Computational Pattern Analysis and Statistical Learning

Lecture 3: patterns in numbers and vectors

Tijl De Bie, Konstantin Tretyakov
(Largely based on joint work with Nello Cristianini and John Shawe-Taylor)

Tartu, Estonia

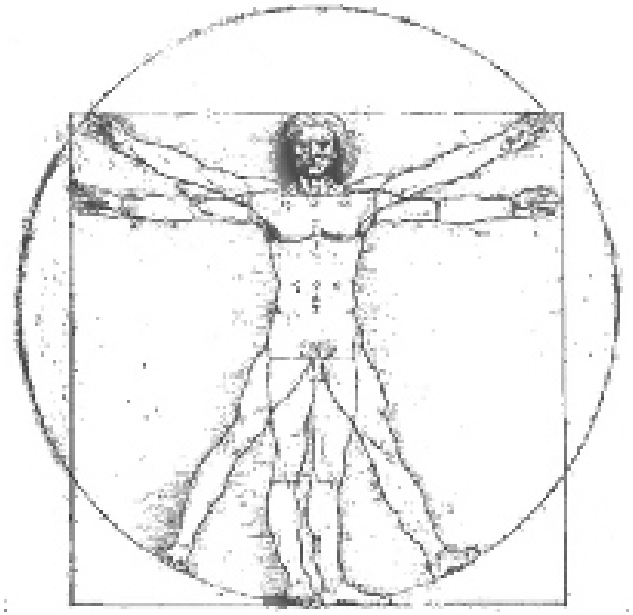
November 2006

1. Lecture 3A: Patterns in numbers
   - The importance of numbers
   - Physical laws
   - Other laws in nature
   - Types of patterns in vector spaces
2. Lecture 3B: Patterns in vector spaces
   - Linear regression: least squares and ridge regression
   - Your first kernel method: kernel ridge regression
   - Application of kernel ridge regression
3. Wrap-up Lecture 3

Historical sketch

- Research was traditionally done in a symbolic / qualitative / high level way

- So was the understanding of the world, of the 'patterns' in it
- Leonardo Da Vinci did not do much mathematics, the way we understand it today!



Historical sketch

- Soon after Da Vinci, clever minds started realising a more solid base is needed
  - Galileo $\longrightarrow$ law of acceleration of falling bodies, period of a swing
  - Kepler $\longrightarrow$ Kepler's laws
  - ...
- To quote Galileo: "The language of God is mathematics"
- This was a revolution... Einstein called Galileo "the father of modern science"

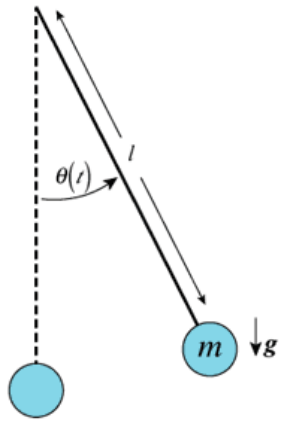
Historical sketch

- It is striking how fast this language was adopted to describe all kinds of physical phenomena
- Unprecedented in terms of versatility, verifiability, and accuracy
- But it was not restricted to physics: today used in all branches of physics, chemistry, economics, business, marketing, data mining,...
- Many of the mathematical techniques are designed to deal with patterns in data

Galilei and the lamp

- Looking at the lamp in the Cathedral of Pisa...
- ...wondered how the swinging of the lamp could be described

- System parameters:
  - Pendulum length
  - Mass of the body at the pendulum
  - Size of the swing



- Suspected that these must explain the period of the swinging
- In the data gathered in many experiments, he found a very *stable* relation:

$$T \propto \sqrt{L}$$

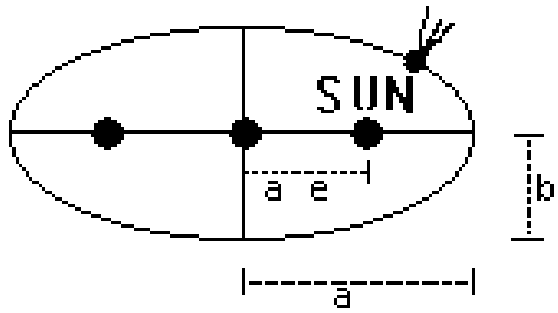
Kepler's laws

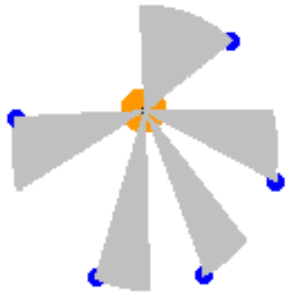
- Kepler was interested in the description of the planet/sun/moon's behaviour
- He could rely on extensive data gathered by Tycho Brahe
- Discovered a multitude of relations that are expressed in mathematical terms

Kepler's laws

- Three of these laws are now known as the 3 laws of Kepler:

1. The orbit of a planet/comet about the sun is an ellipse with the sun's center of mass at one focus



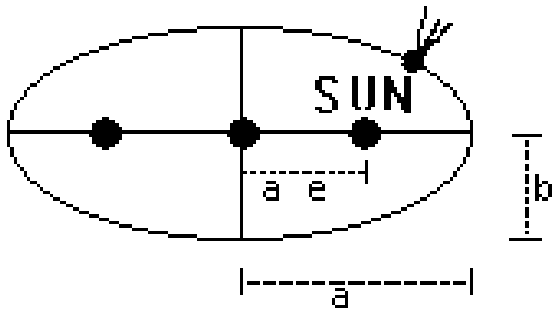


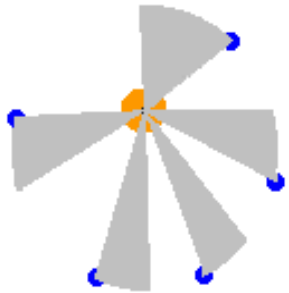
The areas of all triangles are the same size -Kepler's law of Equal Areas -

Kepler's laws

- Three of these laws are now known as the 3 laws of Kepler:

1. The orbit of a planet/comet about the sun is an ellipse with the sun's center of mass at one focus
2. A line joining a planet/comet and the Sun sweeps out equal areas in equal intervals of time



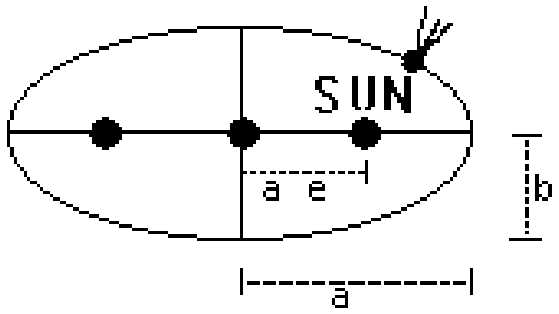


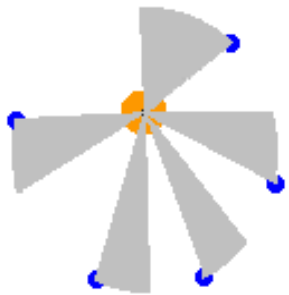
The areas of all triangles are the same size
-Kepler's law of Equal Areas -

