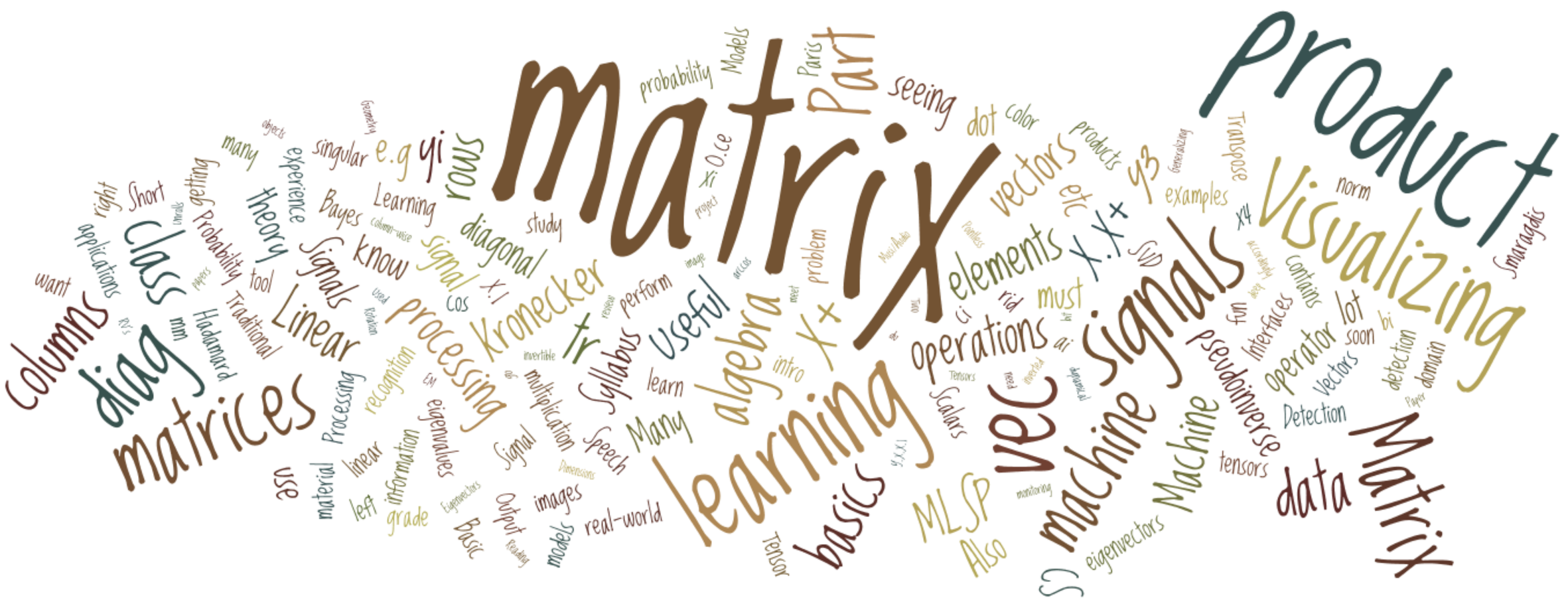


CS545 Machine Learning for Signal Processing

Introduction & Linear Algebra Basics

21 August 2023

What do I need to know?

- I won't assume you know machine learning or signal processing
  - You are here to learn, not to know!

- Be comfortable with the basics
  - Some linear algebra, some probability
    - If you are rusty that's ok, you are here to learn

What is this class?

- Is it a signal processing class?
  - What is signal processing?
- Is it a machine learning class?
  - What is machine learning?

Signal Processing

- The study of capturing, processing and manipulating *signals*

- What is a signal?

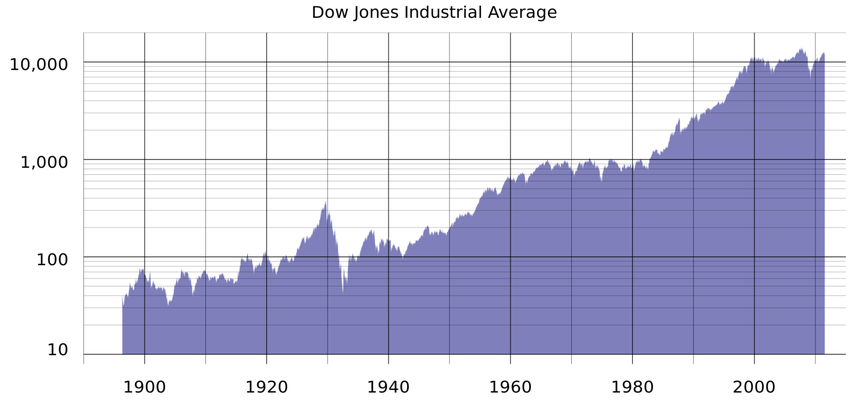
Signals (as far as this class goes)

- *Structured* collections of measurements that convey information

Images

Brains!!

Stocks



Machine Learning

- The study of discovering, extracting, and using information from *data*

- For data we will use signals

Why bother?

- Traditional signal processing doesn't really care (much) about its input's content
  - Most algorithms are "optimal" for all cases

- Traditional machine learning is not very signals-friendly
  - Not obvious how to perform, e.g. time-series clustering

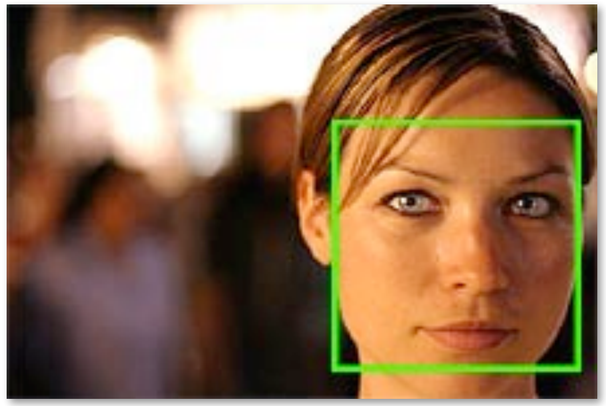
Machine Learning for Signal Processing

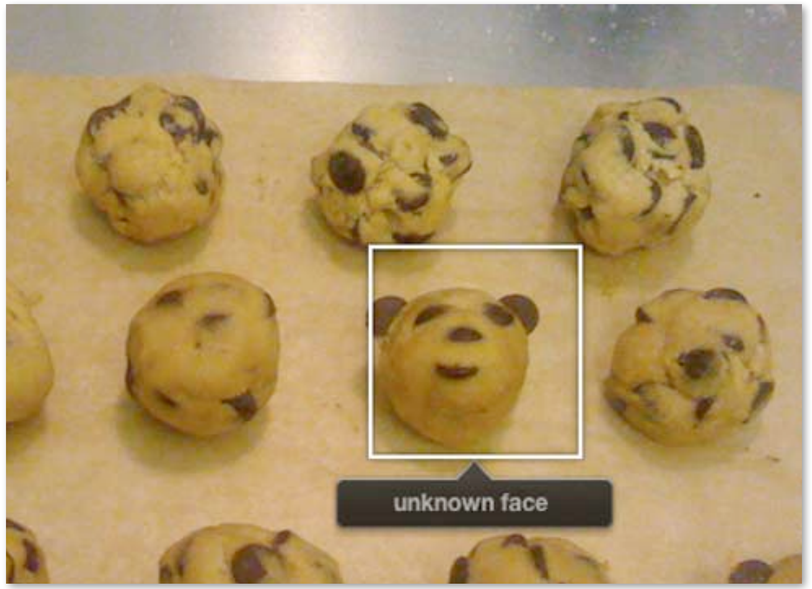
- MLSP combines both disciplines to perform learning on signals data

