Q:

Nullspace in Linear Algebra. I have a matrix B = [2 3 4; 1 4 2; 4 2 8; 0 0 0]. What is the nullspace for B and its geometric interpretation?

A:

The nullspace of a matrix B, denoted $N(B)$, is the set of all vectors $\vec{x}$ that satisfy the homogeneous equation $B\vec{x} = \vec{0}$. Your matrix B is a 4×3 matrix, so it maps vectors from $\mathbb{R}^3$ to $\mathbb{R}^4$. The nullspace will be a subspace of the domain, $\mathbb{R}^3$.

Finding the Nullspace of B

To find the nullspace, we solve the system $B\vec{x} = \vec{0}$ by finding the reduced row echelon form (RREF) of the augmented matrix $[B|\vec{0}]$.

1. **Set up the augmented matrix:**

$$[B|\vec{0}] = \left[\begin{array}{ccc|c} 2 & 3 & 4 & 0 \\ 1 & 4 & 2 & 0 \\ 4 & 2 & 8 & 0 \\ 0 & 0 & 0 & 0 \end{array}\right]$$

2. **Perform row operations to get the RREF:**

   - Swap R_1 and R_2:

$$\left[\begin{array}{ccc|c} 1 & 4 & 2 & 0 \\ 2 & 3 & 4 & 0 \\ 4 & 2 & 8 & 0 \\ 0 & 0 & 0 & 0 \end{array}\right]$$

   - $R_2 = R_2 - 2R_1$
   - $R_3 = R_3 - 4R_1$

$$\left[\begin{array}{ccc|c} 1 & 4 & 2 & 0 \\ 0 & -5 & 0 & 0 \\ 0 & -14 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{array}\right]$$

   - $R_2 = R_2 / -5$:

$$\left[\begin{array}{ccc|c} 1 & 4 & 2 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & -14 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{array}\right]$$

   - $R_1 = R_1 - 4R_2$

- $R_3 = R_3 + 14R_2$

$$\left[\begin{array}{ccc|c} 1 & 0 & 2 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{array}\right]$$

3. **Translate the RREF back into equations:** Let $\vec{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$. The RREF gives us:
   - $x_1 + 2x_3 = 0 \implies \mathbf{x_1 = -2x_3}$
   - $x_2 = 0 \implies \mathbf{x_2 = 0}$
   - x_3 is a **free variable** (since its column has no pivot).
4. **Write the general solution:** We can express the solution vector in terms of the free variable, x_3. Let $x_3 = t$, where t is any scalar.

$$\vec{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} -2t \\ 0 \\ t \end{bmatrix} = t \begin{bmatrix} -2 \\ 0 \\ 1 \end{bmatrix}$$

   The nullspace $N(B)$ is the set of all such vectors. It is the span of the single basis vector:

$$N(B) = \text{span}\left\{ \begin{bmatrix} -2 \\ 0 \\ 1 \end{bmatrix} \right\}$$

Geometric Interpretation

The geometric interpretation of the nullspace $N(B)$ relates to what the transformation B "squashes" or "collapses" to the zero vector.

- **Subspace of $\mathbb{R}^3$:** The nullspace $N(B)$ is a subspace of the input space, $\mathbb{R}^3$.
- **Dimension (Nullity):** The basis for the nullspace consists of **one** vector ($\langle -2, 0, 1 \rangle$). Therefore, the dimension of the nullspace (known as the **nullity**) is 1.
- **Geometric Shape:** A 1-dimensional subspace of $\mathbb{R}^3$ is a **line that passes through the origin**.

In short, the nullspace of your matrix B is the line in 3D space that runs through the origin $(0, 0, 0)$ in the direction of the vector $\langle -2, 0, 1 \rangle$. Any vector $\vec{x}$ that lies on this specific line is mapped by the transformation B to the zero vector $\vec{0} = \langle 0, 0, 0, 0 \rangle$ in $\mathbb{R}^4$.

This video explains the null space concept and its geometric meaning. http://googleusercontent.com/youtube_content/0

Q:

When I compute the nullspace with Julia I get (-0.9, 0,0.45) which has a norm of 1. Yor vector has a norm of 2.236. Is there a particular reason why you choose this vector?

