Kernels, the Feature Space, and Model Selection

Kernel Methods

Kernel methods all learn

$$\sum_{i=1}^{n} k(x, X_i)\alpha_i + \alpha_0$$

Methods differ in how to determine α.

Kernels k must be Mercer-kernels.

Mercer-kernel

Let $\mathcal{X}$ be a compact subset of $\mathbb{R}^n$. A continuous function $k\colon \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is called a *Mercer kernel* if for all $f \in L^2(\mathcal{X})$,

$$\int_{\mathcal{X}} \int_{\mathcal{X}} k(x, y) f(x) f(y) dxdy \geq 0.$$

Mercer kernels correspond to symmetric positive definite matrices.

Mercer's Formula

Mercer's Formula

Let k be a Mercer's kernel. Then there exist $(\psi_i) \in L^2(\mathcal{X})$, and $\lambda_i \geq 0$ with $\sum_i \lambda_i < \infty$ such that

$$k(x, y) = \sum_{i=1}^{\infty} \lambda_i \psi_i(x) \psi_i(y)$$

where the convergence is uniform over $\mathcal{X}$.

The Kernel Trick

Mercer's Formula allows to re-interpret kernel methods as **linear methods on transformed data!**

Define the *feature map*

$$\Psi(x) = (\sqrt{\lambda_1}\psi_1(x), \sqrt{\lambda_2}\psi_2(x), \dots)$$

Ψ maps the data non-linearily into the feature map $\mathcal{F} = \ell^2$ (infinite-dimensional!) such that

$$\langle \Psi(x), \Psi(y) \rangle_{\ell^2} = \sum_{i=1}^{\infty} \sqrt{\lambda_i}\psi_i(x)\sqrt{\lambda_i}\psi_i(y) = k(x, y).$$

(It's actually well-defined since

$$\|\Psi(x)\|_{\ell^2} = \langle \Psi(x), \Psi(y) \rangle_{\ell^2} = k(x, x) < \infty)$$

The Kernel Trick (cont'd)

Now:

$$\begin{aligned}\sum_{i=1}^{n} k(x, X_i)\alpha_i + \alpha_0 &= \sum_{i=1}^{n} \langle \Psi(x), \Psi(X_i) \rangle_{\ell^2} \alpha_i + \alpha_0 \\ &= \langle \Psi(x), \sum_{i=1}^{n} \Psi(X_i)\alpha_i \rangle_{\ell^2} + \alpha_0 \\ &= \langle \Psi(x), w \rangle_{\ell^2} + \alpha_0\end{aligned}$$

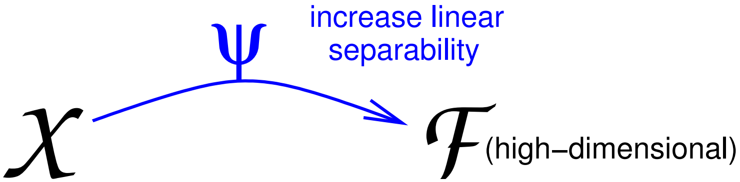
with $w = \sum_{i=1}^{n} \Psi(X_i)\alpha_i$

Implicit feature map and separability

- Mercer kernel implicitly defines feature map Ψ.
- Ψ non-linearly transforms the data, making it more separable.
- Kernel methods are linear methods on transformed data.
- Kernel trick permits to efficiently compute scalar products even in infinite-dimensional spaces.

$\rightsquigarrow$ kernels permit to use well-known linear methods also on non-linear data sets.

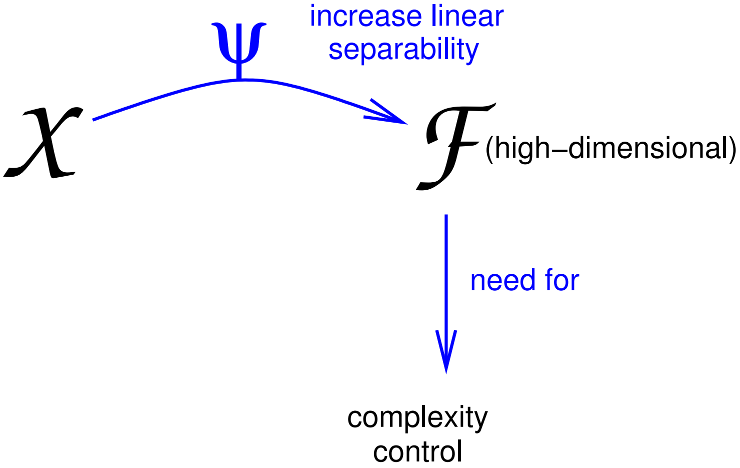
The Standard Picture



The Capacity Control Argument

- Feature spaces are usually quite high-dimensional (even infinite-dimensional).
- Inference in high-dimensional spaces is hard (curse-of-dimensionality).
- Use learning algorithms whose "capacity" is finite and independent of the dimensionality (for example, large margin classifiers).

The Standard Picture—with capacity control



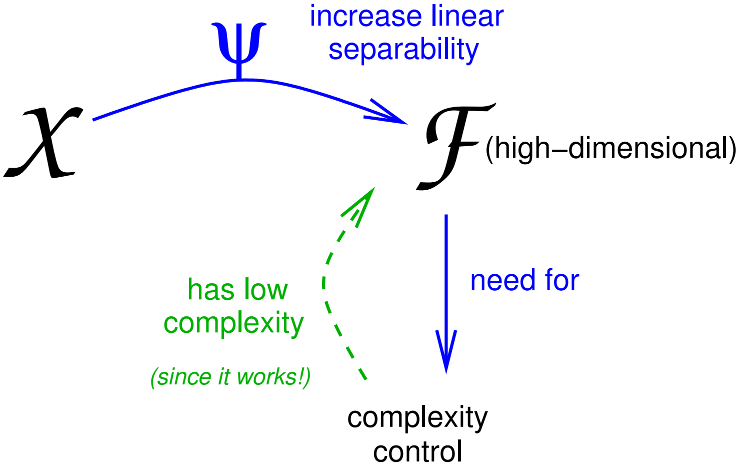
Capacity Control and the Complexity of $\Psi(\mathcal{X})$

- Finite complexity of hypothesis class does not imply good performance! It simply means that overfitting will not occur.
- *Empirically* kernel methods work well, so capacity control and feature map *must* cooperate well.