- Many examples of MLSP in the real-world

Face recognition

- Found in cameras and photo software





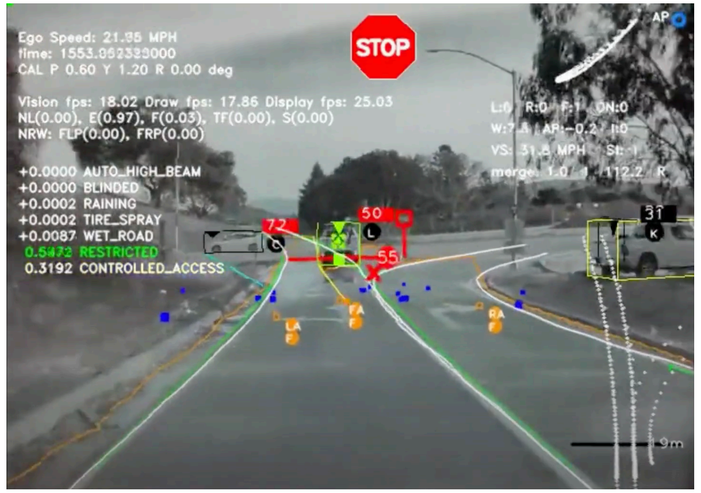
Speech recognition

Surveillance/monitoring

- Detection of specified objects / activity

Gunshot detection

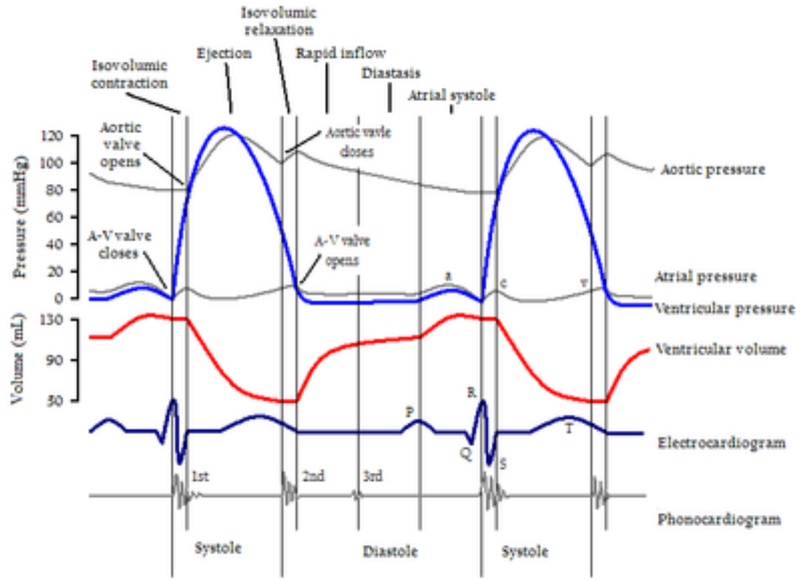
Pedestrian detection



Self-driving cars

Bio-signals

- Interpreting our body's data



Heartbeat processing

Brain Machine Interfaces

Many other applications

- Biometrics
- Music Information Retrieval
- Robotics
- Gesture-based UIs
- Condition monitoring (power grids, structures, automotive, …)
- Financial data mining
- Many more …

All of these have been class projects in this class!

About this course

- Heavy on practical applications
  - Please bring your own domain problem in class!
- Won’t go excruciatingly deep on theory
  - We’ll skip convergence proofs, etc.
    - Many more courses here that cover all that
    - But we will go through the necessary math
  - We’ll go fast. Each lecture could be an entire topic
- Our goal is getting intuition and real-world experience

Syllabus: the basics

- Covering the basics:
  - Part 1: Linear Algebra and Probability
  - Part 2: Signals Theory
  - Part 3: Representations and Features

Syllabus: machine learning review

- Elements of machine learning
  - Part 4: Unsupervised learning basics
  - Part 5: Detection and classification
  - Part 6: Time-series and dynamical models

Syllabus: The fun stuff

- Applications and theory:
  - Matrix/Tensor Factorizations, Bag Models
  - Manifolds and Embedding
  - Deep Learning
  - Graph Signal Processing and Machine Learning
  - ICA, MIMO models, Arrays, Sensor Fusion
  - Compressive Sensing and Sparsity
  - Privacy-aware ML
  - Application areas:
    - Computer Vision, Speech Recognition, Music/Audio, Bio/Brain-signals
    - …

Your part

- Problem sets: 40% of the grade
  - 3 to 5 throughout class, will be mostly coding problems
- Final project: 50% of the grade
  - Mid-semester: Proposal due
  - Last 1-2 weeks: Presentations and/or posters
- Remaining 10% of grade
  - Be active in class, ask questions, make sure I know who you are!
  - Remember, this is an in-person class

The final project is “conference style”

- Teams of 2-3 students (no more than 3, no less than 2)
  - Start making friends now!
- Mid-term: Abstract submission
  - Short abstract describing the problem you want to solve and how you plan to
- Week ~13: Paper Submission
  - 4-6 page paper
  - Peer reviewed by all of you
  - All papers will be accepted and presented

Class platform

- We will use a Microsoft Teams team (sorry)
  - Sign up at: https://go.cs.illinois.edu/CS545
    - Use code: `rdhh903`
  - Will upload in it lectures, problem sets, data, links, etc.
  - Use it for questions, discussions, finding project-mates, etc.
  - Will be used for all class announcements
    - (so sign up if you aren't in it)

Course staff: Who am I?

- Instructor: Paris Smaragdis (CS & ECE)
  - paris@illinois.edu
    - DM me on MS Teams if you have questions or want to meet
- Interested in machine perception, computational audition
- Previously chaired the IEEE MLSP Technical Committee, the IEEE SPS Data Science Initiative, and the ICA/LVA community
- I've written many MLSP papers
- And productized many MLSP products

Who is the TA?

- Krishna Subramani
  - ECE PhD student (living in CS)
  - ks51@illinois.edu

This is Krishna ⟶

- No fixed physical office hours
  - Please ask us questions through MS Teams

- Some advice
  - Seek help early, avoid mad rush before deadlines
  - Use MS Teams first, you'll get answers faster

Who are you?

- Name, department, grad/undergrad?
  - What are your interests in this area?
  - Hint: you will need a project-mate (take notes!)

Final administrative notes

- Enrollment constraints
  - If you need a waiver let me know, I do not foresee problems with class capacity (there was a mixup, we should have lots more slots)
- If you are not formally registered yet:
  - Please add yourself to MS Teams to be in the loop
  - Keep up with the assignments, we won't accept late ones in case you register later
- Questions?

Linear algebra refresher

- Linear algebra is the math tool du jour
  - Compact notation
  - Convenient set operations
  - Used in all modern texts
  - Interfaces well with MATLAB, numpy, R, PyTorch, etc.

- We will use a lot of it!!

Scalars, Vectors, Matrices, Tensors

0th order tensor: Scalar

$$x$$

1st order tensor: Vector

$$\mathbf{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_N \end{bmatrix}$$

2nd order tensor: Matrix

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}_1 \cdots \mathbf{x}_M \end{bmatrix} = \begin{bmatrix} x_{1,1} & \cdots & x_{1,M} \\ \vdots & & \vdots \\ x_{N,1} & \cdots & x_{N,M} \end{bmatrix}$$

1st dimension; 2nd dimension

3rd order tensor: Tensor

$$\mathbf{X} = \{\mathbf{X}_1, \ldots, \mathbf{X}_K\} = \begin{bmatrix} x_{1,1,1} & \cdots & x_{1,M,1} \\ \vdots & & \vdots \\ x_{N,1,1} & \cdots & x_{N,M,1} \end{bmatrix} \ddots \begin{bmatrix} x_{1,1,K} & \cdots & x_{1,M,K} \\ \vdots & & \vdots \\ x_{N,1,K} & \cdots & x_{N,M,K} \end{bmatrix}$$

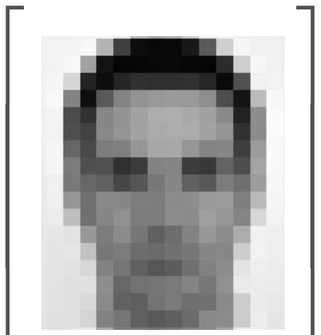
1st dimension; 2nd dimension; 3rd dimension

How will we see these?