A:

You're right, and that's an excellent observation!

There is **no single "correct" vector** for the nullspace. The nullspace is a **subspace**—in this case, a 1-dimensional line. Any vector that lies on that line (except the zero vector) is a perfectly valid basis vector for it.

Both of our vectors describe the exact same line:

- **My Vector:** $\vec{v}_1 = \begin{bmatrix} -2 \\ 0 \\ 1 \end{bmatrix}$
- **Julia's Vector (approx.):** $\vec{v}_2 \approx \begin{bmatrix} -0.8944 \\ 0 \\ 0.4472 \end{bmatrix}$ *(Note: The vector you provided, $\langle -0.9, 0, 0.45 \rangle$, is likely a rounded version of the true unit vector, which is $\langle -2/\sqrt{5}, 0, 1/\sqrt{5} \rangle$)*

If you take my vector and multiply it by ≈ 0.4472 (which is $1/\sqrt{5}$), you get Julia's vector.

$$\begin{bmatrix} -2 \\ 0 \\ 1 \end{bmatrix} \times \frac{1}{\sqrt{5}} = \begin{bmatrix} -2/\sqrt{5} \\ 0 \\ 1/\sqrt{5} \end{bmatrix} \approx \begin{bmatrix} -0.8944 \\ 0 \\ 0.4472 \end{bmatrix}$$

Since they are scalar multiples of each other, they point in the same (or exactly opposite) direction and thus define the same line.

Why the Difference?

1. **My Choice (Manual Calculation):** When solving by hand, we set the free variable $x_3 = t$. The simplest and cleanest choice is to set $t = 1$. This gives us the vector $\langle -2, 0, 1 \rangle$. It's the standard way to do it by hand because it avoids fractions or square roots and gives a simple integer-based vector.

2. **Julia's Choice (Numerical Computation):** Numerical software like Julia, MATLAB, or Python (NumPy) usually returns an **orthonormal basis** for the nullspace. This means the basis vector(s) it provides are "normalized" to have a length (norm) of 1.

   - The norm of my vector $\langle -2, 0, 1 \rangle$ is $\sqrt{(-2)^2 + 0^2 + 1^2} = \sqrt{4+1} = \sqrt{5} \approx 2.236$.

- To normalize it, Julia divides the vector by its norm: $\frac{1}{\sqrt{5}}\langle -2, 0, 1\rangle$.
- This results in the unit vector $\langle -2/\sqrt{5}, 0, 1/\sqrt{5}\rangle$, which is what you saw.

In summary: Both answers are 100% correct. I chose the simplest integer vector. Julia chose the standard unit vector. They both describe the exact same nullspace.

Q:

Is it a general truth, that the nullspace is a subspace that passes through the origin?

A:

Yes, absolutely. That is a fundamental truth.

The nullspace is **always** a subspace, and a defining property of **any** subspace is that it **must** contain the origin (the zero vector, $\vec{0}$).

Your geometric intuition is spot on. Here's the simple proof:

1. The **definition of the nullspace** $N(A)$ is the set of all vectors $\vec{x}$ that solve the equation $A\vec{x} = \vec{0}$.
2. The **definition of a subspace** requires that the zero vector $\vec{0}$ must be in the set.
3. Let's test the zero vector: If we plug $\vec{x} = \vec{0}$ into the nullspace equation, we get:

   $$A\vec{0} = \vec{0}$$

   This is **always true** for any matrix A (a matrix multiplied by the zero vector always yields the zero vector).

Because the zero vector *always* solves the equation, the nullspace *always* contains the origin. Geometrically, this means any line, plane, or higher-dimensional space representing a nullspace is "anchored" at the origin.

Q:

If I have a vector b1 = [3,3,3,0] and want to find a solution to the equation Bx = b1. 1. How do I solve that in Julia 2. The solution I get, I consider this will lie either inside the vector subspace defined by the columns of B, or outside this space. Is there a way to see if it is inside or outside? Please clarify.

A:

Great questions. Here's how to tackle both.

---

1. How to Solve in Julia

To solve $B\vec{x} = \vec{b}_1$ in Julia, you use the backslash operator (\). This operator is smart:

- If an exact, unique solution exists, it will find it.
- If no exact solution exists (an "inconsistent" system), it will find the **least-squares solution**. This is the vector $\vec{x}$ that makes the result $B\vec{x}$ as "close as possible" to $\vec{b}_1$.

