

# FUNDAMENTALS OF AI-AUGMENTED ENGINEERING: A CONVERSATIONAL APPROACH

Taner Onsay, Ph.D. & Sebnem Onsay  
Authors  
August 2026 · Revision 1.0

© 2026 AI First Engineering

Problem



Model



Compute



Analyze



Validate



Improve

*The engineering workflow you will use throughout this book  
by Taner & Sebnem Onsay,*

CHAPTER - 1

PART 2 · ≈ 50 MIN

# One Ball, Six Stages

2

Problem › Model › Compute › Analyze › Validate › Improve — the entire workflow on one problem

Problem

Model

Compute

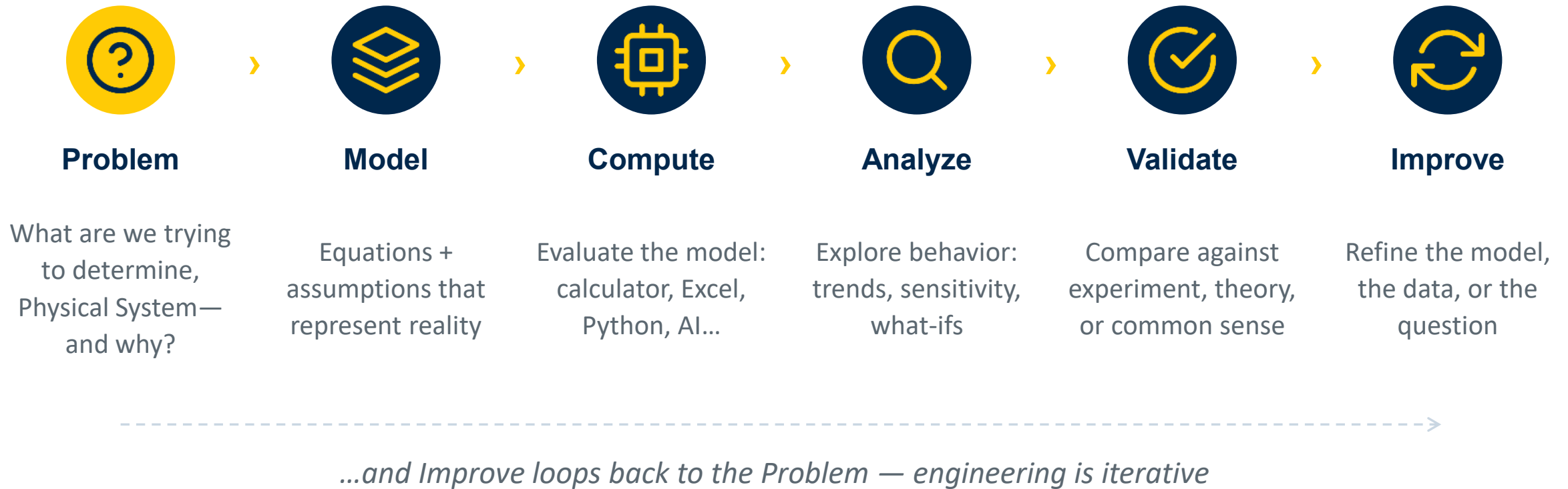
Analyze

Validate

Improve



# How engineers actually use computation



**Memorize this bar. Everything this semester hangs on it.**



## STAGE 1 · PROBLEM

# A ball is released from a known height. How long until it hits the floor?

Deceptively simple — and perfect for learning the workflow.

Before any equation, an engineer pins down the problem:



