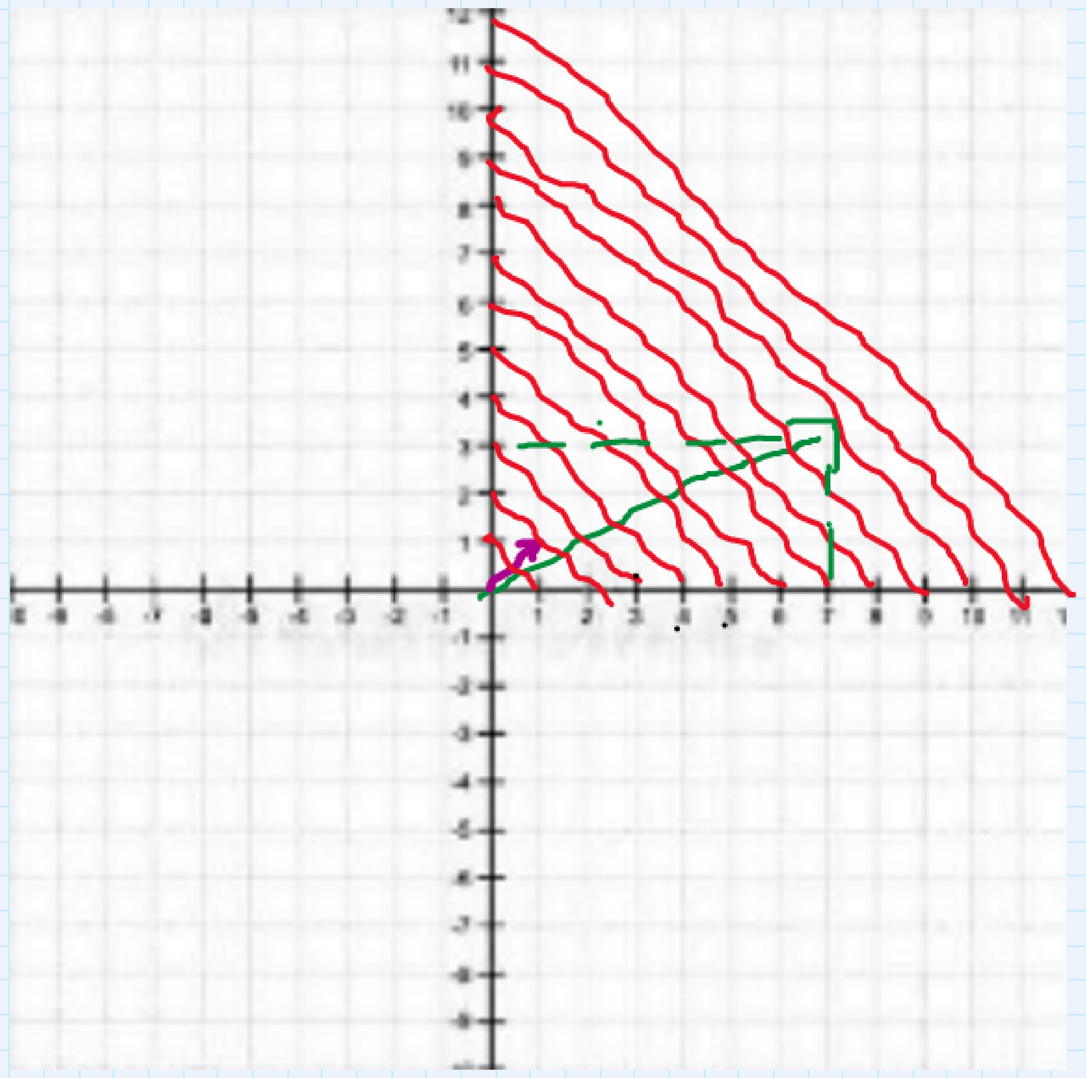
Convector/Dot Product Example

Monday, March 21, 2022 11:48 PM

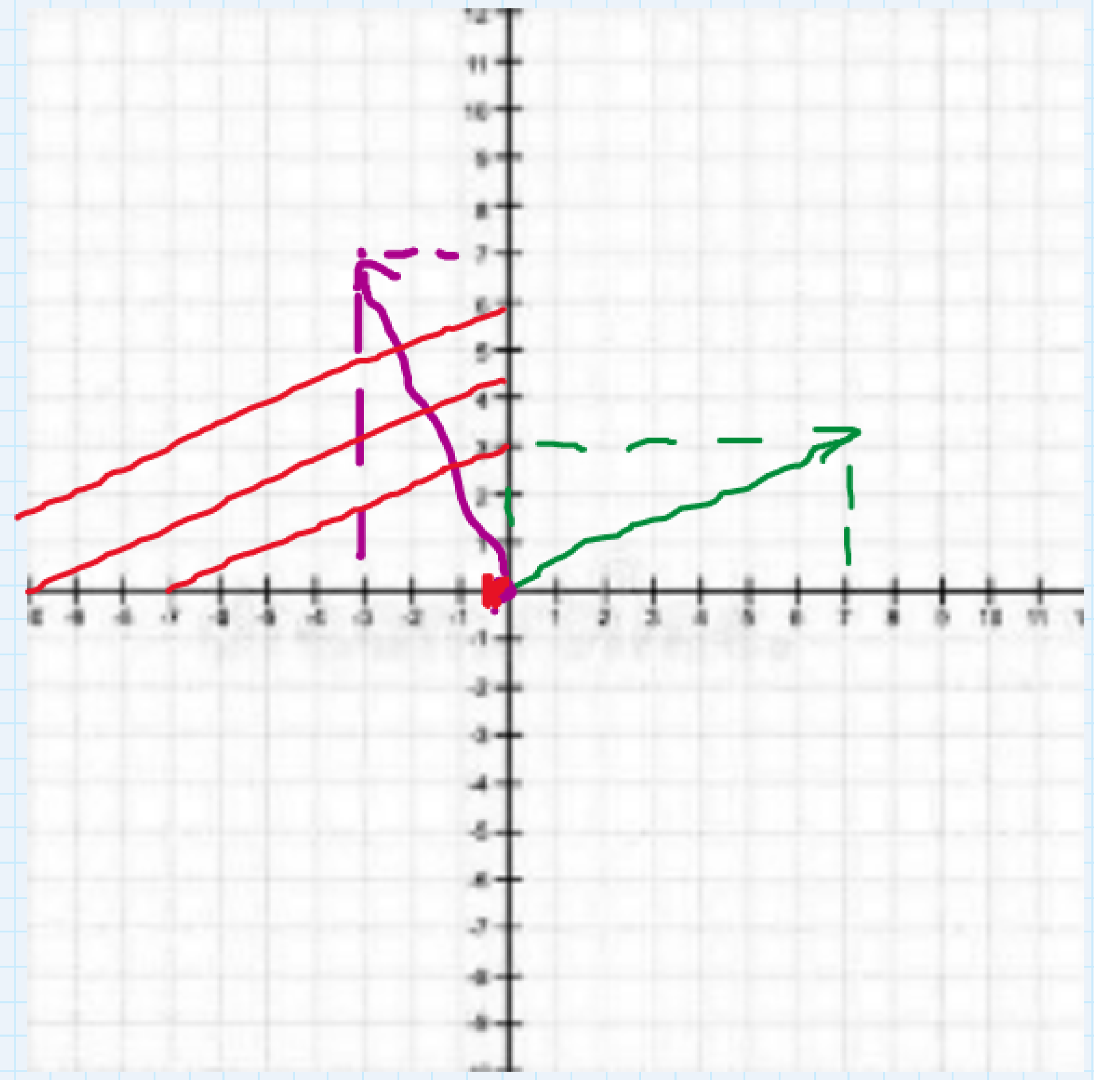


$$x = \begin{bmatrix} 7 \\ 3 \end{bmatrix}$$

$$\tilde{x} = \begin{bmatrix} 1 & 1 \end{bmatrix}$$

$$\tilde{x} \cdot x = \begin{bmatrix} 1 & 1 \end{bmatrix} \begin{bmatrix} 7 \\ 3 \end{bmatrix}$$

