

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 - 5

PART 4 · ≈ 45 MIN

Bisection vs Newton — Which Information Do You Have?

Not “which method is best” — that question has no answer. Which information do you have, what must the answer certify, and can the two methods work together?

4

Problem



Model



Compute



Analyze



Validate



Improve



One slope, one step, one sprint — one question left

THE SLOPE

$f'(x)$ — the sensitivity of the model. Differentiated by hand, checked at one point you can compute. Information bisection threw away.

THE STEP

$x_{n+1} = x_n - f/f'$ — shortfall $\div$ sensitivity. Follow the tangent to zero; repeat until the correction is smaller than the requirement.

THE SPRINT

Near a root on a smooth f , correct digits roughly double each step: two steps did what eleven halvings did. Evidence — never a bound.

THE TRAPS

A flat slope, the wrong root, an iterate off the map. Guards refuse; only validation certifies. Newton converges to A root — you decide whether it is THE answer.

One question left — the Master Plan's question for this week: which method should we trust for this problem, and why? Two contracts follow. Neither wins everywhere.

Problem



Model



Compute



Analyze



Validate



Improve



What each method asks of you — and what it signs for

BISECTION ASKS FOR

a certified bracket · continuity

Two signs and a smooth f . Nothing about how big f is, no slope, no good guess — a crude cage from physics will do.

BISECTION SIGNS FOR

containment · a bound · a schedule

Root $\pm$ half-width you can point at; the halving count known before running. Slow — +3 or 4 halvings per decimal — and sure.

NEWTON ASKS FOR

a justified start · f AND f' · a domain

The derivative of your model, checked; a start the physics defends; a check after every jump that the iterate is still in the world.

NEWTON SIGNS FOR

speed · digits — and nothing else

Two steps for six decimals near a root. No walls, no bound, no schedule: a collapsing correction is evidence, and validation is the only certificate.

No universal winner. The question is never “which method is best?” It is “which information do I have — and what must the answer certify?”

Problem



Model



Compute



Analyze

**Validate**

Improve



Same root, two methods — read the columns, not the winner

	BISECTION (Chapter 4)	NEWTON-RAPHSON (Chapter 5)
information required	a bracket with a sign change; continuity	a start; f and f' at every step; the domain
what it guarantees	containment: root $\pm$ half-width, always	nothing — a collapsing correction is evidence
cost to ± 0.01 here	11 halvings (13 runs) from $[0, 40]$	2 steps from $20.0 \cdot 5$ from 0 — each step a run + a slope
cost per extra decimal	+3 or 4 halvings (additive)	almost none near the root (digits double)
how it fails	no sign change $\rightarrow$ no promise; a jump fakes one	flat slope $\cdot$ wrong root $\cdot$ leaves the domain
Counts belong to THIS f and THESE starts (Parts 2–3). The columns' shapes — guarantee vs evidence, additive vs residual, domain, known case	when to trust it almost free — are what generalize.	only after validation.

READ IT AS A TRADE

Bisection pays in evaluations for a guarantee. Newton pays in information for speed.

READ THE LAST ROW

Both end in the same place: the engineer's validation. One arrives with a bound; the other arrives sooner.

Different information in, different certificate out. A method is a contract. Choose the one whose conditions you can actually meet.

Problem



Model



Compute



Analyze



Validate

5



Improve



Three jobs, one model — choose the method for each

JOB 1 · CERTIFY ONE NUMBER

“Is 20 m/s safe, with margin, to ± 0.05 — for a report that will be signed?” One crossing, one spec, a bound required.
Your call — and your defense?

JOB 2 · REFINE IT, FAST

A design loop solves $f(v) = 0$ ten thousand times, each time from last time’s answer. Six digits every call, on a budget.
Which contract fits?

JOB 3 · A BRAND-NEW f

A function nobody on the team understands yet. No plot, no intuition, no derivative checked — yet.
What comes first — and second?

Reveal — after the argument: bisection certifies (Job 1) · Newton sprints from a start it can trust (Job 2) · reconnaissance and a bracket first, then Newton (Job 3). “Best method” is a property of the method AND the question AND the information you hold.

CHAPTER 4's JUDGMENT, EXTENDED

Then: a march or a hunt. Now: a guarantee or a sprint. Neither is the villain — unexamined use of either is.