Kepler's laws

- Three of these laws are now known as the 3 laws of Kepler:

1. The orbit of a planet/comet about the sun is an ellipse with the sun's center of mass at one focus
2. A line joining a planet/comet and the Sun sweeps out equal areas in equal intervals of time
3. The squares of the periods of the planets are proportional to the cubes of their semimajor axes





The areas of all triangles are the same size
-Kepler's law of Equal Areas -

'Secondary' laws

- Physical laws are supposed to be nearly exact (within current experimental accuracies)
- Somehow fundamental
- Other laws / patterns can be derived from these fundamental axiomatic laws
- Or they can be discovered from data $\longrightarrow$ pattern analysis!

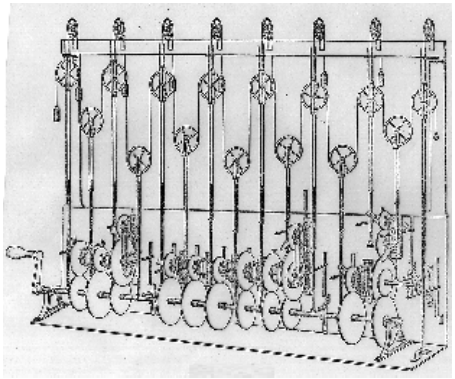
Approximate laws

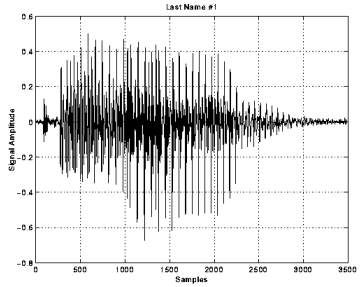
- Today, with lots of data available to analyse, we often do not search for causes or explanations
- We analyse data in search for patterns, without trying to understand those in detail – would be way too complicated!
- We just want to exploit them
- Very very often, these patterns will be expressed in terms of mathematics, just like for physical laws

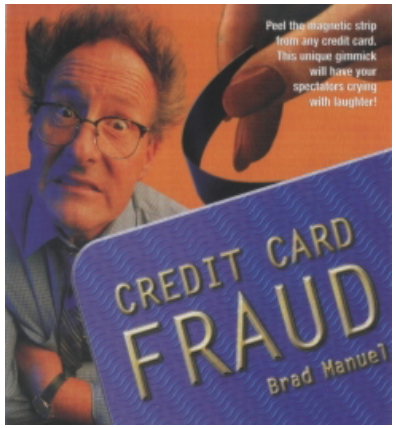
Approximate laws

Examples:

- The tides have a very periodic behaviour – physics teaches us which periods (moon, sun,...), and data analysis teaches us which amplitudes
- Automatic speech recognition
- SPAM filtering
- Credit card fraud detection
- ...







Approximate laws

- Clearly, most of these patterns cannot be described in terms of a single, one-dimensional measurement
- Example: SPAM filtering would take into account counts of certain words

- René Descartes invented 'Cartesian coordinates', to describe 'positions' and 'extensions' of real-life objects $\longrightarrow$ vector representation of data
- Patterns in vector spaces!

Data as a set of vectors

- In the remainder of this course, data presents itself usually as a set of data objects (e.g. measurements):

$$X = \{x_1, x_2, ..., x_n\}$$

  To each object x, a vector $\mathbf{x}$ (note: boldface) corresponds
- Statistical assumption: x_i are *i.i.d.*
- Then, we'll mostly work on averaging pattern functions, of the form

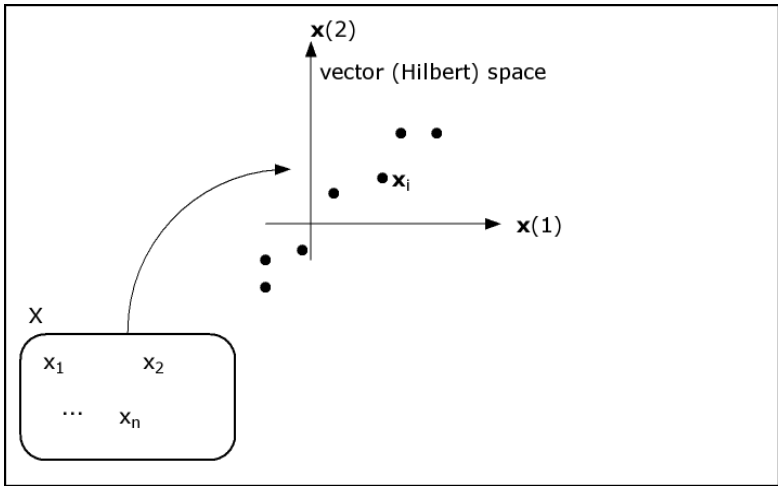
$$\pi(X) = \frac{1}{n} \sum_{i=1}^{n} g_\pi(x_i)$$

Data as a set of vectors

(Nearly) all data can be represented in vector form

- Inherently multidimensional data (of course...), such as Galileo's measurements of length, mass, swing size, and period
- Each measurement represented by a 4-dimensional vector

$$\mathbf{x} = \begin{pmatrix} \mathbf{x}(1) \\ \mathbf{x}(2) \\ \mathbf{x}(3) \\ \mathbf{x}(4) \end{pmatrix} \begin{matrix} \longrightarrow \text{ length} \\ \longrightarrow \text{ mass} \\ \longrightarrow \text{ swing size} \\ \longrightarrow \text{ period} \end{matrix}$$



Data as a set of vectors

- Text (e.g. bag of words)
- Sentence i is x_i
- E.g. $x_i =$"This is a sentence containing the words: this, and, a, and and"
- Vocabulary: {a, and, containing, is, sentence, the, this, words}

- Vector representation:

$$\mathbf{x}_i = \begin{pmatrix} 2 \\ 3 \\ 1 \\ 1 \\ 1 \\ 1 \\ 2 \\ 1 \end{pmatrix}$$

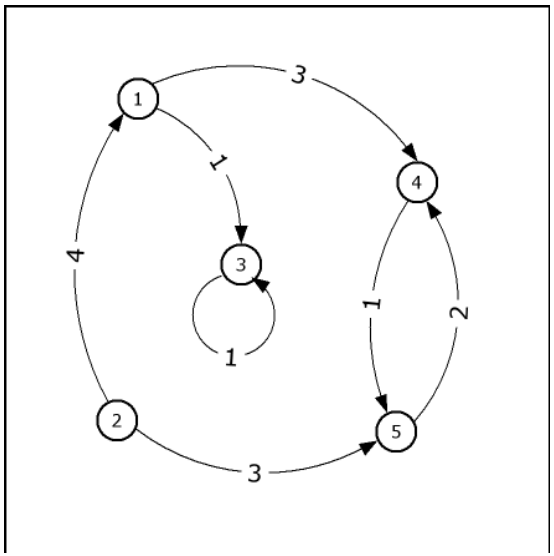
Data as a set of vectors

- x_i = a time series
- Can be represented as:
  - the time series itself in vector form
  - it's FFT (frequency spectrum)
  - wavelet transform
  - ...

