1 Exercise 1. Contour lines in Kernel PCA

a.1: The projections on the $i-th$ eigenvector in feature space is given by:

$$\langle v^i, \phi(x)\rangle = \sum_{j=1}^{M} \alpha_j^i k(x^j, x) \tag{1}$$

With a kernel defined by

$$k(x, x') = exp(-\frac{\|x - x'\|}{2\sigma^2}) \ , \ \sigma \in \mathbb{R} \tag{2}$$

and $M = 1$ data point, the Gram matrix is:

$$K = k(x^1, x^1) \tag{3}$$

kPCA requires data to be centered in feature space. This leads to the following transformation (see Exercise 2):

$$\tilde{\mathbf{K}}_{ij} = \mathbf{K}_{ij} - \frac{1}{M}\sum_{k=1}^{M}\mathbf{K}_{ik} - \frac{1}{M}\sum_{k=1}^{M}\mathbf{K}_{kj} + \frac{1}{M^2}\sum_{k,l=1}^{M}\mathbf{K}_{kl} \tag{4}$$

Thus the (single) eigenvector *of the Gram matrix* (solution of the Dual eigenvalue problem $\tilde{K}\alpha = M\lambda\alpha$) is $\alpha_1^1 = 1$, and we can write the projection on the first (and single) eigenvector in feature space, with $M = 1$ since we have only one point:

$$\langle v^i, \phi(x)\rangle = \sum_{j=1}^{M} \alpha_j^i k(x^j, x) = \alpha_1^1 k(x^1, x) \tag{5}$$

The projection in the original data input space is best seen by plotting the contour lines (also called isolines or level curves), see Figure 1. These form circles centered on the datapoint, whose radius grows radially as it moves away from our single datapoint. The width of the curve is proportional to the value of the projection. The farther away one is from our single datapoint, the smaller the value on the projection on the first eigenvector.

Hence, a rbf kernel with a single point allows to group datapoints that are aligned on a circle. It sorts of provide a polar coordinate transformation. The origin of the transformation is here located on our single datapoint.

a.2 With two datapoints, the Gram matrix is:

$$K = \begin{bmatrix} 1 & k(x^1, x^2) \\ k(x^2, x^1) & 1 \end{bmatrix} \tag{6}$$

After centering we get the following centered Gram matrix (assuming $k(x^1, x^2) = k(x^2, x^1)$):

$$\tilde{K} = \frac{1}{2} \cdot \begin{bmatrix} 1 - k(x^1, x^2) & k(x^1, x^2) - 1 \\ k(x^1, x^2) - 1 & 1 - k(x^1, x^2) \end{bmatrix} \tag{7}$$

which has the eigenvectors $\alpha^1 = \frac{1}{\sqrt{2}}[1, 1]^T$ and $\alpha^2 = \frac{1}{\sqrt{(2)}}[1, -1]^T$. This yields the projections:

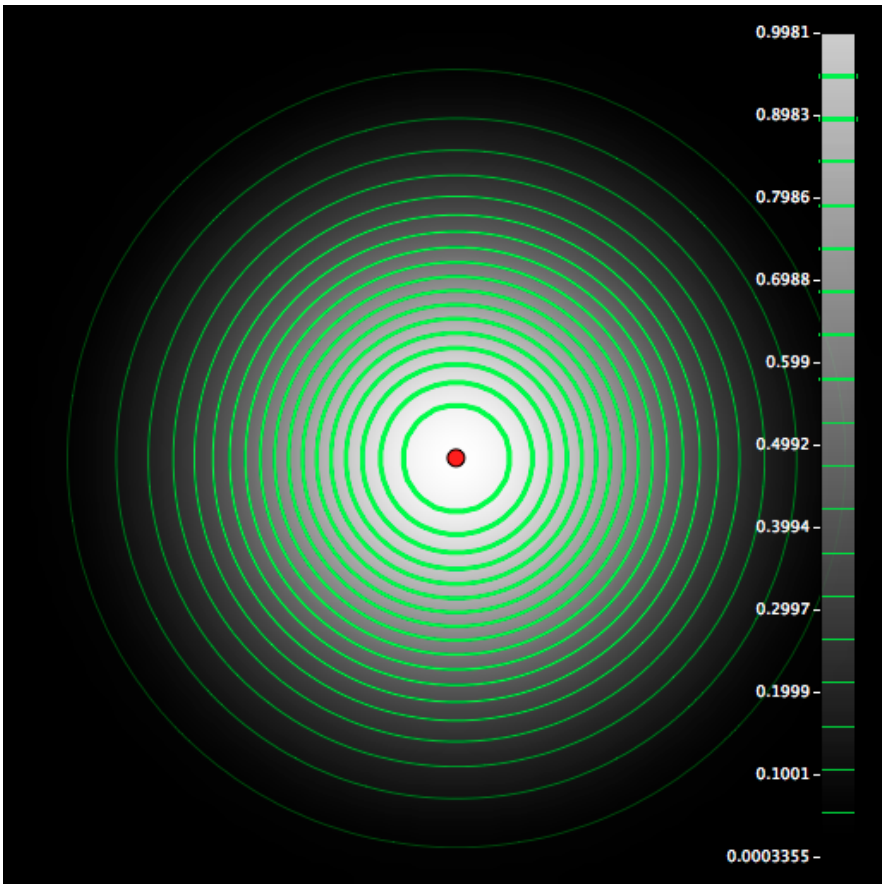

Figure 1: The contour plot of the projection with one data point.