$$= 10$$

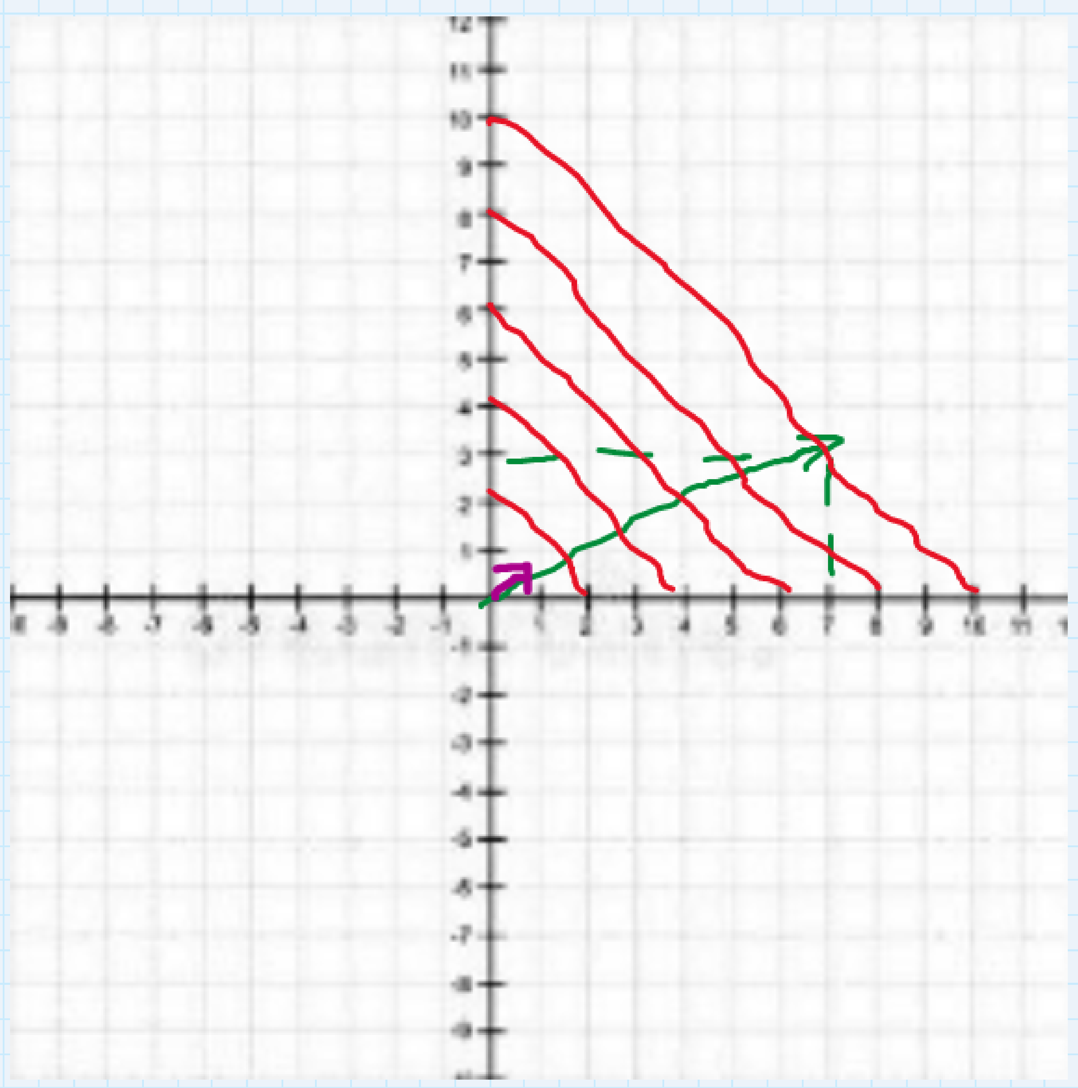


$$x = \begin{bmatrix} 7 \\ 3 \end{bmatrix}$$

$$\tilde{x} = \begin{bmatrix} -3 & 7 \end{bmatrix}$$

$$\tilde{x} \cdot x = \begin{bmatrix} -3 & 7 \end{bmatrix} \begin{bmatrix} 7 \\ 3 \end{bmatrix}$$

$$= 0$$



$$x = \begin{bmatrix} 7 \\ 3 \end{bmatrix}$$

$$\tilde{x} = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \end{bmatrix}$$

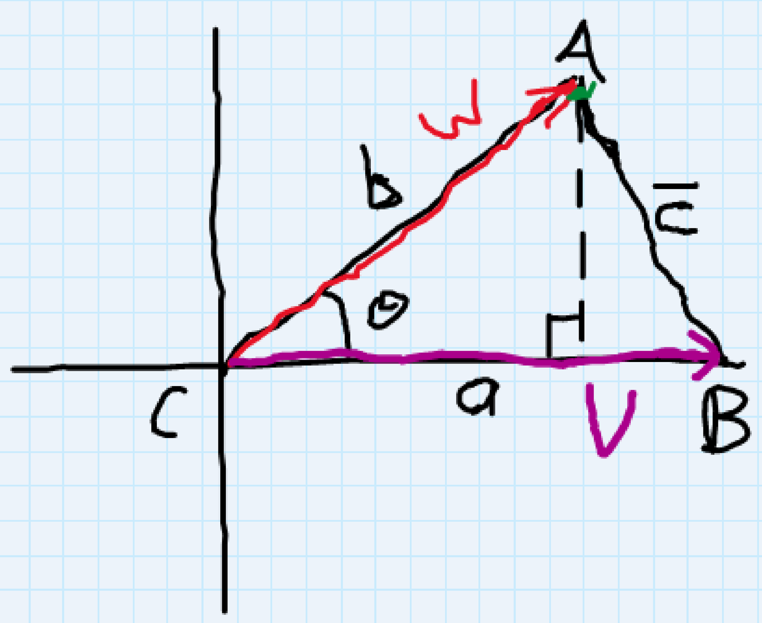
$$\tilde{x} \cdot x = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} \end{bmatrix} \begin{bmatrix} 7 \\ 3 \end{bmatrix}$$

