Trigonometry

Why This Matters

Trigonometry is the mathematics of **angles, rotation, and waves**. It's essential for:

- **Game development**: Rotation, aiming, circular motion
- **Computer graphics**: 3D transformations, lighting, cameras
- **Physics simulations**: Projectiles, pendulums, waves
- **Signal processing**: Audio, video, compression (Fourier transforms)
- **Animation**: Smooth motion, easing, orbits
- **Navigation**: GPS, triangulation

Trigonometry connects **angles** with **coordinates**, making circular and periodic motion mathematically tractable.

The Big Picture: Beyond Triangles

Traditional approach: "Trig is about triangles." **Modern approach**: "Trig is about **rotation** and **periodic motion**."

While trig started with triangles, its real power is in describing:

- **Circular motion**: Wheels, orbits, gears
- **Waves**: Sound, light, tides, oscillations
- **Periodic patterns**: Seasons, clocks, rhythms

---

1. Angles: Rotation, Not Shapes

What Is an Angle?

An **angle** measures **rotation** from a starting direction.

```
     |
     | ← Starting ray
─────●──────
      \
       \ ← Rotated ray
        \
```

Angle = amount of rotation

Degrees

360° = one full rotation (full circle)

```
       0°/360°
          |
    270° ─●─ 90°
          |
         180°
```

Common angles:

```
90° = quarter turn (right angle)
180° = half turn (straight line)
270° = three-quarters turn
360° = full turn (back to start)
```

Radians (The Natural Unit)

Radian: The angle when arc length equals radius

```
   Arc length = radius
         ╱─╲
        ╱   ╲ r
       ●─────╱
        ╲   ╱
         ╲╱
     Angle = 1 radian
```

Key fact: One full circle = 2π radians

```
360° = 2π radians
180° = π radians
90° = π/2 radians
```

Why radians? They make calculus formulas cleaner. Most programming uses radians.

Converting Between Degrees and Radians

```
Degrees to Radians:  multiply by π/180
Radians to Degrees:  multiply by 180/π
```

Examples:

```
45° = 45 × π/180 = π/4 radians
π/6 radians = π/6 × 180/π = 30°
```

Programming:

```
const degToRad = deg => deg * Math.PI / 180;
const radToDeg = rad => rad * 180 / Math.PI;

degToRad(90);   // π/2 ≈ 1.571
radToDeg(Math.PI);  // 180
```

---

2. The Unit Circle (Your Mental Model)

What It Is

A **circle with radius 1** centered at the origin.

```
        y
        │
        1
      •─┼─•
    ╱   │   ╲
   •    •    •  x
    ╲       ╱
      •───•
       -1
```

Every point on the circle has coordinates (x, y) where $x^2 + y^2 = 1$

Angle on the Unit Circle

An angle θ (theta) defines a point on the circle:

```
        y
        │
        │  • (cos θ, sin θ)
        │ ╱│
        │╱ │ sin θ
────────•──┴───────► x
      0 │   cos θ
        θ
```

Key insight:

- **x-coordinate = cos(θ)**
- **y-coordinate = sin(θ)**

This is the definition of sine and cosine!

Special Angles

```
θ = 0°:   (1, 0)      cos(0) = 1,   sin(0) = 0
θ = 90°:  (0, 1)      cos(90°) = 0, sin(90°) = 1
θ = 180°: (-1, 0)     cos(180°) = -1, sin(180°) = 0
θ = 270°: (0, -1)     cos(270°) = 0, sin(270°) = -1
```

Quadrants

```
    II (-,+)  │  I (+,+)
              │
    ──────────•──────────
              │
    III (-,-) │  IV (+,-)
```

- **Quadrant I**: Both positive
- **Quadrant II**: cos negative, sin positive
- **Quadrant III**: Both negative
- **Quadrant IV**: cos positive, sin negative

---

3. Sine and Cosine: The Core Functions

Definitions

On the unit circle:

```
cos(θ) = x-coordinate of point at angle θ
sin(θ) = y-coordinate of point at angle θ
```