$$\langle v^1, \phi(x)\rangle = \alpha_1^1 k(x^1, x) + \alpha_2^1 k(x^2, x) = \frac{1}{\sqrt{2}} k(x^1, x) + \frac{1}{\sqrt{2}} k(x^2, x) \tag{8}$$

and

$$\langle v^2, \phi(x)\rangle = \alpha_1^2 k(x^1, x) + \alpha_2^2 k(x^2, x) = \frac{1}{\sqrt{2}} k(x^1, x) - \frac{1}{\sqrt{2}} k(x^2, x) \tag{9}$$

The first projection is a sum of two radially decreasing functions (RBF functions, i.e. un-normalised Gaussian fct) around the two data points. As they move away from the two datapoints, the additive term leads them to merge and form elliptic contours around the two datapoints, see Figure 2, left. We see that, once we are far away from the datapoints, the datapoints start acting as a single datapoints and we are back to the case seen previously.

The second projection, which substracts the influence of the two RBF functions located on each datapoint, is shown in the right figure of Figure 2. We see that here the two elliptic form squash against each other and hence form an implicit partition of the space.

This illustrate two means of exploiting RBF projection. The first type helps finding features that group datapoints, whereas the seconde type helps separating datapoints.

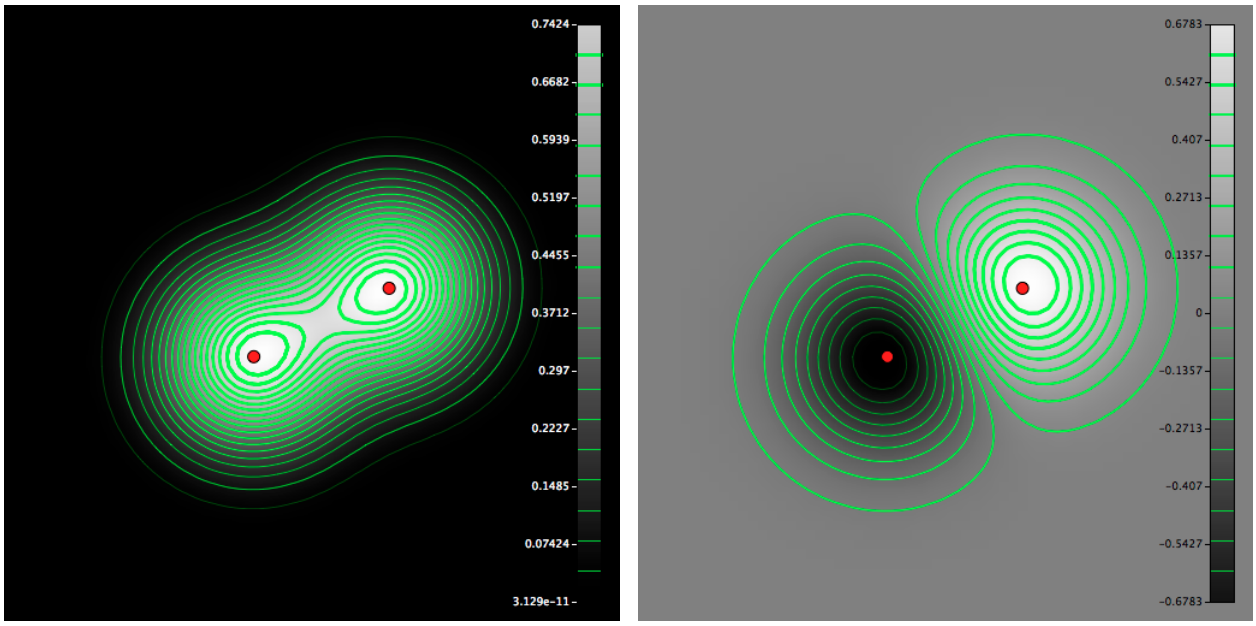

Figure 2: Left: projection onto the first eigenvector with two data points. Here, the contributions from the RBF functions centered on each datapoint is additive and leads them to slowly merge into elliptic contours that grow the datapoints. Right: Projection onto the second eigenvector with two data points. Here, the contributions of the two to the sum have opposed signs.

a.3 Finally with three points we get the Gram matrix:

$$K = \begin{bmatrix} 1 & k(x^1, x^2) & k(x^1, x^3) \\ k(x^2, x^1) & 1 & k(x^2, x^3) \\ k(x^3, x^1) & k(x^3, x^2) & 1 \end{bmatrix} \tag{10}$$

If we assume that points are equidistant, and $k(x^i, x^j) = const = k \quad \forall i, j : i \neq j$, after centering it becomes:

$$\tilde{K} = \frac{1}{3} \cdot \begin{bmatrix} 2-2k & k-1 & k-1 \\ k-1 & 2-2k & k-1 \\ k-1 & k-1 & 2-2k \end{bmatrix} \tag{11}$$

If $k(x^1, x^2) = 0.5$ we get the eigenvectors $\alpha^1 = \frac{1}{\sqrt{3}}[1 \ 1 \ 1]^T$, $\alpha^2 = \frac{1}{\sqrt{2}}[0 \ 1 \ -1]^T$ and $\alpha^3 = \sqrt{\frac{2}{3}}[1 \ -1/2 \ -1/2]^T$.

The projection along the first eigenvector is additive and yields the three RBF to slowly merge as one moves away from the datapoints, and hence tends to group datapoints, see Figure 3. In the case where the kernel width is too small, the three points are not grouped well together anymore , see Figure 4.

