The Resilience and Robustness (and Hopefully Responsibility) of Linear Algebra Lecture 2

Emily J. King

Mathematics Department and Data Science Research Institute
Colorado State University

IK Interdisciplinary College
March 2024

Course Outline

- Lecture 1
  - Signals and vector spaces
  - Linear combinations
- **Lecture 2**
  - **Inner product**
  - **Inner product as similarity**
  - **Moving average**
  - **Convolution and cross correlation**
  - **2*D* convolution and cross correlation**
- Lecture 3
  - Cosine similarity
  - Projection
  - Change of basis
  - Graph theory
- Lecture 4
  - Eigenvalues and eigenvectors
  - Principal component analysis
  - Responsibility vignette

Lecture Outline

What is an inner product?

Say you need to determine nutritional value of a recipe:

Magic-Spice Fruit Salad

- 4 apples chopped into 1/2 inch squares
- 1/2 cantaloupe sliced and sectioned into 1 inch pyramids
- 3/2 cups of grapes (no seeds)
- 1/2 honeydew melon sliced and sectioned
- Magic Spices, Honey drizzle

	Calories	Fat	Sodium	Fiber	Carbs	Protein
1 Apple	130	0g	0mg	5g	34g	1g
1 Banana	110	0g	0mg	3g	30g	1g
1 Cantaloupe	200	0g	80mg	4g	48g	4g
1 cup Grapes	90	0g	15mg	1g	23g	0g
1 Honeydew	500	0g	300mg	10g	120g	10g

We can determine the amount of calories in each fruit salad recipe by multiplying each fruit times the corresponding calories per serving in the FDA nutritional chart:

$$\textbf{4} \text{ apples} \quad \times \textbf{130} \text{ calories/apple} \qquad = \textbf{520} \text{ calories}$$

We can determine the amount of calories in each fruit salad recipe by multiplying each fruit times the corresponding calories per serving in the FDA nutritional chart:

$$\begin{array}{rll} \mathbf{4} \text{ apples} & \times \mathbf{130} \text{ calories/apple} & = \mathbf{520} \text{ calories} \\ \mathbf{0} \text{ bananas} & \times \mathbf{110} \text{ calories/banana} & = \mathbf{0} \text{ calories} \end{array}$$

We can determine the amount of calories in each fruit salad recipe by multiplying each fruit times the corresponding calories per serving in the FDA nutritional chart:

	4 apples	$\times$ **130** calories/apple	$=$ **520** calories
	0 bananas	$\times$ **110** calories/banana	$=$ **0** calories
	1/2 cantaloupe	$\times$ **200** calories/cantaloupe	$=$ **100** calories
	3/2 grape cups	$\times$ **90** calories/grape cup	$=$ **135** calories
$+$	**1/2** honeydew	$\times$ **500** calories/honeydew	$=$ **250** calories
			1005 calories

We can determine the amount of calories in each fruit salad recipe by multiplying each fruit times the corresponding calories per serving in the FDA nutritional chart:

	4 apples	$\times$ **130** calories/apple	$=$ **520** calories
	0 bananas	$\times$ **110** calories/banana	$=$ **0** calories
	1/2 cantaloupe	$\times$ **200** calories/cantaloupe	$=$ **100** calories
	3/2 grape cups	$\times$ **90** calories/grape cup	$=$ **135** calories
$+$	**1/2** honeydew	$\times$ **500** calories/honeydew	$=$ **250** calories
			1005 calories

So, each recipe has **1005** calories.

Since there are **4** servings per recipe, each serving has **1005/4** $\approx$ **250** calories.

Magic-Spice Fruit Salad
4 apples chopped into 1/2 inch squares
1/2 cantaloupe sliced and sectioned into 1 inch pyramids
3/2 cups of grapes (no seeds)
1/2 honeydew melon sliced and sectioned
Magic Spices, Honey drizzle

	Calories	Fat	Sodium	Fiber	Carbs	Protein
1 Apple	130	0g	0mg	5g	34g	1g
1 Banana	110	0g	0mg	3g	30g	1g
1 Cantaloupe	200	0g	80mg	4g	48g	4g
1 cup Grapes	90	0g	15mg	1g	23g	0g
1 Honeydew	500	0g	300mg	10g	120g	10g

Think:
What is the formula for the amount of fiber in one recipe of fruit salad?

We can determine the amount of fiber in each fruit salad recipe by multiplying each fruit times the corresponding calories per serving in the FDA nutritional chart:

$$\textbf{4} \text{ apples} \quad \times \textbf{5} \text{ g. fiber/apple} \qquad = \textbf{20} \text{ g. fiber}$$

We can determine the amount of fiber in each fruit salad recipe by multiplying each fruit times the corresponding calories per serving in the FDA nutritional chart:

$$\begin{array}{rll} \mathbf{4} \text{ apples} & \times \mathbf{5} \text{ g. fiber/apple} & = \mathbf{20} \text{ g. fiber} \\ \mathbf{0} \text{ bananas} & \times \mathbf{3} \text{ g. fiber/banana} & = \mathbf{0} \text{ g. fiber} \end{array}$$

We can determine the amount of fiber in each fruit salad recipe by multiplying each fruit times the corresponding calories per serving in the FDA nutritional chart:

	4 apples	$\times$ **5** g. fiber/apple	$=$ **20** g. fiber
	0 bananas	$\times$ **3** g. fiber/banana	$=$ **0** g. fiber
	1/2 cantaloupe	$\times$ **4** g. fiber/cantaloupe	$=$ **2** g. fiber
	3/2 grape cups	$\times$ **1** g. fiber/grape cup	$=$ **1.5** g. fiber
$+$	**1/2** honeydew	$\times$ **10** g. fiber/honeydew	$=$ **5** g. fiber
			28.5 g. fiber

We can determine the amount of fiber in each fruit salad recipe by multiplying each fruit times the corresponding calories per serving in the FDA nutritional chart:

	4 apples	$\times$ **5** g. fiber/apple	$=$ **20** g. fiber
	0 bananas	$\times$ **3** g. fiber/banana	$=$ **0** g. fiber
	1/2 cantaloupe	$\times$ **4** g. fiber/cantaloupe	$=$ **2** g. fiber
	3/2 grape cups	$\times$ **1** g. fiber/grape cup	$=$ **1.5** g. fiber
$+$	**1/2** honeydew	$\times$ **10** g. fiber/honeydew	$=$ **5** g. fiber
			28.5 g. fiber

So, each recipe has **28.5** grams of fiber,

meaning that each serving has **28.5/4** $\approx$ **7** grams of fiber.

Let's extract the numbers we just used and put them into vectors:

$$\begin{pmatrix} \mathbf{4} \\ \mathbf{0} \\ \mathbf{1/2} \\ \mathbf{3/2} \\ \mathbf{1/2} \end{pmatrix}, \begin{pmatrix} \mathbf{5} \\ \mathbf{3} \\ \mathbf{4} \\ \mathbf{1} \\ \mathbf{10} \end{pmatrix}$$

Let's extract the numbers we just used and put them into vectors:

$$\begin{pmatrix} \mathbf{4} \\ \mathbf{0} \\ \mathbf{1/2} \\ \mathbf{3/2} \\ \mathbf{1/2} \end{pmatrix}, \begin{pmatrix} \mathbf{5} \\ \mathbf{3} \\ \mathbf{4} \\ \mathbf{1} \\ \mathbf{10} \end{pmatrix}$$

We have two vectors with **5** entries each, where the first entries corresponded to apples, the second to bananas, and so on.

The first vector was amounts of those fruits in a recipe while the second was amounts of fiber in each fruit.

By multiplying the corresponding entries and adding up the products, we obtained information; namely, the amount of fiber in grams in each recipe.

Similarly, to obtain the amount of calories in each recipe, we multiplied the corresponding entries of

$$\begin{pmatrix} \mathbf{4} \\ \mathbf{0} \\ \mathbf{1/2} \\ \mathbf{3/2} \\ \mathbf{1/2} \end{pmatrix}, \begin{pmatrix} \mathbf{130} \\ \mathbf{110} \\ \mathbf{200} \\ \mathbf{90} \\ \mathbf{500} \end{pmatrix}$$

and added the products together.

Similarly, to obtain the amount of calories in each recipe, we multiplied the corresponding entries of

$$\begin{pmatrix} \mathbf{4} \\ \mathbf{0} \\ \mathbf{1/2} \\ \mathbf{3/2} \\ \mathbf{1/2} \end{pmatrix}, \begin{pmatrix} \mathbf{130} \\ \mathbf{110} \\ \mathbf{200} \\ \mathbf{90} \\ \mathbf{500} \end{pmatrix}$$

and added the products together.

This way of taking two vectors of the same type and outputting a single number is known as a inner product.

Similarly, to obtain the amount of calories in each recipe, we multiplied the corresponding entries of

$$\begin{pmatrix} \mathbf{4} \\ \mathbf{0} \\ \mathbf{1/2} \\ \mathbf{3/2} \\ \mathbf{1/2} \end{pmatrix}, \begin{pmatrix} \mathbf{130} \\ \mathbf{110} \\ \mathbf{200} \\ \mathbf{90} \\ \mathbf{500} \end{pmatrix}$$

and added the products together.

This way of taking two vectors of the same type and outputting a single number is known as a inner product.

This seemingly simple operation is the powerhouse of data science!

Let $\vec{x}$ and $\vec{y}$ be two vectors with d entries.

Then the (Euclidean) inner product (also known as dot product or scalar product) of $\vec{x}$ and $\vec{y}$ is

$$\begin{aligned}\langle \vec{x}, \vec{y} \rangle &= x_1 y_1 + x_2 y_2 + \ldots x_d y_d \\ &= \sum_{i=1}^{d} x_i y_i.\end{aligned}$$

Getting mathy again

Let V be a vector space.

