Least Squares Approximation, From Scratch

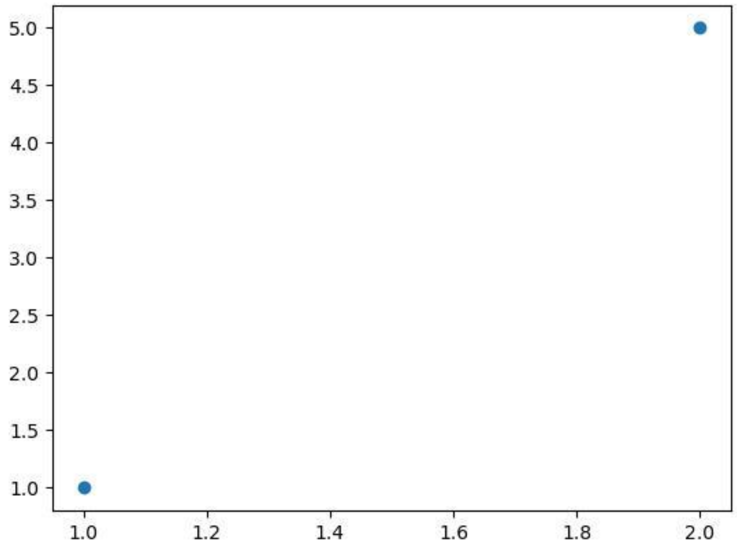
Zayn Patel

3 types of solutions based on m equations, n unknowns

Type of solution	**m and n relationship**
No solution	$m > n$
Infinitely many solutions	$m < n$
Unique solution	$m = n$

Example of a unique solution

Ordered pair: (1, 1); (2, 5)



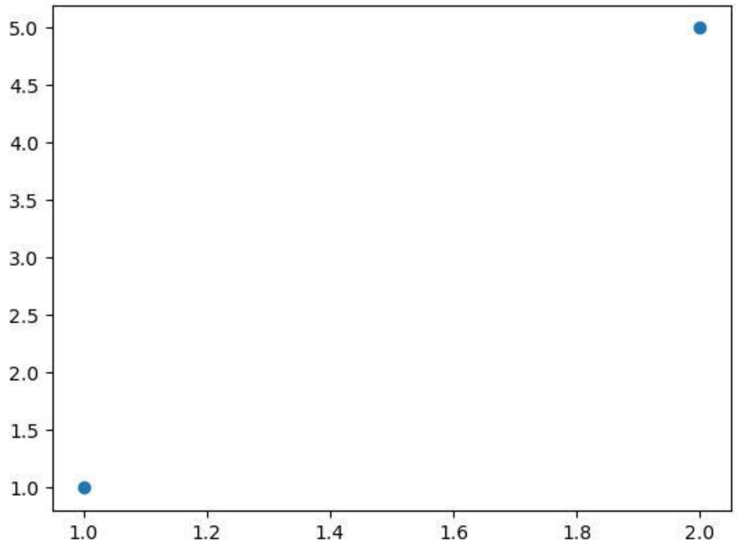
Example of a unique solution

Ordered pair: (1, 1); (2, 5)

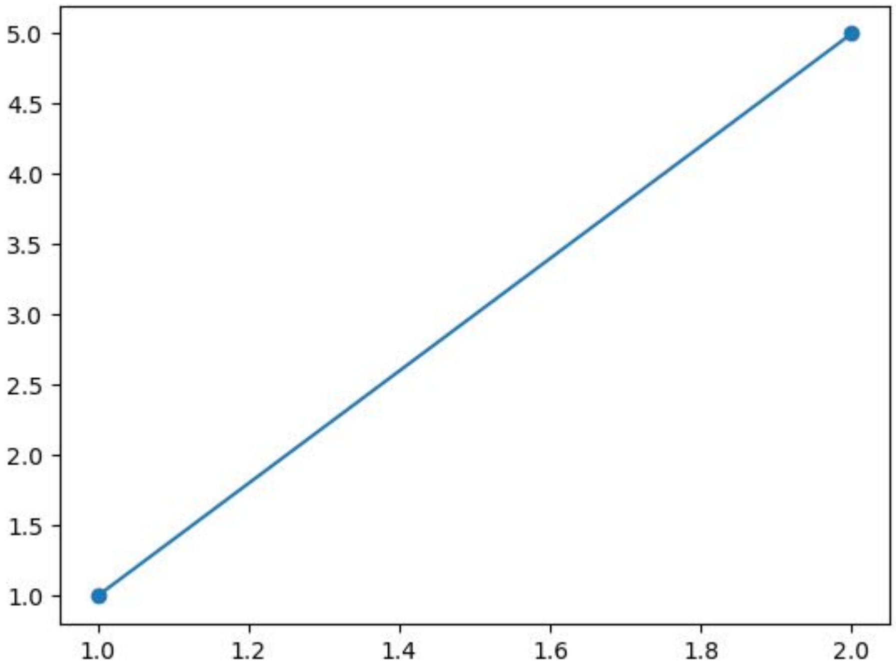
Equations:

$C + D = 1$

$C + 2D = 5$

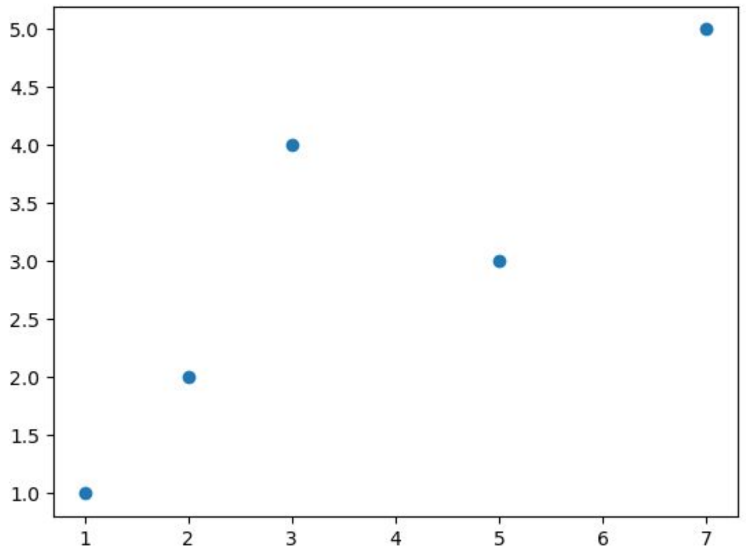


“Best fit line” for two equations, two unknowns



Example of no solution

Ordered pair: (1, 1); (2, 2); (3, 4); (5,3); (7,5)



Example of no solution

Ordered pair: (1, 1); (2, 2); (3, 4); (5,3); (7,5)

Equations:

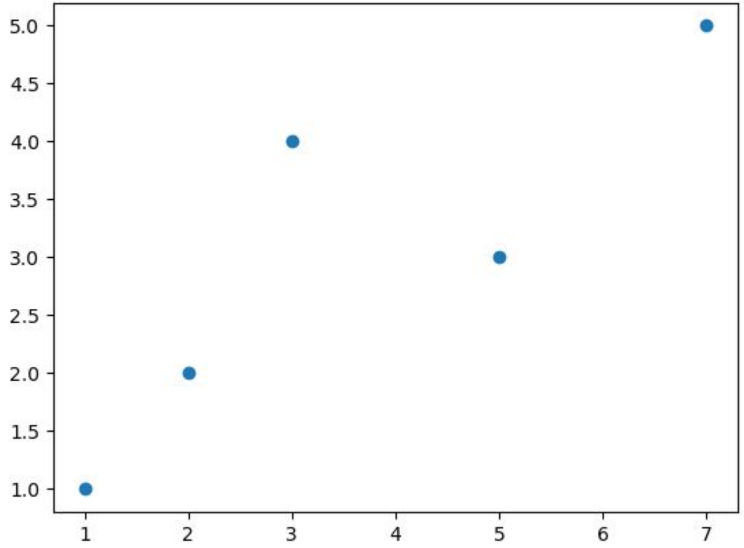
$C + D = 1$

$C + 2D = 2$

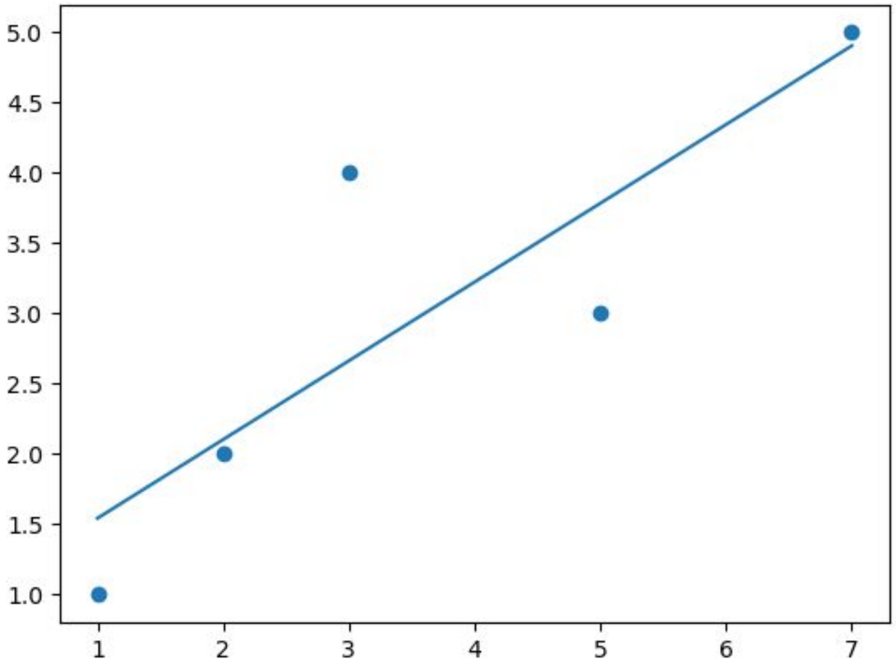
$C + 3D = 4$

$C + 5D = 3$

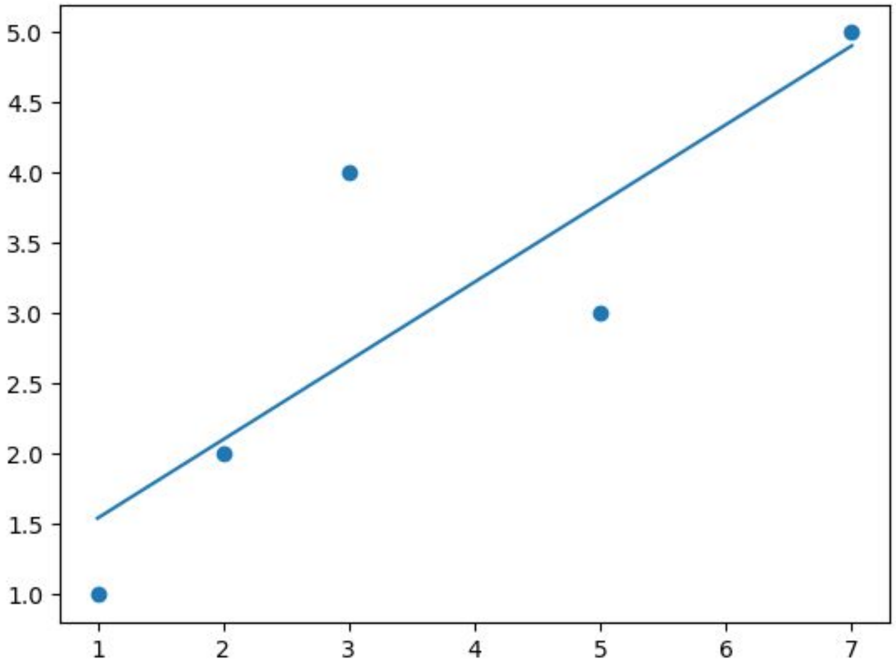
$C + 7D = 5$



“Best fit line” for five equations, two unknowns



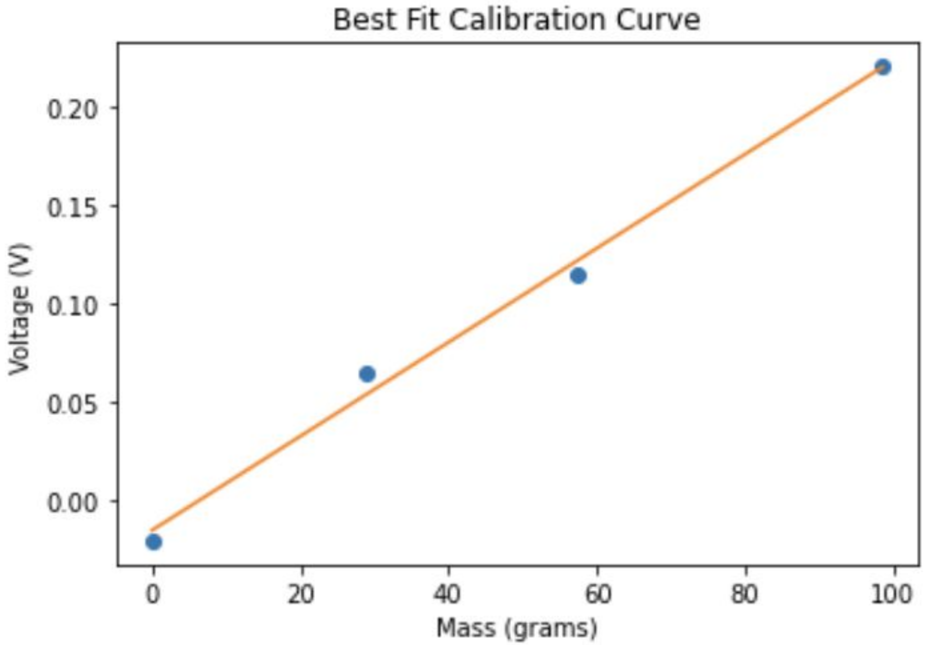
“Best fit line” for five equations, two unknowns



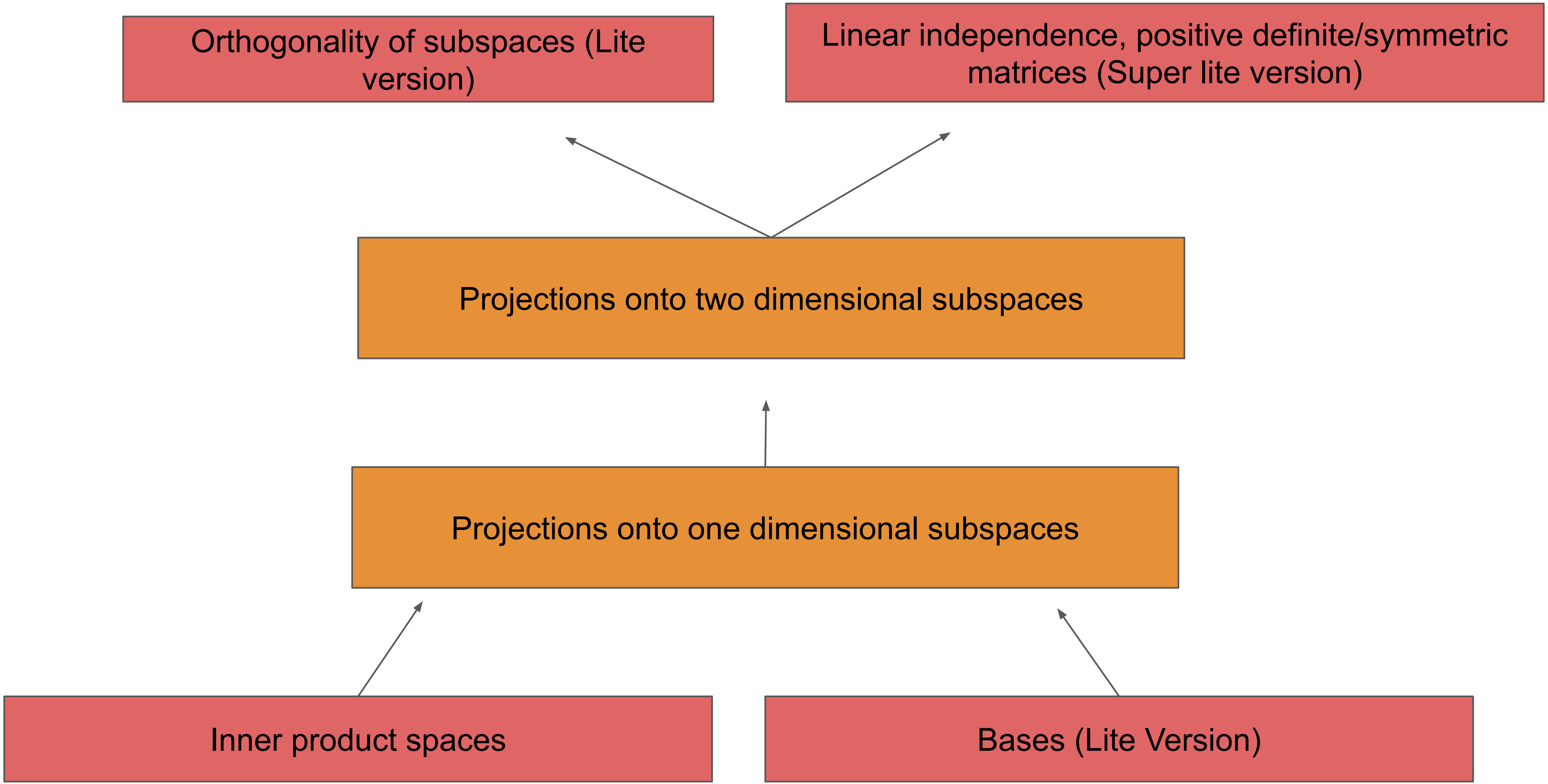
Key idea: Orthogonal projections help us solve Ax = b with no solutions. We call this **Ax = p.**

More motivation for least squares: ISIM example

```
Equation of the line of best fit: y =
0.002379 x - 0.01473
```



Source: [Zayn Patel's ISIM Lab 3]


Orthogonality of subspaces (Lite version)
Linear independence, positive definite/symmetric matrices (Super lite version)
Projections onto two dimensional subspaces
Projections onto one dimensional subspaces
Inner product spaces
Bases (Lite Version)

Math Foundation #1: Inner products

- We define inner products via bilinear mappings. Bilinear mappings take **two arguments** and are **linear in each argument**
  - $\Omega(\lambda x + \psi y, z) = \lambda\Omega(x, z) + \psi\Omega(y, z)$
  - $\Omega(x, \lambda y + \psi z) = \lambda\Omega(x, y) + \psi\Omega(x, z)$

Math Foundation #1: Inner products

- We define inner products via bilinear mappings. Bilinear mappings take **two arguments** and are **linear in each argument**
  - $\Omega(\lambda x + \psi y, z) = \lambda\Omega(x, z) + \psi\Omega(y, z)$
  - $\Omega(x, \lambda y + \psi z) = \lambda\Omega(x, y) + \psi\Omega(x, z)$
- Bilinear mappings ensure vector operations (scalar multiplication and vector addition) hold
  - Important because we will define inner products as “special” vector spaces so we want vector space operations to hold

Math Foundation #1: Inner products

- We define inner products via bilinear mappings. Bilinear mappings take **two arguments** and are **linear in each argument**
  - $\Omega(\lambda x + \psi y, z) = \lambda\Omega(x, z) + \psi\Omega(y, z)$
  - $\Omega(x, \lambda y + \psi z) = \lambda\Omega(x, y) + \psi\Omega(x, z)$
- Bilinear mappings ensure vector operations (scalar multiplication and vector addition) hold
  - Important because we will define inner products as “special” vector spaces so we want vector space operations to hold
- Bilinear mappings take two vectors and maps them to a real number (V x V) -> R

Math Foundation #1 cont: Inner products

- Inner products are symmetric <x, y> = <y, x>
    - Without this property, distance calculations wouldn’t hold

Math Foundation #1 cont: Inner products

- Inner products are symmetric <x, y> = <y, x>
  - Without this property, distance calculations wouldn’t hold
- Inner products are positive definite <x, x> is > 0 if x > 0, <x, x> = 0 if x = 0
  - Without this property, angle calculations wouldn’t hold

Math Foundation #1 cont: Inner products

- Inner products are symmetric <x, y> = <y, x>
  - Without this property, distance calculations wouldn't hold
- Inner products are positive definite <x, x> is > 0 if x > 0, <x, x> = 0 if x = 0
  - Without this property, angle calculations wouldn't hold
- Math definition: positive definite, symmetric bilinear mapping (V x V) -> R is called an inner product on V.
  - You can now see that *because* we want the inner product for distances, angles, orthogonality we need more properties than vector spaces