**What exactly do we want?**

Predicted fall time, in seconds



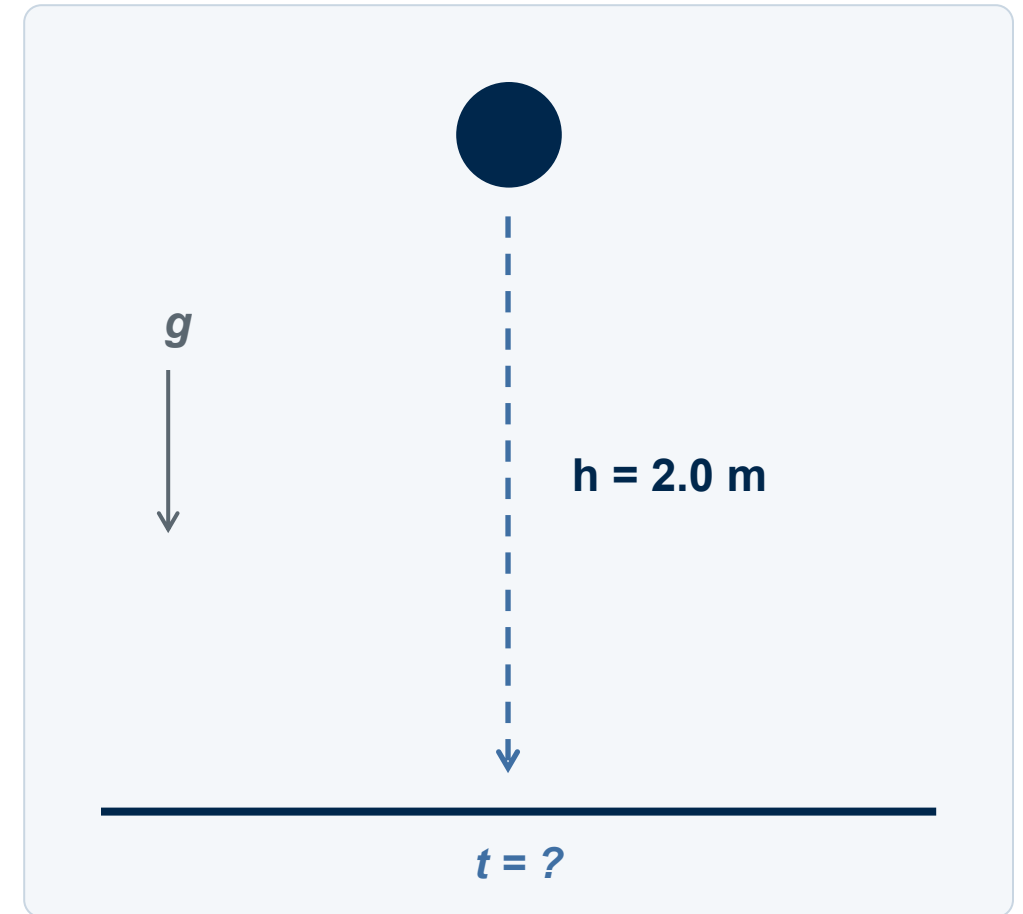
**What do we know?**

Release height  $h$ ; local gravity  $g$



**What will count as success?**

Prediction agrees with a real drop experiment



Problem

Model

Compute

Analyze

Validate

Improve



# Choosing equations means choosing assumptions

## ASSUMPTIONS — SAY THEM OUT LOUD



**Released from rest**

*no initial velocity*



**Constant gravitational acceleration**

*$g = 9.81 \text{ m/s}^2$  near Earth's surface*



**Negligible air resistance**

*reasonable for a dense ball over 2 m ... probably*

**A model is a simplified stand-in for reality. Its assumptions are its fine print.**

### THE MODEL

$$h = \frac{1}{2} g t^2$$

*solve for t:*

$$t = \sqrt{\left(\frac{2h}{g}\right)}$$

*Rusty on this algebra? Ask AI to walk you through the rearrangement, step by step. Then check it by substituting back.*

Problem

**Model**

Compute

Analyze

Validate

Improve



# Evaluate the model — the tool is your choice

By hand / calculator

$$t = \sqrt{\left(\frac{2 \times 2.0 \text{ m}}{9.81 \text{ m/s}^2}\right)}$$
$$t = \sqrt{(0.4077 \text{ s}^2)}$$

$$t \approx 0.64 \text{ s}$$

```
fall_time.py

import math

h = 2.0      # height (m)
g = 9.81     # gravity (m/s^2)

t = math.sqrt(2*h/g)
print(t)

0.6385508568141011
```

**Same model, same answer — different tools.**

Notice the units cancel to seconds. If they didn't, the model — not the calculator — is wrong.



PREDICT

# If the height doubles, does the fall time double?



*Commit to an answer before we compute.*

*Vote: doubles · more than doubles · less than doubles. Defend your vote with the equation.*

