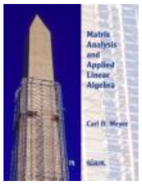
Lecture 12: Projections (class)

Admin:  5.13  3.1–3.3

no HW today

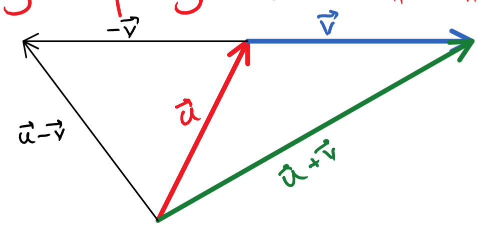
no class next Thursday
exam instead

Recall:

Inner product: $\vec{u}\cdot\vec{v} = \sum_i u_i^* v_i$ ← complex conjugate

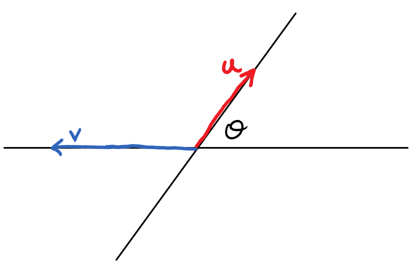
Length: $\|\vec{u}\| = \sqrt{\vec{u}\cdot\vec{u}}$

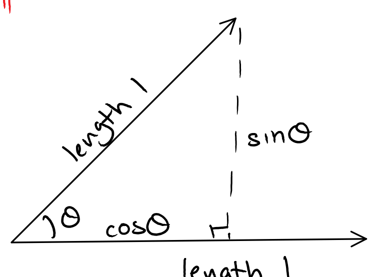
Triangle inequality: $\|\vec{u}+\vec{v}\| \le \|\vec{u}\| + \|\vec{v}\|$



Corollary: $\|u\| \le \|v\| + \|u-v\|$
$\Rightarrow \|u-v\| \geq \|u\| - \|v\|$

Angles: angle between lines Span(u) and Span(v)
$\cos\theta = \dfrac{|\vec{u}\cdot\vec{v}|}{\|u\|\cdot\|v\|}$





Cauchy-Schwarz inequality: $|\vec{u}\cdot\vec{v}| \le \|\vec{u}\|\cdot\|\vec{v}\|$
(with equality iff collinear)

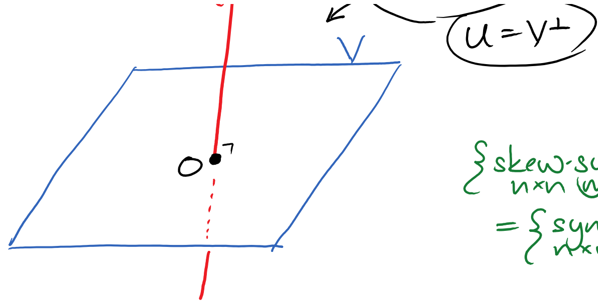
Example: $\sum_{i=1}^{n} u_i = (1,1,\dots,1)\cdot\vec{u}$
$\le \|(1,\dots,1)\|\cdot\|\vec{u}\|$
$= \sqrt{n}\cdot\|\vec{u}\|$

Orthogonal vectors: $\vec{u}\perp\vec{v} \iff \vec{u}\cdot\vec{v} = 0$

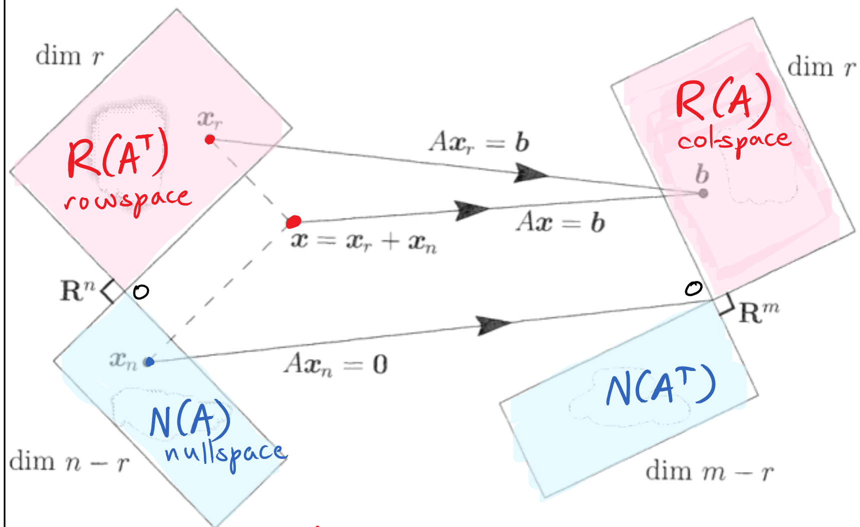
Orthogonal subspaces ≠ orthogonal complements:

$U \perp V$

$U = V^\perp$



$\begin{Bmatrix} \text{skew-symmetric} \\ n\times n \text{ matrices} \end{Bmatrix} = \begin{Bmatrix} \text{symmetric} \\ n\times n \text{ matrices} \end{Bmatrix}^\perp$



$N(A) = R(A^T)^\perp$ $\qquad$ $N(A^T) = R(A)^\perp$

Consequences:

* $\{x \mid Ax = b\}$ is an $(n-r)$-dim affine space.
* Any subspace U can be described either
  - by a basis for U ($\dim U$ vectors), or
  - by a basis for $U^\perp$ ($n - \dim U$ vectors)

