Geometric Modeling
Summer Semester 2010

Mathematical Tools (1)
Recap: Linear Algebra
Today...

Topics:

- Mathematical Background
  - Linear algebra
  - Analysis & differential geometry
  - Numerical techniques
Mathematical Tools
Linear Algebra
Overview

Linear algebra

- Vector spaces
- Linear maps
- Quadrics
Vectors

Vector spaces

- Vectors, Coordinates & Points
- Formal definition of a vector space
- Vector algebra
- Generalizations:
  - Infinite dimensional vector spaces
  - Function spaces
  - Approximation with finite dimensional spaces
- More Tools:
  - Dot product and norms
  - The cross product
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

Many classes of objects share the same *structure*:

- Geometric Objects
  - 1,2,3,4... dimensional Euclidian vectors
- But also a lot of other mathematical objects
  - Vectors with complex numbers, or finite fields
  - Certain sets of functions
  - Polynomials
  - ...
- Approach the problem from a more abstract level
  - More general: Saves time, reduces number of proofs
  - Can still resort to geometric vectors to get an intuition about what’s going on
Vector Spaces

Definition: *Vector Space $V$ over a Field $F$*

- Consists of a set of vectors $V$
- $F$ is a field (usually: Real numbers, $F = \mathbb{R}$)
- Provides two operations:
  - Adding vectors $u = v + w$ ($u, v, w \in V$)
  - Scaling vectors $w = \lambda v$ ($u \in V, \lambda \in F$)
- The two operations are *closed*, i.e.: operations on any elements of the vector space will yields elements of the vector space itself.
- ...and finally: A number of *properties* that have to hold:
Vector Spaces

Definition: *Vector Space $V$ over a Field $F$* (cont.)

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

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

(a3) $\exists 0_v \in V : \forall v \in V : v + 0_v = v$

(a4) $\forall v \in V : \exists w \in V : v + w = 0_v$

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

(s2) for $1_F \in F : \forall v \in V : 1_F v = v$

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

(s4) $\forall \lambda, \mu \in F, v \in V : (\lambda + \mu)v = \lambda v + \mu v$
Vector Spaces

Vector spaces

- Out of these formal assumptions, a long list of derivative properties (theorems) can be deduced.
- Will hold for any vector space.
- In particular, we will see that the assumptions are sufficient to obtain the columns with coordinates, we started with (in the finite dimensional case).
Properties

Some properties you can easily prove:

- The zero vector $\mathbf{0}_v$ is unique. For 2D vectors: $\mathbf{0}_v = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$

- Multiplication with the scalar $0_F$ yields the zero vector.

- The additive inverse $-\mathbf{v}$ is unique given $\mathbf{v}$.

- Multiplication by $-1$ yields the inverse vector.

- And so on...
Span and Basis

span \{v\} – line through the origin

\{u,v\} \subseteq \{u,v,w\}
form a basis of \( \mathbb{R}^2 \)

span \{u,v,w\} = \mathbb{R}^2
Example Spaces

Examples of finite-dimensional vector spaces:

- Of course: \( \mathbb{R}, \mathbb{R}^2, \mathbb{R}^3, \mathbb{R}^4 \ldots \)

- Standard basis of \( \mathbb{R}^3 \): \[
\begin{bmatrix}
1 \\
0 \\
0
\end{bmatrix}, \begin{bmatrix}
0 \\
1 \\
0
\end{bmatrix}, \begin{bmatrix}
0 \\
0 \\
1
\end{bmatrix}
\]

- Coordinates:
\[
\begin{pmatrix}
x \\
y \\
z
\end{pmatrix} = x \begin{pmatrix}
1 \\
0 \\
0
\end{pmatrix} + y \begin{pmatrix}
0 \\
1 \\
0
\end{pmatrix} + z \begin{pmatrix}
0 \\
0 \\
1
\end{pmatrix} = xi + yj + zk
\]
Example Spaces

