How far can you trust NeuralFoil below Re = 500,000? A two-tunnel test on 94 airfoils, with XFoil run at every point

Author: Neel Madhav **Date:** 2026

Abstract

Small drones and small wind turbines fly slowly. That puts them in a range engineers call low Reynolds number, roughly 50,000 to 500,000. Down there a bubble of stalled air forms on the wing and makes most of the drag, and a wing tuned for one exact condition falls apart as soon as things change.

To design wings fast, the field now uses NeuralFoil. It's a neural network that copies an older solver called XFoil, and it answers in a fraction of a second. It also reports a confidence score with every answer. But its only published wind-tunnel check sits at Re = 1,800,000, about ten times faster than a drone flies. Nobody knew how far off it is where people actually use it, or whether that confidence score means anything real.

This paper tests it against **9,130 wind-tunnel measurements on 94 airfoils in two separate tunnels**. One set comes from the UIUC low-speed archive (55 airfoils, Re = 60,000 to 500,000). The other comes from an older Princeton data set, *Airfoils at Low Speeds* (54 airfoils, Re = 60,000 to 300,000). On top of that there are 2,629 points with a trip strip and 412 lift sweeps that go past the stall. XFoil itself was run at all 9,130 conditions, which splits the error into XFoil's share and the network's share. Fifteen airfoils were tested in both tunnels, so the tunnels can be checked against each other. The measured error is then carried through a rotor, in hover and in forward flight, and round a weight-closure loop, so the benchmark says what the error costs a design and not just how big it is. Every number carries an error bar built by resampling whole airfoils, and a fitted model turns any prediction into an expected drag error.

What came out. NeuralFoil's drag is off by 11 to 12 percent on average, 8 percent in a typical case, and its lift by 0.07 to 0.09. Both tunnels agree. At Re = 60,000 the drag error grows to 17 to 22 percent. Almost none of that belongs to the network. XFoil misses the same points by 12 percent, and NeuralFoil differs from XFoil by only 2.8 percent. The two errors track each other at r = 0.95. NeuralFoil even lands closer to the tunnel than XFoil does, on 55 percent of points. Match every Princeton polar to its nearest UIUC one and the two archives differ by 10.9 percent in drag. Read that as two archives disagreeing rather than as one tunnel's reproducibility limit: a matched pair is two different hand-built models. The confidence score warns about drag but not lift, and it tracks XFoil's error more strongly than the network's. Honest error bars run two to five times wider than naive ones, and every headline result survives them. One result doesn't. A link between an airfoil's average

confidence and its average error shows up in the UIUC data and vanishes in the Princeton data, and this paper shows why. A fitted model reads confidence, Reynolds number, angle and camber and predicts held-out drag error 14 percent better than a flat guess, in either tunnel. Running the model on the measured shape of each Princeton wing, instead of its drawing, removes about a sixth of the drag error. Carried into a hovering rotor whose blade sits entirely inside the benchmark's Reynolds range, the drag error arrives divided by three, because two thirds of hover power is induced. Close the design loop, letting battery mass feed back into take-off weight, and it doubles again: the amplification factor from section drag to take-off weight is 0.63 to 0.74, and the 90 percent interval of the measured error is worth three and a half kilograms on a 15.7 kg aircraft. Hover is the mild case. In forward flight the profile share of shaft power climbs from 41 to 68 percent, and the same error costs 1.7 times as much. Finally, a design pipeline uses all of this: a small reward for staying where NeuralFoil is confident moves a design out of its worst region for about 2 percent of predicted worst-case L/D.

---

1. Introduction

1.1 Why slow air is hard

A delivery drone, a hand-launched plane and a rooftop wind turbine all share one problem. They move slowly, and slow air behaves differently.

Below a Reynolds number of about 500,000, the thin layer of air hugging the wing lifts off the surface before it goes turbulent. It then reattaches further back, trapping a pocket of stalled air underneath. That pocket is called a laminar separation bubble. It adds drag, it slides around as speed and angle change, and it can burst without warning and stall the wing. This is also the range where solvers like XFoil are least reliable, and where wind-tunnel results are hardest to repeat.

1.2 One operating point isn't enough

Tune an airfoil for a single speed and a single angle and it can lose most of its performance a short step away in either direction. Real drones and turbines never hold one condition. So the goal isn't a sharp peak. It's decent performance across the whole range the wing will actually see.

