

FUNDAMENTALS OF AI-AUGMENTED ENGINEERING: A CONVERSATIONAL APPROACH

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

© 2026 AI First Engineering



The engineering workflow you will use throughout this book

CHAPTER - 9

PART 5 · ≈ 35 MIN

Can We Trust the Solver — and the AI?

An AI assistant hands us MATLAB code, a numerical answer and a confident explanation. Part 5 asks what evidence would justify trusting any of them — and closes Chapter 9.

5

Problem

›

Model

›

Compute

›

Analyze

›

Validate

›

Improve



CONTENTS OF CHAPTER 9:

Chapter 8: a few coupled unknowns, solved directly. **Chapter 9:** far too many — solved by repeated correction, and checked.

1

Why do we need another way to solve equations?

From two unknowns to 100,000 · mostly-empty matrices · the 80 GB surprise · direct vs. iterative



2

How does Jacobi actually work?

Repeated correction by hand · using only the previous guess · pseudocode and compact MATLAB



3

When do iterations converge — and when do they fail?

Convergence and divergence · equation order · change vs. residual vs. error · stopping safely



4

Is Jacobi the only way? What happens at engineering scale?

Gauss-Seidel · a growing sparse network · LU in concept · what industrial solvers do



5

Can we trust the solver — and the AI? · you are here

Auditing AI-written solver code · ACCEPT / MODIFY / REJECT · a verification checklist



Course spiral: Ch 4–5 — one unknown, improved step by step until a stopping test is met (bisection, Newton) · Ch 8 — several coupled unknowns, solved directly · Ch 9 — the same kind of system, solved by repeated correction, and judged by evidence.



The AI Says It Works. Should We Believe It?

WE ASKED

"Write a MATLAB Jacobi solver for our two-node circuit:

$$3V_1 - V_2 = 5, \quad -V_1 + 2V_2 = 0$$

(mS, mA, V)."

AI ASSISTANT

"Here is a clean Jacobi implementation. It converges quickly, and the result matches the expected node voltages."

```
>> ai_script_A
Jacobi converged in 6 iterations: V1 = 2.0000 V, V2 = 1.0000 V
Solution accurate to within 0.001 V.
```

THREE THINGS ON THE TABLE

The code. The numbers. The explanation. **Which of them can we trust — and what evidence would convince us?**

AI proposes; the engineer verifies and decides. Today: evidence that holds up no matter how confident the explanation sounds.

Problem



Model



Compute



Analyze



Validate



Improve



What Exactly Did We Ask the AI to Do?

THE SPECIFICATION

"Implement the Jacobi iterative method in MATLAB for $Ax = b$. Start from $x_0 = [0; 0]$. At each iteration, compute every new component using only values from the previous iteration. Stop when the maximum absolute change across all components is less than 0.001 V, or when the maximum iteration count is reached. Detect invalid numerical states and report convergence status, final voltages, iteration count, and residual."

TESTABLE CRITERIA

- **Method:** every update uses previous-iteration values
- **Start:** $x_0 = (0, 0)$ V
- **Stop:** max change over ALL components < 1 mV — or max_iter
- **Safety:** zero diagonal, NaN, Inf reported
- **Report:** status, V, k, residual (mA)

Test system: $A = [3 \ -1; -1 \ 2]$ mS, $b = [5; 0]$ mA

Trusted reference: $x_{\text{ref}} = (2, 1)$ V

Remember Part 3: meeting the stopping rule is **not** an accuracy guarantee. The spec asks for a residual so the answer can be checked.

You cannot audit code against a vague request. A precise specification turns "it looks fine" into pass or fail.