An inner product on V is a map $V \times V \to \mathbb{R}$ such that

- $\langle \vec{x}, \vec{y} \rangle = \langle \vec{y}, \vec{x} \rangle$ for all $\vec{x}, \vec{y} \in V$.

Getting mathy again

Let V be a vector space.

An inner product on V is a map $V \times V \to \mathbb{R}$ such that

- $\langle \vec{x}, \vec{y} \rangle = \langle \vec{y}, \vec{x} \rangle$ for all $\vec{x}, \vec{y} \in V$.
- $\langle \vec{x}, \vec{x} \rangle \geq 0$ for all $\vec{x} \in V$.

Getting mathy again

Let V be a vector space.

An inner product on V is a map $V \times V \to \mathbb{R}$ such that

- $\langle \vec{x}, \vec{y} \rangle = \langle \vec{y}, \vec{x} \rangle$ for all $\vec{x}, \vec{y} \in V$.
- $\langle \vec{x}, \vec{x} \rangle \geq 0$ for all $\vec{x} \in V$.
- $\langle \vec{x}, \vec{x} \rangle = 0$ if and only if $\vec{x} = \vec{0}$.

Getting mathy again

Let V be a vector space.

An inner product on V is a map $V \times V \to \mathbb{R}$ such that

- $\langle \vec{x}, \vec{y} \rangle = \langle \vec{y}, \vec{x} \rangle$ for all $\vec{x}, \vec{y} \in V$.
- $\langle \vec{x}, \vec{x} \rangle \geq 0$ for all $\vec{x} \in V$.
- $\langle \vec{x}, \vec{x} \rangle = 0$ if and only if $\vec{x} = \vec{0}$.
- $\langle a\vec{x}, \vec{y} \rangle = a \langle \vec{x}, \vec{y} \rangle$ for all $\vec{x} \in V$ and $a \in \mathbb{R}$.

Getting mathy again

Let V be a vector space.

An inner product on V is a map $V \times V \to \mathbb{R}$ such that

- $\langle \vec{x}, \vec{y} \rangle = \langle \vec{y}, \vec{x} \rangle$ for all $\vec{x}, \vec{y} \in V$.
- $\langle \vec{x}, \vec{x} \rangle \geq 0$ for all $\vec{x} \in V$.
- $\langle \vec{x}, \vec{x} \rangle = 0$ if and only if $\vec{x} = \vec{0}$.
- $\langle a\vec{x}, \vec{y} \rangle = a \langle \vec{x}, \vec{y} \rangle$ for all $\vec{x} \in V$ and $a \in \mathbb{R}$.
- $\langle \vec{x} + \vec{z}, \vec{y} \rangle = \langle \vec{x}, \vec{y} \rangle + \langle \vec{z}, \vec{y} \rangle$ for all $\vec{x}, \vec{y}, \vec{z} \in V$.

The application of inner product to recipes is related to linear cost functions that might appear in applications in business.

The application of inner product to recipes is related to linear cost functions that might appear in applications in business.

In data science and other fields, the most informative uses of the inner product are as a similarity measure.

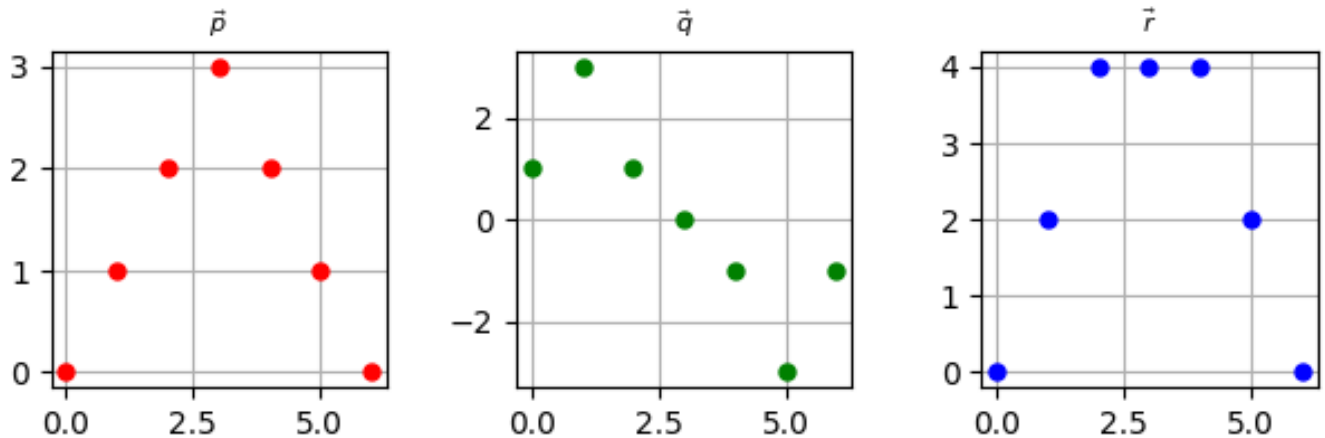
Lecture Outline

Inner product as scaled similarity measure

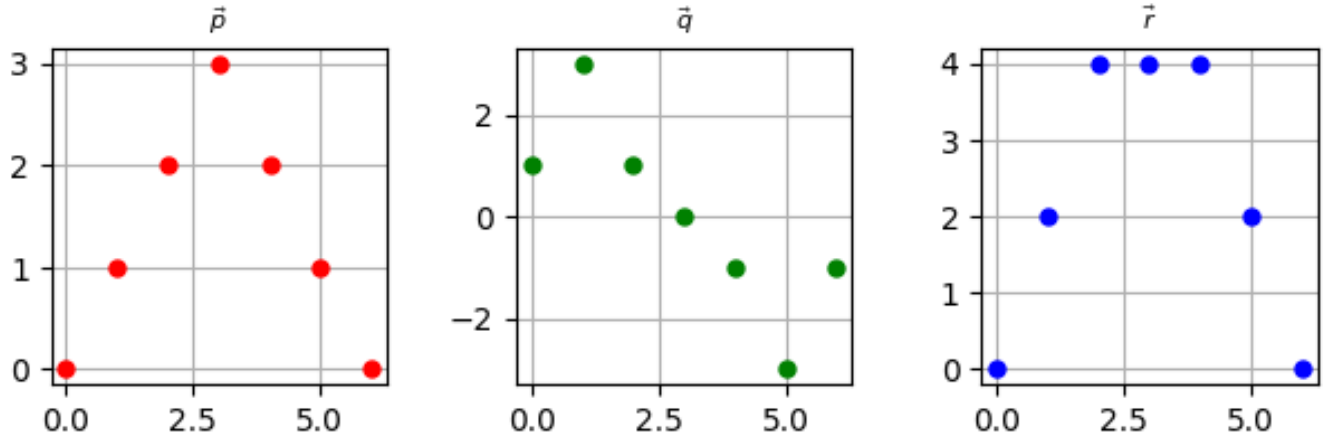
Consider the following vectors:

$$\vec{p}^{\top} = \begin{pmatrix} 0 & 1 & 2 & 3 & 2 & 1 & 0 \end{pmatrix},$$
$$\vec{q}^{\top} = \begin{pmatrix} 1 & 3 & 1 & 0 & -1 & -3 & -1 \end{pmatrix},$$
$$\vec{r}^{\top} = \begin{pmatrix} 0 & 2 & 4 & 4 & 4 & 2 & 0 \end{pmatrix},$$

visualized below



Which pair has the greatest inner product? Why?



We compute

$$\langle \vec{p}, \vec{q} \rangle = 0(1) + 1(3) + 2(1) + 3(0) + 2(-1) + 1(-3) + 0(-1) = 0,$$
$$\langle \vec{p}, \vec{r} \rangle = 0(0) + 1(2) + 2(4) + 3(4) + 2(4) + 1(2) + 0(0) = 32 \text{ and}$$
$$\langle \vec{q}, \vec{r} \rangle = 1(0) + 3(2) + 1(4) + 0(4) - 1(4) - 3(2) - 1(0) = 0.$$

Neural networks

A layer of a (feedforward, fully connected) neural network is a function that takes an input vector $\vec{x}$ and maps it first to

$$\begin{pmatrix} \langle \vec{x}, \vec{a}_1 \rangle \\ \langle \vec{x}, \vec{a}_2 \rangle \\ \vdots \\ \langle \vec{x}, \vec{a}_n \rangle \end{pmatrix} + \vec{b},$$

where $\vec{a}_1, \vec{a}_2, \ldots, \vec{a}_n$ are all vectors of the same type as $\vec{x}$ and $\vec{b}$ has n entries,

before applying a so-called activation function.

Training a neural network means learning the best $\vec{a}_1, \vec{a}_2, \ldots, \vec{a}_n, \vec{b}$ to use for your goals.