1.3 A fast stand-in nobody had checked

Finding a good shape takes thousands of aerodynamic calculations, and a real viscous solver is far too slow to sit inside that loop. So the field turned to **NeuralFoil** (Sharpe and Hansman, 2025), a neural network trained on a huge pile of XFoil results. It's fast. It's also differentiable, which means the optimizer can see exactly how a small shape change moves lift and drag. And it hands back a confidence score every time.

Here's the gap. NeuralFoil's one published comparison against a real experiment is a single NACA 0012 case at Re = 1,800,000. Its main check is against XFoil itself, on about 400,000 held-out cases. That tells you how well it copies XFoil. It doesn't tell you whether XFoil or NeuralFoil match a wind tunnel below Re = 500,000. It doesn't say what the confidence score means. And it doesn't say how much of any error belongs to the network rather than to the solver it was built to imitate. This paper answers all three.

1.4 Related work

Slow-speed airfoils and how they get measured. The separation bubble and its effect on drag are well documented (Drela, 1989; Selig, 2003), and the UIUC group has written up both the tunnel methods and their limits (Selig, Deters and Williamson, 2011). Two points from that work matter here. First, XFoil is known to struggle in exactly this range. A recent study at Re = 68,000 to 159,000 found it unreliable at the low end and better at the high end (Demie, Ancha and Kahsay, 2026). Second, the experiments have limits of their own. Slow-speed drag measured with a wake rake shifts with the tunnel's turbulence, the model's accuracy and where the rake sits (Selig, Deters and Williamson, 2011). Neither point had been turned into a number a NeuralFoil user could act on. Sections 3.2 and 3.3 do that.

Neural stand-ins for airfoil analysis. Machine-learned surrogates are common in shape optimization now (Li, Du and Martins, 2022). NeuralFoil stands out for the size of its XFoil training set, for being differentiable, and for reporting a confidence score. Its published checks are against XFoil and against one experiment at Re = 1,800,000. No slow-speed experimental benchmark of it had been published.

Robust and multi-objective design. People have known for a long time that single-point optimization is fragile and that you should design across a range (Drela, 1998; Li, Huyse and Padula, 2002). The worst-case setup, the weighted multi-objective score and the sample-based build-error robustness in sections 2.3 to 2.5 are all standard. This paper uses them. It doesn't claim them.

Feeding a model's uncertainty back into the optimizer. This is an established field too. Trust-region methods keep each step inside the region where the approximation has been checked (Alexandrov, Dennis, Lewis and Torczon, 1998). Kriging methods put the surrogate's own predicted variance into the search rule (Jones, Schonlau and Welch, 1998; Queipo et al., 2005; Forrester, Sobester and Keane, 2008). And recent work penalizes a neural surrogate's predicted uncertainty directly (Yang, Li, Zhang and Chen, 2026). So what's different in section 3.9 is narrower than "nobody has done this". Those methods use an uncertainty the model computes about itself. This paper first measures what NeuralFoil's confidence score is worth in real wind-tunnel error, then uses the score with that measured meaning attached. The optimizer is ordinary. The scale on the penalty is the part that was measured.

1.5 What this paper adds

1. **A two-tunnel, 94-airfoil test of NeuralFoil below Re = 500,000.** 9,130 clean measurements, 2,629 with a trip strip, 412 lift sweeps through stall. It gives a measured error for lift, drag, L/D and maximum lift at each Reynolds number. It shows the error is the same in two separate labs, and shows what the confidence score can and can't predict.
2. **A split of that error into XFoil's part and the network's part**, by running XFoil at every one of the 9,130 conditions.
3. **The noise floor of the experiments themselves**, from the 15 airfoils measured in both tunnels, plus a way to pull the wings' build error out of the model's error using Princeton's measured coordinates.
4. **Honest statistics.** Error bars that resample whole airfoils, a breakdown of where the confidence-versus-error link actually lives, and a cross-checked model that turns a prediction into an expected error.
5. **What the error costs a design.** The measured drag error propagated through a blade-element rotor in hover and in forward flight, and through a weight-closure loop, with the correlation assumption stated and both bounds reported. It gives an amplification factor from section drag to take-off weight, and a rule for carrying a section error into any rotor.
6. **A working design pipeline** for worst-case, multi-objective and build-tolerant airfoils, plus a version that uses the measured confidence calibration inside the design loop.