Overview of theoretical results on $\Psi(\mathcal{X})$:

- At scale 0, infinite VC-dimension.
- Finite fat-shattering dimensions at finite scale $\gamma > 0$.
- Variance concentrated in a few dimensions.

Low complexity of $\Psi(X)$



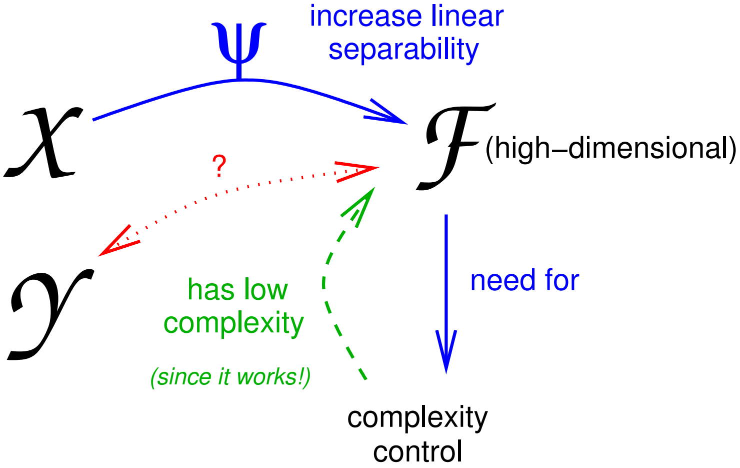
Extending the Standard Picture—What about Y?

- Feature map Ψ built independently of Y.
- Relevant information about Y should be contained in a low-complexity subspace of $\mathcal{F}$.

Overview of theoretical results:

- If kernel matches problem, information about Y contained in low-dimensional subspace.
- Relevant dimensionality can be estimated practically.

The Extended Standard Picture



Summary

- Kernels define implicit non-linear transformations.
- Kernel methods are linear methods on this transformed data in high-dimensional spaces.
- Nevertheless, the relevant information seems to be contained in a subspace, otherweise capacity control could not work.

The Feature Map

Usually, the feature space is not explicitly constructed.

Typical kernel functions

1. $k(x, y) = (\langle x, y \rangle + 1)^d$
2. $k(x, y) = exp(-\|x - y\|^2 / 2w)$
3. others: spectrum kernel, string kernel, etc.

How does the associated feature space look like?

General Properties

General statements on the shape of $\Psi(\mathcal{X})$ are easy to come by, but they are not very helpful:

- Given sufficient smoothness of k, $\Psi(\mathcal{X})$ is a *submanifold* of $\mathcal{F}$ with the same dimension as the input space.
- n points can only span an n-dimensional subspace of $\mathcal{F}$.

More insights needed.

Explicit Feature Spaces

For some kernels, the feature map can be written down explicitly.

For example, for a polynomial kernel $k(x,y) = (\langle x, y\rangle + 1)^d$:

$$\phi(x) = \left(\sqrt{\binom{n}{k}\binom{n}{i_1,\ldots,i_n}} x_1^{i_1}\cdots x_n^{i_n} \right)_{k=1,\ldots,d,\; i_1+\cdots+i_n=k}$$

with

$$\binom{n}{k} = \frac{n!}{k!(n-k)!}, \quad \binom{n}{i_1,\ldots,i_r} = \frac{n!}{i_1!\cdots i_r!}, \text{ for } i_1+\cdots+i_r = n.$$

(multi-nomial coefficients)

Simple Geometric Properties

Norms in feature space:

$$\|\Psi(x)\| = \sqrt{\langle x, x\rangle} = \sqrt{k(x,x)}.$$

Angles in feature space:

$$\cos \angle(x,y) = \cos \frac{\langle x, y\rangle}{\|x\|\|y\|} = \cos \frac{k(x,y)}{\sqrt{k(x,x)k(y,y)}}$$

Distances in feature space:

$$\begin{aligned}\|x-y\| &= \sqrt{\langle x-y, x-y\rangle} = \sqrt{\langle x,x\rangle - 2\langle x,y\rangle + \langle y,y\rangle} \\ &= \sqrt{k(x,x) - 2k(x,y) + k(y,y)}.\end{aligned}$$

Simple Geometric Properties of the RBF-Kernel

Gaussian kernel:

$$k(x, y) = \exp\left(-\frac{\|x - y\|^2}{2w}\right)$$

Feature map $\mathcal{F} = \ell^2 =$ all sequences (z_i) with $\sum_{i=1}^{\infty} z_i^2 \leq \infty$.

Data lives on the surface of the infinite-dimensional unit-sphere:

- $k(x, x) = 1 \rightsquigarrow \|\Psi(x)\| = 1$.
- $k(x, y) \geq 0 \rightsquigarrow \angle(\Psi(x), \Psi(y)) \leq 90°$.
- $\|\Psi(x) - \Psi(y)\| \leq \sqrt{2}$

The Infinite-Dimensional Unit-Sphere

The infinite-dimensional unit-sphere has some funny properties:

- Although it is bounded, it is not compact: The sequence $e_i = (0, \ldots, 0, 1, 0, \ldots)$ with the 1 at ith position does not converge.
- Although two points cannot be further than $\sqrt{2}$ apart, there is an infinite amount of directions to go from each point.
- In d-dimensions, compare the volume V_d of a ball of radius $1/2$ with that of its containing unit cube. While the volume of the unit cube is 1, $V_d \to 0$ as $d \to \infty$.