Problem



Model



Compute



Analyze



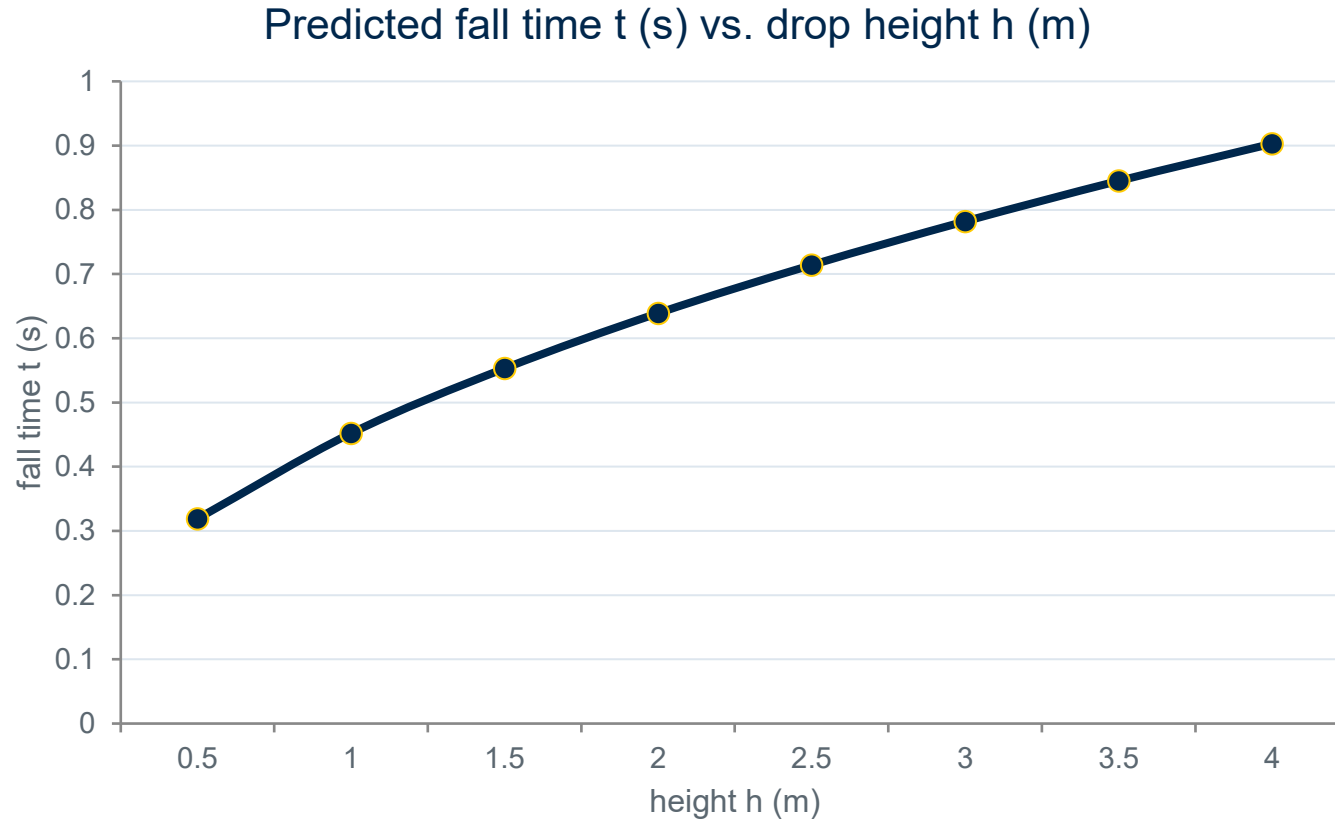
Validate



Improve



# No — fall time grows like the square root



$$t = \sqrt{(2h/g)} \quad \text{— } t \text{ grows like } \sqrt{h}$$

Double  $h$ : 2.0 m  $\rightarrow$  4.0 m  
 $t$ : 0.639 s  $\rightarrow$  0.903 s

**$\times 1.41$ , not  $\times 2$**   
*(that's  $\sqrt{2}$ )*

Analysis = interrogating the model:  
trends, sensitivities, what controls the  
answer.



# A prediction is a hypothesis until reality weighs in



MODEL PREDICTS

$$t \approx 0.64 \text{ s}$$

*from  $t = \sqrt{2h/g}$ ,  $h = 2.0 \text{ m}$*

vs



EXPERIMENT MEASURES

$$t = ?$$

*This week's lab: drop it, film it, time it*

If they disagree — and they will, a little — who is wrong: the model, the measurement, the code... or us?

Problem



Model



Compute



Analyze



**Validate**



Improve



# Disagreement is information

When prediction  $\neq$  experiment, engineers ask better questions:



## Which assumption might be failing?

Air resistance? Truly released from rest?



## How good was the measurement?

Height tape-measured? Timing by eye or by video frames?



## What would we change next?

Better model, better data, better experiment — or a better question

***Improve closes the loop — and usually starts it again.***

Problem



Model



Compute



Analyze



Validate



**Improve**



# One ball, six stages

## Problem



Predict how long a ball dropped from 2 m takes to land

## Model



Free fall from rest, constant  $g$ , no air ›  $t = \sqrt{2h/g}$

## Compute



$t \approx 0.64$  s — by hand, Excel, Python, or AI

## Analyze



$t$  grows like  $\sqrt{h}$  — doubling height gives only  $\times 1.41$  the time

## Validate



Compare with a filmed drop experiment (this week's lab)

## Improve



Explain the gap; question assumptions and measurements



# A calculator solved this in ten seconds...

**1 height**



Calculator is fine.

**100 heights**



A designer needs the full curve — a spec table for every floor of a building.

**10,000 combinations**



Vary h, g, drag, object size... parameter studies are how real designs get chosen.

***Computation scales. Your judgment has to scale with it.***



TOOLS ARE INTERCHANGEABLE. IDEAS ARE NOT.

# One model — three implementations



Excel

```
A1: 2.0          (h)
A2: 9.81         (g)
A3: =SQRT(2*A1/A2)
    > 0.6386
```

*Great for tables, quick studies,  
sharing*



Python

```
import math
h, g = 2.0, 9.81
t = math.sqrt(2*h/g)
# 0.6386
```

*Our main tool: flexible, huge  
ecosystem, AI-friendly*



MATLAB

```
h = 2.0;
g = 9.81;
t = sqrt(2*h/g)
% 0.6386
```

*Industry standard in many  
engineering domains*

**The model is the engineering idea. The software is the tool used to evaluate it.**

*This week we set up Python + VS Code + Jupyter. Excel and MATLAB join later — same ideas, different keystrokes.*