Projections onto one-dimensional subspaces

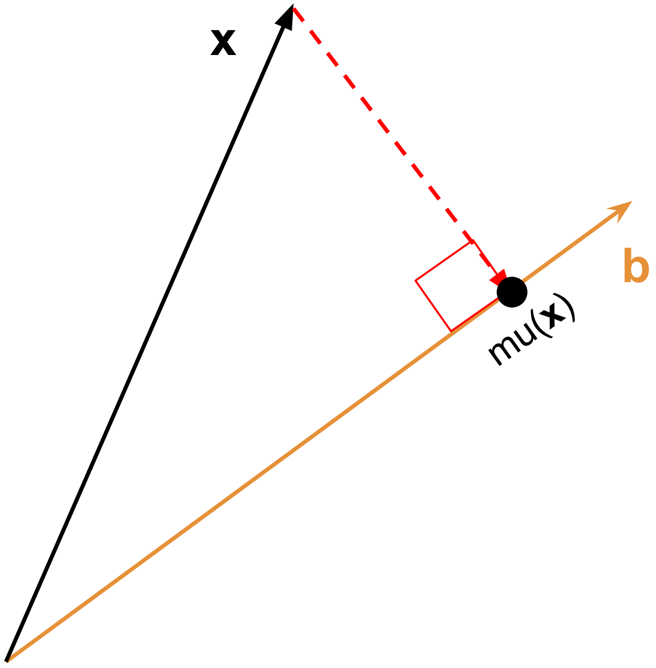
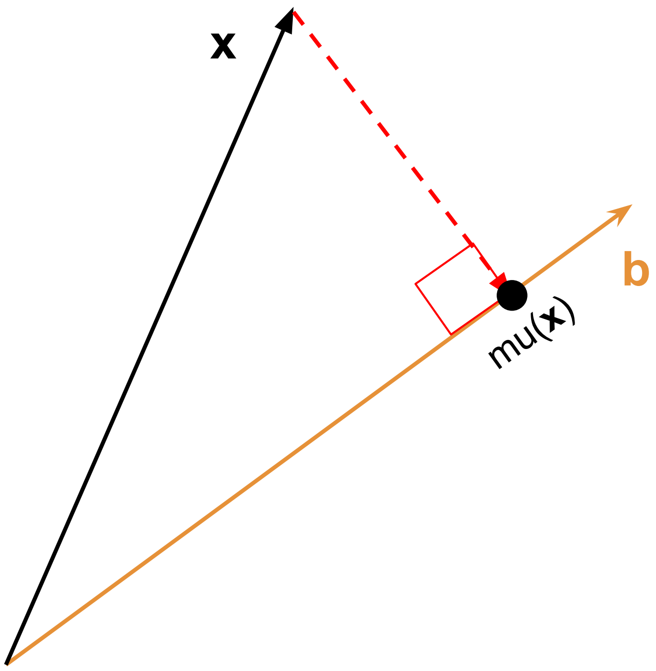


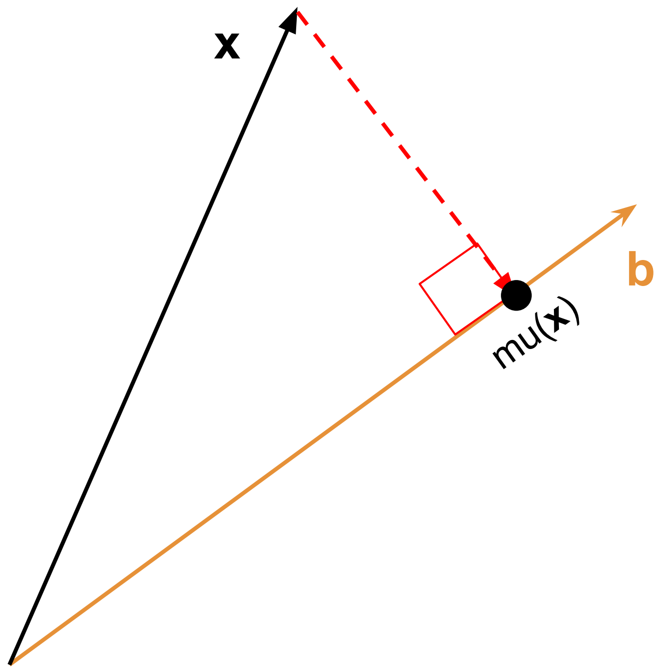
Figure inspired by MML textbook, Section 3.8

Projections onto one-dimensional subspaces



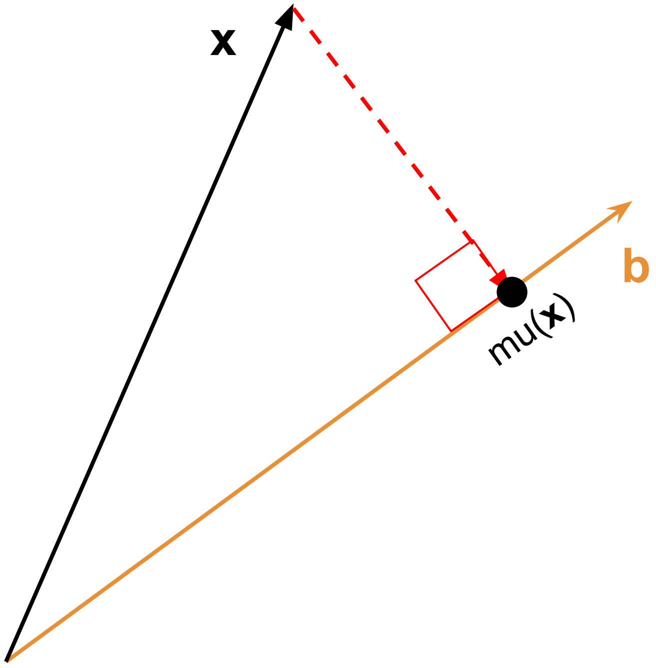
- We want to minimize distance between **x** and mu(**x**) so **orthogonal** projection. b is a **basis vector.**

Projections onto one-dimensional subspaces



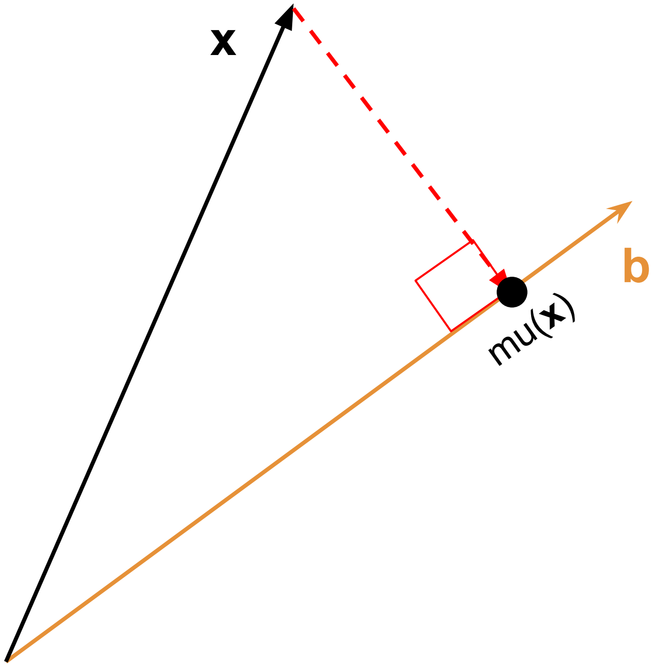
- We want to minimize distance between **x** and mu(**x**) so **orthogonal** projection. b is a **basis vector.**
- Projection point mu(**x**) is now apart of b.Specifically it's *some scalar multiple* of b.

Projections onto one-dimensional subspaces



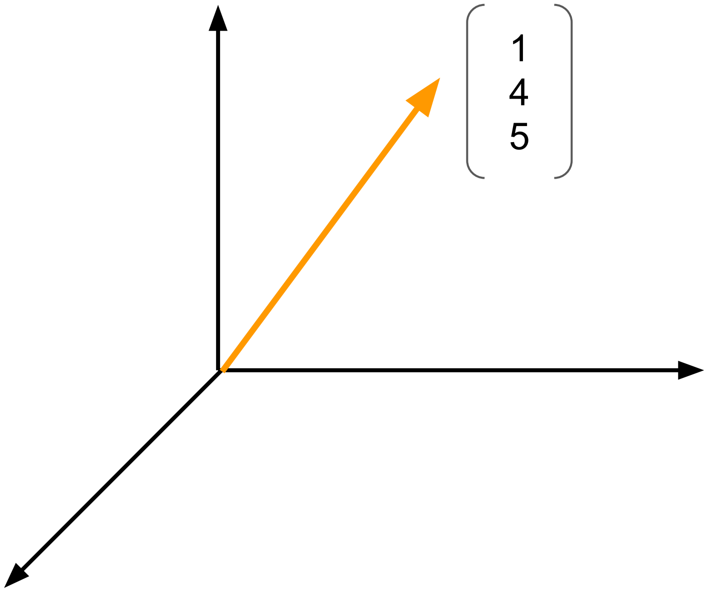
- We want to minimize distance between **x** and mu(**x**) so **orthogonal** projection. b is a **basis vector.**
- Projection point mu(**x**) is now apart of b. Specifically it's *some scalar multiple* of b.
  - Let's define mu(**x**) = λ**b**

Projections onto one-dimensional subspaces



- To find λ in mu(**x**) = λ**b** we want to figure out what **linear combination** of λ and **b** gets us to mu(**x**).
  - We need to do this *because* x ∈ R and mu(**x**) ∈ U. And U ⊆ R.