Today: Projections and orthogonal bases

PROJECTIONS

Motivation: Principal component analysis (PCA)

High-dimensional data set $\longrightarrow$ fewer dimensions

Example:

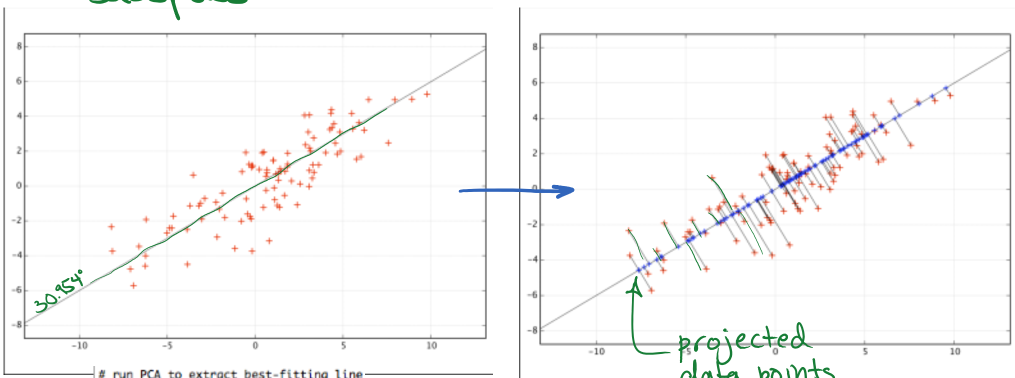
generate noisy data:

```
n = 100; largerstddev = 5; rotationangle = 30;
c = cosd(rotationangle); s = sind(rotationangle);
data = [c,-s; s,c] * [largerstddev,0; 0,1] * randn(2, n);
```

① Find best-fitting low-dim subspace

② Project data to subspace

subspace




```
# run PCA to extract best-fitting line
[coeff,score] = princomp(data');
slope = coeff(2,1) / coeff(1,1);
fitangle = atand(slope)
```

```
projecteddata = coeff(:,1) * (coeff(:,1)' * data);
```

Applications: Machine learning, statistics, data analysis, compression,...

We'll cover PCA when we get to the singular-value decomp.

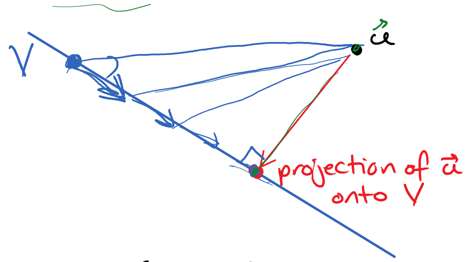
Problem: How can we make all these PROJECTIONS?

Definition: Let V be a subspace of $\mathbb{R}^n$ (or $\mathbb{C}^n$) (not a finite field!)

The orthogonal projection onto V maps

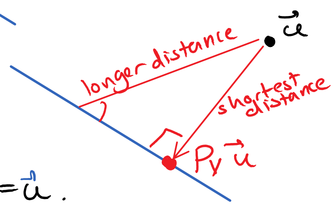
any point $u \in \mathbb{R}^n$ $\longmapsto$ the closest point in V

$\mathrm{Tr}\, A \equiv \sum_i A_{ii}$



3a. $\mathrm{Rank}(P_V) = \dim V$

Observe: 1. $(P_V\vec{u} - \vec{u}) \perp V$.



3b. $\mathrm{Tr}(P_V) = \dim V$

2. If $\vec{u} \in V$, $P_V\vec{u} = \vec{u}$.
3. $P_V^2 = P_V$

why? b/c for any $\vec{u}$, $P_V\vec{v} \in V$

$\overset{②}{\Rightarrow} P_V(P_V v) = P_V \vec{v}$

$\Rightarrow P^2 = P$

4. Projections are linear transformations
$(P_V(\alpha\vec{u}) = \alpha P_V\vec{u}$, $P_V(\vec{u}+\vec{w}) = P_V\vec{u} + P_V\vec{w})$

5. $I - P_V = P_{V^\perp}$ projection onto $V^\perp$
where $V^\perp = \{$all vectors $\perp$ to $V\}$, the "orthogonal complement" of V. Recall $\dim(V^\perp) = n - \dim(V)$.

$P_V + P_{V^\perp} = I$
$\downarrow \mathrm{Tr} \quad \downarrow \mathrm{Tr} \quad \downarrow \mathrm{Tr}$
$\dim V \qquad\qquad\qquad n$

$n - \dim V = \dim V^\perp$

→ We'll prove these properties later!

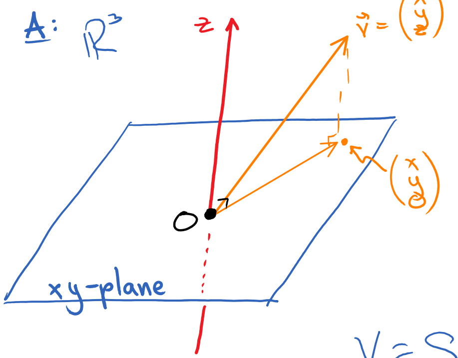
We'll prove these properties later!

n-dim, - dim

Intuition: Projection to a coordinate subspace

Q: What is the projection from $\mathbb{R}^3$ to the xy-plane?