$$= 5$$

$$\cos(\theta) = \frac{\langle a, b \rangle}{\|a\| \, \|b\|}$$

$$= \frac{1}{\|a\|} \frac{1}{\|b\|} \langle a, b \rangle$$

$$= \left\langle \frac{a}{\|a\|}, \frac{b}{\|b\|} \right\rangle$$



$$A = (x, y) = (r\cos\theta, r\sin\theta)$$

$$= (b\cos\theta, b\sin\theta)$$

$$\bar{c} = \sqrt{(b\cos\theta - a)^2 + (b\sin\theta)^2}$$

$$\bar{c}^2 = (b\cos\theta - a)^2 + (b\sin\theta)^2$$

$$\bar{c}^2 = a^2 + b^2 + 2ab\cos\theta$$

$$\|v - w\|^2 = \|v\|^2 + \|w\|^2 - 2\|v\| \, \|w\| \cos(\theta)$$

$$\|v - w\|^2 = \langle v - w, v - w \rangle = \langle v, v - w \rangle - \langle w, v - w \rangle$$

$$= \langle v, v \rangle - \langle v, w \rangle - \langle w, v \rangle + \langle w, w \rangle$$

bilinearity

$$= \langle v, v \rangle + \langle w, w \rangle - 2\langle v, w \rangle$$

$$= \|v\|^2 + \|w\|^2 - 2\langle v, w \rangle$$

But

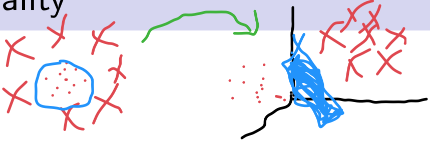
$$\|v - w\|^2 = \|v\|^2 + \|w\|^2 - 2\|v\| \, \|w\| \cos(\theta)$$

so

$$\langle v, w \rangle = \|v\| \, \|w\| \cos(\theta)$$

Analytic Geometry 2/2

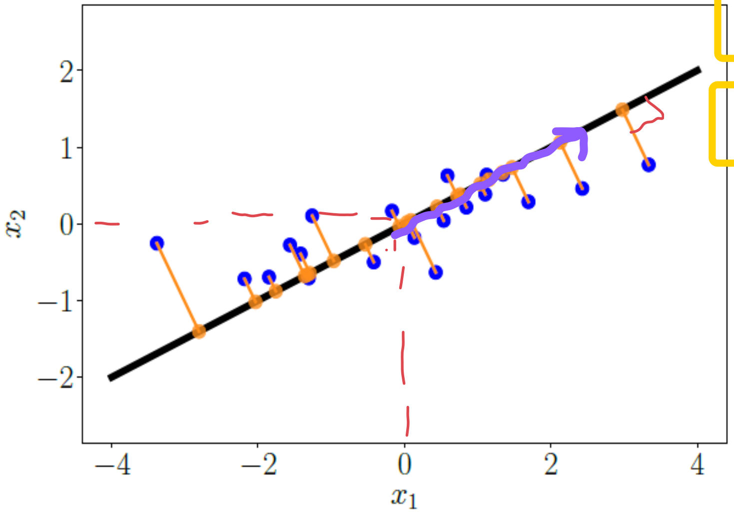
Changing our Dimensionality



Many powerful machine learning approaches involve dimensionality changes

- In the case of a SVM we work in a dimensionality typically higher than the original data.
- Many visualizations require projecting down into a more manageable dimensionality.
  - Ideally we want this mapping to preserve as much *information* from the original data as possible.
  - There are often different aspects of the *information* we might prioritize.
    - ★ For example with PCA we try and preserve the data variance as much as possible

Orthogonal Projections from $\mathbb{R}^2$ to $\mathbb{R}$



Source: M.P. Deisenroth *et al*, Mathematics for Machine Learning (First Edition)

Projections

Projection

Let V be a vector space and $U \subseteq V$ a subspace of V.

- A linear mapping $\pi : V \rightarrow U$ is called a *projection* if $\pi^2 = \pi \circ \pi = \pi$

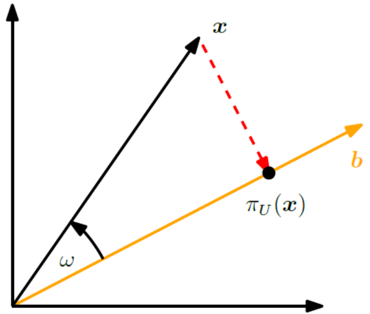
Since linear mappings can be expressed by transformation matrices we also have that

- $\mathbf{P}_\pi$ is a projection matrix if $\mathbf{P}_\pi^2 = \mathbf{P}_\pi$.

Projecting onto One-Dimensional Subspaces (Line)

We want to project $\mathbf{x} \in \mathbb{R}^n$ onto a line (one dimensional subspace) through the origin and defined by the basis vector $\mathbf{b} \in \mathbb{R}^n$ where $span[\mathbf{b}] = U \subseteq \mathbb{R}^n$.

- Specifically when we project $\mathbf{x} \in \mathbb{R}^n$ onto U we wish to obtain $\pi_U(\mathbf{x}) \in U$ that is the closest to $\mathbf{x}$.



(a) Projection of $\boldsymbol{x} \in \mathbb{R}^2$ onto a subspace U with basis vector $\boldsymbol{b}$.

Source: M.P. Deisenroth *et al.* Mathematics for Machine Learning (First Edition)

Projecting onto One-Dimensional Subspaces (Line)

It is worth being precise in our requirements on π_u:

- The projection $\pi_U(\mathbf{x})$ is closest to $\mathbf{x}$ when
  - $\|\mathbf{x} - \pi_U(\mathbf{x})\|$ is minimal.

  This occurs when $\pi_U(\mathbf{x}) - \mathbf{x}$ is orthogonal to U.
  - This orthogonality means that $\langle \pi_U(\mathbf{x}) - \mathbf{x}, \mathbf{b} \rangle = 0$
- Since $\pi_U(\mathbf{x})$ maps $\mathbf{x}$ into U there exists a $\lambda \in \mathbb{R}$ such that $\pi_U(\mathbf{x}) = \lambda \mathbf{b}$

Building the Projection Matrix

We are going to build $\mathbf{P}_\pi$ that maps any $\mathbf{x} \in \mathbb{R}^n$ onto U. This is broken into three steps

- Determine λ coordinate
- Determine $\pi_U(\mathbf{x}) \in U$
- Finally construct $\mathbf{P}_\pi$.

Building the Projection Matrix

$\langle \pi_u(x) - x, b \rangle = 0$

$\Rightarrow -1 \langle x - \pi_u(x), b \rangle = 0$

$\Rightarrow \langle x - \pi_u(x), b \rangle = 0$

Finding the coordinate λ

- From the orthogonality condition we have that

$$0 = \langle \mathbf{x} - \pi_u(\mathbf{x}), \mathbf{b} \rangle \quad (1)$$

$$= \langle \mathbf{x} - \lambda \mathbf{b}, \mathbf{b} \rangle \quad (2)$$

$$= \langle \mathbf{x}, \mathbf{b} \rangle - \lambda \langle \mathbf{b}, \mathbf{b} \rangle \text{ from the bilinearity if } \langle \cdot, \cdot \rangle \quad (3)$$

We can now rearrange to obtain

$$\lambda = \frac{\langle \mathbf{x}, \mathbf{b} \rangle}{\langle \mathbf{b}, \mathbf{b} \rangle} \quad (4)$$

$$= \frac{\langle \mathbf{b}, \mathbf{x} \rangle}{\|\mathbf{b}\|^2} \quad (5)$$

$\langle b, b \rangle > 0$ for $b \neq 0$ and b spans U which is 1Dim so b can't $= 0$