Lecture Outline

Let's compute

$$\left\langle \begin{pmatrix} \mathbf{5} \\ \mathbf{-3} \\ \mathbf{2} \end{pmatrix}, \begin{pmatrix} \mathbf{1/3} \\ \mathbf{1/3} \\ \mathbf{1/3} \end{pmatrix} \right\rangle =$$

Let's compute

$$\left\langle \begin{pmatrix} 5 \\ -3 \\ 2 \end{pmatrix}, \begin{pmatrix} 1/3 \\ 1/3 \\ 1/3 \end{pmatrix} \right\rangle =$$

$$\left\langle \begin{pmatrix} 2 \\ 1 \\ 0 \\ -1 \\ -2 \end{pmatrix}, \begin{pmatrix} 1/5 \\ 1/5 \\ 1/5 \\ 1/5 \\ 1/5 \end{pmatrix} \right\rangle =$$

Let's compute

$$\left\langle \begin{pmatrix} 5 \\ -3 \\ 2 \end{pmatrix}, \begin{pmatrix} 1/3 \\ 1/3 \\ 1/3 \end{pmatrix} \right\rangle =$$

$$\left\langle \begin{pmatrix} 2 \\ 1 \\ 0 \\ -1 \\ -2 \end{pmatrix}, \begin{pmatrix} 1/5 \\ 1/5 \\ 1/5 \\ 1/5 \\ 1/5 \end{pmatrix} \right\rangle =$$

$$\left\langle \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_d \end{pmatrix}, \begin{pmatrix} 1/d \\ 1/d \\ \vdots \\ 1/d \end{pmatrix} \right\rangle =$$

Taking the inner product of any vector $\vec{x}$ with d entries

with the vector with all entries equal to $1/d$

results in the average value of the entries of $\vec{x}$.[1]

[1] According to Benjamin, we've just done machine learning. :-.D

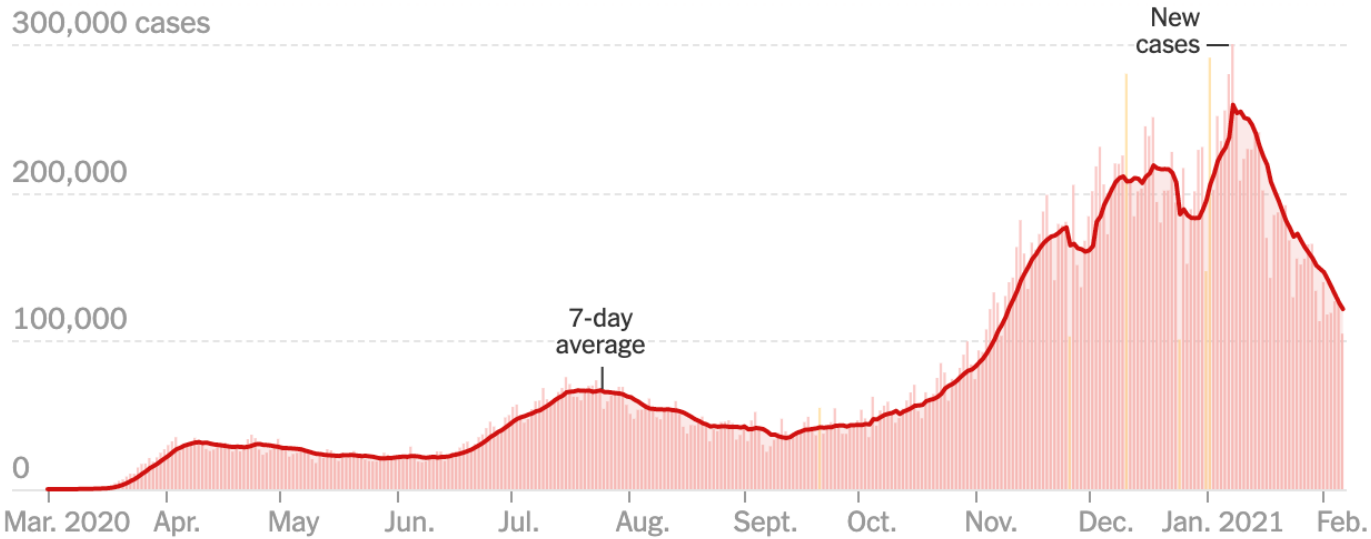
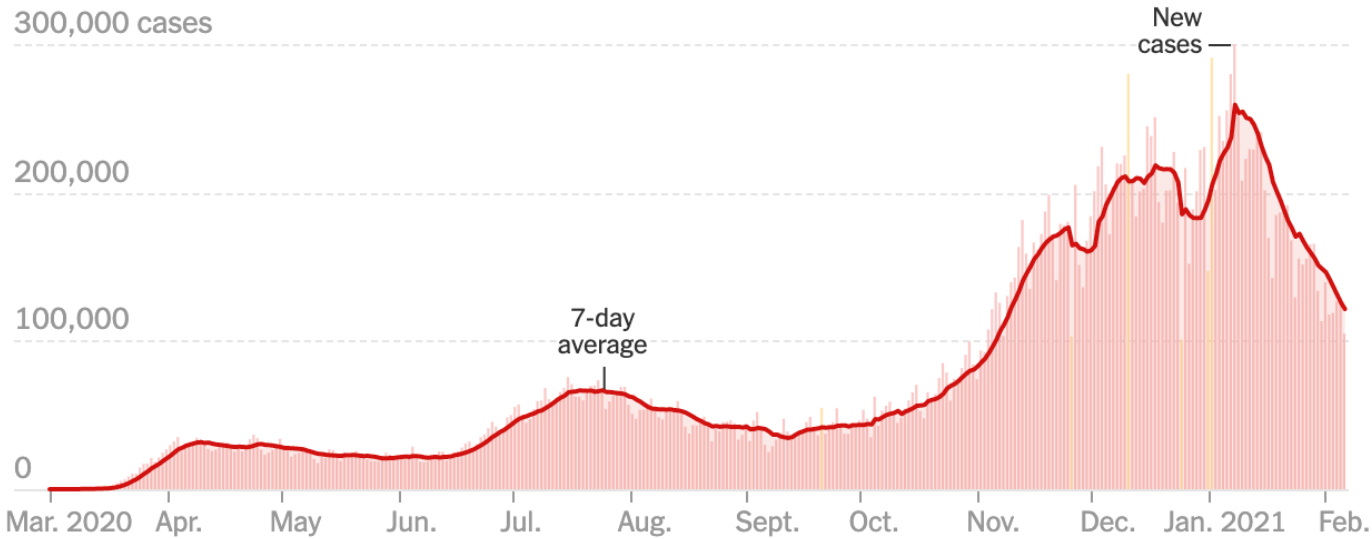


Image source: New York Times February 7, 2021.

Because COVID numbers were not reported consistently,

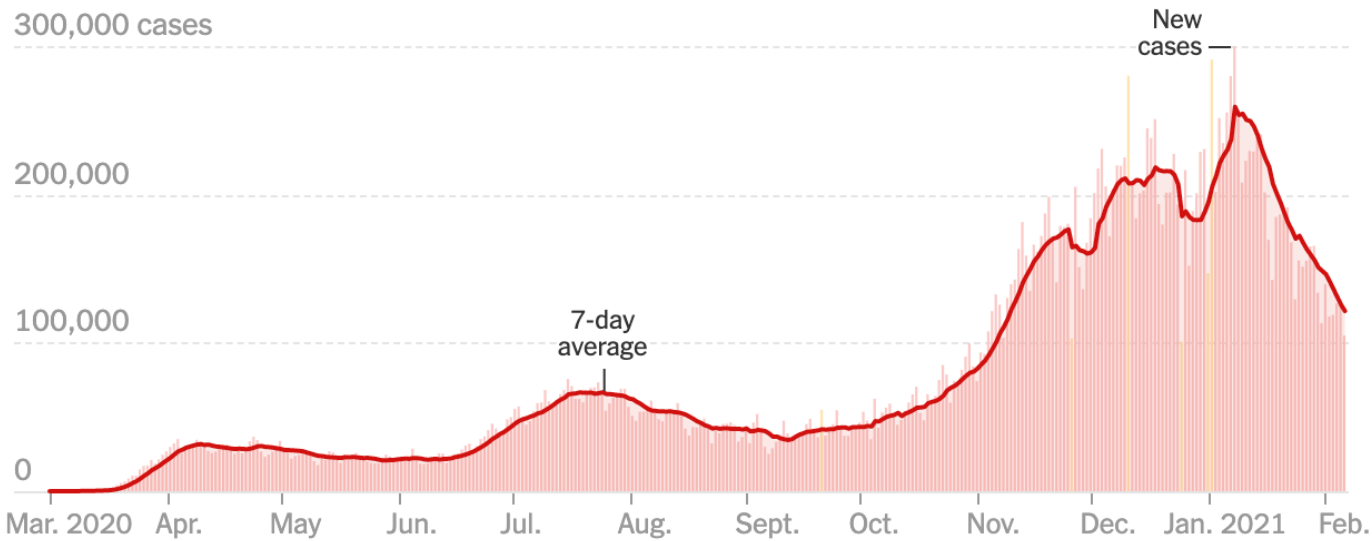
e.g., some municipalities didn't report on Sundays or fell behind in testing,

a moving average was a truer representation of the numbers.



Let $\vec{a} = \begin{pmatrix} 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 & 1/7 \end{pmatrix}$.

The plot above was generated by taking the inner product of the first seven days of case numbers with $\vec{a}$, then the inner product of the second through eighth days of case numbers with $\vec{a}$, then the inner product of the third through ninth days with $\vec{a}$ ...



The values of those inner products are plotted as the thick red line.

Given a vector $\vec{x}$,

taking inner products of d consecutive elements of $\vec{x}$ at a time with $\underbrace{\begin{pmatrix} 1/d & \dots & 1/d \end{pmatrix}}_{d \text{ entries}}$

is called a moving average (of window length d),

and $\underbrace{\begin{pmatrix} 1/d & \dots & 1/d \end{pmatrix}}_{d \text{ entries}}$ is called the window.