Examples of finite-dimensional vector spaces:

• Polynomials of fixed degree
  
  ▪ For example, all polynomials of 2nd order:

    general form: \( ax^2 + bx + c \)

    addition: \((a_1x^2 + b_1x + c_1) + (a_2x^2 + b_2x + c_2)\)
    
    \[ \begin{align*}
    \text{addition: } & (a_1x^2 + b_1x + c_1) + (a_2x^2 + b_2x + c_2) \\
    & = (a_1 + a_2)x^2 + (b_1 + b_2)x + (c_1 + c_2)
    \end{align*} \]

    scalar multiplication: \( \lambda(ax^2 + bx + c) = (\lambda a)x^2 + (\lambda b)x + (\lambda c) \)

  ▪ Might be confusing: Evaluation of polynomials at \( x \) is non-linear, does not relate to the vector space structure

  ▪ Coordinates: \([a, b, c]^T\)

  ▪ Basis for these coordinates: \(\{x^2, x, 1\}\)
Example Spaces

Infinite-dimensional vector spaces:

- Polynomials (of any degree)
- Need to represent coefficients of arbitrary degree
- Coordinate vectors can potentially become arbitrarily long

- General form: \( poly(x) = \sum_{i=0}^{\infty} a_i x^i \) (only a finite subset of the \( a_i \) nonzero)

- Basis: \( \{x^i \mid i = 0,1,2,...\} \)

- Coordinate vectors: \( (a_0,a_1,a_2,a_3,...) \)
Spaces of Sequences

First generalization:

- Make vectors infinitely long
- Spaces of sequences

\[
\begin{pmatrix}
  x \\
  y \\
  z
\end{pmatrix}
\rightarrow
\begin{pmatrix}
  a_1 \\
  a_2 \\
  a_3 \\
  \vdots \\
  a_i \\
  \vdots
\end{pmatrix}
\]

- Dimension = $\infty$, countable
Example Spaces

More infinite-dimensional vector spaces:

- Function spaces
  - Space of all functions $f: \mathbb{R} \rightarrow \mathbb{R}$
  - Space of all smooth $C^k$ functions $f: \mathbb{R} \rightarrow \mathbb{R}$
  - Space of all functions $f: [0..1] \rightarrow \mathbb{R}$
  - Not a vector space: $f: [0..1] \rightarrow [0..1]$
Function Spaces

Vector operations

For $f: \Omega \rightarrow \mathbb{R}$, define:

- $(f + g)(x) := f(x) + g(x)$ ($\forall x \in \Omega$)
- $(\lambda f)(x) := \lambda(f(x))$ ($\forall x \in \Omega$)

The zero vector is:

- $0_V = (f: f(x) \equiv 0)$
Function Spaces

Intuition:

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

\[
\begin{align*}
[f_1, f_2, \ldots, f_9]^T & \quad [f_1, f_2, \ldots, f_{18}]^T & \quad f(x)
\end{align*}
\]
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

Here is the “recipe”:

- We are given an infinite-dimensional function space $V$.
- We are looking for $f \in V$ with a certain property.
- From a function space $V$ we choose linearly independent functions $f_1, \ldots, f_d \in V$ to form the $d$-dimensional subspace $\text{span}\{f_1, \ldots, f_d\}$.
- Instead of looking for the $f \in V$, we look only among the $\tilde{f} := \sum_{i=1}^{d} \lambda_i f_i$ for a function that best-matches the desired property (might be just an approximation, though).
- The good thing: $\tilde{f}$ is described by $(\lambda_1, \ldots, \lambda_d)$. Good for the computer...
Approximation of Function Spaces

Two Approaches:

- Construct a basis, that already provides a subspace containing the functions you want
  - Typically, the coefficients will have an intuitive meaning then
  - Bezier Splines, B-Splines, NURBS are all about that

- Choose a basis that can approximate the functions you might want, then pick the closest
  - Standard approach in numerical solutions to *partial differential equations* and *integral equations*
  - Basic idea: Define a measure of correctness $C(f)$, then try to maximize $C(\tilde{f})$
Typical Basis Sets:

• Consider the space of functions $f: [a, b] \rightarrow \mathbb{R}$.