$\|b\|^2 = \sqrt{\langle b, b \rangle}^2$

$= \langle b, b \rangle$

induced Norm

Building the Projection Matrix

Finding the coordinate λ

- If we use our current context and choose $\langle \cdot, \cdot \rangle$ to be the dot product. Using the dot product we can obtain λ as

$$\lambda = \frac{\mathbf{b}^T \mathbf{x}}{\mathbf{b}^T \mathbf{b}} = \frac{\mathbf{b}^T \mathbf{x}}{\|\mathbf{b}\|^2} \tag{6}$$

If $\|\mathbf{b}\| = 1$, then $\lambda = \mathbf{b}^T \mathbf{x}$.

b ⇒ Normal

Building the Projection Matrix

Finding the projection point $\pi_U(\mathbf{x}) \in U$

- Since $\pi_U(\mathbf{x}) = \lambda \mathbf{b}$ it immediately follows that

$$\pi_U(\mathbf{x}) = \lambda \mathbf{b} = \frac{\langle \mathbf{x}, \mathbf{b} \rangle}{\|\mathbf{b}\|^2}\mathbf{b} = \frac{\mathbf{b}^T\mathbf{x}}{\|\mathbf{b}\|^2}\mathbf{b} \tag{7}$$

- What can we say about $\|\pi_U(\mathbf{x})\|$?

$$\|\pi_U(\mathbf{x})\| = \|\lambda \mathbf{b}\| = |\lambda|\|\mathbf{b}\|. \tag{8}$$

If the dot product is used we can tie in our earlier angle definition

$$\frac{|b^T x|}{\|b\|\|x\|} = \cos(\omega)$$

$$|b^T x| = \|b\|\|x\| |\cos(\omega)|$$

$$\|\pi_U(\mathbf{x})\| = \frac{|\mathbf{b}^T\mathbf{x}|}{\|\mathbf{b}\|^2}\|\mathbf{b}\| \tag{9}$$

$$= |cos\omega|\|\mathbf{x}\|\|\mathbf{b}\|\frac{\|\mathbf{b}\|}{\|\mathbf{b}\|^2} \tag{10}$$

$$= |cos\omega|\|\mathbf{x}\| \tag{11}$$

where ω is the angle between $\mathbf{x}$ and $\mathbf{b}$.

Building the Projection Matrix

Finding the projection matrix $\mathbf{P}_\pi$

- Firstly since the projection is a linear mapping and our domain and co-domain are finite dimensional vector spaces there has to exist a matrix $\mathbf{P}_\pi$, such that $\pi_U(\mathbf{x}) = \mathbf{P}_\pi \mathbf{x}$
- Using the dot product we can make the following observation

$$\pi_U(\mathbf{x}) = \lambda \mathbf{b} = \mathbf{b}\lambda \tag{12}$$

$$= \mathbf{b}\frac{\mathbf{b}^T\mathbf{x}}{\|\mathbf{b}\|^2} \tag{13}$$

$$= \frac{\mathbf{b}\mathbf{b}^T}{\|\mathbf{b}\|^2}\mathbf{x} \tag{14}$$

$A(BC) = (AB)C$

Associativity of mat mul

Recall that $\mathbf{b}$ is a $n \times 1$ matrix and $\mathbf{b}^T$ is a $1 \times n$ matrix, therefore $\mathbf{b}\mathbf{b}^T$ is a $n \times n$ matrix (and is also symmetric).

$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}\begin{bmatrix} x_1 & x_2 \end{bmatrix} = \begin{bmatrix} x_1x_1 & x_1x_2 \\ x_1x_2 & x_2x_2 \end{bmatrix}$

- It follows that

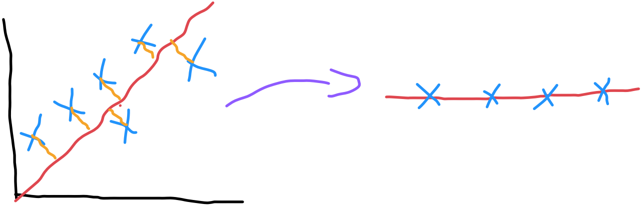
$P_\pi X = \pi_U(x) \Rightarrow$

$$\mathbf{P}_\pi = \frac{\mathbf{b}\mathbf{b}^T}{\|\mathbf{b}\|^2} \tag{15}$$

Important Projection Remark

The projection we have been working with produces a vector in $\mathbb{R}^n$, $\pi_U(\mathbf{x}) \in \mathbb{R}^n$.

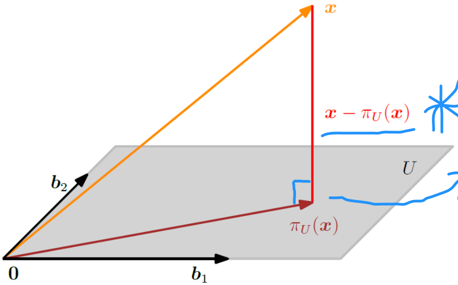
- However, we no longer require n coordinates to represent the projection, but only a single one if we want to express it with respect to the basis vector $\mathbf{b}$ that spans the subspace $U : \lambda$.



Projection onto General Subspaces

We now want to generalize from projecting onto a 1-dimensional subspace (a line) to a arbitrary lower dimensional subspace $U \subset \mathbb{R}^n$ with $dim(U) = m \geq 1$.

- Here is an illustration for a projection onto a two-dimensional subspace of $\mathbb{R}^3$



$x - \Pi_u(x)$ is orthogonal to u

Source: M.P. Deisenroth *et al*, Mathematics for Machine Learning (First Edition)

Projection onto General Subspaces

dim(U) = m

Assume that $(\mathbf{b}_1, \ldots, \mathbf{b}_m)$ is an ordered basis of U

- We know that for any $\mathbf{x} \in \mathbb{R}^n$ that $\pi_U(\mathbf{x}) \in U$, this means that there are λ_js in $\mathbb{R}$ such that

$$\pi_U(\mathbf{x}) = \sum_{i=1}^{m} \lambda_i \mathbf{b}_i \tag{16}$$

- A three step procedure will be used again.

Projection onto General Subspaces

Find the coordinates $\lambda_1, \ldots, \lambda_m$ of the projection, such that the linear combination

$$\pi_U(\mathbf{x}) = \sum_{i=1}^{m} \lambda_i \mathbf{b}_i = \mathbf{B}\boldsymbol{\lambda}, \tag{17}$$

$$\mathbf{B} = [\mathbf{b}_1, \ldots, \mathbf{b}_m] \in \mathbb{R}^{n \times m},\ \boldsymbol{\lambda} = [\lambda_1, \ldots, \lambda_m]^T \in \mathbb{R}^m \tag{18}$$

where $\pi_U(\mathbf{x})$ is the closest point in U to $\mathbf{x}$.

- This occurs if $\pi_U(\mathbf{x}) - \mathbf{x}$ is orthogonal to U.
- Which implies that $\pi_U(\mathbf{x}) - \mathbf{x}$ must be orthogonal to all the basis vectors of U.