---

2. Methods

2.1 Describing the shape

Every airfoil here is written as 17 numbers: 8 for the top surface, 8 for the bottom, and 1 for the leading edge. This is the Kulfan, or CST, basis (Kulfan, 2008), with the trailing edge fixed at 0.25 percent of the chord. It's smooth, it uses few numbers, it always gives a shape you could build, and it plays well with calculus. That last part is what a gradient-based optimizer needs.

2.2 Getting lift and drag

Lift, drag and moment all come from NeuralFoil's `large` network, with the bigger `xxlarge` run as a check. It returns CL, CD, CM, a confidence score between 0 and 1, and the state of the boundary layer. Because it's differentiable, the exact effect of each of the 17 shape numbers on lift and drag is available for free. That's what turns an optimization over a 25-point grid of conditions into a few seconds of work.

2.3 Chasing the worst case

The design goal is the worst lift-to-drag ratio anywhere in the operating range, not the average. That sounds awkward to optimize, because "the worst" isn't a smooth function. The fix is a standard trick. Add one extra number, call it g, and force it to sit at or below the L/D at every grid point. Then push

g as high as it will go. Since g can never rise above the worst point, pushing it up drags the worst case up with it, and the whole thing stays smooth enough to solve quickly.

The grid is five Reynolds numbers, spaced evenly on a log scale across a decade, by five angles of attack. The problem has more than one hilltop, so each design is solved from several NACA starting shapes and only the best is kept. Families of designs are traced by starting each solve from the previous answer. Everything runs through AeroSandbox and IPOPT.

2.4 Four goals at once

Four real goals are combined into one weighted score. Each is scaled first, so the weights can be compared fairly.

Goal	What gets measured	Direction
Efficiency	worst-case L/D across the range	higher is better
Safety	lift held at 12 degrees, Re = 50,000	higher is better
Structure	maximum thickness, meaning spar depth	higher is better
Noise	trailing-edge boundary-layer thickness	lower is better

2.5 Designing for a wing built badly

No real wing matches its drawing, especially a 3D-printed or hand-built one. So small random bumps are added to the 17 shape numbers, sized to about 0.5 percent of the chord in surface error. One fixed set of these errors is drawn once, and the worst case is taken across both the operating grid and that set. Then the finished design is tested against a fresh, larger batch of errors it never saw during optimization.

2.6 The wind-tunnel data

UIUC. The measurements come from the UIUC Low-Speed Airfoil Tests archive (Selig et al., 1995, 1996; Lyon et al., 1998; Selig and McGranahan, 2004), which publishes its tables as plain text. Every airfoil in those four volumes with coordinates in the AeroSandbox database is included, in its plain form, since the surrogate can't represent flaps or gurney flaps. That gives 57 airfoils, 91 files and 5,441 drag-run points at Re = 40,000 to 500,000, plus 21,000 lift-run points that sweep through stall and back. Clean runs and trip-strip runs live in separate files and are kept separate here. A parser turns it all into one table, so every number traces back to its source file.

For 55 of the 57 airfoils, the 17-number description reproduces the published shape to better than 0.5 percent of the chord, which is the same build-error scale used elsewhere in this paper. The two that don't, A18 and BE50 at 0.9 to 1.3 percent, are reported but left out of the statistics. Their disagreement with experiment can't be separated from their shape error. The E387 comes out at 0.15

percent. Even that little matters: run XFoil on the true E387 and on its 17-number version and lift shifts by about 3 percent, drag by 4 to 7 percent. That's roughly half the E387 lift error and a third of the drag error in section 3.1. So the errors here are what a designer using this pipeline actually gets, and an upper bound on the network alone.

NeuralFoil was run at every measured condition. Clean runs got free transition with n_crit = 9. Trip-strip runs got run twice, once free and once with transition forced at the trip positions in the file header.

Princeton. The whole test then repeats on *Airfoils at Low Speeds* (Selig, Donovan and Fraser, 1989). Those runs happened in Princeton's 3 by 4 foot smoke tunnel between 1986 and 1989, and the same archive shares them as plain text. It holds 127 airfoil-and-setup blocks (7,762 drag points) and 135 lift files (15,565 points) on 54 airfoils at Re = 60,000 to 300,000. Clean and trip-strip blocks were kept. Flaps, gurney flaps, blowing, clay leading edges and mixed setups were dropped. This set has one thing UIUC doesn't: somebody measured the actual wind-tunnel models with a profiler and published those coordinates next to the design ones. So NeuralFoil was run twice on every model, on the drawing and on the real thing. All 68 models fit the 17-number description to better than 0.07 percent of chord. The tunnel's turbulence level isn't recorded anywhere, so n_crit was swept from 5 to 11 on this data; 9 came out best.

