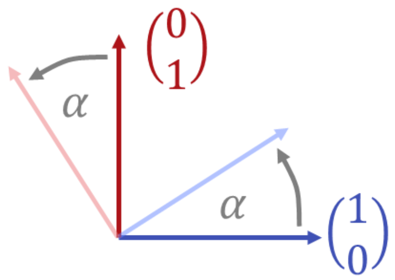
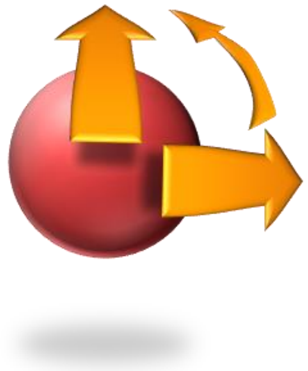
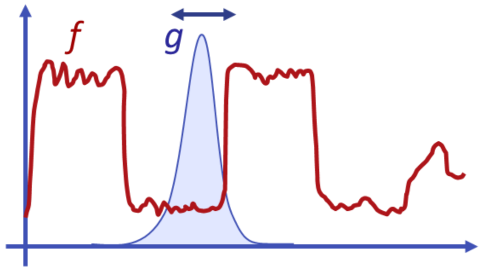
# Modelling 1

SUMMER TERM 2020







## LECTURE 8

# (Linear) Information Loss

Michael Wand · Institut für Informatik · Michael.Wand@uni-mainz.de

## Information Loss in Linear Mappings

## Linear Maps

**A function**

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

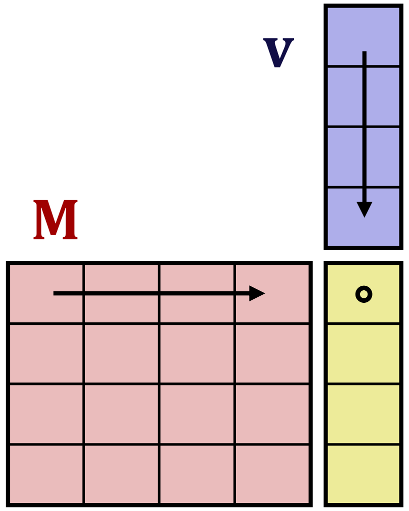
**is linear if and only if:**

- $\forall \mathbf{v}_1, \mathbf{v}_2 \in V$: $f(\mathbf{v}_1 + \mathbf{v}_2) = f(\mathbf{v}_1) + f(\mathbf{v}_2)$
- $\forall \mathbf{v} \in V, \lambda \in \mathbb{R}$: $f(\lambda \mathbf{v}) = \lambda f(\mathbf{v})$