Projection onto General Subspaces

In order to ensure that $\pi_U(\mathbf{x}) - \mathbf{x}$ is orthogonal to all the basis vectors of U, we must satisfy m simultaneous conditions. Specifically, using the dot product as the inner product we end up with

$$\langle \mathbf{b}_1, \mathbf{x} - \pi_U(\mathbf{x}) \rangle = \mathbf{b}_1^T(\mathbf{x} - \pi_U(\mathbf{x})) = 0 \tag{19}$$

$$\vdots$$

$$\langle \mathbf{b}_m, \mathbf{x} - \pi_U(\mathbf{x}) \rangle = \mathbf{b}_m^T(\mathbf{x} - \pi_U(\mathbf{x})) = 0 \tag{20}$$

Using equation (17) this system can be more compactly written as

$$\mathbf{b}_1^T(\mathbf{x} - \mathbf{B}\boldsymbol{\lambda}) = 0 \qquad \pi_u(x) = B\lambda \tag{21}$$

$$\vdots$$

$$\mathbf{b}_m^T(\mathbf{x} - \mathbf{B}\boldsymbol{\lambda}) = 0 \tag{22}$$

Projection onto General Subspaces

From which we obtain a homogeneous linear equation system

$m \times n$

$(m \times n)(n \times 1) = (m \times 1)$

$$\begin{bmatrix} \mathbf{b}_1^T \\ \vdots \\ \mathbf{b}_m^T \end{bmatrix} \begin{bmatrix} \mathbf{x} - \mathbf{B}\boldsymbol{\lambda} \end{bmatrix} = \mathbf{0} \tag{23}$$

$n \times 1$

$$\iff \mathbf{B}^T(\mathbf{x} - \mathbf{B}\boldsymbol{\lambda}) = \mathbf{0} \tag{24}$$

$$\iff \mathbf{B}^T\mathbf{B}\boldsymbol{\lambda} = \mathbf{B}^T\mathbf{x} \tag{25}$$

- Equation (25) is called the *normal equation*.
- Note that $\mathbf{B}^T\mathbf{B} \in \mathbb{R}^{m \times m}$ is invertible because $\mathbf{b}_1, \ldots \mathbf{b}_m$ are linearly independent. This means we can solve for $\boldsymbol{\lambda}$

$$\boldsymbol{\lambda} = (\mathbf{B}^T\mathbf{B})^{-1}\mathbf{B}^T\mathbf{x} \tag{26}$$

Projection onto General Subspaces

Finding the projection point $\pi_U(\mathbf{x}) \in U$:

- We have from equation (17) and equation (26)

$$\pi_U(\mathbf{x}) = \sum_{i=1}^{m} \lambda_i \mathbf{b}_i = \mathbf{B}\boldsymbol{\lambda} \tag{27}$$

$$= \mathbf{B}(\mathbf{B}^T\mathbf{B})^{-1}\mathbf{B}^T\mathbf{x} \tag{28}$$

Find the projection matrix $\mathbf{P}_\pi$:

- From equation (27) we can directly see that the matrix that satisfies

$$\mathbf{P}_\pi \mathbf{x} = \pi_U(\mathbf{x}) \tag{29}$$

  must be

$$\mathbf{P}_\pi = \mathbf{B}(\mathbf{B}^T\mathbf{B})^{-1}\mathbf{B}^T \tag{30}$$

Important Projection Remark

The projection we have been working with produces a vector in $\mathbb{R}^n$, $\pi_U(\mathbf{x}) \in \mathbb{R}^n$.

- However, we no longer require n coordinates to represent the projection, but only a $dim(U) = m$ coordinates with respect to the basis vectors $\mathbf{b}_1, \ldots, \mathbf{b}_m$ that spans the subspace $U : \lambda_1, \ldots, \lambda_m$.

Projection and a Orthonormal Basis

If we are projecting using an **orthonormal basis** for U can be simplified as follows

$$
\begin{aligned}
\pi_U(\mathbf{x}) &= \mathbf{B}\boldsymbol{\lambda} && (31)\\
&= \mathbf{B}(\mathbf{B}^T\mathbf{B})^{-1}\mathbf{B}^T\mathbf{x} && (32)\\
&= \mathbf{B}\mathbf{I}^{-1}\mathbf{B}^T\mathbf{x} \qquad \text{since } \mathbf{B}^T\mathbf{B} = \mathbf{I} && (33)\\
&= \mathbf{B}\mathbf{B}^T\mathbf{x} && (34)
\end{aligned}
$$

This is a much faster computation as it avoids the costly inverse operation.

Projection and Inhomogeneous Systems

Recall that if

$$\mathbf{Ax} = \mathbf{b} \tag{35}$$

has no solution it means that $\mathbf{b}$ is not in the span of $\mathbf{A}$.

- But we can try and find the $\mathbf{x} \in span(\mathbf{A})$ that is closest to $\mathbf{b}$ (as an approximate solution)
- Specifically, we try solve

$$\mathbf{A}^T\mathbf{Ax} = \mathbf{A}^T\mathbf{b} \tag{36}$$

$x = (A^TA)^{-1}A^Tb$

where the $\mathbf{x}$, if it exists, is the *least-squares solution*.

Gram-Schmidt Orthogonalization

Working with a orthonormal basis comes with many advantages. A question worth asking is:

- If I have a ordered basis $(\mathbf{b}_1, \ldots, \mathbf{b}_n)$ of a n-dimensional vector space V. Is there a equivalent orthonormal basis? Furthermore, can we build ~~from~~ a orthonormal basis from the given basis?
  - The answer is yes, there always is a orthonormal basis $(\mathbf{u}_1, \ldots, \mathbf{u}_n)$ such that $span[\mathbf{b}_1, \ldots, \mathbf{b}_n] = span[\mathbf{u}_1, \ldots, \mathbf{u}_n]$
    - ⋆ See Liesen, Jörg, and Mehrmann, Volker. 2015. Linear Algebra. Springer
  - The answer to the second question also yes, via the *Gram-Schmidt orthogonalization method*.

Gram-Schmidt Orthogonalization

The Gram-Schmidt orthogonalization method constructs an orthogonal basis $(\mathbf{u}_1, \ldots, \mathbf{u}_n)$ from any basis $(\mathbf{b}_1, \ldots, \mathbf{b}_n)$ of V as follows:

$$u_k \perp \{u_1 \ldots u_{k-1}\} \Longrightarrow u_k \perp span[u_1 .. u_{k-1}]$$

$$\mathbf{u}_1 := \mathbf{b}_1 \tag{37}$$

$$\mathbf{u}_k := \mathbf{b}_k - \pi_{span[\mathbf{u}_1,\ldots,\mathbf{u}_{k-1}]}(\mathbf{b}_k), \; k = 2, \ldots, n \tag{38}$$

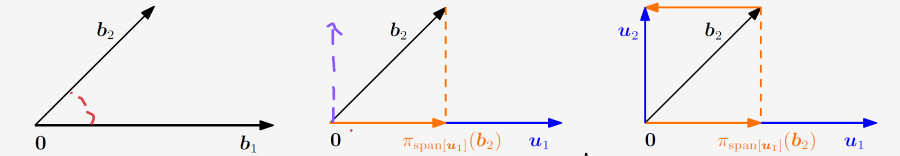
Let us deep dive into equation (38)

- the kth basis vector $\mathbf{b}_k$ is projected onto the subspace spanned by the first $k-1$ constructed orthogonal vectors $\mathbf{u}_1, \ldots, \mathbf{u}_{k-1}$
- This projection is then subtracted from $\mathbf{b}_k$ and yields a vector $\mathbf{u}_k$
  - that is orthogonal to the $k-1$-dimensional subspace spanned by $\mathbf{u}_1, \ldots, \mathbf{u}_{k-1}$
- Repeating this procedure for all n basis vectors yeilds orthogonal basis.
  - If we go step further and rather use $\hat{\mathbf{u}}_k = \frac{\mathbf{u}_k}{\|\mathbf{u}_k\|}$ we have a orthonormal basis!