Graph of sin(θ):

```
  1 ├──•────────•──────
    │ ╱  ╲      ╱  ╲
  0 ├•────•────•────•─▶ θ
    │      ╲  ╱      ╲
 -1 ├───────•─────────•
    0  π/2  π  3π/2  2π
```

Graph of cos(θ):

```
  1 ├•────────•────────•
    │  ╲      ╱  ╲    ╱
  0 ├───•────•────•──▶ θ
    │    ╲  ╱      ╲
 -1 ├─────•──────────
    0  π/2  π  3π/2  2π
```

Key Properties

Range: Both oscillate between -1 and 1

```
-1 ≤ sin(θ) ≤ 1
-1 ≤ cos(θ) ≤ 1
```

Period: Repeat every 2π (360°)

```
sin(θ + 2π) = sin(θ)
cos(θ + 2π) = cos(θ)
```

Phase shift: Cosine is sine shifted left by π/2

```
cos(θ) = sin(θ + π/2)
```

Pythagorean identity:

```
sin²(θ) + cos²(θ) = 1

Because: x² + y² = 1 on unit circle
```

Programming

```
Math.sin(Math.PI / 2);  // 1  (sin(90°))
Math.cos(0);            // 1  (cos(0°))
Math.sin(0);            // 0  (sin(0°))
Math.cos(Math.PI);      // -1 (cos(180°))
```

---

4. Tangent and Other Functions

Tangent

```
       sin(θ)    y
tan(θ) = ────── = ─
       cos(θ)    x
```

Interpretation: Slope of the line from origin to (cos θ, sin θ)

Graph:

```
    |   |   |
  / | / | / | /
 /  |/  |/  |/
────┼───┼───┼───▶ θ
  -π/2  0  π/2  π

Vertical asymptotes at ±π/2, ±3π/2, ...
(where cos(θ) = 0)
```

Range: All real numbers (-∞ to +∞)

Period: π (repeats every 180°)

Reciprocal Functions

```
            1
csc(θ) = ──────  (cosecant)
          sin(θ)

            1
sec(θ) = ──────  (secant)
          cos(θ)

            1        cos(θ)
cot(θ) = ────── = ──────  (cotangent)
          tan(θ)    sin(θ)
```

Less common, but useful in some contexts.

---

5. Right Triangle Interpretation

SOH-CAH-TOA

For a **right triangle** with angle θ:

```
      ⁄|
hyp  ⁄ | opp
    ⁄  |
   ⁄θ──|
     adj
```

```
       opposite
sin(θ) = ──────────
       hypotenuse

       adjacent
cos(θ) = ──────────
       hypotenuse

       opposite
tan(θ) = ────────
       adjacent
```

Example: Triangle with sides 3, 4, 5

```
      ⁄|
  5  ⁄ | 3
    ⁄  |
   ⁄θ──|
     4

sin(θ) = 3/5 = 0.6
cos(θ) = 4/5 = 0.8
tan(θ) = 3/4 = 0.75
```

Special Right Triangles

45-45-90 triangle:

```
     ⁄|
  √2⁄ |1
   ⁄45°|
    1

sin(45°) = 1/√2 = √2/2 ≈ 0.707
cos(45°) = 1/√2 = √2/2 ≈ 0.707
tan(45°) = 1
```

30-60-90 triangle:

```
     ⁄|
  2 ⁄ |√3
   ⁄60°|
    1
```

```
sin(30°) = 1/2 = 0.5
cos(30°) = √3/2 ≈ 0.866
sin(60°) = √3/2 ≈ 0.866
cos(60°) = 1/2 = 0.5
```

---

6. Inverse Trig Functions

What They Do

Inverse functions answer: "What angle gives this value?"

```
sin(θ) = 0.5  →  θ = ?
arcsin(0.5) = 30° (or π/6)
```

Notation

```
arcsin(x) or sin⁻¹(x)  (inverse sine)
arccos(x) or cos⁻¹(x)  (inverse cosine)
arctan(x) or tan⁻¹(x)  (inverse tangent)
```

Note: $\sin^{-1}(x) \neq 1/\sin(x)$. It means "inverse", not "reciprocal".