The second projection brings us back to the case seen for two datapoints with substractive influence, see solution to a.2. It leads to a partition of the dataspace according to solely two datapoints, ignoring the first datapoint. This is an example of *sparse* projection, where part of the data has been discarded to determine the projection. Sparsity is interesting, as this means that the algorithm managed to extract features common to solely a subgroup of datapoints.

The third projection shows how the symmetric shapes, seen in the 1-datapoint and 2-datapoints

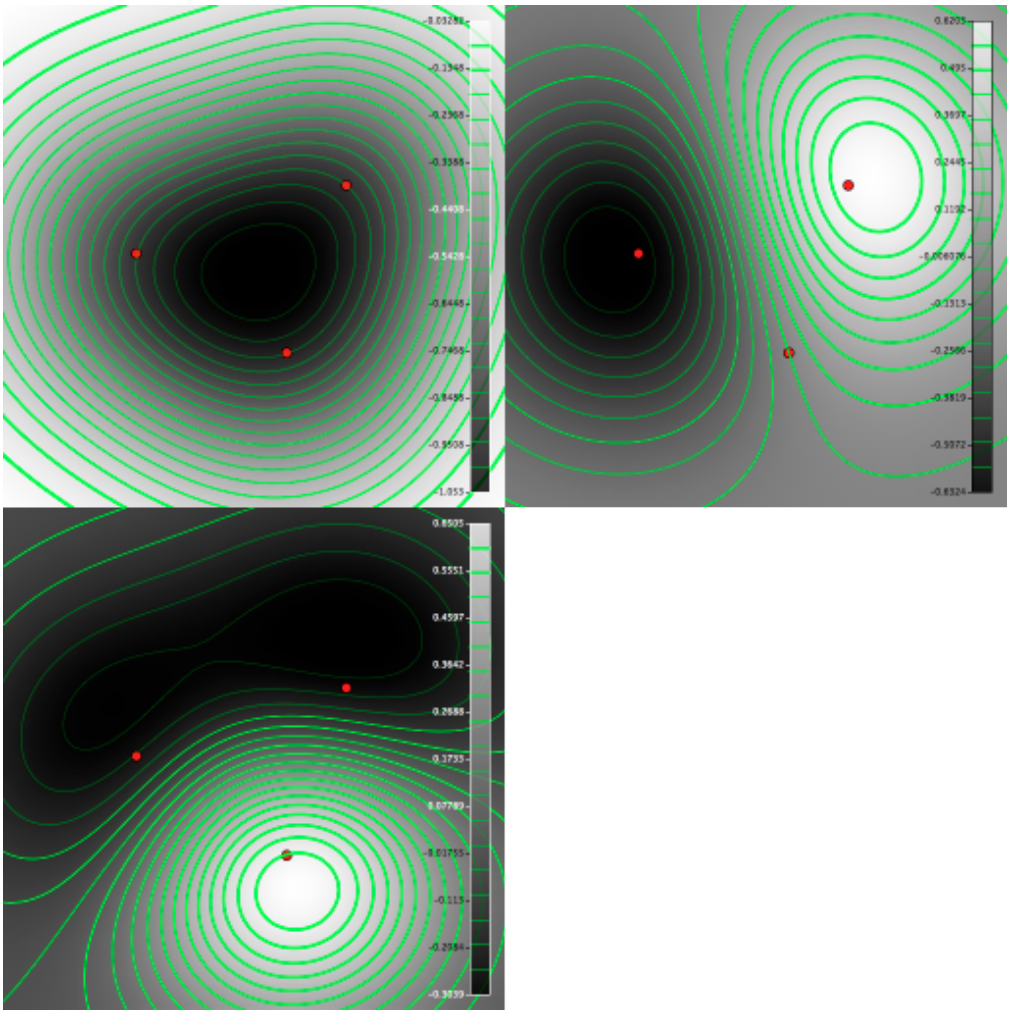

Figure 3: Projections with three equidistant data points. Kernel width $\sigma = 0.01$.

cases, start breaking, as soon as the influence of the datapoints on the projection is no longer equally balanced.

If we change the kernel width, we still get the $[111]^T$ first eigenvector, which groups the 3 points. The other two vectors can result in other type of grouping across the pair of points (you can test this in matlab or with the mldemos software). We non-equidistant points, we find similar type of decompositions with a grouping of all points and then a pairwise grouping of points. However, the isolines become more complex and less symmetric as the entries on the dual eigenvectors are no longer necessarily equal, yet, see Figure 5.

b.1 With a polynomial kernel, $k(x', x) =< x, x' >^p$ is the projection of x onto x'.