A: $\mathbb{R}^3$



The matrix $P_V = \begin{pmatrix} 1 & & 0 \\ & 1 & \\ 0 & & 0 \end{pmatrix}$ discards the z-coordinate of any point.

$$P_V \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} = \begin{pmatrix} u_1 \\ u_2 \\ 0 \end{pmatrix}$$

More generally

$$V = \text{Span}\{\vec{e}_1, \ldots, \vec{e}_k\}$$

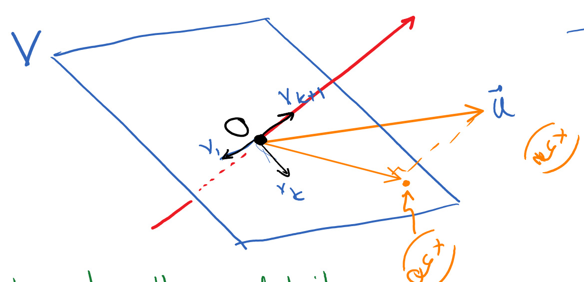
$$P_V = \begin{array}{c} k\{ \\ n-k\{ \end{array} \left(\begin{array}{ccc|ccc} 1 & & & & & \\ & \ddots & & & 0 & \\ & & 1 & & & \\ \hline & & & 0 & & \\ & 0 & & & \ddots & \\ & & & & & 0 \end{array}\right) \qquad P_V \vec{u} = \begin{pmatrix} u_1 \\ u_2 \\ \vdots \\ u_k \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

in $\mathbb{R}^n$.

More generally:

Projecting onto an arbitrary subspace V $\longleftrightarrow$

- Change basis to orthogonal coordinate system starting with basis for V
- Discard the coordinates $\perp$ to V



Let $\vec{v}_1, \ldots, \vec{v}_k$ be an orthonormal basis for V ($k = \dim V$)

Extend this $v_{k+1}, \ldots, \vec{v}_n$ o.n. basis for $V^\perp$

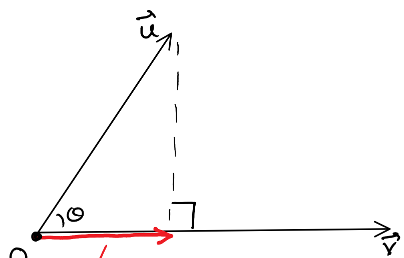
coordinates of $P_V \vec{u}$ are $\vec{v}_1 \cdot \vec{u}, \vec{v}_2 \cdot \vec{u}, \ldots, \vec{v}_k \cdot \vec{u}, \cancel{v_{k+1} \cdot u}, \ldots, \cancel{v_n \cdot u}$

$$\Rightarrow \boxed{P_V \vec{u} = \sum_{j=1}^{k} (\vec{v}_j \cdot \vec{u}) \vec{v}_j}$$ if $v_1, \ldots, v_k$ is an orthonormal basis for V

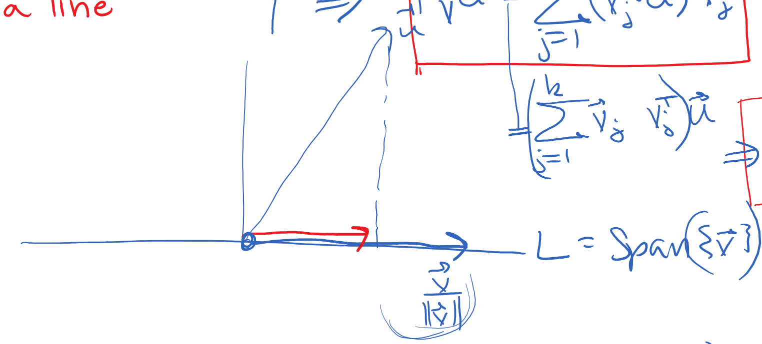
$$= \Big(\sum_{j=1}^{k} \vec{v}_j \vec{v}_j^T\Big) \vec{u} \Rightarrow \boxed{P_V = \sum_{j=1}^{k} \vec{v}_j \vec{v}_j^T}$$

We'll next explain this in detail...

Example: Projection onto a line



projection of $\vec{u}$ onto the line $\text{Span}(\vec{v})$ $= (\vec{v} \cdot \vec{u}) \vec{v}$ if $\|\vec{v}\| = 1$



$$P_L \vec{u} = \left(\frac{\vec{v}}{\|v\|} \cdot \vec{u}\right) \frac{\vec{v}}{\|\vec{v}\|} = \frac{1}{\|v\|^2} \vec{v}\,(v \cdot u)$$

Lemma: The projection of $\vec{u}$ onto the line $\text{Span}(\vec{v})$ is $= \frac{1}{\|v\|^2} \vec{v}(\vec{v}^T u)$

$\mathrm{Span}(\vec{v})$ $\qquad P_L \vec{u} = \left(\frac{\vec{v}}{\|\vec{v}\|}\cdot \vec{u}\right)\frac{\vec{v}}{\|\vec{v}\|}$

$$= \frac{1}{\|v\|^2}\vec{v}(\vec{v}^T u)$$

$$= \left(\frac{1}{\|v\|^2}\vec{v}\vec{v}^T\right)\vec{u}$$

Lemma: The projection of $\vec{u}$ onto the line $\mathrm{Span}(\vec{v})$ is

$(\star)$ $\qquad (\vec{v}\cdot\vec{u})\vec{v}$ if $\|\vec{v}\|=1$ $\qquad (\star)$

$$\frac{\vec{v}\vec{v}^T}{\|\vec{v}\|^2}\vec{u} = \frac{(\vec{v}\cdot\vec{u})\vec{v}}{\|\vec{v}\|^2} \quad \text{in general}$$