Data as a set of vectors

- x_i = a node in a graph
- E.g., 'weighted adjacency vectors'
- Other options exist...

$$\mathbf{x}_1 = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 3 \\ 0 \end{pmatrix}, \mathbf{x}_2 = \begin{pmatrix} 4 \\ 0 \\ 0 \\ 0 \\ 3 \end{pmatrix}, \ldots, \mathbf{x}_5 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 2 \\ 0 \end{pmatrix}$$



Data as a set of vectors

- Quite trivially, if x is a vector itself, we can use it directly...
- Or, we can use a transformation of it, for example, with x having two coordinates, $x(1)$ and $x(2)$:

$$\mathbf{x} = \begin{pmatrix} x(1)^2 \\ x(1)x(2) \\ x(2)^2 \end{pmatrix}$$

Data as a set of vectors

- In this way we can then represent nonlinear patterns as linear patterns in the new representation
- With

$$\mathbf{x} = \begin{pmatrix} x(1)^2 \\ x(1)x(2) \\ x(2)^2 \end{pmatrix}$$

the expressions like $(x(1) + x(2))^2 = 1$ can be written in terms of an inner product in the vector space used:

$$\mathbf{x}' \begin{pmatrix} 1 \\ 2 \\ 1 \end{pmatrix} = x(1)^2 + 2x(1)x(2) + x(2)^2 = 1$$

- Any other transformation could be used, in order to find nonlinear patterns in the data... (more on this later!)

Averaging pattern functions

- Rest of course: pattern analysis on a set of objects $X = \{x_1, x_2, ..., x_n\}$ (measurements, sentences, nodes,...)
- $\forall x \in \mathcal{X}, \exists$ a vector representation $\mathbf{x}$ (aka *feature vector*)
- Total 'data set' summarised in data matrix $\mathbf{X} = \begin{pmatrix} \mathbf{x}_1' \\ \mathbf{x}_2' \\ \vdots \\ \mathbf{x}_n' \end{pmatrix}$
- Pattern functions will be averaging pattern functions $\pi(X) = \frac{1}{n} \sum_{i=1}^{n} g_\pi(x_i)$

Averaging pattern functions

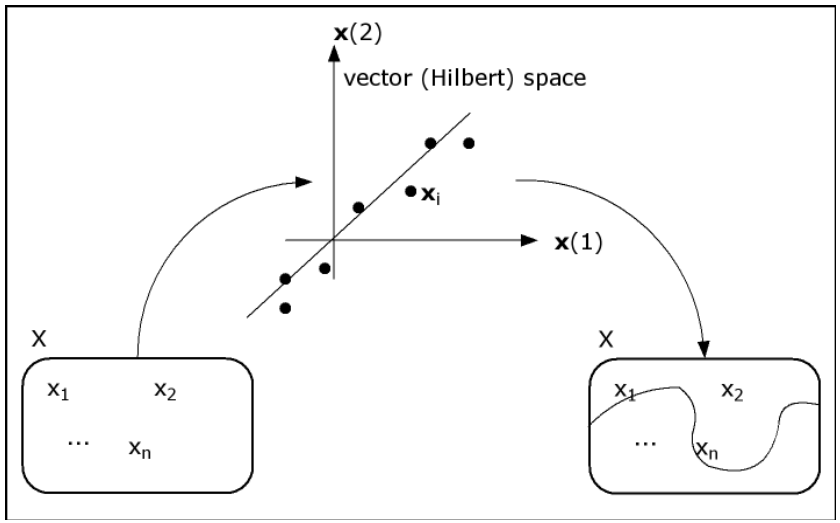
- Important fact: for averaging pattern functions $\pi(X) = \frac{1}{n}\sum_{i=1}^{n} g_\pi(x_i)$ from additivity of the expectation operator and assuming i.i.d. data:

$$\begin{aligned} E\{\pi(\underline{X})\} &= \frac{1}{n}\sum_{i=1}^{n} E\{g_\pi(\underline{x}_i)\} \\ &= E\{g_\pi(\underline{x})\} \end{aligned}$$

- This is relevant in studying stability and significance... (we will see this later in examples)

Using vector representations

- Usually (for convenience), $\pi(X)$ will be written in terms $\mathbf{X}$
- Then we can do pattern analysis in the vector space
- Corresponds to patterns in the object space
  - Sometimes obviously so, such as for the experimental measurements
  - Sometimes less obviously, such as for the graph nodes



Using vector representations

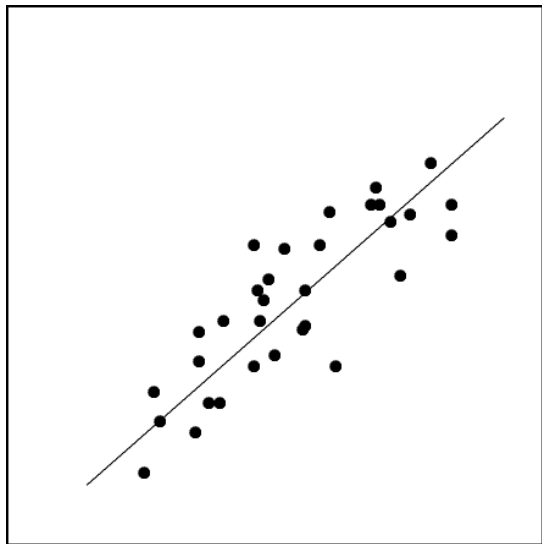
- Admittedly, there may be issues to deal with:
  1. There is not always a unique best representation – which one to choose?
  2. Sometimes it is not convenient to write out the vector representation explicitly (e.g. text with large vocabulary, or all cross-products between the $x(i)$ up to order d,...)
- We will see ways how to deal with these issues...
- The remainder of this class will be about patterns in vector spaces

- Many pattern analysis prototype problems have been defined in vector spaces
- A few examples here, only qualitatively...