The Empirical Feature Map

The λ_is and ψ_is in Mercer's formula are not unique.

If one considers only the given points $X_1, \ldots, X_n$, *empirical kernel maps* can be constructed.

Both use the eigenvalues and eigenvectors of the kernel matrix

$$\mathbf{K} u_i = l_i u_i$$

summarized as $\mathbf{KU} = \mathbf{LU}$.

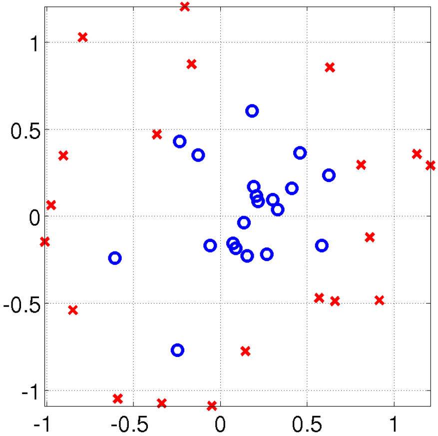
Empirical feature maps:

1. $\Psi(X_i) = \sum_{j=1}^{n} \sqrt{l_j} u_j [u_j]_i$, in matrix notation $\mathbf{F} = \mathbf{U}\mathbf{L}^{1/2}\mathbf{U}^\top$.
2. $\Psi(X_i) = (\sqrt{l_1}[u_1]_i, \ldots, \sqrt{l_n}[u_n]_i)$, or $\mathbf{F} = \mathbf{L}^{1/2}\mathbf{U}^\top$.

Then:

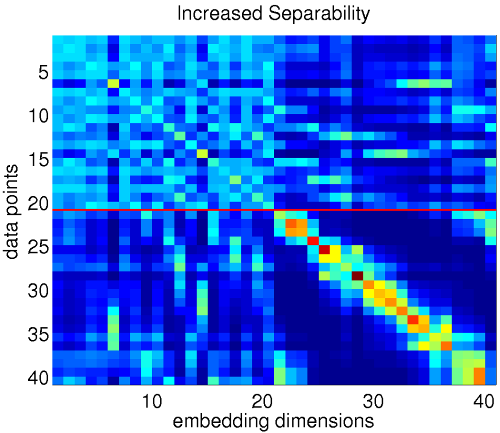
$$\mathbf{F}^\top\mathbf{F} = \mathbf{U}\mathbf{L}^{1/2}\mathbf{U}^\top\mathbf{U}\mathbf{L}^{1/2}\mathbf{U}^\top = \mathbf{U}\mathbf{L}\mathbf{U}^\top = \mathbf{K} = \Psi(\mathbf{X})^\top\Psi(\mathbf{X}).$$

Visualizing the Feature Space



Example data set (not linearly separable)

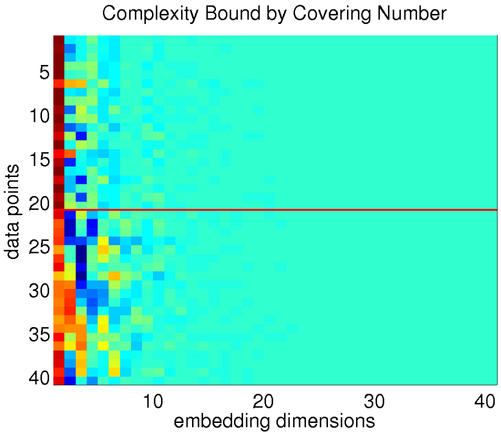
Visualizing the Feature Space



Using empirical feature map $\mathbf{F} = \mathbf{U}\mathbf{L}^{1/2}\mathbf{U}^\top$.

Shows increases separability

Visualizing the Feature Space



Using empirical feature map $\mathbf{F} = \mathbf{L}^{1/2}\mathbf{U}^{\top}$.

Also shows that higher dimensions do not contain much variance.

Kernel Principal Component Analysis

Second feature map $\Psi(X_i) = (\sqrt{l_1}[u_1]_i, \dots, \sqrt{l_n}[u_n]_i)$ closely related to *Kernel PCA*.

Classical PCA: compute eigenvectors of *covariance matrix*

$$\mathbf{C} = \frac{1}{n}\sum_{l=1}^{n}[X_l]_i[X_l]_j = \frac{1}{n}\mathbf{X}\mathbf{X}^\top$$

Size of $\mathbf{C}$ is dimensionality of the X_is $\rightsquigarrow$ infeasible for infinite-dimensional feature spaces.

Kernel PCA

Solution:

- $\mathbf{X}^\top\mathbf{X}$ has the same eigenvalues, eigenvectors are related via $u \mapsto \mathbf{X}^\top u = v$. ($u$ eigenvector of $\mathbf{C}$, v eigenvector of $\mathbf{K}$.
- $\mathbf{X}^\top\mathbf{X}$ computes all pair-wise scalar products $\rightsquigarrow$ $\Psi(\mathbf{X})^\top\Psi(\mathbf{X}) = \mathbf{K}$.

Instead of principal directions $v_i \in \ell^2$, consider *principal components* $f_u \colon \mathcal{X} \to \mathbb{R}$

$$f_i(x) = \langle \Psi(x), v_i \rangle = \frac{1}{l_i} \sum_{j=1}^{n} k(x, X_j)[u_i]_j.$$

Evaluated on $\mathbf{X} = (X_1, \ldots, X_n)$, $f_i(\mathbf{X}) = u_i$ (eigenvectors of $\mathbf{K}$).

