HPI Hasso Plattner Institut
Digital Engineering • Universität Potsdam

Introduction to Probabilistic Machine Learning

Ralf Herbrich

Linear Basis Function Models

Overview

1. Linear Basis Function Models
2. Modelling Data
   - Modelling Text
   - Modelling Images
3. Linear Algebra
   - Vector Spaces
   - Linear Mappings and Matrices
   - Matrix Derivatives
4. Maximum A Posterior Learning and (Regularized) Least Squares

Overview

1. **Linear Basis Function Models**
2. Modelling Data
   - Modelling Text
   - Modelling Images
3. Linear Algebra
   - Vector Spaces
   - Linear Mappings and Matrices
   - Matrix Derivatives
4. Maximum A Posterior Learning and (Regularized) Least Squares

Linear Basis Function Models

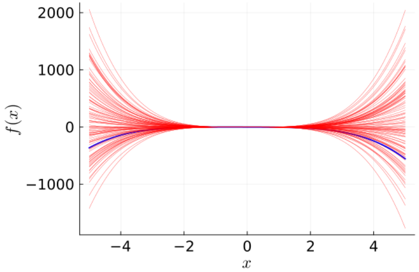
- **Linear Basis Function Models**. *Given an input space* $\mathcal{X}$ *and* D basis functions $\phi_j\colon \mathcal{X} \to \mathbb{R}$, *a* linear basis function model *is a function of the form*

$$f(x;\boldsymbol{w}) = w_0 + w_1 \cdot \phi_1(x) + w_2 \cdot \phi_2(x) + \cdots + w_D \cdot \phi_D(x)$$

- It's called a linear model, but the linearity is w.r.t. $\boldsymbol{w}$ not $x \in \mathcal{X}$!
- **Examples**: 4 basis function ($D = 4$) and 100 random parameters $\boldsymbol{w}$.

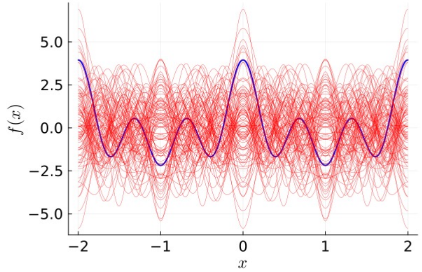
Polynomial Basis

$$\phi_j(x) = x^j$$



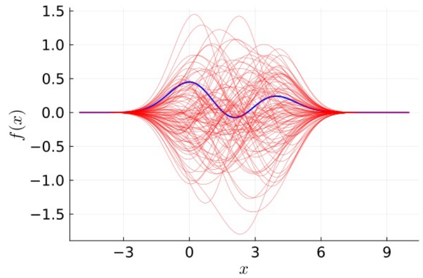
Fourier Basis

$$\phi_j(x) = \cos(\pi j \cdot x)$$



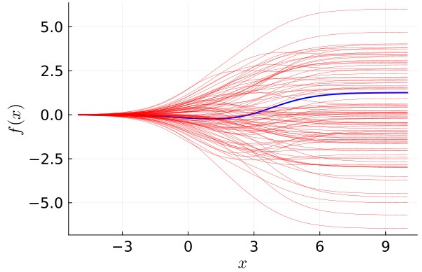
Gaussian Basis

$$\phi_j(x) = \mathcal{N}(x; j, 1)$$



Sigmoid Basis

$$\phi_j(x) = \frac{\exp(x-j)}{1+\exp(x-j)}$$

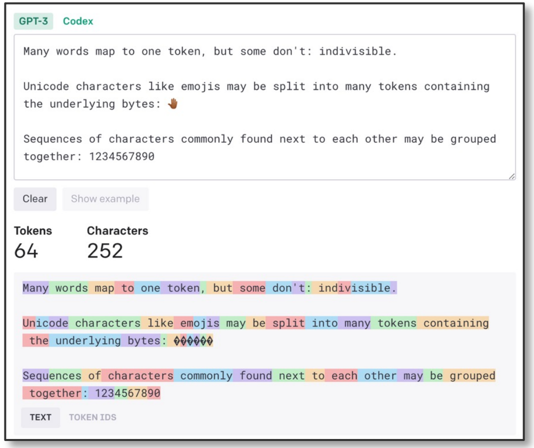


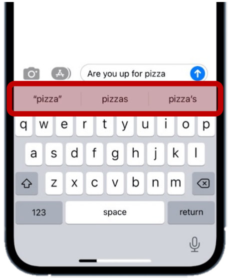
Overview

1. Linear Basis Function Models
2. **Modelling Data**
   - **Modelling Text**
   - Modelling Images
3. Linear Algebra
   - Vector Spaces
   - Linear Mappings and Matrices
   - Matrix Derivatives
4. Maximum A Posterior Learning and (Regularized) Least Squares

Modelling Text

- Text can be modelled at three levels of granularity:
  1. **Letters** ($\approx 10^2$ different letters in most alphabets)
  2. **Tokens** (10^3 to 10^4 different tokens in most alphabets)
  3. **Words** (10^5 to 10^6 different words in most languages)
- Modelling level of granularity depends on the application
  1. **Letters**: Compression algorithms for textual data
  2. **Tokens**: Sequence prediction for question-answering
  3. **Words**: Auto-correct function in smart keyboards





One-Hot Encoding

- Text is a sequence of text elements $s_1, s_2, s_3, \ldots$
- We often want to know the importance of each text element s_i to the target
  - **Example.** "$\underbrace{\boxed{\text{This}}}_{s_1}$ $\underbrace{\boxed{\text{product}}}_{s_2}$ $\underbrace{\boxed{\text{shipped}}}_{s_3}$ $\underbrace{\boxed{\text{fast}}}_{s_4}$ $\underbrace{\boxed{\text{but}}}_{s_5}$ $\underbrace{\boxed{\text{was}}}_{s_6}$ $\underbrace{\boxed{\text{disappointing}}}_{s_7}$" has negative sentiment

Text element that is likely the expression of bad sentiment

- **One-Hot Encoding**. *Given a dictionary S and a text element $s \in S$, a one-hot encoding $\boldsymbol{\phi}_{\mathrm{OHE}}(s)$ is an $|S|$-dimensional unit vector indexed by the elements of S that consists of 0s in all dimensions with the exception of a single 1 in the dimension indexed by s.*
  - **Example (ctd)**. If we assume the indices are $s_1, s_2, \ldots, s_7$ then

$$\boldsymbol{\phi}_{\mathrm{OHE}}(\text{"fast"}) = \begin{array}{l} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \end{array} \begin{array}{l} \leftarrow \text{This} \\ \leftarrow \text{product} \\ \leftarrow \text{shipped} \\ \leftarrow \text{fast} \\ \leftarrow \text{but} \\ \leftarrow \text{was} \\ \leftarrow \text{disappointing} \end{array}$$

- A linear model captures the *effect* of presence of a text element!

