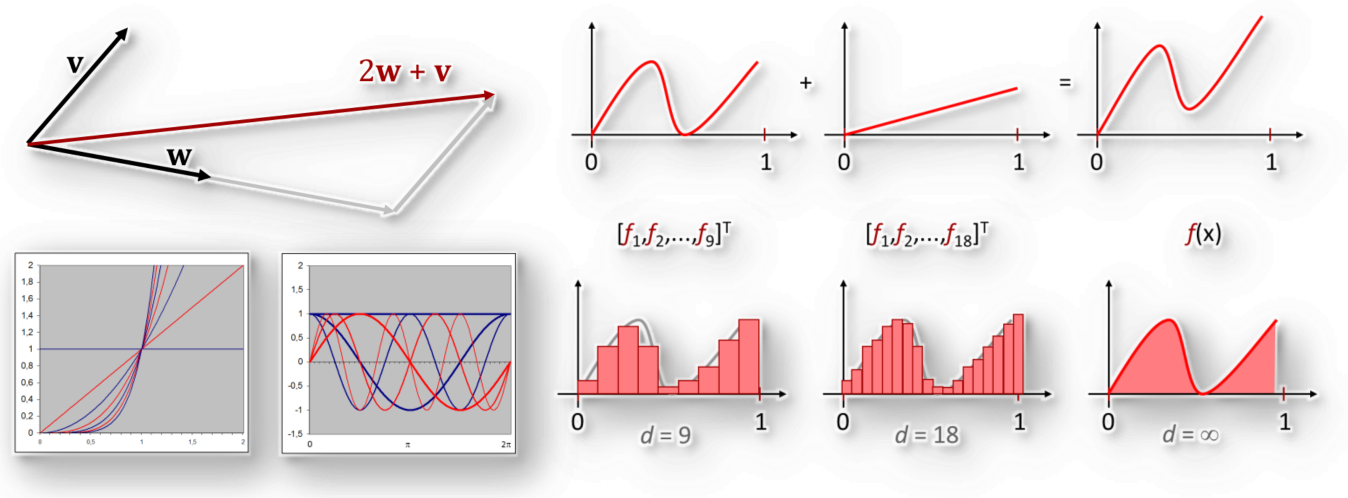
Statistical Geometry Processing

Winter Semester 2011/2012



Linear Algebra, Function Spaces & Inverse Problems

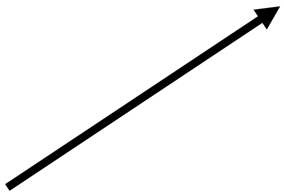
UNIVERSITÄT DES SAARLANDES

CLUSTER OF EXCELLENCE

max planck institut informatik

Vector and Function Spaces

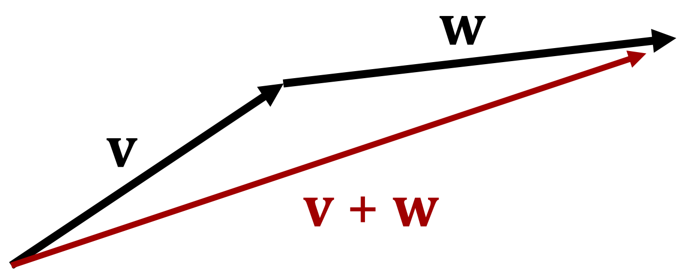
Vectors



vectors are arrows in space

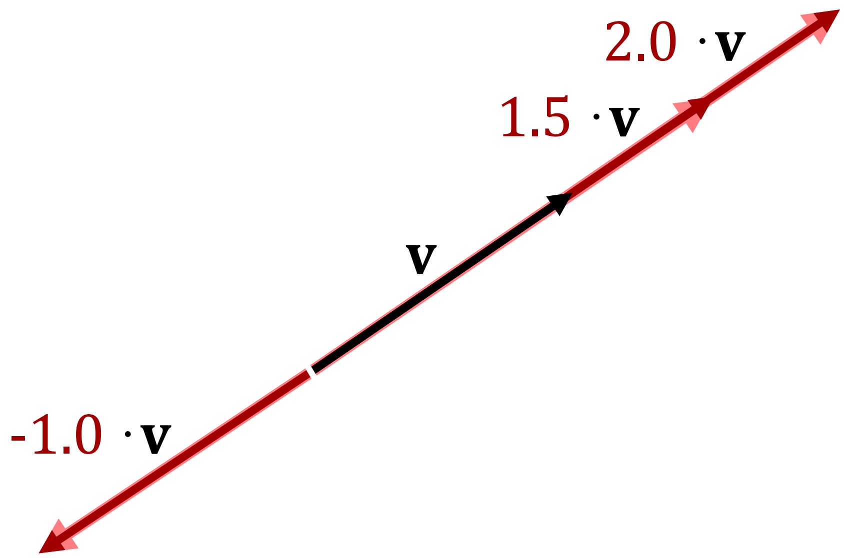
classically: 2 or 3 dim. Euclidian space

Vector Operations



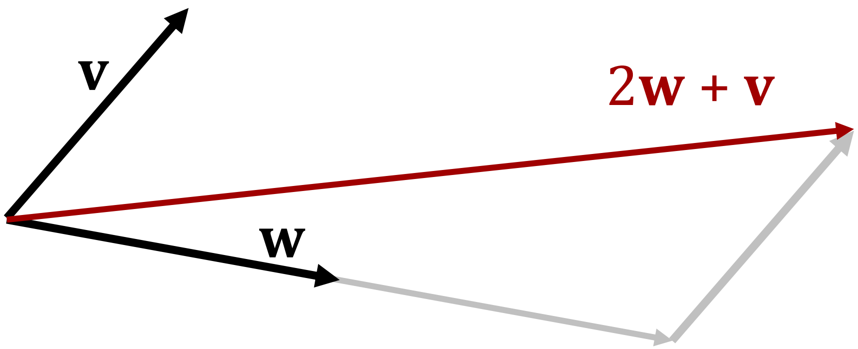
“Adding” Vectors:
Concatenation

Vector Operations



Scalar Multiplication:
Scaling vectors (incl. mirroring)

You can combine it...



Linear Combinations:

This is basically all you can do.

$$\mathbf{r} = \sum_{i=1}^{n} \lambda_i \mathbf{v}_i$$

Vector Spaces

Vector space:

- Set of vectors V
- Based on field F (we use only F = $\mathbb{R}$)
- Two operations:
  - Adding vectors **u** = **v** + **w** (**u**, **v**, **w** ∈ V)
  - Scaling vectors w = λv (**u** ∈ V, λ ∈ F)
- Vector space *axioms*:

(a1) $\forall \mathbf{u},\mathbf{v},\mathbf{w} \in V: (\mathbf{u}+\mathbf{v})+\mathbf{w} = \mathbf{u}+(\mathbf{v}+\mathbf{w})$

(a2) $\forall \mathbf{u},\mathbf{v} \in V: \mathbf{u}+\mathbf{v} = \mathbf{v}+\mathbf{u}$

(a3) $\exists \mathbf{0}_V \in V: \forall \mathbf{v} \in V: \mathbf{v}+\mathbf{0}_V = \mathbf{v}$

(a4) $\forall \mathbf{v} \in V: \exists \mathbf{w} \in V: \mathbf{v}+\mathbf{w} = \mathbf{0}_V$

(s1) $\forall \mathbf{v} \in V, \lambda, \mu \in F: \lambda(\mu \mathbf{v}) = (\lambda\mu)\mathbf{v}$

(s2) $\text{for } 1_F \in F: \forall \mathbf{v} \in V: 1_F \mathbf{v} = \mathbf{v}$

(s3) $\forall \lambda \in F: \forall \mathbf{v},\mathbf{w} \in V: \lambda(\mathbf{v}+\mathbf{w}) = \lambda\mathbf{v} + \lambda\mathbf{w}$

(s4) $\forall \lambda, \mu \in F, \mathbf{v} \in V: (\lambda+\mu)\mathbf{v} = \lambda\mathbf{v} + \mu\mathbf{v}$

Additional Tools

More concepts:

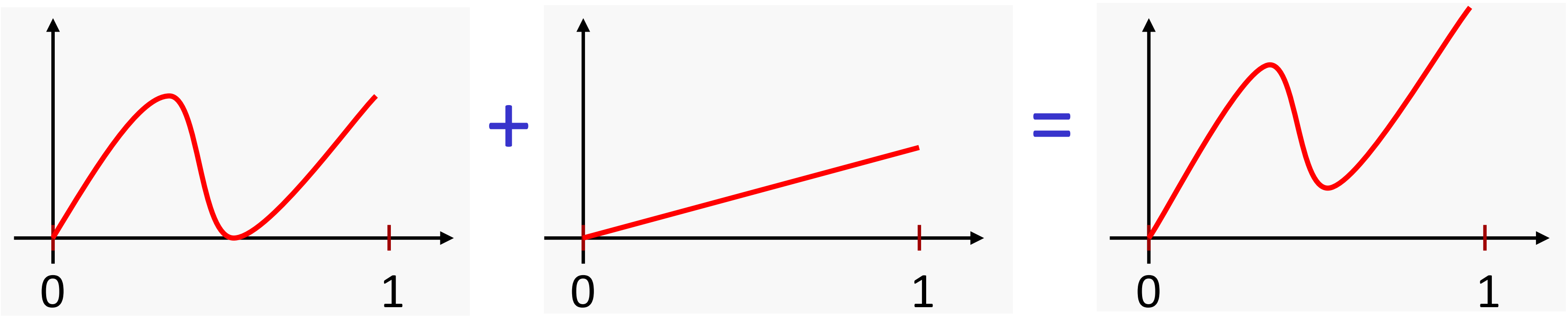
- Subspaces, linear spans, bases
- Scalar product
  - Angle, length, orthogonality
  - Gram-Schmidt orthogonalization
- Cross product ($\mathbb{R}^3$)
- Linear maps
  - Matrices
- Eigenvalues & eigenvectors
- Quadratic forms

(Check your old math books)

Example Spaces

Function spaces:

- Space of all functions $f\colon \mathbb{R} \to \mathbb{R}$
- Space of all smooth C^k functions $f\colon \mathbb{R} \to \mathbb{R}$
- Space of all functions $f\colon [0..1]^5 \to \mathbb{R}^8$
- etc...

