# LeanSearch

Find theorems in Mathlib4 using natural language query

Query   Name or description of the theorem or definition you are looking for

Affine Space Characterization Theorem: An affine space is completely and uniquely defined by two fundamental operations: the translation operation that moves points along vectors, and the difference operation that measures the vector between two points. These operations must satisfy the torsor axioms, which are a set of three fundamental conditions: the translation of a point by the zero vector leaves the point

Number of results 50

Clear    Query Augmentation    Search

Tip: **Query Augmentation** augments your query to increase the chance to find relevant results.

**vadd_right_mem_affineSpan_pair**    theorem

```
{p₁ p₂ : P} {v : V} : v +ᵥ p₂ ∈ line[k, p₁, p₂] ↔ ∃ r : k, r • (p₁ -ᵥ p₂) = v
```

▶ **Characterization of Points in Affine Span via Right Translation:** $v + p_2 \in \text{affineSpan}\{p_1, p_2\} \leftrightarrow \exists r \in k, v = r \cdot (p_1 - p_2)$

Doc    Doc-Next-Gen (beta)    Similar

**vadd_left_mem_affineSpan_pair**    theorem

```
{p₁ p₂ : P} {v : V} : v +ᵥ p₁ ∈
line[k, p₁, p₂] ↔ ∃ r : k, r •
(p₂ -ᵥ p₁) = v
```

▶ Characterization of Affine Span Membership via Scaled Difference: $v + p_1 \in \mathrm{affineSpan}_k\{p_1, p_2\} \leftrightarrow \exists r \in k, v = r \cdot (p_2 - p_1)$

Doc Doc-Next-Gen (beta) Similar

**mem_affineSpan_iff_eq_affineCombination** theorem

```
[Nontrivial k] {p1 : P} {p : ι →
P} :
  p1 ∈ affineSpan k (Set.range p) ↔
∃ (s : Finset ι) (w : ι → k), ∑ i ∈
s, w i = 1 ∧ p1 =
s.affineCombination k p w
```

▶ Characterization of Points in Affine Span via Affine Combinations

Doc Doc-Next-Gen (beta) Similar

**mem_affineSpan_iff_eq_weightedVSubOfPoint_vadd** the

```
[Nontrivial k] (p : ι → P) (j : ι)
(q : P) :
  q ∈ affineSpan k (Set.range p) ↔
∃ (s : Finset ι) (w : ι → k), q =
s.weightedVSubOfPoint p (p j) w +ᵥ
p j
```

▶ Characterization of Affine Span Membership via Weighted Vector Subtraction

Doc Doc-Next-Gen (beta) Similar

**mem_affineSpan_iff_exists** theorem

```
{p : P} {s : Set P} : p ∈
affineSpan k s ↔ ∃ p₁ ∈ s, ∃ v ∈
vectorSpan k s, p = v +ᵥ p₁
```

▶ Characterization of Points in Affine Span via Vector Span

Doc Doc-Next-Gen (beta) Similar

**AffineEquiv.constVSub** definition

```
(p : P₁) : P₁ ≃ᵃ[k] V₁
```

▶ Affine equivalence by vector subtraction from a fixed point

Doc Doc-Next-Gen (beta) Similar

**affineSpan_singleton_union_vadd_eq_top_of_span_eq_**

```
{s : Set V} (p : P) (h :
Submodule.span k (Set.range ((↑) :
s → V)) = ⊤) :
  affineSpan k ({ p } ∪ (fun v ⇒ v
+ᵥ p) '' s) = ⊤
```

▶ Affine Span of a Point and its Translations by a Spanning Set is the Entire Space

Doc Doc-Next-Gen (beta) Similar

**linearIndependent_set_iff_affineIndependent_vadd_**

```
{s : Set V} (hs : ∀ v ∈ s, v ≠ (0
: V)) (p₁ : P) :
  LinearIndependent k (fun v ⇒ v :
s → V) ↔
    AffineIndependent k (fun p ⇒ p
: ({ p₁ } ∪ (fun v ⇒ v +ᵥ p₁) '' s
: Set P) → P)
```