Problem



Model



Compute



Analyze



Validate



Improve



Audit 1: The Algorithm Has Changed

AI CODE · CORE LOOP

```
for k = 1:max_iter
    x_prev = x;
    x(1) = (5 + x(2))/3;
    x(2) = x(1)/2;
    if any(~isfinite(x)), break; end
    if max(abs(x - x_prev)) < tol
        converged = true; break;
    end
end
```

Highlighted: the second update reads the $x(1)$ computed one line earlier.

```
x_old = x;  x(1) = (5 + x_old(2))/3;
x(2) = x_old(1)/2;  % Jacobi: old values only
```

DIAGNOSIS

- Not Jacobi: **$x(2)$ uses the newest $x(1)$** — a Gauss-Seidel update (Parts 2 and 4).
- Evidence: first iterate **(1.6667, 0.8333) V**; Jacobi gives **(1.6667, 0) V**.
- It does reach (2, 1) V — in 6 iterations; real Jacobi needs 10.
- It even saves **x_prev** — but uses it only for the change test.
- **Algorithm-identity mismatch** — not a wrong numerical answer.

Right numbers, wrong method. It violates the specification even though the answer is close to (2, 1) V.

Problem ›

Model ›

Compute ›

Analyze ›

Validate ›

Improve



Audit 2: The Solver Stops Too Soon

AI CODE · CORE LOOP

```

for k = 1:max_iter
    x_old = x;
    x(1) = (5 + x_old(2))/3;
    if abs(x(1) - x_old(1)) < tol
        converged = true; break;
    end
    x(2) = x_old(1)/2;
end

```

CORRECTED · UPDATE ALL, THEN TEST ALL

```

x_old = x;
x(1) = (5 + x_old(2))/3; x(2) = x_old(1)/2;
change = max(abs(x - x_old));
if change < tol, converged = true; break; end

```

k	x(1) (V)	test $ \Delta x(1) $	action	x(2) (V)
1	1.6667	$1.6667 \geq \text{tol}$	continue	0
2	1.6667	$0 < \text{tol}$	break	never updated

Table 1. Executed trace from (0, 0) V, tol = 0.001 V.

WHAT IT RETURNS

"Jacobi converged in 2 iterations"

x = **(1.666667, 0) V** vs x_ref = (2, 1) V

Reference error: **1.000 V**

Residual: $Ax - b = (0, -5/3) \text{ mA} \rightarrow \max \text{1.666667 mA}$

Not a loose tolerance — an incomplete test, checked before the iteration was finished.

Problem ›

Model ›

Compute ›

Analyze ›

Validate ›

Improve



The Output Looks Reasonable. Is It Right?

	AI code A	AI code B
Runs without a MATLAB error	✓	✓
Prints voltages and "converged"	✓	✓
Implements the specified Jacobi method	✗ Gauss-Seidel update	✓ updates are Jacobi
Stops only after a complete update	✓	✗ tests x(1) alone, too early
Reference error of the returned x	0.043 mV	1.000 V
Max residual of the returned x	0.00011 mA	1.667 mA
Diagnosis	algorithm substitution	premature termination
Fix	read x_old in both updates	update all, then test max change

Table 2. Both scripts executed as shown; A: $x = (1.999957, 0.999979)$ V at $k = 6$; B: $x = (1.666667, 0)$ V at $k = 2$.

RUNS ≠ CORRECT. Different defects need different diagnoses — and different fixes.

Problem



Model



Compute

**Analyze**

Validate



Improve



Build a Test the AI Cannot Talk Its Way Around

EXPECTED — WITHOUT THE AI

Trusted solve: $\mathbf{x}_{\text{ref}} = \mathbf{A} \backslash \mathbf{b}$

$= (2, 1) \text{ V}$

Substitution: $3(2) - 1 = 5 \checkmark \quad -2 + 2(1) = 0 \checkmark$

By hand, Jacobi from zero: $\mathbf{x}(1) = (1.6667, 0) \text{ V}$

These two facts alone catch both audits.