Kernel PCA and Feature Maps

Summary:

- $\begin{matrix}\text{Eigenvectors}\\ \text{Eigenvalues}\end{matrix}$ of $\mathbf{K} \equiv$ principal $\begin{matrix}\text{directions}\\ \text{values}\end{matrix}$ in $\mathcal{F}$.
- Eigenvector $u_i \equiv i$th coordinate in $\mathcal{F}$.
- Scaled by $\sqrt{l_i} \rightarrow$ only leading dimensions carry much variance.

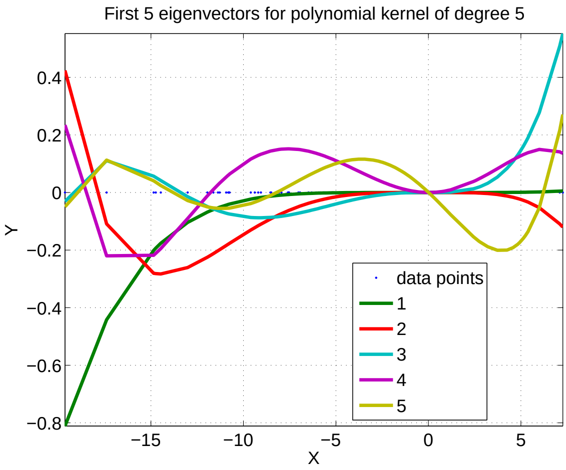
Therefore, one can visualize the mapping into feature space by looking at the leading eigenvectors.

Typical effects

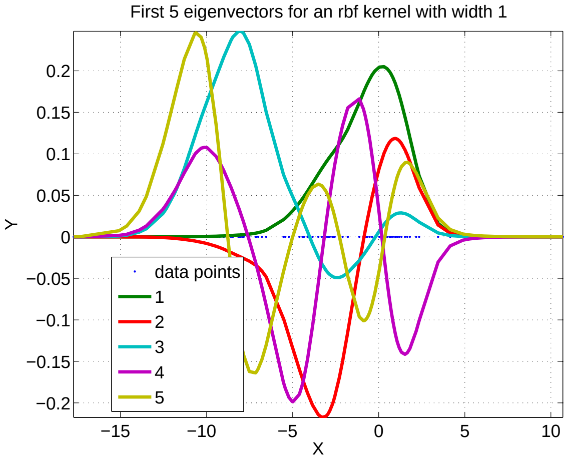
There are two typical effects:

- Eigenvectors become increasingly complex.
- For data sets with separated clusters, the eigenvectors for the clusters can become independent.

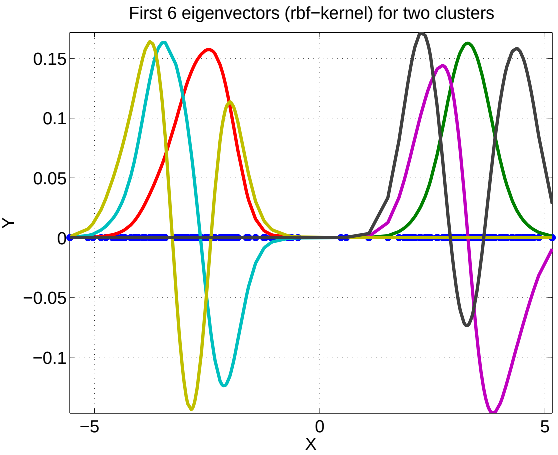
Increasingly Complexity of Eigenvectors



Increasingly Complexity of Eigenvectors



Independent sets of eigenvectors for clusters



Data consists of two clusters at 3 and -3.

Summary

- Kernels can be visualized using the eigenvectors of the kernel matrix.
- This corresponds to plotting the Kernel PCA dimensions.
- Higher dimensions have only small complexity.
- Higher dimensions are more complex.

Next: theoretical results.

Embedded image has finite complexity

Theoretically, for n data points, $\Psi(\mathbf{X})$ has at most dimension n.

For rbf-kernels (up to numerical errors), every data set can be separated perfectly (infinite VC-dimension).

But at "finite scale", situation looks different!

Guaranteed Fast Decay of Kernel Principal Values

The space spanned by the leading p kernel PCA directions minimizes the projection error among all such spaces.

$\rightsquigarrow$ at a finite scale γ, $\Psi(\mathcal{X})$ is essentially contained in a low-dimensional subspace.

This also holds for the empirical kernel map.

Convergence for kernel PCA

The reason is that kernel PCA approximates the asymptotic kernel PCA with relative perturbation bounds.

finite sample setting		**asymptotic setting**
$[\mathbf{K}x]_i = \sum_{j=1}^{n} k(X_i, X_j)X_j$	$\rightsquigarrow$	$T_k f(s) = \int_{\mathcal{X}} k(s,t) f(t) P(dt)$
l_i eigenvalues of $\mathbf{K}$		λ_i eigenvalues of T_k

Convergence results

Eigenvalues are approximated with high relative error (approximation error for smaller eigenvalues is smaller)

- Individual eigenvalues [2]:

$$|l_i - \lambda_i| \leq \lambda_i C(r, n) + E(r, n)$$

  with $C(r, n) \to 0$ for $n \to \infty$, $E(r, n) \to 0$ for $r \to \infty$.

- Tail sums of eigenvalues [3]:

$$\left| \sum_{i=d}^{n} l_i - \sum_{i=d}^{\infty} \lambda_i \right| \leq C \sqrt{\sum_{i=1}^{\infty} \lambda_i} + E.$$

$\rightsquigarrow$ also for empirical feature spaces, $\Psi(X)$ is concentrated in leading dimensions.

Summary

- For a given scale, $\Psi(X)$ has finite dimensionality.
- Result obtained by linking finite to asymptotic setting.
- Accurate approximation errors for individual eigenvalues and tail-sums of eigenvalues.