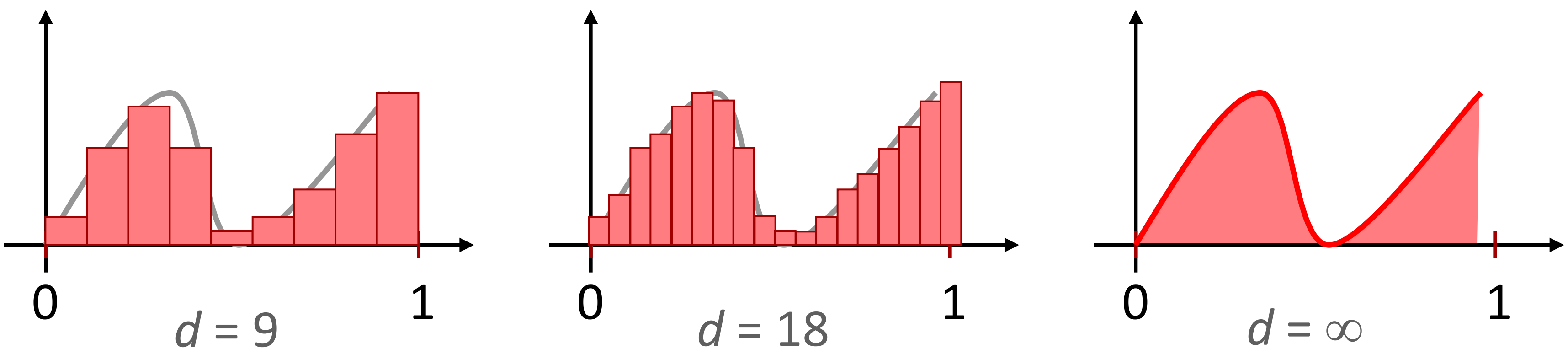


Function Spaces

Intuition:

- Start with a finite dimensional vector
- Increase sampling density towards infinity
- Real numbers: uncountable amount of dimensions

$[f_1, f_2, \ldots, f_9]^T$ $[f_1, f_2, \ldots, f_{18}]^T$ $f(x)$



Dot Product on Function Spaces

Scalar products

- For square-integrable functions $f, g : \Omega \subseteq \mathbb{R}^n \to \mathbb{R}$, the *standard scalar product* is defined as:

$$f \cdot g := \int_{\Omega} f(x) g(x) dx$$

- It measures an abstract norm and “angle” between function (not in a geometric sense)
- **Orthogonal functions:**
  - Do not influence each other in linear combinations.
  - Adding one to the other does not change the value in the other ones direction.

Approximation of Function Spaces

Finite dimensional subspaces:

- Function spaces with infinite dimension are hard to represented on a computer
- For numerical purpose, finite-dimensional subspaces are used to approximate the larger space
- Two basic approaches

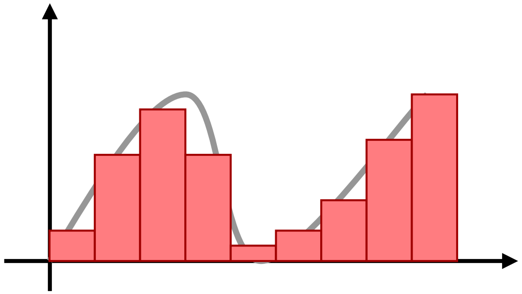
Approximation of Function Spaces

Task:

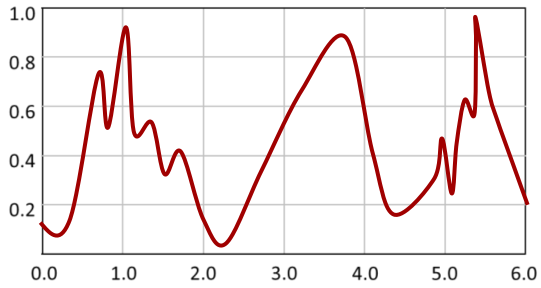
- **Given:** Infinite-dimensional function space V.
- **Task:** Find $f \in$ V with a certain property.

Recipe: “Finite Differences”

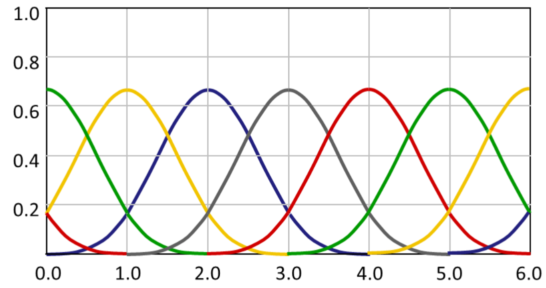
- Sample function f on discrete grid
- Approximate property discretely
  - Derivatives => finite differences
  - Integrals => Finite sums
- Optimization: Find best discrete function



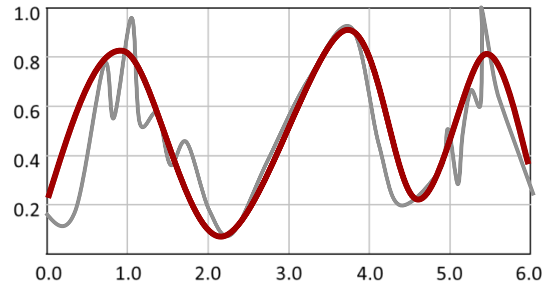
Approximation of Function Spaces



actual solution



function space basis

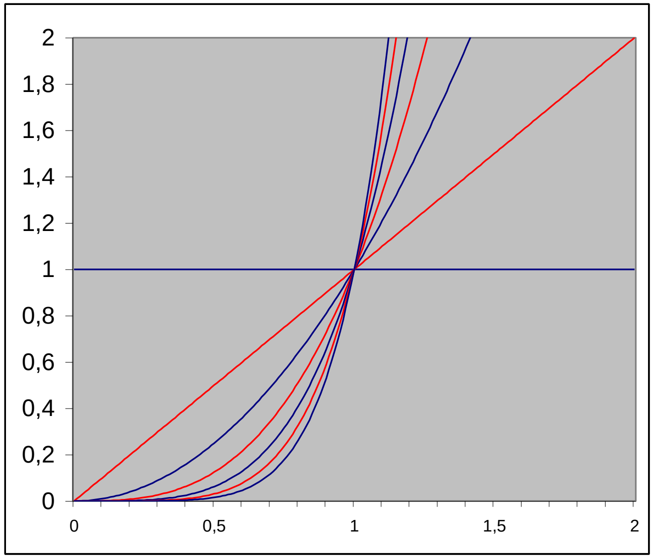


approximate solution

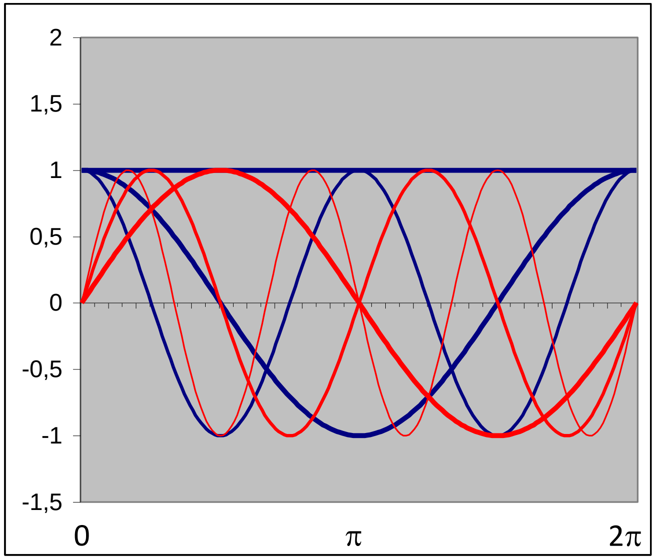
Recipe: "Finite Elements"

- Choose basis functions $b_1, \ldots, b_d \in V$
- Find $\tilde{f} = \sum_{i=1}^{d} \lambda_i b_i$ that matches the property best
- $\tilde{f}$ is described by $(\lambda_1, \ldots, \lambda_d)$

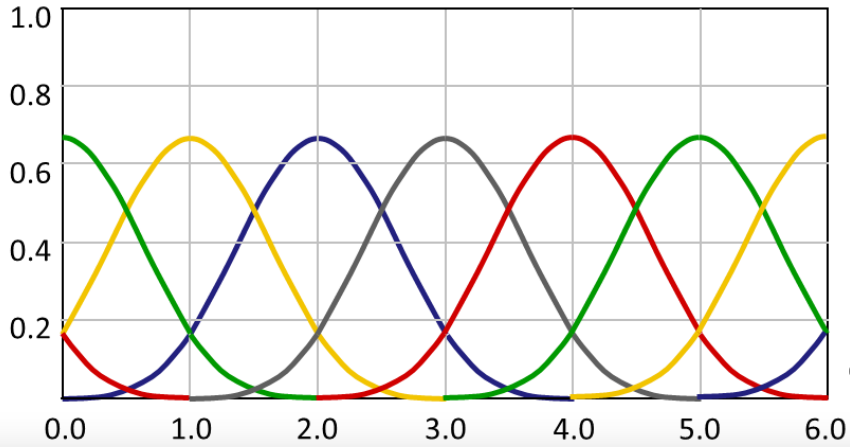
Examples



Monomial basis



Fourier basis



B-spline basis,

Gaussian basis

"Best Match"

Linear combination matches best

- **Solution 1:** Least squares minimization

$$\int_{\mathbb{R}} \left(f(x) - \sum_{i=1}^{n} \lambda_i b_i(x) \right)^2 dx \;\rightarrow min$$

- **Solution 2:** Galerkin method

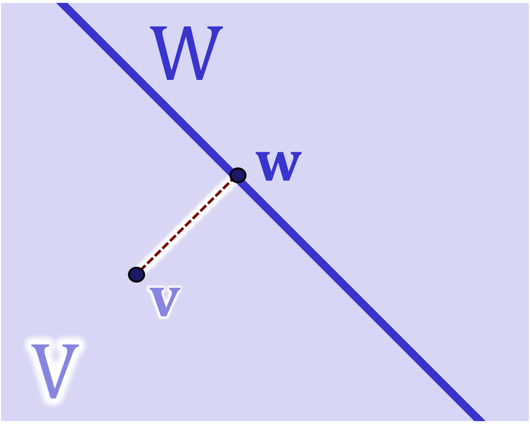
$$\forall i = 1..n\!: \left\langle f - \sum_{i=1}^{n} \lambda_i b_i \,,\; b_i \right\rangle = 0$$