Zooming in on a **3**-entry moving average:

Position	...	6	7	8	9	10	11	...
Input:	...	$x_6 = 10$	$x_7 = 11$	$x_8 = 0$	$x_9 = 5$	$x_{10} = 9$	$x_{11} = -1$	...
Output:	...	$\frac{x_5+10+11}{3}$	$\frac{10+11+0}{3}$	$\frac{11+0+5}{3}$	$\frac{0+5+9}{3}$	$\frac{5+9-1}{3}$	$\frac{9-1+x_{12}}{3}$	...

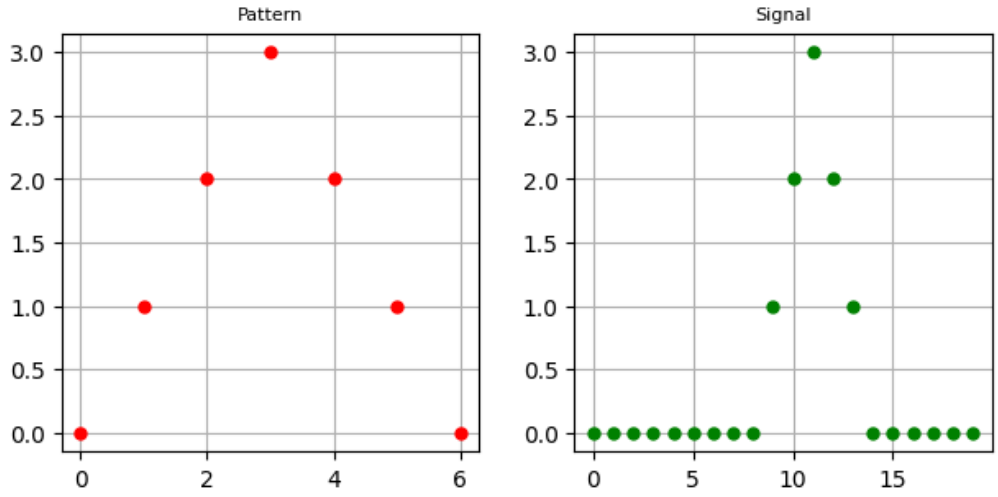
(Go to Matlab)

Taking sliding inner products with different kinds of vectors,

not just ones of the form $\begin{pmatrix} 1/d & \dots & 1/d \end{pmatrix}$

is called convolution or cross-correlation.

Lecture Outline



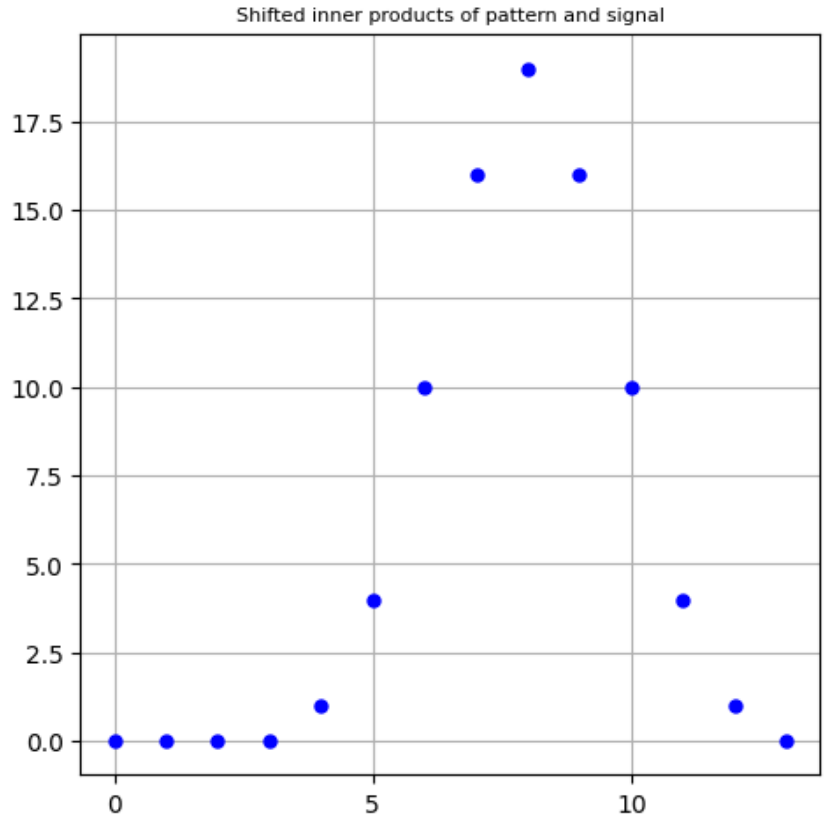
Left: A pattern $\vec{p}$ that we are trying to find. Right: A signal $\vec{s}$ we are trying to analyze.

$$\vec{p}^{\top} = \begin{pmatrix} 0 & 1 & 2 & 3 & 2 & 1 & 0 \end{pmatrix}$$

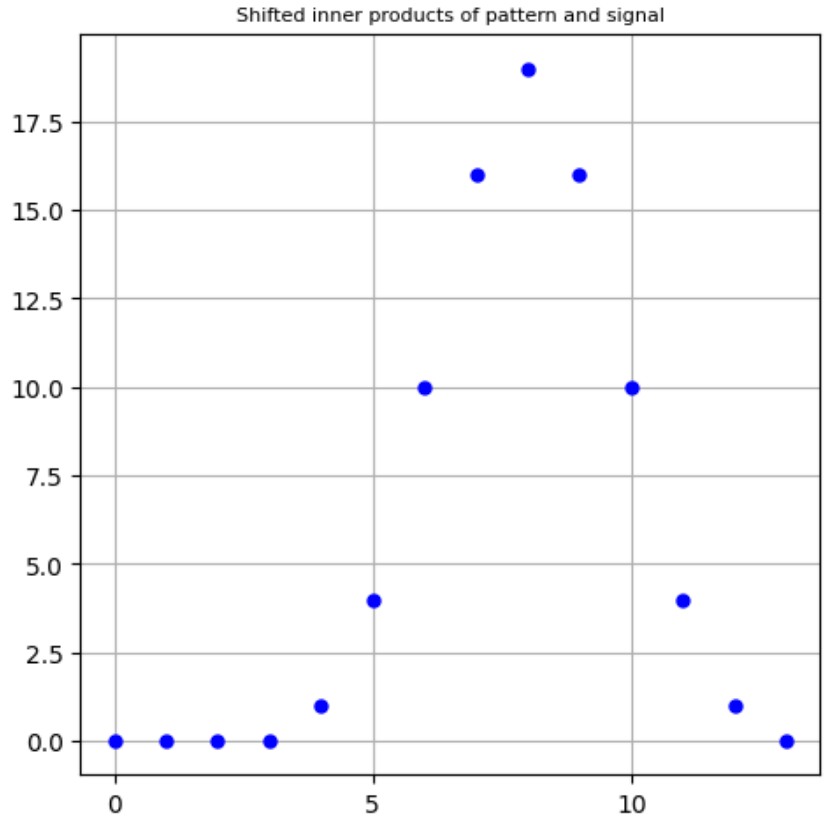
$$\vec{s}^{\top} =$$

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

Much like with a moving average, we want to slide the pattern $\vec{p}$ along the signal $\vec{s}$ and take inner products to find where pattern-like shapes are.

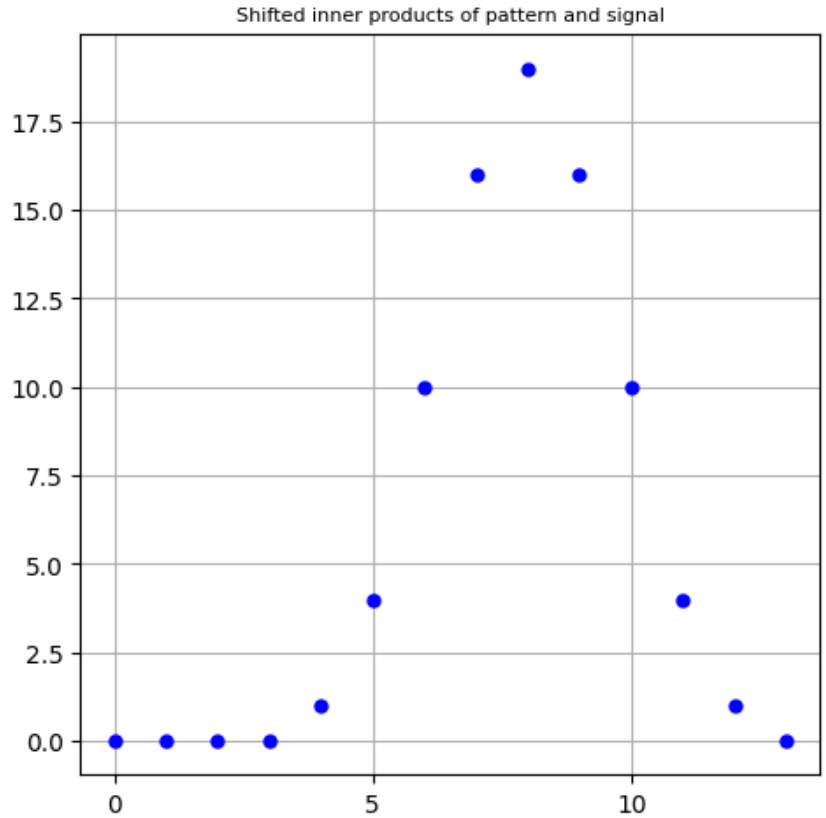


The sliding inner products / shifted inner products / cross-correlation / "convolution" $\vec{c}$ of the pattern $\vec{p}$ with the signal $\vec{s}$.



The sliding inner products / shifted inner products / cross-correlation / "convolution" $\vec{c}$ of the pattern $\vec{p}$ with the signal $\vec{s}$.