Examples

```
sin⁻¹(1) = 90° = π/2
cos⁻¹(0) = 90° = π/2
tan⁻¹(1) = 45° = π/4
```

Domains and Ranges

```
arcsin: Domain [-1, 1], Range [-π/2, π/2]
arccos: Domain [-1, 1], Range [0, π]
arctan: Domain all reals, Range (-π/2, π/2)
```

Programming

```
Math.asin(0.5);   // π/6 ≈ 0.524 (30°)
Math.acos(0);     // π/2 ≈ 1.571 (90°)
Math.atan(1);     // π/4 ≈ 0.785 (45°)

// atan2: handles all quadrants correctly
Math.atan2(y, x);  // angle to point (x, y)
```

---

7. Real-World Applications

Rotation and Direction

Point a character toward target:

```
function angleTo(from, to) {
  const dx = to.x - from.x;
  const dy = to.y - from.y;
  return Math.atan2(dy, dx);  // angle in radians
}

const player = {x: 0, y: 0};
const enemy = {x: 3, y: 4};
const angle = angleTo(player, enemy);  // ~0.927 rad (53°)
```

Circular Motion

Move in a circle:

```
function circularMotion(centerX, centerY, radius, angle) {
  return {
    x: centerX + radius * Math.cos(angle),
    y: centerY + radius * Math.sin(angle)
  };
}

// Orbit around (100, 100) with radius 50
for (let angle = 0; angle < 2 * Math.PI; angle += 0.1) {
  const pos = circularMotion(100, 100, 50, angle);
  // Plot or draw at pos
}
```

Projectile Motion

Initial velocity at angle θ:

```
vₓ = v₀ cos(θ)  (horizontal component)
vᵧ = v₀ sin(θ)  (vertical component)

x(t) = vₓ × t
y(t) = vᵧ × t - ½gt²  (with gravity)
```

```
function shoot(speed, angleDegrees) {
  const angleRad = angleDegrees * Math.PI / 180;
  return {
    vx: speed * Math.cos(angleRad),
    vy: speed * Math.sin(angleRad)
  };
}

const velocity = shoot(100, 45);  // 45° angle
// vx ≈ 70.7, vy ≈ 70.7
```

Waves and Oscillation

Sine wave for smooth oscillation:

```
// Bounce up and down
function bounce(time, amplitude, frequency) {
  return amplitude * Math.sin(frequency * time);
}

// Animate
let time = 0;
function animate() {
  const y = bounce(time, 50, 2);  // 50px amplitude, 2 Hz
  sprite.y = centerY + y;
  time += 0.1;
  requestAnimationFrame(animate);
}
```

Camera/3D Rotation

Rotate point around origin:

```
function rotate(point, angle) {
  const cos = Math.cos(angle);
  const sin = Math.sin(angle);
  return {
    x: point.x * cos - point.y * sin,
    y: point.x * sin + point.y * cos
  };
}

const rotated = rotate({x: 1, y: 0}, Math.PI / 2);
// Result: {x: 0, y: 1} (rotated 90°)
```

Distance and Triangulation

Find distance using angles:

```
If you know angle and one side, you can find others:

         ╱|
      c ╱ | a
       ╱θ—|
         b

a = c × sin(θ)
b = c × cos(θ)
c = a / sin(θ) = b / cos(θ)
```

---

8. Transformations of Trig Functions

General Form

```
y = A sin(B(x - C)) + D

A = amplitude (height)
B = frequency (speed of oscillation)
C = phase shift (horizontal shift)
D = vertical shift
```

Amplitude (A)

Stretches vertically:

```
y = 2 sin(x)     (amplitude 2, oscillates -2 to 2)
y = 0.5 sin(x)   (amplitude 0.5, oscillates -0.5 to 0.5)
```

Frequency (B)

Changes period:

```
Period = 2π / B

y = sin(2x)      (period = π, faster)
y = sin(0.5x)    (period = 4π, slower)
```

Phase Shift (C)

Horizontal shift:

```
y = sin(x - π/2)  (shifted right by π/2)
y = sin(x + π/4)  (shifted left by π/4)
```

Vertical Shift (D)