- Both are equivalent

Optimality Criterion

Given:

- Subspace $\mathrm{W} \subseteq \mathrm{V}$
- An element $\mathbf{v} \in \mathrm{V}$



Then we get:

- $\mathbf{w} \in \mathrm{W}$ minimizes the quadratic error $(\mathrm{w} - \mathbf{v})^2$ (i.e. the Euclidean distance) if and only if:
- the residual $(\mathrm{w} - \mathbf{v})$ is orthogonal to W

Least squares = minimal Euclidean distance

Formal Derivation

Least-squares

$$E(f) = \int_{\mathbb{R}} \left(f(x) - \sum_{i=1}^{n} \lambda_i b_i(x) \right)^2 dx$$

$$= \int_{\mathbb{R}} \left(f^2(x) - 2\sum_{i=1}^{n} \lambda_i f(x) b_i(x) + \sum_{i=1}^{n}\sum_{i=1}^{n} \lambda_i \lambda_j b_i(x) b_j(x) \right) dx$$

Setting derivatives to zero:

$$\nabla E(f) = -2 \begin{pmatrix} \lambda_1 \langle f, b_1 \rangle \\ \vdots \\ \lambda_n \langle f, b_n \rangle \end{pmatrix} + [\lambda_1, \ldots, \lambda_n] \begin{pmatrix} \ddots & \vdots & \iddots \\ \cdots & \langle b_i(x), b_j(x) \rangle & \cdots \\ \iddots & \vdots & \ddots \end{pmatrix}$$

Result:

$$\forall j = 1..n: \left\langle \left(f - \sum_{i=1}^{n} \lambda_i b_i \right), b_j \right\rangle = 0$$

Linear Maps

Linear Maps

A Function

- $f: V \rightarrow W$ between vector spaces V, W

is linear if and only if:

- $\forall v_1, v_2 \in V$: $f(v_1 + v_2) = f(v_1) + f(v_2)$
- $\forall v \in V, \lambda \in F$: $f(\lambda v) = \lambda f(v)$

Constructing linear mappings:

A linear map is uniquely determined if we specify a mapping value for each basis vector of V.

Matrix Representation

Finite dimensional spaces

- Linear maps can be represented as matrices
- For each basis vector $\mathbf{v}_i$ of V, we specify the mapped vector $\mathbf{w}_i$.
- Then, the map f is given by:

$$f(\mathbf{v}) = f\left(\begin{pmatrix} v_1 \\ \vdots \\ v_n \end{pmatrix}\right) = v_1\mathbf{w}_1 + \ldots + v_n\mathbf{w}_n$$

Matrix Representation

This can be written as matrix-vector product:

$$f(v) = \begin{pmatrix} | & & | \\ \mathbf{w}_1 & \cdots & \mathbf{w}_n \\ | & & | \end{pmatrix} \cdot \begin{pmatrix} v_1 \\ \vdots \\ v_n \end{pmatrix}$$

The columns are the images of the basis vectors (for which the coordinates of **v** are given)

Linear Systems of Equations

Problem: Invert an affine map

- Given: $\mathbf{M}\mathbf{x} = \mathbf{b}$
- We know $\mathbf{M}$, $\mathbf{b}$
- Looking for $\mathbf{x}$

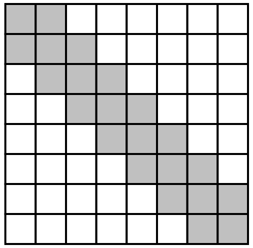
Solution

- Set of solutions: always an *affine subspace* of $\mathbb{R}^n$, or the empty set.
  - Point, line, plane, hyperplane...
- Innumerous algorithms for solving linear systems

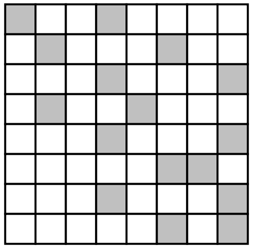
Solvers for Linear Systems

Algorithms for solving linear systems of equations:

- Gaussian elimination: $O(n^3)$ operations for $n \times n$ matrices
- We can do better, in particular for special cases:
  - **Band matrices:**
    constant bandwidth



  - **Sparse matrices:**
    constant number of non-zero entries per row
    - Store only non-zero entries
    - Instead of (3.5, 0, 0, 0, 7, 0, 0), store [(1:3.5), (5:7)]



Solvers for Linear Systems

Algorithms for solving linear systems of *n* equations:

- Band matrices, O(1) bandwidth:
  - Modified O(n) elimination algorithm.
- Iterative Gauss-Seidel solver
  - converges for diagonally dominant matrices
  - Typically: O(n) iterations, each costs O(n) for a sparse matrix.
- Conjugate Gradient solver
  - Only symmetric, positive definite matrices
  - Guaranteed: O(n) iterations
  - Typically good solution after O($\sqrt{n}$) iterations.

More details on iterative solvers: *J. R. Shewchuk: An Introduction to the Conjugate Gradient Method Without the Agonizing Pain, 1994.*

Eigenvectors & Eigenvalues

Definition:

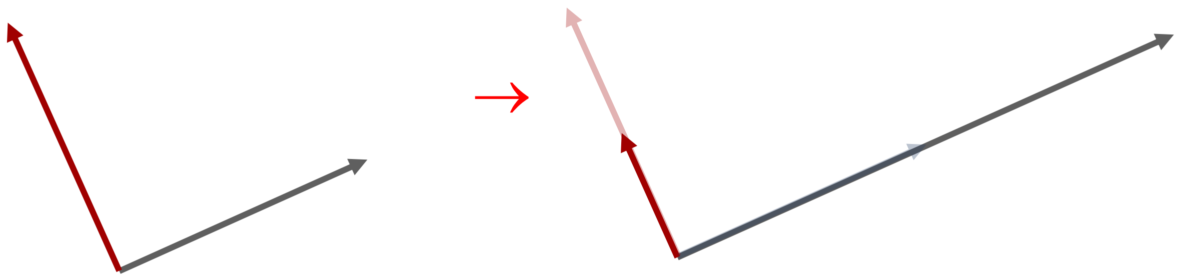
- Linear map $\mathbf{M}$, non-zero vector $\mathbf{x}$ with

  $\mathbf{Mx} = \lambda\mathbf{x}$
- λ an is *eigenvalue* of $\mathbf{M}$
- $\mathbf{x}$ is the corresponding *eigenvector*.

Example

Intuition:

- In the direction of an eigenvector, the linear map acts like a scaling



- Example: two eigenvalues (0.5 and 2)
- Two eigenvectors
- Standard basis contains no eigenvectors

Eigenvectors & Eigenvalues

Diagonalization:

In case an $n \times n$ matrix **M** has n linear independent eigenvectors, we can *diagonalize* **M** by transforming to this coordinate system: $\mathbf{M} = \mathbf{T}\mathbf{D}\mathbf{T}^{-1}$.

Spectral Theorem

Spectral Theorem:

If **M** is a symmetric $n \times n$ matrix of real numbers (i.e. **M** = $\mathbf{M}^T$), there exists an *orthogonal* set of n eigenvectors.

This means, every (real) symmetric matrix can be *diagonalized*:

$\mathbf{M} = \mathbf{T}\mathbf{D}\mathbf{T}^T$ with an orthogonal matrix **T**.

Computation

Simple algorithm

- "Power iteration" for symmetric matrices
- Computes largest eigenvalue even for large matrices
- Algorithm:
  - Start with a random vector (maybe multiple tries)
  - Repeatedly multiply with matrix
  - Normalize vector after each step
  - Repeat until ration before / after normalization converges (this is the eigenvalue)
- Intuition:
  - Largest eigenvalue = "dominant" component/direction

Powers of Matrices

What happens:

- A symmetric matrix can be written as:

$$\mathbf{M} = \mathbf{T}\mathbf{D}\mathbf{T}^{\mathrm{T}} = \mathbf{T}\begin{pmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{pmatrix}\mathbf{T}^{\mathrm{T}}$$

- Taking it to the *k*-th power yields:

$$\mathbf{M}^k = \mathbf{T}\mathbf{D}\mathbf{T}^{\mathrm{T}}\mathbf{T}\mathbf{D}\mathbf{T}^{\mathrm{T}}\ldots\mathbf{T}\mathbf{D}\mathbf{T}^{\mathrm{T}} = \mathbf{T}\mathbf{D}^k\mathbf{T}^{\mathrm{T}} = \mathbf{T}\begin{pmatrix} \lambda_1{}^k & & \\ & \ddots & \\ & & \lambda_n{}^k \end{pmatrix}\mathbf{T}^{\mathrm{T}}$$

- Bottom line: Eigenvalue analysis key to understanding powers of matrices.

Improvements

Improvements to the power method:

- Find smallest? – use inverse matrix.
- Find all (for a symmetric matrix)? – run repeatedly, orthogonalize current estimate to already known eigenvectors in each iteration (Gram Schmidt)
- How long does it take? – ratio to next smaller eigenvalue, gap increases exponentially.

There are more sophisticated algorithms based on this idea.