Here is the Julia code:

```julia
using LinearAlgebra

# Your matrix B
B = [ 2 3 4;
      1 4 2;
      4 2 8;
      0 0 0]

# Your vector b1
b1 = [3; 3; 3; 0]

# Solve the system Bx = b1
x = B \ b1

println("The solution vector x is:")
println(x)
```

---

2. Clarification: Is the Solution "Inside or Outside"?

This is a fantastic question, and your intuition is pointing to a critical concept. There's a small (but very important) clarification to make.

Your question: "Is the solution $\vec{x}$ inside or outside the column space of B?"

The Clarification: The solution vector $\vec{x}$ and the column space of B live in different "universes."

- The **solution** $\vec{x}$ is an *input* vector. Since B is a 4×3 matrix, $\vec{x}$ must be a 3×1 vector. It lives in the **domain**, $\mathbb{R}^3$.
- The **column space of** B is the set of all *possible outputs*. Since B is 4×3, its columns are 4×1 vectors. The column space is a subspace of the **codomain**, $\mathbb{R}^4$.

Therefore, $\vec{x}$ (which is in $\mathbb{R}^3$) can never be *in* the column space of B (which is in $\mathbb{R}^4$).

The Real Question (and how to answer it)

I think the question you're *really* asking is:

"Is the vector $\vec{b}_1$ inside the column space of B?"

This is the most important question you can ask about a linear system! Here's what it means:

- The **column space** is the "span" of B's columns—it's the set of all vectors that can be *created* by B.
- If $\vec{b}_1$ is **inside** the column space, it means an exact solution exists. The system is called **consistent**.
- If $\vec{b}_1$ is **outside** the column space, it means it's an "impossible" target for B. No exact solution exists, and the system is called **inconsistent**.

How to Check if $\vec{b}_1$ is in the Column Space

The easiest way to check this computationally is to find the least-squares solution $\vec{x}$ and then see how much "error" there is. This error is called the **residual**.

1. Solve for $\vec{x}$: `x = B \ b1`
2. Calculate the result: `p = B * x` (This `p` is the projection of $\vec{b}_1$ onto the column space of B).
3. Find the residual: `r = b1 - p`

If the residual `r` is the zero vector (or extremely close to it), then $\vec{b}_1$ was **inside** the column space. If the residual `r` is *not* zero, then $\vec{b}_1$ was **outside** the column space.

Here's the Julia code to check:

```julia
using LinearAlgebra

# (B and b1 defined as before)
B = [ 2 3 4; 1 4 2; 4 2 8; 0 0 0]
b1 = [3; 3; 3; 0]

# 1. Find the least-squares solution
x = B \ b1

# 2. Calculate the projection (the closest possible vector to b1)
p = B * x

# 3. Calculate the residual (the error)
residual = b1 - p
```

```
println("b1 vector: ", b1)
println("Closest vector (projection): ", p)
println("Residual (error): ", residual)

# 4. Check the norm (length) of the residual
residual_norm = norm(residual)
println("Norm of residual: ", residual_norm)

if residual_norm < 1e-10  # Check if it's practically zero
    println("Conclusion: b1 IS in the column space. An exact solution exists.")
else
    println("Conclusion: b1 is NOT in the column space. The solution is a least-squares appr
end
```

When you run this, you'll see that the residual is *not* zero. This proves that for your specific problem, the vector $\vec{b}_1 = [3, 3, 3, 0]$ is **outside** the column space of B, and the $\vec{x}$ Julia finds is the best possible approximation.

This video explains how to check if a system of equations has a solution, which is directly related to whether the vector $\vec{b}$ is in the column space. Watch: Checking for Consistency of Equations http://googleusercontent.com/youtube_content/1

Q:

The column space. Is it so that the column space for a matrix always contain the origin? If it does not, how will such a matrix differ from one that does?

A:

Yes, absolutely. The column space of **any** matrix **always** contains the origin (the zero vector, $\vec{0}$).

This isn't just a coincidence; it's a fundamental requirement for *any* vector subspace, and the column space is a perfect example of a subspace.