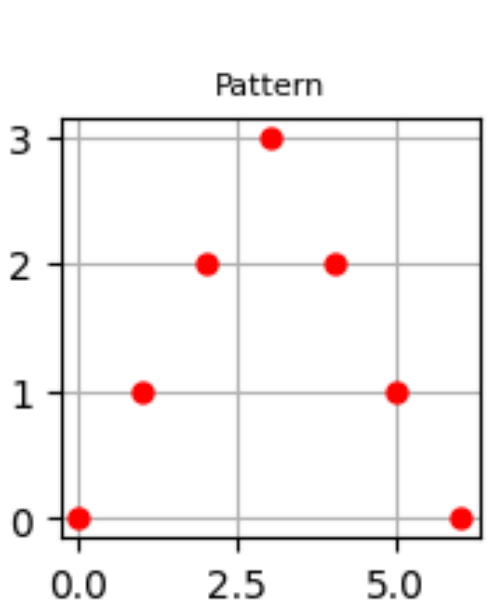
The location of the highest value in $\vec{c}$ tells the shift of $\vec{p}$ that best matches $\vec{s}$.

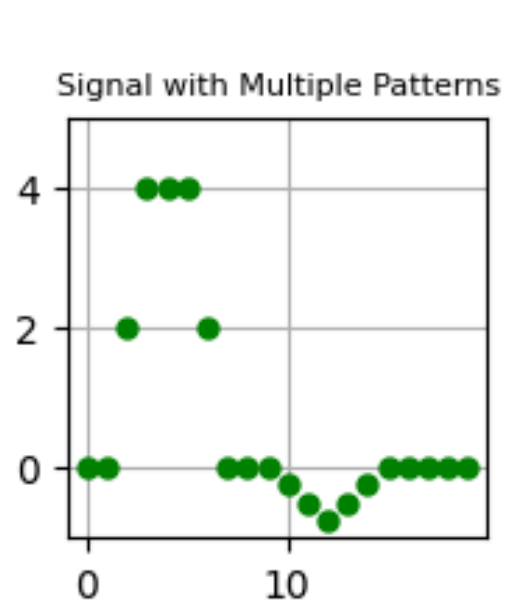


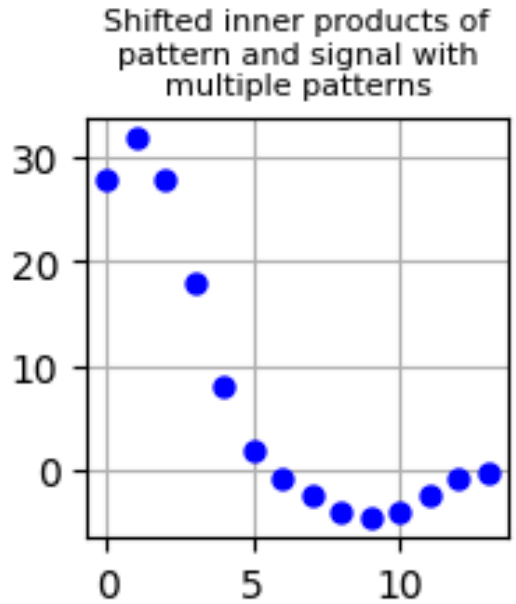
The sliding inner products / shifted inner products / cross-correlation / "convolution" $\vec{c}$ of the pattern $\vec{p}$ with the signal $\vec{s}$.

Notice that $\vec{c}$ is "smoother" than $\vec{s}$.

SIGNAL WITH MULTIPLE FEATURES







Go to GIFs

https://en.wikipedia.org/wiki/File:
Convolution_of_box_signal_with_itself2.gif

https://en.wikipedia.org/wiki/File:
Convolution_of_spiky_function_with_box2.gif

Let $\vec{p} \in \mathbb{R}^d$ and $\vec{s} \in \mathbb{R}^n$ with $n \geq d$.

Then the sliding inner product, also known as the cross-correlation of $\vec{p}$ and $\vec{s}$,

denoted $\vec{p} \star \vec{s}$,

is the vector in $\mathbb{R}^{n-d+1}$ with entries

$$(\vec{p} \star \vec{s})_i = \left\langle \vec{p}, \begin{pmatrix} s_i \\ s_{i+1} \\ \vdots \\ s_{i+d-1} \end{pmatrix} \right\rangle, \quad 1 \leq i \leq n-d+1.$$

The vector $\vec{p}$ is often called the window or kernel.

What's in a name?

Cross-correlation is often incorrectly called convolution in data science.

Convolution is a related operation, except the window is "flipped" before being slid around to make inner products.

Convolution is often denoted with $*$ not $\star$.

More variants

There are different ways to handle the end points

i.e.: Can we slide $\vec{p}$ past the ends of $\vec{s}$?

More variants

There are different ways to handle the end points

i.e.: Can we slide $\vec{p}$ past the ends of $\vec{s}$?

If we slide $\vec{p}$ past, do the extra values of $\vec{p}$ wrap around or not?

More variants

There are different ways to handle the end points

i.e.: Can we slide $\vec{p}$ past the ends of $\vec{s}$?

If we slide $\vec{p}$ past, do the extra values of $\vec{p}$ wrap around or not?

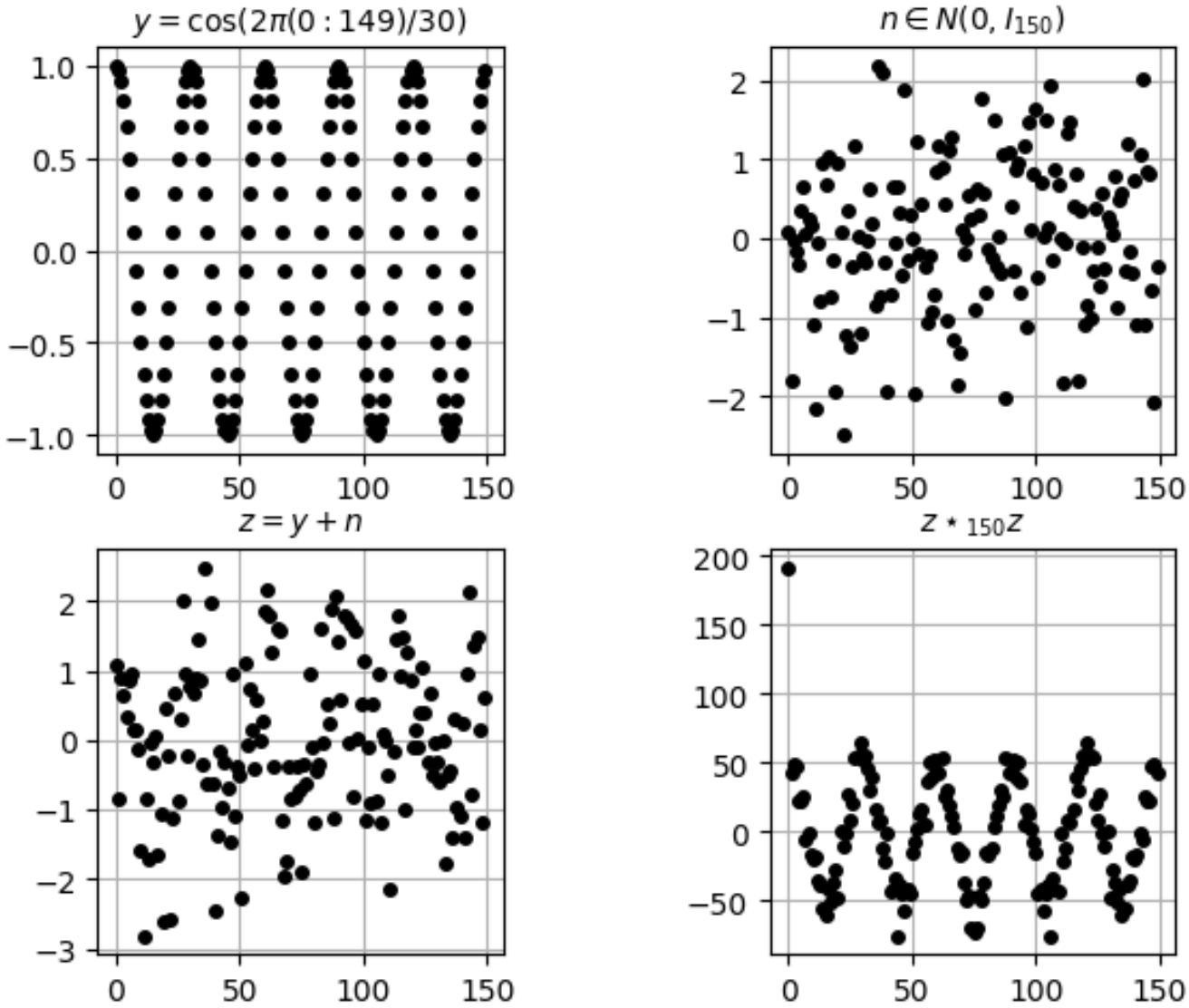
Also, one can slide $\vec{p}$ by larger amounts than $\mathbf{1}$.

This is called stride.

$\vec{s} \star \vec{s}$ is called the autocorrelation of $\vec{s}$.

What do you think:

Why might one compute an autocorrelation?



The $\star_{150}$ means that the values cyclicly wrapped around a the ends.

Lecture Outline

Inner product

Inner product as similarity

Moving average

Convolution & cross correlation

2*D* Convolution / cross correlation

"Convolution" is often applied to images,

in particular in convolutional neural networks,

where one trains on data to learn the convolutional kernel.

When convolution is applied to images, the inner product used is the Frobenius inner product.