With a single datapoint ($M = 1$), we get:

$$\langle v^i, \phi(x) \rangle = \sum_{j=1}^{M} \alpha_j^i k(x^j, x) = k(x^1, x) \tag{12}$$

First, note that, since the data is centered, the single datapoint has coordinates $x^1 = (0, 0)$ and $k(x^1, x) =< x^1, x >= 0, \forall x$. The isolines are then degenerate. If we, however, assume that the

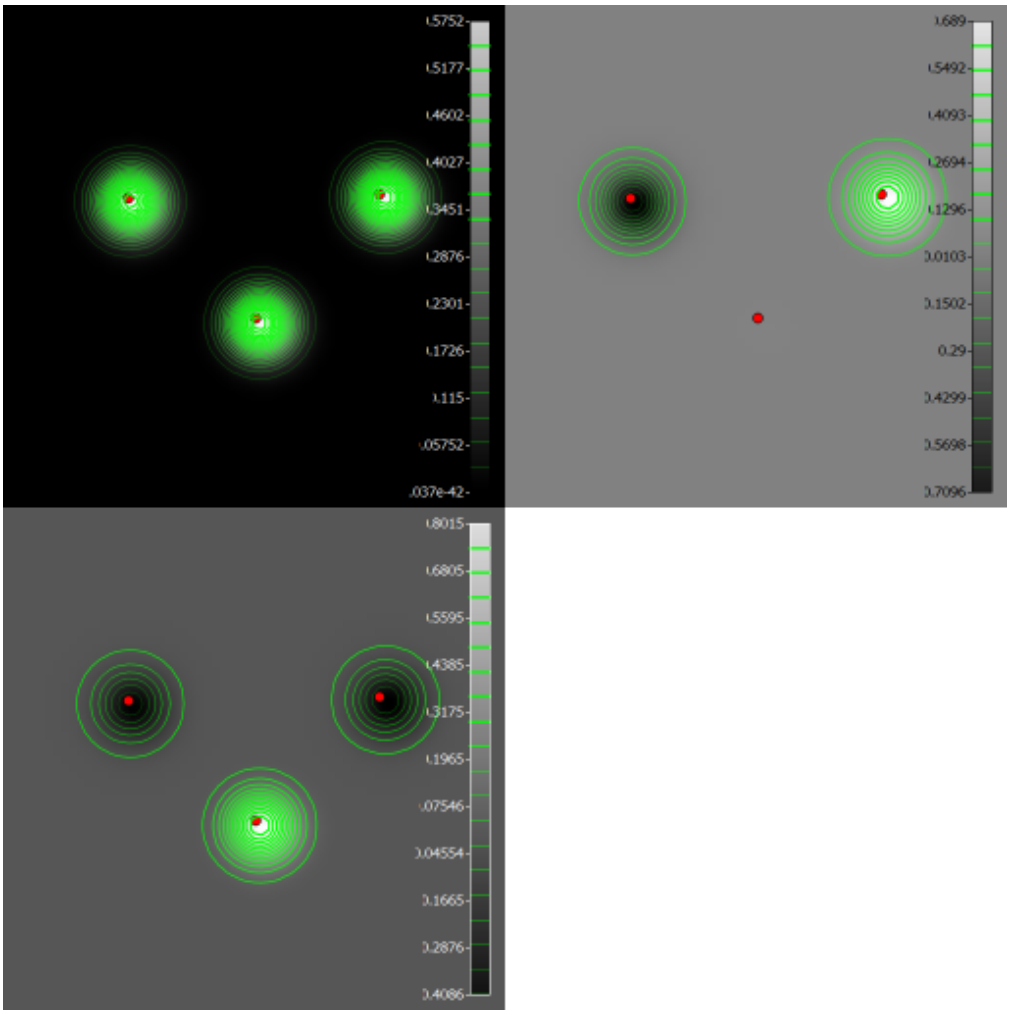

Figure 4: Projections with three equidistant data points. Kernel width $\sigma = 0.0001$.

data are not centered, then, the contour lines correspond to all the points with equal projection onto the vector x^1, see Figure 6. These are, thus, lines orthogonal to the vector pointing from the origin of the system to our single data point, see Figure 7. Changing the order of the polynomial kernel with only one data point does not curve the contour lines - as the points that have the same value will still lie on the lines orthogonal to the data point. It only changes the value of the projection. The higher the order, the faster the value decreases as one goes away from the datapoint.

As mentioned above, kernel PCA assumes that the data are centered. Centering means that the origin of the system is moved to the center of the datapoints. The fact that the contour lines in polynomial kernel are orthogonal to the vector between the datapoint and the origin gives you an intuition as to why it is useful to center the data (when having more than one datapoint). Centering the data creates naturally a symmetry in the projection around the mean of the distribution of datapoints when using polynomial kernel.

b.2

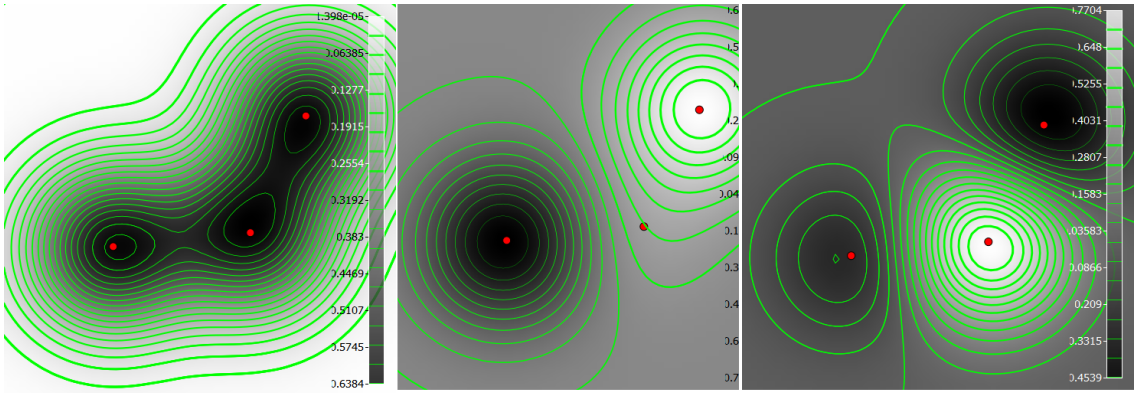

Figure 5: Projections with three non-equidistant data points.