ITERATION CHANGE

$\max(\text{abs}(\mathbf{x}_{\text{new}} - \mathbf{x}_{\text{old}}))$

units: V

what B tested ($x(1)$ only):

0 V

RESIDUAL

$\max(\text{abs}(\mathbf{A} * \mathbf{x} - \mathbf{b}))$

units: mA

of B's returned x:

1.667 mA

REFERENCE ERROR

$\max(\text{abs}(\mathbf{x} - \mathbf{x}_{\text{ref}}))$

units: V

of B's returned x:

1.000 V

- The AI's own stopping quantity said "no change". The residual and the reference error — computed independently — said **wrong**.
- A trusted reference exists for small teaching examples — rarely for large engineering systems, where residuals and known test cases carry the load.

Decide what "correct" looks like before you run the code. Then the explanation cannot change the verdict.

Problem



Model



Compute



Analyze



Validate



Improve



Test the Solver Before Trusting the Circuit

Check	AI code A	AI code B	Corrected Jacobi
1 Known solution: within 1 mV of (2, 1) V?	PASS 0.04 mV	FAIL 1 V	PASS 0.26 mV
2 First iteration from zero = (1.6667, 0) V?	FAIL (1.67, 0.83)	PASS	PASS
3 Identity: one step from (1, 1) gives (2, 0.5)?	FAIL (2, 1)	PASS	PASS
4 Stopping: one more Jacobi sweep moves < 1 mV?	PASS 0.04 mV	FAIL 833 mV	PASS 0.21 mV
5 Residual of the answer below 0.01 mA?	PASS	FAIL 1.67 mA	PASS
6 Cap (max_iter = 3): reports not converged?	PASS	FAIL claims converged	PASS
7 Zero diagonal caught before dividing?	FAIL divides → Inf	FAIL divides → Inf	PASS
8 NaN input: no convergence claimed?	PASS	PASS	PASS
9 Strong coupling: no accuracy claim from small change?	FAIL claims 1 mV	FAIL claims 1 mV	PASS

Table 3. Every cell produced by executing `ch9p5_audit.m`. Nine checks raise confidence for these cases — they do not prove correctness for every matrix.

Small, sharp tests with known answers — each one aimed at a specific way the solver could be wrong.

Problem



Model



Compute



Analyze

**Validate**

Improve



ACCEPT, MODIFY, or REJECT?

Classify each AI artifact — the code, its result, its claim — and defend the decision with evidence.

AI CODE A · algorithm substitution

MODIFY

Evidence

- Answer within 0.043 mV of (2, 1) V; residual 0.00011 mA.
- But the update is Gauss-Seidel; checks 2 and 3 fail.

Action: Change both updates to read `x_old`, rerun the suite, then decide again.

AI CODE B · premature termination

REJECT

MODIFY

the result

the code

Evidence

- Returned `x` is 1.000 V from the reference; residual 1.667 mA.
- Update formulas are fine; the stopping test is incomplete.

Action: Reject the returned result. Repair the stopping logic, rerun, verify again.

A decision applies to a specific artifact. A wrong result can be rejected while the code behind it is repairable.