2D MOVING AVERAGE

Say we wanted a **2D** moving average. We might compute:

$$\begin{pmatrix} \boxed{5} & \boxed{0} & 1 & -1 \\ \boxed{-1} & \boxed{2} & 0 & 6 \\ 1 & 1 & 1 & 1 \\ 2 & 4 & 2 & 4 \\ 0 & 1 & 0 & -1 \end{pmatrix} \qquad \boxed{\begin{matrix} 5 & 0 \\ -1 & 2 \end{matrix}} \longrightarrow \frac{5+0+(-1)+2}{4} = 1.5$$

...

2D MOVING AVERAGE

Say we wanted a **2D** moving average. We might compute:

$$\begin{pmatrix} 5 & 0 & 1 & -1 \\ -1 & 2 & 0 & 6 \\ 1 & 1 & 1 & 1 \\ 2 & 4 & 2 & 4 \\ 0 & 1 & 0 & -1 \end{pmatrix}$$

$$\begin{matrix} 0 & 1 \\ 2 & 0 \end{matrix} \longrightarrow \frac{0+1+2+0}{4} = 0.75$$

...

2D MOVING AVERAGE

Say we wanted a **2D** moving average. We might compute:

$$\begin{pmatrix} 5 & 0 & 1 & -1 \\ -1 & 2 & 0 & 6 \\ 1 & 1 & 1 & 1 \\ 2 & 4 & 2 & 4 \\ 0 & 1 & 0 & -1 \end{pmatrix} \qquad \begin{matrix} 1 & -1 \\ 0 & 6 \end{matrix} \longrightarrow \frac{1-1+0+6}{4} = 1.5$$

...

2D MOVING AVERAGE

Say we wanted a **2D** moving average. We might compute:

$$\begin{pmatrix} 5 & 0 & 1 & -1 \\ -1 & 2 & 0 & 6 \\ 1 & 1 & 1 & 1 \\ 2 & 4 & 2 & 4 \\ 0 & 1 & 0 & -1 \end{pmatrix} \qquad \begin{matrix} -1 & 2 \\ 1 & 1 \end{matrix} \longrightarrow \frac{-1+2+1+1}{4} = 0.75$$

...

...

$$\begin{pmatrix} 1.5 & 0.75 & 1.5 \\ 0.75 & 1 & 2 \\ 2 & 2 & 2 \\ 1.75 & 1.75 & 1.25 \end{pmatrix}$$

Alternate ways to handle boundaries I

One might zero-pad, e.g.:

$$\begin{pmatrix} \mathbf{0} & \mathbf{0} & 0 & 0 & 0 & 0 \\ \mathbf{0} & \mathbf{5} & 0 & 1 & -1 & 0 \\ 0 & -1 & 2 & 0 & 6 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 2 & 4 & 2 & 4 & 0 \\ 0 & 0 & 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

$$\begin{matrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{5} \end{matrix} \longrightarrow \frac{0+0+0+5}{4} = \mathbf{1.25}$$

Alternate ways to handle boundaries II

One might perform the operation cyclicly, e.g.:

$$\left(\begin{array}{cccc|cccc|cccc}
5 & 0 & 1 & -1 & 5 & 0 & 1 & -1 & 5 & 0 & 1 & -1 \\
-1 & 2 & 0 & 6 & -1 & 2 & 0 & 6 & -1 & 2 & 0 & 6 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
2 & 4 & 2 & 4 & 2 & 4 & 2 & 4 & 2 & 4 & 2 & 4 \\
0 & 1 & 0 & -1 & 0 & 1 & 0 & -1 & 0 & 1 & 0 & -1 \\
\hline
5 & 0 & 1 & -1 & 5 & 0 & 1 & -1 & 5 & 0 & 1 & -1 \\
-1 & 2 & 0 & 6 & -1 & 2 & 0 & 6 & -1 & 2 & 0 & 6 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
2 & 4 & 2 & 4 & 2 & 4 & 2 & 4 & 2 & 4 & 2 & 4 \\
0 & 1 & 0 & -1 & 0 & 1 & 0 & -1 & 0 & 1 & 0 & -1 \\
\hline
5 & 0 & 1 & -1 & 5 & 0 & 1 & -1 & 5 & 0 & 1 & -1 \\
-1 & 2 & 0 & 6 & -1 & 2 & 0 & 6 & -1 & 2 & 0 & 6 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
2 & 4 & 2 & 4 & 2 & 4 & 2 & 4 & 2 & 4 & 2 & 4 \\
0 & 1 & 0 & -1 & 0 & 1 & 0 & -1 & 0 & 1 & 0 & -1
\end{array}\right)$$

Alternate ways to handle boundaries III

One might compute a smaller average, e.g.:

$$\begin{pmatrix} \mathbf{5} & \mathbf{0} & \mathbf{1} & \mathbf{-1} \\ \mathbf{-1} & \mathbf{2} & \mathbf{0} & \mathbf{6} \\ \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} \\ \mathbf{2} & \mathbf{4} & \mathbf{2} & \mathbf{4} \\ \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{-1} \end{pmatrix} \qquad \mathbf{5} \longrightarrow \tfrac{5}{1} = \mathbf{5}$$

Alternate ways to handle boundaries III

One might compute a smaller average, e.g.:

$$\begin{pmatrix} \mathbf{5} & \mathbf{0} & \mathbf{1} & \mathbf{-1} \\ \mathbf{-1} & \mathbf{2} & \mathbf{0} & \mathbf{6} \\ \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} \\ \mathbf{2} & \mathbf{4} & \mathbf{2} & \mathbf{4} \\ \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{-1} \end{pmatrix}$$

$$\mathbf{5 \quad 0} \longrightarrow \mathbf{\frac{5+0}{2} = 2.5}$$

Frobenius inner product

The Frobenius inner product of two $d \times n$ matrices $\mathbf{A}$ and $\mathbf{B}$ is

$$\langle \mathbf{A}, \mathbf{B} \rangle_{\text{Fro}} = \sum_{i=1}^{d} \sum_{j=1}^{n} A_{i,j} B_{i,j}.$$

Frobenius inner product

The Frobenius inner product of two $d \times n$ matrices $\mathbf{A}$ and $\mathbf{B}$ is

$$\langle \mathbf{A}, \mathbf{B} \rangle_{\text{Fro}} = \sum_{i=1}^{d} \sum_{j=1}^{n} A_{i,j} B_{i,j}.$$

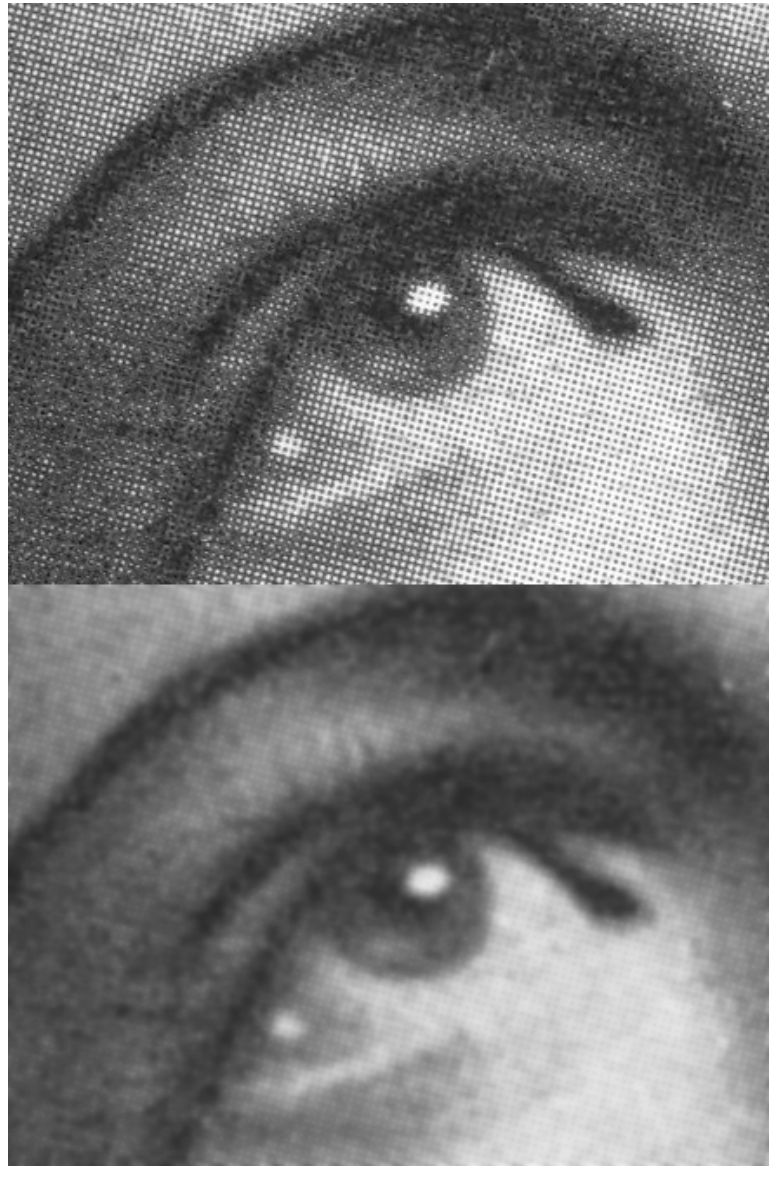
E.g.,

$$\left\langle \begin{pmatrix} 5 & 0 \\ -1 & 2 \end{pmatrix}, \begin{pmatrix} 1/4 & 1/4 \\ 1/4 & 1/4 \end{pmatrix} \right\rangle_{\text{Fro}} = 5(1/4) + 0(1/4) + (-1)(1/4) + 2(1/4$$

$$= \frac{5 + 0 + (-1) + 2}{4} = 1.5$$

Just as **1*D*** moving averages generalized to convolution / cross-correlation of vectors,

2*D* moving averages generalize to convolution / cross-correlation of matrices.