'Unsupervised learning'

- Discover some type of structure in the data

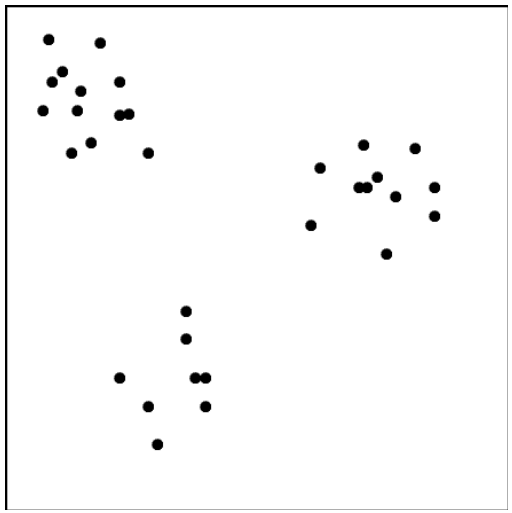
- manifolds / lower dimensional subspaces?
- Find a subspace that summarises the data well
- Or: find a limited number of sources of variation (factors) to account for as much as possible in the data variation



'Unsupervised learning'

- Discover some type of structure in the data

- clusters / groups?
- Find a limited number of prototypes, such that each data object is similar to exactly one of them



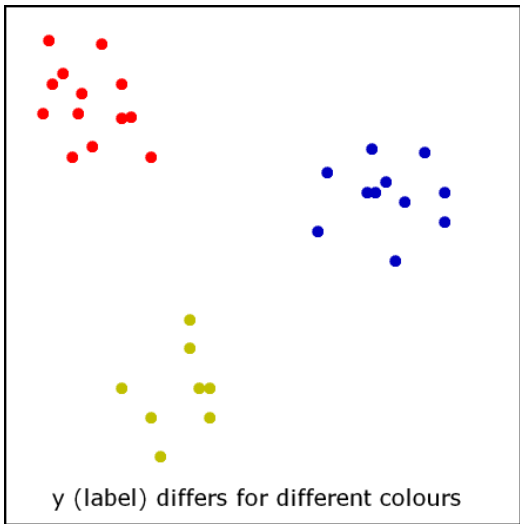
'Supervised learning'

- Discover some relation between the data vectors and some label y_i associated to it
- The data consists of (data object, label) pairs: $Z = \{(x_1, y_1), (x_2, y_2), \ldots (x_n, y_n)\}$
- Label vector $\mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}$, data matrix $\mathbf{X} = \begin{pmatrix} \mathbf{x}_1' \\ \mathbf{x}_2' \\ \vdots \\ \mathbf{x}_n' \end{pmatrix}$

'Supervised learning'

- Discover some relation between the data vectors and some *discrete class label* y_i associated to it

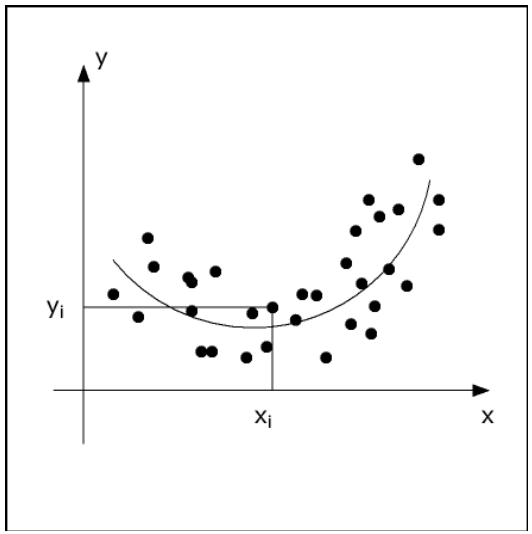
- classification: how does the data vector determine its class?



'Supervised learning'

- Discover some relation between the data vectors and some *real-valued label* y_i associated to it

- regression: determine the value of a function over (part of) the vector space
- Here shown is nonlinear regression(we will first focus on linear regression)



Supervised versus unsupervised?

- Distinction supervised/unsupervised is rather arbitrary...
- Semi-supervised? $Z = \{(x_1, y_1), ..., (x_m, y_m), x_{m+1}, ...x_{m+n}\}$
- We can regard supervised problems as supervised with vectors $\mathbf{z}_i = \begin{pmatrix} \mathbf{x}_i \\ y_i \end{pmatrix}$
- Many other problems for which this distinction is less clear can be imagined (we will discuss one)
- Still, this distinction is often made

Least squares regression

- Training data:
  $Z = \{z_1, z_2, ..., z_n\} = \{(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)\}$
- Find a function that maps the data X to the labels $\mathbf{y}$
- Different motivations (see Konstantins preparatory lectures):
  - maximum likelihood (assuming Gaussian noise)
  - minimise risk – the quadratic cost function
- We will recapitulate the result here, without going too deeply in the motivations

Least squares regression

- Task: approximate y_i as a linear function of $\mathbf{x}_i$
- In terms of a weight vector $\mathbf{w}$, this means: $y_i \approx \mathbf{x}_i'\mathbf{w}$, or, $\|y_i - \mathbf{x}_i'\mathbf{w}\| \approx 0$
- Pattern function is parameterised by $\mathbf{w}$ (note the $-$ sign):

$$\pi_{\mathbf{w}}(Z) = \frac{1}{n}\sum_{i=1}^{n} g_{\mathbf{w}}(z_i) = -\frac{1}{n}\sum_{i=1}^{n}\left(y_i - \mathbf{x}_i'\mathbf{w}\right)^2 = -\frac{1}{n}\|\mathbf{y} - \mathbf{X}\mathbf{w}\|^2$$

- Formal pattern recognition problem:

$$\max_{\mathbf{w}} \pi_{\mathbf{w}}(Z) \Leftrightarrow \max_{\mathbf{w}} -\frac{1}{n}\|\mathbf{y} - \mathbf{X}\mathbf{w}\|^2 \Leftrightarrow \min_{\mathbf{w}} \|\mathbf{y} - \mathbf{X}\mathbf{w}\|^2$$