Why the Origin is Always Included

The column space is defined as the set of **all possible linear combinations** of the matrix's column vectors.

Let's say your matrix B has columns $\vec{c}_1$, $\vec{c}_2$, and $\vec{c}_3$. A vector $\vec{v}$ is in the column space if it can be written as:

$$\vec{v} = x_1\vec{c}_1 + x_2\vec{c}_2 + x_3\vec{c}_3$$

where x_1, x_2, and x_3 are any scalar weights.

To get the zero vector $\vec{0}$, we can simply choose the "trivial" linear combination where all weights are zero:

$$\vec{0} = (0)\vec{c}_1 + (0)\vec{c}_2 + (0)\vec{c}_3$$

Since this is always a valid choice, the zero vector $\vec{0}$ is guaranteed to be in the set.

What This Means for Your Second Question

Your second question is, "If it does not, how will such a matrix differ from one that does?"

Based on the proof above, this is a trick question! A matrix whose column space does *not* contain the origin **cannot exist**.

The very property that distinguishes a **vector subspace** (like a nullspace or a column space) from just any random set of vectors (like a line or a plane) is that the subspace **must** pass through the origin.

- A line $y = 2x$ in $\mathbb{R}^2$ **is** a subspace. It passes through $(0, 0)$.
- A line $y = 2x + 1$ in $\mathbb{R}^2$ **is not** a subspace. It's a "shifted" subspace (an affine space), and it can't be the column space of any matrix.

So, there is no difference to describe, because all matrices fall into the first category.

Q:

Is it so that the vector subspace of a matrix expand into infinity? If not, how is the boundary given?

A:

Yes, your intuition is correct. A vector subspace (like a nullspace or column space) will "expand to infinity" in all its directions.

The more technical term for this is that subspaces are **unbounded**.

They do not have a "boundary" in the way a shape like a circle or a square does. Here is the simple reason why.

The Key Property: Closure Under Scalar Multiplication

A defining rule of a vector subspace is that it must be **"closed under scalar multiplication."**

This means that if you take *any* vector $\vec{v}$ that is in the subspace, then $c \cdot \vec{v}$ must *also* be in the subspace for *any* scalar c.

Let's see what that implies:

1. Let's use your nullspace basis vector: $\vec{v} = \begin{bmatrix} -2 \\ 0 \\ 1 \end{bmatrix}$. This vector is in the nullspace.
2. Because it's a subspace, $10 \cdot \vec{v} = \begin{bmatrix} -20 \\ 0 \\ 10 \end{bmatrix}$ must also be in it.
3. $1,000,000 \cdot \vec{v} = \begin{bmatrix} -2,000,000 \\ 0 \\ 1,000,000 \end{bmatrix}$ must *also* be in it.
4. There is no limit. You can choose any scalar c, no matter how large, and the resulting vector is still part of the subspace.

This forces the subspace to stretch out to infinity.

Why There Is No Boundary

Because they are unbounded, these subspaces do not have a "boundary" (unless it's the special case described below).

- A 1-dimensional subspace (like your nullspace) is not a **line segment** (which has two endpoints, or "boundaries"). It is an **infinite line** that passes through the origin.
- A 2-dimensional subspace is not a **finite sheet of paper** (which has edges, or "boundaries"). It is an **infinite plane** that passes through the origin.

The subspace includes every point along that infinite line or plane, from the origin all the way out to infinity in all of its directions.

The Only Exception: The Trivial Subspace

There is exactly *one* subspace that does not "expand to infinity."

It's called the **trivial subspace**, and it contains **only the zero vector** $\{\vec{0}\}$.

This is the "smallest" possible subspace. You can think of it as a single point at the origin. It is "bounded" (it doesn't go anywhere), but it's the only one.

Any subspace that contains even *one* non-zero vector must, by the rule of scalar multiplication, be unbounded.

Q

So based on what you have said, I conclude that the nullspace is a part of the column space?

A:

No, that's a very common point of confusion, but the nullspace is **not** part of the column space.

They are fundamentally separate: they live in completely different "universes" or dimensions.