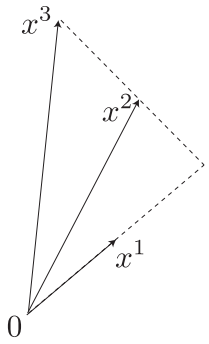

Figure 6: The figure illustrates that points with the same projection on a vector lie on lines orthogonal to the vector. Here, x^2 and x^3 both lie on a line orthogonal to x^1 and as seen their projection on x^1 is the same.

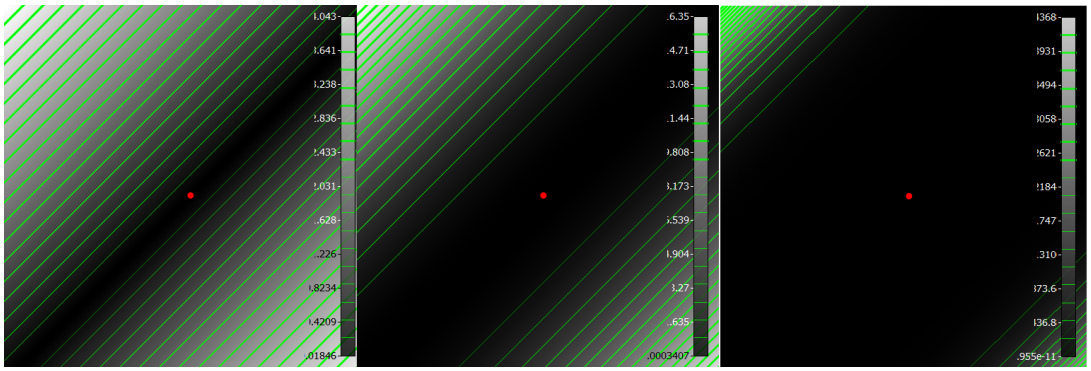

Figure 7: Contour lines of single projection with one data point and polynomial kernel of order 1, 2 and 6.

With $M = 2$ and $p = 2$ we get a Gram matrix proportional to

$$K = \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \tag{13}$$

The dual eigenvectors are $[-1, 1]$ and $[-1, -1]$. The projection onto the first eigenvector is then:

$$\langle v^1, \phi(x) \rangle = \alpha_1^1 k(x^1, x) + \alpha_2^1 k(x^2, x) = \frac{1}{\sqrt{2}} k(x^1, x) - \frac{1}{\sqrt{2}} k(x^2, x) \tag{14}$$

In the general case, we would get a hyperbola for the first projection and an ellipse for the second (see solutions discussed in the kernel lecture). If the vector points are orthogonal to one another, then, the dual eigenvectors are $\alpha^1 = [10]^T$ and $\alpha^2 = [01]^T$. For the first and second projections, the isolines are lines orthogonal to each vector $x^2 + x^1$. In the second projection, the lines are orthogonal to the vector $x^2 - x^1$. The value of the projection will change depending on the order of polynomial. With even order, the values of the projections are symmetric around the origin, whereas with odd order (i.e. $p = 1, 3, 5, ...$), the contour lines are positive on one side and negative on the other side, see Figure 10.

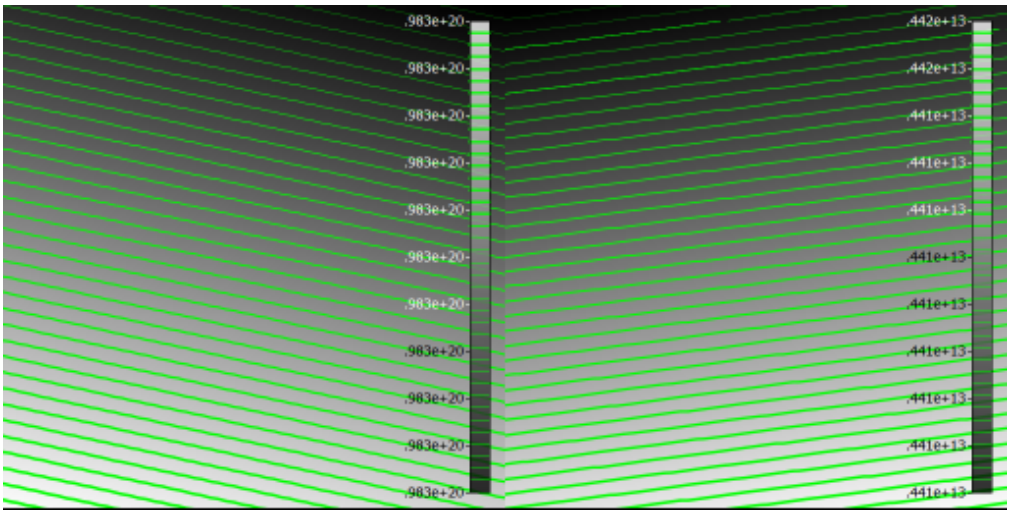

Figure 8: Contour lines of second projection with two data points and polynomial kernel of order 2 (left: 1st projection; right: 2nd projection).

b.3

Increasing the number of datapoints with polynomial of order 2 yields interesting structures. If we again consider the case where the three points are equidistant we get the eigenvectors $\alpha^1 = \frac{1}{\sqrt{3}}[1\ 1\ 1]^T$, $\alpha^2 = \frac{1}{\sqrt{2}}[0\ 1\ -1]^T$ and $\alpha^3 = \sqrt{\frac{2}{3}}[1\ -1/2\ -1/2]^T$. The isolines on the first eigenvector are ellipses with center at the origin and one axe passing through one of the 3 points. The value on the isolines is a function of the distance across the datapoints. The isolines on the second eigenvector are two hyperbolic functions deflected by the two datapoints with zero in-between x^1 and x^2. In the second projection, The isolines on the third eigenvector are also hyperbolic functions deflected with one point acting against the pair of the two other points. The curves remain unchanged as we increase the order of the polynomial for even order. For odd order (i.e. p=3,5,7, etc), we get a different behavior. The first projection is a series of hyperbolic functions, while the second and third projections start to form curves, see Figure 10.