- 1D signals (e.g. sounds) will be vectors

$$\mathbf{x}^\top = \begin{bmatrix} x(0) & \cdots & x(T) \end{bmatrix} =$$ 

- 2D signals (e.g. images) will be matrices

$$\mathbf{X} = \begin{bmatrix} x_{1,1} & \cdots & x_{1,M} \\ \vdots & & \vdots \\ x_{N,1} & \cdots & x_{N,M} \end{bmatrix} =$$ 

- 3D data will be videos, etc ...

Element-wise operations

- Addition/subtraction

$$\mathbf{a} \pm \mathbf{b} = \mathbf{c} \rightarrow a_i \pm b_i = c_i$$

- Multiplication (Hadamard product)

$$\mathbf{a} \odot \mathbf{b} = \mathbf{c} \rightarrow a_i \; b_i = c_i$$

  - Other times denoted as $\mathbf{a} \circ \mathbf{b}$

- No named operator for element-wise division
  - Just use Hadamard with reciprocal $\mathbf{b}$

Transpose

- Transpose
  - Change rows to columns (and vice versa)

$$\mathbf{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_N \end{bmatrix}, \quad \mathbf{x}^\top = \begin{bmatrix} x_1 & \cdots & x_N \end{bmatrix} \qquad \mathbf{x} = \begin{bmatrix} x_1 & x_3 \\ x_2 & x_4 \end{bmatrix}, \quad \mathbf{x}^\top = \begin{bmatrix} x_1 & x_2 \\ x_3 & x_4 \end{bmatrix}$$

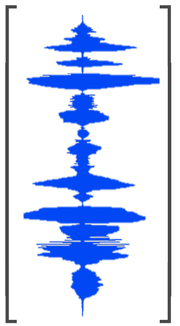
not T!

- Hermitian (conjugate transpose)
  - Notated as $\mathbf{X}^\mathsf{H}, \mathbf{X}^*, \mathbf{X}^\dagger$
  - In addition to transposing we also take the complex conjugate of each entry
    - If you have complex-valued data this makes a big difference!

Visualizing transposition

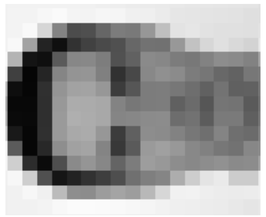
- Mostly pointless for 1D signals

$$\mathbf{x} = \left[\ \right], \quad \mathbf{x}^\top = \left[\ \right]$$



- Swap dimensions for 2D signals

$$\mathbf{x} = \left[\ \right], \quad \mathbf{x}^\top = \left[\ \right]$$



Reshaping operators

- The vec operator
  - Unrolls elements column-wise
    - Or in order of dimensions for tensors
  - Useful for getting rid of matrices/tensors

$$\text{vec}(\mathbf{x}) = \text{vec}\begin{bmatrix} x_1 & x_3 \\ x_2 & x_4 \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$$

- The vec-transpose
  - For (p) of an $m \times n$ matrix make each column a $p \times (m/p)$ matrix
  - Useful for inverting vec and getting rid of tensors $\mathbf{X} = \text{vec}(\mathbf{X})^{(M)}, \mathbf{X} \in \mathbb{R}^{M \times N}$

$$\mathbf{x}^{(2)} = \begin{bmatrix} x_{11} & x_{12} \\ x_{21} & x_{22} \\ x_{31} & x_{32} \\ x_{41} & x_{42} \\ x_{51} & x_{52} \\ x_{61} & x_{62} \end{bmatrix}^{(2)} = \begin{bmatrix} x_{11} & x_{31} & x_{51} \\ x_{21} & x_{41} & x_{61} \\ x_{12} & x_{32} & x_{52} \\ x_{22} & x_{42} & x_{62} \end{bmatrix}$$

Trace and diag

- Matrix trace
  - Sum of diagonal elements

$$\mathrm{tr}(\mathbf{X}) = \mathrm{tr}\begin{bmatrix} x_{11} & \cdots & x_{1N} \\ \vdots & \ddots & \vdots \\ x_{N1} & \cdots & x_{NN} \end{bmatrix} = \sum_i x_{ii}$$

- The diag operator

$$\mathrm{diag}(\mathbf{x}) = \mathrm{diag}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} x_1 & 0 \\ 0 & x_2 \end{bmatrix}$$

Make a diagonal matrix from a vector

$$\mathrm{diag}^{-1}\begin{bmatrix} x_1 & a \\ b & x_2 \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

Extract diagonal as a vector

Norms

- 2-norm:

$$||\mathbf{x}|| = \sqrt{\sum |x|^2}_i$$

← *Often lazily written as $|x|$ as well*

- p-norms:

$$||\mathbf{x}||_p = \left(\sum_i |x_i|^p \right)^{1/p}$$

- Frobenius norm:

$$||\mathbf{X}||_F = \sqrt{\sum \sum |x_{i,j}|^2} = \sqrt{\mathrm{tr}(\mathbf{X} \cdot \mathbf{X}^{\mathsf{H}})} = ||\,\mathrm{vec}(\mathbf{X})||$$

The dot product

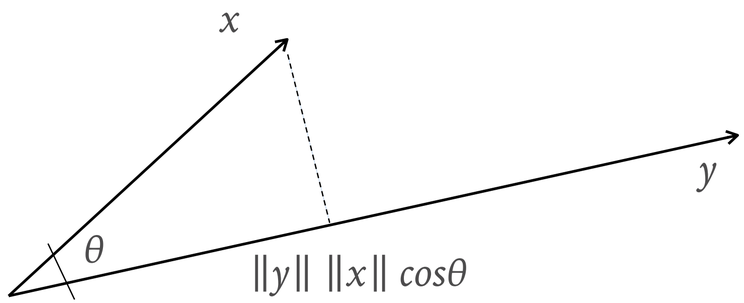
- Dot product
  - Shorthand for multiply and accumulate

$$\mathbf{x}^\top \cdot \mathbf{y} = \sum_i x_i \cdot y_i = \|\mathbf{x}\| \cdot \|\mathbf{y}\| \cos\theta$$

- Geometry
  - For unit vectors:

$$\theta = \arccos(\mathbf{x}^\top \cdot \mathbf{y})$$

  - Great tool for checking out similarity



Matrix-vector product

- Generalizing the dot product

$$\mathbf{X} \cdot \mathbf{y} = \begin{bmatrix} \mathbf{x}_1^\top \\ \mathbf{x}_2^\top \\ \mathbf{x}_3^\top \end{bmatrix} \cdot \mathbf{y} = \begin{bmatrix} \mathbf{x}_1^\top \cdot \mathbf{y} \\ \mathbf{x}_2^\top \cdot \mathbf{y} \\ \mathbf{x}_3^\top \cdot \mathbf{y} \end{bmatrix} = \begin{bmatrix} \sum x_{1,i} \cdot y_i \\ \sum x_{2,i} \cdot y_i \\ \sum x_{2,i} \cdot y_i \end{bmatrix}$$

  - $\mathbf{X}$ must have as many columns a $\mathbf{y}$ has elements
    - Does not commute!