Efficient One-Hot Encoding

- In practice, the feature vector for one-hot encoded data is *never* explicitly computed because

$$\boldsymbol{w}^{\mathrm{T}}\boldsymbol{\phi}_{\mathrm{OHE}}(s) = \boxed{\sum_{i=1}^{|S|} w_i \cdot \phi_i(s)} = \boxed{w_{\mathrm{idx}(s)}}$$

$\mathcal{O}(|S|)$

$\mathcal{O}(\log_2|S|)$

- All we need is the inverse function mapping of an actual text element s

$$\mathrm{idx}(s) \coloneqq i \quad \Leftrightarrow \quad \phi_{\mathrm{OHE},i}(s) = 1$$

- Two common techniques for encoding a whole text
  - **Variable-length text** $s_1 s_2 \cdots$: Sum of all one-hot encoded vectors ("bag of words")

$$\boldsymbol{\phi}_{\mathrm{BOW}}(s_1 s_2 \cdots) = \boxed{\sum_j \boldsymbol{\phi}_{\mathrm{OHE}}(s_j)}$$

  - **Fixed-length text** $s_1 s_2 \cdots s_n$: A stacked $n \cdot |S|$ dimensional vector

$$\boldsymbol{\phi}_{\mathrm{FL}}(s_1 s_2 \cdots s_n) = \boxed{\begin{bmatrix} \boldsymbol{\phi}_{\mathrm{OHE}}(s_1) \\ \vdots \\ \boldsymbol{\phi}_{\mathrm{OHE}}(s_n) \end{bmatrix}}$$

$\mathcal{O}(n \cdot \log_2|S|)$

Bag of words **cannot** learn "positional" effect of text elements

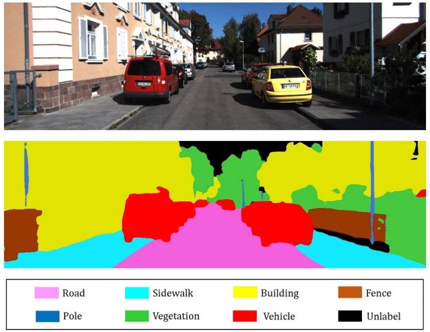
Overview

1. Linear Basis Function Models
2. **Modelling Data**
   - Modelling Text
   - **Modelling Images**
3. Linear Algebra
   - Vector Spaces
   - Linear Mappings and Matrices
   - Matrix Derivatives
4. Maximum A Posterior Learning and (Regularized) Least Squares

Modelling Images

- **Raw image data**. *An image is a rectangular array of picture elements ("pixels") that consist of a triple of intensities of the base colors red, blue and green.*

- Images can also be modelled at three levels of granularity depending on application
  1. **Pixels**: Image segmentation
  2. (Non-overlapping) **patches**: Object recognition
  3. **Whole image**: Image classification



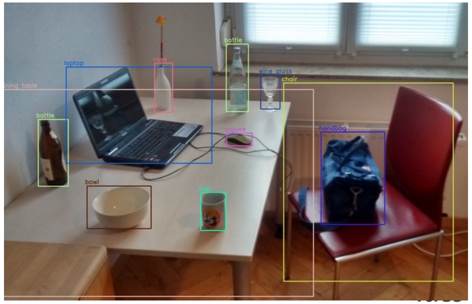


Image Content in Frequency and Location

- Image data contains signal (photon counts) at *fixed* locations in the image

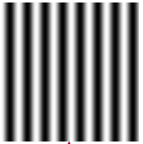
$\boldsymbol{\phi}_{\mathrm{RGB}}\Bigg($  $\Bigg) =$

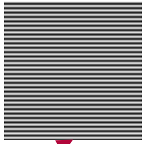
$$\begin{array}{cl} \begin{bmatrix} 127 \\ 0 \\ \vdots \\ 32 \\ 6 \end{bmatrix} & \begin{array}{l} \leftarrow \text{Red value at location (1,1)} \\ \leftarrow \text{Green value at location (1,1)} \\ \\ \leftarrow \text{Green value at location (800,600)} \\ \leftarrow \text{Blue value at location (800,600)} \end{array} \end{array}$$

- **Observation**. Recurring patterns are more visible in the frequency domain.
- **Idea**. Discrete Fourier transform (DFT) to transform the image

$$\boldsymbol{\phi}_{\mathrm{DFT}}(x) = \boldsymbol{F}_{\mathrm{DFT}}\boldsymbol{\phi}_{\mathrm{RGB}}(x)$$

  - In practice, there is a $\mathcal{O}(\#\mathrm{pixel} \cdot \log(\#\mathrm{pixel}))$ algorithm for fast Fourier transform (FFT)

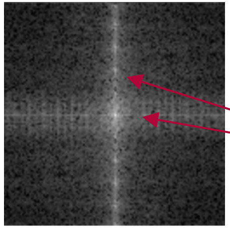




$$\boldsymbol{F}_{\mathrm{DFT}} = \boldsymbol{W}_{\mathrm{DFT}} \otimes \boldsymbol{W}_{\mathrm{DFT}}$$

Kronecker product of 1D discrete Fourier transform $\boldsymbol{W}_{\mathrm{DFT}}$

Raw image (in single-grey channel)



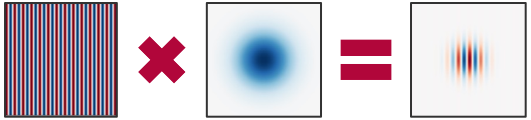
Fourier transform of image

Horizontal and vertical patterns by single features

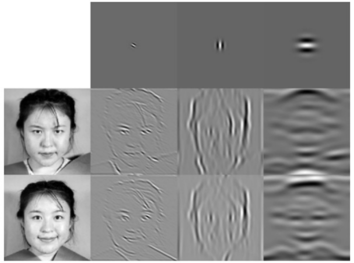
Gabor Wavelets: Mixing Location and Frequency

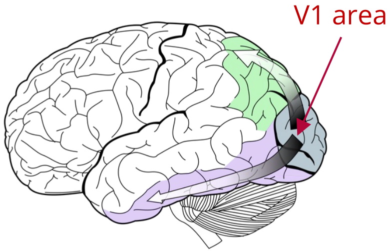
Dennis Gabor
(1900 – 1978)

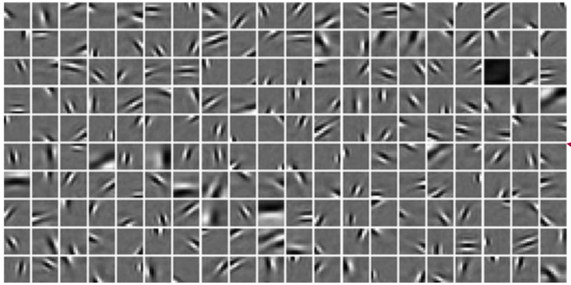
- Both representations are extremes:
  - **Raw image**: Single features $\phi_{\mathrm{RGB},i}(x)$ at a location without frequency
  - **DFT image**: Single features $\phi_{\mathrm{DFT},i}(x)$ in frequency without a location
- **Gabor wavelets**. A *Gabor wavelet* is a set of basis functions that is obtained by multiplying a Fourier basis function with a Gaussian density.