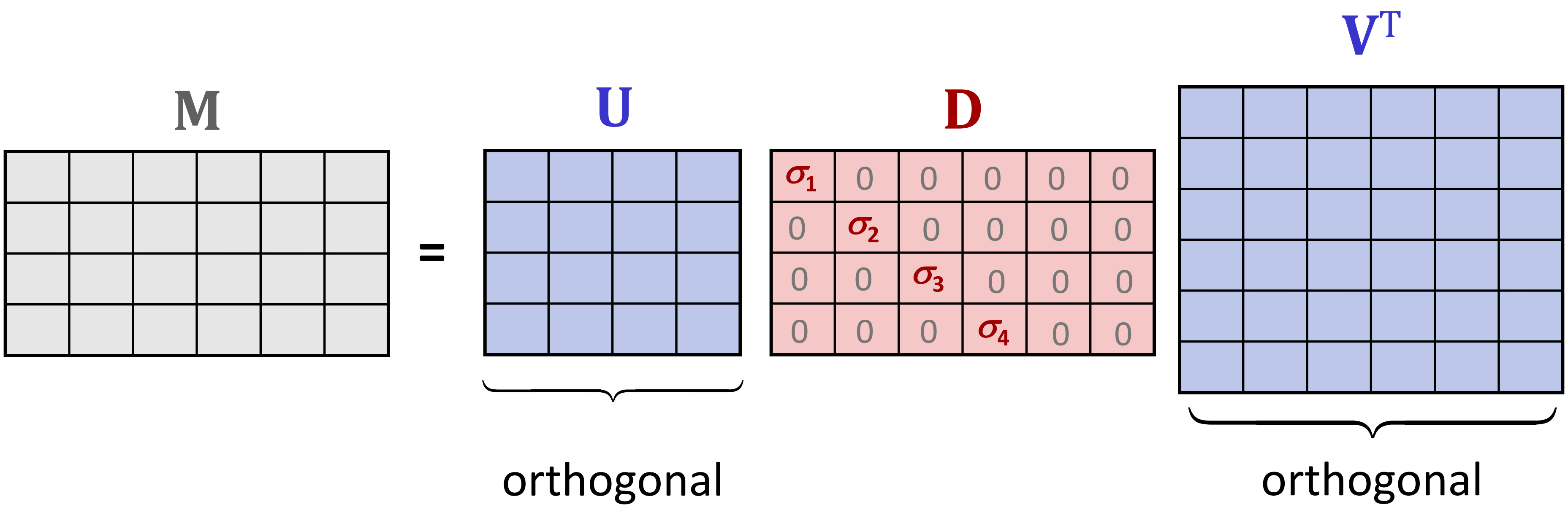
Generalization: SVD

Singular value decomposition:

- Let **M** be an arbitrary real matrix (may be rectangular)
- Then **M** can be written as:
  - $\mathbf{M} = \mathbf{U}\,\mathbf{D}\,\mathbf{V}^T$
  - The matrices **U**, **V** are orthogonal
  - **D** is a diagonal matrix (might contain zeros)
  - The diagonal entries are called *singular values*.
- **U** and **V** are different in general. For diagonalizable matrices, they are the same, and the singular values are the eigenvalues.

Singular Value Decomposition

Singular value decomposition



Singular Value Decomposition

Singular value decomposition

- Can be used to solve linear systems of equations
- For full rank, square **M**:

$$\mathbf{M} = \mathbf{U}\ \mathbf{D}\ \mathbf{V}^T$$

$$\Rightarrow \mathbf{M}^{-1} = (\mathbf{U}\ \mathbf{D}\ \mathbf{V}^T)^{-1} = (\mathbf{V}^T)^{-1}\ \mathbf{D}^{-1}\ (\mathbf{U}^{-1}) = \mathbf{V}\ \mathbf{D}^{-1}\ \mathbf{U}^T$$

- Good numerical properties (numerically stable)
- More expensive than iterative solvers
- The OpenCV library provides a very good implementation of the SVD

Linear Inverse Problems

Inverse Problems

Settings

- A (physical) process f takes place
- It transforms the original input **x** into an output **b**
- Task: recover **x** from **b**

Examples:

- 3D structure from photographs
- Tomography: values from line integrals
- 3D geometry from a noisy 3D scan

Linear Inverse Problems

Assumption: f is linear and finite dimensional

$f(\mathbf{x}) = \mathbf{b} \Rightarrow \mathbf{M}_f\mathbf{x} = \mathbf{b}$

Inversion of f is said to be an ill-posed problem, if one of the following three conditions hold:

- There is no solution
- There is more than one solution
- There is exactly one solution, but the SVD contains very small singular values.

Ill posed Problems

Ratio: Small singular values amplify errors

- Assume inexact input
  - Measurement noise
  - Numerical noise
- Reminder: $\mathbf{M}^{-1} = \mathbf{V}\ \mathbf{D}^{-1}\ \mathbf{U}^{T}$

does not hurt (orthogonal) — does not hurt (orthogonal)

this one is decisive

- Orthogonal transforms preserve norm of **x**, so **V** and **U** do not cause problems

Ill posed Problems

Ratio: Small singular values amplify errors

- Reminder: $\mathbf{x} = \mathbf{M}^{-1}\mathbf{b} = (\mathbf{V}\,\mathbf{D}^{-1}\,\mathbf{U}^{T})\mathbf{b}$
- Say **D** looks like that:

$$\mathbf{D} := \begin{pmatrix} 2.5 & 0 & 0 & 0 \\ 0 & 1.1 & 0 & 0 \\ 0 & 0 & 0.9 & 0 \\ 0 & 0 & 0 & 0.000000001 \end{pmatrix}$$

- Any input noise in **b** in the direction of the fourth right singular vector will be amplified by 10^9.
- If our measurement precision is less than that, the result will be unusable.
- Does *not* depend on *how* we invert the matrix.
- Condition number: $\sigma_{max} / \sigma_{min}$

Ill Posed Problems

Two problems:

(1) Mapping destroys information
- goes below noise level
- cannot be recovered by any means

(2) Inverse mapping amplifies noise
- yields garbage solution
- even remaining information not recovered
- extremely large random solutions are obtained

$$\mathbf{D} := \begin{pmatrix} 2.5 & 0 & 0 & 0 \\ 0 & 1.1 & 0 & 0 \\ 0 & 0 & 0.9 & 0 \\ 0 & 0 & 0 & 0.000000001 \end{pmatrix}$$

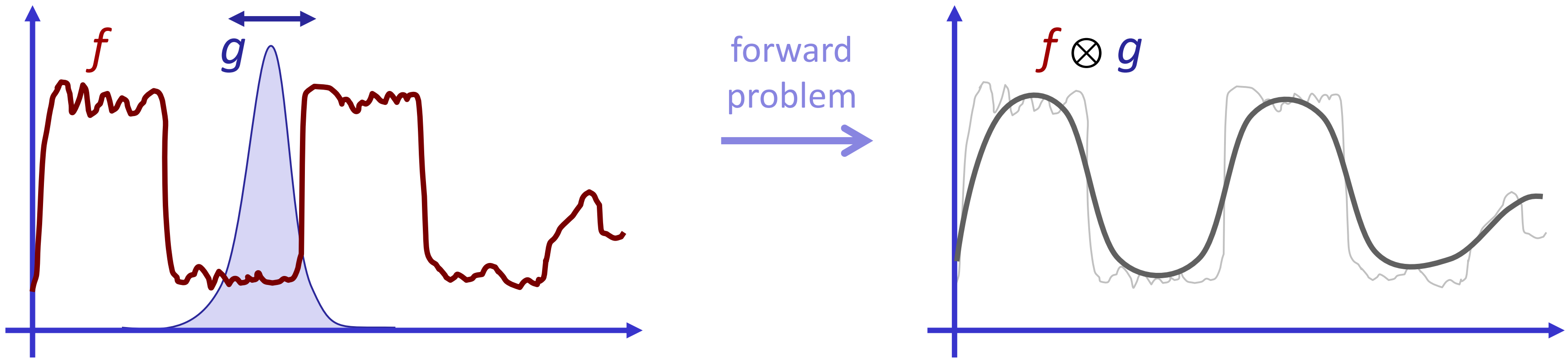
We can do something about problem #2

Regularization

Regularization

- Avoiding destructive noise caused by inversion
  - Various techniques
  - Goal: ignore the misleading information
- Subspace inversion:
  - Ignore subspace with small singular values
    - needs an SVD, risk of ringing
  - Additional assumptions:
    - smoothness (or something similar)
    - make compound problem (f^{-1} + assumptions) well posed
- We will look at this in *detail* later