The matrix that projects onto $\mathrm{Span}(\vec{v})$ is

$$\frac{\vec{v}\,\vec{v}^T}{\|\vec{v}\|^2}$$

← an $n\times n$ matrix if $\vec{v}$ is $n\times 1$

– since $(\vec{v}\vec{v}^T)\vec{u} = \vec{v}(\vec{v}^T\vec{u}) = \vec{v}(\underbrace{\vec{v}\cdot\vec{u}}_{=\|\vec{u}\|\cos\theta})$ ✓

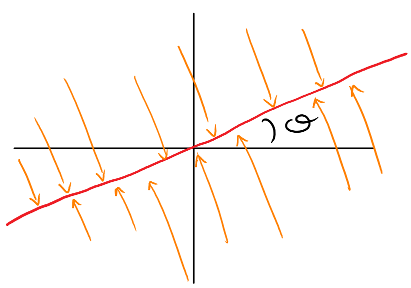
Example: $\mathbb{R}^3$:

$$\vec{e}_x = \begin{pmatrix}1\\0\\0\end{pmatrix},\ \vec{e}_y = \begin{pmatrix}0\\1\\0\end{pmatrix},\ \vec{e}_z = \begin{pmatrix}0\\0\\1\end{pmatrix}$$

$$\vec{e}_y\vec{e}_y^T = \begin{pmatrix}0&0&0\\0&1&0\\0&0&0\end{pmatrix}$$

$$\vec{e}_y\vec{e}_y^T\begin{pmatrix}\alpha\\\beta\\\gamma\end{pmatrix} = \begin{pmatrix}0\\\beta\\0\end{pmatrix}$$

Example: Projection onto the line at angle θ

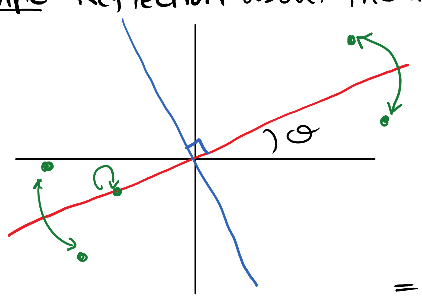


$$\vec{v} = \begin{pmatrix}\cos\theta\\\sin\theta\end{pmatrix}$$

$$\begin{pmatrix}\cos\theta\\\sin\theta\end{pmatrix}(\cos\theta, \sin\theta)$$

$$= \underbrace{\begin{pmatrix}\cos^2\theta & \cos\theta\sin\theta\\ \sin\theta\cos\theta & \sin^2\theta\end{pmatrix}}_{=P_\theta}$$

Example: Reflection about the line at angle θ



$$P_\theta - (I - P_\theta)$$

P_θ: leaves vector in red space unchanged

$(I-P_\theta)$: puts a minus sign on vectors in the orthogonal blue line

$$= 2P_\theta - I = \begin{pmatrix}2c^2-1 & 2cs\\ 2cs & 2s^2-1\end{pmatrix}$$

$$= \begin{pmatrix}\cos 2\theta & \sin 2\theta\\ \sin 2\theta & -\cos 2\theta\end{pmatrix}$$

Example: Projection onto $(1,1,\dots,1)$

$$P = \frac{1}{\sqrt{n}}\begin{pmatrix}1\\\vdots\\1\end{pmatrix}\cdot\frac{1}{\sqrt{n}}(1\ 1\ \cdots\ 1) = \frac{1}{n}\begin{pmatrix}1 & 1 & \cdots & 1\\ \vdots & \vdots & \ddots & \vdots\end{pmatrix}$$

Example: Projection onto $(1,1,\dots,1)$

$$P = \frac{1}{\sqrt{n}}\begin{pmatrix}1\\1\\\vdots\\1\end{pmatrix}\cdot\frac{1}{\sqrt{n}}\begin{pmatrix}1 & 1 & \cdots & 1\end{pmatrix} = \frac{1}{n}\begin{pmatrix}1 & 1 & \cdots & 1\\ 1 & 1 & \cdots & 1\\ \vdots & \vdots & \ddots & \vdots\\ 1 & 1 & \cdots & 1\end{pmatrix}$$

Observe: $P v = \left(\text{avg. of } v\text{'s coords}\right)\cdot\begin{pmatrix}1\\\vdots\\1\end{pmatrix}$.

Example: Projection to $(1,1,\dots,1)^{\perp} = R\left(\begin{pmatrix}-2 & 1 & & & 1\\ 1 & -2 & 1 & & \\ & 1 & -2 & 1 & \\ & & 1 & \ddots & \\ 1 & & & & \end{pmatrix}\right)$.