Looking at the Labels

- So far: Only few principal directions carry large variance.
- Projecting to finite subspace in $\mathcal{F}$ leads only to small error.
- What about the label information?

Further applications:

- Preprocessing for classification.
- De-noise labels Y.
- Explicitly construct low-dimensional feature space.
- Estimate data set complexity / noise for given kernel.

Y and the PCA Directions in Feature Space

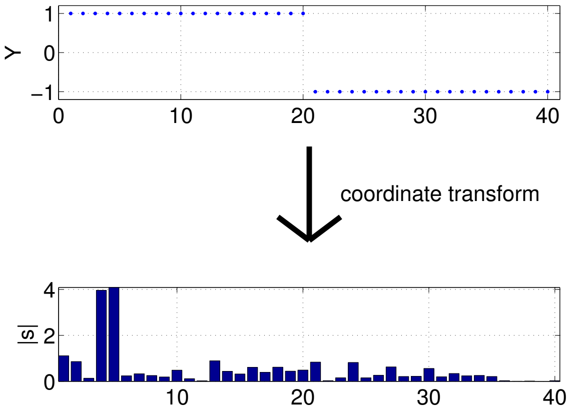
Recall: eigenvector u_i are principal components f_i evaluated on $X_1, \ldots, X_n$.

Decomposition of $Y = (Y_1, \ldots, Y_n)$ along kernel PCA directions

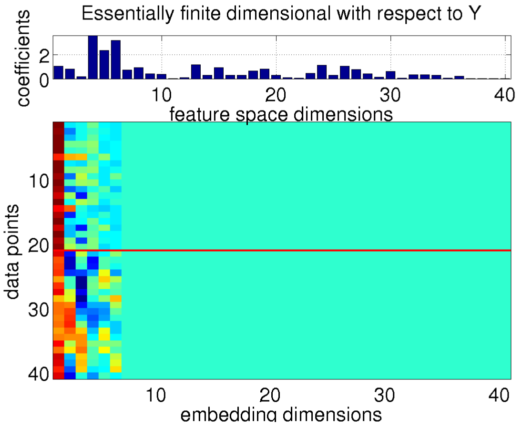
$$s = \mathbf{U}^\top Y = ({u_1}^\top Y, \ldots, {u_n}^\top Y)$$

Since eigenvectors are orthogonal u_i (since $\mathbf{K}$ is symmetric), this amounts to a change of basis via a rotation in $\mathbb{R}^n$.

Coordinate transform of the Ys

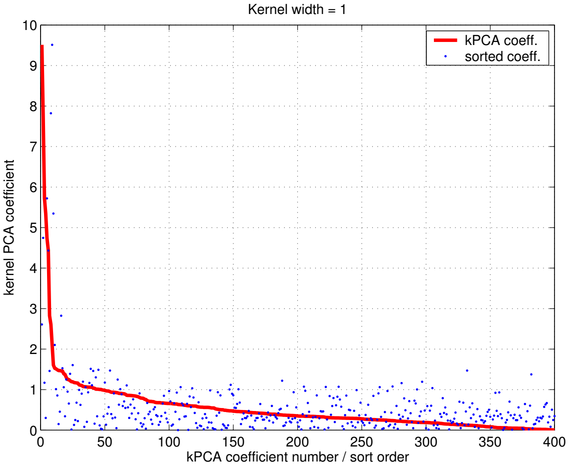


Including Y in the feature map picture



It seems that information is contained in leading PCA directions

Alternative view: Sorting coefficients



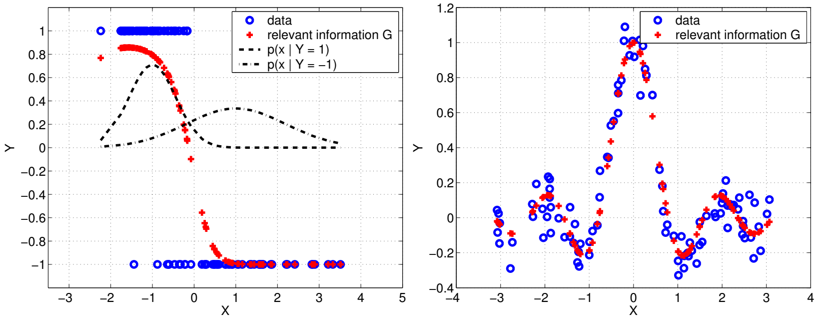
Even if you sort the contributions s_i, the cut-off stays at roughly the same position.

Theoretical Analysis

Goal: Separate "relevant information" from noise:

$$
\begin{aligned}
Y_i &= g(X_i) + \varepsilon_i, \\
g(x) &= E(Y|X = x), \\
G &= (g(X_1), \ldots, g(X_n)).
\end{aligned}
$$

The relevant information



classification / regression

Go to the asymptotic setting

Again, the answer is found in the asymptotic setting and convergence bounds.

finite sample setting		asymptotic setting
$[\mathbf{K}x]_i = \sum_{j=1}^{n} k(X_i, X_j)X_j$	$\rightsquigarrow$	$T_k f(s) = \int_{\mathcal{X}} k(s,t)f(t)P(dt)$
u_i eigenvector of $\mathbf{K}$		ψ_i eigenfunction of T_k
$s_i = u_i{}^\top G$		$\sigma_i = \langle \psi_i, g \rangle$

This time, for spectral projections.

Assumption

Not all data sets will be well-behaved!

Minimal assumption: Kernel and data set match in the following sense:

g asymptotically representable by T_k, (exists h such that $g = T_k h$):

$$\rightsquigarrow g(x) = \sum_{i=1}^{\infty} \lambda_i \alpha_i \psi_i(x) \qquad \rightsquigarrow \sigma_i = \lambda_i \alpha_i = O(\lambda_i).$$

Note: Constant unspecified, measures fit between kernel and data set.