What the sample really is. That headline of 94 airfoils is a union: 55 UIUC plus 54 Princeton minus the 15 that appear in both. The 68 Princeton models include several sections built twice by different builders, among them the E387, E374, E205, E214, S2091, S3021, S4061, HQ 2/9 and FX 63-137, plus three repeat runs of the same model. The 9,130 clean points sit inside 667 polars, 356 from UIUC and 311 from Princeton, and every point in one polar is a single angle sweep on a single model at a single speed. Those points are not 9,130 independent facts. The independent unit is the airfoil, and section 3.4 treats it that way.

2.7 Splitting XFoil's error from the network's

NeuralFoil copies XFoil. So when it disagrees with a wind tunnel, two separate errors are stacked: XFoil's disagreement with the tunnel, plus the network's disagreement with XFoil. Only the second one is really the network's.

To separate them, a headless XFoil 6.99 was run at all 9,130 clean conditions. It got exactly the 17-number shape NeuralFoil saw, run viscous with n_crit = 9 and free transition, at the measured speed and angles. Each polar sweeps outward from 0 degrees in half-degree steps with the measured angles slotted in, so every point starts from its neighbour's answer. XFoil's viscous solver needs that warm start at these speeds. Points where it failed to converge are recorded as missing and counted. The three comparisons, NeuralFoil against tunnel, XFoil against tunnel, and NeuralFoil against XFoil, are then made on the points that did converge.

2.8 Statistics

Honest error bars. Every headline number gets a 95 percent interval from a bootstrap that resamples airfoils, not points. Each airfoil comes with all of its points attached, 2,000 times over. The naive point bootstrap runs alongside so the two can be compared. In the pooled version, an airfoil tested in both tunnels counts as two clusters, because those really are two different physical models in two different labs.

Where a correlation lives. The link between confidence and drag error gets split three ways. First, the correlation between airfoil averages. Second, what's left after subtracting each airfoil's average from both sides. Third, the same split done at the level of single polars.

A fitted error model. Expected drag error is fitted as a Gamma model with a log link, solved by iteratively reweighted least squares. Its inputs are log10(1.001 minus confidence), log10(Re / 100,000), angle of attack and its square, maximum camber and maximum thickness. Every number reported for it is out of sample. That means 10-fold cross-validation with whole airfoils held out, where an airfoil in both tunnels is held out of both at once, plus training on one tunnel and testing on the other in both directions. The baselines are a flat constant and the six-bin confidence table from section 3.1, both refitted on each training fold.

---

3. Results

3.1 The test against the UIUC tunnel

Start with the reference airfoil. The Eppler E387 has been measured in more tunnels than any other slow-speed section, so it's the natural place to begin. Against its 112 clean measurements at six speeds, with errors given as absolute values averaged, and bias as the signed average:

Re	Lift error (ΔCL)	Lift error, relative	Drag error (%)	Drag bias (%)	Confidence
100,000	0.057	11%	22	−5	0.88
200,000	0.030	5.4%	12	0	0.82
300,000	0.025	4.9%	12	−1	0.81
350,000	0.024	4.7%	12	−3	0.81
460,000	0.023	5.1%	12	−4	0.81
500,000	0.023	5.1%	12	−5	0.81

Lift is good to about 5 percent from Re = 200,000 up, and 11 percent at Re = 100,000. Drag is the harder one. It sits near 12 percent from Re = 200,000 up and jumps to 22 percent at Re = 100,000, where the separation bubble parks itself right in the middle of the low-drag range. There's no big

one-way drag bias for the E387, just 3 percent low overall. Look at the drag curves and you'll see both NeuralFoil and XFoil missing that bubble drag.