$= I - P$

Exercise: What is the projection of
$$\vec{u} = (2,-2,3)$$
onto the line
$$L = \left\{ (x,y,z) \;\middle|\; \begin{matrix} x + 2y + 3z = 0 \\ x - y + 2z = 0 \end{matrix} \right\} \quad ?$$

Answer:

① First find a basis for L.

$$L = N\left(\begin{pmatrix}1 & 2 & 3\\ 1 & -1 & 2\end{pmatrix}\right) = N\left(\begin{pmatrix}1 & 0 & 7/3\\ 0 & 3 & 1\end{pmatrix}\right)$$

$$= \text{Span}\left(\begin{pmatrix}-7/3\\ -1/3\\ 1\end{pmatrix}\right) = \text{Span}\left(\underbrace{\frac{1}{\sqrt{59}}\begin{pmatrix}7\\ 1\\ -3\end{pmatrix}}_{=\,v}\right)$$

② Normalize it:

③ Project

$$P_{\vec{v}}\,\vec{u} = \vec{v}\,\vec{v}^{T}\vec{u}$$
$$= \vec{v}\,(v\cdot u)$$
$$= \frac{1}{59}\begin{pmatrix}7\\1\\-3\end{pmatrix}\cdot\left(\begin{pmatrix}7\\1\\-3\end{pmatrix}\cdot\begin{pmatrix}2\\-2\\3\end{pmatrix}\right)$$
$$= \boxed{\frac{3}{59}\begin{pmatrix}7\\1\\-3\end{pmatrix}}$$

④ Check the answer:

$\frac{3}{59}(7,1,-3) \in L$: ✓ since it is a multiple of $\vec{v}$,

and yes $x+2y+3z=0$
$x-y+2z=0$.

$(\vec{u} - P_L\vec{u}) \perp L$:

$$(2,-2,3) - \frac{3}{59}(7,1,-3) = \frac{1}{59}(97,-121,186)$$
$$(97,-121,186)\cdot(7,1,-3) = 0 \;\checkmark$$

Exercise: What is the projection of
$$\vec{u} = (4,0,3)$$
onto the line
$$L' = \left\{ (x,y,z) \;\middle|\; \begin{matrix} x + 2y + 3z = 6 \\ x - y + 2z = 0 \end{matrix} \right\} \quad ?$$

Answer:

$\{ \ldots \mid x - y + 2z = 0 \}$.

Answer:

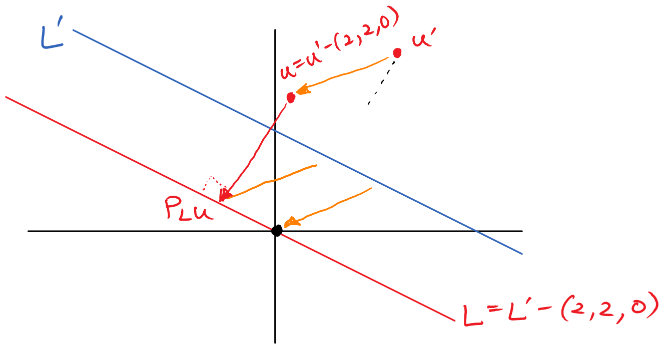
① Find a constructive formulation for L':

General principle: To work with an affine subspace,

A. translate everything so it goes through 0

B. work there

C. translate back!



Since $L' - (2,2,0) = L$ (from above)

and $u' - (2,2,0) = u$,

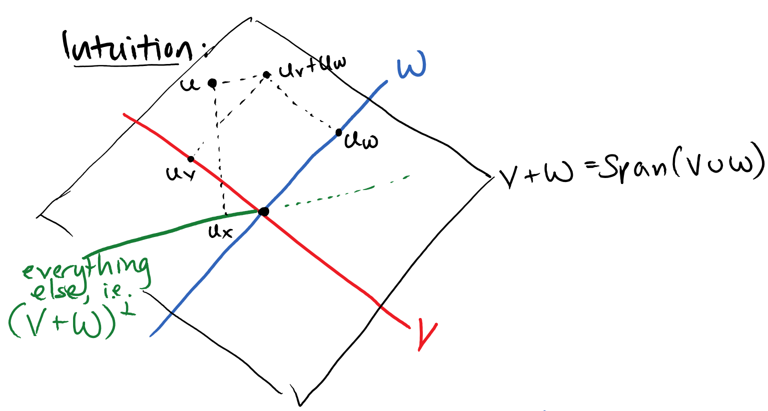
the projection of u' onto L' is

$$\boxed{(2,2,0) + \frac{3}{59}(7,1,-3) = \frac{1}{59}(139,121,-9)}$$

Problem: How can we construct the projector onto higher-dim. subspaces?

Key property: If $V \perp W$,

$$P_V + P_W = P_{V+W}$$



any vector $\vec{u}$ can be expanded as

$$\vec{u} = \vec{u}_V + \vec{u}_W + \vec{u}_X$$

where $\underset{P_V \vec{u}}{\vec{u}_V} \in V$, $\underset{P_W \vec{u}}{\vec{u}_W} \in W$, $\underset{P_X \vec{u}}{\vec{u}_X} \in X$

$P_{V+W}\vec{u} = \vec{u}_V + \vec{u}_W$

Examples & counterexamples:

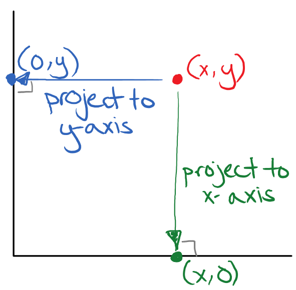
$P_{e_1} =$

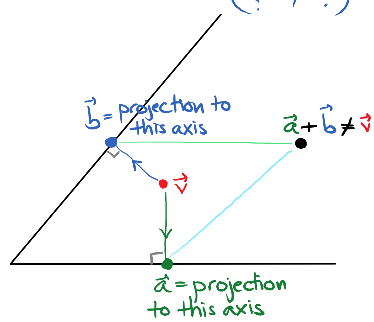
$P_{e_2} =$

$P_{e_1} + P_{e_2} =$

This does not work if V is not $\perp$ to W:

$$\begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} + \begin{pmatrix} \cos^2\theta & \cos\theta\sin\theta \\ \cos\theta\sin\theta & \sin^2\theta \end{pmatrix} = \text{? not a projection } (?^2 \neq ?)$$





It only works because $e_1 \perp e_2$!