Math Foundation #2: Bases (lite version)

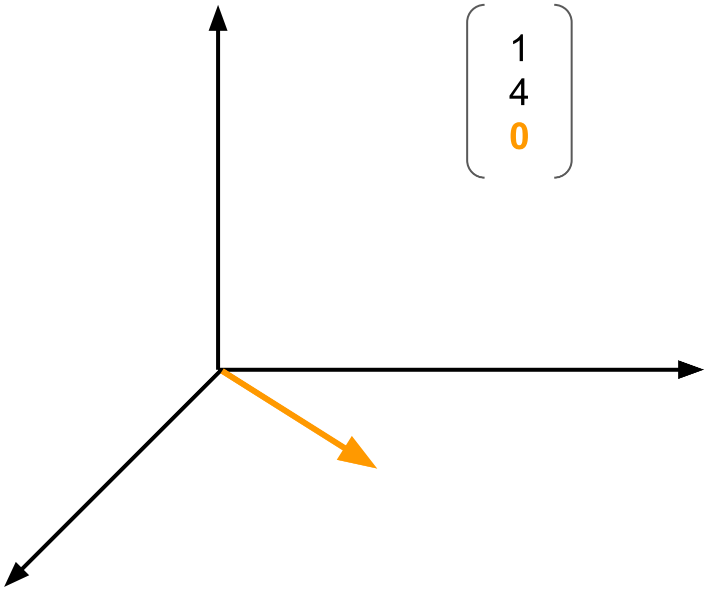


$$\begin{pmatrix} 1 \\ 4 \\ 5 \end{pmatrix}$$

Using the “standard” basis vectors for R^3:

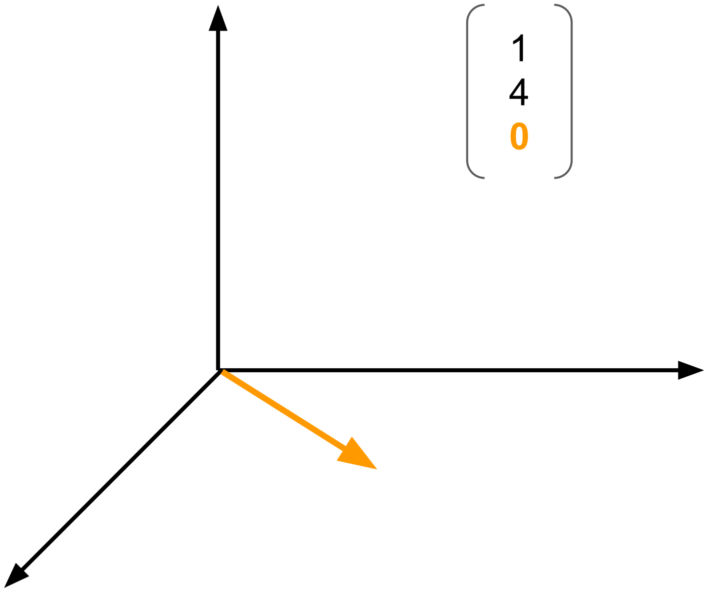
$$1 \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} + 4 \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} + 5 \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

Math Foundation #2: Bases (lite version)



$$\begin{pmatrix} 1 \\ 4 \\ 0 \end{pmatrix}$$

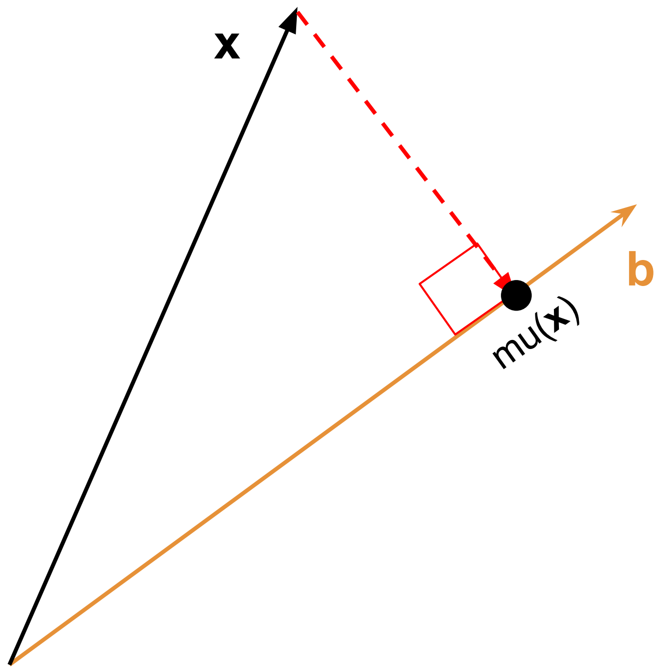
Math Foundation #2: Bases (lite version)



Using the “standard” basis vectors for R^2 we need to find a **new basis representation** since are not in R^3 anymore.

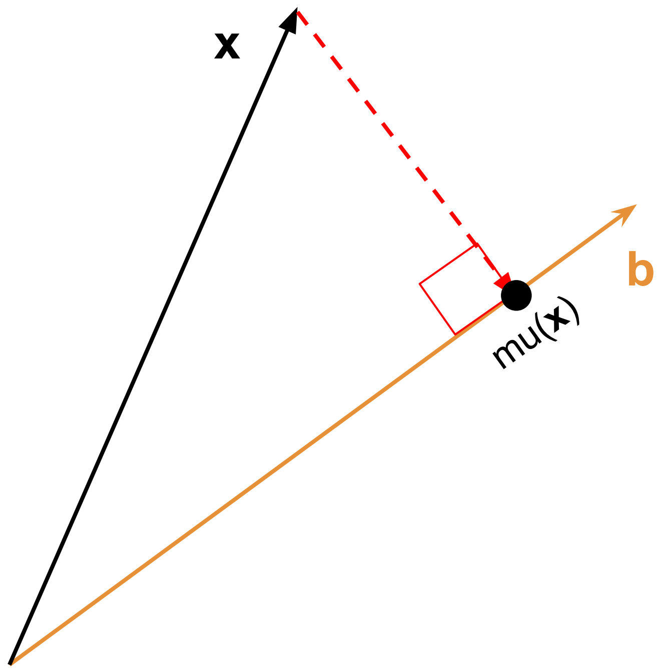
Key idea: Vectors are defined w.r.t a basis. When the vector is projected onto a new space we need to figure out what basis represents this new vector.

Projections onto one-dimensional subspaces



- To find λ in mu(**x**) = λ**b** we want to figure out what **linear combination** of λ and **b** gets us to mu(**x**).
  - We need to do this *because* x ∈ R and mu(**x**) ∈ U. And U ⊆ R.

Projections onto one-dimensional subspaces



- To find λ in mu(**x**) = λ**b** we want to figure out what **linear combination** of λ and **b** gets us to mu(**x**).
  - We need to do this *because* x ∈ R and mu(**x**) ∈ U. And U ⊆ R.
- We can write this as

  <**x** - mu(**x**), **b**> = 0

  or

  <x - λb, b> = 0

Why? Because we know their inner product has to be zero since these are orthogonal.

Projections onto one-dimensional subspaces

- Let's use $\langle \mathbf{x} - \lambda \mathbf{b}, \mathbf{b} \rangle = 0$ to solve for λ.

Projections onto one-dimensional subspaces

- Let's use $\langle \mathbf{x} - \lambda\mathbf{b}, \mathbf{b} \rangle = 0$ to solve for λ.
- Recall that we learned a bilinear mapping is **linear in each argument**. We can use this fact here by rewriting this:
  - $\langle \mathbf{x}, \mathbf{b} \rangle + \langle -\lambda\mathbf{b}, \mathbf{b} \rangle = 0$

Projections onto one-dimensional subspaces

- Let's use $\langle \mathbf{x} - \lambda\mathbf{b}, \mathbf{b}\rangle = 0$ to solve for λ.
- Recall that we learned a bilinear mapping is **linear in each argument**. We can use this fact here by rewriting this:
  - $\langle \mathbf{x}, \mathbf{b}\rangle + \langle -\lambda\mathbf{b}, \mathbf{b}\rangle = 0$
  - $\langle \mathbf{x}, \mathbf{b}\rangle - \lambda\langle \mathbf{b}, \mathbf{b}\rangle = 0$

Projections onto one-dimensional subspaces

- Let's use $\langle \mathbf{x} - \lambda\mathbf{b}, \mathbf{b} \rangle = 0$ to solve for λ.
- Recall that we learned a bilinear mapping is **linear in each argument**. We can use this fact here by rewriting this:
    - $\langle \mathbf{x}, \mathbf{b} \rangle + \langle -\lambda\mathbf{b}, \mathbf{b} \rangle = 0$
    - $\langle \mathbf{x}, \mathbf{b} \rangle - \lambda\langle \mathbf{b}, \mathbf{b} \rangle = 0$
    - $\langle \mathbf{x}, \mathbf{b} \rangle = \lambda\langle \mathbf{b}, \mathbf{b} \rangle$

Projections onto one-dimensional subspaces

- Let's use $\langle \mathbf{x} - \lambda\mathbf{b}, \mathbf{b} \rangle = 0$ to solve for λ.
- Recall that we learned a bilinear mapping is **linear in each argument**. We can use this fact here by rewriting this:
  - $\langle \mathbf{x}, \mathbf{b} \rangle + \langle -\lambda\mathbf{b}, \mathbf{b} \rangle = 0$
  - $\langle \mathbf{x}, \mathbf{b} \rangle - \lambda\langle \mathbf{b}, \mathbf{b} \rangle = 0$
  - $\langle \mathbf{x}, \mathbf{b} \rangle = \lambda\langle \mathbf{b}, \mathbf{b} \rangle$
  - $\langle \mathbf{x}, \mathbf{b} \rangle / \langle \mathbf{b}, \mathbf{b} \rangle = \lambda$

Projections onto one-dimensional subspaces

- Let's use $\langle \mathbf{x} - \lambda\mathbf{b}, \mathbf{b}\rangle = 0$ to solve for λ.
- Recall that we learned a bilinear mapping is **linear in each argument**. We can use this fact here by rewriting this:
  - $\langle \mathbf{x}, \mathbf{b}\rangle + \langle -\lambda\mathbf{b}, \mathbf{b}\rangle = 0$
  - $\langle \mathbf{x}, \mathbf{b}\rangle - \lambda\langle \mathbf{b}, \mathbf{b}\rangle = 0$
  - $\langle \mathbf{x}, \mathbf{b}\rangle = \lambda\langle \mathbf{b}, \mathbf{b}\rangle$
  - $\langle \mathbf{x}, \mathbf{b}\rangle / \langle \mathbf{b}, \mathbf{b}\rangle = \lambda$
  - **Note:** I can divide because $\langle \mathbf{b}, \mathbf{b}\rangle$ is a number when we compute the dot product. When we get to matrices we will take the inverse.

Projections onto one-dimensional subspaces

- Now that we have λ we can look back at the way we define mu(**x**).
  - mu(**x**) = λ**b**

Projections onto one-dimensional subspaces

- Now that we have λ we can look back at the way we define mu(**x**).
  - mu(**x**) = λ**b**
  - We now have λ so our equation for the point is:

$$\boxed{\frac{\langle \mathbf{x}, \mathbf{b} \rangle}{\langle \mathbf{b}, \mathbf{b} \rangle} \mathbf{b}}$$

DEFINITION:

$$\mathrm{Proj}_{\bar{b}}\, \bar{a} = \left(\frac{\bar{a} \cdot \bar{b}}{\bar{b} \cdot \bar{b}} \right) \bar{b}$$

Key idea: This is the same formula for the projection as day 8/SSA's derivation. But, we used bilinear mappings instead of geometry.

Projections onto one-dimensional subspaces

- Lastly, we obtain the projection matrix, P**x**. This will exist because mu(x) = **P**x.
    - This matrix will project every vector x onto the line spanned by b.

Projections onto one-dimensional subspaces

- Lastly, we obtain the projection matrix, P**x**. This will exist because mu(x) = **P**x.
  - This matrix will project every vector x onto the line spanned by b.
  - **Key idea:** We can obtain a matrix for the projection *because* projections are a linear map. A linear map is a transformation between two spaces that preserves linear combinations.