Least squares regression

- $\min_{\mathbf{w}} \|\mathbf{y} - \mathbf{X}\mathbf{w}\|^2$
- Easy to solve: $\|\mathbf{y} - \mathbf{X}\mathbf{w}\|^2 = \mathbf{y}'\mathbf{y} - 2\mathbf{y}'\mathbf{X}\mathbf{w} + \mathbf{w}\mathbf{X}'\mathbf{X}\mathbf{w}$ is a *convex function* in $\mathbf{w}$
- Convex means:
  $\forall c \in [0,1] : cf(\mathbf{w}_1) + (1-c)f(\mathbf{w}_2) \geq f(c\mathbf{w}_1 + (1-c)\mathbf{w}_2)$
- Property: at the minimum (and only there), the gradient is $\mathbf{0}$
- Compute gradient w.r.t. $\mathbf{w}$ and equate to $\mathbf{0}$:

$$2\mathbf{X}'\mathbf{X}\mathbf{w} - 2\mathbf{X}'\mathbf{y} = 0 \Leftrightarrow \mathbf{w} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y}$$

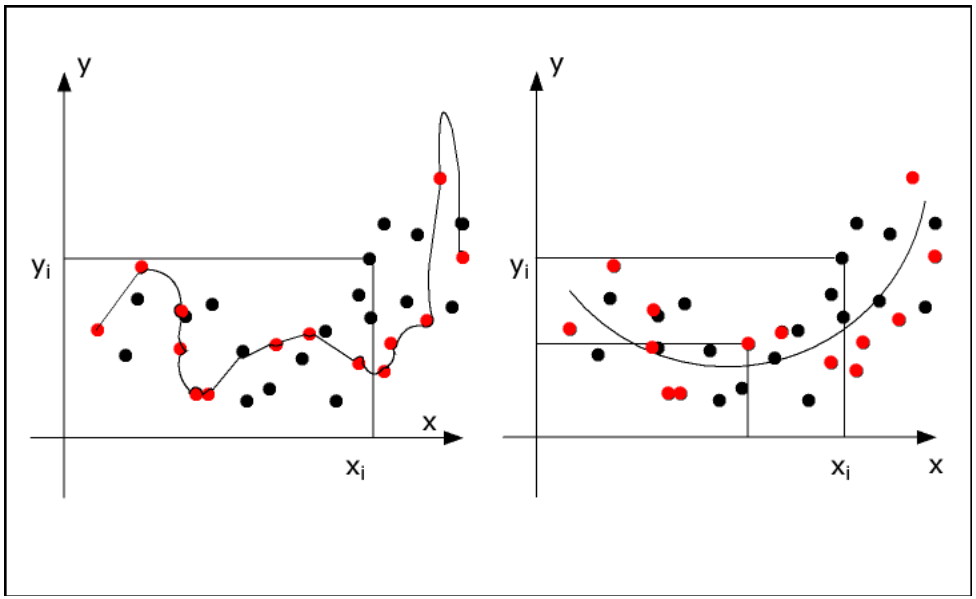
- In practice solved as a linear set of equations: $(\mathbf{X}'\mathbf{X})\mathbf{w} = \mathbf{X}'\mathbf{y}$

Least squares regression

- Works fine as long as $n >> d$
- Problems when this does not hold!
  - algebraically: underdetermined set of equations ($\mathbf{X}'\mathbf{X}$ singular), but more importantly,
  - statistically: too large pattern space, and hence too easy to find a strong pattern
    $\Rightarrow$ such a pattern will be unstable and insignificant
    $\Rightarrow$ for a test data point x, the difference $y - \mathbf{x}'\mathbf{w}$ may be large
    I.e., $E\left\{-g_{\mathbf{w}}\left(\underline{z}\right)\right\} = E\left\{\left(y - \mathbf{x}'\mathbf{w}\right)^2\right\}$ is large
    $\Rightarrow$ bad *generalisation*
- We will make this claim rigorous in a *later lecture*
- You will experience this in the *practical session*

Capacity control and ridge regression

- Instability in regression is known as overfitting '*overfitting*'
- Even though the pattern is strong (very good fit), it is instable, such that in expectation on data drawn from the distribution, the pattern is weak



- $\|\mathbf{y} - \mathbf{X}\mathbf{w}\|^2$ is small
- But, $E\left\{\left(\underline{y} - \underline{\mathbf{x}}'\mathbf{w}\right)^2\right\}$ is large...
- $\Rightarrow$ bad generalisation / overfitting

Capacity control and ridge regression

- Solution: reduce the pattern space! Impose a capacity constraint
- Capacity functional: take something easy (why not?), such as

$$C\left(\pi_{\mathbf{w}}\right) = \|\mathbf{w}\|^2$$

- Then, solve same pattern recognition problem, subject to capacity constraint (stratification formulation):

$$\begin{aligned} \min_{\mathbf{w}} \quad & \|\mathbf{y} - \mathbf{X}\mathbf{w}\|^2 \\ \text{s.t.} \quad & \|\mathbf{w}\|^2 \leq c \end{aligned}$$

- More common variant (where γ is known as the *regularisation parameter*):

$$\min_{\mathbf{w}} \|\mathbf{y} - \mathbf{X}\mathbf{w}\|^2 + \gamma \|\mathbf{w}\|^2$$

Capacity control and ridge regression

- Regularised pattern recognition problem:

$$\min_{\mathbf{w}} \|\mathbf{y} - \mathbf{X}\mathbf{w}\|^2 + \gamma \|\mathbf{w}\|^2$$

- Note: $\|\mathbf{w}\|^2$ is a *convex function* $\longrightarrow$ easy to solve!
- Equate gradient w.r.t $\mathbf{w}$ to 0, then:

$$2\left(\mathbf{X}'\mathbf{X} + \gamma\mathbf{I}\right)\mathbf{w} - 2\mathbf{X}'\mathbf{y} = 0 \Leftrightarrow \mathbf{w} = \left(\mathbf{X}'\mathbf{X} + \gamma\mathbf{I}\right)^{-1}\mathbf{X}'\mathbf{y}$$

- Note: a 'side-result' is that now a regular matrix $\mathbf{X}'\mathbf{X} + \gamma\mathbf{I}$ is inverted, instead of the potentially singular matrix $\mathbf{X}'\mathbf{X}$

Capacity control and ridge regression

- As pointed out, regularisation is particularly important for $d \not\leq n$, i.e. for large dimension
- Examples: the edges in the graph (where $d = n$), sentences in a text (often $d >> n$), DNA sequences (idem),...
- Note: these are exactly the cases where we do not want to write out the vectors explicitly, and where $\mathbf{X}'\mathbf{X}$ would be inconveniently large (size $d \times d$)
- We'll treat such cases now using a clever trick...