The "Inputs" vs. The "Outputs"

The easiest way to think about this is:

- **Nullspace (The Inputs):** The nullspace is a subspace of the **domain** (the *input* space). For your 4×3 matrix B, the inputs $\vec{x}$ are in $\mathbb{R}^3$. Therefore, your nullspace is a line *inside* $\mathbb{R}^3$.
- **Column Space (The Outputs):** The column space is a subspace of the **codomain** (the *output* space). Your matrix B has 4-row columns, so the outputs $\vec{b}$ are in $\mathbb{R}^4$. The column space is a subspace *inside* $\mathbb{R}^4$.

A line in $\mathbb{R}^3$ (a 3D vector) can never be "part of" a subspace in $\mathbb{R}^4$ (a 4D vector).

Clarifying The "Origin" Connection

You're likely making this conclusion because I confirmed that **both** contain the origin. This is the key:

- The nullspace contains the origin of the **input space** ($\vec{0}_3 = \langle 0, 0, 0 \rangle$).
- The column space contains the origin of the **output space** ($\vec{0}_4 = \langle 0, 0, 0, 0 \rangle$).

They are two different zero vectors in two different spaces. The only thing they truly "share" is the *concept* of being the origin.

Summary: * **Nullspace:** All *input* vectors $\vec{x}$ that B squashes to zero. (Lives in $\mathbb{R}^3$). * **Column Space:** All *output* vectors $\vec{b}$ that B can create. (Lives in $\mathbb{R}^4$).

Q:

So, the input vectors must be in R3 because the rank of B is 3? If B had had rank 4, the input vectors should have been in R4?

A:

No, that's a crucial distinction to make. The dimension of the input space has **nothing to do with the rank**.

The dimension of the input space (the "universe" $\vec{x}$ lives in) is determined *only* by the number of **columns** of the matrix.

The Rules are Simpler Than You Think

Let's look at your matrix B again. It is a 4×3 matrix ($m = 4$ rows, $n = 3$ columns).

To perform the multiplication $B\vec{x}$, the "shapes" must align:

$$\begin{array}{cccc} B & \vec{x} & & \vec{b} \\ \\ (4 \times \mathbf{3}) & (\mathbf{3} \times 1) & = & (4 \times 1) \end{array}$$

1. **Input Space (Domain):** The number of **columns** ($n = 3$) dictates that the input vector $\vec{x}$ *must* be in $\mathbb{R}^3$. This is a fixed rule. It doesn't matter what the rank is.
2. **Output Space (Codomain):** The number of **rows** ($m = 4$) dictates that the output vector $\vec{b}$ *must* be in $\mathbb{R}^4$.

So, What is Rank?

The **rank** tells you the dimension of the **column space**—that is, the *actual* space of all possible outputs.

Let's correct one small detail: when we row-reduced your matrix B, we got:

$$\text{RREF}(B) = \begin{bmatrix} 1 & 0 & 2 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

This matrix has only **2 pivots** (the leading 1s). Therefore:

The rank of your matrix B is 2, not 3.

Let's Put It All Together

Here is the complete picture for *your* matrix B:

- It has **3 columns**, so its input space (domain) is $\mathbb{R}^3$.
- It has **4 rows**, so its output space (codomain) is $\mathbb{R}^4$.
- It has **rank 2**, which means its column space (the set of all *actual* outputs) is a 2-dimensional plane that lives inside the 4-dimensional output space $\mathbb{R}^4$.
- It has **nullity 1** (since rank + nullity = columns, so $2 + 1 = 3$), which means its nullspace is a 1-dimensional line inside the 3-dimensional input space $\mathbb{R}^3$.

To answer your second question: "If B had had rank 4, the input vectors should have been in R4?"

- This is an impossible situation. Your matrix B is 4×3. The rank can *never* be larger than the number of columns or rows. The max possible rank for B is 3.
- But, if you had a *different* matrix A that was, for example, 5×4 (4 columns), its input space would be $\mathbb{R}^4$. If that matrix A had rank 4, it would mean its column space is a 4D subspace inside the 5D output space $\mathbb{R}^5$.

Q:

Some questions about Least Squares method and vector spaces. If I have a 4x4 matrix A with rank 3, I suppose the output space for A is a subspace of R4, which is a hyperplane(?) or hypervolume in R4 (please clarify the correct use of terms). If on the other hand, the rank of A had been 4, the the output space (or solution space perhaps) would be the whole R4. Am I correct?