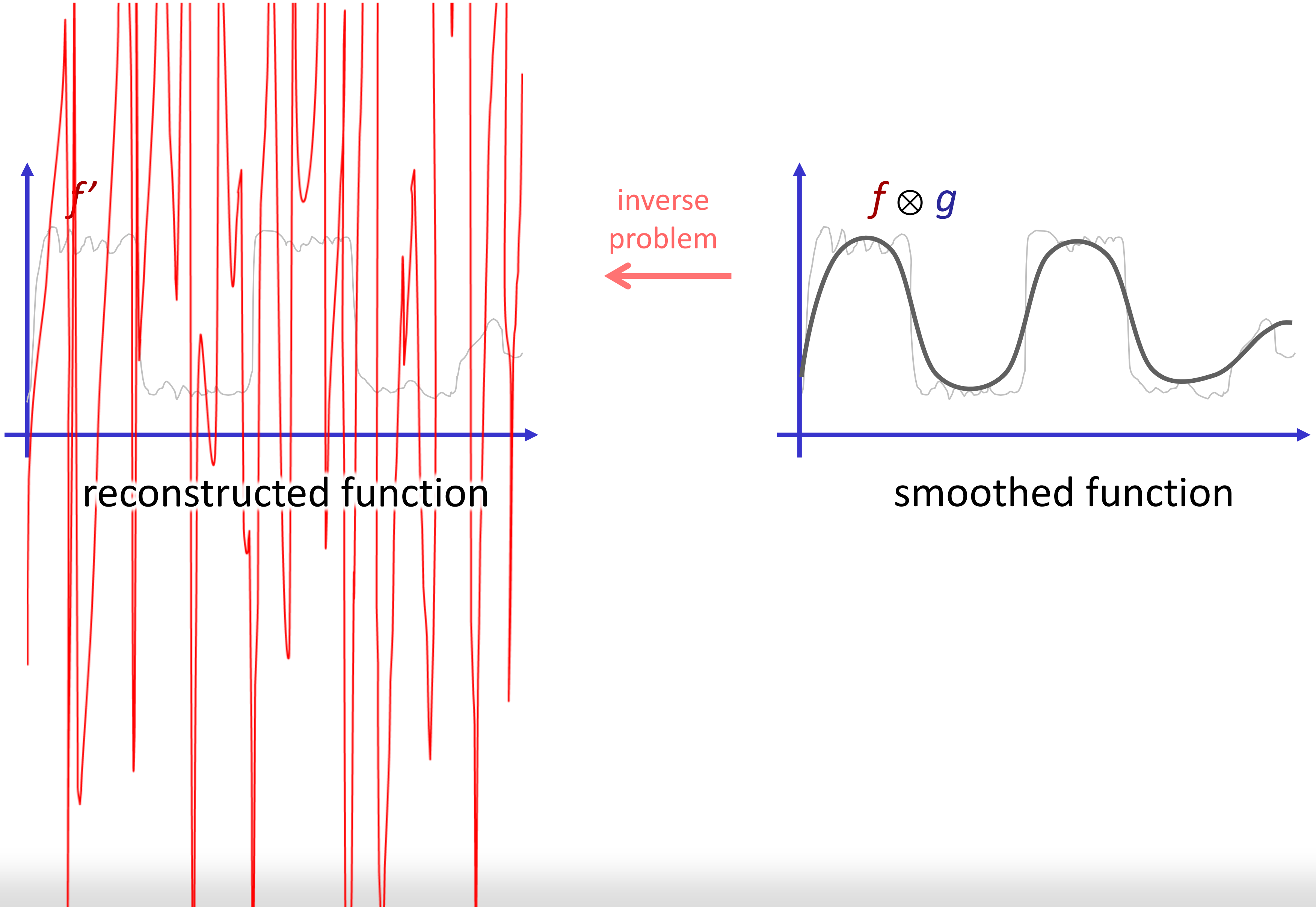
Illustration of the Problem



original function

smoothed function

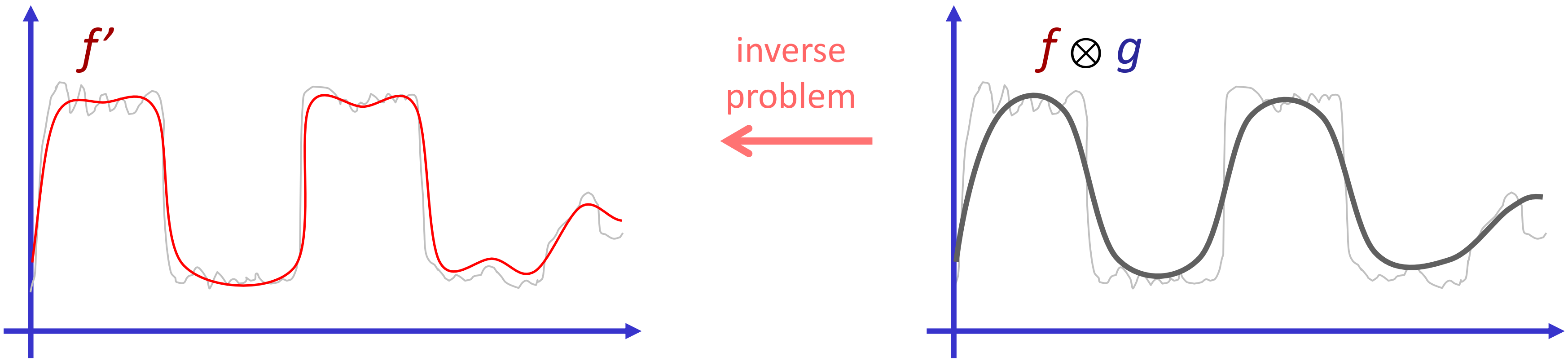
Illustration of the Problem



reconstructed function

smoothed function

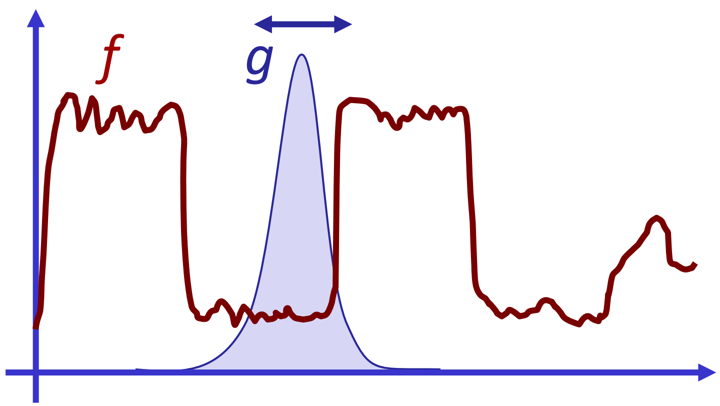
Illustration of the Problem



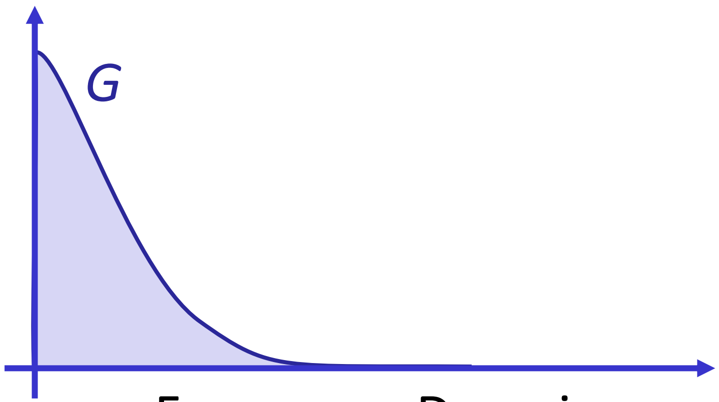
regularized
reconstructed function

smoothed function

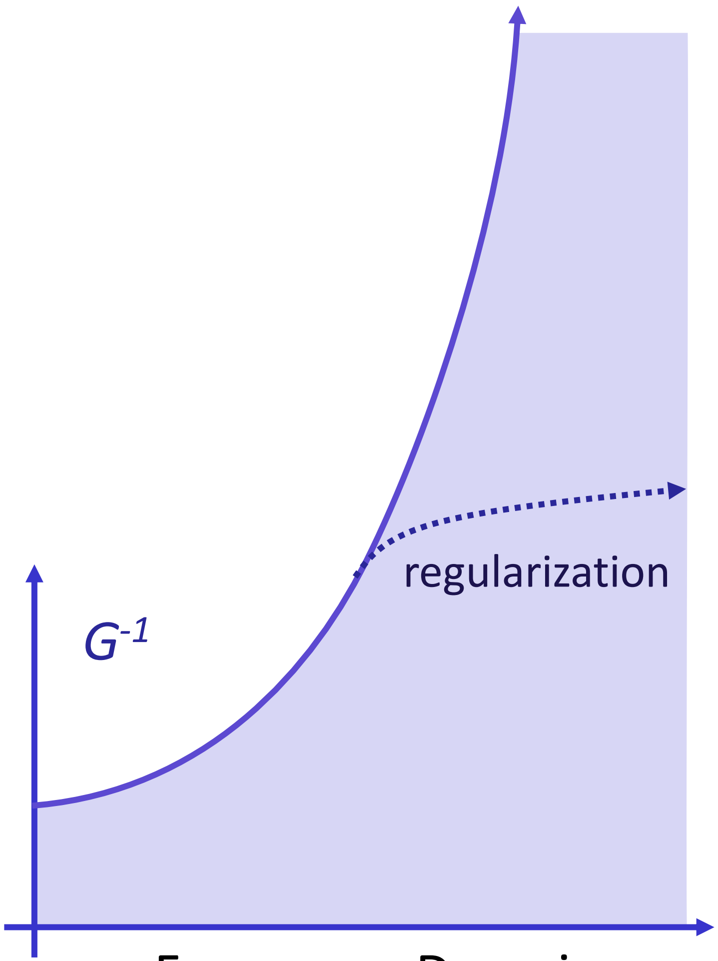
Illustration of the Problem



original function

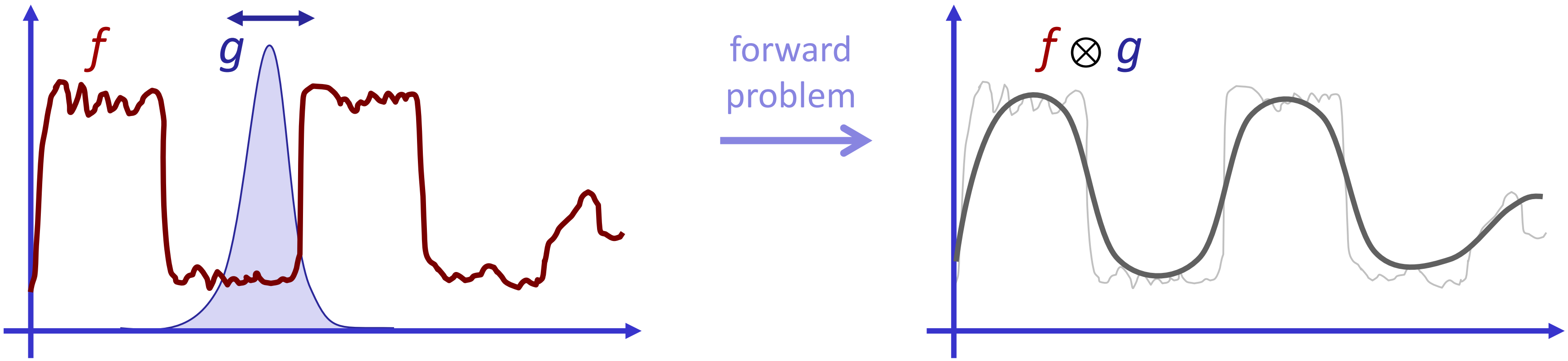


Frequency Domain



Frequency Domain

Illustration of the Problem

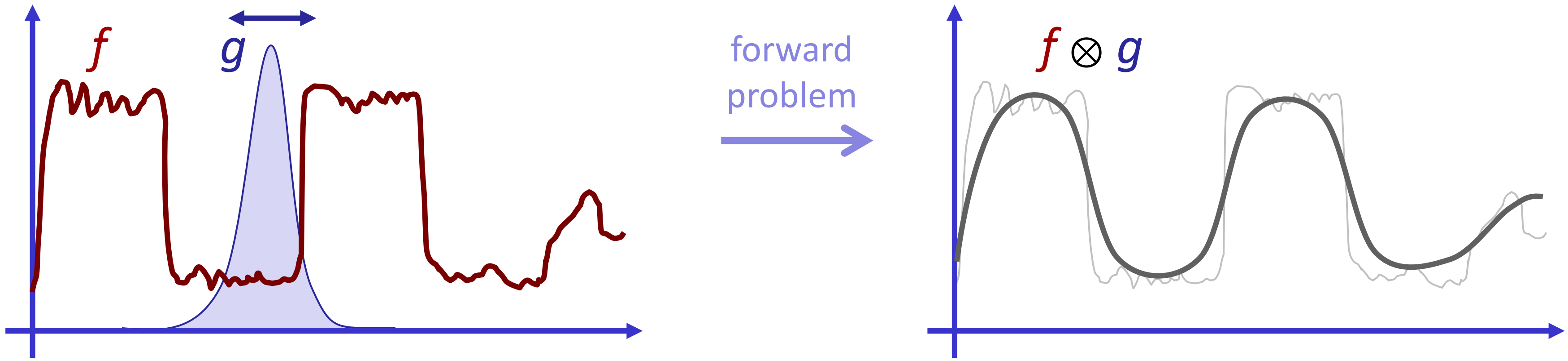


original function

smoothed function

$$\begin{pmatrix} 1.2 \\ 1.5 \\ 0.3 \\ 0.4 \\ 1.6 \\ 0.2 \\ 0.3 \end{pmatrix} * \frac{1}{3} \begin{pmatrix} 2 & 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 2 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 2 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 2 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 2 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 2 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 & 2 \end{pmatrix} = \begin{pmatrix} 1.4 \\ 1.5 \\ 0.8 \\ 0.9 \\ 1.3 \\ 0.8 \\ 0.7 \end{pmatrix}$$