Blurring/smoothing is one example of an application of convolution/correlation. “Halftone images can be smoothed digitaly by applying a Gaussian blur filter. Original image taken from an interbellic newspaper.” Source: `https://en.wikipedia.org/wiki/File:Halftone,_Gaussian_Blur.jpg`.

Canny edge detection

The white pixels in the image on the right-hand-side represent pixels that likely have an edge in the left-hand-side, using Canny edge detection. Source: `https://en.wikipedia.org/wiki/Edge_detection#/media/File:%C3%84%C3%A4retuvastuse_n%C3%A4ide.png`.

3 $\times$ **3** Gaussian filter

$$\begin{pmatrix} \mathbf{0.0113} & \mathbf{0.0838} & \mathbf{0.0113} \\ \mathbf{0.0838} & \mathbf{0.6193} & \mathbf{0.0838} \\ \mathbf{0.0113} & \mathbf{0.0838} & \mathbf{0.0113} \end{pmatrix}$$

Like a moving average except the center pixel is most important. So, it's a "smarter" blur than a moving average.

vertical deriv Sobel

$$\begin{pmatrix} \mathbf{1} & \mathbf{2} & \mathbf{1} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{-1} & \mathbf{-2} & \mathbf{-1} \end{pmatrix}$$

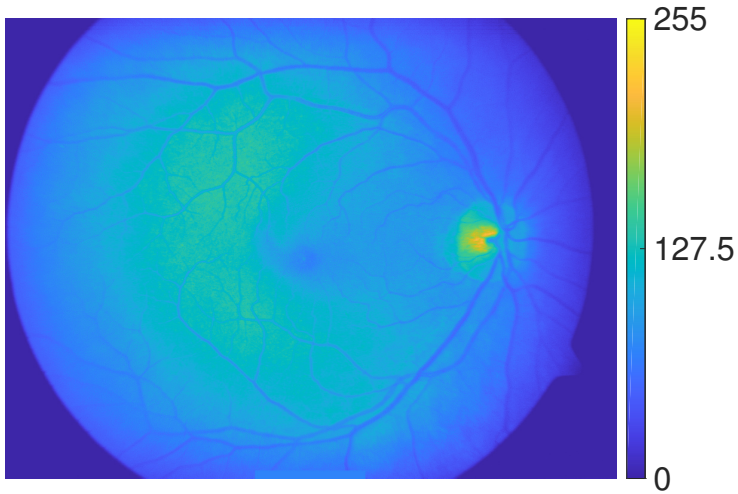
Matches best / has biggest inner product at a horizontal edge.

horizontal deriv Sobel

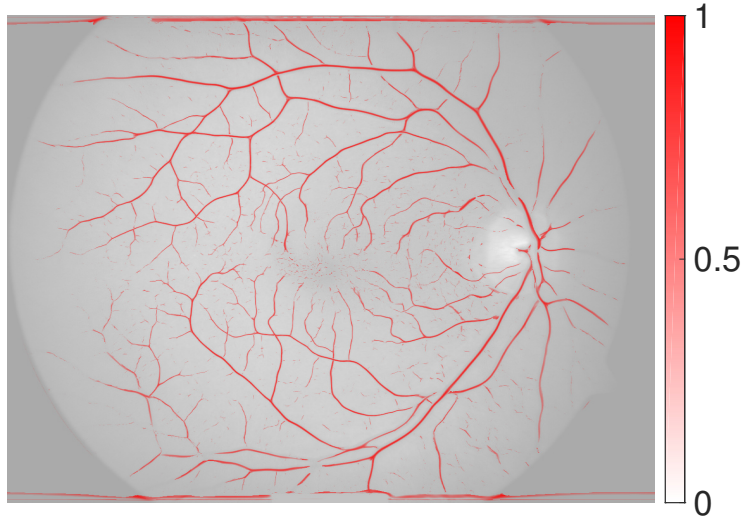
$$\begin{pmatrix} \mathbf{1} & \mathbf{0} & \mathbf{-1} \\ \mathbf{2} & \mathbf{0} & \mathbf{-2} \\ \mathbf{1} & \mathbf{0} & \mathbf{-1} \end{pmatrix}$$

Matches best / has biggest inner product at a vertical edge.

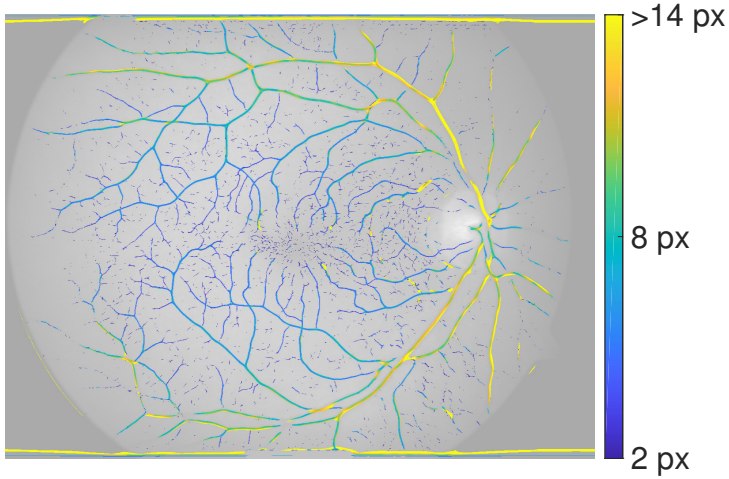
(Go to Matlab)



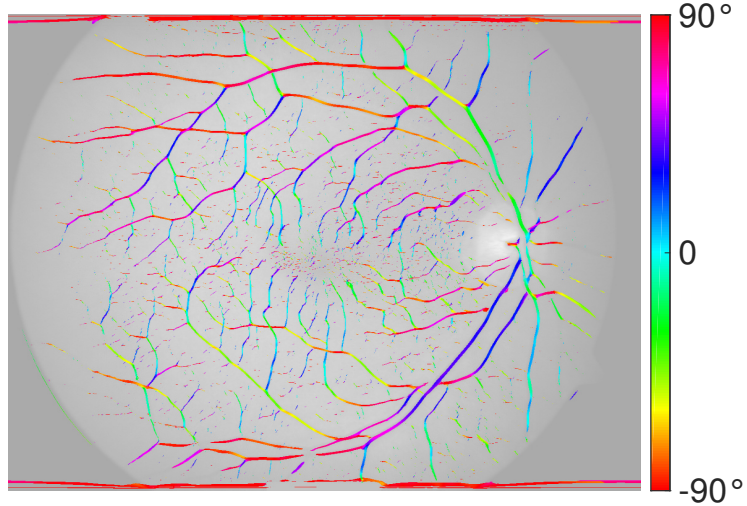
Input (image 7 of VDIS dataset).



Detected ridges.



Predicted ridge widths.



Predicted ridge tangent directions.

Width and orientation measurements of retinal blood vessels yielded by SymFD (Reisenhofer, King 2019). The input image is part of the VDIS dataset from the REVIEW retinal vessel reference database (Al-Diri et al. 2012).

Moving average,

Canny edge detection, and

the retinal vessel extraction

get good to great results with pre-defined convolutional kernels/filters/windows.

Moving average,

Canny edge detection, and

the retinal vessel extraction

get good to great results with pre-defined convolutional kernels/filters/windows.

The idea behind a convolutional neural network (CNN) is to use machine learning to learn the best convolutional kernel for your desired task.

Moving average,

Canny edge detection, and

the retinal vessel extraction

get good to great results with pre-defined convolutional kernels/filters/windows.

The idea behind a convolutional neural network (CNN) is to use machine learning to learn the best convolutional kernel for your desired task.

In more classical image processing, if one convolves an RGB image, one uses the same filter on each layer and outputs another RGB image.

Moving average,

Canny edge detection, and

the retinal vessel extraction

get good to great results with pre-defined convolutional kernels/filters/windows.

The idea behind a convolutional neural network (CNN) is to use machine learning to learn the best convolutional kernel for your desired task.

In more classical image processing, if one convolves an RGB image, one uses the same filter on each layer and outputs another RGB image.

In CNNs, one might convolve each channel with a different filter and then sum the outputs component-wise.