$\|\hat{u}_k\| = 1$

Gram-Schmidt Orthogonalization

Visual illustration:



(a) Original non-orthogonal basis vectors $\boldsymbol{b}_1, \boldsymbol{b}_2$.

(b) First new basis vector $\boldsymbol{u}_1 = \boldsymbol{b}_1$ and projection of $\boldsymbol{b}_2$ onto the subspace spanned by $\boldsymbol{u}_1$.

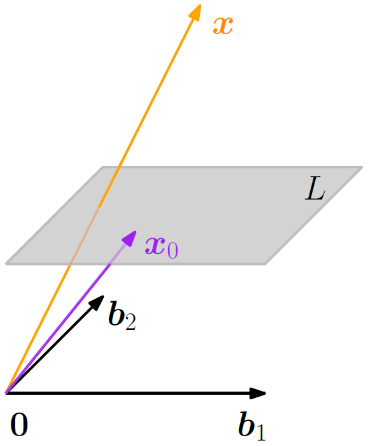
(c) Orthogonal basis vectors $\boldsymbol{u}_1$ and $\boldsymbol{u}_2 = \boldsymbol{b}_2 - \pi_{\text{span}[\boldsymbol{u}_1]}(\boldsymbol{b}_2)$.

Source: M.P. Deisenroth *et al*, Mathematics for Machine Learning (First Edition)

Projection onto Affine Subspaces

Up until now we have been restricted to projecting onto a vector subspace.

- Can we extend our approach to allow for an **affine subspace**?

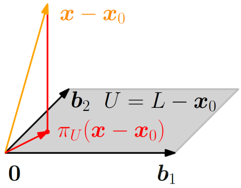


(a) Setting.

Projection onto Affine Subspaces

Let $L = \mathbf{x}_0 + U$ be our affine space where $\mathbf{x}_0$ is our support vector and U is the vector subspace of V spanned by the basis vectors $\mathbf{b}_1, \mathbf{b}_2$.

- We want to find the orthogonal projection $\pi_L(\mathbf{x})$ of $\mathbf{x} \in V$ onto L
- In order to do this we transform our problem to an easier context to perform the projection and then undo the original transformation. Specifically:
  - Consider the point $\mathbf{x} - \mathbf{x}_0$ and $L - \mathbf{x}_0 = U$
  - We already know how to project $\mathbf{x} - \mathbf{x}_0$ onto U. Specifically using $\pi_U(\mathbf{x} - \mathbf{x}_0)$



(b) Reduce problem to projection π_U onto vector subspace.

Source: M.P. Deisenroth *et al*, Mathematics for Machine Learning (First Edition)

Projection onto Affine Subspaces

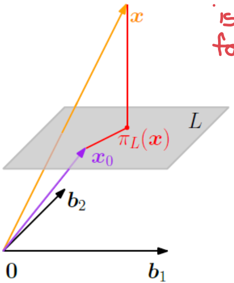
All that is left to do is to undo our initial transformation. Specifically,

- We only need to add the support vector back, leading to

$$\pi_L(\mathbf{x}) = \mathbf{x}_0 + \pi_U(\mathbf{x} - \mathbf{x}_0) \tag{39}$$

$$= x_0 - \pi_U(x_0) + \pi_U(x)$$

is more efficient for lots of data



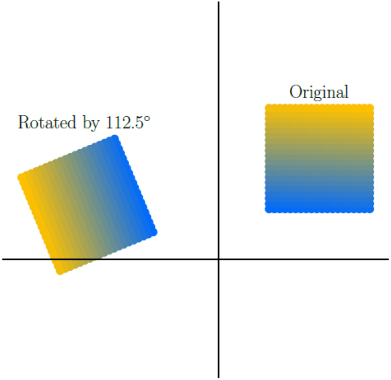
(c) Add support point back in to get affine projection π_L.

Source: M.P. Deisenroth *et al*, Mathematics for Machine Learning (First Edition)

Rotations

A rotation is an automorphism $\mathbf{R} : V \to V$, in our context V is typically a Euclidean vector space.

- We typically discuss the rotation in term of angle θ around an axis
- The angle θ refers to the counterclockwise movement.



Source: M.P. Deisenroth *et al*, Mathematics for Machine Learning (First Edition)

Rotations in $\mathbb{R}^2$

If we have the standard coordinate system in $\mathbb{R}^2$, with the corresponding basis $(\mathbf{e}_1, \mathbf{e}_2)$.

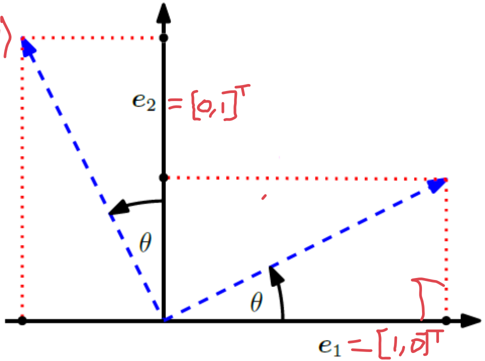
- How can we rotate each of these vector by θ as seen in the diagram below?

$\cos(\theta) = \frac{x}{r}$

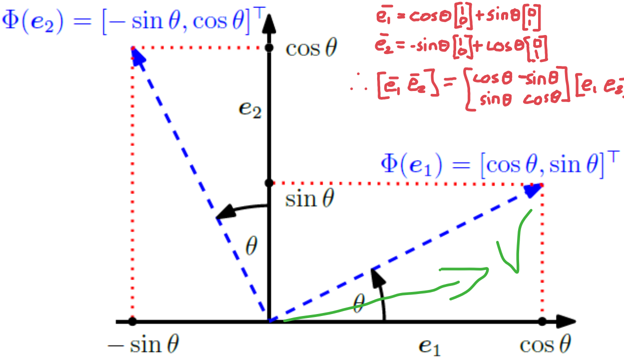
$= 1\cos(\theta)$ [1]

$\sin(\theta) = \frac{y}{r}$

$= \sin(\theta)$



Source: M.P. Deisenroth *et al*, Mathematics for Machine Learning (First Edition)



$e_1 = \begin{bmatrix}1\\0\end{bmatrix}$; $e_2 = \begin{bmatrix}0\\1\end{bmatrix}$

$\bar{e}_1 = \cos\theta\begin{bmatrix}1\\0\end{bmatrix} + \sin\theta\begin{bmatrix}0\\1\end{bmatrix}$

$\bar{e}_2 = -\sin\theta\begin{bmatrix}1\\0\end{bmatrix} + \cos\theta\begin{bmatrix}0\\1\end{bmatrix}$

$\therefore \begin{bmatrix}\bar{e}_1 & \bar{e}_2\end{bmatrix} = \begin{bmatrix}\cos\theta & -\sin\theta\\ \sin\theta & \cos\theta\end{bmatrix}\begin{bmatrix}e_1 & e_2\end{bmatrix}$