Projections onto one-dimensional subspaces

- Lastly, we obtain the projection matrix, P**x**. This will exist because mu(x) = **P**x.
  - This matrix will project every vector x onto the line spanned by b.
  - **Key idea:** We can obtain a matrix for the projection *because* projections are a linear map. A linear map is a transformation between two spaces that preserves linear combinations.
- mu(x) = λ**b**

Projections onto one-dimensional subspaces

- Lastly, we obtain the projection matrix, P**x**. This will exist because mu(x) = **P**x.
    - This matrix will project every vector x onto the line spanned by b.
    - **Key idea:** We can obtain a matrix for the projection *because* projections are a linear map. A linear map is a transformation between two spaces that preserves linear combinations.
- mu(x) = λ**b**
- Recall that λ = <**x**, **b**> / <**b**, **b**>

Projections onto one-dimensional subspaces

$$\mathbf{P} = \mathbf{b} * \langle \mathbf{x}, \mathbf{b} \rangle / \langle \mathbf{b}, \mathbf{b} \rangle$$

Projections onto one-dimensional subspaces

$$\mathbf{P} = \mathbf{b} * \langle \mathbf{x}, \mathbf{b} \rangle / \langle \mathbf{b}, \mathbf{b} \rangle$$

$\mathbf{P} = \mathbf{b} * \langle \mathbf{x}, \mathbf{b} \rangle / ||b||^2$ (Step: Convert dot product of b into norm.)

Projections onto one-dimensional subspaces

$\mathbf{P} = \mathbf{b} * \langle \mathbf{x}, \mathbf{b} \rangle / \langle \mathbf{b}, \mathbf{b} \rangle$

$\mathbf{P} = \mathbf{b} * \langle \mathbf{x}, \mathbf{b} \rangle / ||b||^2$ (Step: Convert dot product of b into norm.)

$\mathbf{P} = \mathbf{b} * \mathbf{b}^\mathbf{T} * \mathbf{x} / ||b||^2$ (Step: Convert <x, b> into written equation.)

Projections onto one-dimensional subspaces

P = b * <x, b> / <b, b>

P = b * <**x**, **b**> / $||b||^2$ (Step: Convert dot product of b into norm.)

P = b * $\mathbf{b}^{\mathbf{T}}$ * **x** / $||b||^2$ (Step: Convert <x, b> into written equation.)

P = ($\mathbf{b}$*$\mathbf{b}^{\mathbf{T}}$ // $||b||^2$) * x (Step: Rewritten in **Px** form.)

Projections onto one-dimensional subspaces

$$\mathbf{P} = \mathbf{b} * \langle \mathbf{x}, \mathbf{b} \rangle / \langle \mathbf{b}, \mathbf{b} \rangle$$

$\mathbf{P} = \mathbf{b} * \langle \mathbf{x}, \mathbf{b} \rangle / ||b||^2$ (Step: Convert dot product of b into norm.)

$\mathbf{P} = \mathbf{b} * \mathbf{b}^{\mathbf{T}} * \mathbf{x} \; / ||b||^2$ (Step: Convert <x, b> into written equation.)

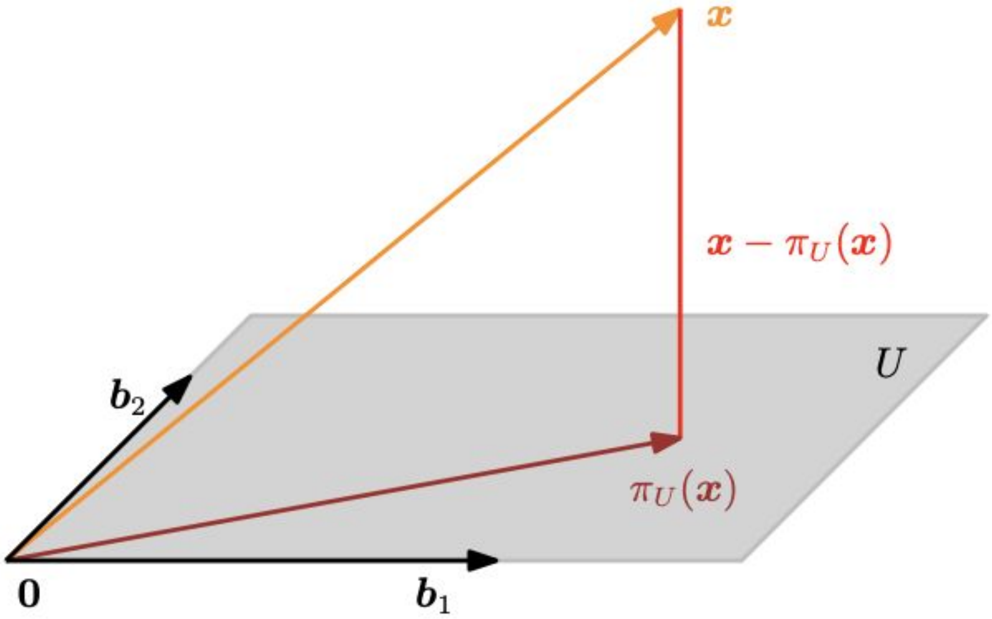
$\mathbf{P} = (\mathbf{b}*\mathbf{b}^{\mathbf{T}} // ||b||^2) * x$ (Step: Rewritten in **Px** form.)

Key idea: We can multiply any x by P to see if it’s in the column space of P. In other words, we can see if an x is in the subspace spanned by b or not.

Example 3.10 from MML on chalkboard

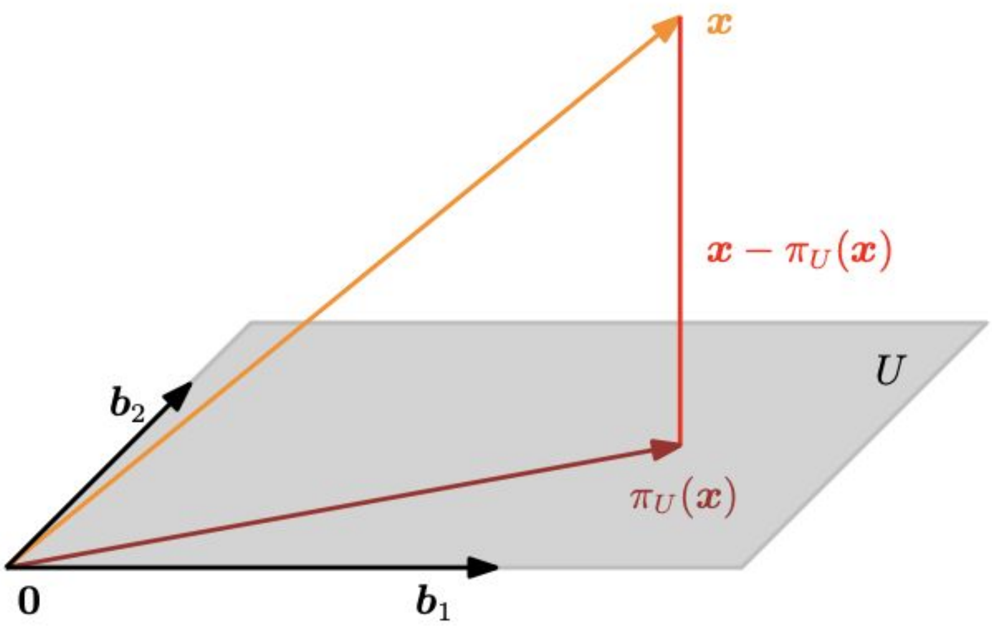
Source: MML Textbook Chapter 3

Projections onto general subspaces



[Figure taken from MML textbook, Section 3.8]()

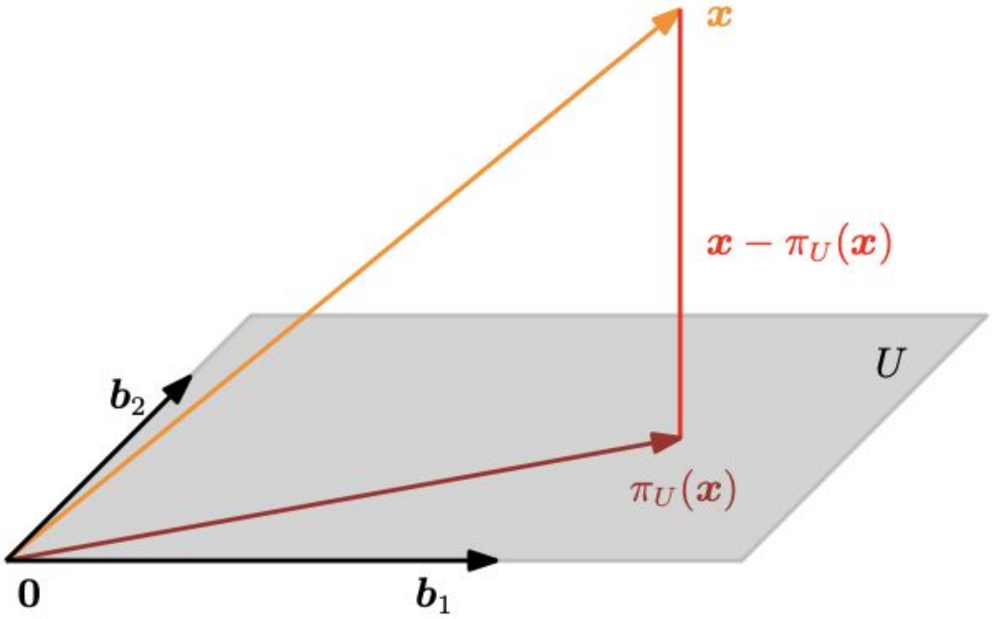
Projections onto general subspaces



- We will use matrices instead of vectors because subspace has dimension ≥ 1.
  - Matrix inverses instead of division.