Problem



Model



Compute



Analyze



Validate



Improve



Bracket first — move intelligently second

bracket-then-newton — PSEUDOCODE

```
GET a, b, opposite signs      # certify the cage (Ch. 4)
SET x = (a + b) / 2           # a start you can defend
REPEAT
    SET x_new = x - f(x)/f'(x)  # the sprint (Ch. 5)
    IF x_new outside [a, b]:
        SET x_new = (a + b)/2  # fall back: halve
        keep the half of [a, b] with the sign change
    SET x = x_new
UNTIL |correction| ≤ tol
CERTIFY: f(x - tol), f(x + tol) opposite signs
REPORT x ± tol — a root WITH a bound
```

WHY IT WORKS

The bracket supplies what Newton lacks — containment, and a fallback when a jump leaves the cage. Newton supplies what the bracket lacks — speed. Every iterate stays inside two walls; most iterates are tangent jumps.

WHAT YOU GET

A root reached in a few steps AND a bound you can point at — because the last line certifies the sprint with two evaluations. Professional root finders are built this way; you can now read them.

The engineer's answer to “bisection or Newton?” is “both — in that order.” Trap it, then sprint inside the trap, then certify where you landed.

Problem



Model



Compute



Analyze



Validate

7



Improve



Seven evaluations, one bound: 20.368 ± 0.01 m/s

step	action	evaluations	what you now hold
1	certify the cage $[0, 40]$: $f(0) = -50$, $f(40) = +104.3$	2	a bracket 40 wide — Chapter 4's physics cage
2	evaluate the midpoint: $f(20.0) = -1.43$	1	bracket $[20, 40]$ and a defensible start
3	Newton from 20.0: $20.370 \rightarrow 20.36786$	2 (+ 2 slopes)	a settled iterate, inside the cage — evidence
4	certify $x \pm 0.01$: $f(20.358) = -0.039$, $f(20.378) = +0.040$	2	20.368 ± 0.01 — GUARANTEED by a sign change

total: 7 model runs + 2 slopes $\rightarrow$ the ± 0.01 certificate bisection earned with 13 runs · the bound comes from step 4, not from Newton

READ STEP 4

Two evaluations, one sign change: Newton's settled iterate becomes a certified bracket 0.02 wide. The sprint got the walls it never had.

READ THE TOTAL

Seven runs, with a bound — versus thirteen (bisection) or two-without-a-bound (Newton). Information bought speed; two more evaluations bought the certificate.

Trap it. Sprint inside the trap. Certify where you landed. Every number in this table is Chapter 2 arithmetic — and the contract behind the $\pm$ is Chapter 4's.

Problem



Model



Compute



Analyze



Validate

8



Improve



CAN YOU DO NEWTON WITH A STEEL BEAM?

THE MOMENT · A LOAD ON A BEAM

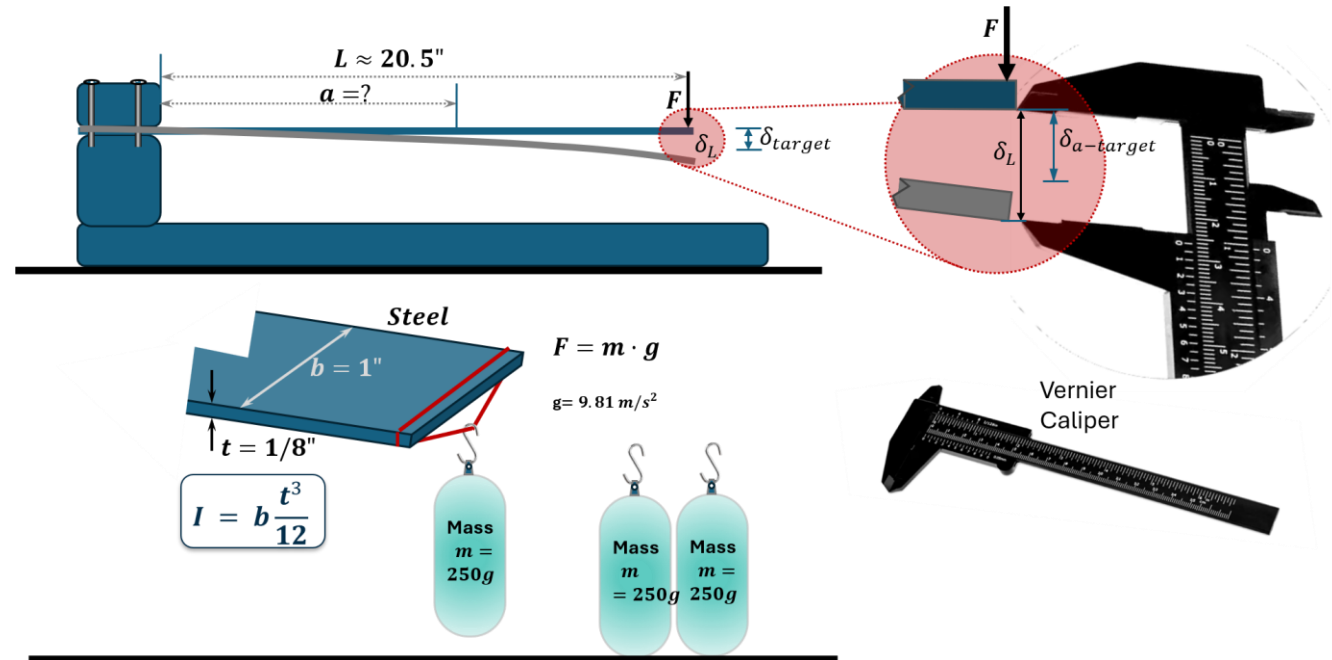
A clamped steel bar, a hanging weight that slides along it, and a mark the free end must reach.