Source: M.P. Deisenroth *et al*, Mathematics for Machine Learning (First Edition)

$$\|\Phi(e_1)\| = \sqrt{\cos^2\theta + \sin^2\theta}$$
$$= \sqrt{1}$$
$$= 1$$

The two vectors form our new basis

$$\Phi(\mathbf{e}_1) = \begin{bmatrix} \cos\theta \\ \sin\theta \end{bmatrix}, \Phi(\mathbf{e}_2) = \begin{bmatrix} -\sin\theta \\ \cos\theta \end{bmatrix}. \tag{40}$$

from which we can concatenate them to form our change of basis matrix

$$\mathbf{R}(\theta) = \begin{bmatrix} \Phi(\mathbf{e}_1) & \Phi(\mathbf{e}_2) \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \tag{41}$$

$$\begin{bmatrix} \bar{e}_1 & \bar{e}_2 \end{bmatrix} = F \begin{bmatrix} e_1 & e_2 \end{bmatrix}$$

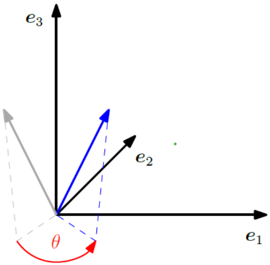
Rotations in $\mathbb{R}^3$

When we move to $\mathbb{R}^3$ we can rotate any two-dimensional plane about a one-dimensional axis.

- For which there are three choices

The easiest way to specify the general rotation matrix is to specify how the images of the standard basis $\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3$ are supposed to be rotated while ensuring that

- $\mathbf{Re}_1$, $\mathbf{Re}_2$, $\mathbf{Re}_3$ remain orthonormal to each other.



Source: M.P. Deisenroth *et al*, Mathematics for Machine Learning (First Edition)

Rotations in $\mathbb{R}^3$

- Rotation about the $\mathbf{e}_1$-axis

$$\mathbf{R}_1(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix} \tag{42}$$

Here, the $\mathbf{e}_1$ coordinate is fixed, and the counterclockwise rotation is performed in the $\mathbf{e}_2\mathbf{e}_3$ plane.

- Rotation about the $\mathbf{e}_2$-axis

$$\mathbf{R}_2(\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \tag{43}$$

If we rotate the $\mathbf{e}_1\mathbf{e}_3$ plane about the $\mathbf{e}_2$ axis, we need to look at the $\mathbf{e}_2$ axis from its "tip" towards the origin.

- Rotation about the $\mathbf{e}_3$-axis

$$\mathbf{R}_3(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{44}$$

Rotations in $\mathbb{R}^n$

The generalization of rotations from 2D and 3D to n-dimensional Euclidean vector spaces can be described as

- Fixing $n-2$ dimensions and restrict the rotation to a two-dimensional plane in the n-dimensional space.
- This type of rotation in a n-dimensional space is called a Givens rotation. Other do exist.

Givens Rotation

Let V be a n-dimensional Euclidean vector space and $\mathbf{R}_{ij} : V \to V$ an automorphism with transformation matrix

$$\mathbf{R}_{ij}(\theta) = \begin{bmatrix} \mathbf{I}_{i-1} & \mathbf{0} & \cdots & \cdots & \mathbf{0} \\ \mathbf{0} & \cos\theta & \mathbf{0} & -\sin\theta & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I}_{j-i-1} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \sin\theta & \mathbf{0} & \cos\theta & \mathbf{0} \\ \mathbf{0} & \cdots & \cdots & \mathbf{0} & \mathbf{I}_{n-j} \end{bmatrix} \tag{45}$$

for $1 \leq i < j \leq n$ and $\theta \in \mathbb{R}$. Then $\mathbf{R}_{ij}$ is a *Givens rotation*

- Essentially, $\mathbf{R}_{ij}$ is the identity matrix $\mathbf{I}_n$ with

$$r_{ii} = \cos\theta, \quad r_{ij} = -\sin\theta, \quad r_{ji} = \sin\theta, \quad r_{jj} = \cos\theta. \tag{46}$$

Properties of Rotations

Rotations have a number of useful properties (which follows from them being orthogonal matrices)

- Rotations preserve distances
- Rotations preserve angles

That said rotation matrices are still not generally commutative (in 3 and more dimensions), so the order of application must be considered.

Gram-Schmidt Process

Tuesday, March 29, 2022 10:12 PM

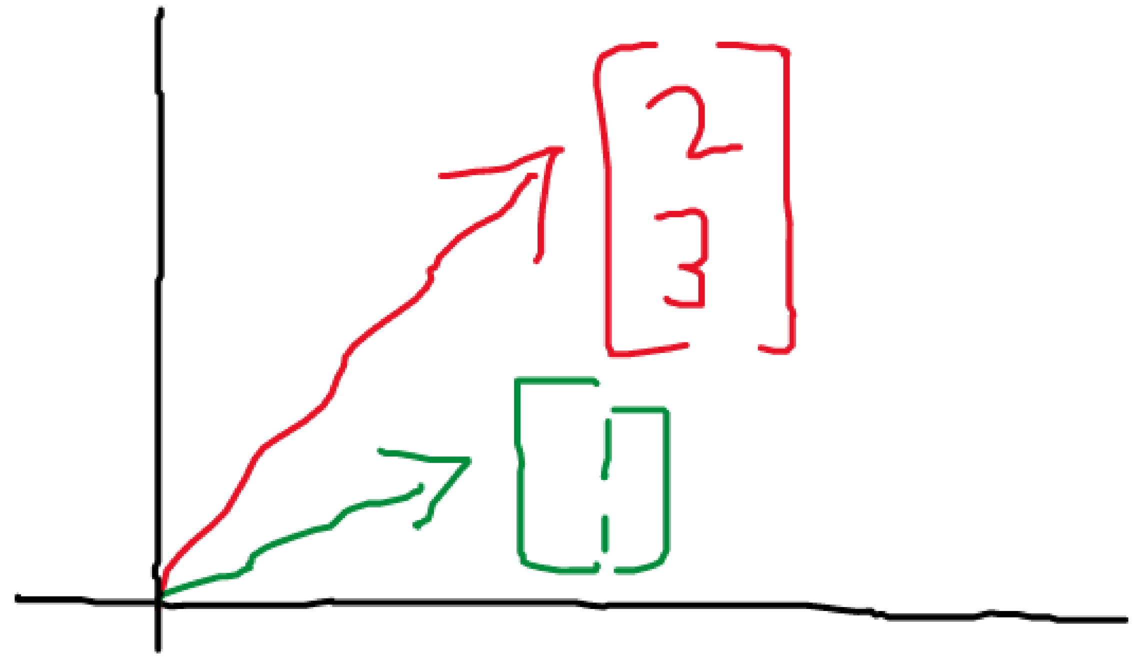
$$\mathrm{Proj}_u(v) = \frac{\langle u, v\rangle}{\langle u, u\rangle} u \Big\} \text{ Proj of } v \text{ onto } u$$