A:

Yes, you are perfectly correct on both points. Your understanding is solid.

Here's a clear breakdown of the terminology.

Scenario 1: A is 4×4 with Rank 3

You're right, the output space (the **column space**) is a subspace of $\mathbb{R}^4$.

Since the rank is 3, the column space is a **3-dimensional subspace** of $\mathbb{R}^4$.

The correct geometric term for this is a **hyperplane**.

- A **hyperplane** is the general term for a subspace that has exactly *one dimension less* than the space it lives in (i.e., dimension $n - 1$).

- A 2D plane is a "hyperplane" inside $\mathbb{R}^3$.
- A 3D space is a "hyperplane" inside $\mathbb{R}^4$.

A "hypervolume" would imply a 4-dimensional volume, but your rank 3 column space is "flat" in comparison to the 4D space it's in.

Scenario 2: A is 4×4 with Rank 4

You are again 100% correct.

If the rank is 4, the column space is a **4-dimensional subspace** of $\mathbb{R}^4$.

The *only* 4-dimensional subspace of $\mathbb{R}^4$ is $\mathbb{R}^4$ itself.

This means the matrix A can reach *any* vector $\vec{b}$ in $\mathbb{R}^4$. The transformation covers the entire output space. Such a matrix is **invertible**, and the system $A\vec{x} = \vec{b}$ has a unique, exact solution for every possible $\vec{b}$.

(A small note on terms: "output space" or **"column space"** is the most precise term here. "Solution space" is usually reserved for the set of $\vec{x}$ vectors that solve a *specific* equation, which is a different concept.)

Quick Terminology Guide (for subspaces in $\mathbb{R}^4$)

- **Dimension 0:** A **point** (just the origin).
- **Dimension 1:** A **line** (through the origin).
- **Dimension 2:** A **plane** (through the origin).
- **Dimension 3:** A **hyperplane** (through the origin).
- **Dimension 4:** The **entire space** $\mathbb{R}^4$.

Q:

More on least squares method for solving linear equations. My understanding is that we use this method when we have over-determined systems, i.e. we have "tall" matrices, e.g. a 12x4 matrix T. This matrix lives in R12, but we seek a solution for the equation Tx = b, where x and b are R4 vectors. Is it a proper description to say that by solving this equation we are doing dimension reduction?

A:

That's a great question, and you're very close, but "dimension reduction" isn't the most accurate term here. The better term is **projection**.

The process is less about *reducing* the dimension of the space and more about *finding the best approximation* within a lower-dimensional subspace.

Let's break it down.

A Quick Correction on Dimensions

First, let's clarify the "universes" these vectors live in, using your 12×4 matrix T:

- **Matrix T:** 12×4
- **Input $\vec{x}$:** To multiply with T, $\vec{x}$ must be 4×1. It lives in $\mathbb{R}^4$ (the domain).
- **Output $\vec{b}$:** The result $T\vec{x}$ is 12×1. So your target vector $\vec{b}$ must also live in $\mathbb{R}^{12}$ (the codomain).

So, the equation $T\vec{x} = \vec{b}$ is trying to find a recipe $\vec{x}$ (in $\mathbb{R}^4$) that combines the 4 columns of T to create a vector $\vec{b}$ (in $\mathbb{R}^{12}$).

Projection: What's Really Happening

As you said, this is an **overdetermined system**. You have 12 equations (the rows) but only 4 "knobs to turn" (the elements of $\vec{x}$).

Here's the problem: 1. **Column Space:** The 4 columns of T define a subspace *inside* $\mathbb{R}^{12}$. This subspace (the column space) is at most 4-dimensional. 2. **The Target:** Your 12D target vector $\vec{b}$ is almost certainly *not* inside that 4D subspace. It's just some random point in $\mathbb{R}^{12}$. 3. **The "Error":** Because $\vec{b}$ is not in the column space, there is no $\vec{x}$ that can perfectly solve $T\vec{x} = \vec{b}$. An exact solution is impossible.

The least-squares method says: "If I can't find an $\vec{x}$ that *hits* $\vec{b}$, I'll find an $\vec{x}$ that comes *as close as possible*."