Figure taken from MML textbook, Section 3.8

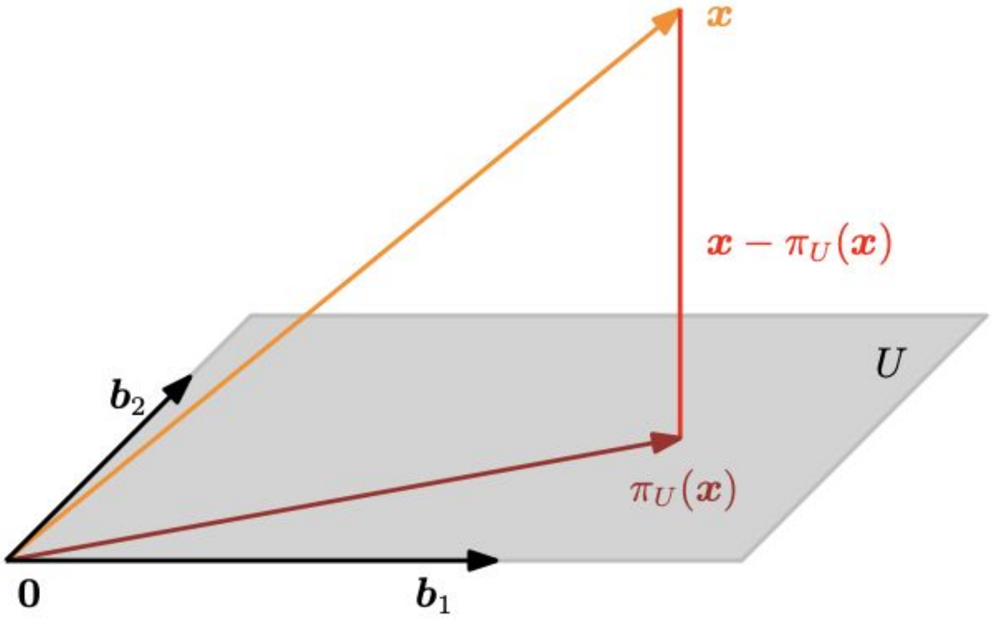
Projections onto general subspaces



- We will use matrices instead of vectors because subspace has dimension ≥ 1.
  - Matrix inverses instead of division.
- We will follow the same steps as before: find coordinate, projection, and projection matrix.

Figure taken from MML textbook, Section 3.8

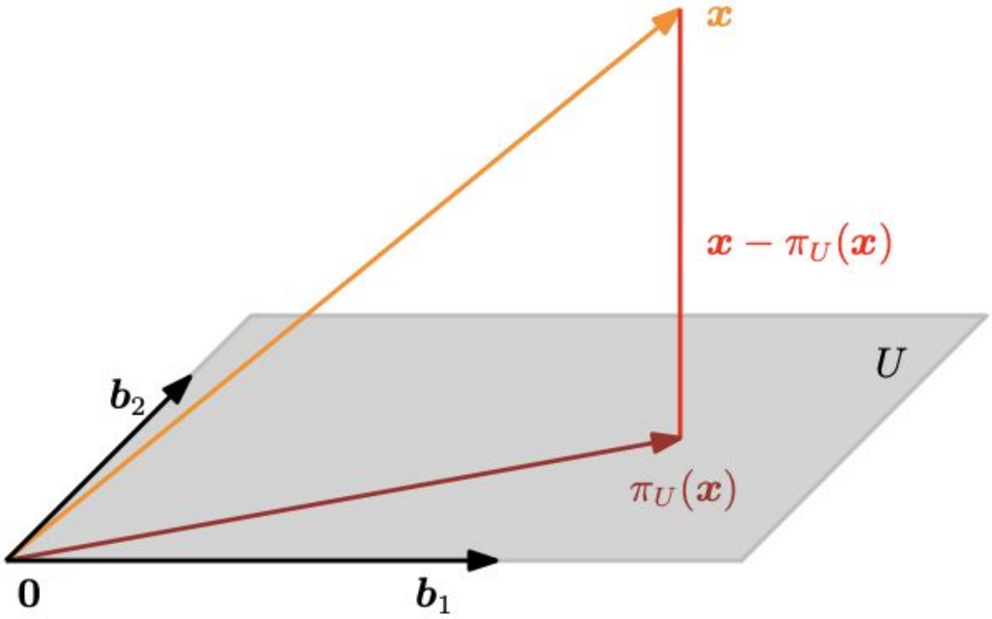
Quick aside: Orthogonality of subspaces



- When we project **x** onto U it's in the column space. U is defined by some matrix and b_1 and b_2 are standard basis vectors of U.

Figure taken from MML textbook. Section 3.8

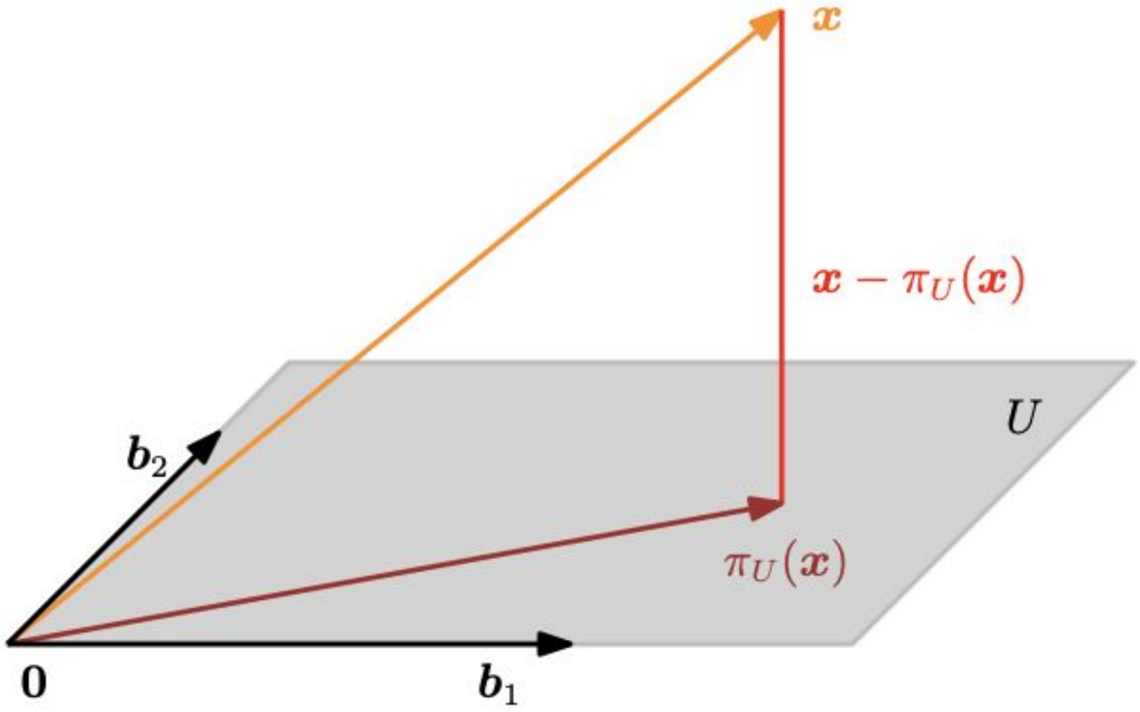
Quick aside: Orthogonality of subspaces



- When we project **x** onto U it's in the column space. U is defined by some matrix and b_1 and b_2 are standard basis vectors of U.
- **x** - mu(**x**) is orthogonal to *everything* in the column space, U. **Called N(A^T).**
  - Orthogonal because this is our minimum distance condition.

Figure taken from MML textbook. Section 3.8

Projections onto general subspaces: Find λ



[Figure taken from MML textbook, Section 3.8]

Projections onto general subspaces: Find λ

- Find coordinates $\lambda_1, \ldots, \lambda_m$ with respect to the basis of U
  - Why? We recall that the vector **x** is no longer in R^3 but in a subspace, U, which is in R^2. We need to find how it's "represented by the vectors in R^2.

Figure taken from MML textbook. Section 3.8

Projections onto general subspaces: Find λ

- Find coordinates $\lambda_1, \ldots, \lambda_m$ with respect to the basis of U
  - Why? We recall that the vector **x** is no longer in R^3 but in a subspace, U, which is in R^2. We need to find how it's "represented by the vectors in R^2.
- We will define mu(**x**) = **B**λ where **B**'s columns are the basis vectors of U.
  - So $\mathbf{B} = [b_1, \ldots, b_m]$

Figure taken from MML textbook, Section 3.8

Projections onto general subspaces: Find λ

- Find coordinates $\lambda_1, \ldots, \lambda_m$ with respect to the basis of U
  - Why? We recall that the vector **x** is no longer in R^3 but in a subspace, U, which is in R^2. We need to find how it's "represented by the vectors in R^2.
- We will define mu(**x**) = **B**λ where **B**'s columns are the basis vectors of U.
- Because of our orthogonality of subspaces observation we obtain:

$$b_1^T(\mathbf{x} - mu(\mathbf{x})) = 0$$

$$b_2^T(\mathbf{x} - mu(\mathbf{x})) = 0$$

$$b_n^T(\mathbf{x} - mu(\mathbf{x})) = 0$$

Key idea: All dot products of basis vector and error (**x** - mu(**x**)) are orthogonal.

Projections onto general subspaces: Find λ

$$\begin{pmatrix} \mathbf{b}_1^T \\ \cdot \\ \cdot \\ \cdot \\ \mathbf{b}_m^T \end{pmatrix} \begin{pmatrix} \mathbf{x} - \mathbf{B}\lambda \end{pmatrix} = 0$$

Projections onto general subspaces: Find λ

$$\begin{pmatrix} \mathbf{b}_1^T \\ \cdot \\ \cdot \\ \cdot \\ \mathbf{b}_m^T \end{pmatrix} \begin{pmatrix} \mathbf{x} - \mathbf{B}\lambda \end{pmatrix} = 0$$

Key idea (*again but in matrix form*): All dot products of basis vector and error (**x** - mu(**x**)) are orthogonal. Recall on slide 53 we said mu(**x**) = **B**λ.

Projections onto general subspaces: Find λ

- $\mathbf{B}^T * (\mathbf{x} - \mathbf{B}\lambda) = 0$
  - Recall that **B** is the set of basis vectors in our space. We transposed it on slide 56, $\mathbf{B}^T$ to make the matmul work.
  - Also recall that everything is set to zero because these two equations are orthogonal.

Projections onto general subspaces: Find λ

- $\mathbf{B}^T * (\mathbf{x} - \mathbf{B}\lambda) = 0$
  - Recall that **B** is the set of basis vectors in our space. We transposed it on slide 56, $\mathbf{B}^T$ to make the matmul work.
  - Also recall that everything is set to zero because these two equations are orthogonal.
- $\mathbf{B}^T\mathbf{B}\lambda = \mathbf{B}^T\mathbf{x}$
  - We distributed to get the **“normal equation”**. This is one of the most important equations in statistics (according to Gil Strang).

