol{\Phi}}$ are equal to $\tilde{\mathbf{k}}_{\mathbf{x}}^T$ and $\tilde{\mathbf{k}}_{\mathbf{y}}^T$, respectively. Therefore, we obtain the Mahalanobis distance between the centralized data points $\mathbf{x}$ and $\mathbf{y}$ in the *kernel space* as

$$d^2_{IC}(\mathbf{x},\mathbf{y}) = d^2_{IC}(\phi(\tilde{\mathbf{x}}),\phi(\tilde{\mathbf{y}});\{\boldsymbol{\phi}_\mu,\mathbf{C}\}) = n\,[\tilde{\mathbf{k}}_{\mathbf{x}}-\tilde{\mathbf{k}}_{\mathbf{y}}]^T\,(\widetilde{\mathbf{K}}^{\dagger})^2[\tilde{\mathbf{k}}_{\mathbf{x}}-\tilde{\mathbf{k}}_{\mathbf{y}}] \tag{87}$$

2.3 Kernel Mahalanobis Distance for Regularized Covariance

When the dimension of $\mathscr{H}$ is higher than n or infinite, the covariance operatior is non-invertible. Therefore, it is regularized so that it will not be singular. Regularized covariance, denoted by $\mathbf{C}_{reg}$, becomes

$$\mathbf{C}_{reg} = \mathbf{C}+\sigma^2\mathbf{I}_{\mathscr{H}} = \frac{1}{n}\tilde{\boldsymbol{\Phi}}\tilde{\boldsymbol{\Phi}}^T+\sigma^2\mathbf{I}_{\mathscr{H}}\,, \tag{88}$$

where $\mathbf{I}_{\mathscr{H}}$ is an identity matrix with Hilbert space dimension and σ is a predefined parameter. When (88) is multiplied with $\tilde{\boldsymbol{\Phi}}$ from the right, we obtain

$$\mathbf{C}_{reg}\tilde{\boldsymbol{\Phi}} = \frac{1}{n}\tilde{\boldsymbol{\Phi}}\tilde{\boldsymbol{\Phi}}^T\tilde{\boldsymbol{\Phi}}+\sigma^2\mathbf{I}_{\mathscr{H}}\tilde{\boldsymbol{\Phi}}\,. \tag{89}$$

Note that $\tilde{\boldsymbol{\Phi}}^T\tilde{\boldsymbol{\Phi}}$ is equal to $\widetilde{\mathbf{K}}$. Therefore, (89) can be written as

$$\mathbf{C}_{reg}\tilde{\boldsymbol{\Phi}} = \frac{1}{n}\tilde{\boldsymbol{\Phi}}\left(\widetilde{\mathbf{K}}+n\sigma^2\mathbf{I}_n\right), \tag{90}$$

which can be defined as

$$\mathbf{C}_{reg}\tilde{\boldsymbol{\Phi}} = \frac{1}{n}\tilde{\boldsymbol{\Phi}}\widetilde{\mathbf{K}}_{reg}\,, \tag{91}$$

where $\widetilde{\mathbf{K}}_{reg} = \widetilde{\mathbf{K}}+n\sigma^2\mathbf{I}_n$. When $n\sigma^2>0$, then $\mathbf{C}_{reg}$ and $\widetilde{\mathbf{K}}_{reg}$ become strictly positive definite and nonsingular. Multiplying (91) with $\mathbf{C}_{reg}$ from the left and $\widetilde{\mathbf{K}}_{reg}$ from the right gives

$$\tilde{\boldsymbol{\Phi}}\widetilde{\mathbf{K}}_{reg}^{-1} = \frac{1}{n}\mathbf{C}_{reg}^{-1}\tilde{\boldsymbol{\Phi}}\,. \tag{92}$$

When $\mathbf{C}_{reg}$ in (88) is multiplied with $\tilde{\phi}(\mathbf{x})$, we obtain

$$\mathbf{C}_{reg}\,\tilde{\phi}(\mathbf{x}) = \left(\frac{1}{n}\tilde{\mathbf{\Phi}}\tilde{\mathbf{\Phi}}^T + \sigma^2\mathbf{I}_{\mathscr{H}}\right)\tilde{\phi}(\mathbf{x}) = \frac{1}{n}\tilde{\mathbf{\Phi}}\tilde{\mathbf{\Phi}}^T\tilde{\phi}(\mathbf{x}) + \sigma^2\mathbf{I}_{\mathscr{H}}\tilde{\phi}(\mathbf{x})\,. \tag{93}$$