  - Combine both spatial and frequency information in image features
  - Basis functions for sparse encoding by the brain in V1







Filters "implemented" in the V1
(sparse coding!)

Bruno Olshausen & David Field (1997). Sparse Coding with Overcomplete Basis Set: A Strategy Employed by V1?

Overview

1. Linear Basis Function Models
2. Modelling Data
   - Modelling Text
   - Modelling Images
3. **Linear Algebra**
   - **Vector Spaces**
   - Linear Mappings and Matrices
   - Matrix Derivatives
4. Maximum A Posterior Learning and (Regularized) Least Squares

Vector Spaces

- **Vector Space**. *A vector space is a set* V *that satisfies the following axioms: There exists a null element* $\mathbf{0} \in V$ *such that for all* $\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w} \in V$ *and scalars* $a, b \in \mathbb{R}$
  1. **Associativity**: $\boldsymbol{u} + (\boldsymbol{v} + \boldsymbol{w}) = (\boldsymbol{u} + \boldsymbol{v}) + \boldsymbol{w}$
  2. **Commutativity**: $\boldsymbol{u} + \boldsymbol{v} = \boldsymbol{v} + \boldsymbol{u}$
  3. **Identity element (of vector addition)**: $\boldsymbol{v} + \mathbf{0} = \boldsymbol{v}$
  4. **Inverse element**: $\boldsymbol{v} + (-\boldsymbol{v}) = \mathbf{0}$
  5. **Identity element (of scalar multiplication)**: $1 \cdot \boldsymbol{v} = \boldsymbol{v}$
  6. **Distributivity (w.r.t. vector addition)**: $a \cdot (\boldsymbol{u} + \boldsymbol{v}) = a \cdot \boldsymbol{u} + a \cdot \boldsymbol{v}$
  7. **Distributivity (w.r.t. scalar addition)**: $(a + b) \cdot \boldsymbol{v} = a \cdot \boldsymbol{v} + b \cdot \boldsymbol{v}$
- **Two Examples**
  - **Points in space**: $V = \mathbb{R}^d$ with $\begin{pmatrix} u_1 \\ \vdots \\ u_d \end{pmatrix} + \begin{pmatrix} v_1 \\ \vdots \\ v_d \end{pmatrix} = \begin{pmatrix} u_1 + v_1 \\ \vdots \\ u_d + v_d \end{pmatrix}$ and $a \cdot \begin{pmatrix} u_1 \\ \vdots \\ u_d \end{pmatrix} = \begin{pmatrix} au_1 \\ \vdots \\ au_d \end{pmatrix}$
  - **Functions on** $\mathbb{R}$: $V = \mathbb{R}^{\mathbb{R}}$ with $f + g = x \mapsto f(x) + g(x)$ and $a \cdot f = x \mapsto a \cdot f(x)$

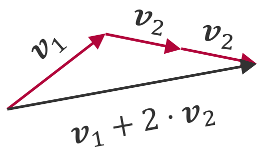
Bernhard Bolzano
(1781 – 1848)

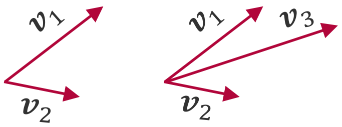
Linear Independence, Span and Basis

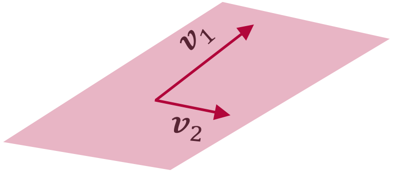
- **Linear combination**. *Given a set* $\boldsymbol{v}_1, \boldsymbol{v}_2, \dots, \boldsymbol{v}_n$ *of a vector space* V, *a* linear combination *is defined by* ($a_1, a_2, \dots, a_n$ *are called coefficients)*

$$a_1 \cdot \boldsymbol{v}_1 + a_2 \cdot \boldsymbol{v}_2 + \cdots + a_n \cdot \boldsymbol{v}_n$$

- **Linear independence**. *A set* $\boldsymbol{v}_1, \boldsymbol{v}_2, \dots, \boldsymbol{v}_n$ *is called* linearly independent *if no* $\boldsymbol{v}_i$ can be written as a linear combination of $\boldsymbol{v}_1, \dots, \boldsymbol{v}_{i-1}, \boldsymbol{v}_{i+1}, \dots, \boldsymbol{v}_n$.
- **Span**. *Given a set* $\boldsymbol{v}_1, \boldsymbol{v}_2, \dots, \boldsymbol{v}_n$ *of a vector space* V, *the* span $\mathrm{span}(\boldsymbol{v}_1, \boldsymbol{v}_2, \dots, \boldsymbol{v}_n)$ *is the set of all* linear combinations of $\boldsymbol{v}_1, \boldsymbol{v}_2, \dots, \boldsymbol{v}_n$.
  - **Example**:
    - $\mathrm{span}(\boldsymbol{v}_1)$ is the line through the origin along $\boldsymbol{v}_1$
    - $\mathrm{span}(\boldsymbol{v}_1, \boldsymbol{v}_2)$ is the plane through the origin along $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$
- **Basis**. *A subset* $\boldsymbol{b}_1, \boldsymbol{b}_2, \dots, \boldsymbol{b}_n$ *of a vector space* V *is called a* basis *if its span equals the whole vector space, that is,* $span(\boldsymbol{b}_1, \boldsymbol{b}_2, \dots, \boldsymbol{b}_n) = V$.
- **Dimensionality**. *All bases of a vector space* V *have the same cardinality called the* dimensionality *of the vector space,* $\dim(V)$.