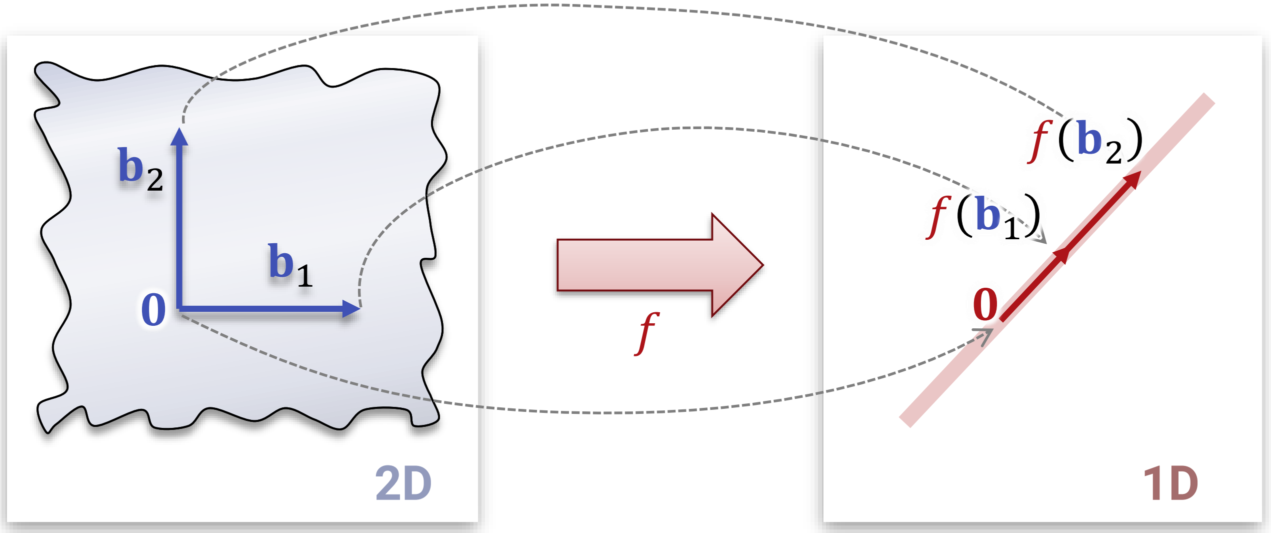
# Matrix Product

**All operations are matrix-matrix products:**

- Matrix-Vector product:
- $f(\mathbf{x}) = \mathbf{M}_f \cdot \mathbf{x}$



# Not invertible

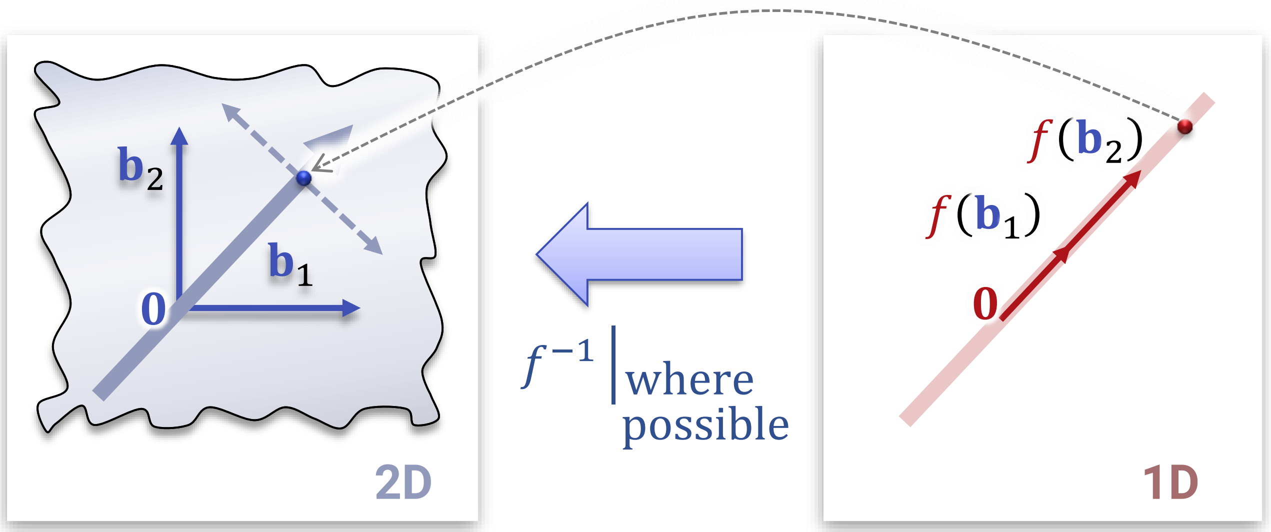


$$f(\mathbf{x}) = \begin{pmatrix} x_1 + x_2 \\ 2x_1 + 2x_2 \end{pmatrix}$$
$$= \begin{pmatrix} 1 & 1 \\ 2 & 2 \end{pmatrix} \cdot \mathbf{x}$$

**Information flow:**

- After $f$, we can recover $b_1 + b_2$
  - Sum of inputs
- We do not know $b_1 - b_2$ anymore
  - Difference of inputs