$(c, s) = (\cos\theta, \sin\theta)$

$$\begin{pmatrix} c \\ s \end{pmatrix}\begin{pmatrix} c & s \end{pmatrix} + \begin{pmatrix} s \\ -c \end{pmatrix}\begin{pmatrix} s & -c \end{pmatrix}$$

$$= \begin{pmatrix} c^2 & cs \\ cs & s^2 \end{pmatrix} + \begin{pmatrix} s^2 & -cs \\ -cs & c^2 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \checkmark$$

$\Rightarrow$ To project onto V, it is enough to have a basis of pairwise orthogonal vectors for V...

FACT: For a subspace $V \subseteq \mathbb{R}^n$ with orthonormal basis $\{\vec{v}_1, \dots, \vec{v}_k\}$, orthogonal projection onto V

$$P_{\mathcal{U}} = \boxed{\sum_{j=1}^{k} \underbrace{\vec{v}_j \vec{v}_j^T}_{n\times n \text{ matrix}}}$$

Proof: $P_V = P_{\text{span}\{v_1,\dots,v_{k-1}\}} + \underset{\underset{v_k v_k^T}{\parallel}}{P_{\text{span}\{v_k\}}}$

Claim: For subspaces $V, V^\perp \subseteq \mathbb{R}^n$,

$$\boxed{P_V + P_{V^\perp} = I}$$

Proof:

Example: This can save a lot of time!

Let $A = \begin{pmatrix} -2 & 1 & & & 1 \\ 1 & -2 & 1 & & \\ & 1 & -2 & 1 & \\ & & & \ddots & \\ 1 & & & & \end{pmatrix}_{n\times n}$, and $\vec{v} \in \mathbb{R}^n$.

$A = A^T$

Compute $P_{R(A)}\vec{v}$.

Answer 1: Recall that Rank$(A) = n-1$,

$N(A) = \text{Span}\left(\begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}\right)$, $\begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix} = \vec{1}$

basis for $R(A)$: $\{\vec{e}_1 - \vec{e}_2, \vec{e}_2 - \vec{e}_3, \vec{e}_3 - \vec{e}_4, \ldots, \vec{e}_{n-1} - \vec{e}_n\}$

$R(A) = \{\vec{v} \mid \sum_j v_j = 0\} = N((1 \; 1 \; \cdots \; 1))$

compute an orthonormal basis

$P_{R(A)} = \sum_{j=1}^{n-1} u_j u_j^T$ $\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix}$

Better answer:

Remember $P_V = I - P_{V^\perp}$

$\Downarrow$

$P_{R(A)} = I - P_{N(A)}$

$= I - \dfrac{\vec{1}\vec{1}^T}{\|\vec{1}\|^2} = I - \dfrac{\vec{1}\vec{1}^T}{n}$

In general, if $\dim V$ is close to n, then using $P_V = I - P_{V^\perp}$ will often simplify your calculations.

Recipe: How to make projections in Matlab

Problem: Given vectors $\vec{v}_1, \vec{v}_2, \ldots, \vec{v}_k$, how do we project onto their span?

1. $A = \begin{pmatrix} | & | & & | \\ \vec{v}_1 & \vec{v}_2 & \cdots & \vec{v}_k \\ | & | & & | \end{pmatrix}_{n\times k}$ $V = R(A)$
2. $B = \text{orth}(A)$ in Matlab
3. projection matrix $= BB^T$

in Python:
scipy.linalg.orth()
numpy.linalg.qr()

B is $n \times \text{rank}(A)$ matrix

Why? $B = \begin{pmatrix} | & | & & | \\ u_1 & u_2 & \cdots & u_d \\ | & | & & | \end{pmatrix}$ $d \le k$

orthonormal

from above $P_V = \sum_j u_j u_j^T$

$= \sum_{j=1}^{d} \vec{u}_j \vec{e}_j^T$

$$BB^T = \sum_{j,k=1} u_j \underbrace{e_j^T e_k}_{} u_k^T = \sum_j u_j u_j^T$$

$$e_j \cdot e_k = \begin{cases} 1 & \text{if } j=k \\ 0 & j \neq k \end{cases}$$

Exercise:
Compute the projection of $\vec{e}_1$ onto a random 100-dimensional subspace V of $\mathbb{R}^{10000}$.

Also: What is the expected squared length of the projection?

Answer:

```
n = 10000;
d = 100;
e1 = zeros(n,1); e1(1) = 1;
A = randn(n, d);
B = orth(A);
```

~~projection = B * B' * e1;~~ ← don't do this neither in Matlab nor in hand calculations

```
projection = B * (B' * e1);
```

```
>> P = B * B';
>> whos P
  Name          Size              Bytes
  P         10000x10000         800000000
```

needs n^2 memory

```
>> whos B
  Name          Size              Bytes
  B          10000x100             8000000
```

needs $n \times d$ memory

```
>> format long
>> norm(projection)^2

ans =

   0.009994012962916
```

$\approx .01 = \frac{d}{n}$ Why?