Where along the beam must the weight hang so the tip lands on the mark? One unknown, one crossing — $f(a) = 0$.

NEWTON, IN YOUR HANDS

Compute the step — slide the weight — let the beam settle — measure the tip.

You will physically walk $a_0 \rightarrow a_1 \rightarrow a_2 \rightarrow \dots$ along the bar while the tip walks toward the mark. The model drives every step; the beam judges it.



IN LAB 5 YOU WILL ① measure the beam and the masses · ② validate the deflection model with one, two, three weights · ③ commit a start · ④ physically march Newton to the target · ⑤ ask “Newton converged — did the beam?” · ⑥ run bisection on the same question and compare

Problem



Model



Compute



Analyze



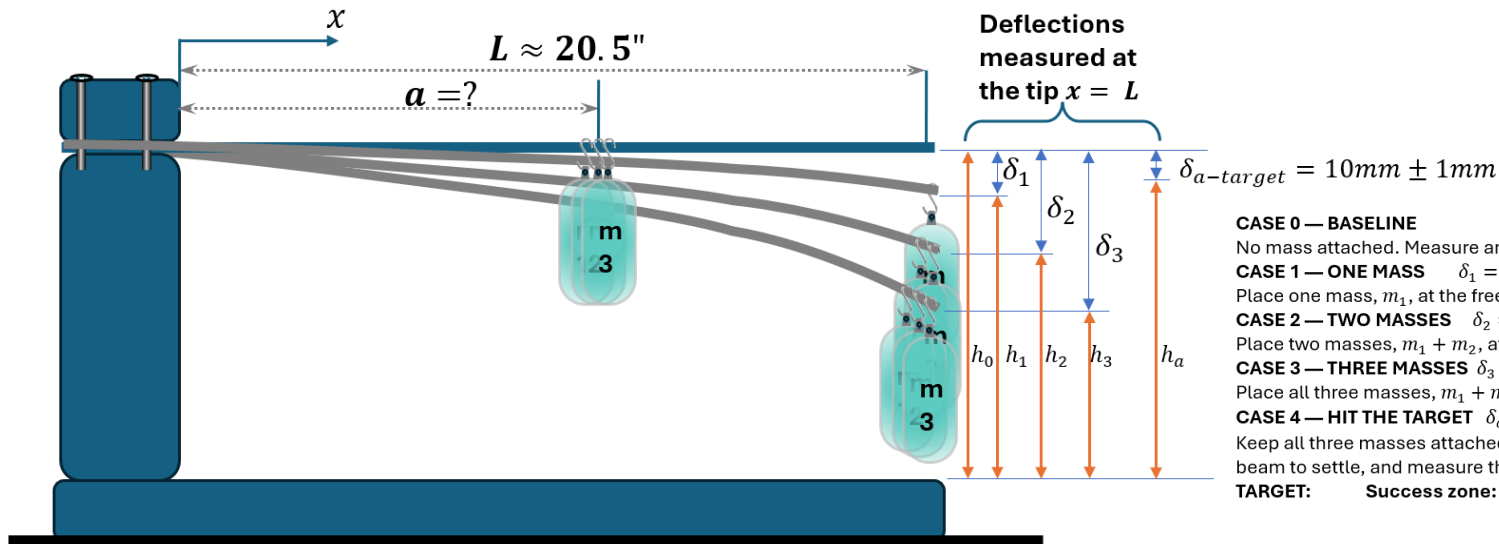
Validate



Improve



DEFLECTION OF A STEEL BEAM:



CASE 0 — BASELINE

No mass attached. Measure and record the unloaded tip position, y_0 . This is the reference position for all deflection measurements.