• Some $d$-dimensional subspaces:
  
  ▪ span $\{1, x, x^2, ..., x^{d-1}\}$ (Monomial basis of degree $d-1$)
  
  ▪ span $\{\sin x, \cos x, \sin 2x, \cos 2x, ..., \sin (d-1)x/2, \cos (d-1)x/2\}$
    (Fourier basis of order $(d-1)/2$, usually $a = 0, b = 2\pi$)

• It depends all on the application, of course...
Examples

Monomial basis

Fourier basis
More Tools for Vectors

More operations:

- Dot product / scalar product / inner product (measures distances, angles)
- Cross product (only $\mathbb{R}^3$)
The standard dot product for vectors $v, w \in \mathbb{R}^d$ is defined as:

$$v \cdot w = \langle v, w \rangle = v^T w := \sum_{i=1}^{d} v_i w_i$$

For $v, w \in \mathbb{R}^3$:

$$v \cdot w = \begin{pmatrix} v_x \\ v_y \\ v_z \end{pmatrix} \cdot \begin{pmatrix} w_x \\ w_y \\ w_z \end{pmatrix} = v_x w_x + v_y w_y + v_z w_z$$
Properties

Geometric properties:

- $length(v) := \|v\|_2 = \sqrt{v \cdot v}$ (Pythagoras)

- $|v \cdot w| = \|v\| \cdot \|w\| \cdot \cos \angle(v, w)$ (projection property)
Properties

Geometric properties:

• \( \text{length}(\mathbf{v}) := \|\mathbf{v}\|_2 = \sqrt{\mathbf{v} \cdot \mathbf{v}} \) (Pythagoras)

• \( |\mathbf{v} \cdot \mathbf{w}| = \|\mathbf{v}\| \cdot \|\mathbf{w}\| \cdot \cos \angle(\mathbf{v}, \mathbf{w}) \) (projection property)

In particular:

• \( \mathbf{v} \) orthogonal to \( \mathbf{w} \) \( \iff \mathbf{v} \cdot \mathbf{w} = 0 \)
Properties

Gram-Schmidt Orthogonalization:

- Repeat for multiple vectors to create orthogonal set of vectors \( \{v'_1, ..., v'_n\} \) from set \( \{v_1, ..., v_n\} \)
Properties

Scalar product properties:

- Symmetric: \( \mathbf{v} \cdot \mathbf{w} = \mathbf{w} \cdot \mathbf{v} \)
- Bi-linear: \( \mathbf{u} \cdot (\lambda \mathbf{v} + \mathbf{w}) = \mathbf{u} \cdot \lambda \mathbf{v} + \mathbf{u} \cdot \mathbf{w} \)
- Positive: \( \mathbf{v} \cdot \mathbf{v} = 0 \iff \mathbf{v} = \mathbf{0} \)
Dot Product on Function Spaces

We need dot products on function spaces...

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

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

• It measures an abstract normal and angle between function (not in a geometric sense)

• **Orthogonal functions**: Don’t influence each other in linear combinations. Adding one to the other does not change the value in the other ones direction.
Linear Maps

Linear maps

- Linear maps and matrices
- Inverting and linear systems of equations
- Eigenvectors and eigenvalues
- Ill-posed problems
Linear Maps

A function $f: V \rightarrow W$ between vector spaces $V, W$ over a field $F$ is a *linear map*, if and only if:

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

**Theorem:**

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

Any linear map \( f \) between finite dimensional spaces can be represented as a matrix:

- We fix a basis (usually the standard basis)
- For each basis vector \( \mathbf{v}_i \) of \( V \), we specify the mapped vector \( \mathbf{w}_i \).
- Then, the map \( f \) is given by:

\[
f(\mathbf{v}) = f\left( \begin{pmatrix} \mathbf{v}_1 \\ \vdots \\ \mathbf{v}_n \end{pmatrix} \right) = \mathbf{v}_1 \mathbf{w}_1 + \ldots + \mathbf{v}_n \mathbf{w}_n
\]
Matrix Representation

This can be written as matrix-vector product:

\[
f(v) = \begin{bmatrix} | & | & | \\
  w_1 & \cdots & w_n \\
  | & | & |
\end{bmatrix} \cdot \begin{pmatrix} v_1 \\
  \vdots \\
  v_n \end{pmatrix}
\]

The columns are the images of the basis vectors (for which the coordinates of \( v \) are given)
Matrix Multiplication

Composition of linear maps corresponds to matrix products:

- \( f(g) = f \circ g = M_f \cdot M_g \)
- Matrix product calculation:

The \((x,y)\)-th entry is the dot product of row \(x\) of \(M_f\) and column \(y\) of \(M_g\)
Example

Example: rotation matrix

\[
\begin{pmatrix}
0 \\
1 \\
\end{pmatrix}
\]

\[
\begin{pmatrix}
1 \\
0 \\
\end{pmatrix}
\]

\[
\begin{pmatrix}
0 \\
1 \\
\end{pmatrix}
\]

\[
\begin{pmatrix}
\cos \alpha & -\sin \alpha \\
\sin \alpha & \cos \alpha \\
\end{pmatrix}
\]

Example: identity matrix

\[
I :=
\begin{pmatrix}
1 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 1 \\
\end{pmatrix}
\]
Orthogonal Matrices

Orthogonal Matrix

• A matrix is called *orthogonal* if all of its columns (rows) are *orthonormal*, i.e. \( c_i \cdot c_i = 1, \ c_i \cdot c_j = 0 \) for \( i \neq j \)

• The inverse of an orthogonal matrix is its transpose:

\[
\mathbf{MM}^{-1} = \mathbf{MM}^T = \mathbf{I}
\]
Affine Maps

Affine Maps

- Translations are not linear (except for zero translation)
- A combination of a linear map and a translation can be described by:
  \[ f(\mathbf{x}) = \mathbf{Mx} + \mathbf{t} \]
- This is called an affine map
- Composition of affine maps are affine:
  \[
  f(g(x)) = \mathbf{M}_f \left( \mathbf{M}_g \mathbf{x} + \mathbf{t}_g \right) \mathbf{x} + \mathbf{t}_f \\
  = \left( \mathbf{M}_f \mathbf{M}_g \right) \mathbf{x} + \left( \mathbf{M}_f \mathbf{t}_g + \mathbf{t}_f \right)
  \]
- For a vector space \( V \), a subspace \( S \subseteq V \) and a point \( \mathbf{p} \in V \), the set \( \{ \mathbf{x} \mid \mathbf{x} = \mathbf{p} + \mathbf{v}, \mathbf{v} \in V \} \) is called an affine subspace of \( V \). If \( \mathbf{p} \neq \mathbf{0} \), this is not a vector space.
Problem: Invert an affine map

• Given: $Mx = b$
• We know $M, b$
• Looking for $x$

Solution

• The set of solution is always an affine subspace of $\mathbb{R}^n$ (i.e., a point, a line, a plane...), or the empty set.
• There are innumerous algorithms for solving linear systems, here is a brief summary...
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 equations:

• Band matrices, constant bandwidth: modified elimination algorithm with $O(n)$ operations.

• Iterative Gauss-Seidel solver: converges for diagonally dominant matrices. Typically: $O(n)$ iterations, each costs $O(n)$ for a sparse matrix.