Equivalence between Finite Sample and Asymptotic Setting

Theorem

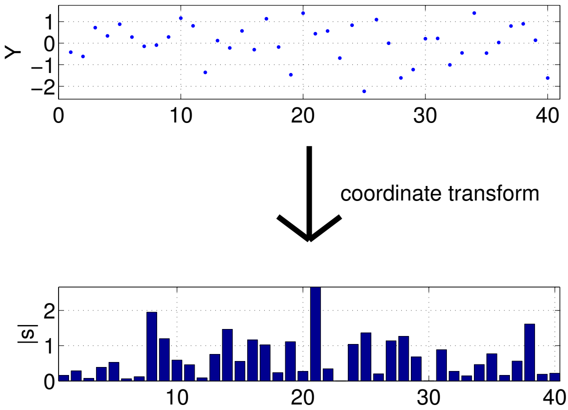
Let $g(x) = \sum_{i=1}^{\infty} \alpha_i \lambda_i \psi_i(x)$, $\quad G = (g(X_1), \ldots, g(X_n))$ Then, with high probability,

$$\frac{1}{\sqrt{n}} |u_i{}^\top G| < 2 l_i a_r c_i (1 + O(rn^{-1/4})) + r a_r \Lambda_r O(1) + T_r + \sqrt{A T_r} O(n^{-1/4}) + r a_r \sqrt{\Lambda_r} O(n^{-1/2}),$$

where

c_i:	measures size of the eigenvalue cluster around l_i
$a_r = \sum_{i=1}^{r} \lvert\alpha_i\rvert$:	measure for size of the first r components
Λ_r:	sum of all eigenvalues smaller than λ_r
A:	supremum norm of g
T_r:	error of projecting g to the space spanned by the first r eigenfunctions

The location of zero-mean noise



Since $\mathbf{U}^\top$ is a random rotation, noise does not change its shape under the coordinate transformation.

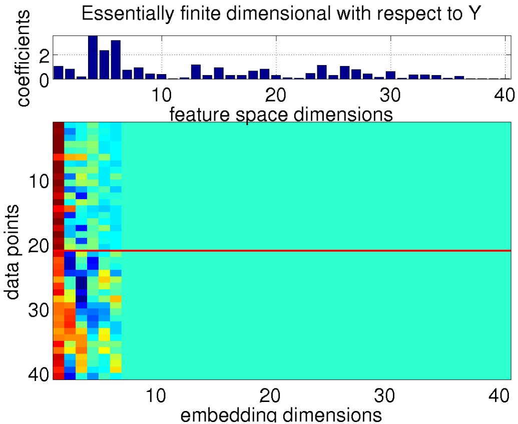
Summary

We see that also the information about Y is contained in the leading kernel PCA coefficients (if the kernel matches the problem)

- Relation between Y and kernel PCA components analyzed by scalar products $s_i = {u_i}^\top Y$.
- Decompose Y into informative part G and noise ε.
- Informative part G is contained in the first few directions.
- Noise is evenly spread over everything.

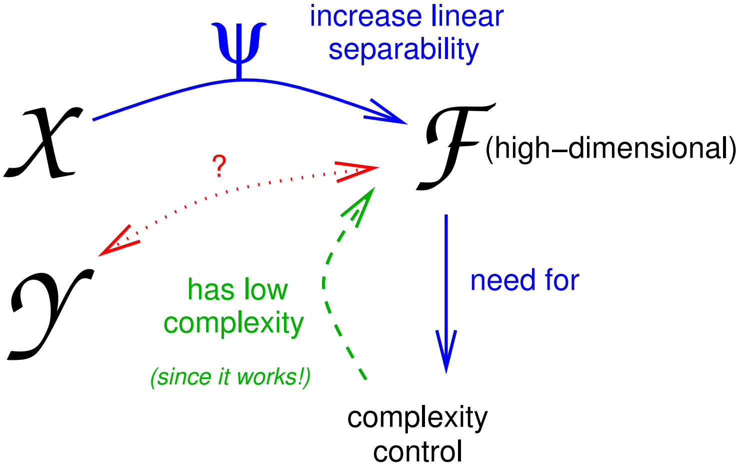
$\rightsquigarrow$ A good kernel optimally increases the power of linear discriminant functions while keeping the dimensionality low.

Example



Information is contained in leading PCA directions

The Extended Standard Picture (revisited)



Overview

- Estimating the relevant dimensionality on a data set.
- Analyze fit between kernel and data set.
- Use criterion for model selection.

Some reminders

$X_1, \ldots, X_n \in \mathbb{R}^d$ objects

$Y_1, \ldots, Y_n \in \mathbb{R}$ labels

$\mathbf{K} = k(X_i, X_j)$ kernel matrix

$\mathbf{K} u_i = l_i u_i$ eigenvectors and eigenvalues

u_i kernel PCA components

l_i kernel PCA values (variances)

$s_i = u_i^\top Y$ kernel PCA coefficients of Y

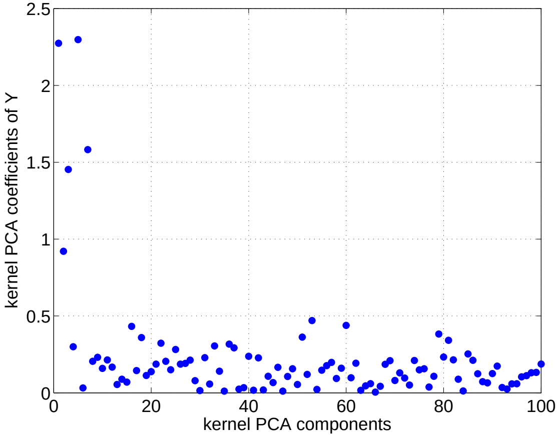
Properties of Kernel PCA Quantities

u_i become increasingly complex.

l_i decay quickly.