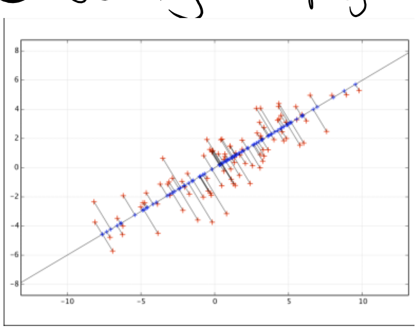
Gram-Schmidt orthogonalization

(see the other notes)

More examples of projections

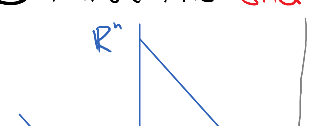
① Find the shortest $\vec{x}$ solving $A\vec{x} = \vec{b}$

② Working with projected data



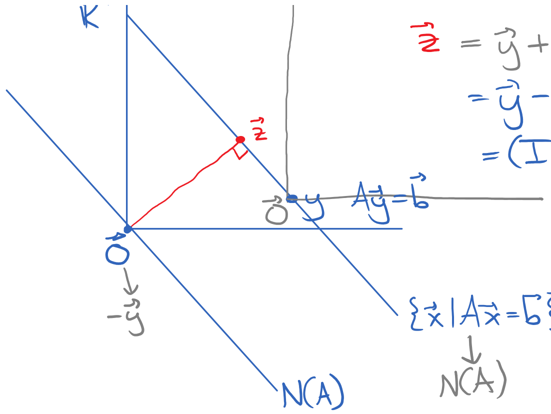
vs.

① Find the shortest $\vec{x}$ solving $A\vec{x} = \vec{b}$

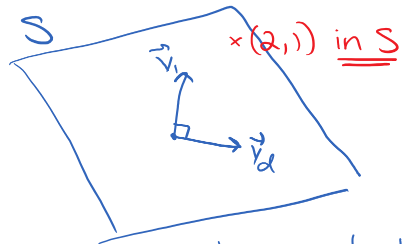


$\vec{z} = \vec{y} + P_{N(A)}(-\vec{y})$

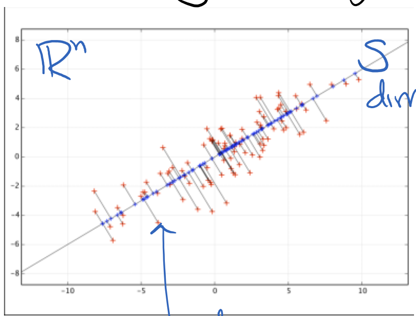
$= \vec{y} - P_{N(A)}\vec{y}$



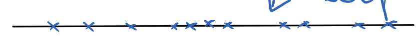
$$\begin{aligned}\vec z &= \vec y + P_{N(A)}(-\vec y)\\ &= \vec y - P_{N(A)}\vec y\\ &= (I - P_{N(A)})\vec y\\ &= P_{N(A)^\perp}\vec y\\ &= P_{R(A^T)}\vec y\end{aligned}$$



② Working with projected data



vs.



keep only the d coords within S

$$P_S\vec x = \sum_{j=1}^{d} \vec v_j(\vec v_j^T\vec x)$$

if $\vec v_1, \dots, \vec v_d \in \mathbb{R}^n$ are an orthonormal basis for S

$$Q_S\vec x = \begin{pmatrix} \vec v_1 \cdot \vec x \\ v_2 \cdot x \\ \vdots \\ \vec v_d \cdot \vec x \end{pmatrix} = \sum_{j=1}^{d} \vec e_j(\vec v_j^T\vec x) = \left(\sum_{j=1}^{d} \vec e_j \vec v_j^T\right)\vec x$$

$$\left(\sum_{j=1}^{d} \vec e_j \vec v_j^T\right) = \begin{pmatrix} \text{---} & \vec v_1^T & \text{---} \\ \text{---} & \vec v_2^T & \text{---} \\ & \vdots & \end{pmatrix}$$

Example: Image completion

Original image

Erased 40% of the coordinates

Solved to minimize $\|\vec x_{Haar}\|^2$ with constrained coordinates

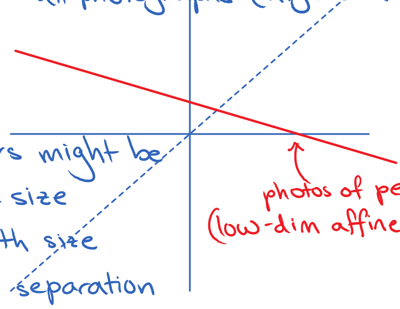
Experiment: How many degrees of freedom are there in a face?

Hypothesis: Not many! (?)

Idea:

$\vec{\text{photograph}} \in \mathbb{R}^{\text{number of pixels}}$

all photographs (high-dim space)

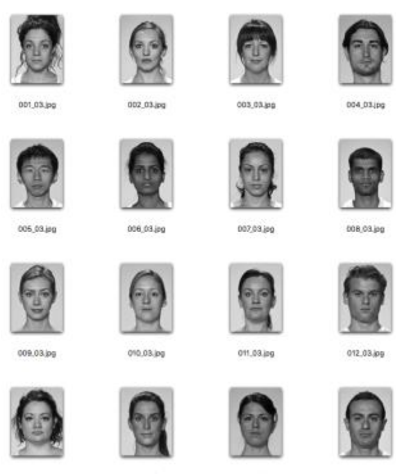


e.g., parameters might be
- nose size
- mouth size
- eye separation
- ⋮

photos of people (low-dim affine subspace?)