• Conjugate Gradient solver: works for symmetric, positive definite matrices in $O(n)$ iterations, but typically we get a good solution already after $O(\sqrt{n})$ iterations.

Determinants

- Assign a scalar $\det(M)$ to square matrices $M$
- The scalar measures the volume of the *parallelepiped* formed by the column vectors:

$$M = \begin{vmatrix} v_1 & v_2 & v_3 \end{vmatrix}$$
Properties

A few properties:

- $\det(A) \det(B) = \det(AB)$
- $\det(\lambda A) = \lambda^n \det(A)$ (\(n \times n\) matrix $A$)
- $\det(A^{-1}) = \det(A)^{-1}$
- $\det(A^T) = \det(A)$

- Can be computed efficiently using Gaussian elimination
Eigenvectors & Eigenvalues

Definition:

If for a linear map $M$ and a non-zero vector $x$ we have

$$Mx = \lambda x$$

we call $\lambda$ an *eigenvalue* of $M$ and $x$ 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: $M = T D T^{-1}$. 
Spectral Theorem:

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

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

$$\mathbf{M} = \mathbf{TDT}^T$$

with an orthogonal matrix $\mathbf{T}$. 
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 ratio of before / after normalization converges (this is the eigenvalue)
- Important intuition: Largest eigenvalue is the “dominant” component of the linear map.
Powers of Matrices

What happens:

• A symmetric matrix can be written as:

\[ M = T D T^T = T \begin{pmatrix} \lambda_1 & \cdots & \lambda_n \end{pmatrix} T^T \]

• Taking it to the $k$-th power yields:

\[ M^k = T D T^T T D T^T \cdots T D T^T = T D^k T^T = T \begin{pmatrix} \lambda_1^k & \cdots & \lambda_n^k \end{pmatrix} T^T \]

• Bottom line: Eigenvalue analysis is the key to understanding powers of matrices.
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:
  - $M = U \, D \, V^T$
  - 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**

\[
M = U D V^T
\]

- **M**: Matrix being decomposed
- **U**: Orthogonal matrix
- **D**: Diagonal matrix with singular values
- **V^T**: Transpose of orthogonal matrix

The singular values are denoted as \(\sigma_1, \sigma_2, \sigma_3, \sigma_4\).
Singular Value Decomposition

Singular value decomposition

- Can be used to solve linear systems of equations
- For full rank, square $M$:
  \[ M = U D V^T \]
  \[ \Rightarrow M^{-1} = (U D V^T)^{-1} = (V^T)^{-1} D^{-1} (U^{-1}) = V D^{-1} U^T \]
- Good numerical properties (numerically stable), but expensive
- The OpenCV library provides a very good implementation of the SVD
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

\[
f(x) = b \implies M_f x = b
\]

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 our input is inexact (e.g. measurement noise)
- Reminder: $\mathbf{M}^{-1} = \mathbf{V} \mathbf{D}^{-1} \mathbf{U}^T$
  
  - does not hurt
  - (orthogonal)
  
  - does not hurt
  - (orthogonal)

- Orthogonal transforms preserve the norm of $\mathbf{x}$, so $\mathbf{V}$ and $\mathbf{U}$ do not cause problems
Ill posed Problems

**Ratio:** Small singular values amplify errors

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

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

- 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: $\sigma_{\text{max}} / \sigma_{\text{min}}$
Regularization

Regularization

- Aims at avoiding the inversion problems
- Various techniques; in general the goal is to ignore the misleading information
  - Subspace inversion: do not use directions with small singular values (needs an SVD)
  - Additional assumptions: Assume smoothness (or something similar) in case of unclear or missing information so that compound problem \((f + \text{assumptions})\) is well posed
Quadrics

Quadrics

- Multivariate polynomials
- Quadratic optimization
- Quadrics & eigenvalue problems
Multivariate Polynomials

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

- A function $f: \mathbb{R}^n \rightarrow \mathbb{R}$, $x \rightarrow f(x)$
- $f$ is a polynomial in the components of $x$
- In any direction $f(s+tr)$, we obtain a one-dimensional polynomial of maximum degree $d$ in $t$.