${u_i}^\top Y$ leading coefficients contain relevant information, superimposed noise-floor.

$\rightsquigarrow$ estimating the relevant dimensionality.



Idea: fit a two-component model to the s_i!

Fitting the Two-Component Model

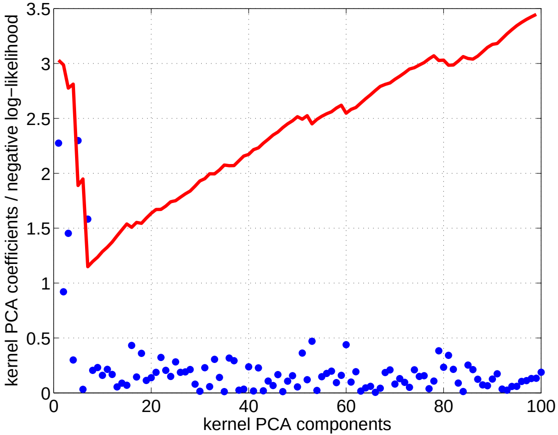
Assumption:

$$s_i \sim \begin{cases} \mathcal{N}(0, \sigma_1^2) & 1 \leq i \leq d \\ \mathcal{N}(0, \sigma_2^2) & d < i \leq n \end{cases}$$

The negative log-likelihood is proportional to

$$-\log \ell(d) \sim \frac{d}{n} \log \frac{1}{d} \sum_{i=1}^{d} s_i^2 + \frac{n-d}{n} \log frac1n - d \sum_{i=d+1}^{n} s_i^2.$$

$\rightsquigarrow$ choose d which minimizes $-\log \ell(d)$.


kernel PCA coefficients / negative log−likelihood
kernel PCA components
0
0.5
1
1.5
2
2.5
3
3.5
0
20
40
60
80
100

Appliation: Denoising the Labels

Project Y to leading d kernel PCA components

$$\hat{Y} = \sum_{i=1}^{d} u_i {u_i}^\top Y.$$

Amounts to estimating the relevant information vector $\hat{Y} \approx G$.

Theorem

If $d \to \infty$ and $dn^{-1/4} = O(1)$, then $\hat{Y} \to G$.

Application: Estimating the Noise Level

Since $Y = G + \varepsilon$, the noise ε is simply $Y - Y'$.

Computing using appropriate loss function L

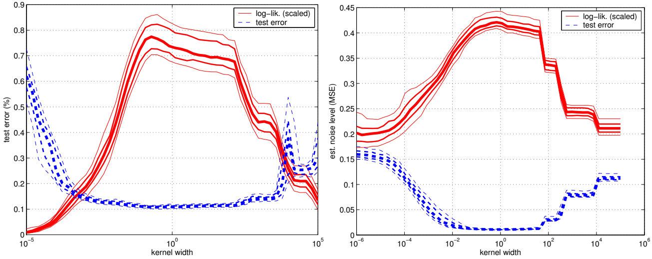
$$\hat{\varepsilon} = \frac{1}{n} \sum_{i=1}^{n} L(Y_i, \hat{Y}_i).$$

(Usually, this estimate is a bit too optimistic)

Application: Model Selection

Idea: Use kernel which separates noise from data best.

$\rightsquigarrow$ choose kernel such that log-likelihood value at $\hat{d}$ is maximal.



classification / regression

Benchmark Data Sets

data set	dim	dim (cv)	est. error rate	kPCR	KRR	SVM
banana	24	26	8.8 ± 1.5	11.3 ± 0.7	10.6 ± 0.5	11.5 ± 0.7
breast-cancer	2	2	25.6 ± 2.1	27.0 ± 4.6	26.5 ± 4.7	26.0 ± 4.7
diabetis	9	9	21.5 ± 1.3	23.6 ± 1.8	23.2 ± 1.7	23.5 ± 1.7
flare-solar	10	10	32.9 ± 1.2	33.3 ± 1.8	34.1 ± 1.8	32.4 ± 1.8
german	12	12	22.9 ± 1.1	24.1 ± 2.1	23.5 ± 2.2	23.6 ± 2.1
heart	4	5	15.8 ± 2.5	16.7 ± 3.8	16.6 ± 3.5	16.0 ± 3.3
image	272	368	1.7 ± 1.0	4.2 ± 0.9	2.8 ± 0.5	3.0 ± 0.6
ringnorm	36	37	1.9 ± 0.7	4.4 ± 1.2	4.7 ± 0.8	1.7 ± 0.1
splice	92	89	9.2 ± 1.3	13.8 ± 0.9	11.0 ± 0.6	10.9 ± 0.6
thyroid	17	18	2.0 ± 1.0	5.1 ± 2.1	4.3 ± 2.3	4.8 ± 2.2
titanic	4	6	20.8 ± 3.8	22.9 ± 1.6	22.5 ± 1.0	22.4 ± 1.0
twonorm	2	2	2.3 ± 0.7	2.4 ± 0.1	2.8 ± 0.2	3.0 ± 0.2
waveform	14	23	8.4 ± 1.5	10.8 ± 0.9	9.7 ± 0.4	9.9 ± 0.4

kPCR: (kernel) least-squares on the denoised data
KRR: kernel ridge regression
SVM: support vector machines

Benchmark Data Sets: Categorizing Data Sets

	low noise	high noise
low dimensional	banana, thyroid, waveform	breast-cancer, diabetis flare-solar, german heart, titanic
high dimensional	image, ringnorm	splice