Projections onto general subspaces: Find λ

- $\mathbf{B}^T * (\mathbf{x} - \mathbf{B}\lambda) = 0$
  - Recall that **B** is the set of basis vectors in our space. We transposed it on slide 56, $\mathbf{B}^T$ to make the matmul work.
  - Also recall that everything is set to zero because these two equations are orthogonal.
- $\mathbf{B}^T\mathbf{B}\lambda = \mathbf{B}^T\mathbf{x}$
  - We distributed to get the **"normal equation"**. This is one of the most important equations in statistics (according to Gil Strang).
- $\lambda = (\mathbf{B}^T\mathbf{B})^{-1}\mathbf{B}^T\mathbf{x}$
  - We solve for lambda by taking the *inverse* of the LHS. We can do this because:
    - We know B is linearly independent (basis vectors)
    - We know B^TB is square (n x m) (m x n) = n x n
    - **Aside:** B^TB is also symmetric and positive definite

Projections onto general subspaces: Find **B**λ

- The equation we just found is important because we now know the linear combination of λ's that represent the projection point in the subspace.

Projections onto general subspaces: Find **B**λ

- The equation we just found is important because we now know the linear combination of λ's that represent the projection point in the subspace.
- We want to find **B**λ now since this will give us the mu(**x**) which is the projection from the original space.
  - Recall we defined mu(**x**) = Bλ.

Projections onto general subspaces: Find **B**λ

- The equation we just found is important because we now know the linear combination of λ's that represent the projection point in the subspace.
- We want to find **B**λ now since this will give us the mu(**x**) which is the projection from the original space.
    - Recall we defined mu(**x**) = Bλ.
    - Note that we found the representation of λ in the subspace basis. But we will multiply it by the **original** basis from the **original** space to represent the projection point in that space.
        - We do this because we want the dimensions of the vector in the subspace to be equal to the dimensions in the original space. More formally, we want to represent our projection in the original space *because* we want a vector in our original space that is best approximated by what we have in our subspace.

Projections onto general subspaces: Find **B**λ

- We want to find **B**λ now since this will give us the mu(**x**) which is the projection from the original space.

Projections onto general subspaces: Find **B**λ

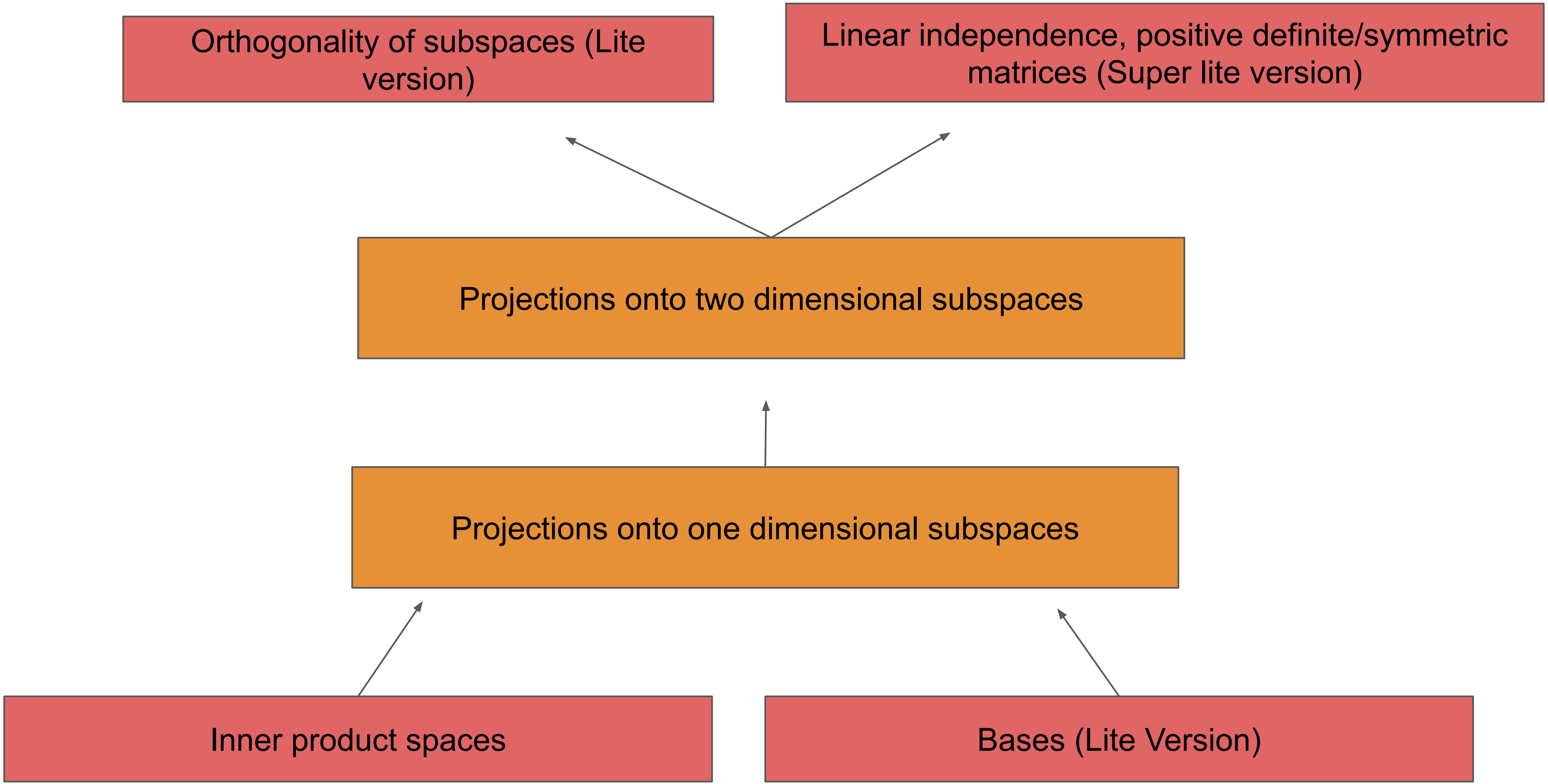
- We want to find **B**λ now since this will give us the mu(**x**) which is the projection from the original space.
- Knowing that $\lambda = (\mathbf{B}^T\mathbf{B})^{-1}\mathbf{B}^T\mathbf{x}$ we get:
  - $\mathbf{B}\lambda = \mathbf{B}(\mathbf{B}^T\mathbf{B})^{-1}\mathbf{B}^T\mathbf{x}$
  - $\mathrm{mu}(\mathbf{x}) = \mathbf{B}(\mathbf{B}^T\mathbf{B})^{-1}\mathbf{B}^T\mathbf{x}$

Projections onto general subspaces: Find **P**

- We want to find **B**λ now since this will give us the mu(**x**) which is the projection from the original space.
- Knowing that $\lambda = (\mathbf{B}^T\mathbf{B})^{-1}\mathbf{B}^T\mathbf{x}$ we get:
  - $\mathbf{B}\lambda = \mathbf{B}(\mathbf{B}^T\mathbf{B})^{-1}\mathbf{B}^T\mathbf{x}$
  - $\mathrm{mu}(\mathbf{x}) = \mathbf{B}(\mathbf{B}^T\mathbf{B})^{-1}\mathbf{B}^T\mathbf{x}$
- Now we want to find the projection matrix.
  - We can think about what's *really* happening in **B**λ.
  - $\mathbf{B}\lambda = \mathbf{B}(\mathbf{B}^T\mathbf{B})^{-1}\mathbf{B}^T\mathbf{x}$
    - In red we have the projection. This is our λ and the **B** which represents the original space.
    - We apply this projection to **x**.

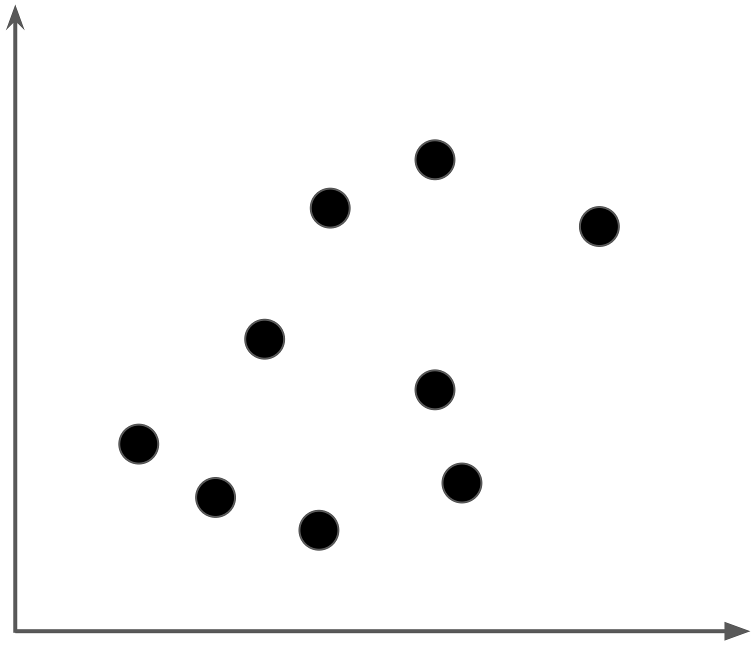
Projections onto general subspaces: Find **P**

- We want to find **B**λ now since this will give us the mu(**x**) which is the projection from the original space.
- Knowing that $\lambda = (\mathbf{B}^T\mathbf{B})^{-1}\mathbf{B}^T\mathbf{x}$ we get:
  - $\mathbf{B}\lambda = \mathbf{B}(\mathbf{B}^T\mathbf{B})^{-1}\mathbf{B}^T\mathbf{x}$
  - $\mathrm{mu}(\mathbf{x}) = \mathbf{B}(\mathbf{B}^T\mathbf{B})^{-1}\mathbf{B}^T\mathbf{x}$
- Now we want to find the projection matrix.
  - We can think about what's *really* happening in **B**λ.
  - $\mathbf{B}\lambda = \color{red}{\mathbf{B}(\mathbf{B}^T\mathbf{B})^{-1}\mathbf{B}^T}\mathbf{x}$
    - In red we have the projection. This is our λ and the **B** which is the basis for the original space.
    - We apply this projection to **x**.
    - So we define the projection matrix to be: $\mathbf{P} = \mathbf{B}(\mathbf{B}^T\mathbf{B})^{-1}\mathbf{B}$