Problem



Model



Compute



Analyze



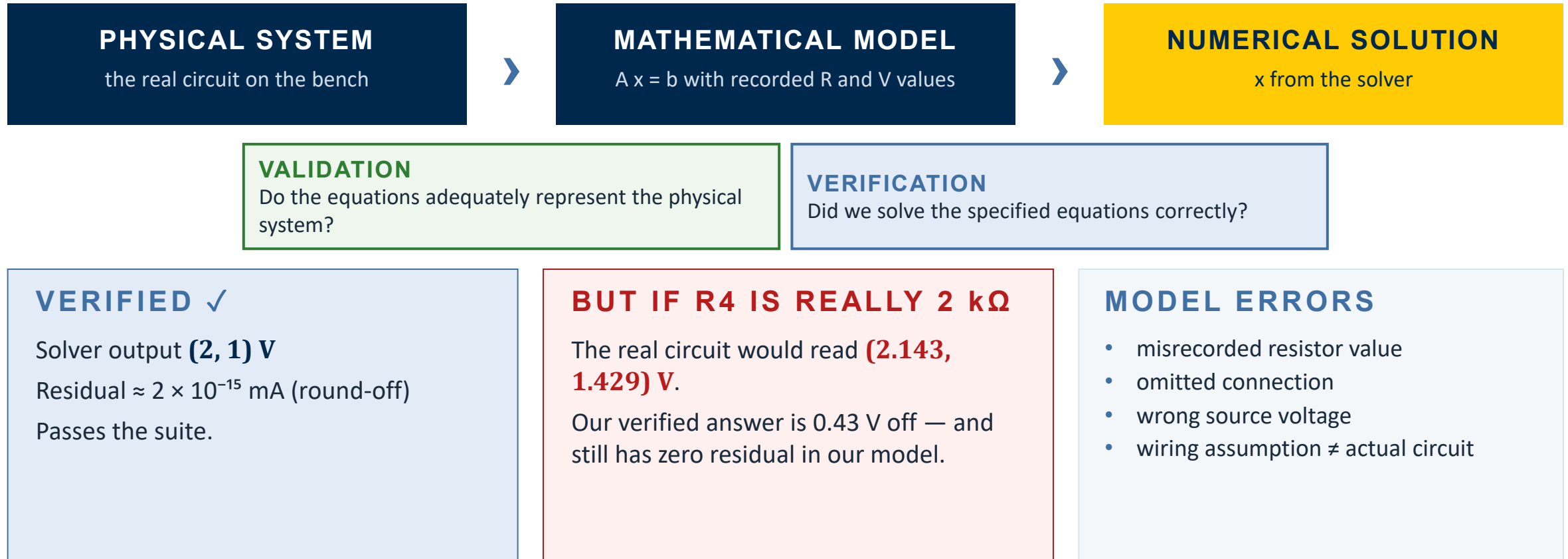
Validate



Improve



A Correct Solver Can Still Solve the Wrong Problem



Verification checks the computation; validation checks the model. Only measurements of the real system can validate it.

Problem



Model



Compute



Analyze

**Validate**

Improve



The Engineer's Verification Workflow



...inside the course workflow



IMPROVE: ACCEPT / MODIFY / REJECT — based on evidence

WHERE TODAY'S FINDINGS WERE CAUGHT

VERIFY Code A: first-iteration test exposed the Gauss-Seidel update.

VERIFY Code B: residual 1.67 mA and reference error 1 V exposed early stopping.

VALIDATE R4 misread: only a comparison with the real circuit can reveal it.

Specify and predict before you run; verify and validate before you decide.





YOUR TURN

An AI-Generated Engineering Claim

EVIDENCE · STRONGLY COUPLED CIRCUIT

```
A ..... [20 -19; -19 20] mS
b ..... [5; 0] mA
x_ref ..... (100/39, 95/39) V
method ..... Jacobi, x0 = (0, 0) V
status ..... converged
iterations .... 109
final change .. 0.98195 mV
max ref error . 9.56772 mV
max residual .. 0.018657 mA
```

AI CLAIM

"My Jacobi solver converged because the maximum change was less than 1 mV. Therefore, my computed voltages are accurate to within 1 mV."

ACCEPT, MODIFY, or REJECT this claim? What numerical evidence supports your decision?

ACCEPT

MODIFY

REJECT

Correct stopping behaviour does not automatically imply sufficient engineering accuracy.

Problem



Model



Compute



Analyze



Validate



Improve



From Running Code to Defensible Engineering

PART 1

Scale makes direct storage impossible — iterate.

PART 2

Jacobi: repeated updates from the previous vector.

PART 3

Convergence must be shown, not assumed.

PART 4

Gauss-Seidel and other solver families are choices.

PART 5

AI-assisted work needs independent verification.

A computed answer is a candidate. **Evidence makes it defensible.**

THE SAME QUESTIONS, EVERY NUMERICAL METHOD FROM HERE ON

differentiation

integration

differential equations

simulation

Which method? Converged? Verified? Valid?
Accurate enough?

RUNS $\neq$ CORRECT. AI proposes; the engineer verifies and decides.

Problem



Model



Compute



Analyze



Validate



Improve