

FUNDAMENTALS OF AI-AUGMENTED ENGINEERING: A CONVERSATIONAL APPROACH

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

© 2026 AI First Engineering



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

CHAPTER - 3

PART 4 · ≈ 35 MIN

Convergence, Tolerance & Knowing When to Stop

The answers are settling toward something. What counts as evidence of that — and who decides how settled is settled enough?

4

Problem



Model



Compute



Analyze



Validate



Improve



The answers settle down — that behavior has a name

step (m/s)	reported v_max	change from previous
0.5	20.0	—
0.1	20.3	+0.300
0.05	20.35	+0.050
0.01	20.36	+0.010
0.001	20.367	+0.007

The change column needs no reference value — it is computed from the answers alone.

CONVERGENCE

As the numerical parameter is refined, the computed result approaches a stable limiting value. Working evidence: shrinking successive differences — consistent with convergence, not yet proof.

WHY ENGINEERS WATCH THE CHANGE

On a real problem there is no 20.368 to compare against. The change column is reference-free evidence consistent with convergence — **the answer interrogating itself.** Evidence, not proof — hold that distinction.

Seen first, named after — the chapter's rule. You experienced **convergence** in Chapter 2. As of today it is vocabulary.

Problem



Model



Compute

**Analyze**

Validate



Improve



THINK



**Refining forever is not an option.
At what point would YOU stop —
and what number goes in the report?**

Decide, and write one sentence defending your stopping point.

Almost every defensible answer contains an idea like: “when successive answers differ by less than ...”. That blank is about to get a name.

Problem



Model



Compute



Analyze



Validate



Improve



Tolerance is an engineering specification

1

POSTING A SPEED LIMIT

Limits are posted in 5 mph increments. ± 0.2 m/s is already far finer than the decision — a coarse tolerance is the honest one here.

2

DESIGNING A BRAKING CONTROLLER

An automated system acts on this number directly. A tighter requirement justifies a tighter tolerance — and buys the computation it costs.

3

ACCIDENT RECONSTRUCTION

Results must be traceable and defensible. Reported precision and the tolerance carry documented justification — engineering traceability, end to end.

Four questions to ask of any tolerance: Who chose it? · What requirement justifies it? · What engineering decision depends on it? · Would another application require another value?

The number may live in the code. The requirement that justifies it lives in the problem statement — same model, three applications, three different tolerances. A person owns each one.

Problem



Model



Compute



Analyze



Validate



Improve



Stop when the evidence meets the specification

```
●●● refine_until_within_tol — PSEUDOCODE
# PSEUDOCODE — not run today
GET tol          # required numerical accuracy
SET step = 0.5
REPEAT
    SET v_new = search(step)
    RECORD step, v_new
    IF step <= tol → STOP
    SET step = step / 2
REPORT v_new, step, tol, refinement evidence

# FOR THIS SEARCH: step <= tol means the
# known error bound (about one step, Part 3)
# is within the required tolerance.
```

TWO DIFFERENT IDEAS

Convergence evidence: do successive refined answers appear to settle?

Stopping specification: has this algorithm's known error bound — about one step — become small enough for the engineering requirement?

Two runs agreeing is never, alone, the stopping rule.

THREE CHAPTERS, ONE STOPPING RULE

Chapter 2: a loop needs a stopping condition.

Chapter 3: the stop must be justified by the numerical method **AND** the engineering requirement.

Chapter 4: a more efficient search whose own error bound gives an even better stopping rule.

A stopping criterion is an engineering decision wearing code.

It is only as trustworthy as the tolerance behind it — and the method's known error bound beneath it.

Problem



Model



Compute



Analyze



Validate



Improve



It runs. It looks careful. It is wrong.

```
●●● check_convergence.py — it runs.  
v1 = find_v_max(step=1.0)  
v2 = find_v_max(step=0.5)  
  
if v2 == v1:  
    print("Converged: v_max =", v2,  
          "m/s (two resolutions)")  
  
Converged: v_max = 20.0 m/s (two  
resolutions)
```

1 · PREDICT

Is the claim sound? Commit before anything else runs.

2 · RUN

The third refinement: find_v_max(step=0.25) → 20.25

3 · INSPECT

The “agreement” was Part 3's grid coincidence — 20.0 sits on both coarse grids. No convergence happened.

4 · EXPLAIN

Agreement between refinements is evidence, not proof. Check a further refinement — and compare the change against the required tolerance, not against zero.

RUNS + LOOKS PLAUSIBLE ≠ ADEQUATELY RESOLVED — Chapter 2's RUNS ≠ CORRECT, one level up.

CONVERGED $\neq$ CORRECT

Convergence is evidence about the NUMERICAL PROCEDURE. It does not validate the model, the parameters, the units, the assumptions, or the measurements.

A beautifully converged answer can still solve the wrong engineering problem. Feed the search a wrong reaction time, a wrong braking value, km/h where it expects m/s — it converges all the same, without complaint. Chapter 2 taught you that a running program proves little. Chapter 3 adds: so does a converging one. Level 4 — validation against independent evidence — is still where engineering lives.





Why $\text{abs}(v2 - v1) < \text{tolerance}$ Does NOT Fix It

●●● THE TEMPTING “FIX”

```
v1 = find_v_max(step=1.0)    # 20.0
v2 = find_v_max(step=0.5)    # 20.0
tol = 0.01
```

```
if abs(v2 - v1) < tol:
    print("Converged")
```

```
abs(v2 - v1) = 0.0
0.0 < 0.01 → TRUE
```

WHY IT STILL FAILS

Changing == to a tolerance test looks more numerical — but the same grid coincidence remains.

The two coarse searches both land on 20.0, so their difference is ZERO.

Any positive tolerance passes. The test still cannot tell whether the apparent agreement will survive another refinement.

WHAT THE THIRD REFINEMENT REVEALS

step = 1.00 → 20.00

step = 0.50 → 20.00 $\Delta = 0.00$ ← looks settled

step = 0.25 → 20.25 $\Delta = 0.25$ ← it moves again

The zero difference was a property of the two grids — not proof that the numerical answer had settled.

A SMALL CHANGE BETWEEN TWO ANSWERS IS CONVERGENCE EVIDENCE — NOT, BY ITSELF, A JUSTIFIED STOPPING CRITERION.

Problem



Model



Compute



Analyze



Validate



Improve



Then Why Not Just Keep Refining?

●●● TWO TRAPS — NEITHER IS THE ENGINEERING RULE

TRAP 1

"Two answers agree → STOP."

TRAP 2

"Not convinced? Keep making
the resolution finer and finer."

Neither rule asks what accuracy
the engineering decision requires.

TRAP 1

Agreement can be accidental — as the 1.0 / 0.5 grid coincidence showed.

TRAP 2

Finer and finer resolution can buy
computation the decision does not need.

THE BETTER QUESTION

What accuracy does THIS engineering
decision require — and what evidence
supports it?

THE ENGINEERING PATH

Requirement → appropriate numerical
method → method-specific error / resolution
information → convergence evidence →
justified stop → validation.

THE GOAL IS NOT THE FINEST ANSWER WE CAN AFFORD — IT IS AN ANSWER SUFFICIENTLY SUPPORTED FOR THE ENGINEERING DECISION.

Problem



Model



Compute



Analyze



Validate



Improve