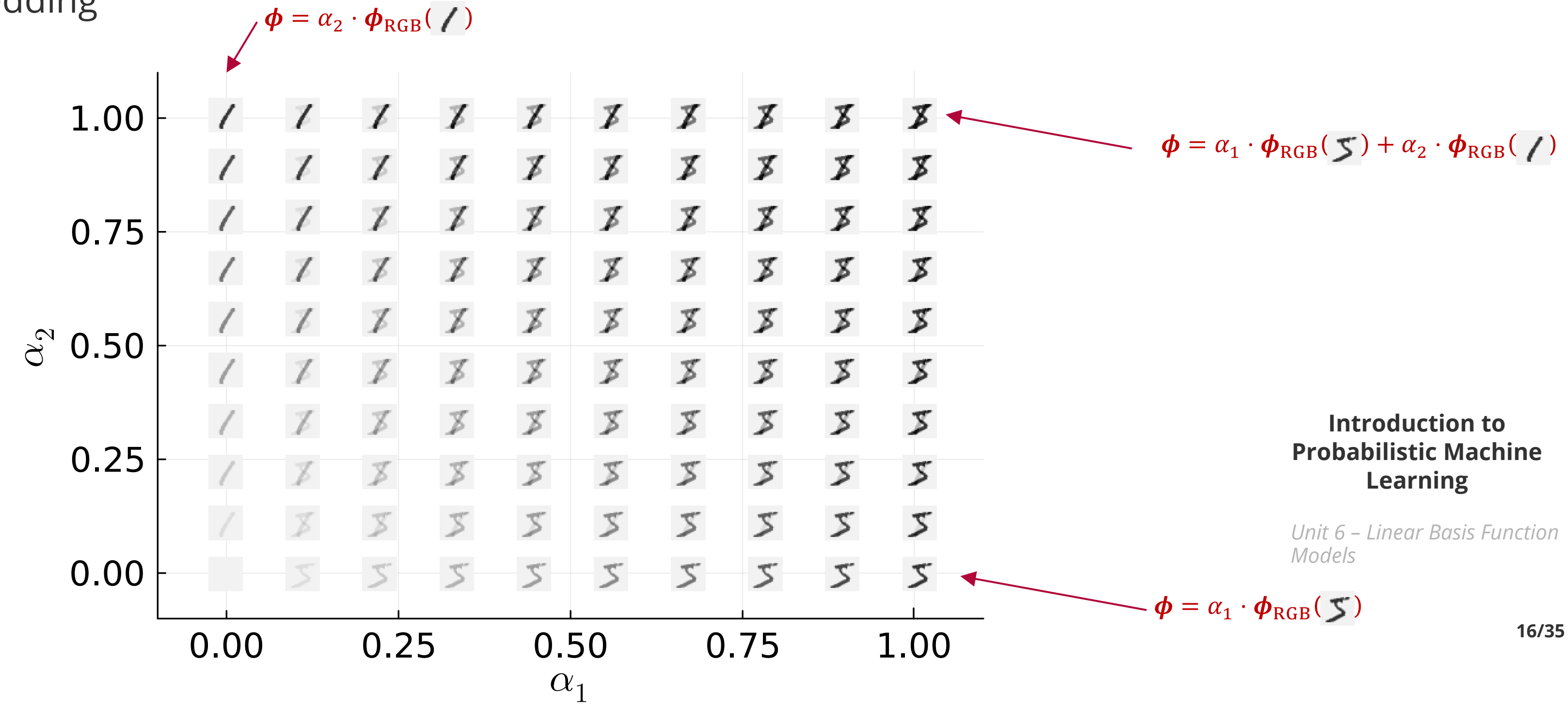




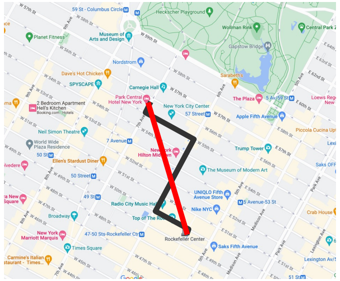


Linear Independence & Span Example

- Consider two 28x28 pictures and and their $28^2 = 784$-dimensional $\boldsymbol{\phi}_{\text{RGB}}$ embedding



Scalar Products and Inner Product Spaces



- So far, a vector space is a set but there is no notion of **distance**!
  - In New York, the **walking distance** between two points is much longer than the **flying distance**!
- **Metric space**. *A metric space is a vector space V together with a metric $d: V \times V \to \mathbb{R}^+$ that has the following properties for all $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}$:*

$$d(\boldsymbol{x}, \boldsymbol{y}) = 0 \Leftrightarrow \boldsymbol{x} = \boldsymbol{y}$$

$$d(\boldsymbol{x}, \boldsymbol{y}) = d(\boldsymbol{y}, \boldsymbol{x})$$

symmetry

$$d(\boldsymbol{x}, \boldsymbol{y}) \leq d(\boldsymbol{x}, \boldsymbol{z}) + d(\boldsymbol{z}, \boldsymbol{y})$$

triangle inequality

- **Inner product space**. *An inner product space is a vector space together with an inner product function $\langle \cdot, \cdot \rangle: V \times V \to \mathbb{R}^+$ that has the following properties for all $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}$:*

$$\langle \boldsymbol{x}, \boldsymbol{x} \rangle = 0 \Leftrightarrow \boldsymbol{x} = \boldsymbol{0}$$

$$\langle \boldsymbol{x}, \boldsymbol{y} \rangle = \langle \boldsymbol{y}, \boldsymbol{x} \rangle$$

symmetry

$$\langle a \cdot \boldsymbol{x} + b \cdot \boldsymbol{y}, \boldsymbol{z} \rangle = a \langle \boldsymbol{x}, \boldsymbol{z} \rangle + b \langle \boldsymbol{y}, \boldsymbol{z} \rangle$$

linearity

- *Every inner product space is a metric space with $d(\boldsymbol{x}, \boldsymbol{y}) = \|\boldsymbol{x} - \boldsymbol{y}\| = \sqrt{\langle \boldsymbol{x} - \boldsymbol{y}, \boldsymbol{x} - \boldsymbol{y} \rangle}$!*

Overview

1. Linear Basis Function Models
2. Modelling Data
   - Modelling Text
   - Modelling Images
3. **Linear Algebra**
   - Vector Spaces
   - **Linear Mappings and Matrices**
   - Matrix Derivatives
4. Maximum A Posterior Learning and (Regularized) Least Squares

Linear Mappings and Matrices

- **Linear mapping**. *A function $f: V \to W$ is a* linear mapping *between vector space V and W if for any two vectors $\boldsymbol{u}, \boldsymbol{v} \in V$ and any scalar $c \in \mathbb{R}$:*

$$f(\boldsymbol{u} + \boldsymbol{v}) = f(\boldsymbol{u}) + f(\boldsymbol{v})$$
$$f(c \cdot \boldsymbol{u}) = c \cdot f(\boldsymbol{u})$$

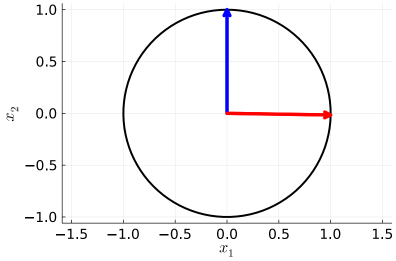
  - Addition and scalar multiplication can be applied before or after the map!
  - It follows that $f(\boldsymbol{0}) = f(0 \cdot \boldsymbol{v}) = 0 \cdot f(\boldsymbol{v}) = \boldsymbol{0}$!