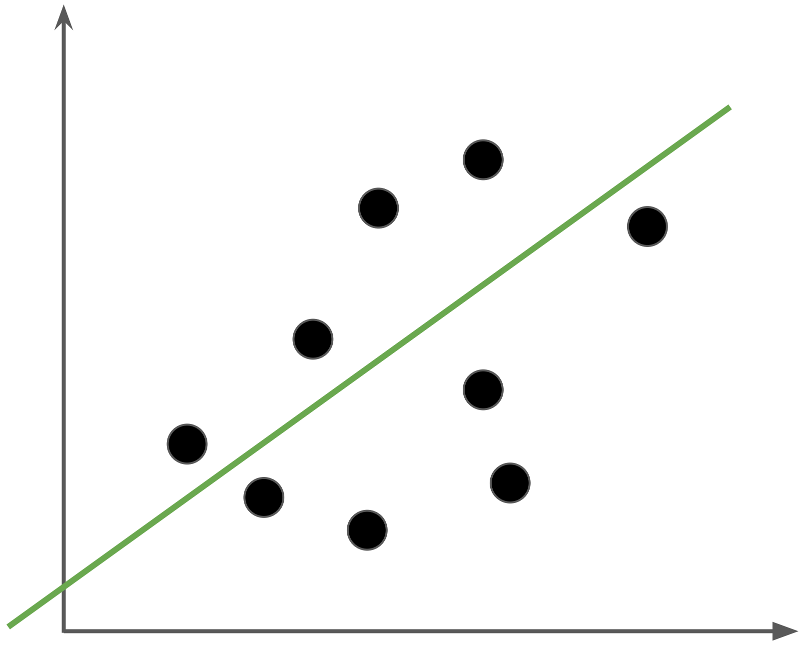

Orthogonality of subspaces (Lite version)
Linear independence, positive definite/symmetric matrices (Super lite version)
Projections onto two dimensional subspaces
Projections onto one dimensional subspaces
Inner product spaces
Bases (Lite Version)

Relation to least squares

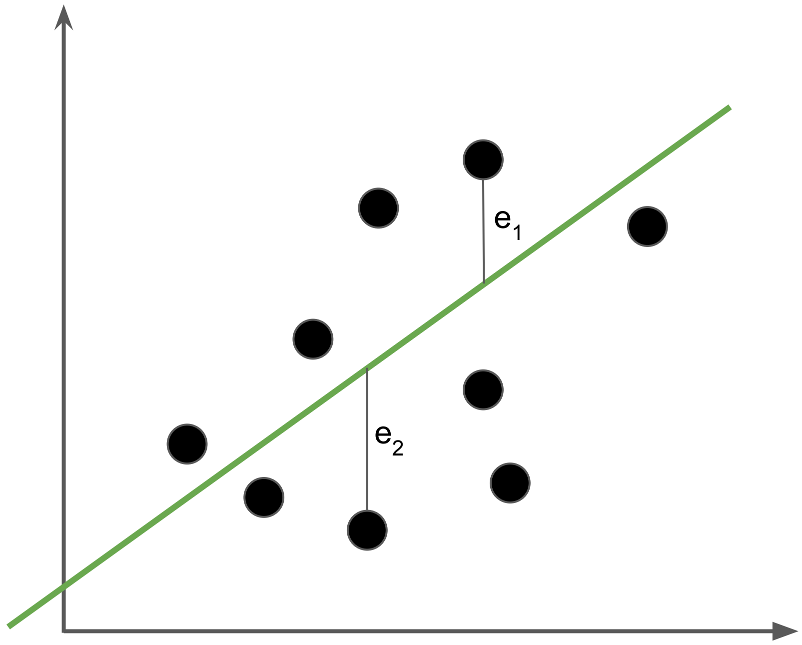
Relation to least squares



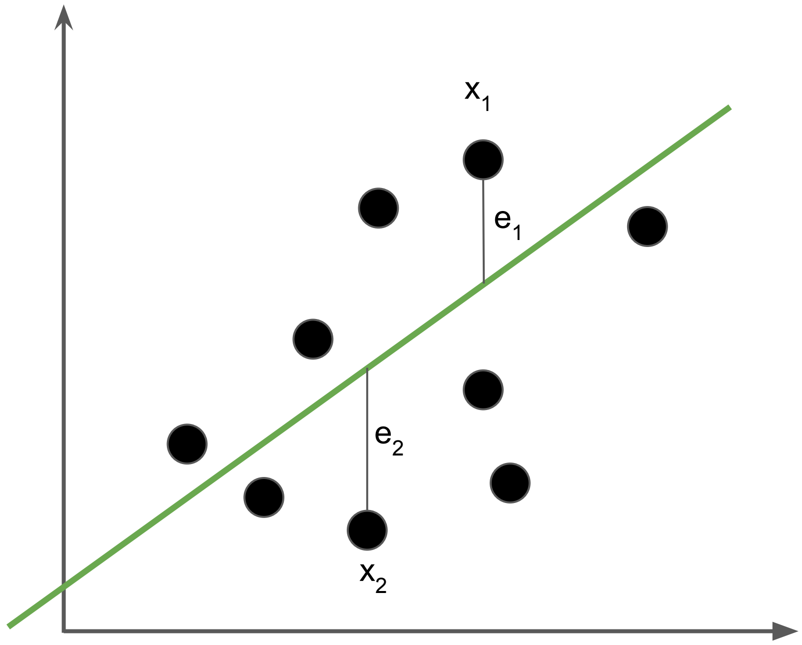
Relation to least squares



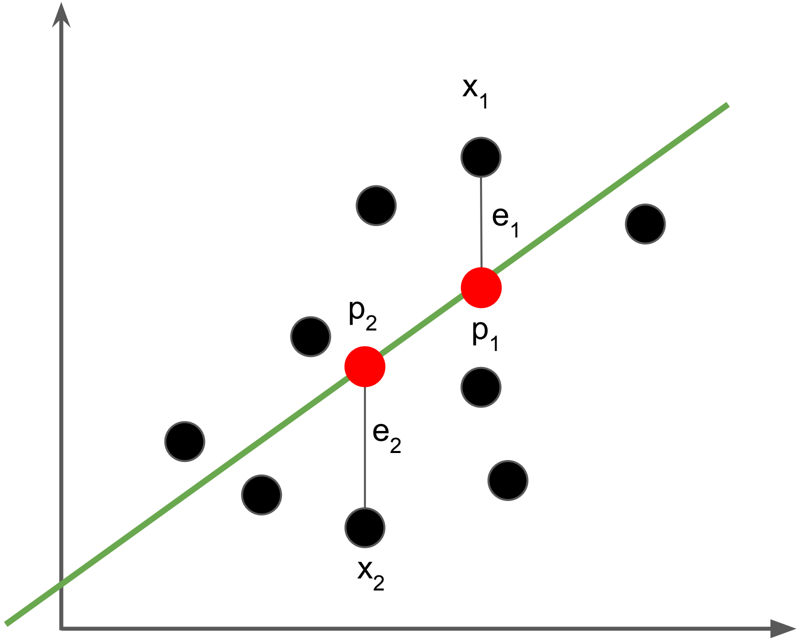
Relation to least squares



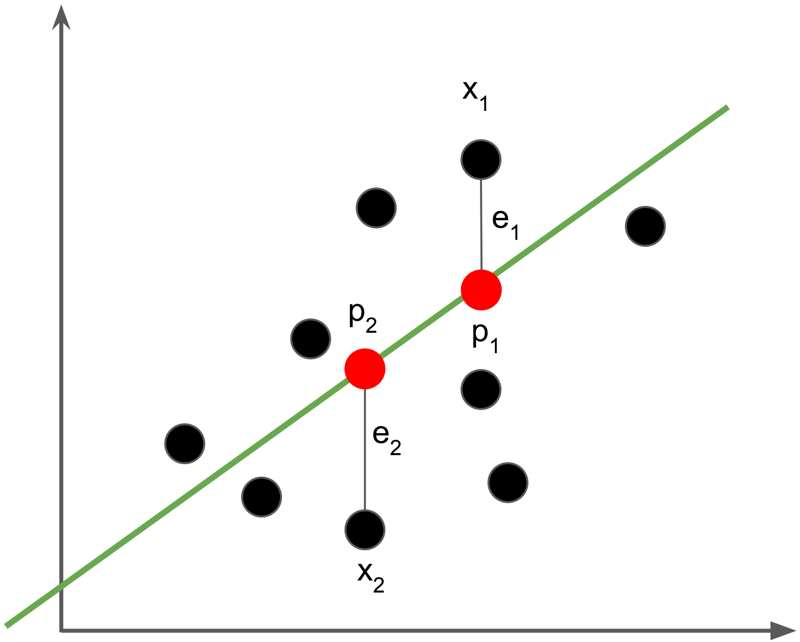
Relation to least squares

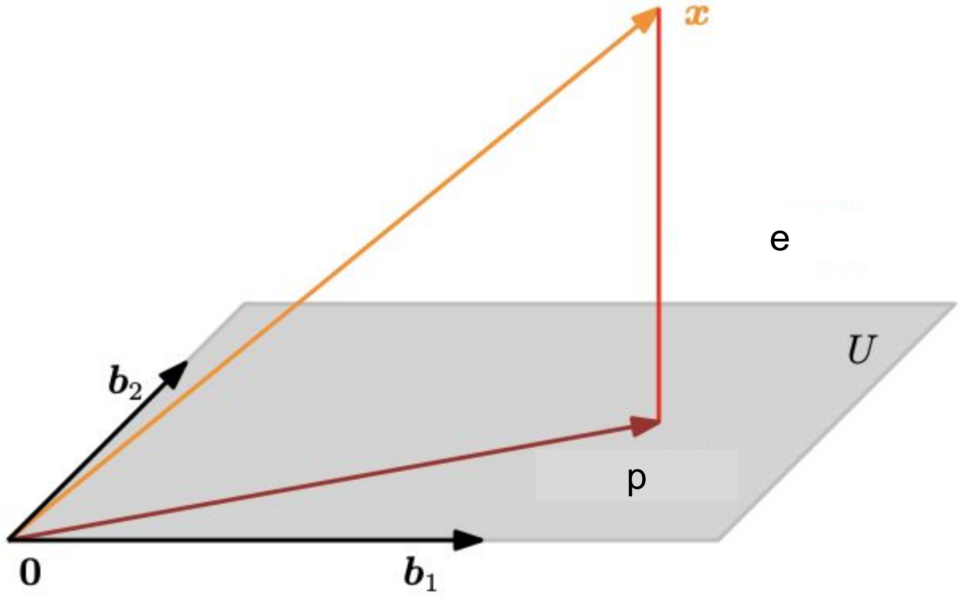


Relation to least squares



Picture of least squares and general subspace projections





Key idea: In least squares we project each point x onto the line which results in $p_1, \ldots, p_n$.

Example in code

[Deliverable 1 – TAAS]

Final fun facts

- Orthogonal projections are defined by two main properties:
  - $P^2 = P$ (Impodent matrices)
  - $P^T = P$ (Symmetric matrices)
- We can use multivariable optimization from multivariable calculus and we will get the same result as the linear algebra.
- Least squares normal equations become *much simpler* using orthonormal bases.
  - Suggestion: If you are curious look into Gram-Schmidt process.
  - Tl;dr: Take any linearly independent matrix and create an orthonormal basis by removing the projection from each component until all columns are orthogonal. Then normalize them all.
  - Orthonormal bases are the most independent bases we can get.