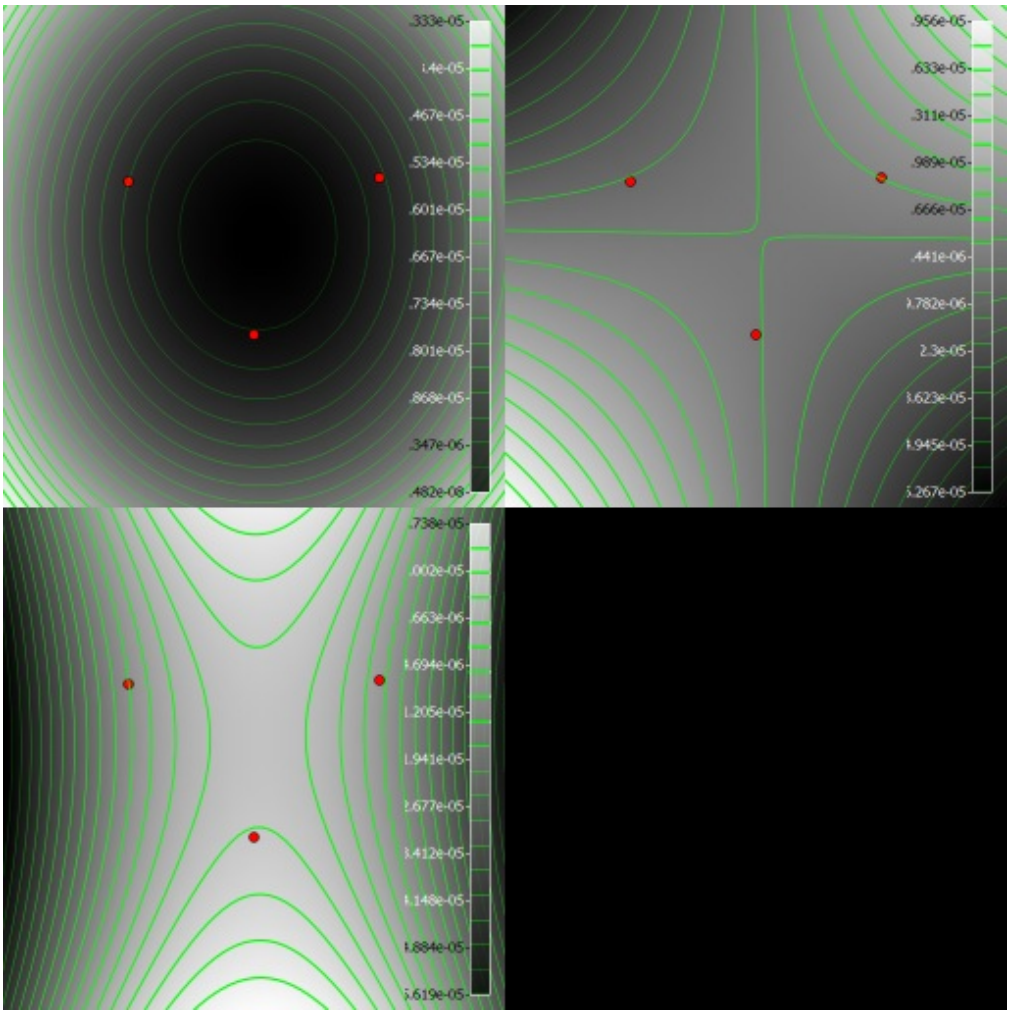

Figure 9: Contour lines of the 3 projections with 3 equidistant datapoints and a polynomial kernel of order 2).

2 Exercise 2: Kernel PCA - data centering

Kernel PCA, like PCA, assumes that the data $\mathbf{X} = \{\mathbf{x}^i\}_{i=1}^M$ are centered, i.e. $\langle \mathbf{X} \rangle = 0$.

2.1

Assuming that the data are not centered prior to computing the Gram matrix $\mathbf{K}$, how can you modify $\mathbf{K}$ to make sure that the projections in the feature space have zero mean?

Recall that each element of $\mathbf{K}$ computes the inner product across two data points, i.e.

$$\mathbf{K}_{ij} = \langle \phi(\mathbf{x}^i), \phi(\mathbf{x}^j) \rangle$$

2.2

True or false: If $\mathbf{K}$ is a positive definite kernel matric, then all of its entries are positive.