- **Theorem**. *Every linear mapping $f: V \to W$ between V and W of dimension n and m can be represented via a matrix multiplication with an $m \times n$ matrix $\boldsymbol{A}$. The rank of $\boldsymbol{A}$ is the dimensionality of the image of W, that is $\operatorname{rank}(\boldsymbol{A}) = \dim(W)$.*
  - **Proof**. If $\{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n\}$ is a basis for V and $\{\boldsymbol{w}_1, \ldots, \boldsymbol{w}_m\}$ is a basis for W then we know

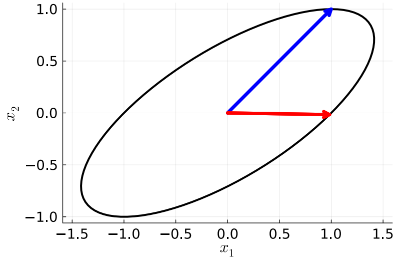
$$f(\boldsymbol{v}_j) = a_{1j} \cdot \boldsymbol{w}_1 + \cdots + a_{mj} \cdot \boldsymbol{w}_m$$

$$f(\boldsymbol{v}) = f(c_1\boldsymbol{v}_1 + \cdots + c_n\boldsymbol{v}_n) = c_1 \cdot f(\boldsymbol{v}_1) + \cdots + c_n \cdot f(\boldsymbol{v}_n) = \sum_{i=1}^{m} [\boldsymbol{A}\boldsymbol{c}]_i \cdot \boldsymbol{w}_i$$

  - For $V = \mathbb{R}^n$ and $W = \mathbb{R}^m$, the columns of $\boldsymbol{A}$ are the images of the basis vectors in $\mathbb{R}^n$.
  - For any matrix $\boldsymbol{A} \in \mathbb{R}^{n \times m}$, $\operatorname{rank}(A) \leq \min(n, m)$.

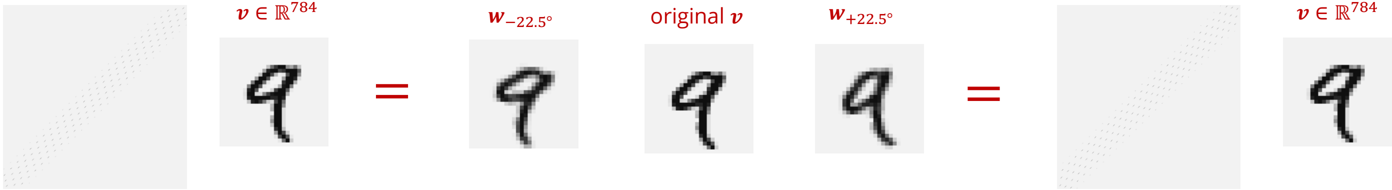


$$f(\boldsymbol{x}) = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \boldsymbol{x}$$

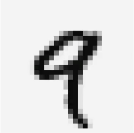


Linear Mappings and Matrices Example

- Consider a 28x28 picture  and its $28^2 = 784$-dimensional $\boldsymbol{\phi}_{\mathrm{RGB}}$ embedding
- Then the rotation of the image content can be expressed as a linear mapping from 784-dimensional onto itself
  - By computing the four fractional source pixel that contributed to a target pixel



- The following movie was produced by 120 linear mappings of 3° applied to the same 784-dimensional $\boldsymbol{\phi}_{\mathrm{RGB}}$ embedding of the source image



Matrix Types and Properties

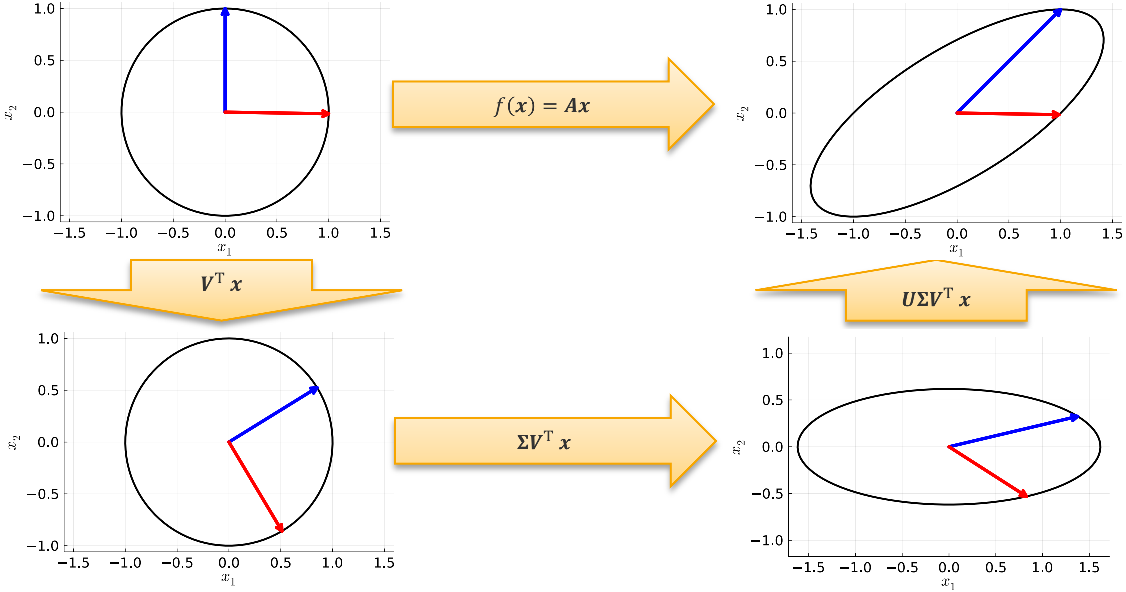
- **Square Matrix**. *A matrix $\boldsymbol{A} \in \mathbb{R}^{n \times m}$ where $m = n$ is called a* square matrix.
  - It parameterizes mappings of a space onto itself.
- **Diagonal Matrix**. *A square matrix $\boldsymbol{A}$ is called* diagional *if $\boldsymbol{A}_{ij} = 0$ for all $i \neq j$.*
  - Geometrically, a diagonal matrix is an axis scaling (mapping).
  - A special diagonal matrix is $\mathbf{A}_{ii} = 1$ which is also called identity matrix.
- **Orthogonal Matrix**. *A square matrix $\boldsymbol{A}$ is called orthogonal if $\boldsymbol{A}\boldsymbol{A}^{\mathrm{T}} = \boldsymbol{I}$.*
  - Geometrically, an orthogonal matrix is a rotation and mirroring (mapping).
- **Symmetric Matrix**. *A square matrix $\boldsymbol{A}$ is called* symmetric *if and only if $\boldsymbol{A} = \boldsymbol{A}^{\mathrm{T}}$.*
- **Positive semi-definite matrix**. *A symmetric matrix $\boldsymbol{A}$ is called* positive definite *if and only if*

$$\forall \boldsymbol{x} \neq \boldsymbol{0} \colon \boldsymbol{x}^{\mathrm{T}} \boldsymbol{A} \boldsymbol{x} > 0$$