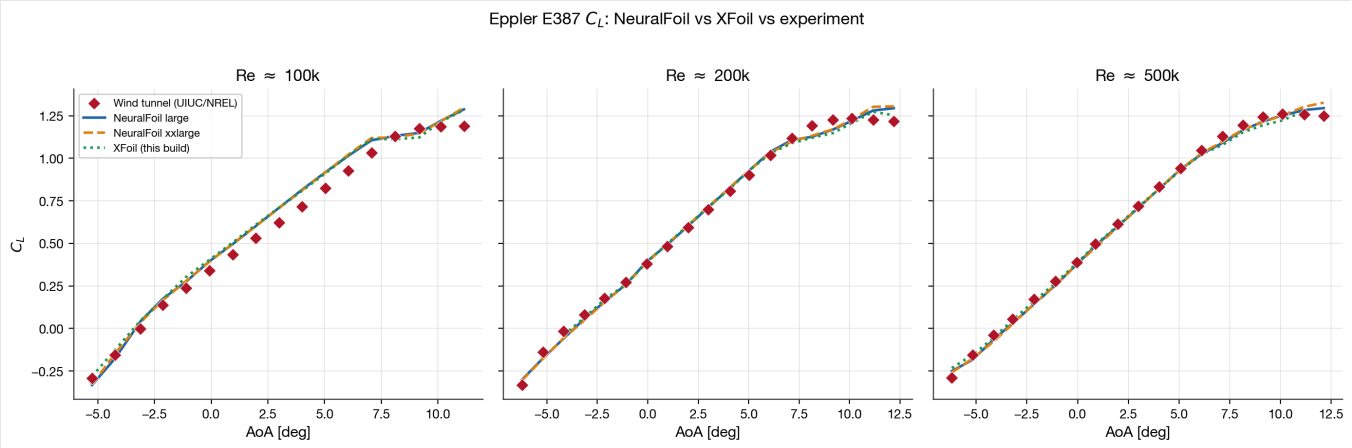


Figure 1. Lift coefficient versus angle of attack for the clean Eppler E387 at Re of about 100k, 200k and 500k: NeuralFoil, headless XFoil, and wind-tunnel measurement (UIUC; Selig & McGranahan 2004).

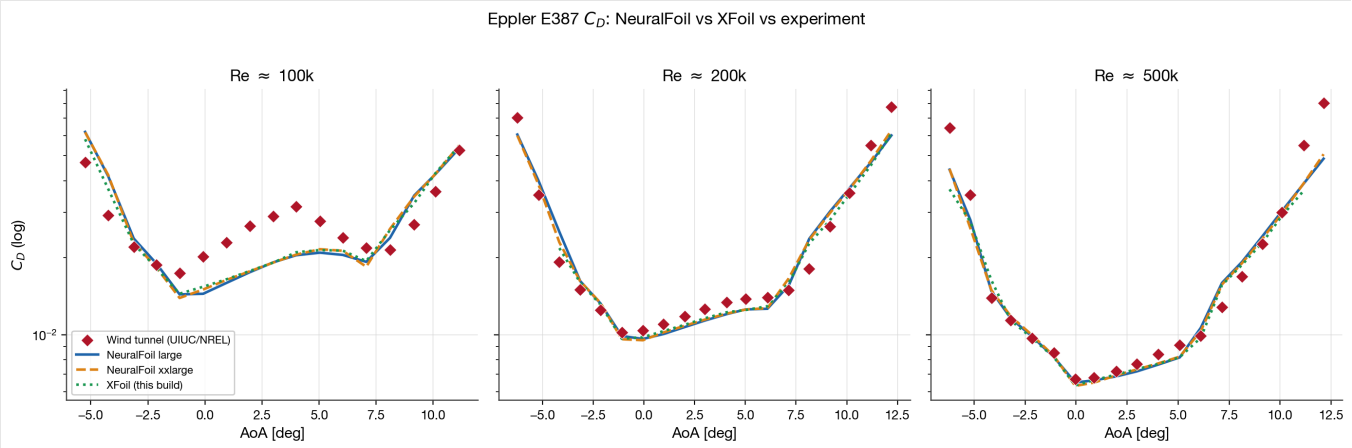


Figure 2. Drag coefficient (log scale) for the same runs. At Re of about 100k the measured drag sits above both predictions across the low-drag range: the laminar separation bubble that both NeuralFoil and XFoil under-predict.

Now all 55 airfoils. The E387 is a well-behaved shape. Across the full benchmark set, 4,428 clean measurements, the picture widens:

Re	Points	Lift error (ΔCL)	Drag error, mean (%)	Drag error, median (%)	Drag bias (%)	Confidence
60,000	565	0.084	22	15	+13	0.92
100,000	1,105	0.085	14	10	+2	0.91
200,000	1,183	0.071	10	7	0	0.91
300,000 to 400,000	1,187	0.061	10	7	−4	0.91

Re	Points	Lift error (ΔCL)	Drag error, mean (%)	Drag error, median (%)	Drag bias (%)	Confidence
400,000 to 500,000	388	0.049	9	7	−7	0.95
All	**4,428**	**0.072**	**12**	**8**	**0**	**0.91**

The three coefficients behave differently.