1: Observe first that when computing in the original space, the inner product can be rewritten

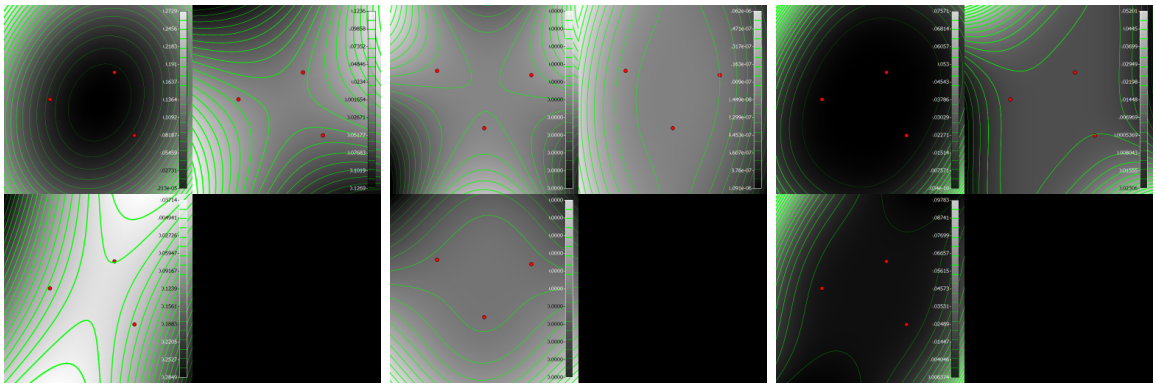

Figure 10: Contour lines of the three projection with three data points and polynomial kernel of order (from left to right) p=2, 3 and 4.

as:

$$\begin{aligned}\langle \mathbf{x}^i - \langle \mathbf{X} \rangle, \mathbf{x}^j - \langle X \rangle \rangle &= \left\langle \mathbf{x}^i - \frac{1}{M}\sum_{k=1}^{M} \mathbf{x}^k, \mathbf{x}^j - \frac{1}{M}\sum_{k=1}^{M} \mathbf{x}^k \right\rangle \\ &= \langle \mathbf{x}^i, \mathbf{x}^j \rangle - \frac{1}{M}\sum_{k=1}^{M} \langle \mathbf{x}^i, \mathbf{x}^k \rangle - \frac{1}{M}\sum_{k=1}^{M} \langle \mathbf{x}^k, \mathbf{x}^j \rangle + \frac{1}{M^2}\sum_{k,l=1}^{M} \langle \mathbf{x}^k, \mathbf{x}^l \rangle\end{aligned}$$

Replacing $\mathbf{x}^i$ by its projection $\phi(\mathbf{x}^i)$ and using the kernel expression $\mathbf{K}_{ij} = \langle \phi(\mathbf{x}^i), \phi(\mathbf{x}^j) \rangle$, we get a new expression for the centered kernel matrix:

$$\tilde{\mathbf{K}}_{ij} = \mathbf{K}_{ij} - \frac{1}{M}\sum_{k=1}^{M} \mathbf{K}_{ik} - \frac{1}{M}\sum_{k=1}^{M} \mathbf{K}_{kj} + \frac{1}{M^2}\sum_{k,l=1}^{M} \mathbf{K}_{kl}$$

2: No, positive definiteness requires only that all of the eigenvalues of $\mathbf{K}$ be positive. Eigenvalues are computed by solving $|\mathbf{K} - \lambda \mathbf{I}| = 0$, so by example ($\mathbf{K} = \begin{bmatrix} 2 & -1 \\ -1 & 2 \end{bmatrix}$) we see that a positive definite matrix may have negative elements.

3 Exercise 3: Matrix Decompositions

A: i) To find the eigenvalues, characteristic equation $det|A - \lambda I| = 0$ must be solved

The eigenvalues are $\lambda_1 = 6$, $\lambda_2 = 1$.

To find the eigenvectors e_j, one computes for each eigenvalues λ_j, $j = 1, 2$: $(A - \lambda I)\, e_j = 0$.

We find: $e_1 = \frac{1}{\sqrt{53}} \begin{bmatrix} 2 \\ 7 \end{bmatrix}$, and $e_2 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix}$.
The matrix here are normalized, but any combination combination of two vectors which are parallel to e^1 and e^2 respectively is can be chosen.
Note: In the case that of a **symmetric matrix** A and normalized eigenvectors, we have $V^{-1} = V^T$ (orthonormal matrix). For the general case, we only have all linearly independent eigenvectors for a full rank of the initial matrix A, i.e. $\text{rank}(V) = \text{rank}(A)$.

ii) The eigenvalues in
Matlab:

```
A = [-1  2;
     -7  8];
[V, D] = eig(A);
```

or python:

```
import numpy as np
A = np.array([[-1,  2],
              [-7,  8]])
D, V = np.linalg.eig(A)
```

Note: There are many similarities in numpy and matlab. Generally speaking, it is adviced to use *numpy.arrays()* for matrix representation and most matlab function are found in the *numpy* and *scipy* libraries. Several guides can be found online, e.g `https://docs.scipy.org/doc/numpy/user/numpy-for-matlab-users.html`.

B: This exercise gives an inside of a geometric interpretation of matrices and the eigenvalue decomposition.
i) The multiplication of a vector with the inverse of a matrix V^{-1} is the representation of this vector in the reference frame spanned by column vectors e_j of the original matrix V.
In the exercise, the vector is parallel to the second eigenvector e_2, hence can be fully represented by e_2 : $x' = V^{-1}x = \begin{bmatrix}0 & \sqrt{2}\end{bmatrix}^T$

ii) The multiplication of a vector with a diagonal matrix Λ is equivalent to stretching along the axis with diagonal elements λ_j. In the exercise, the eigenvalue corresponding to the second eigenvector is equal to 1, i.e. no change for the vector: $x'' = \Lambda x' = \lambda_2 \|x\| = \begin{bmatrix}0 & \sqrt{2}\end{bmatrix}^T$

iii) The multiplication of a vector with a matrix A is the projection back from the space spanned by the column vectors e_j of A to the original space: $x''' = Vx'' = e_2\|x\| = \begin{bmatrix}1 & 1\end{bmatrix}^T$.