# Not invertible

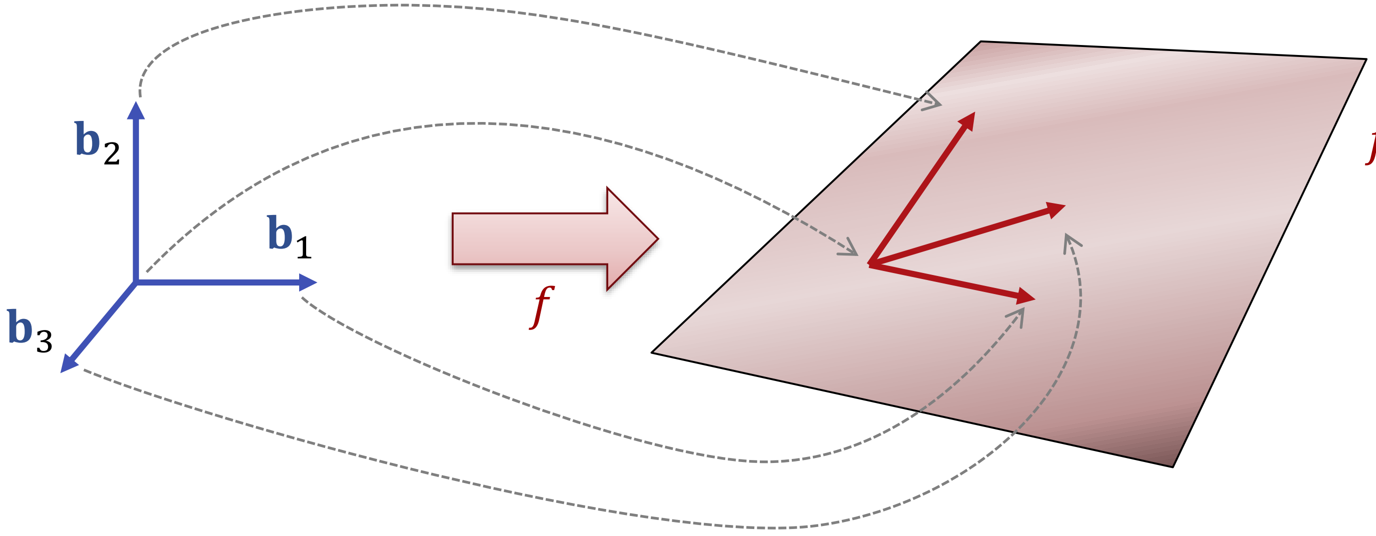


$$f(\mathbf{x}) = \begin{pmatrix} x_1 + x_2 \\ 2x_1 + 2x_2 \end{pmatrix}$$
$$= \begin{pmatrix} 1 & 1 \\ 2 & 2 \end{pmatrix} \cdot \mathbf{x}$$

## Information flow:

- After $f$, we can recover $b_1 + b_2$
  - Sum of inputs
- We do not know $b_1 - b_2$ anymore
  - Difference of inputs

# Not invertible



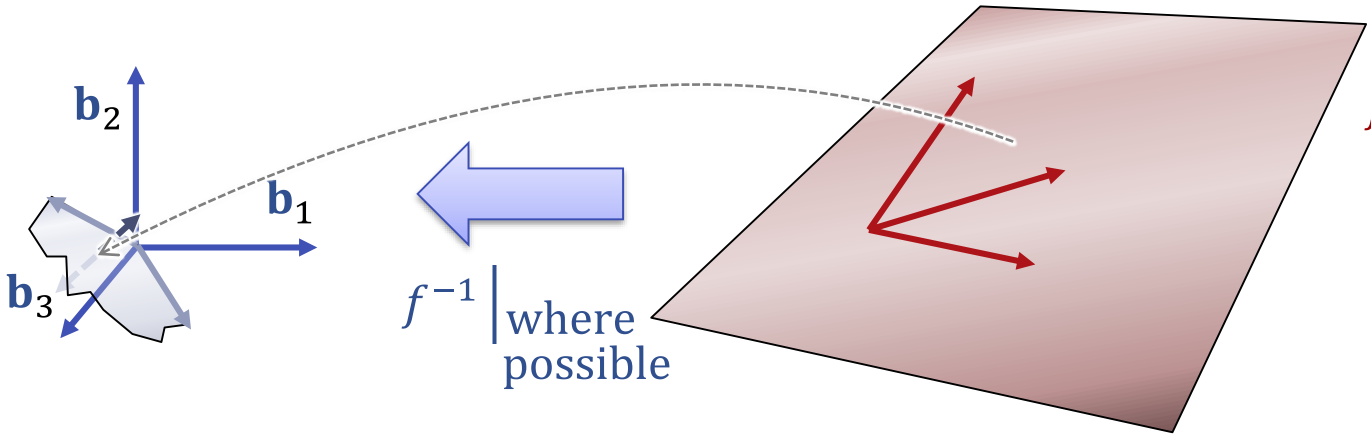
$$f(\mathbf{x}) = \begin{pmatrix} x_1 + x_3 \\ x_2 + x_3 \\ 0 \end{pmatrix}$$