Examples:

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

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

- $x^T A x + b^T x + c$

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

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

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

\[
= x1x + x2y + y3x + y4y
\]

\[
= 1x^2 + (2 + 3)xy + 4y^2
\]

\[
= 1x^2 + (2.5 + 2.5)xy + 4y^2
\]

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

Specifying quadratic polynomials:

- \( x^\top Ax + b^\top x + c \)
- \( b \) shifts the function in space (if \( A \) has full rank):
  \[
  (x - \mu)^\top A(x - \mu) + c
  \]
  \[
  = x^\top Ax - \mu^\top Ax - x^\top A\mu + \mu \cdot \mu + c
  \]
  (Asym.)
  \[
  = x^\top Ax - (2A\mu)x + \mu \cdot \mu + c
  \]
  \[
  = b
  \]
- \( c \) is an additive constant
Some Properties

Important properties

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

\[ 2x^2 + 3y^2 + 4xy + 2x + 2y + 4 \]
\[ + 3x^2 + 2y^2 + 1xy + 5x + 5y + 5 \]
\[ = 5x^2 + 5y^2 + 5xy + 7x + 7y + 9 \]

• In vector notation:

\[ \mathbf{x}^T A_1 \mathbf{x} + b_1^T \mathbf{x} + c_1 \]
\[ + \lambda (\mathbf{x}^T A_2 \mathbf{x} + b_2^T \mathbf{x} + c_2) \]
\[ = \mathbf{x}^T (A_1 + \lambda A_2) \mathbf{x} + (b_1 + \lambda b_2)^T \mathbf{x} + (c_1 + \lambda c_2) \]
Quadrics

- The zero level set of such a quadratic polynomial is called a “quadric”
- Shape depends on eigenvalues of $A$
- $b$ shifts the object in space
- $c$ sets the level
Shapes of Quadrics

Shape analysis:

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

\[
x^T A x = x^T \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{bmatrix} Q x
\]

\[
= (Qx)^T \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{bmatrix} (Qx)
\]

- $Q$ 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

- Assume we want to minimize a quadratic objective function $x^T A x + b^T x + c$
- $A$ has only positive eigenvalues.
- Means: It’s a paraboloid with a unique minimum
- The vertex (critical point) of the paraboloid can be determined by simply solving a linear system
- More on this later (need some more analysis first)
Rayleigh Quotient

Relation to eigenvalues:

- The minimum and maximum eigenvalues of a symmetric matrix $A$ can be expressed as constraint quadratic optimization problem:

$$
\lambda_{\text{min}} = \min \frac{x^T Ax}{x^T x} = \min \left( x^T Ax \right) \quad \lambda_{\text{max}} = \max \frac{x^T Ax}{x^T x} = \max \left( x^T Ax \right)
$$

- 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:

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

$$\sqrt{D} = \begin{pmatrix} \sqrt{\lambda_1} & \cdots & \sqrt{\lambda_n} \end{pmatrix}$$
**SPD Quadrics**

\[ x^T x \]

Identity \( I \)

\[ \begin{align*}
    M &= T D T^T = \left( T \sqrt{D} \right)^2 \\
    \end{align*} \]

**Interpretation:**

- Start with a unit positive quadric \( x^T x \).
- 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")
Software
**GeoX** comes with several linear algebra libraries:

- 2D, 3D, 4D vectors and matrices: `LinearAlgebra.h`
- Large (dense) vectors and matrices: `DynamicLinearAlgebra.h`
- Gaussian elimination: `invertMatrix()`
- Sparse matrices: `SparseLinearAlgebra.h`
- Iterative solvers (Gauss-Seidel, conjugate gradients, power iteration): `IterativeSolvers.h`