▶ Linear Independence of Vectors vs. Affine Independence of Translated Points

Doc Doc-Next-Gen (beta) Similar

**AffineSubspace.vadd_mem_pointwise_vadd_iff** theorem

```
{v : V} {s : AffineSubspace k P}
{p : P} : v +ᵥ p ∈ v +ᵥ s ↔ p ∈ s
```

▶ Translation Invariance of Membership in Affine Subspaces

Doc Doc-Next-Gen (beta) Similar

**AffineSubspace.pointwise_vadd_span** theorem

```
(v : V) (s : Set P) : v +ᵥ affineSpan k s = affineSpan k (v +ᵥ s)
```

▶ Affine Span Commutes with Translation

Doc Doc-Next-Gen (beta) Similar

**mem_vsub_const_affineSegment** theorem

```
{x y z : P} (p : P) : z -ᵥ p ∈ affineSegment R (x -ᵥ p) (y -ᵥ p) ↔ z ∈ affineSegment R x y
```

▶ Translation Invariance of Affine Segment Membership under Vector Subtraction

Doc Doc-Next-Gen (beta) Similar

**affineSpan_induction** theorem

```
{x : P} {s : Set P} {p : P → Prop} (h : x ∈ affineSpan k s) (mem : ∀ x : P, x ∈ s → p x)
  (smul_vsub_vadd : ∀ (c : k) (u v w : P), p u → p v → p w → p (c • (u -ᵥ v) +ᵥ w)) : p x
```

▶ Induction Principle for Affine Span Membership

Doc Doc-Next-Gen (beta) Similar

**AffineEquiv** structure

```
(k P₁ P₂ : Type*) {V₁ V₂ : Type*} [Ring k] [AddCommGroup V₁] [AddCommGroup V₂]
  [Module k V₁] [Module k V₂] [AddTorsor V₁ P₁] [AddTorsor V₂ P₂]
extends P₁ ≃ P₂
```

▶ Affine equivalence between affine spaces

Doc Doc-Next-Gen (beta) Similar

**AffineSubspace.affineSpan_eq_top_iff_vectorSpan_e**

```
{s : Set P} [Nontrivial P] :
affineSpan k s = ⊤ ↔ vectorSpan k s
= ⊤
```

▶ Affine Span Equals Space iff Vector Span Equals Module in Nontrivial Affine Space

Doc   Doc-Next-Gen (beta)   Similar

**`AffineSubspace.coe_pointwise_vadd`** theorem

```
(v : V) (s : AffineSubspace k P) :
((v +ᵥ s : AffineSubspace k P) :
Set P) = v +ᵥ (s : Set P)
```

▶ Translation of Affine Subspace Underlying Set by Vector

Doc   Doc-Next-Gen (beta)   Similar

**`AffineSubspace.mem_affineSpan_insert_iff`** theorem

```
{s : AffineSubspace k P} {p₁ : P}
(hp₁ : p₁ ∈ s) (p₂ p : P) :
  p ∈ affineSpan k (insert p₂ (s :
Set P)) ↔ ∃ r : k, ∃ p0 ∈ s, p = r
• (p₂ -ᵥ p₁ : V) +ᵥ p0
```

▶ Characterization of Affine Span Membership After Insertion via Scalar Multiple and Base Point

Doc   Doc-Next-Gen (beta)   Similar

**`sbtw_vadd_const_iff`** theorem

```
{x y z : V} (p : P) : Sbtw R (x +ᵥ
p) (y +ᵥ p) (z +ᵥ p) ↔ Sbtw R x y z
```

▶ Translation Invariance of Strict Betweenness in Affine Space

Doc   Doc-Next-Gen (beta)   Similar

**`mem_vadd_const_affineSegment`** theorem

```
{x y z : V} (p : P) : z +ᵥ p ∈
affineSegment R (x +ᵥ p) (y +ᵥ p) ↔
z ∈ affineSegment R x y
```