$$= \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 0 \end{pmatrix} \cdot \mathbf{x}$$

## Information flow:

- After $f$, we can recover $b_1 + b_3$ and $b_2 + b_3$
- We do not know $b_2 - b_3$ anymore

# Not invertible



$$f(\mathbf{x}) = \begin{pmatrix} x_1 + x_3 \\ x_2 + x_3 \\ 0 \end{pmatrix}$$

$$= \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 0 \end{pmatrix} \cdot \mathbf{x}$$

## Information flow:

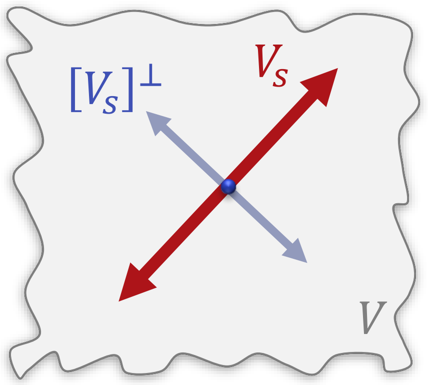
- After $f$, we can recover $b_1 + b_3$ and $b_2 + b_3$
- We do not know $b_2 - b_3$ anymore

## Orthogonal Comlement

### Definition

- **Given:** Subspace $V_s \subseteq V$
- Orthogonal complement

$$V_s^{\perp} := \{\mathbf{v} \in V \,|\, \forall \mathbf{w} \in V_s : \langle \mathbf{v}, \mathbf{w} \rangle = 0\}$$



### Intuition

- Set of all vectors orthogonal to $V_s$
- Zero projection onto any $\mathbf{w} \in V_s$

### Theorem

$$V_s \subset V \Rightarrow V = \mathrm{span}\{V_s, V_s^{\perp}\} \quad [:= V_s \oplus V_s^{\perp}]$$

## In general

**Consider mapping**

$$f: V_1 \to V_2$$

**Subspaces of $V_1$**

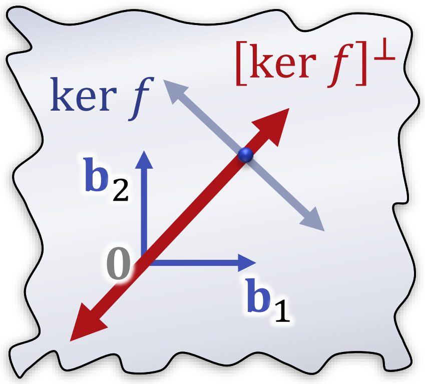
- **Kernel:** Subspace that is lost

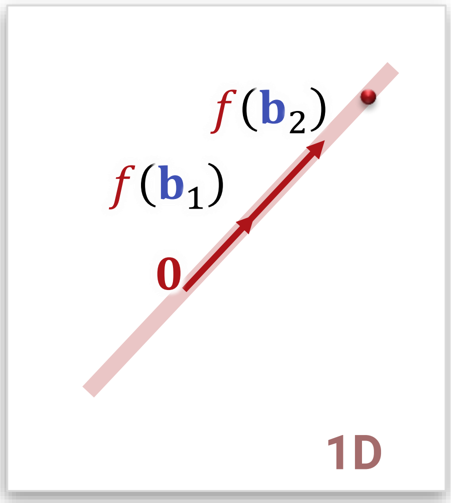
$$\ker f := \{\mathbf{x} \in V_1 | f(\mathbf{x}) = 0\}$$