From $\tilde{\mathbf{\Phi}}^T\tilde{\phi}(\mathbf{x}) = \tilde{\tilde{\mathbf{k}}}$, (93) will be

$$\mathbf{C}_{reg}\,\tilde{\phi}(\mathbf{x}) = \frac{1}{n}\tilde{\mathbf{\Phi}}\tilde{\mathbf{k}}_{\mathbf{x}} + \sigma^2\tilde{\phi}(\mathbf{x})\,. \tag{94}$$

Since C_{reg} is invertible, multiply (94) with C_{reg}^{-1} from the left. Then

$$\tilde{\phi}(\mathbf{x}) = \frac{1}{n}\mathbf{C}_{reg}^{-1}\tilde{\mathbf{\Phi}}\tilde{\mathbf{k}}_{\mathbf{x}} + \sigma^2\mathbf{C}_{reg}^{-1}\tilde{\phi}(\mathbf{x})\,. \tag{95}$$

Afterwards, multiplying each side of (95) with $\tilde{\phi}(\mathbf{x})^T$ from the left gives

$$\tilde{\phi}(\mathbf{x})^T\tilde{\phi}(\mathbf{x}) = \frac{1}{n}\tilde{\phi}(\mathbf{x})^T\mathbf{C}_{reg}^{-1}\tilde{\mathbf{\Phi}}\tilde{\mathbf{k}}_{\mathbf{x}} + \sigma^2\tilde{\phi}(\mathbf{x})^T\mathbf{C}_{reg}^{-1}\tilde{\phi}(\mathbf{x})\,. \tag{96}$$

Substitute (92) into (96), we get

$$\tilde{\phi}(\mathbf{x})^T\tilde{\phi}(\mathbf{x}) = \tilde{\phi}(\mathbf{x})^T\tilde{\mathbf{\Phi}}\tilde{\mathbf{K}}_{reg}^{-1}\tilde{\mathbf{k}}_{\mathbf{x}} + \sigma^2\tilde{\phi}(\mathbf{x})^T\mathbf{C}_{reg}^{-1}\tilde{\phi}(\mathbf{x})\,. \tag{97}$$

Then the second term, $\tilde{\phi}(\mathbf{x})^T\mathbf{C}_{reg}^{-1}\tilde{\phi}(\mathbf{x})$, in (97) will be

$$\tilde{\phi}(\mathbf{x})^T\mathbf{C}_{reg}^{-1}\tilde{\phi}(\mathbf{x}) = \frac{\tilde{\phi}(\mathbf{x})^T\tilde{\phi}(\mathbf{x}) - \tilde{\phi}(\mathbf{x})^T\tilde{\mathbf{\Phi}}\tilde{\mathbf{K}}_{reg}^{-1}\tilde{\mathbf{k}}_x}{\sigma^2}\,. \tag{98}$$

which is equal to kernel Mahalanobis distance for regularized covariance, shown by $d_{RC}^2(\mathbf{x})$. From $\tilde{\phi}(\mathbf{x})^T\tilde{\phi}(\mathbf{x}) = \tilde{k}_{\mathbf{xx}}$ and $\tilde{\phi}(\mathbf{x})^T\tilde{\mathbf{\Phi}} = \tilde{\mathbf{k}}_{\mathbf{x}}^T$,

$$d_{RC}^2(\mathbf{x}) = d^2(\phi(\mathbf{x}); \{\boldsymbol{\phi}_\mu, \mathbf{C}_{reg}\}) = \frac{1}{\sigma^2}\left(\tilde{k}_{\mathbf{xx}} - \tilde{\mathbf{k}}_{\mathbf{x}}^T\mathbf{K}_{reg}^{-1}\tilde{\mathbf{k}}_{\mathbf{x}}\right)\,. \tag{99}$$

2.3.1 Mahalanobis Distance between Two Data Points

As in (82), the expression (99) provides the distance between $\phi(\mathbf{x})$ and the mean of the centralized data points in the *kernel space*. Following (99), we can also find the distance between two centralized data points in the kernel space. Using the equation in (95), the difference between two data points will be

$$\tilde{\phi}(\mathbf{x}) - \tilde{\phi}(\mathbf{y}) = \frac{1}{n}\mathbf{C}_{reg}^{-1}\tilde{\mathbf{\Phi}}[\tilde{\mathbf{k}}_{\mathbf{x}} - \tilde{\mathbf{k}}_{\mathbf{y}}] + \sigma^2\mathbf{C}_{reg}^{-1}[\tilde{\phi}(\mathbf{x}) - \tilde{\phi}(\mathbf{y})]\,. \tag{100}$$