Approach:

① Scrape 100+ photos from http://faceresearch.org/
Resize and convert to grayscale (for simplicity)



② Load images into Matlab as vectors

```
files = {'001_03.jpg', '002_03.jpg', '003_03.jpg', '004_03.jpg', '005_03.jpg', '006_03.jpg', '007_03.jpg',
'008_03.jpg', '009_03.jpg', '010_03.jpg', '011_03.jpg', '012_03.jpg', '013_03.jpg', '014_03.jpg', '016_03.jpg',
'017_03.jpg', '018_03.jpg', '019_03.jpg', '020_03.jpg', '021_03.jpg', '022_03.jpg', '024_03.jpg', '025_03.jpg',
'026_03.jpg', '027_03.jpg', '029_03.jpg', '030_03.jpg', '031_03.jpg', '032_03.jpg', '033_03.jpg', '034_03.jpg',
'036_03.jpg', '037_03.jpg', '038_03.jpg', '039_03.jpg', '041_03.jpg', '042_03.jpg', '043_03.jpg', '044_03.jpg',
'045_03.jpg', '061_03.jpg', '062_03.jpg', '063_03.jpg', '064_03.jpg', '066_03.jpg', '067_03.jpg', '068_03.jpg',
'069_03.jpg', '070_03.jpg', '081_03.jpg', '082_03.jpg', '083_03.jpg', '086_03.jpg', '087_03.jpg', '088_03.jpg',
'090_03.jpg', '091_03.jpg', '092_03.jpg', '094_03.jpg', '096_03.jpg', '097_03.jpg', '099_03.jpg', '100_03.jpg',
'101_03.jpg', '102_03.jpg', '103_03.jpg', '104_03.jpg', '105_03.jpg', '107_03.jpg', '108_03.jpg', '111_03.jpg',
'112_03.jpg', '113_03.jpg', '114_03.jpg', '115_03.jpg', '117_03.jpg', '118_03.jpg', '119_03.jpg', '120_03.jpg',
'121_03.jpg', '122_03.jpg', '123_03.jpg', '124_03.jpg', '125_03.jpg', '126_03.jpg', '127_03.jpg', '128_03.jpg',
'129_03.jpg', '130_03.jpg', '131_03.jpg', '132_03.jpg', '134_03.jpg', '135_03.jpg', '136_03.jpg', '137_03.jpg',
'138_03.jpg', '139_03.jpg', '140_03.jpg', '141_03.jpg', '142_03.jpg', '143_03.jpg', '144_03.jpg', '172_03.jpg',
'173_03.jpg'};

n = length(files);                      % n = 103
imagesize = size(imread(files{1}))      % 100 x 75
m = imagesize(1) * imagesize(2)         % m = 7500
```

flattens 100×75 array into 7500×1 column vector

```
n = length(files);                  % n = 103
imagesize = size(imread(files{1}))  % 100 x 75
m = imagesize(1) * imagesize(2)     % m = 7500
```

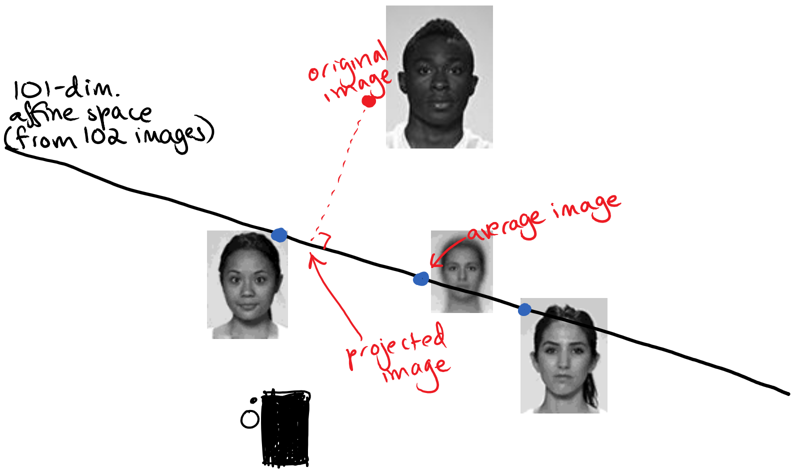
flattens 100×75 array into 7500×1 column vector

```
images = zeros(m,n);
for i = 1:n
  image = imread(files{i});
```

3 Remove a random image

```
% pick a random image to remove
randomindex = randi(n);
randomimage = images(:,randomindex);  % select and remove that image
images(:,[randomindex]) = [];
n = n - 1;                            %  and decrement n
imwrite(reshape(randomimage, imagesize(1), imagesize(2))/255, 'randomimage.png');
```

4 Project it onto the affine span of the others




```
% recenter about the mean image
meanimage = sum(images,2) / n;  % sum(A,2) adds up the columns of A (i.e., the second dimension)
for i = 1:n
  images(:,i) = images(:,i) - meanimage;
end

O = orth(images);   % size(O) is 7500x102
projectedimage = meanimage + O * (O' * (randomimage-meanimage));
imwrite(reshape(projectedimage, imagesize(1), imagesize(2))/255, 'projectedimage.png');
```

Idea: If faces form a low-dimensional affine subspace, then the projection should be close to the original image.

Result:

Original image | Projected image