Illustration of the Problem



original function · smoothed function

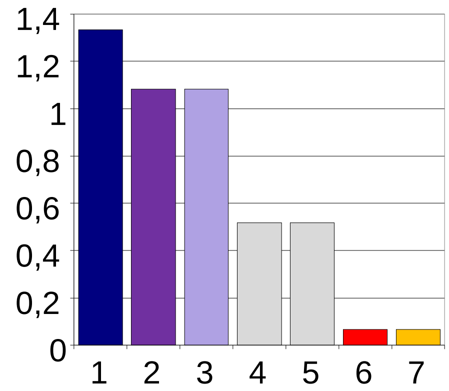
$$
\begin{pmatrix} 1.4 \\ 1.5 \\ 0.8 \\ 0.9 \\ 1.3 \\ 0.8 \\ 0.7 \end{pmatrix} * 3 \begin{pmatrix} 2 & 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 2 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 2 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 2 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 2 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 2 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 & 2 \end{pmatrix}^{-1} = \begin{pmatrix} 1.4 \\ 1.5 \\ 0.8 \\ 0.9 \\ 1.2 \\ 0.8 \\ 0.7 \end{pmatrix} \qquad \begin{pmatrix} 1.2 \\ 1.5 \\ 0.3 \\ 0.4 \\ 1.6 \\ 0.2 \\ 0.3 \end{pmatrix}
$$

solution (from 2 digits) · correct

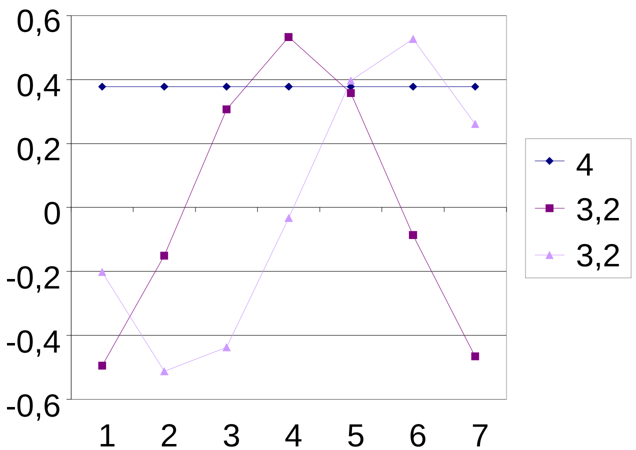
Analysis

$$\frac{1}{3}\begin{pmatrix} 2 & 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 2 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 2 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 2 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 2 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 2 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 & 2 \end{pmatrix}$$

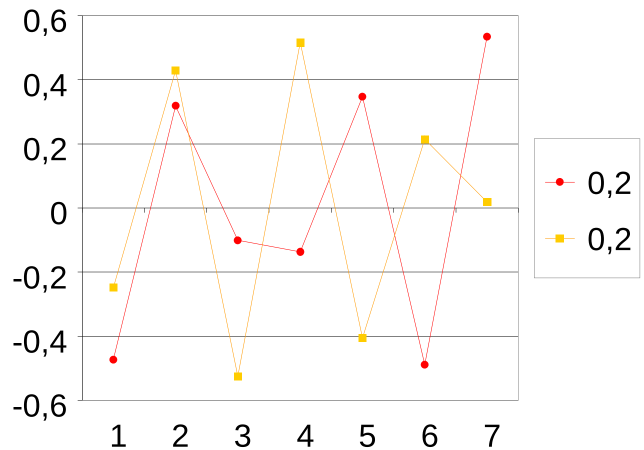
Matrix



Spectrum



Dominant Eigenvectors



Smallest Eigenvectors

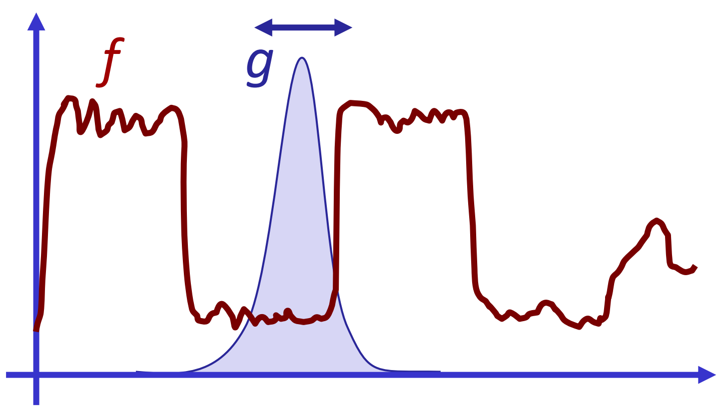
Digging Deeper...

Let's look at this example again:

- Convolution operation:

$$f'(x) = \int_{\mathbb{R}} f(t) \cdot g(t-x)dt$$

- Shift-invariant linear operator



continuous

$$\frac{1}{3}\begin{pmatrix} 2 & 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 2 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 2 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 2 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 2 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 2 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 & 2 \end{pmatrix}$$

discrete

Convolution Theorem

Fourier Transform

- Function space:

$$f: [0..2\pi] \rightarrow \mathbb{R} \text{ (suff. smooth)}$$

- Fourier basis:

$$\{1, \cos kx, \sin kx \mid k = 1,2, \ldots\}$$

- Properties
  - The Fourier basis is an orthogonal basis (standard scalar prod.)
  - The Fourier basis diagonalizes all shift-invariant linear operators

Convolution Theorem

This means:

- Let $F: \mathbb{N} \to \mathbb{R}$ be the Fourier transform (FT) of
  $f: [0..2\pi] \to \mathbb{R}$
- Then: $f \otimes g = FT^{-1}(F \cdot G)$

Consequences

- Fourier spectrum of convolution operator ("filter") determines well-posedness of inverse operation (deconvolution)
- Certain frequencies might be lost
  - In our example: high frequencies

Quadratic Forms

Multivariate Polynomials

A *multi-variate* polynomial of total degree *d*:

- A function $f: \mathbb{R}^n \rightarrow \mathbb{R}, \quad \mathbf{x} \rightarrow f(\mathbf{x})$
- f is a polynomial in the components of $\mathbf{x}$
- Any 1D direction $f(\mathbf{s} + t\mathbf{r})$ is a polynomial of maximum degree d in t.

Examples:

- $f(\mathrm{x}, \mathrm{y}) := \mathrm{x} + \mathrm{xy} + \mathrm{y}$ is of total degree 2. In diagonal direction, we obtain $f(t[1/\sqrt{2}, 1/\sqrt{2}]^{\mathrm{T}}) = t^2$.
- $f(\mathrm{x}, \mathrm{y}) := c_{20}\mathrm{x}^2 + c_{02}\mathrm{y}^2 + c_{11}\mathrm{xy} + c_{10}\mathrm{x} + c_{01}\mathrm{y} + c_{00}$ is a quadratic polynomial in two variables

Quadratic Polynomials

In general, any quadratic polynomial in *n* variables can be written as:

- $\mathbf{x}^T \mathbf{A} \mathbf{x} + \mathbf{b}^T \mathbf{x} + c$
- $\mathbf{A}$ is an $n \times n$ matrix, $\mathbf{b}$ is an n-dim. vector, c is a number
- Matrix $\mathbf{A}$ can always be chosen to be symmetric
- If it isn't, we can substitute by $0.5 \cdot (\mathbf{A} + \mathbf{A}^T)$, not changing the polynomial

Example

Example:

$$f\left(\begin{pmatrix} x \\ y \end{pmatrix}\right) = f(\mathbf{x}) = \mathbf{x}^{\mathrm{T}} \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} \mathbf{x}$$

$$= [x\ y] \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = [x\ y] \begin{pmatrix} 1x & 2y \\ 3x & 4y \end{pmatrix}$$

$$= x1x + x2y + y3x + y4y$$

$$= 1x^2 + (2+3)xy + 4y^2$$

$$= 1x^2 + (2.5+2.5)xy + 4y^2$$

$$= \mathbf{x}^{\mathrm{T}} \frac{1}{2} \left[\begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} + \begin{pmatrix} 1 & 3 \\ 2 & 4 \end{pmatrix} \right] \mathbf{x} = \mathbf{x}^{\mathrm{T}} \begin{pmatrix} 1 & 2.5 \\ 2.5 & 4 \end{pmatrix} \mathbf{x}$$

Quadratic Polynomials

Specifying quadratic polynomials:

- $\mathbf{x}^{\mathrm{T}}\mathbf{A}\mathbf{x} + \mathbf{b}^{\mathrm{T}}\mathbf{x} + c$
- **b** shifts the function in space (if **A** has full rank):

$$(x-\mu)^{\mathrm{T}}\mathbf{A}(x-\mu)+c$$

$$= x^{\mathrm{T}}\mathbf{A}x - \mu^{\mathrm{T}}\mathbf{A}x - x^{\mathrm{T}}\mathbf{A}\mu + \mu\cdot\mu + c$$

$$\overset{(\text{A sym.})}{=} x^{\mathrm{T}}\mathbf{A}x - \underbrace{(2\mathbf{A}\mu)\mathbf{x}}_{=\,\mathbf{b}} + \mu\cdot\mu + c$$