▶ Translation Invariance of Affine Segment Membership

Doc Doc-Next-Gen (beta) Similar

**AffineSubspace.sOppSide_vadd_left_iff** theorem

```
{s : AffineSubspace R P} {x y : P}
{v : V} (hv : v ∈ s.direction) :
s.SOppSide (v +ᵥ x) y ↔ s.SOppSide
x y
```

▶ Translation Invariance of Strictly Opposite Sides Condition for Affine Subspaces

Doc Doc-Next-Gen (beta) Similar

**vsub_mem_vectorSpan_of_mem_affineSpan_of_mem_affi**

```
{s : Set P} {p₁ p₂ : P} (hp₁ : p₁
∈ affineSpan k s) (hp₂ : p₂ ∈
affineSpan k s) : p₁ -ᵥ p₂ ∈
vectorSpan k s
```

▶ Difference of Points in Affine Span Belongs to Vector Span

Doc Doc-Next-Gen (beta) Similar

**AffineEquiv.constVAdd_add** theorem

```
(v w : V₁) : constVAdd k P₁ (v +
w) = (constVAdd k P₁ w).trans
(constVAdd k P₁ v)
```

▶ Additivity of Translation Affine Equivalences: $t_{v+w} = t_w \circ t_v$

Doc Doc-Next-Gen (beta) Similar

**AffineSubspace.pointwise_vadd_top** theorem

```
(v : V) : v +ᵥ (⊤ : AffineSubspace k P) = ⊤
```

▶ Translation of Entire Affine Space by a Vector Preserves the Space

Doc Doc-Next-Gen (beta) Similar

**vadd_mem_affineSpan_of_mem_affineSpan_of_mem_vect**

```
{s : Set P} {p : P} {v : V} (hp : p ∈ affineSpan k s) (hv : v ∈ vectorSpan k s) : v +ᵥ p ∈ affineSpan k s
```

▶ Affine span is closed under translation by vectors in its direction

Doc Doc-Next-Gen (beta) Similar

**AffineSubspace.vadd_mem_iff_mem_direction** theorem

```
{s : AffineSubspace k P} (v : V) {p : P} (hp : p ∈ s) : v +ᵥ p ∈ s ↔ v ∈ s.direction
```

▶ Characterization of Affine Subspace Membership via Direction Vectors

Doc Doc-Next-Gen (beta) Similar

**AffineMap.lineMap_vsub_left** theorem

```
(p₀ p₁ : P1) (c : k) : lineMap p₀ p₁ c -ᵥ p₀ = c • (p₁ -ᵥ p₀)
```

▶ Vector difference property of affine line map: $\text{lineMap}(p_0, p_1)(c) - p_0 = c \cdot (p_1 - p_0)$

Doc Doc-Next-Gen (beta) Similar

**affineIndependent_iff_finrank_vectorSpan_eq** theore

```
[Fintype ι] (p : ι → P) {n : ℕ} (hc : Fintype.card ι = n + 1) :
  AffineIndependent k p ↔ finrank k (vectorSpan k (Set.range p)) = n
```

▶ Affine Independence Characterized by Dimension of Vector Span: $\mathrm{AffineIndependent}(k, p) \leftrightarrow \dim_k(\mathrm{vectorSpan}_k(\mathrm{range}(p))) = n$

Doc Doc-Next-Gen (beta) Similar

**smul_vsub_rev_vadd_mem_affineSpan_pair** theorem

```
(r : k) (p₁ p₂ : P) : r • (p₁ -ᵥ p₂) +ᵥ p₂ ∈ line[k, p₁, p₂]
```

▶ Scaled Reverse Difference Lies in Affine Span of Two Points

Doc Doc-Next-Gen (beta) Similar

**AffineSubspace.mem_direction_iff_eq_vsub_left** theo

```
{s : AffineSubspace k P} {p : P} (hp : p ∈ s) (v : V) : v ∈ s.direction ↔ ∃ p₂ ∈ s, v = p -ᵥ p₂
```