If we multiply both sides with $[\tilde{\phi}(\mathbf{x}) - \tilde{\phi}(\mathbf{y})]^T$, it becomes

$$[\tilde{\phi}(\mathbf{x}) - \tilde{\phi}(\mathbf{y})]^T[\tilde{\phi}(\mathbf{x}) - \tilde{\phi}(\mathbf{y})] = \frac{1}{n}[\tilde{\phi}(\mathbf{x}) - \tilde{\phi}(\mathbf{y})]^T\mathbf{C}_{reg}^{-1}\tilde{\mathbf{\Phi}}[\tilde{\mathbf{k}}_{\mathbf{x}} - \tilde{\mathbf{k}}_{\mathbf{y}}] + \sigma^2[\tilde{\phi}(\mathbf{x}) - \tilde{\phi}(\mathbf{y})]^T\mathbf{C}_{reg}^{-1}[\tilde{\phi}(\mathbf{x}) - \tilde{\phi}(\mathbf{y})]\,. \tag{101}$$

When (92) is substituted for $\mathbf{C}_{reg}^{-1}\tilde{\mathbf{\Phi}}$ in (101), we obtain

$$[\tilde{\phi}(\mathbf{x}) - \tilde{\phi}(\mathbf{y})]^T[\tilde{\phi}(\mathbf{x}) - \tilde{\phi}(\mathbf{y})] = [\tilde{\phi}(\mathbf{x}) - \tilde{\phi}(\mathbf{y})]^T\tilde{\mathbf{\Phi}}\tilde{\mathbf{K}}_{reg}^{-1}[\tilde{\mathbf{k}}_{\mathbf{x}} - \tilde{\mathbf{k}}_{\mathbf{y}}] + \sigma^2[\tilde{\phi}(\mathbf{x}) - \tilde{\phi}(\mathbf{y})]^T\mathbf{C}_{reg}^{-1}[\tilde{\phi}(\mathbf{x}) - \tilde{\phi}(\mathbf{y})]\,. \tag{102}$$

Then the Mahalanobis distance between $\phi(\tilde{\mathbf{x}})$ and $\phi(\tilde{\mathbf{y}})$ is

$$d^2_{RC}(\mathbf{x},\mathbf{y}) = d^2_{RC}(\phi(\tilde{\mathbf{x}}),\phi(\tilde{\mathbf{y}});\{\boldsymbol{\phi}_\mu,\mathbf{C}\}) = [\tilde{\phi}(\mathbf{x})-\tilde{\phi}(\mathbf{y})]^T\mathbf{C}^{-1}_{reg}[\tilde{\phi}(\mathbf{x})-\tilde{\phi}(\mathbf{y})]$$

$$= \frac{1}{\sigma^2}\left([\tilde{\phi}(\mathbf{x})-\tilde{\phi}(\mathbf{y})]^T[\tilde{\phi}(\mathbf{x})-\tilde{\phi}(\mathbf{y})] - [\tilde{\phi}(\mathbf{x})-\tilde{\phi}(\mathbf{y})]^T\tilde{\boldsymbol{\Phi}}\widetilde{\mathbf{K}}^{-1}_{reg}[\tilde{\mathbf{k}}_\mathbf{x}-\tilde{\mathbf{k}}_\mathbf{y}]\right). \tag{103}$$

Again from $\tilde{\phi}(\mathbf{x})^T\,\tilde{\boldsymbol{\Phi}} = \tilde{\mathbf{k}}^T_\mathbf{x}$ and $\tilde{\phi}(\mathbf{y})^T\,\tilde{\boldsymbol{\Phi}} = \tilde{\mathbf{k}}^T_\mathbf{y}$, (103) will be

$$d^2_{RC}(\mathbf{x},\mathbf{y}) = \frac{1}{\sigma^2}\left[(\tilde{k}_{\mathbf{xx}}-2\tilde{k}_{\mathbf{xy}}+\tilde{k}_{\mathbf{yy}}) - (\tilde{\mathbf{k}}_\mathbf{x}-\tilde{\mathbf{k}}_\mathbf{y})^T\widetilde{\mathbf{K}}^{-1}_{reg}(\tilde{\mathbf{k}}_\mathbf{x}-\tilde{\mathbf{k}}_\mathbf{y})\right]. \tag{104}$$

Note that $\tilde{k}_{\mathbf{xy}}$ is the inner product of kernel of $\mathbf{x}$ with that of $\mathbf{y}$.