- c is an additive constant

Some Properties

Important properties

- Multivariate polynomials form a vector space
- We can add them component-wise:

$$
\begin{array}{rl}
 & 2x^2 + 3y^2 + 4xy + 2x + 2y + 4 \\
+ & 3x^2 + 2y^2 + 1xy + 5x + 5y + 5 \\
\hline\hline
= & 5x^2 + 5y^2 + 5xy + 7x + 7y + 9
\end{array}
$$

- In vector notation:

$$
\begin{aligned}
& \mathbf{x}^{\mathrm{T}}\mathbf{A}_1\mathbf{x} + \mathbf{b}_1^{\mathrm{T}}\mathbf{x} + c_1 \\
+ \; & \lambda(\mathbf{x}^{\mathrm{T}}\mathbf{A}_2\mathbf{x} + \mathbf{b}_2^{\mathrm{T}}\mathbf{x} + c_2) \\
= \; & \mathbf{x}^{\mathrm{T}}(\mathbf{A}_1 + \lambda\mathbf{A}_2)\mathbf{x} + (\mathbf{b}_1 + \lambda\mathbf{b}_2)^{\mathrm{T}}\mathbf{x} + (c_1 + \lambda c_2)
\end{aligned}
$$

Quadratic Polynomials

Quadrics

- Zero level set of a quadratic polynomial: “quadric”
- Shape depends on eigenvalues of $\mathbf{A}$
- $\mathbf{b}$ shifts the object in space
- c sets the level

Shapes of Quadrics

Shape analysis:

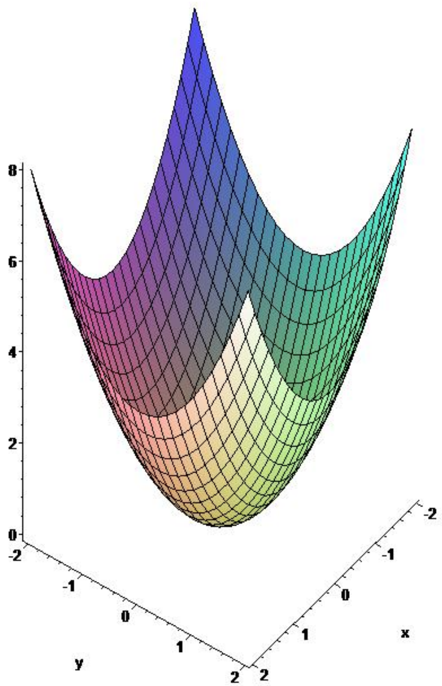
- **A** is symmetric
- **A** can be *diagonalized* with orthogonal *eigenvectors*

$$\mathbf{x}^{\mathrm{T}}\mathbf{A}\mathbf{x} = x^{\mathrm{T}}\left[\mathbf{Q}^{\mathrm{T}}\begin{pmatrix}\lambda_1 & & \\ & \ddots & \\ & & \lambda_n\end{pmatrix}\mathbf{Q}\right]x$$

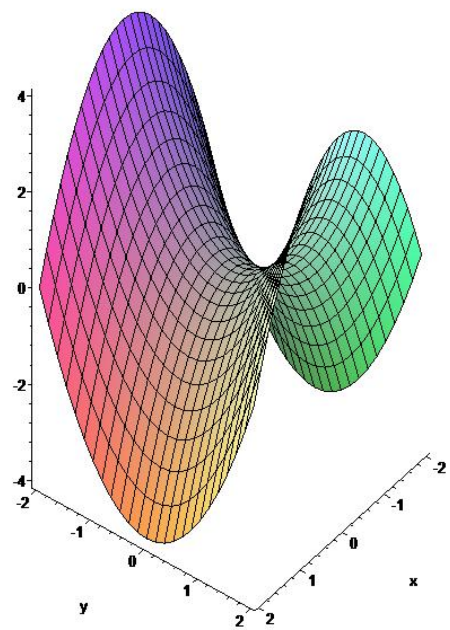
$$= (\mathbf{Q}x)^{\mathrm{T}}\begin{pmatrix}\lambda_1 & & \\ & \ddots & \\ & & \lambda_n\end{pmatrix}(\mathbf{Q}x)$$

- **Q** contains the principal axis of the quadric
- The eigenvalues determine the quadratic growth (up, down, speed of growth)

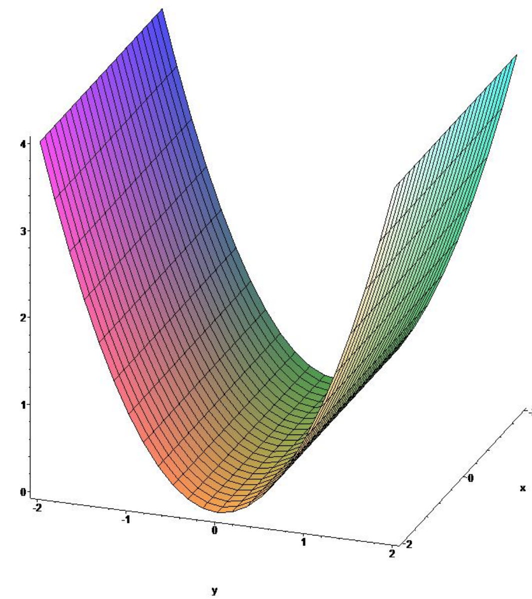
Shapes of Quadratic Polynomials



$\lambda_1 = 1, \lambda_2 = 1$



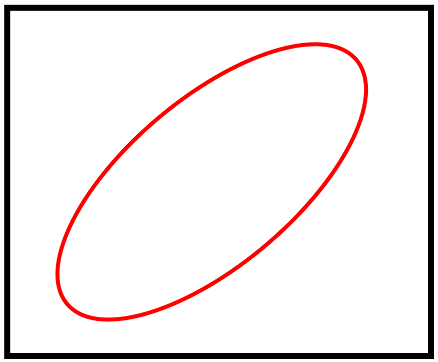
$\lambda_1 = 1, \lambda_2 = -1$



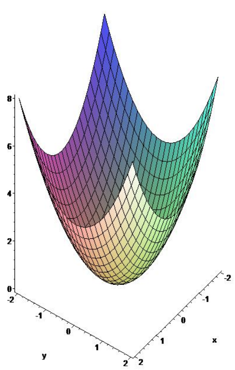
$\lambda_1 = 1, \lambda_2 = 0$

The Iso-Lines: Quadrics

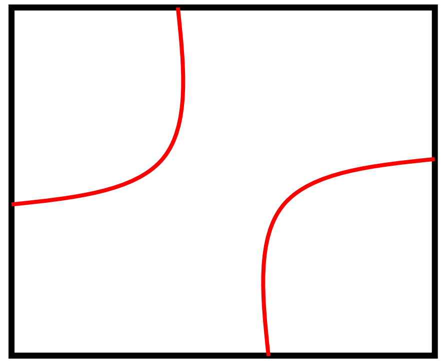
elliptic



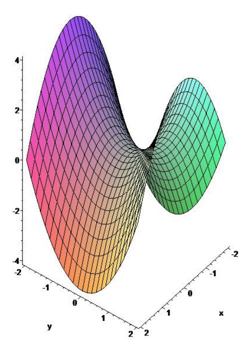
$\lambda_1 > 0, \lambda_2 > 0$



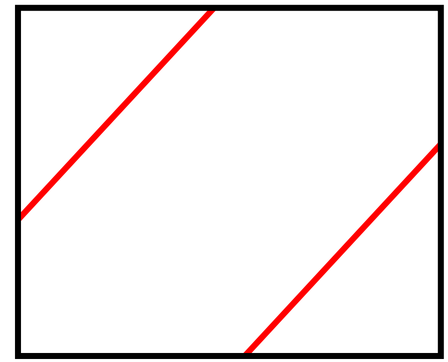
hyperbolic



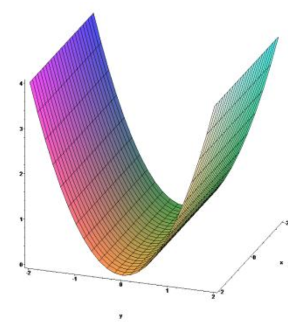
$\lambda_1 < 0, \lambda_2 > 0$



degenerate case



$\lambda_1 = 0, \lambda_2 \neq 0$

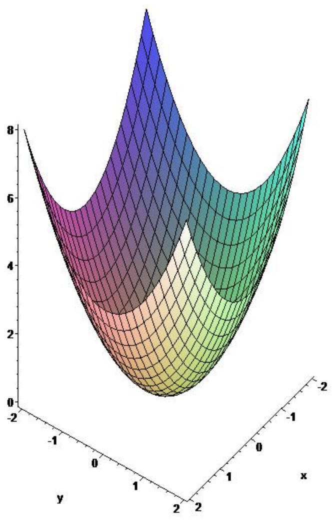


Quadratic Optimization

Quadratic Optimization

- Minimize quadratic objective function

$$\mathbf{x}^{\mathrm{T}}\mathbf{A}\mathbf{x} + \mathbf{b}^{\mathrm{T}}\mathbf{x} + c$$



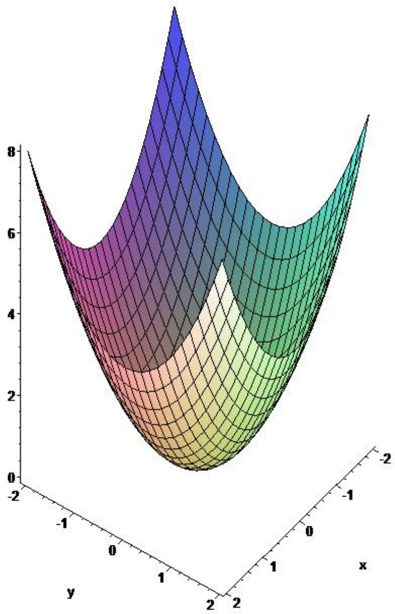
- Required: $\mathbf{A} > 0$ (only positive eigenvalues)
  - It's a paraboloid with a unique minimum
  - The vertex (critical point) can be determined by simply solving a linear system
- Necessary and sufficient condition

$$2\mathbf{A}\mathbf{x} = -\mathbf{b}$$

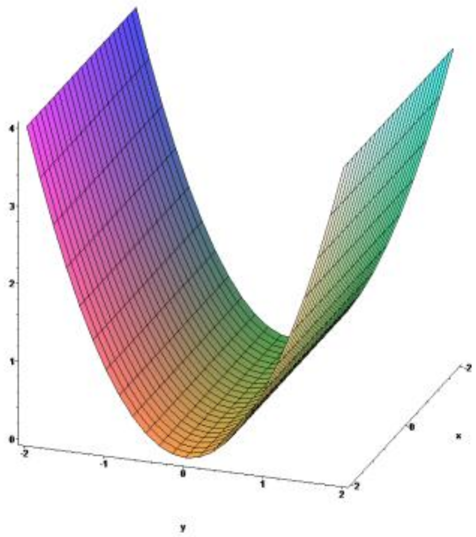
Condition Number

How stable is the solution?