  - Positive definiteness means that no axis is inverted or removed in the mapping

- **Singular Value Decomposition**. *Any matrix $\boldsymbol{A} \in \mathbb{R}^{n \times m}$ has a decomposition into three matrices $\boldsymbol{U} \in \mathbb{R}^{n \times n}$, $\boldsymbol{\Sigma} \in \mathbb{R}^{n \times m}$ and $\boldsymbol{V} \in \mathbb{R}^{m \times m}$ such that $\boldsymbol{U}$ and $\boldsymbol{V}$ are orthogonal and $\boldsymbol{\Sigma}$ is only non-zero on diagonal elements*

$$\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^{\mathrm{T}}$$

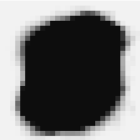


Singular Value Decomposition Example

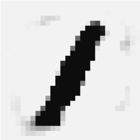
- Imagine we arrange the 784-dimensional RGB vectors for 1000 images of digits in the rows of a matrix $\boldsymbol{A}$. Then the SVD has the following property

$$\boldsymbol{A} = \underbrace{\boldsymbol{U}}_{\boldsymbol{U} \in \mathbb{R}^{1000\times 784}}\underbrace{\boldsymbol{\Sigma}\boldsymbol{V}^{\mathrm{T}}}_{= \boldsymbol{B} \in \mathbb{R}^{784\times 784}} = \boldsymbol{U}\boldsymbol{B} = \boldsymbol{U} \cdot [\boldsymbol{b}_1 \quad \cdots \quad \boldsymbol{b}_{784}]$$

$\boldsymbol{b}_1$

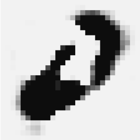


$\boldsymbol{b}_2$



$\boldsymbol{b}_3$

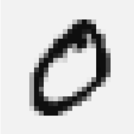
$\boldsymbol{b}_4$



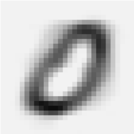
$\boldsymbol{b}_5$

- Reconstruction with $\widehat{\boldsymbol{A}} = \begin{bmatrix} \boldsymbol{u}_1^T \\ \vdots \\ \boldsymbol{u}_{1000}^T \end{bmatrix} [\boldsymbol{b}_1 \quad \cdots \quad \boldsymbol{b}_k]$ for the first k singular values

Original $\boldsymbol{a}_1$



Reconstruction $\widehat{\boldsymbol{a}}_1$



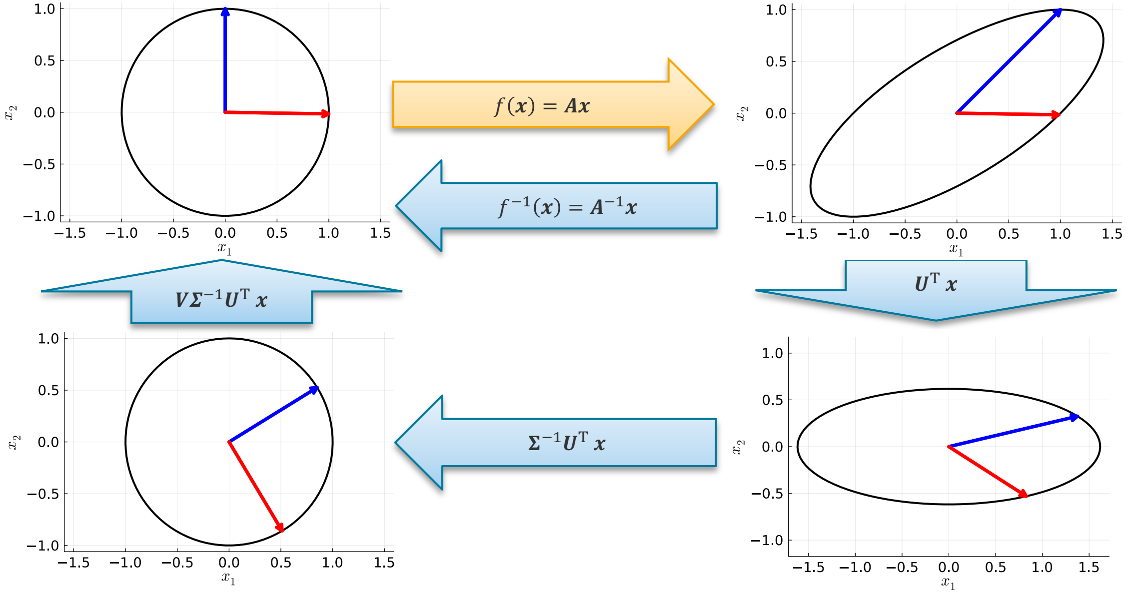
Inverse of a Matrix and SVD

- **Inverse**. *The inverse $\boldsymbol{A}^{-1}$ of a full-rank square matrix $\boldsymbol{A}$ has the property that*

$$\boldsymbol{A}^{-1}\boldsymbol{A} = \boldsymbol{A}\boldsymbol{A}^{-1} = \boldsymbol{I}$$

  - One way to compute it is with the singular value decomposition

$$\boldsymbol{A}^{-1} = \boldsymbol{V}\boldsymbol{\Sigma}^{-1}\boldsymbol{U}^{\mathrm{T}}$$



Cholesky Decomposition

Hasso Plattner Institut
Digital Engineering · Universität Potsdam

- **Numerical Challenge**: Given a symmetric, positive-definite matrix $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ and a vector $\boldsymbol{b} \in \mathbb{R}^n$ find the solution $\boldsymbol{x}$ such that

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$$

- **Naïve Solution**: Invert the matrix $\boldsymbol{A}$ and compute

$$\boldsymbol{x} = \boldsymbol{A}^{-1}\boldsymbol{b}$$

  - **Challenges**: If $\boldsymbol{A}$ has some singular values close to zero, this is numerically unstable!