Kernel ridge regression

- Note:

$$2\left(\mathbf{X}'\mathbf{X}+\gamma\mathbf{I}\right)\mathbf{w}-2\mathbf{X}'\mathbf{y}=0\Leftrightarrow\mathbf{w}=\mathbf{X}'\cdot\left(\frac{1}{\gamma}\left(\mathbf{y}-\mathbf{X}\mathbf{w}\right)\right)$$

- Let's denote $\boldsymbol{\alpha}=\left(\frac{2}{\gamma}\left(\mathbf{y}-\mathbf{X}\mathbf{w}\right)\right)$, then

$$\mathbf{w}=\mathbf{X}'\boldsymbol{\alpha}=\sum_{i=1}^{n}\alpha_i\mathbf{x}_i$$

- The weight vector is a linear combination of the data points *(representer theorem)*
- Projection of a data point on the weight vector is a weighted sum of inner products:

$$\mathbf{x}'\mathbf{w}=\mathbf{x}'\mathbf{X}'\boldsymbol{\alpha}=\sum_{i=1}^{n}\alpha_i\left(\mathbf{x}'\cdot\mathbf{x}_i\right)$$

Kernel ridge regression

- Let's plug this in the equations (assuming that $\mathbf{X}\mathbf{X}'$ is full rank):

$$\begin{aligned} 2\left(\mathbf{X}'\mathbf{X}+\gamma\mathbf{I}\right)\mathbf{w}-2\mathbf{X}'\mathbf{y}=0 \quad &\Leftrightarrow \quad 2\left(\mathbf{X}'\mathbf{X}+\gamma\mathbf{I}\right)\mathbf{X}'\boldsymbol{\alpha}-2\mathbf{X}'\mathbf{y}=0 \\ &\Leftrightarrow \quad \mathbf{X}\left(\mathbf{X}'\mathbf{X}+\gamma\mathbf{I}\right)\mathbf{X}'\boldsymbol{\alpha}-\mathbf{X}\mathbf{X}'\mathbf{y}=0 \\ &\Leftrightarrow \quad \boldsymbol{\alpha}=\left(\mathbf{X}\mathbf{X}'+\gamma\mathbf{I}\right)^{-1}\mathbf{y} \end{aligned}$$

- Note: $\mathbf{X}\mathbf{X}'$ is a matrix containing the inner product $\mathbf{x}_i' \cdot \mathbf{x}_j$ at row i and column j
- Hence, also $\boldsymbol{\alpha}$ can be found using just inner products!

Kernel ridge regression

- $\boldsymbol{\alpha}$ can be found using just inner products $\mathbf{x}_i' \cdot \mathbf{x}_j$
- Projection of a new point on the weight vector can be done using just inner products

$$\mathbf{x}'\mathbf{w} = \mathbf{x}'\mathbf{X}'\boldsymbol{\alpha} = \sum_{i=1}^{n} \alpha_i \left(\mathbf{x}' \cdot \mathbf{x}_i\right)$$

- For $d \geq n$, $\mathbf{X}\mathbf{X}'$ is more convenient to work with than $\mathbf{X}'\mathbf{X}$
- This reformulation in terms of inner products can be extremely useful!
- Known as the *kernel trick*

Kernel ridge regression

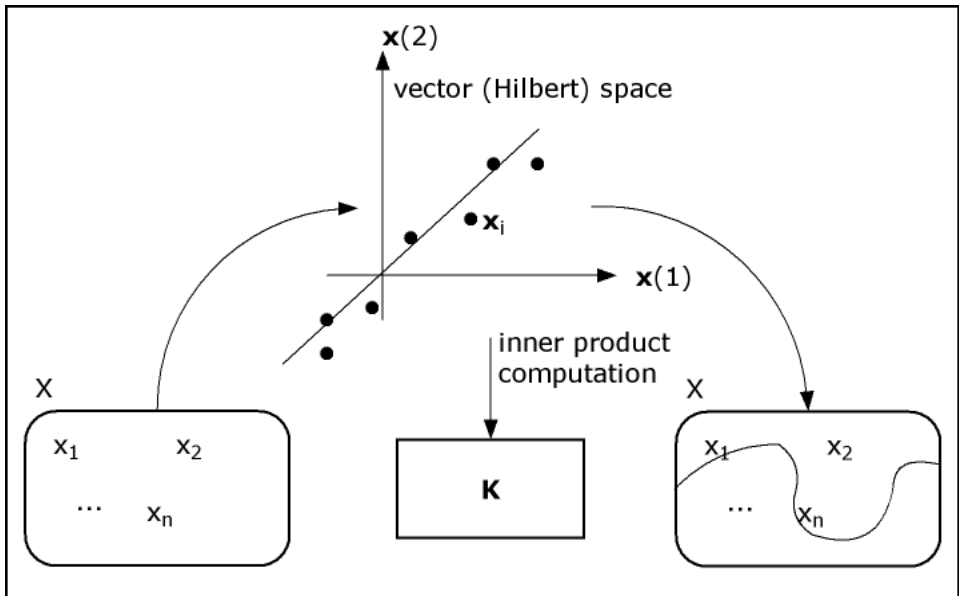
- Sometimes, it is even impossible to represent the feature vectors $\mathbf{x}$ explicitly, while it may be possible to compute inner products
- Then we can just use these inner products!
- In that case, these inner products are known as *kernel functions*
- Denoted as (with x_i and x_j data objects):

$$k\left(x_i, x_j\right) = \mathbf{x}_i' \cdot \mathbf{x}_j$$

- $\mathbf{K}$ denotes the matrix with $k\left(x_i, x_j\right)$ at row i and column j
- Kernel ridge regression: $\boldsymbol{\alpha} = \left(\mathbf{K} + \gamma \mathbf{I}\right)^{-1} \mathbf{y}$

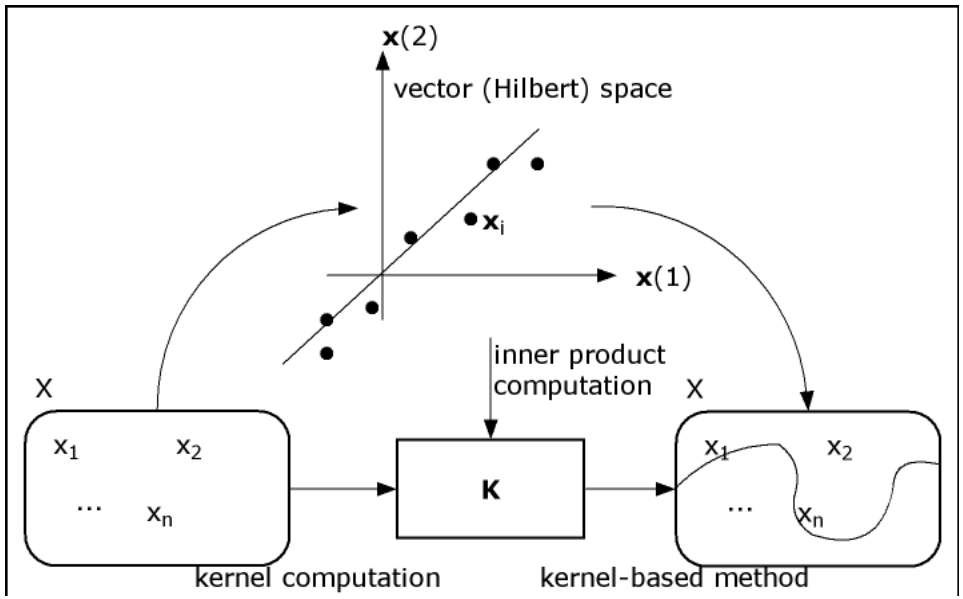
Kernel ridge regression

- I.e., one can compute the kernel matrix, containing the inner products between the feature vectors