- Depends on Matrix $\boldsymbol{A}$



good

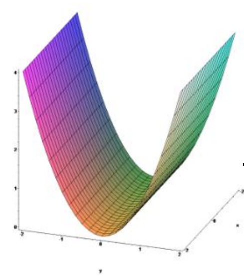


bad

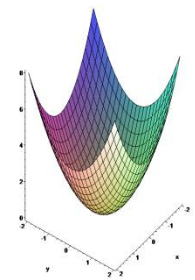
Regularization

Regularization

- Sums of positive semi-definite matrices are positive semi-definite
- Add regularizing quadric
  - "Fill in the valleys"
  - Bias in the solution



$\mathbf{A} + \lambda\mathbf{I}$



Example

- Original: $\mathbf{x}^{\mathrm{T}}\mathbf{A}\mathbf{x} + \mathbf{b}^{\mathrm{T}}\mathbf{x} + c$
- Regularized: $\mathbf{x}^{\mathrm{T}}(\mathbf{A} + \lambda\mathbf{I})\mathbf{x} + \mathbf{b}^{\mathrm{T}}\mathbf{x} + c$

Rayleigh Quotient

Relation to eigenvalues:

- Min/max eigenvalues of a symmetric **A** expressed as constraint quadratic optimization:

$$\lambda_{\min} = \min \frac{\mathbf{x}^{\mathrm{T}}\mathbf{A}\mathbf{x}}{\mathbf{x}^{\mathrm{T}}\mathbf{x}} = \min_{\|x\|=1} \left(\mathbf{x}^{\mathrm{T}}\mathbf{A}\mathbf{x}\right) \qquad \lambda_{\max} = \max \frac{\mathbf{x}^{\mathrm{T}}\mathbf{A}\mathbf{x}}{\mathbf{x}^{\mathrm{T}}\mathbf{x}} = \max_{\|x\|=1} \left(\mathbf{x}^{\mathrm{T}}\mathbf{A}\mathbf{x}\right)$$

- The other way round – eigenvalues solve a certain type of constrained, (non-convex) optimization problem.

Coordinate Transformations

One more interesting property:

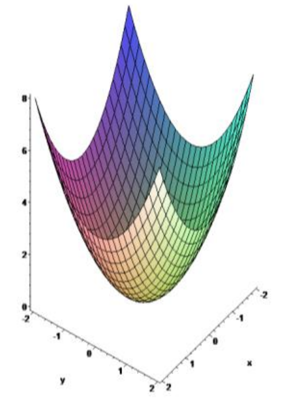
- Given a positive definite symmetric ("SPD") matrix **M** (all eigenvalues positive)
- Such a matrix can always be written as square of another matrix:

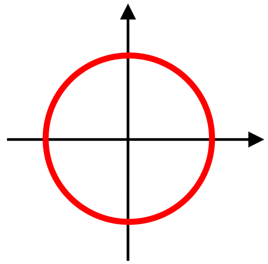
$$\mathbf{M} = \mathbf{T}\mathbf{D}\mathbf{T}^{\mathrm{T}} = \left(T\sqrt{D}\right)\left(\sqrt{D}^{T}T^{T}\right) = \left(T\sqrt{D}\right)\left(T\sqrt{D}\right)^{T} = \left(T\sqrt{D}\right)^{2}$$

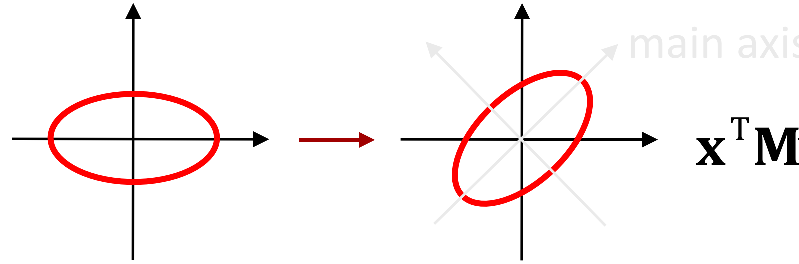
$$\sqrt{D} = \begin{pmatrix} \sqrt{\lambda_1} & & \\ & \ddots & \\ & & \sqrt{\lambda_n} \end{pmatrix}$$

SPD Quadrics

$\mathbf{x}^{\mathrm{T}}\mathbf{x}$







$\mathbf{x}^{\mathrm{T}}\mathbf{M}\mathbf{x}$

Identity $\mathbf{I}$

$$\mathbf{M} = \mathbf{T}\mathbf{D}\mathbf{T}^{\mathrm{T}} = \left(T\sqrt{D}\right)^2$$

Interpretation:

- Start with a unit positive quadric $\mathbf{x}^{\mathrm{T}}\mathbf{x}$.
- Scale the main axis (diagonal of D)
- Rotate to a different coordinate system (columns of **T**)
- Recovering main axis from **M**: Compute eigensystem ("principal component analysis")

Why should I care?

What are quadrics good for?

- *log-probability* of Gaussian models
- Estimation in Gaussian probabilistic models...
  - ...is quadratic optimization.
  - ...is solving of linear systems of equations.
- Quadratic optimization
  - easy to use & solve
  - feasible :-)
- Approximate more complex models locally

Gaussian normal distribution

$$p_{\mu,\sigma}(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

Groups and Transformations

Groups

Definition:

A set G with operation $\otimes$ is called a group $(G, \otimes)$, iff:

- **Closed:** $\otimes: G \times G \rightarrow G$ (always maps back to G)
- **Associativity:** $(f \otimes g) \otimes h = f \otimes (g \otimes h)$
- **Neutral element:** there exists $id \in G$ such that for any $g \in G$: $g \otimes id = g$
- **Inverse element:** For each $g \in G$ there exists $g^{-1} \in G$ such that $g \otimes g^{-1} = g^{-1} \otimes g = id$

Abelian Groups

- The group is *commutative* iff always $f \otimes g = g \otimes f$

Examples of Groups

Examples:

- G = invertible matrices, $\otimes$ = composition (matrix mult.)
- G = invertible affine transformation of $\mathbb{R}^d$, $\otimes$ = composition
  (matrix form: homogeneos coordinates)
- G = bijections of a set S to itself, $\otimes$ = composition
  - G = smooth C^k bijections of a set S to itself, $\otimes$ = composition
  - G = global symmetry transforms of a shape, $\otimes$ = composition
  - G = permutation of a discrete set, $\otimes$ = composition

Examples of Groups

Examples:

- G = invertible matrices, $\otimes$ = composition (matrix mult.)
- G = invertible affine transformation of $\mathbb{R}^d$

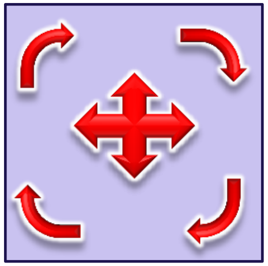
  Subgroups:

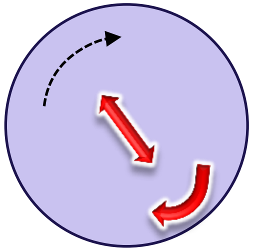
  - G = similarity transform (translation, rotation, mirroring, scaling $\neq 0$)
  - **E(d):** G = rigid motions (translation, rotation, mirroring)
  - **SE(d):** G = rigid motions (translation, rotation)
  - **O(d):** G = orthogonal matrix (rotation, mirroring) (columns/rows *orthonormal*)
  - **SO(d):** G = orthogonal matrix (rotation) (columns/rows orthonormal, determinant 1)
  - G = translations (the only commutative group out of these)

Examples of Groups

Examples:

- *G* = global symmetry transforms of a 2D shape

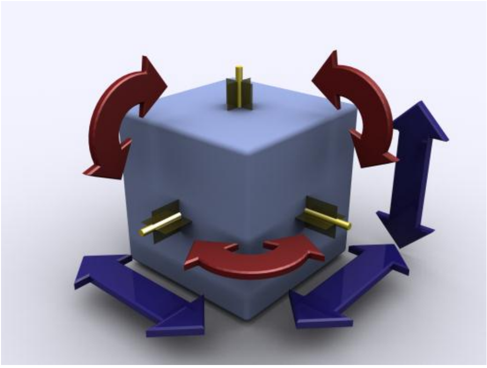


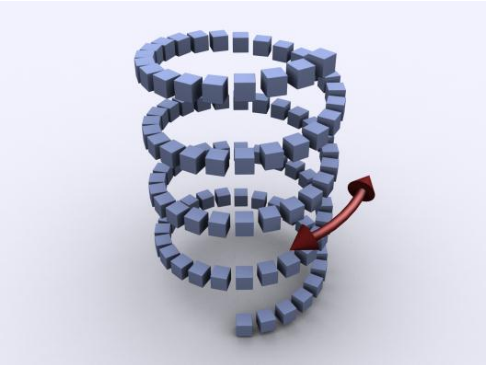


Examples of Groups

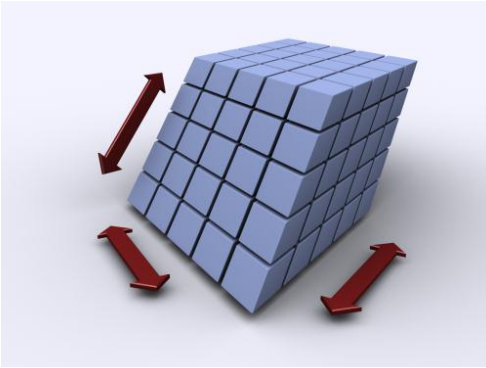
Examples:

- *G* = global symmetry transforms of a 3D shape





(extended to infinity)



(extended to infinity)

Outlook

More details on this later

- Symmetry groups
- Structural regularity
- Crystalographic groups and regular lattices