- **Cholesky Decomposition**: Every positive-definitive matrix $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ has a unique decomposition into a lower-triangular matrix $\boldsymbol{L} \in \mathbb{R}^{n \times n}$, $L_{ij} = 0$ if $j > i$

$$\boldsymbol{A} = \boldsymbol{L}\boldsymbol{L}^T$$

  - **Advantage**: Finding $\boldsymbol{y}$ such that $\boldsymbol{L}\boldsymbol{y} = \boldsymbol{b}$ can be done in $O(n^2)$ without an inverse simply using back-substitution (written as $\boldsymbol{y} = \boldsymbol{L} \backslash \boldsymbol{b}$)!

- **Cholesky Solution**: Find the solution $\boldsymbol{x}$ for $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ is a two-step algorithm

1. Compute $\boldsymbol{y} = \boldsymbol{L} \backslash \boldsymbol{b}$
2. Compute $\boldsymbol{x} = \boldsymbol{L}^T \backslash \boldsymbol{y}$

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \Leftrightarrow \boldsymbol{L}\boxed{\boldsymbol{L}^T\boldsymbol{x}} = \boldsymbol{b}$$

$$\boldsymbol{L}\boldsymbol{y} = \boldsymbol{b}$$

André-Louis Cholesky
(1875 – 1918)

$$\begin{pmatrix} L_{11} & 0 & 0 \\ L_{21} & L_{22} & 0 \\ L_{31} & L_{32} & L_{33} \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix}$$

$$y_1 = \frac{b_1}{L_{11}}$$

$$y_2 = \frac{b_2 - L_{21} \cdot y_1}{L_{22}}$$

$$y_3 = \frac{b_3 - L_{31} \cdot y_1 - L_{32} \cdot y_2}{L_{33}}$$

Overview

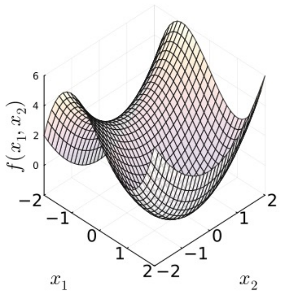
1. Linear Basis Function Models
2. Modelling Data
   - Modelling Text
   - Modelling Images
3. Linear Algebra
   - Vector Spaces
   - Linear Mappings and Matrices
   - **Matrix Derivatives**
4. Maximum A Posterior Learning and (Regularized) Least Squares

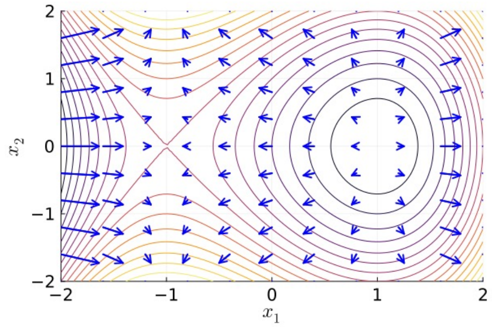
Derivates of Vector-Valued Functions

- **Derivate of vector-valued functions**. *Let $g: \mathbb{R}^m \to \mathbb{R}^n$ be a vector-valued function. Then the $m \times n$ matrix $\frac{\partial g(\boldsymbol{x})}{\partial \boldsymbol{x}}$ of derivates is defined by*

$$\frac{\partial g(\boldsymbol{x})}{\partial \boldsymbol{x}} := \left[\frac{\partial g_j(\boldsymbol{x})}{\partial x_i}\right]_{i,j=1}^{m,n} = \begin{bmatrix} \frac{dg_1(\boldsymbol{x})}{dx_1} & \cdots & \frac{dg_n(\boldsymbol{x})}{dx_1} \\ \vdots & \ddots & \vdots \\ \frac{dg_1(\boldsymbol{x})}{dx_m} & \cdots & \frac{dg_n(\boldsymbol{x})}{dx_m} \end{bmatrix}$$

- **Example**. Let $g(x_1, x_2) = x_1^3 - 3x_1 + x_2^2$





Introduction to Probabilistic Machine Learning

Unit 6 – Linear Basis Function Models

Derivates of Linear Mappings

- **Theorem**. *Let $f(\boldsymbol{x})$ be a linear mapping, $f(\boldsymbol{x}) = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{b}$. Then*
$$\frac{\partial f(\boldsymbol{x})}{\partial \boldsymbol{x}} = \boldsymbol{A}^{\mathrm{T}}$$
- **Theorem**. *Let $f(\boldsymbol{x})$ be a quadratic form, $f(\boldsymbol{x}) = \boldsymbol{x}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{x}$. Then*
$$\frac{\partial f(\boldsymbol{x})}{\partial \boldsymbol{x}} = 2\boldsymbol{A}^{\mathrm{T}}\boldsymbol{x}$$
- **Example**. Let us consider $f(\boldsymbol{x}) = \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|^2 = (\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b})^{\mathrm{T}}(\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b})$

$$f(\boldsymbol{x}) = \boldsymbol{x}^{\mathrm{T}}\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{x} - 2\boldsymbol{b}^{T}\boldsymbol{A}\boldsymbol{x} + \boldsymbol{b}^{\mathrm{T}}\boldsymbol{b}$$
$$\left.\frac{\partial f(\boldsymbol{x})}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}=\boldsymbol{x}^*} = 2\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{x}^* - 2\boldsymbol{A}^{\mathrm{T}}\boldsymbol{b} = \boldsymbol{0}$$
$$\left(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}\right)\boldsymbol{x}^* = \boldsymbol{A}^{\mathrm{T}}\boldsymbol{b}$$
normal equations
$$\boldsymbol{x}^* = \left(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}\right)^{-1}\boldsymbol{A}^{\mathrm{T}}\boldsymbol{b}$$
alternatively, use Cholesky decomposition and backsubstitution

Overview

1. Linear Basis Function Models
2. Modelling Data
   - Modelling Text
   - Modelling Images
3. Linear Algebra
   - Vector Spaces
   - Linear Mappings and Matrices
   - Matrix Derivatives
4. **Maximum A Posterior Learning and (Regularized) Least Squares**

Maximum Likelihood and Least Squares

- **Gaussian Likelihood Model**. Let's assume that our observations y are obtained from adding zero-mean normally distributed noise to $f(x;\boldsymbol{w})$

$$p(y|x,\boldsymbol{w}) = \mathcal{N}(y; f(x;\boldsymbol{w}), \sigma^2)$$

- **Maximum Likelihood Problem**. Noting that $\log(\cdot)$ is a strictly monotonic function, we see that

$$\boldsymbol{w}_{\mathrm{ML}} \coloneqq \mathrm{argmax}_{\boldsymbol{w}} \prod_i p(y_i|x_i,\boldsymbol{w}) = \mathrm{argmax}_{\boldsymbol{w}} \sum_i \log\left(\mathcal{N}(y_i; f(x_i;\boldsymbol{w}), \sigma^2)\right)$$

  - Product and sum have the same maximum, but the objective function is numerically **much** more stable as a sum of logarithms of densities (why?)