4 Exercise 4: Probabilistic PCA - derivation (optional)

Probabilistic PCA assumes that the N dimensional observed dataset $\mathbf{X} = \{\mathbf{x}^i\}_{i=1}^M$ was generated by an underlying process modeled as:

$$\mathbf{x}^i = \mathbf{W}^\top \mathbf{z}^i + \mu + \mathcal{N}(0, \mathbf{I}\sigma_\epsilon^2)$$

where $\mathbf{Z}$ are a set of $Q < N$-dimensional latent variables, μ is an offset, and σ_ϵ^2 is the variance of the zero mean Gaussian noise.

Show that the log likelihood estimate of the projection matrix $\mathbf{W}$ is:

$$\log(L(\mathbf{B}, \mu))) = -\frac{M}{2}\{N\log(2\pi) + \log|\mathbf{B}| + tr(\mathbf{B}^{-1}\mathbf{C})\}$$

where

$$\mathbf{C} = \frac{1}{M}\sum_{i=1}^{M}(\mathbf{x}^i - \mu)(\mathbf{x}^i - \mu)^\top$$

is the covariance matrix of $\mathbf{X}$ and

$$\mathbf{B} = \mathbf{W}^\top \mathbf{W} + \sigma_\epsilon^2 \mathbf{I}$$

The conditional pdf $p(\mathbf{x}|\mathbf{z})$ is given by:

$$p(\mathbf{x}|\mathbf{z}) = \mathcal{N}(\mathbf{W}^\top \mathbf{z} + \mu, \sigma_\epsilon^2 \mathbf{I})$$

marginalizing over the latent variables, assuming $p(\mathbf{z}) = \mathcal{N}(0, \mathbf{I})$, we get:

$$p_{\mathbf{z}}(\mathbf{x}) = \mathcal{N}(\mu, \mathbf{W}^\top \mathbf{W} + \sigma_\epsilon^2 \mathbf{I})$$

The likelihood:

$$L(\mathbf{B}, \mu) = \prod_{i=1}^{M} p_{\mathbf{z}}(\mathbf{x}^i|\mathbf{B}, \mu)$$

then:

$$\log\left(\prod_{i=1}^{M} p_{\mathbf{z}}(\mathbf{x}^i|\mathbf{B}, \mu)\right) = \sum_{i=1}^{M} \log\left(p_{\mathbf{z}}(\mathbf{x}^i|\mathbf{B}, \mu)\right)$$

Replacing $p_{\mathbf{z}}$ by the definition of the multidimensional Gaussian distribution:

$$\begin{aligned}
\log(L(\mathbf{B}, \mu))) &= \sum_{i=1}^{M} \left(\log\left(\frac{1}{\sqrt{(2\pi)^N |\mathbf{B}|}}\right) - \frac{1}{2}\left((\mathbf{x}^i - \mu)^\top \mathbf{B}^{-1}(\mathbf{x}^i - \mu)\right)\right) \\
&= -M\log(\sqrt{(2\pi)^N |\mathbf{B}|}) - \frac{1}{2}\sum_{i=1}^{M}\left((\mathbf{x}^i - \mu)^\top \mathbf{B}^{-1}(\mathbf{x}^i - \mu)\right) \\
&= -\frac{M}{2}\{N\log(2\pi) + \log(|\mathbf{B}|)\} - \frac{1}{2}\sum_{i=1}^{M}\left((\mathbf{x}^i - \mu)^\top \mathbf{B}^{-1}(\mathbf{x}^i - \mu)\right)
\end{aligned}$$

recalling that $\mathbf{x}^\top \mathbf{B}^{-1}\mathbf{x} = tr\left(\mathbf{B}^{-1}(\mathbf{x}\mathbf{x}^\top)\right)$ and $tr(\Sigma) + tr(\Theta) = tr(\Sigma + \Theta)$

$$\begin{aligned}
\log(L(\mathbf{B}, \mu))) &= -\frac{M}{2}\{N\log(2\pi) + \log(|\mathbf{B}|)\} - \frac{1}{2}\sum_{i=1}^{M} tr\left(\mathbf{B}^{-1}(\mathbf{x}^i - \mu)(\mathbf{x}^i - \mu)^\top\right) \\
&= -\frac{M}{2}\{N\log(2\pi) + \log(|\mathbf{B}|) + tr\left(\frac{1}{M}\sum_{i=1}^{M}\mathbf{B}^{-1}(\mathbf{x}^i - \mu)(\mathbf{x}^i - \mu)^\top\right)\} \\
&= -\frac{M}{2}\{N\log(2\pi) + \log(|\mathbf{B}|) + tr\left(\mathbf{B}^{-1}\frac{1}{M}\sum_{i=1}^{M}(\mathbf{x}^i - \mu)(\mathbf{x}^i - \mu)^\top\right)\} \\
\log(L(\mathbf{B}, \mu))) &= -\frac{M}{2}\{N\log(2\pi) + \log|\mathbf{B}| + tr(\mathbf{B}^{-1}\mathbf{C})\}
\end{aligned}$$