- Useful for computing multiple dot products
  - Pack all vectors that you want to multiply in a matrix

Matrix-matrix product

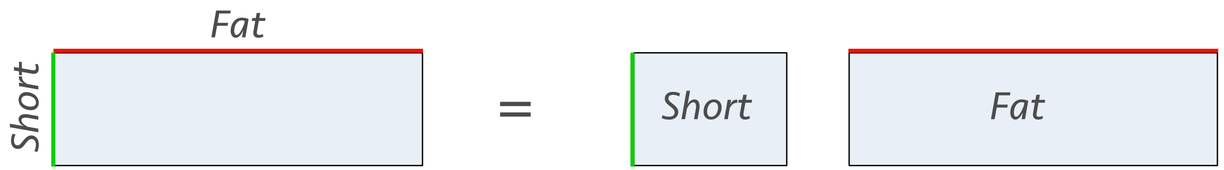
- Between two matrices:

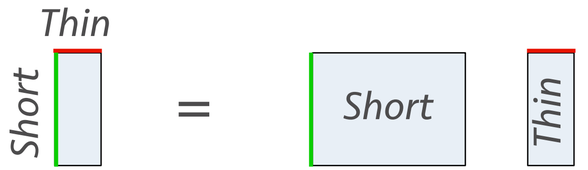
$$\mathbf{X}\cdot\mathbf{Y}=\begin{bmatrix}\mathbf{x}_1^\top\\ \mathbf{x}_2^\top\\ \mathbf{x}_3^\top\end{bmatrix}\cdot\begin{bmatrix}\mathbf{y}_1 & \mathbf{y}_2 & \mathbf{y}_3\end{bmatrix}=\begin{bmatrix}\mathbf{x}_1^\top\cdot\mathbf{y}_1 & \mathbf{x}_1^\top\cdot\mathbf{y}_2 & \mathbf{x}_1^\top\cdot\mathbf{y}_3\\ \mathbf{x}_2^\top\cdot\mathbf{y}_1 & \mathbf{x}_2^\top\cdot\mathbf{y}_2 & \mathbf{x}_2^\top\cdot\mathbf{y}_3\\ \mathbf{x}_3^\top\cdot\mathbf{y}_1 & \mathbf{x}_3^\top\cdot\mathbf{y}_2 & \mathbf{x}_3^\top\cdot\mathbf{y}_3\end{bmatrix}$$

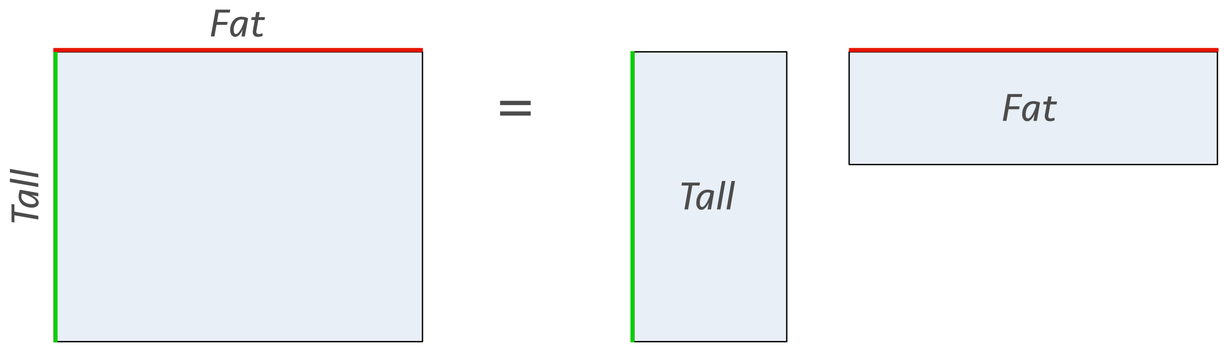
  - $\mathbf{X}$ must have as many columns as $\mathbf{Y}$ has rows
    - $(M\times N)=(M\times K)\cdot(K\times N)$
    - Remember this operation doesn't commute!
- All linear operations can be represented as a matrix product
  - We'll be seeing that a lot!

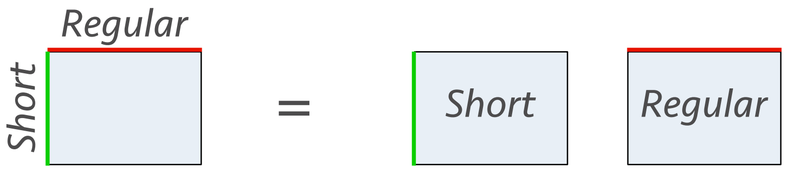
Matrix products

- # of output rows == left matrix # of rows
- # of output columns == right matrix # of columns