Kernel ridge regression

- In this way, we can bypass the representation of the feature vectors themselves, and directly search for patterns



Kernel ridge regression

- When working with data objects that are themselves vectors, examples of such kernel functions are

$$
\begin{aligned}
k_{\text{RBF}}\left(x_i, x_j\right) &= \exp\left(-\frac{\left\|x_i - x_j\right\|^2}{2\sigma^2}\right) \\
k_{\text{linear}}\left(x_i, x_j\right) &= x_i' \cdot x_j \\
k_{\text{polynomial},d}\left(x_i, x_j\right) &= \left(x_i' \cdot x_j + 1\right)^d
\end{aligned}
$$

- By using the RBF kernel in the kernel ridge regression algorithm, one does linear regression in an infinite dimensional space that is hard to imagine...
- The resulting regression function is of the form $\sum_{i=1}^{n} \alpha_i k_{\text{RBF}}\left(x_i, x\right)$, i.e. highly nonlinear!

Kernel ridge regression

- Which functions $k : \mathcal{X} \times \mathcal{X} \rightarrow \mathcal{R}$ are valid kernel functions?
- I.e., which can be written (in principle) as an inner product between two mappings of the data points into a Hilbert space?

- Two properties of an inner product:
  - Symmetric
  - Positive semi-definite
- It turns out that these two properties are sufficient for a function k to be a kernel
- *Mercer's theorem*

Kernel ridge regression

- Hence, the kernel trick allows to do regression over discrete spaces, such as graph nodes, sentences, DNA sequences, etc
- What's more: it allows to do nonlinear regression by just solving a linear system of equations
- All this is statistically well-founded – we'll see more about that in a later lecture

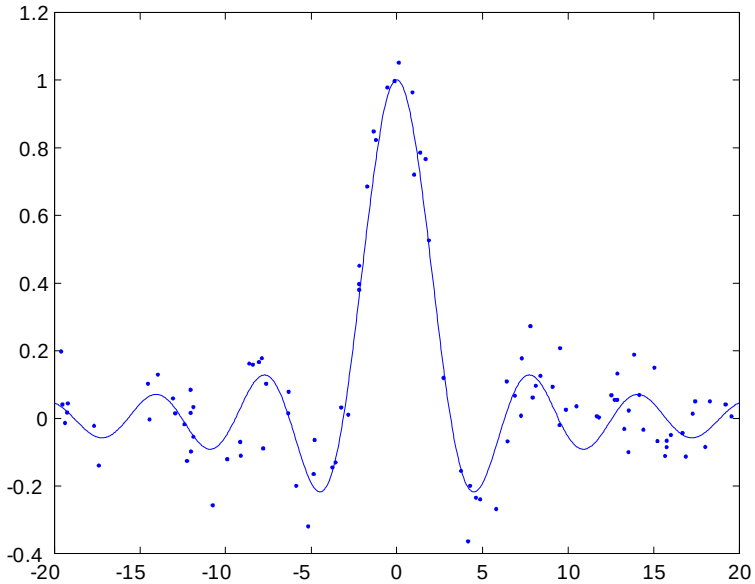
The noisy sinc function – the importance of regularisation

- Let's consider regression with an RBF kernel
- The vector space in which regression is carried out, then, is infinite dimensional
- Hence, the pattern space is huge
- Very hard to achieve *uniform convergence*, i.e. to achieve a statistically stable result!
- Hence, we reduced the capacity by means of the capacity functional $C(\pi_{\mathbf{w}}) = \|\mathbf{w}\|^2$

The noisy sinc function – the importance of regularisation

Let's consider the following example:

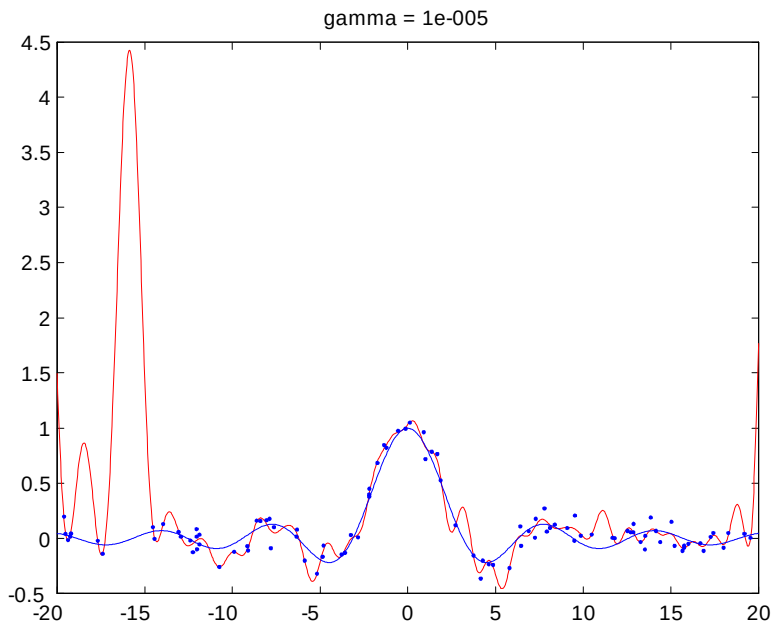
- Perform regression on noisy sinc data (note: $\text{sinc}(x) \triangleq \frac{\sin(x)}{x}$)
- x uniformly distributed in interval $[-20, 20]$
- $y = \text{sinc}(x) + n$, with n additive Gaussian noise (std 0.1)



The noisy sinc function – the importance of regularisation

Let's consider the following example:

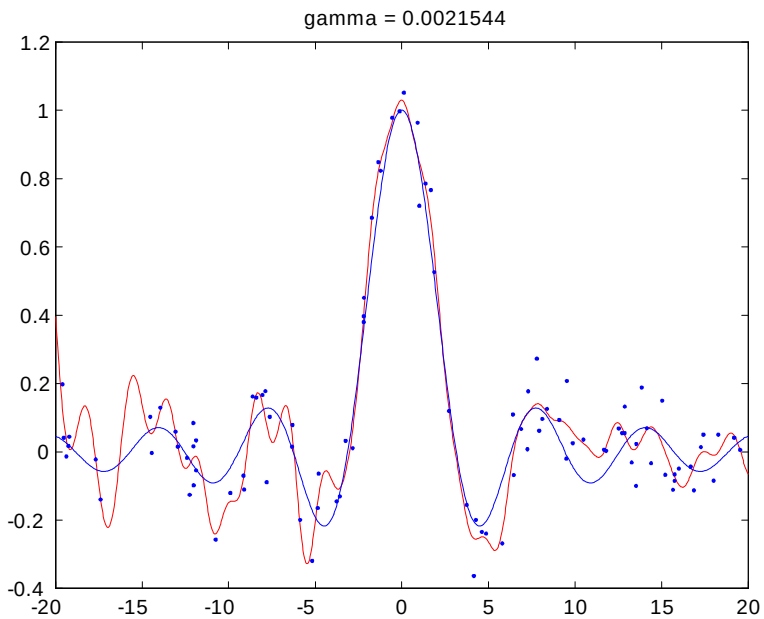
- Perform regression on noisy sinc data (note: $\text{sinc}(x) \triangleq \frac{\sin(x)}{x}$)
- x uniformly distributed in interval $[-20, 20]$
- $y = \text{sinc}(x) + n$, with n additive Gaussian noise (std 0.1)



The noisy sinc function – the importance of regularisation

Let's consider the following example:

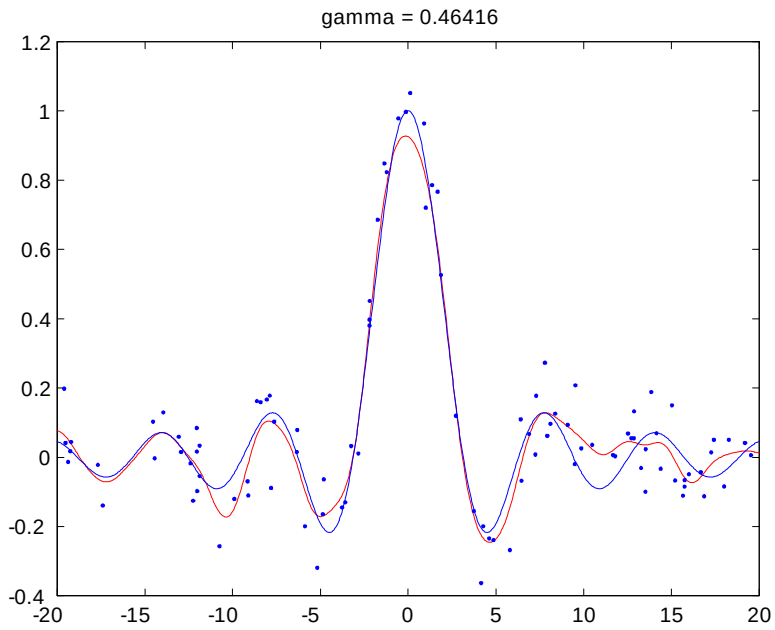
- Perform regression on noisy sinc data (note: $\text{sinc}(x) \triangleq \frac{\sin(x)}{x}$)
- x uniformly distributed in interval $[-20, 20]$
- $y = \text{sinc}(x) + n$, with n additive Gaussian noise (std 0.1)



The noisy sinc function – the importance of regularisation

Let's consider the following example:

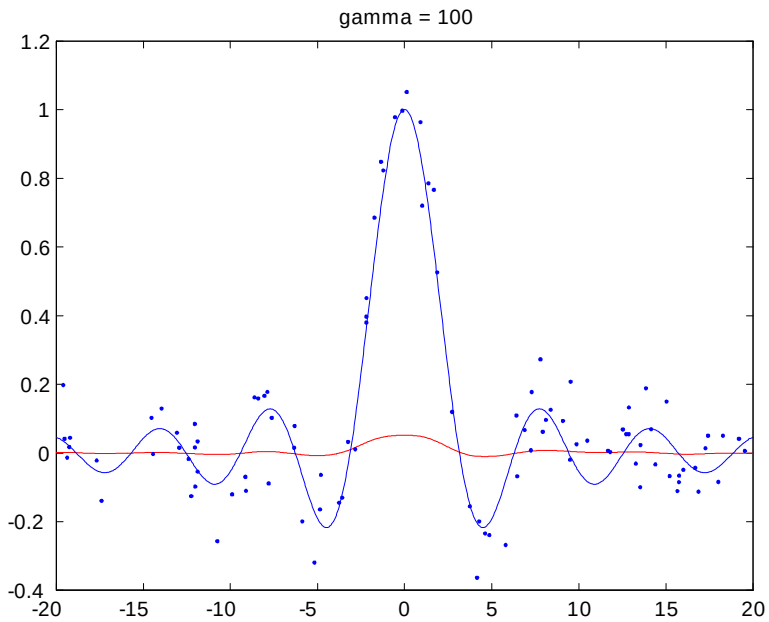
- Perform regression on noisy sinc data (note: $\text{sinc}(x) \triangleq \frac{\sin(x)}{x}$)
- x uniformly distributed in interval $[-20, 20]$
- $y = \text{sinc}(x) + n$, with n additive Gaussian noise (std 0.1)



The noisy sinc function – the importance of regularisation

Let's consider the following example:

- Perform regression on noisy sinc data (note: $\text{sinc}(x) \triangleq \frac{\sin(x)}{x}$)
- x uniformly distributed in interval $[-20, 20]$
- $y = \text{sinc}(x) + n$, with n additive Gaussian noise (std 0.1)



The noisy sinc function – the importance of regularisation

- Clearly, the choice of γ is critical
  - trades off bias versus variance
  - trades off pattern magnitude versus capacity
- How to decide?
  - Rules of thumb...
  - Based on a hold-out set – divide the given data set in a training set, and use the remaining part to 'test', i.e. to see how well the pattern actually matches data on which it was not trained.
  - If it matches well,
  - (This is a way to overcome the uniform convergence problem)

The noisy sinc function – the importance of regularisation

- Better strategy: cross-validation, e.g. 10-fold cross-validation
  - Partition the data set in 10
  - Train on the union of 9 of these subsets
  - Evaluate the pattern function on the 10th
- Do this 10 times, and average the test set result – this is the cross-validation performance
- Do this for each value of γ and pick the one with the best cross-validation performance

- Importance of language of mathematics – in fact a relatively new insight
- Importance of vector spaces
- Linear regression as an example, and first confrontation with the kernel trick (we'll see that again)