Moves up/down:

```
y = sin(x) + 1   (oscillates 0 to 2)
y = sin(x) - 2   (oscillates -3 to -1)
```

Example: Ocean Wave

```
function oceanHeight(x, time) {
  const amplitude = 2;      // 2m waves
  const frequency = 0.5;    // slower waves
  const speed = 0.1;        // wave moves
  return amplitude * Math.sin(frequency * (x - speed * time));
}
```

---

9. Important Identities

Pythagorean Identity

```
sin²(θ) + cos²(θ) = 1

Variations:
1 + tan²(θ) = sec²(θ)
1 + cot²(θ) = csc²(θ)
```

Even/Odd Functions

```
cos(-θ) = cos(θ)   (even function, symmetric)
sin(-θ) = -sin(θ)  (odd function, antisymmetric)
tan(-θ) = -tan(θ)  (odd function)
```

Sum/Difference

```
sin(A + B) = sin(A)cos(B) + cos(A)sin(B)
cos(A + B) = cos(A)cos(B) - sin(A)sin(B)
```

Double Angle

```
sin(2θ) = 2sin(θ)cos(θ)
cos(2θ) = cos²(θ) - sin²(θ)
        = 2cos²(θ) - 1
        = 1 - 2sin²(θ)
```

You don't need to memorize these unless you use them often. They're derivable from the unit circle.

Common Mistakes & Misconceptions

❌ "Trig is only for triangles"

Trig is fundamentally about rotation and periodic motion.

❌ "Degrees and radians are the same"

They're different units. Most math/programming uses radians.

❌ "$\sin^{-1}(x)$ means $1/\sin(x)$"

No! $\sin^{-1}(x)$ means $\arcsin(x)$ (inverse function). The reciprocal is $\csc(x) = 1/\sin(x)$.

❌ "tan(90°) = 0"

No! tan(90°) is **undefined** (cos(90°) = 0, division by zero).

❌ "Sine and cosine can be greater than 1"

Not for real angles. They're always in [-1, 1].

❌ "Trig functions only work for 0° to 90°"

They work for all angles, including negative and > 360°.

Tiny Practice

Convert:

1. 180° to radians
2. π/3 radians to degrees

Evaluate (without calculator): 3. sin(0°) 4. cos(90°) 5. tan(45°) 6. sin(180°)

Find angles: 7. If sin(θ) = 0.5, what is θ (0° to 90°)? 8. If cos(θ) = 0, what is θ (0° to 180°)?

Application: 9. A point moves in a circle of radius 5. At angle 60°, what are its (x, y) coordinates?

▶ Answers

Summary Cheat Sheet

Angles

```
360° = 2π radians (full circle)
180° = π radians
90° = π/2 radians

Deg to Rad: × π/180
Rad to Deg: × 180/π
```

Unit Circle

```
Point at angle θ: (cos θ, sin θ)

sin²(θ) + cos²(θ) = 1
```

Core Functions

```
sin(θ) = y-coordinate
cos(θ) = x-coordinate
tan(θ) = sin(θ)/cos(θ) = y/x
```

Special Values

```
      0°   30°  45°  60°  90°
sin:  0   1/2  √2/2  √3/2   1
cos:  1  √3/2  √2/2  1/2    0
tan:  0  √3/3   1    √3    undefined
```

Properties

```
Range: -1 ≤ sin, cos ≤ 1
Period: 2π (360°)
tan period: π (180°)
```

Programming

```
Math.sin(θ), Math.cos(θ), Math.tan(θ)  // radians
Math.asin(x), Math.acos(x), Math.atan(x)  // inverse
Math.atan2(y, x)  // angle to (x,y)

// Circular motion
x = centerX + radius * Math.cos(angle);
y = centerY + radius * Math.sin(angle);

// Rotation
x' = x*cos(θ) - y*sin(θ);
y' = x*sin(θ) + y*cos(θ);
```

Next Steps

Trigonometry connects angles, coordinates, and periodic motion. You now understand:

- The unit circle
- Sine, cosine, tangent
- Applications in rotation and waves

Next, we'll explore **Polynomials**—functions with multiple power terms.

Continue to: 09-polynomials.md