(1) $u_1 = v_1 \Rightarrow e_1 = \frac{u_1}{\|u_1\|}$

(2) $u_2 = v_2 - \mathrm{Proj}_{u_1}(v_2) \Rightarrow e_2 = \frac{u_2}{\|u_2\|}$

$\vdots$

(n) $u_n = v_n - \sum_{j=1}^{n-1} \mathrm{Proj}_{u_j}(v_n) \Rightarrow e_n = \frac{u_n}{\|u_n\|}$



$$u_1 = \begin{bmatrix}2\\3\end{bmatrix}$$

$$u_2 = \begin{bmatrix}1\\1\end{bmatrix} - \frac{\langle u_1, v_2\rangle}{\langle u_1, u_1\rangle} u_1$$

$$= \begin{bmatrix}1\\1\end{bmatrix} - \frac{\begin{bmatrix}2 & 3\end{bmatrix}\cdot\begin{bmatrix}1\\1\end{bmatrix}}{\begin{bmatrix}2 & 3\end{bmatrix}\begin{bmatrix}2\\3\end{bmatrix}}\begin{bmatrix}2\\3\end{bmatrix}$$

$$= \begin{bmatrix}1\\1\end{bmatrix} - \frac{5}{13}\begin{bmatrix}2\\3\end{bmatrix}$$

$$= \frac{1}{13}\begin{bmatrix}3\\-2\end{bmatrix}$$

$$\frac{1}{\sqrt{13}}\begin{bmatrix}2 & 3\end{bmatrix} \cdot \frac{1}{\sqrt{13}}\begin{bmatrix}3\\-2\end{bmatrix}$$

$$= \frac{1}{13}(6-6) = 0$$

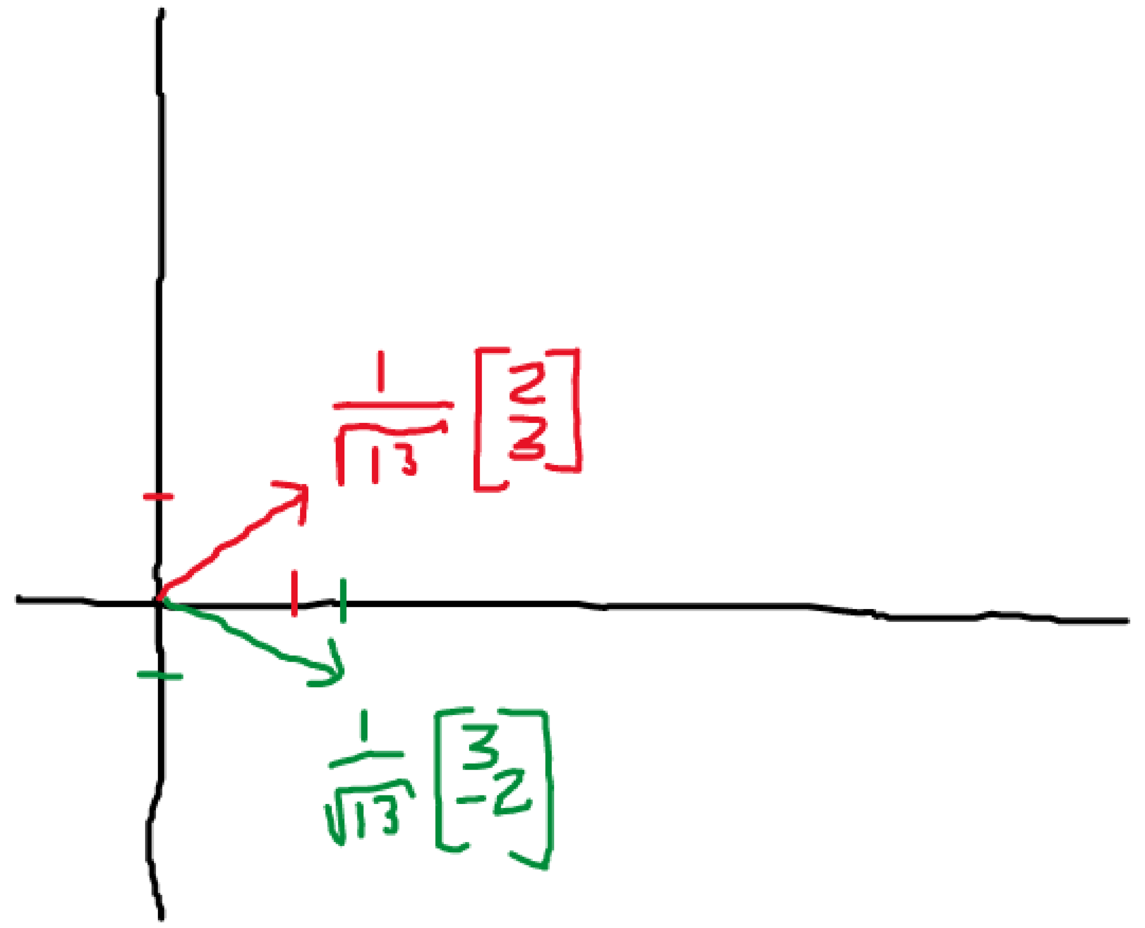
$$e_1 = \frac{1}{\sqrt{\begin{bmatrix}2 & 3\end{bmatrix}\begin{bmatrix}2\\3\end{bmatrix}}}\begin{bmatrix}2\\3\end{bmatrix}$$

$$= \frac{1}{\sqrt{13}}\begin{bmatrix}2\\3\end{bmatrix}$$

$$e_2 = \frac{1}{\sqrt{\frac{1}{13}\begin{bmatrix}3 & -2\end{bmatrix}\frac{1}{13}\begin{bmatrix}3\\-2\end{bmatrix}}} \cdot \frac{1}{13}\begin{bmatrix}3\\-2\end{bmatrix}$$

$$= \frac{13}{\sqrt{13}} \cdot \frac{1}{13}\begin{bmatrix}3\\-2\end{bmatrix}$$

$$= \frac{1}{\sqrt{13}}\begin{bmatrix}3\\-2\end{bmatrix}$$



$$\Pi_b(x) = \frac{b b^T}{\|b\|^2} x \qquad b = \frac{1}{\sqrt{13}}\begin{bmatrix}2\\3\end{bmatrix} = u_1$$

$$x = \begin{bmatrix}1\\1\end{bmatrix} = v_2$$

$$\Pi_b(x) = \frac{\left(\frac{1}{\sqrt{13}}\right)\left(\frac{1}{\sqrt{13}}\right)\begin{bmatrix}2\\3\end{bmatrix}\begin{bmatrix}2 & 3\end{bmatrix}}{\left(\frac{1}{\sqrt{13}}\right)\left(\frac{1}{\sqrt{13}}\right)\begin{bmatrix}2 & 3\end{bmatrix}\begin{bmatrix}2\\3\end{bmatrix}}\begin{bmatrix}1\\1\end{bmatrix}$$

$$u_2 = \begin{bmatrix}1\\1\end{bmatrix} - \frac{1}{13}\begin{bmatrix}4 & 6\\6 & 9\end{bmatrix}\begin{bmatrix}1\\1\end{bmatrix}$$

$$u_2 = \begin{bmatrix}1\\1\end{bmatrix} - \frac{1}{13}\begin{bmatrix}10\\15\end{bmatrix} = \frac{1}{13}\begin{bmatrix}3\\-2\end{bmatrix}$$

$\mathrm{Proj}_b(x)$

$$= \frac{b b^T}{\|b\|^2} x = b\frac{b^T x}{\|b\|^2}$$

$$= \frac{b^T x}{\|b\|^2} b$$

$$= \frac{\langle b, x\rangle}{\langle b, b\rangle} b$$