▶ Characterization of Direction Vectors via Left Subtraction in Affine Subspace

Doc Doc-Next-Gen (beta) Similar

**AffineIsometryEquiv.map_vsub** theorem

```
(p1 p2 : P) : e.linearIsometryEquiv (p1 -ᵥ p2) = e p1 -ᵥ e p2
```

▶ Affine Isometric Equivalence Preserves Vector Difference

Doc Doc-Next-Gen (beta) Similar

**affineIndependent_set_iff_linearIndependent_vsub** t

```
{s : Set P} {p₁ : P} (hp₁ : p₁ ∈ s) :
  AffineIndependent k (fun p ⇒ p : s → P) ↔
    LinearIndependent k (fun v ⇒ v : (fun p ⇒ (p -ᵥ p₁ : V)) '' (s \ { p₁ }) → V)
```

▶ Affine Independence of a Set via Linear Independence of Difference Vectors

Doc Doc-Next-Gen (beta) Similar

**AffineMap.coe_sub** theorem

```
(f g : P1 →ᵃ[k] V2) : ⇑(f - g) = f - g
```

▶ Pointwise Difference of Affine Maps

Doc Doc-Next-Gen (beta) Similar

**Finset.sum_smul_vsub_eq_affineCombination_vsub** the

```
(w : ι → k) (p₁ p₂ : ι → P) :
  (∑ i ∈ s, w i • (p₁ i -ᵥ p₂ i)) = s.affineCombination k p₁ w -ᵥ s.affineCombination k p₂ w
```

▶ Weighted Sum of Vector Subtractions Equals Difference of Affine Combinations

Doc Doc-Next-Gen (beta) Similar

**Finset.affineCombination_sdiff_sub** theorem

```
[DecidableEq ι] {s₂ : Finset ι} (h : s₂ ⊆ s) (w : ι → k) (p : ι → P) :
  (s \ s₂).affineCombination k p w -ᵥ s₂.affineCombination k p (-w) = s.weightedVSub p w
```

▶ Difference of Affine Combinations Equals Weighted Vector Subtraction

Doc Doc-Next-Gen (beta) Similar

**AffineSubspace.mem_mk'_iff_vsub_mem** theorem

```
{p₁ p₂ : P} {direction : Submodule k V} : p₂ ∈ mk' p₁ direction ↔ p₂ -ᵥ p₁ ∈ direction
```

▶ Membership in Affine Subspace via Difference Vector

Doc Doc-Next-Gen (beta) Similar

**AffineMap.lineMap_vsub_lineMap** theorem

```
(p₁ p₂ p₃ p₄ : P1) (c : k) : lineMap p₁ p₂ c -ᵥ lineMap p₃ p₄ c = lineMap (p₁ -ᵥ p₃) (p₂ -ᵥ p₄) c
```

▶ Vector Difference of Affine Line Maps Equals Affine Combination of Vector Differences

Doc Doc-Next-Gen (beta) Similar

**AffineSubspace.pointwise_vadd_eq_map** theorem

```
(v : V) (s : AffineSubspace k P) : v +ᵥ s = s.map (AffineEquiv.constVAdd k P v)
```

▶ Translation of Affine Subspace as Image under Translation Map

Doc Doc-Next-Gen (beta) Similar

**AffineSubspace.coe_vsub** theorem

```
(s : AffineSubspace k P) [Nonempty s] (a b : s) : ↑(a -ᵥ b) = (a : P) -ᵥ (b : P)
```

▶ Coercion of Vector Subtraction in Affine Subspace Equals Vector Subtraction in Ambient Space

Doc Doc-Next-Gen (beta) Similar

**AffineSubspace.affineSpan_eq_top_iff_vectorSpan_e**

```
{s : Set P} (hs : s.Nonempty) :
affineSpan k s = ⊤ ↔ vectorSpan k s
= ⊤
```

▶ Affine Span Equals Entire Space iff Vector Span Equals Entire Module for Nonempty Sets

Doc Doc-Next-Gen (beta) Similar

**AffineSubspace.map_pointwise_vadd** theorem

```
(f : P₁ →ᵃ[k] P₂) (v : V₁) (s :
AffineSubspace k P₁) : (v +ᵥ s).map
f = f.linear v +ᵥ s.map f
```