Structure Determination

Known gene sequences $>$ known protein structures $>$ known protein complex structures

Structure Determination

Known gene sequences $>$ known protein structures $>$ known protein complex structures

Structure determination methods:

- X-ray crystallography
  (not all proteins crystallize easily)

Structure Determination

Known gene sequences $>$ known protein structures $>$ known protein complex structures

Structure determination methods:

- X-ray crystallography
  (not all proteins crystallize easily)

- NMR spectroscopy
  (can't handle arbitrarily large molecules)

Structure Determination

Known gene sequences $>$ known protein structures $>$ known protein complex structures

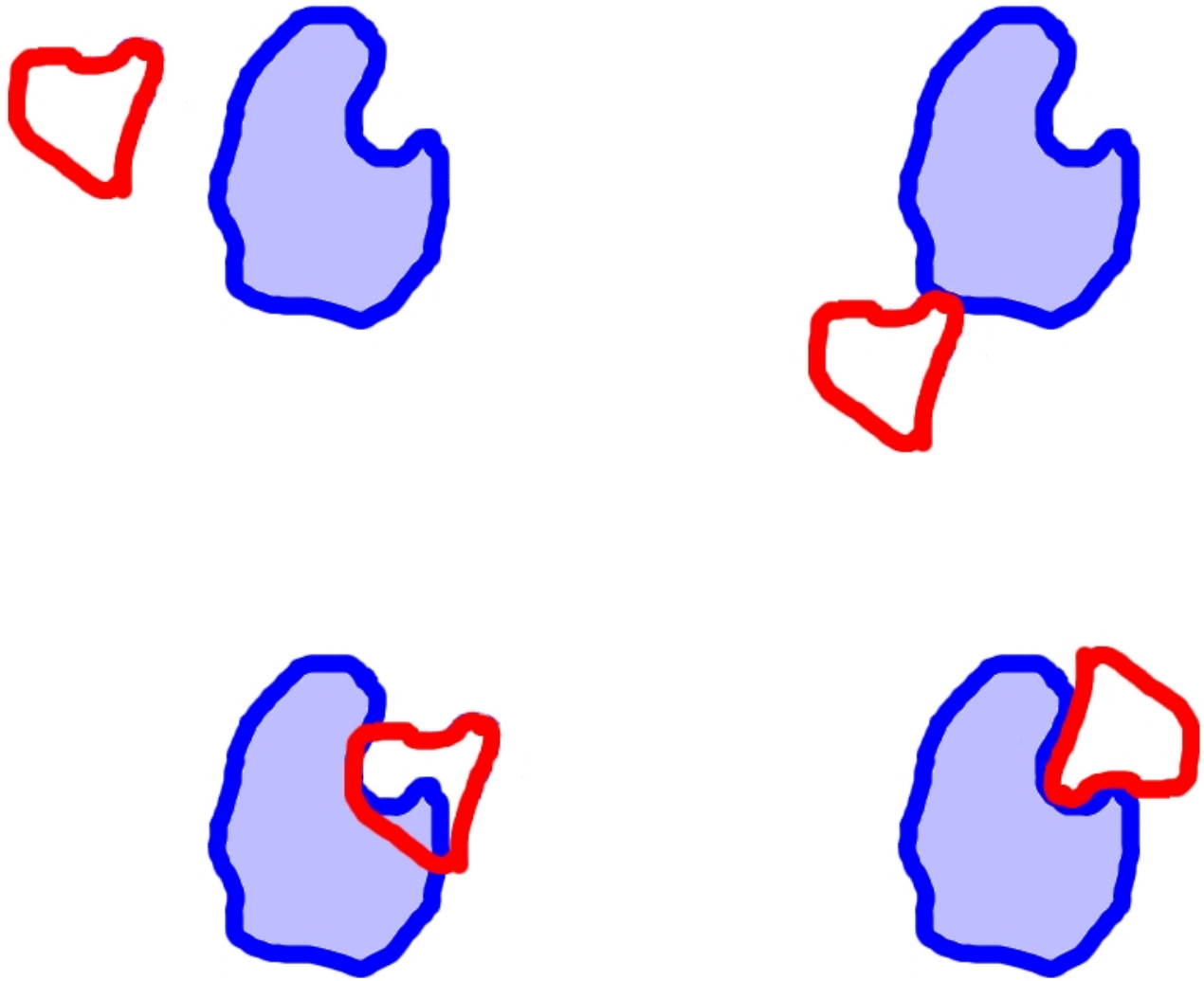
Structure determination methods:

- X-ray crystallography
  (not all proteins crystallize easily)

- NMR spectroscopy
  (can't handle arbitrarily large molecules)

- Protein structure prediction
  (not as accurate but huge breakthrough with Alphafold)

Shape Matching



Fourier Correlation Algorithm

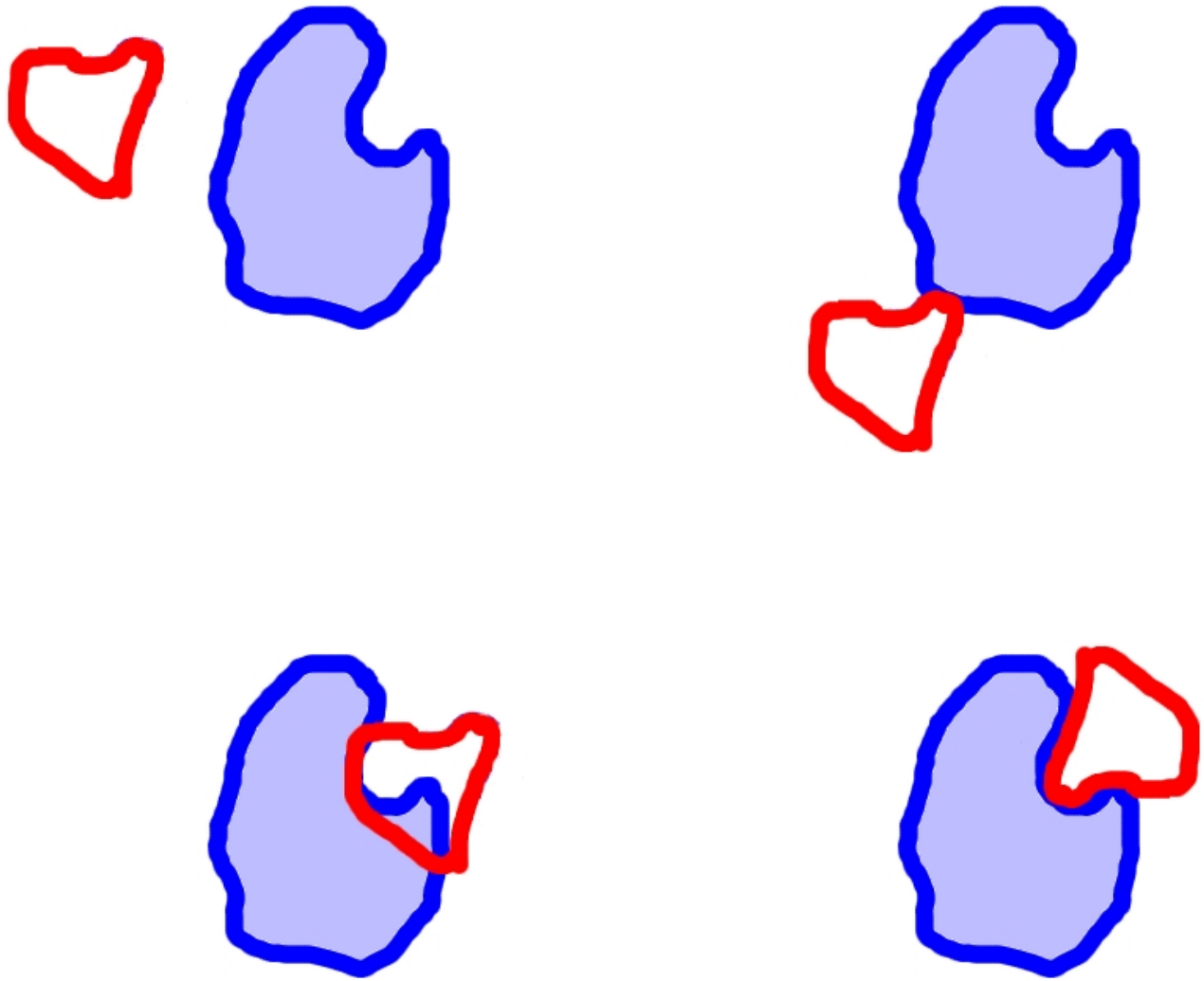
For two molecules A and B, discretize on an N^3 grid as follows:

$$f_A[\ell,m,n] = \begin{cases} 1 & (\ell,m,n) \text{ on } A\text{'s surface} \\ \rho & (\ell,m,n) \text{ in interior of } A \\ 0 & (\ell,m,n) \text{ outside } A \end{cases}$$

$$f_B[\ell,m,n] = \begin{cases} 1 & (\ell,m,n) \text{ on } B\text{'s surface} \\ (\rho & (\ell,m,n) \text{ in interior of } B) \\ 0 & (\ell,m,n) \text{ outside } B \end{cases}$$

Katchalski-Katzir et al. 1992, Gabb et al. 1997

Shape Matching



Correlation

Define the cross-correlation between f_A and f_B as

$$f_A \star f_B[\alpha, \beta, \gamma] = \sum_{\ell=1}^{N} \sum_{m=1}^{N} \sum_{n=1}^{N} f_A[\ell, m, n] f_B[\ell + \alpha, m + \beta, n + \gamma].$$

Correlation

Define the cross-correlation between f_A and f_B as

$$f_A \star f_B[\alpha, \beta, \gamma] = \sum_{\ell=1}^{N} \sum_{m=1}^{N} \sum_{n=1}^{N} f_A[\ell, m, n] f_B[\ell + \alpha, m + \beta, n + \gamma].$$

If ρ is an "appropriate" negative number, this function will roughly be

- large and positive when the molecules are well-docked,
- negative or small when they are overlapping, and
- zero when they aren't touching.

Basic Idea

1. Compute $f_A \star f_B$.
2. Find peaks of $f_A \star f_B$.
3. Rotate f_B by a small θ to obtain new f_B.
4. Repeat Steps 1 – 3.

Basic Idea

1. Compute $f_A \star f_B$.
2. Find peaks of $f_A \star f_B$.
3. Rotate f_B by a small θ to obtain new f_B.
4. Repeat Steps 1 – 3.

This is computationally intensive.

Basic Idea

1. Compute $f_A \star f_B$.
2. Find peaks of $f_A \star f_B$.
3. Rotate f_B by a small θ to obtain new f_B.
4. Repeat Steps 1 – 3.

This is computationally intensive.

$\Rightarrow$ Change of basis (via Fourier) to the rescue! More on change of basis in the next lecture.