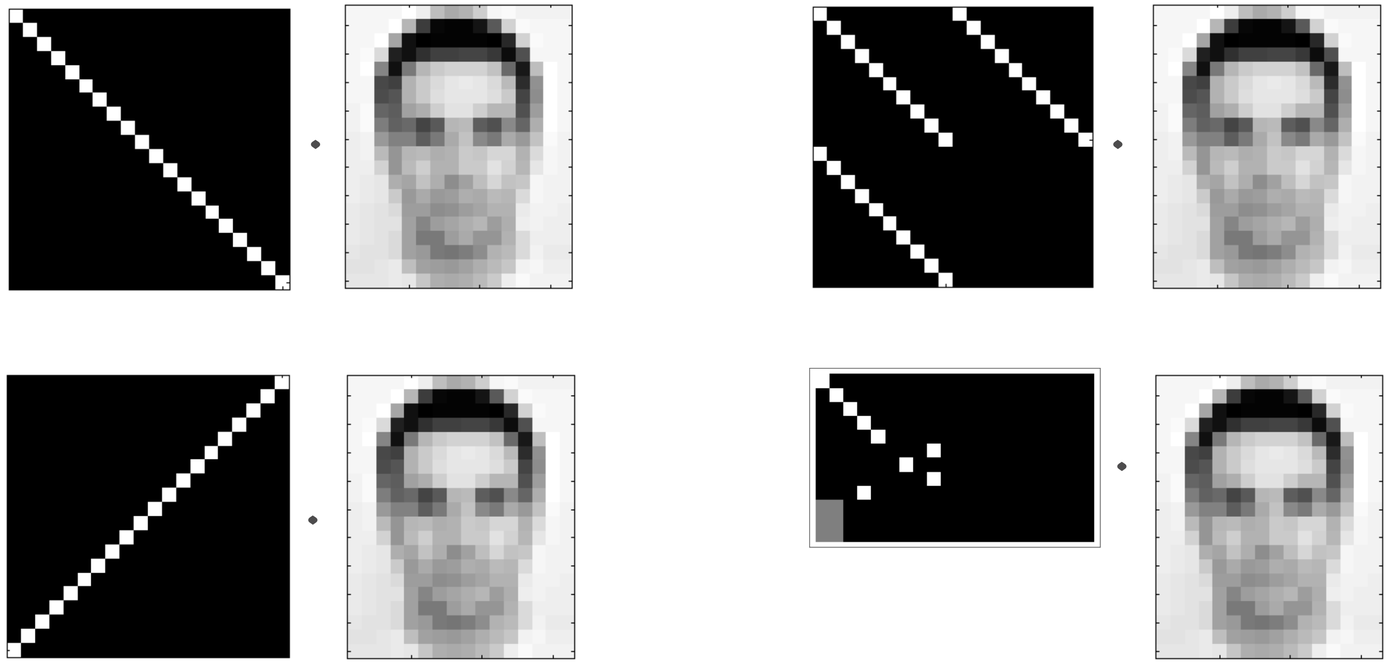




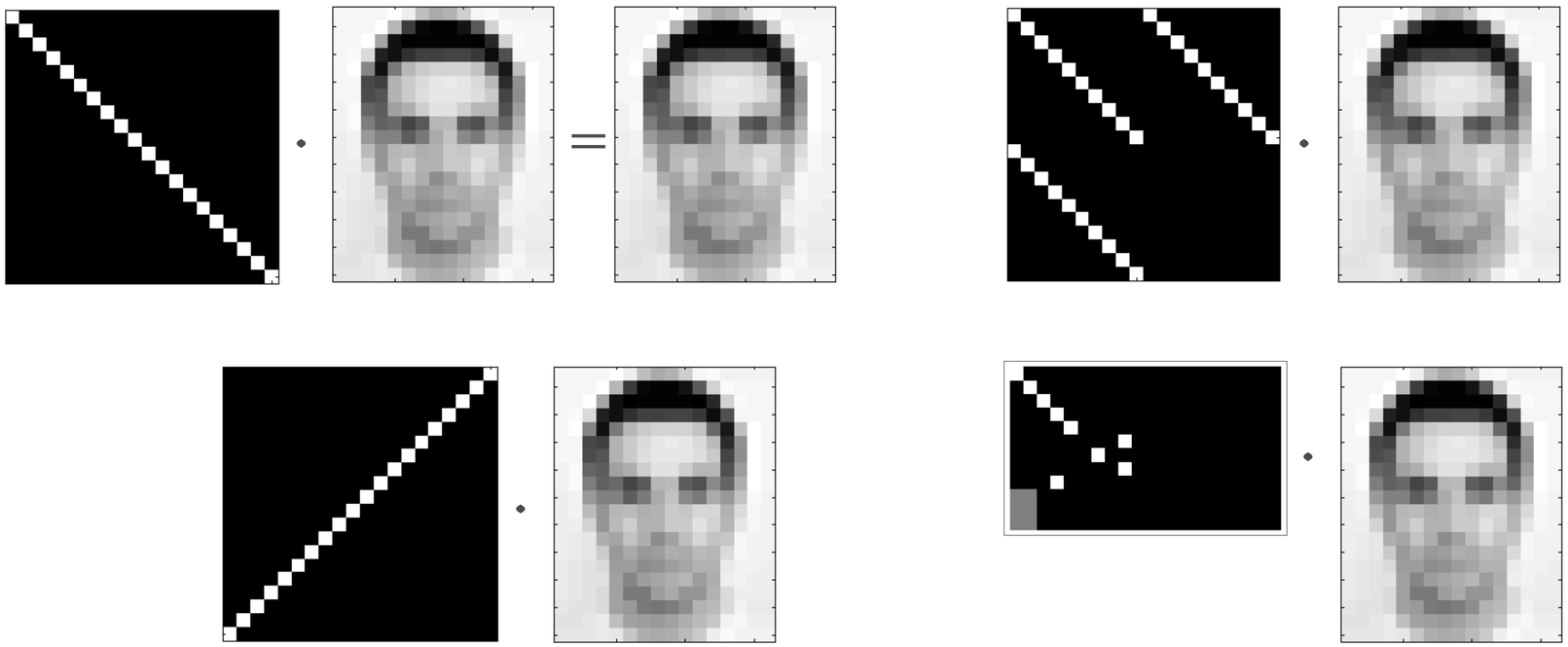




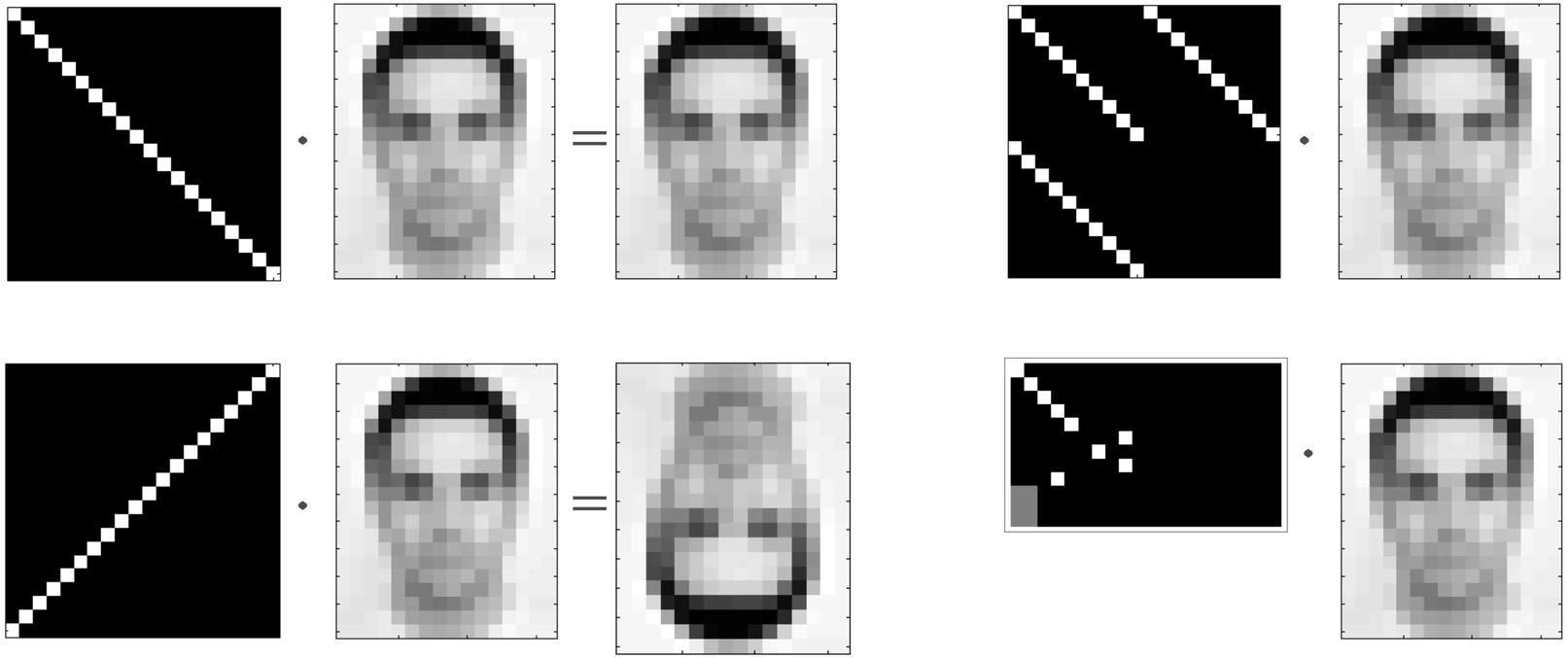
Visualizing the matrix product



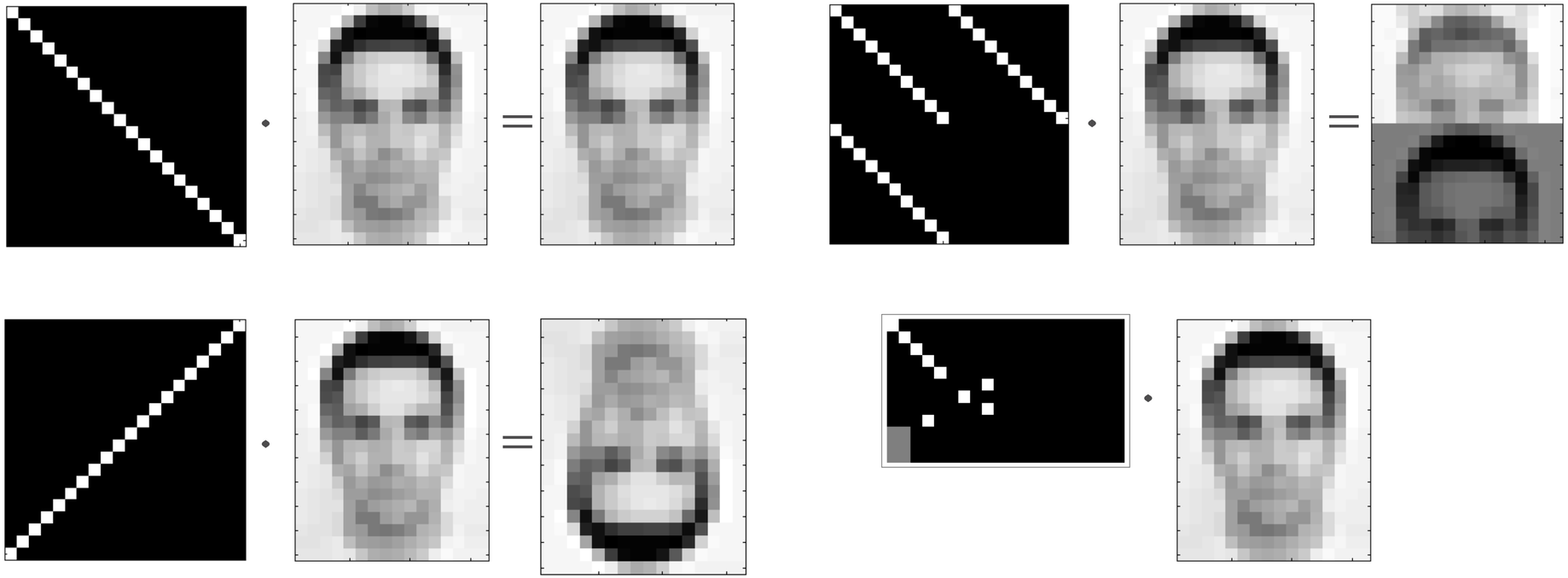
Visualizing the matrix product



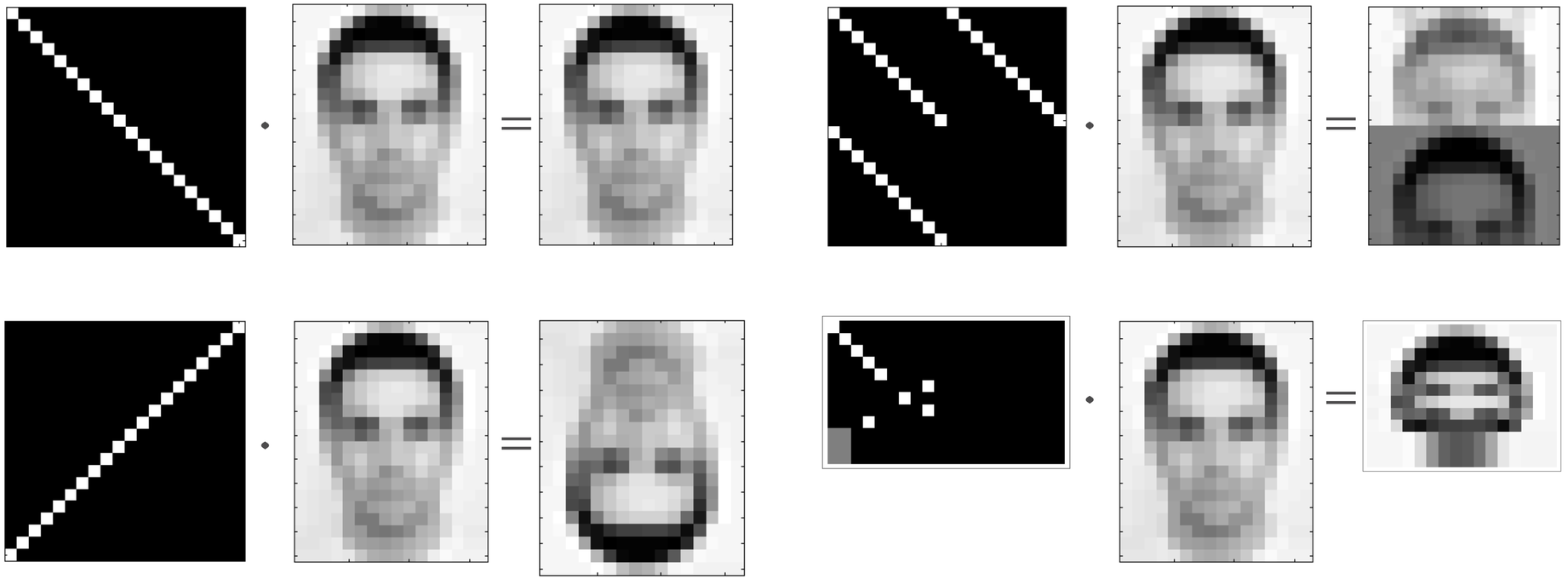
Visualizing the matrix product



Visualizing the matrix product

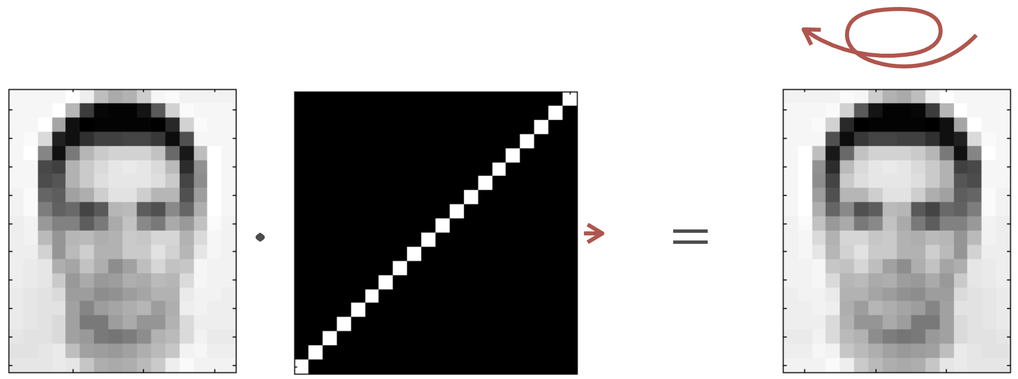


Visualizing the matrix product



How do we get to rearrange the columns?

- Previous examples rearranged the rows
  - What if we want to rearrange columns?



  - Multiply with a matrix from the right!

Kronecker product

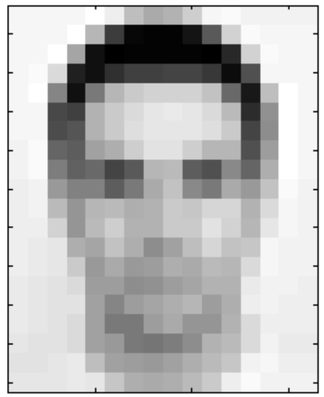
- A bit more complex
  - Replicate and multiply right matrix with each scalar of left matrix

$$\begin{bmatrix} x_{11} & x_{12} \\ x_{21} & x_{22} \end{bmatrix} \otimes \mathbf{Y} = \begin{bmatrix} x_{11} \cdot \mathbf{Y} & x_{12} \cdot \mathbf{Y} \\ x_{21} \cdot \mathbf{Y} & x_{22} \cdot \mathbf{Y} \end{bmatrix}$$

- Useful result:

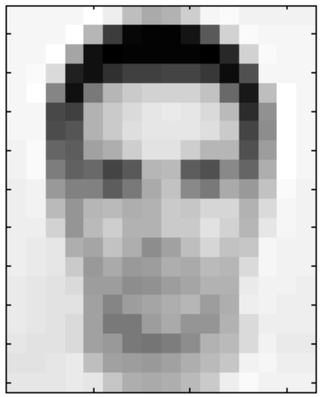
$$\mathrm{vec}(\mathbf{X} \cdot \mathbf{Y} \cdot \mathbf{Z}) = (\mathbf{Z}^{\top} \otimes \mathbf{X})\,\mathrm{vec}(\mathbf{Y})$$

Visualizing Kronecker

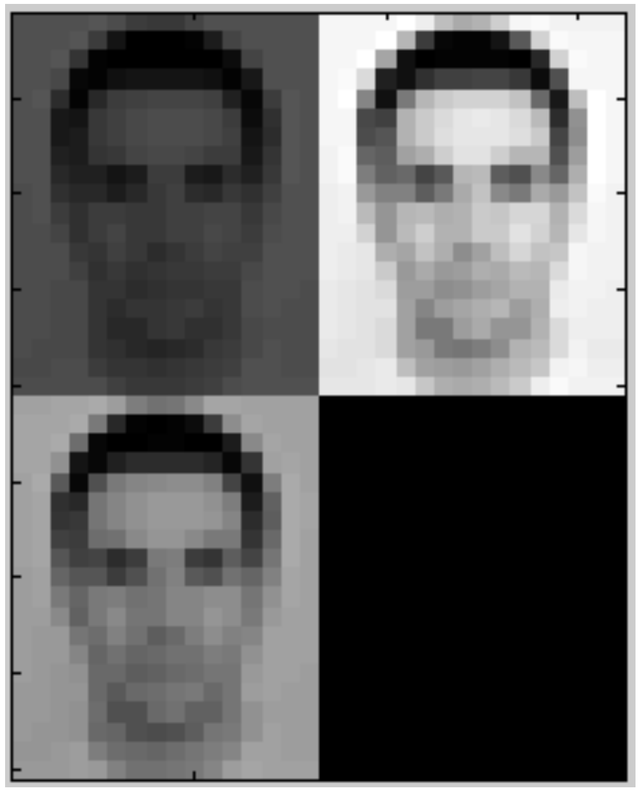
$$\begin{bmatrix} 1 & 3 \\ 2 & 0 \end{bmatrix} \otimes$$  $= ?$

Visualizing Kronecker

$$\begin{bmatrix} 1 & 3 \\ 2 & 0 \end{bmatrix} \otimes$$

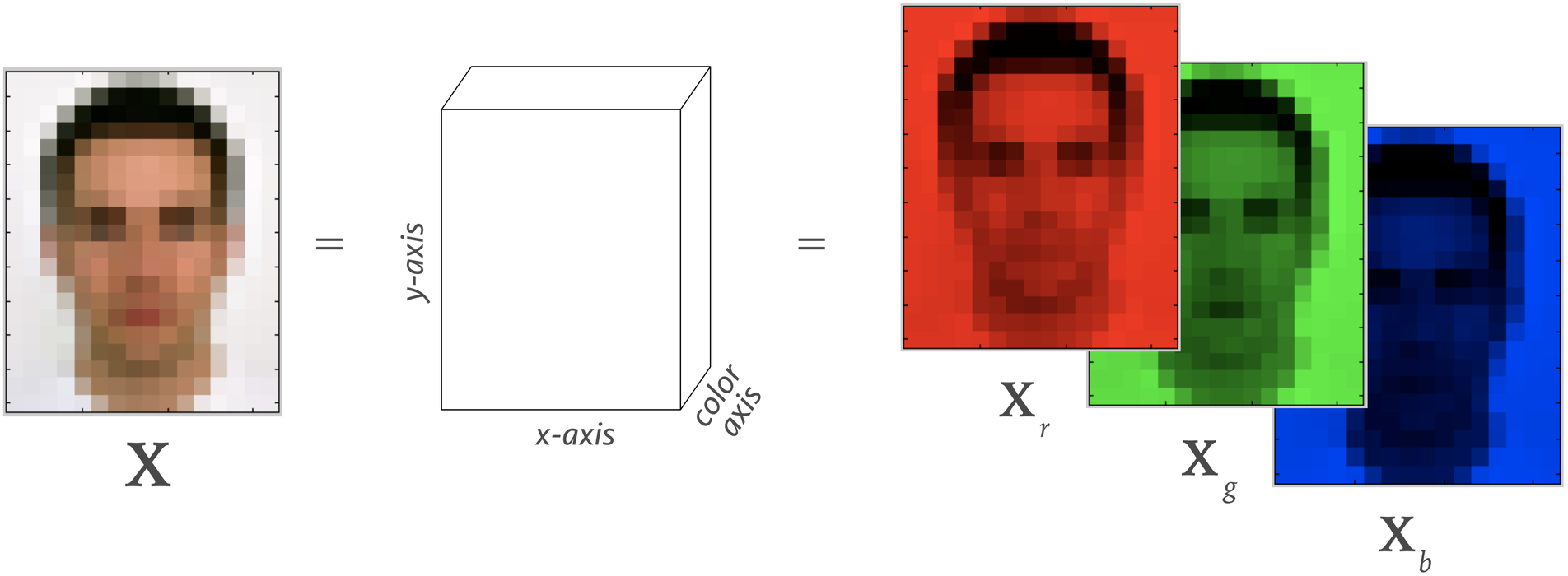


$=$



Dealing with tensors

- Using Kronecker products and the vec operator we can perform *multilinear transforms*
  - Tensor example with RGB images:



Mixing over all dimensions

- Define color, horizontal, and vertical mixing

$$\Big(\underbrace{\mathbf{C}^\top}_{3^{rd}\ dim} \otimes \underbrace{\mathbf{H}^\top}_{2^{nd}\ dim} \otimes \underbrace{\mathbf{V}^\top}_{1^{st}\ dim}\Big) \cdot \mathrm{vec}(\mathbf{X})$$

- Example: color mixing

$$\Big(\mathrm{diag}\begin{bmatrix} 1 & 0 & 1 \end{bmatrix} \otimes \mathbf{I} \otimes \mathbf{I}\Big)\mathrm{vec}(\mathbf{X}) =$$

  - Caution: Matrix sizes will quickly get out of hand this way

Special matrices

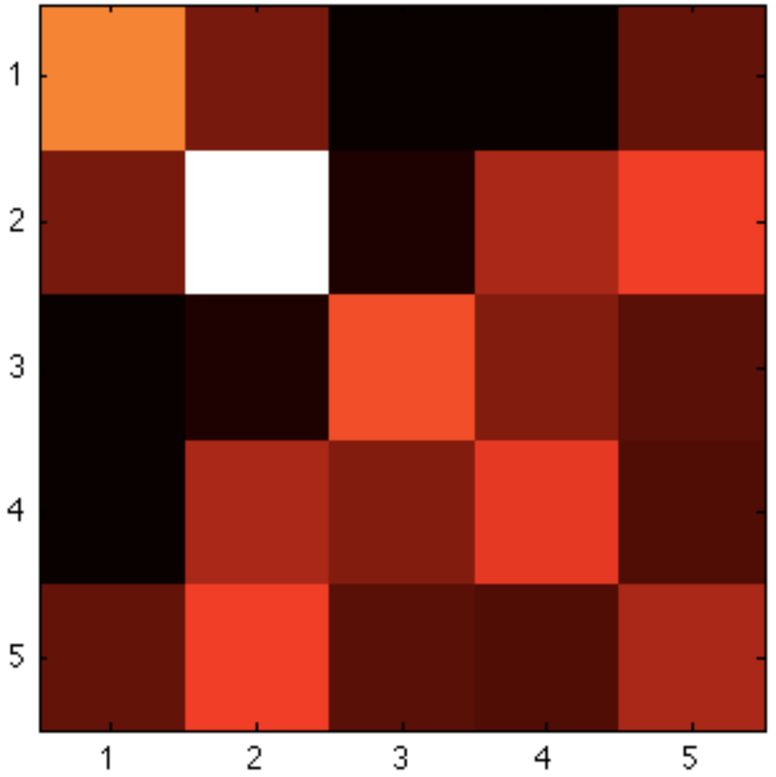
- Symmetric:

$$\mathbf{X} = \mathbf{X}^\top \Rightarrow x_{ij} = x_{ji}$$

- Positive definite
  - If: $\mathbf{y}^\top \cdot \mathbf{X} \cdot \mathbf{y} > 0, \quad \forall\ \mathbf{y}$
  - Also symmetric