▶ Compatibility of Affine Map with Translation and Subspace Image

Doc Doc-Next-Gen (beta) Similar

**AffineEquiv.constVAdd_zero** theorem

```
: constVAdd k P₁ 0 =
AffineEquiv.refl _ _
```

▶ Translation by Zero Vector is Identity Affine Equivalence

Doc Doc-Next-Gen (beta) Similar

**AffineEquiv.pointReflection_apply** theorem

```
(x y : P₁) : pointReflection k x y
= (x -ᵥ y) +ᵥ x
```

▶ Point Reflection Formula in Affine Space

Doc Doc-Next-Gen (beta) Similar

**AffineMap.instAddTorsor** instance

```
: AffineSpace (P1 →ᵃ[k] V2) (P1 →ᵃ[k] P2)
```

▶ Affine Space Structure on Affine Maps Between Affine Spaces

Doc Doc-Next-Gen (beta) Similar

**AffineMap.left_vsub_lineMap** theorem

```
(p₀ p₁ : P1) (c : k) : p₀ -ᵥ lineMap p₀ p₁ c = c • (p₀ -ᵥ p₁)
```

▶ Left Vector Difference Property of Affine Line Map: $p_0 - \text{lineMap}(p_0, p_1)(c) = c \cdot (p_0 - p_1)$

Doc Doc-Next-Gen (beta) Similar

**affineSpan_eq_affineSpan_lineMap_units** theorem

```
[Nontrivial k] {s : Set P} {p : P} (hp : p ∈ s) (w : s → Units k) :
  affineSpan k (Set.range fun q : s ⇒ AffineMap.lineMap p ↑q (w q : k)) = affineSpan k s
```

▶ Invariance of Affine Span Under Scaled Line Transports from Base Point

Doc Doc-Next-Gen (beta) Similar

**Finset.sum_smul_vsub_const_eq_affineCombination_v**

```
(w : ι → k) (p₁ : ι → P) (p₂ : P) (h : ∑ i ∈ s, w i = 1) :
  (∑ i ∈ s, w i • (p₁ i -ᵥ p₂)) = s.affineCombination k p₁ w -ᵥ p₂
```

▶ Weighted Sum of Vector Differences Equals Affine Combination Minus Fixed Point

Doc Doc-Next-Gen (beta) Similar

**smul_vsub_vadd_mem_affineSpan_pair** theorem

```
(r : k) (p₁ p₂ : P) : r • (p₂ -ᵥ p₁) +ᵥ p₁ ∈ line[k, p₁, p₂]
```

▶ Scaled Difference Added to Point Lies in Affine Span of Pair

Doc Doc-Next-Gen (beta) Similar

**AffineMap.right_vsub_lineMap** theorem

```
(p₀ p₁ : P1) (c : k) : p₁ -ᵥ lineMap p₀ p₁ c = (1 - c) • (p₁ -ᵥ p₀)
```

▶ Right Vector Difference Property of Affine Line Map: $p_1 - \text{lineMap}(p_0, p_1)(c) = (1 - c) \cdot (p_1 - p_0)$

Doc Doc-Next-Gen (beta) Similar

**AffineSubspace.mem_direction_iff_eq_vsub_right** the

```
{s : AffineSubspace k P} {p : P} (hp : p ∈ s) (v : V) : v ∈ s.direction ↔ ∃ p₂ ∈ s, v = p₂ -ᵥ p
```

▶ Characterization of Direction Vectors via Right Subtraction in Affine Subspace

Doc Doc-Next-Gen (beta) Similar

**AffineSubspace.ext** theorem

```
{p q : AffineSubspace k P} (h : ∀ x, x ∈ p ↔ x ∈ q) : p = q
```

▶ Extensionality of Affine Subspaces

Doc Doc-Next-Gen (beta) Similar

**vadd_mem_spanPoints_of_mem_spanPoints_of_mem_vect**

```
{s : Set P} {p : P} {v : V} (hp :
p ∈ spanPoints k s) (hv : v ∈
vectorSpan k s) : v +ᵥ p ∈
spanPoints k s
```

▶ **Affine Span is Closed Under Translation by Vector Span Elements**

Doc   Doc-Next-Gen (beta)   Similar