This "closest possible" vector is the **orthogonal projection** of $\vec{b}$ onto the column space of T. We can call this projection $\vec{p}$.

- You are **not** doing dimension reduction.
- You are **projecting** a 12-dimensional vector ($\vec{b}$) onto a 4-dimensional subspace (the column space of T) to find its "shadow" ($\vec{p}$).
- The least-squares method finds the $\vec{x}$ (the 4D "recipe") that builds this 12D "shadow" $\vec{p}$.

Projection vs. Dimension Reduction

While related, the terms mean different things:

- **Projection (Least Squares):** This is about *solving an unsolvable system*. It finds the best approximation of a single target vector ($\vec{b}$) within a *pre-defined* subspace (Col(T)). The goal is to minimize the error $||\vec{b} - T\vec{x}||^2$.
- **Dimension Reduction (like PCA):** This is about *data compression*. It takes a whole *cloud* of data points (not just one $\vec{b}$) and finds a *new* lower-dimensional subspace that best captures the *variance* of the entire cloud. The goal is to represent the data with less information.

So, while you are using a 4D vector $\vec{x}$ to describe a 12D situation $\vec{b}$, the proper term for the *method* is **projection**.

Q:

In your Dimension Reduction explanation you explain the vector b as a data point. I would also consider each row of T as a data point, where T contains the independent variables, and b is the dependent variables. Please comment.

A:

You are absolutely right. That is a perfect and, in fact, the standard way to interpret this problem in statistics and machine learning.

My previous explanation came from a "pure" linear algebra perspective (the "column picture"). Your description is the "data science" perspective (the "row picture").

Both are correct and describe the same problem! Let's compare.

---

1. Your View: The "Row Picture" (Data & Fitting)

This is the most common way to think about a linear regression.

- **Matrix T (12x4):** This is your **feature matrix**. Each of the 12 **rows** is one **data point** (or "observation"). Each of the 4 columns is a **feature** (an independent variable, like `age`, `square_footage`, etc.).
- **Vector $\vec{b}$ (12x1):** This is your **target vector**. Each entry b_i is the known **outcome** (the dependent variable, like `price`) for the i-th data point (the i-th row of T).
- **Vector $\vec{x}$ (4x1):** This is the **model** you're solving for. It contains the **coefficients** (or "weights") that tell you how much each feature matters.

In this view, the equation $T\vec{x} = \vec{b}$ represents 12 separate equations, one for each row:

$$(\text{row } i \text{ of } T) \cdot \vec{x} \approx b_i$$

The goal of least squares is to find the *single* set of coefficients $\vec{x}$ that defines a 4D "line" (a hyperplane) that **best fits all 12 data points** by minimizing the total error.

---

2. My View: The "Column Picture" (Geometry & Projection)

This is the classic geometric interpretation from linear algebra.

- **Vector $\vec{b}$ (12x1):** This is viewed as a **single target point** in a 12-dimensional space ($\mathbb{R}^{12}$).
- **Columns of T (four 12x1 vectors):** These four vectors define a **subspace**—a "flat" 4D plane—that lives inside the larger 12D space. This plane is the **column space**, which represents *all possible outputs* the matrix T can create.
- **Vector $\vec{x}$ (4x1):** This is the **"recipe"**. It tells you how to mix the four columns to get an output.

In this view, the problem is that the target point $\vec{b}$ is "floating" in 12D space, and it's *not* on the 4D plane. The goal of least squares is to find the **projection** of $\vec{b}$ onto that plane, finding the "shadow" $\vec{p}$ that is the closest possible point to $\vec{b}$. The vector $\vec{x}$ is the recipe for building that shadow.

Conclusion: How They Relate

You've hit on the key point:

When we talk about **Dimension Reduction (like PCA)**, we are 100% in your "Row Picture." We take the *rows* of T (the data points) and try to find a lower-dimensional subspace (e.g., 2D) that best captures their *variance*, completely **ignoring $\vec{b}$**.

When we talk about **Least Squares (Projection)**, we are fitting a model. Your "Row Picture" (fitting a line to data points) is the statistical view. My "Column Picture" (projecting a vector onto a subspace) is the geometric view. They are two sides of the same coin—they give the exact same numerical answer for $\vec{x}$.