Application: Kernel Design for Splice Site Detection

```
AAACAAATAAGTAACTAATCTTTTAGGAAGAACGTTTCAACCATTTTGAG
AAGATTAAAAAAAAACAAATTTTTAGCATTACAGATATAATAATCTAATT
CACTCCCCAAATCAACGATATTTTAGTTCACTAACACATCCGTCTGTGCC
TTAATTTCACTTCCACATACTTCCAGATCATCAATCTCCAAAACCAACAC
TTGTTTTAATATTCAATTTTTTACAGTAAGTTGCCAATTCAATGTTCCAC
CTGTATTCAATCAATATAATTTTCAGAAACCACACATCACAATCATTGAA
TACCTAATTATGAAATTAAAATTCAGTGTGCTGATGGAAACGGAGAAGTC
```

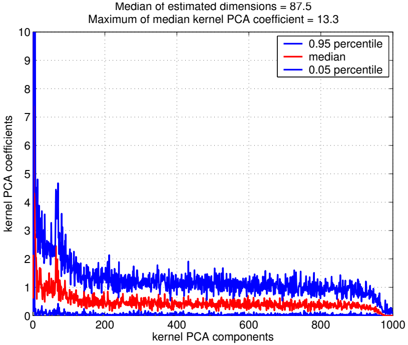
Genes are not encoded in one piece on the DNA, but in multiple parts.

Splice sites indicate where a coding region ends.

First, the whole protein sequence is built from the DNA, then special enzymes “cut out” the non-coding regions based on the splice cites.

Naive Encoding

Aminoacid	Encoded as
A	0
C	1
G	2
T	3



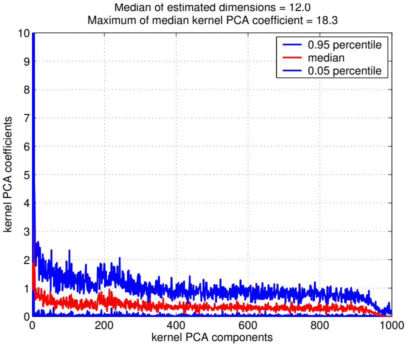
Dimensionality 87, test error $12.9 \pm 0.9\%$.

Using an rbf kernel, over 100 resamples of the data.

Main problem: A, C appear more similar than A, T.

A Better Encoding

Aminoacid	Encoded as
A	(0, 0, 0, 1)
C	(0, 0, 1, 0)
G	(0, 1, 0, 0)
T	(1, 0, 0, 0)



Dimensionality 11, test error $7.6 \pm 0.7\%$.

All aminoacids are comparably far from one another. But only fixed positions are comapred.

A Domain Specific Kernel: Weighted Degree Kernel

Weighted degree kernel is defined as

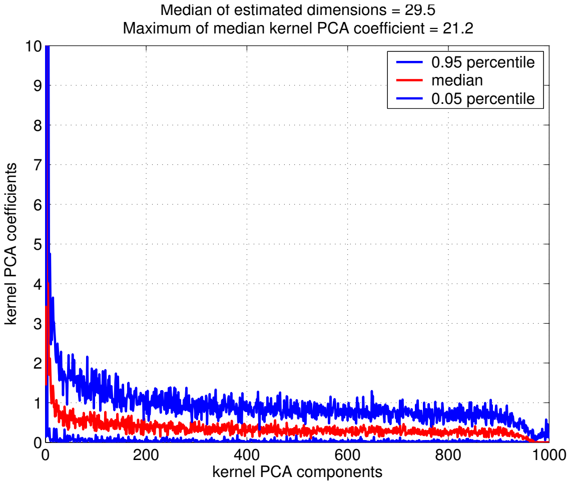
$$k(x, x') = \sum_{j=1}^{d} w_i \sum_{i=1}^{N-d} 1_{\{u_{j,i}(x)=u_{j,i}(x')\}}$$

with:

$$u_{j,i}(x) = x_i x_{i+1} \dots x_{i+j-1} \quad \text{(subword of length } j \text{ starting at } i\text{)}$$

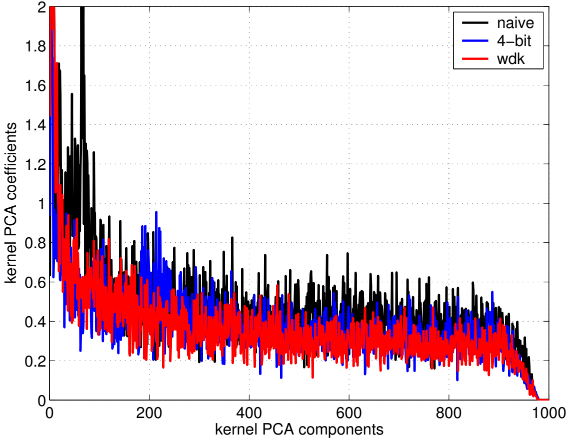
$$w_j = d - j + 1 \quad \text{(longer matches get lower weights)}$$

A Domain Specific Kernel: Weighted Degree Kernel



Dimensionality 29, test error $5.5 \pm 0.7\%$

The Three Spectra Compared



Summary

- Relevant dimension can be estimated well.
- Permits to denoise the data, estimate the noise level.
- Can also be used for model selection.
- E.g. on `splice` data set, better encoding and better kernel lead to better performance, also visible from the relevant dimensionality.

References

M.L. Braun
Accurate bounds for the eigenvalues of the kernel matrix
Journal of Machine Learning Research, Vol. 7(Nov) 2303-2328, 2006.

G. Blanchard, O. Bousquet, L. Zwald
Statistical Properties of Kernel Principal Component Analysis
To appear, Machine Learning, 2006

M.L. Braun, J. Buhmann, K.-R. Müller
Denoising and Dimension Reduction in Feature Space
NIPS 2006

G. Rätsch and S. Sonnenburg
Accurate Splice Site Prediction for Caenorhabditis Elegans
Pages 277-298, MIT Press series on Computational Molecular Biology. 2004.