- **Orthogonal complement of kernel**

$$[\ker f]^{\perp} = \{\mathbf{v} \in V_1 | \forall \mathbf{w} \in \ker f : \langle \mathbf{v}, \mathbf{w} \rangle = 0\}$$

  - In this space, $f$ is invertible





## In general

**Consider mapping**

$$f : V_1 \to V_2$$

**In the target domain**

$$\operatorname{im} f := \{\mathbf{y} \in V_2 | \exists \mathbf{x} \in V_1 : f(\mathbf{x}) = \mathbf{y}\}$$

- Subspace of $V_2$
- Same dimension as kernel complement

$$\dim([\ker f]^\perp) = \dim(\operatorname{im} f)$$

## In general

**Consider mapping**

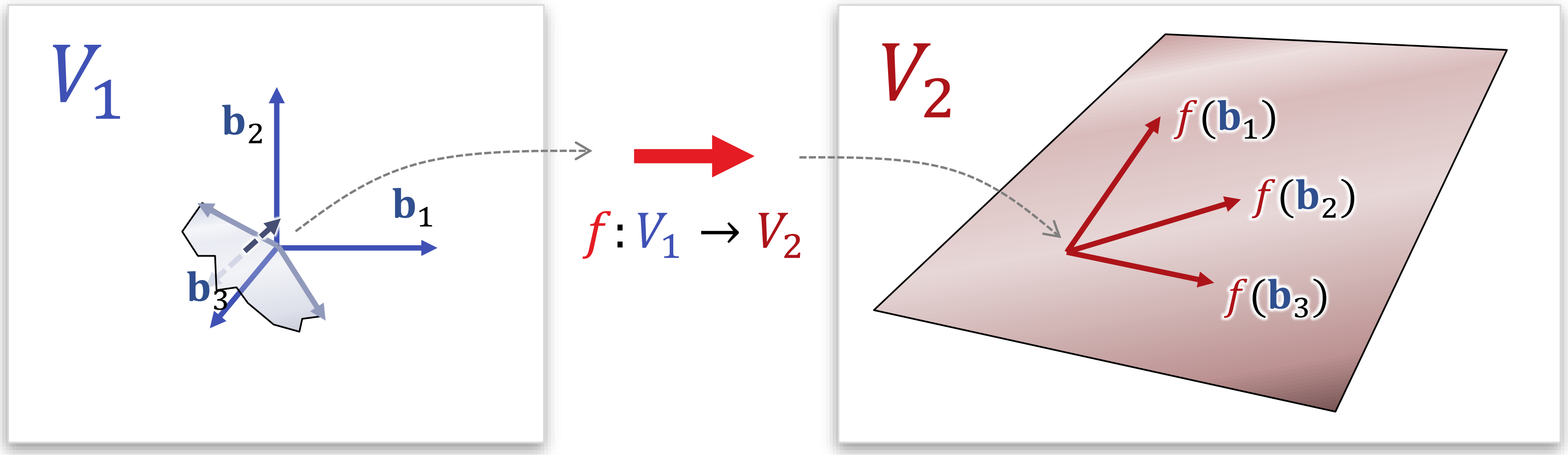
- Rank is the dimension of the mapped space

$$\begin{aligned}\operatorname{rank}(f) &:= \dim(\operatorname{im} f)\\ &= \dim\left(\operatorname{span}(V_1 \backslash \ker f)\right)\end{aligned}$$

- Source space $V_1$ is split:
  - $\dim \operatorname{im}(f)$ = dimensions “preserved” by $f$
  - $\dim \ker(f)$ = dimensions “removed” by $f$
- Sums up:

$$\dim(V_1) = \dim(\operatorname{im} f) + \dim(\ker f)$$

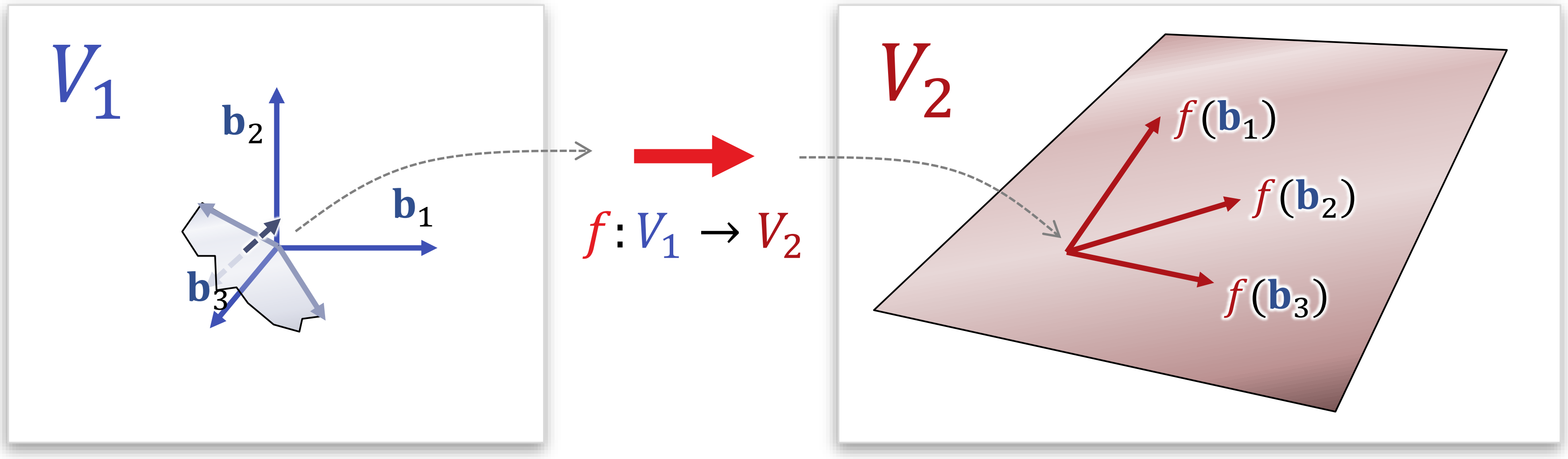
# Structural Insight



## Mapping Subspaces to Subspaces

- Invertible map from $[\ker f]^{\perp} \to \operatorname{im} f$
- Not covered
  - "Source" information lost: coordinates within $\ker f$
  - Unreachable "targets": vectors within $[\operatorname{im} f]^{\perp}$

# Structural Insight



## Dimensions add up

- $\dim[\ker f]^\perp = \dim \operatorname{im} f$
- $\dim V_1 = \dim \ker f + \dim[\ker f]^\perp$
- $\dim V_2 = \dim \operatorname{im} f + \dim[\operatorname{im} f]^\perp$

## In practice?

### In practice

- It always never works:
  - Most matrices have noise (measurement, numerics)
    - Any practical mapping has “full rank”
  - Inverting matrices is not always stable
    - Even full-rank matrices might delete information
  - Need to understand this better!

### We will discuss this soon

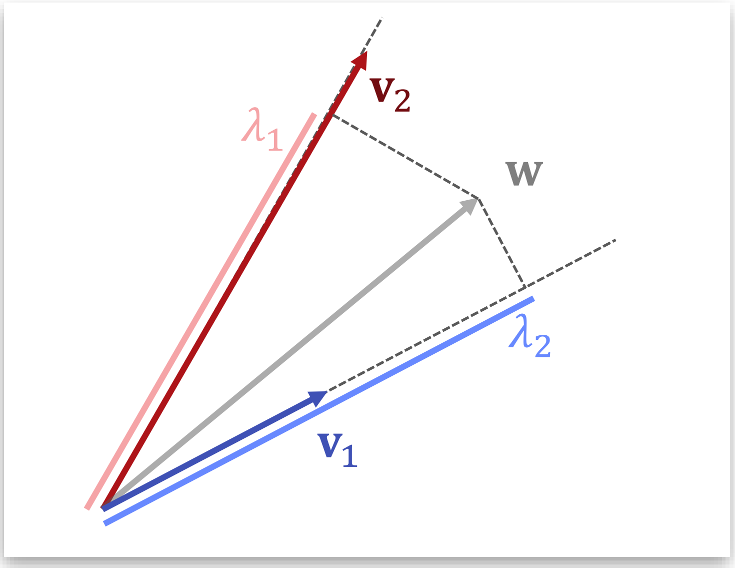
- Tools:
  - Eigenvalues
  - Singular value decomposition (SVD)