CASE 1 — ONE MASS $\delta_1 = h_0 - h_1$ (single mass @ the tip)

Place one mass, m_1 , at the free end. Allow the beam to settle. Measure and record the tip deflection, y_1 .

CASE 2 — TWO MASSES $\delta_2 = h_0 - h_2$ (two masses @ the tip)

Place two masses, $m_1 + m_2$, at the free end. Allow the beam to settle. Measure and record the tip deflection, y_2 .

CASE 3 — THREE MASSES $\delta_3 = h_0 - h_3$ (three masses @ the tip)

Place all three masses, $m_1 + m_2 + m_3$, at the free end. Allow the beam to settle. Measure and record the tip deflection, y_3 .

CASE 4 — HIT THE TARGET $\delta_{a-target} = h_0 - h_a$ (find $x=a$, to park the three masses while still measuring the tip deflection h_a)

Keep all three masses attached. Use Newton-Raphson to determine the load position a^* . Move the three-mass load to a^* , allow the beam to settle, and measure the actual tip deflection.

TARGET: Success zone: $9.0 \leq \delta_{a-target} \leq 11.0\text{mm}$. Or stated based on h_0 : $(h_0 - 11.0) \leq h_a \leq (h_0 - 9.0)$

IN LAB 5 YOU WILL ① measure the beam and the masses · ② validate the deflection model with one, two, three weights · ③ commit a start · ④ physically march Newton to the target · ⑤ ask “Newton converged — did the beam?” · ⑥ run bisection on the same question and compare

Problem



Model



Compute



Analyze



Validate



Improve

CHAPTER 5 · SYNTHESIS

What must be true before the laboratory

THE SLOPE IS INFORMATION —
SPEND IT

NEWTON CONVERGES TO A ROOT —
YOU DECIDE

BRACKET FIRST — MOVE INTELLIGENTLY SECOND

The engineer writes the equation, differentiates it and checks the slope, justifies the start, stops on the requirement and never on a count, keeps every iterate inside the world — and still validates the root against reality before deciding.

Ahead: every f in this chapter was an equation you could write down and differentiate. Next, the engineer's f arrives as DATA — a table of measurements with no formula behind it, and gaps between the numbers you actually have. Chapter 6 asks: how do you compute between the points you measured — and when does estimating between them stop being safe?

Problem

›

Model

›

Compute

›

Analyze

›

Validate

›

Improve

AI = Engineering Analyst → Critic: “Attack my start, my slope and my stop — do not just hand me a root.”



What we established in this Chapter Newton-Raphson & the Intelligent Step:

The slope is information: the tangent step is shortfall ÷ sensitivity.

Near a root on a smooth f , correct digits roughly double per step.

A collapsing correction is evidence — never a bound; validation certifies.

Flat slope · wrong root · off the map: guards refuse, the engineer decides.

BRACKET FIRST, MOVE INTELLIGENTLY SECOND - CONVERGED



ENGINEER'S CHECK BOX:

Claiming a root? Show $f(x^*)$ in engineering units — and that x^* is in the world.

Claiming convergence? Show the corrections, not a count.

Claiming a bound? Show the sign change that certifies it.

Chose Newton? Name the information that justified it — and the start.



AI COMPANION PROMPT: Justified-Step Prompt:

“Here is my $f(x) = 0$, my derivative, my proposed start and the accuracy my decision requires. Before any code: check my derivative at one point, tell me whether my start is defensible from the physics, and say what Newton will and will not certify here. Then we run it — and I will verify the root against a known case and the physical domain.”



AI COMPANION PROMPT: Go One Step Deeper Prompt:

“Construct a case where Newton converges cleanly to a root that is not the engineering answer, and one where it leaves the physical domain. Do not tell me the flaw — let me find the violated condition myself, then question me until I can state, from memory, what Newton’s convergence does and does not certify.”