- **Least Squares**. *The minimizer of the objective function* $\sum_i \left(y_i - f(x_i;\boldsymbol{w})\right)^2$ *is* $\boldsymbol{w}_{\mathrm{ML}}$.
- **Proof**. Looking at the normal density for a given (x_i, y_i) we see that

$$\log\left(\frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{\left(y_i - f(x_i;\boldsymbol{w})\right)^2}{2\sigma^2}\right)\right) = \log\left(\frac{1}{\sqrt{2\pi}\sigma}\right) - \frac{1}{2\sigma^2}\cdot \underbrace{\left(y_i - f(x_i;\boldsymbol{w})\right)^2}_{\text{only part dependent on } \boldsymbol{w}}$$

Maximum Likelihood Revisited

- **Maximum Likelihood Problem (in matrix notation).** If $\Phi_{ij} = \phi_j(x_i)$

$$\boldsymbol{w}_{\mathrm{ML}} \coloneqq \mathrm{argmin}_{\boldsymbol{w}} \|\boldsymbol{\Phi}\boldsymbol{w} - \boldsymbol{y}\|^2$$

  - If we have more data points $x_1, \dots, x_N$ than basis functions $\phi_1, \dots, \phi_D$, then $\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}$ has full rank

$$\boxed{\boldsymbol{w}_{\mathrm{ML}} = \left(\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}\right)^{-1}\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{y}}$$

- **Computational complexity**:$O(ND^2)$
  1. The matrix multiplication $\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}$ is an $O(ND^2)$ operation
  2. The matrix product $\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{y}$ is an $O(ND)$ operation
  3. Computing the inverse $\left(\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}\right)^{-1}$ or the Cholesky decomposition is an $O(D^3)$ operation
  4. The matrix product$\left[\left(\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}\right)^{-1}\right] \cdot \left[\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{y}\right]$ is an $O(D^2)$ operation

Maximum A Posteriori Learning

- **Gaussian Prior Model**. Let's also assume a componentwise Gaussian over $\boldsymbol{w}$

$$p(\boldsymbol{w}) = \prod_d \mathcal{N}(w_d; 0, \tau^2)$$

- **Maximum A-Posterior Problem**. Noting that $\log(\cdot)$ is strictly monotonic

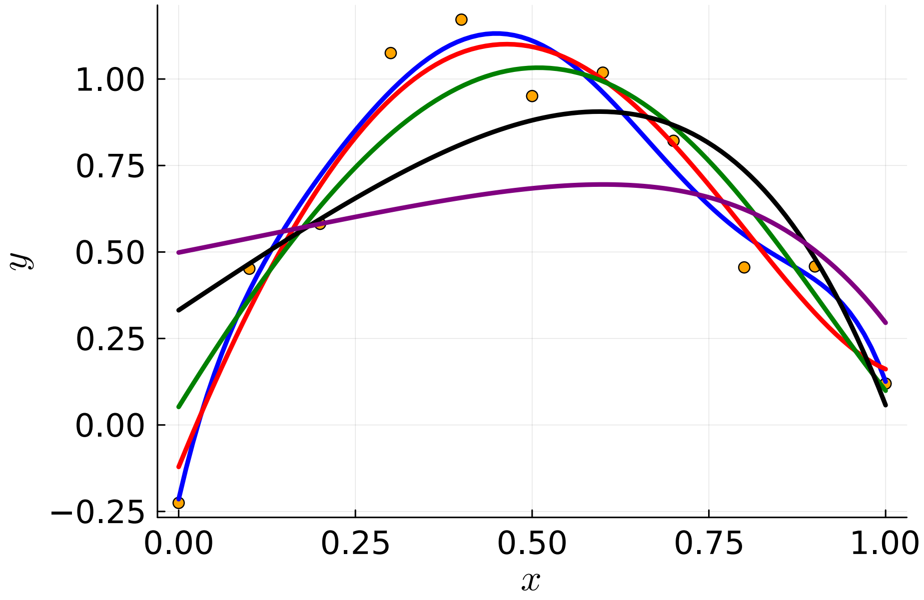
$$\boldsymbol{w}_{\mathrm{MAP}} \coloneqq \mathrm{argmax}_{\boldsymbol{w}} \underbrace{\prod_i p(y_i|x_i, \boldsymbol{w})}_{p(D|\boldsymbol{w})} \cdot \underbrace{\prod_d p(w_d)}_{p(\boldsymbol{w})} = \mathrm{argmin}_{\boldsymbol{w}} \frac{1}{2\sigma^2} \|\boldsymbol{\Phi}\boldsymbol{w} - \boldsymbol{y}\|^2 + \frac{1}{2\tau^2} \boldsymbol{w}^{\mathrm{T}}\boldsymbol{w}$$

$$\left.\frac{\partial \log\big(p(D|\boldsymbol{w}) \cdot p(\boldsymbol{w})\big)}{\partial \boldsymbol{w}}\right|_{\boldsymbol{w}=\boldsymbol{w}_{\mathrm{MAP}}} = \frac{1}{\sigma^2}\left(\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}\boldsymbol{w}_{\mathrm{MAP}} - \boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{y}\right) + \frac{1}{\tau^2}\boldsymbol{w}_{\mathrm{MAP}} = \boldsymbol{0}$$

$$\boxed{\boldsymbol{w}_{\mathrm{MAP}} = \left(\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi} + \frac{\sigma^2}{\tau^2}\boldsymbol{I}\right)^{-1} \boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{y}}$$

Maximum A Posteriori Learning: Polynomial Regression

$$f(x; \boldsymbol{w}) = \sum_{d=0}^{6} w_d \cdot x^d$$



$\frac{\sigma^2}{\tau^2} = 0$

$\frac{\sigma^2}{\tau^2} = 10^{-3}$

$\frac{\sigma^2}{\tau^2} = 10^{-2}$

$\frac{\sigma^2}{\tau^2} = 10^{-1}$

$\frac{\sigma^2}{\tau^2} = 10^{-0}$

Summary

- **Linear Basis Functions Models**
  - Non-linear prediction models can be formed with non-linear basis functions
  - Linearity is in the parameters, *not* the input dimensions of the data
- **Modelling Data**
  - Both textual and image data can be represented at different levels of granularity
  - Images should ideally be represented in terms of feature of location and frequency: Gabor wavelets/filters are excellent candidate functions for such features
- **Linear Mappings and Matrices**
  - All linear mappings can be expressed via matrix products
  - The singular value decomposition is the rotation-scaling-rotation view on a mapping
  - The inverse and Cholesky decomposition are critical for solving least-squares problems
- **Matrix Derivatives**
  - Matrix derivatives are a natural generalization of 1D-function derivatives
  - Regularized least-square has closed-form solution using matrix derivatives

See you next week!