Linear Systems of Equations

## Inverting Linear Maps

# Situation

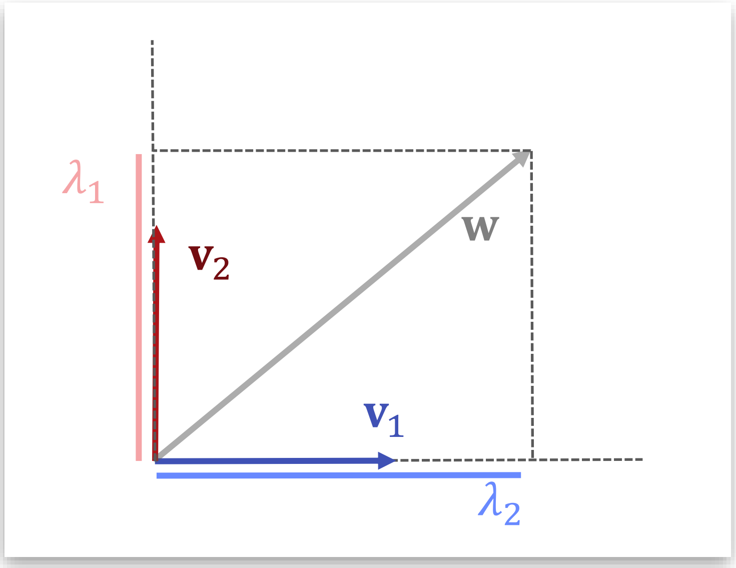
## General Case



Linear System

$$\lambda_1 \cdot \mathbf{v}_1 + \cdots + \lambda_n \cdot \mathbf{v}_n = \mathbf{w}$$

## Orthogonal



Direct Computation

$$\lambda_1 = \mathbf{v}_1 \cdot \mathbf{w}$$
$$\vdots$$
$$\lambda_n = \mathbf{v}_n \cdot \mathbf{w}$$

## Linear Systems of Equations

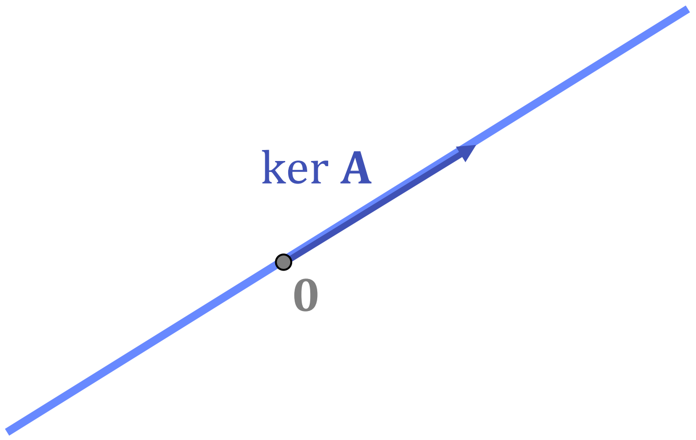
### Problem: Invert an affine map

- Given: $\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$, i.e, $\mathbf{A} \cdot \mathbf{x} - \mathbf{b} = \mathbf{0}$
  - We know $\mathbf{A}$, $\mathbf{b}$
  - Looking for $\mathbf{x}$
- Compute $\mathbf{x} = \mathbf{A}^{-1} \cdot \mathbf{b}$