- Orthonormal:

$$\mathbf{X}^\top \cdot \mathbf{X} = \mathbf{X} \cdot \mathbf{X}^\top = \mathbf{I}$$



Matrix inverse

- "Undoes" a matrix multiplication
  - Only for square matrices
  - Not all matrices are invertible
    - must be a full-rank matrix

$$\mathbf{X}^{-1} \cdot \mathbf{X} = \mathbf{I}$$

$$\mathbf{X}^{-1} \cdot \mathbf{X} \cdot \mathbf{Y} = \mathbf{Y}$$

$$\mathbf{Y} \cdot \mathbf{X} \cdot \mathbf{X}^{-1} = \mathbf{Y}$$

- Remember, in matrix multiplication order matters!

$$\mathbf{X} \cdot \mathbf{Y} \cdot \mathbf{X}^{-1} \neq \mathbf{Y}$$

Matrix pseudoinverse

- Also known as Moore-Penrose (or MP) pseudoinverse
  - For an $m \times n$ matrix $\mathbf{X}$, pseudoinverse is $n \times m$ matrix $\mathbf{X}^+$

$$\mathbf{X} \cdot \mathbf{X}^+ \cdot \mathbf{X} = \mathbf{X}$$
$$\mathbf{X}^+ \cdot \mathbf{X} \cdot \mathbf{X}^+ = \mathbf{X}^+$$
$$(\mathbf{X} \cdot \mathbf{X}^+)^\top = \mathbf{X} \cdot \mathbf{X}^+$$
$$(\mathbf{X}^+ \cdot \mathbf{X})^\top = \mathbf{X}^+ \cdot \mathbf{X}$$

  - We'll be seeing this operation a lot, it's essentially least squares

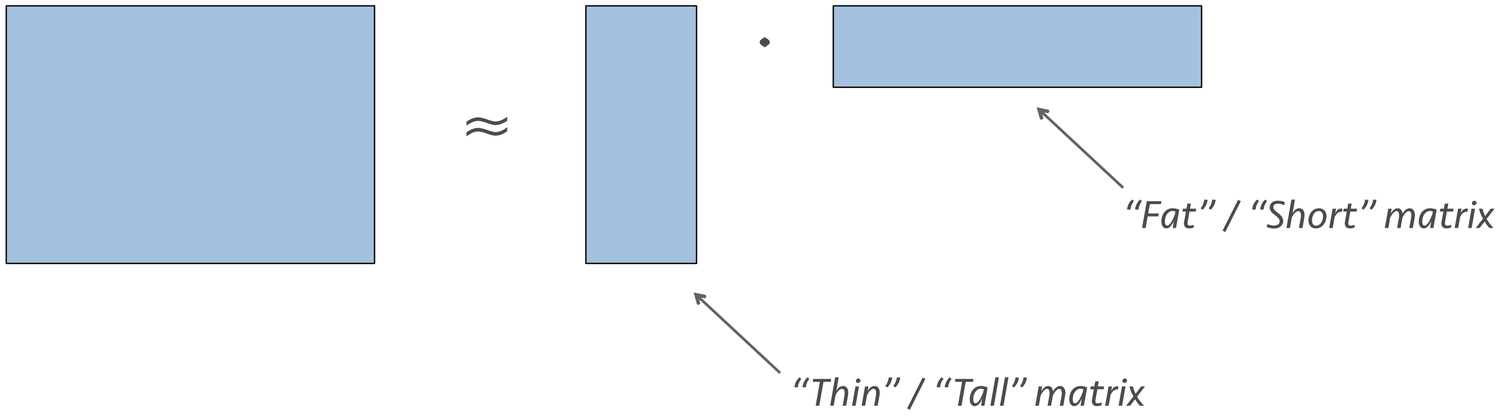
$$\mathbf{A} \cdot \mathbf{x} = \mathbf{y} \rightarrow \mathbf{x} = \mathbf{A}^+ \cdot \mathbf{y}$$

Low rank approximations

- Use smaller matrices to describe a large matrix

$$\mathbf{Y} \approx \mathbf{A} \cdot \mathbf{X}$$

  - With $\mathbf{Y}$ being $m \times n$, $\mathbf{A}$ being $m \times r$, $\mathbf{X}$ being $r \times n$, and $r < m$



Eigenanalysis

- Eigenvectors and eigenvalues
  - For an $n \times n$ matrix $\mathbf{X}$

$$\mathbf{X} \cdot \mathbf{V} = \mathbf{V} \cdot \mathbf{L}$$

$$\mathbf{V} = \begin{bmatrix} \mathbf{v}_1 & \cdots & \mathbf{v}_n \end{bmatrix}$$

$$\mathbf{L} = \mathrm{diag}\begin{bmatrix} \lambda_1 & \cdots & \lambda_n \end{bmatrix}$$

  - $\mathbf{V}$ is $n \times n$ and contains the eigenvectors $\mathbf{v}_i$
    - It will be an orthogonal matrix for positive (semi-)definite matrix inputs
  - $\mathbf{L}$ is $n \times n$ contains the eigenvalues λ_i

The Singular Value Decomposition (SVD)

- Similar decomposition to eigenanalysis
  - For a matrix $\mathbf{X}$

$$\mathbf{X} = \mathbf{U} \cdot \mathbf{S} \cdot \mathbf{V}^\top \quad \text{where} \quad \begin{aligned} \mathbf{U}^\top \cdot \mathbf{U} &= \mathbf{I} \\ \mathbf{V}^\top \cdot \mathbf{V} &= \mathbf{I} \\ \mathbf{S} &= \text{diag}(\sigma_i) \end{aligned}$$

  - $\mathbf{U}$, $\mathbf{V}$ are orthonormal
    - Contain the left and right singular vectors of $\mathbf{X}$
  - $\mathbf{S}$ is diagonal
    - Contains on its diagonal the singular values σ_i

Recap

- What's this class about?
  - Signals, learning, fun, etc …
- Linear algebra basics
  - Algebraic operations, norms, decompositions, linear transforms, form manipulations, tensors, special matrices, …

What now?

- Skim through this material for now
  - We'll be seeing it in context soon
    - e.g., what are the eigenvectors of image matrices?

- Reading material
  - Old and new algebra useful for statistics
    - http://research.microsoft.com/en-us/um/people/minka/papers/matrix/
  - The Matrix Cookbook
    - http://www2.imm.dtu.dk/pubdb/views/publication_details.php?id=3274
  - Best book ever on matrix calculus (use school's library to get PDF):
    - https://onlinelibrary.wiley.com/doi/book/10.1002/9781119541219

Questions?

- Slides are online on our MS Team files section

Next lecture

- Review of:
  - Probability theory
    - Bayes theorem, probability rules, Bayes nets, Basic distributions, transformations of RV's
  - Statistics
    - Basic measures, independence, information
  - Parameter estimation intro
    - Maximum likelihood, MAP, Bayesian, EM intro