Q:

Some clarification on the image of T. You say that the column space of T is all possible outputs of T, i.e. the whole of R12. My understanding is that Im(T) could be a subspace of R4. Please clarify.

A:

You are right to question that. Let's clarify, because you've spotted the two most common points of confusion: 1. Is the column space the *whole* output space? (No) 2. Is the image in $\mathbb{R}^4$? (No, it's in $\mathbb{R}^{12}$)

This is a mix-up between a **space** and its **dimension**.

Here is the correct breakdown for your 12×4 matrix T.

---

1. Codomain vs. Image (Column Space)

- **Codomain (The "Target Space"):** This is the "universe" that all outputs must live in. It is determined by the number of **rows**.
  - Your matrix T has 12 rows.
  - Therefore, any output $T\vec{x}$ is a 12-component vector.
  - The codomain is $\mathbb{R}^{12}$.
- **Image / Column Space (The "Actual Outputs"):** This is the set of all outputs you can *actually* create. It is the span of the **columns**.
  - Your matrix T has only **4 columns**.
  - The image is the set of all linear combinations of these 4 column vectors.
  - The span of 4 vectors can be, at most, a **4-dimensional** subspace.
  - This 4D subspace lives *inside* the 12D codomain, $\mathbb{R}^{12}$.

So, the image is **not** the whole of $\mathbb{R}^{12}$. It's just a tiny (at most 4D) subspace *within* $\mathbb{R}^{12}$.

---

2. Domain vs. Codomain (Inputs vs. Outputs)

You suggested the image could be a subspace of $\mathbb{R}^4$. This is swapping the input and output spaces.

- **Domain (The "Input Space"):** Determined by the number of **columns**.
  - T has 4 columns, so the input vector $\vec{x}$ must be in $\mathbb{R}^4$.
  - The **nullspace** is a subspace of the domain, $\mathbb{R}^4$.
- **Codomain (The "Output Space"):** Determined by the number of **rows**.
  - T has 12 rows, so the output vector $\vec{b}$ is in $\mathbb{R}^{12}$.
  - The **image (column space)** is a subspace of the codomain, $\mathbb{R}^{12}$.

Summary Analogy

Think of your 12×4 matrix T as a machine in a factory:

- **Input / Domain ($\mathbb{R}^4$):** The machine has **4 control knobs**. The set of all possible knob settings is $\mathbb{R}^4$.

- **Output / Codomain ($\mathbb{R}^{12}$):** The machine has a control panel with **12 display lights**. The set of all possible light patterns is $\mathbb{R}^{12}$.
- **Image / Column Space (Subspace of $\mathbb{R}^{12}$):** Because you only have 4 knobs, you can't create every possible pattern on the 12 lights. The set of patterns you *can* create is the image. It's a small (at most 4D) subspace of all possible 12D patterns.

Q:

So, you are actually saying that Column Space and Image is the same concept?

A:

Yes, they are **exactly the same concept**.

They're just two different names for the same set of vectors, coming from two different perspectives:

1. **Column Space:** This is the "construction" view from linear algebra. It's the set of all vectors you can **build** by taking linear combinations of the matrix's **columns**.
2. **Image:** This is the "function" view from transformations. It's the set of all possible **output** vectors $T\vec{x}$ (the **image** of the function T).

Since any output $T\vec{x}$ is just a linear combination of the columns of T, the two sets are identical.

Q:

Is there a general term for a subspace that has a dimension more more than one less than the space in which it lives? E.g a 4-dimensional subspace in R15?

A:

No, there isn't a common, single geometric term for that general situation.

The term **hyperplane** is a special name we give *only* to subspaces with a dimension of $n-1$ (one less than the ambient space).

The concept you're describing is called **codimension**.

- **Codimension** = dim(Ambient Space) − dim(Subspace)

So, we have these specific terms:

- **Codimension 1:** A **hyperplane** (e.g., a 14-dimensional subspace in $\mathbb{R}^{15}$).
- **Codimension 2 or more:** There is no special name.

In your example of a 4-dimensional subspace in $\mathbb{R}^{15}$, its codimension is $15-4=11$.

We would simply call it “a 4-dimensional subspace” or “a subspace of codimension 11.”