### Solution

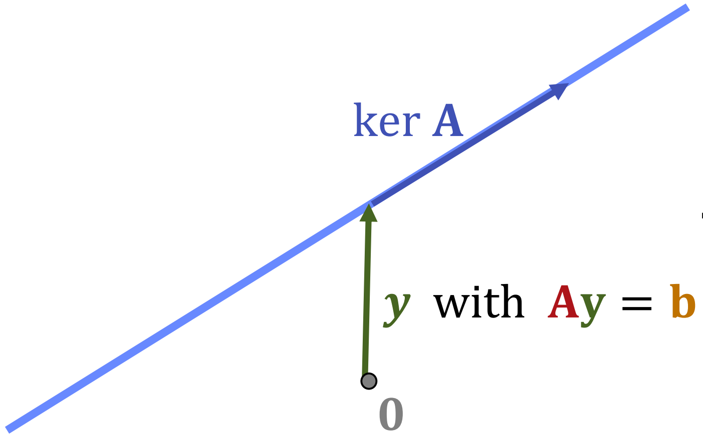
- Set of solutions: *affine subspace* of $\mathbb{R}^n$ (or $\emptyset$)
  - Point, line, plane, hyperplane...
- Innumerous algorithms

# Linear Systems of Equations



$\{\mathbf{x} \mid \mathbf{A}\mathbf{x} = \mathbf{0}\}$ – hyperplane through the origin

“Homogeneous” system



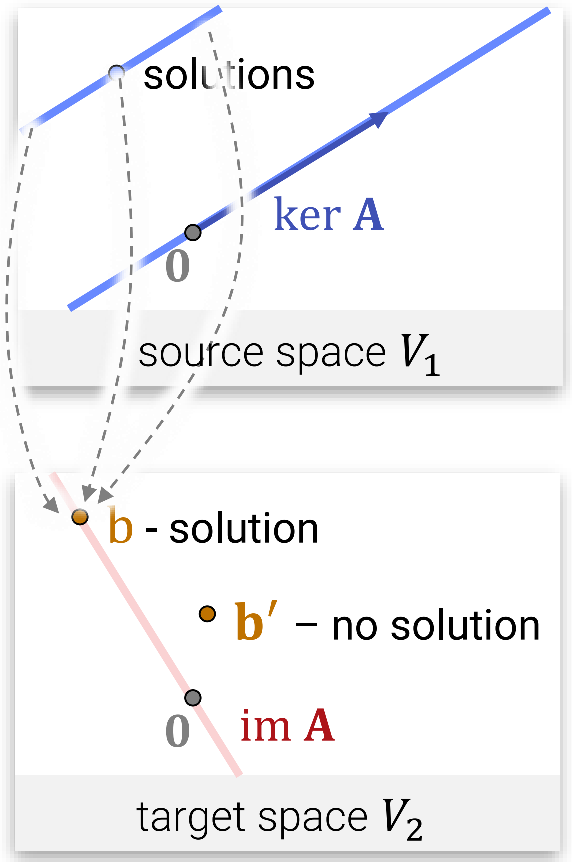
$\{\mathbf{x} \mid \mathbf{A}\mathbf{x} = \mathbf{b}\}$ – hyperplane through any point

“Inhomogeneous” system

## Structure

**Linear System ($\mathbf{A}: V_1 \rightarrow V_2$):**

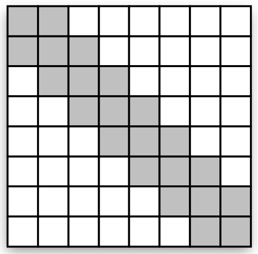
- $\mathbf{Ax} = \mathbf{0}$
  - Solution space = $\ker \mathbf{A}$
- $\mathbf{Ax} = \mathbf{b}$
  - Might or might not have a solution
  - Solution if and only if $\mathbf{b} \in \operatorname{im} \mathbf{A}$
- Set of all solutions:
  - One $\mathbf{y}$ with $\mathbf{Ay} = \mathbf{b}$
  - Add any solution of $\mathbf{Ax} = \mathbf{0}$
  - Solution set: $\mathbf{y} + \ker \mathbf{A}$



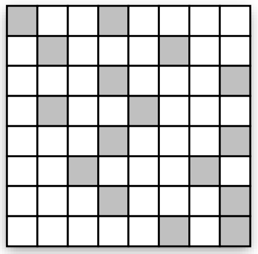
## Solvers for Linear Systems

### Solving linear systems of equations

- **Baseline:** 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



# Solvers for Linear Systems

## Algorithms: 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.

See: J. R. Shewchuk, An Introduction to the Conjugate Gradient Method Without the Agonizing Pain, 1994.