1. **Drag is fine from Re = 100,000 up, and gets worse below that.** The typical drag error is 7 percent from Re = 200,000 up, 10 percent at Re = 100,000, and 15 percent at Re = 60,000, where NeuralFoil reads about 13 percent high. The bias flips sign along the way: high at the slow end, about 7 percent low at the fast end, near zero overall. The bigger `xxlarge` network is no better than `large`.
2. **The lift error hides in the very cambered shapes.** That average of 0.072 splits in two. It's 0.066 for the 45 airfoils under 5 percent camber and 0.113 for the ten above it, including the S1223, S1210, FX 63-137 and NACA 6409. Those are the high-lift shapes sitting near the edge of what XFoil handles well.
3. **L/D is worse than either piece, because it divides one by the other.** Across the 3,151 clean points with CL above 0.2, NeuralFoil's L/D is off by 15 percent in a typical case and 21 percent on average. And the error leans one way. NeuralFoil reads 15 percent high on average, worst at Re = 100,000 where it reads 20 percent high. Every L/D chart in this paper is drawn with that band on it, and every L/D number quoted anywhere here should be read as optimistic.

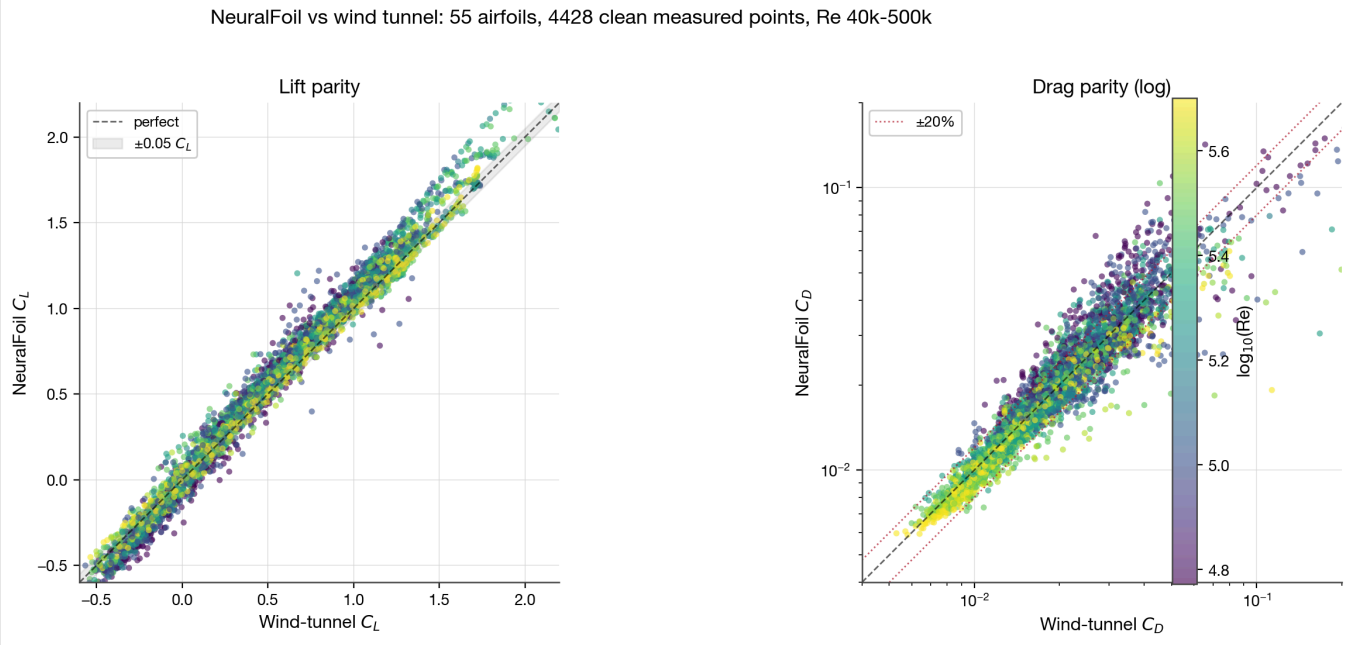


Figure 3. NeuralFoil versus wind tunnel for all 4,428 clean benchmark points on 55 airfoils, coloured by Reynolds number. Lift scatters within about 0.05 for most points; drag within about 20%.

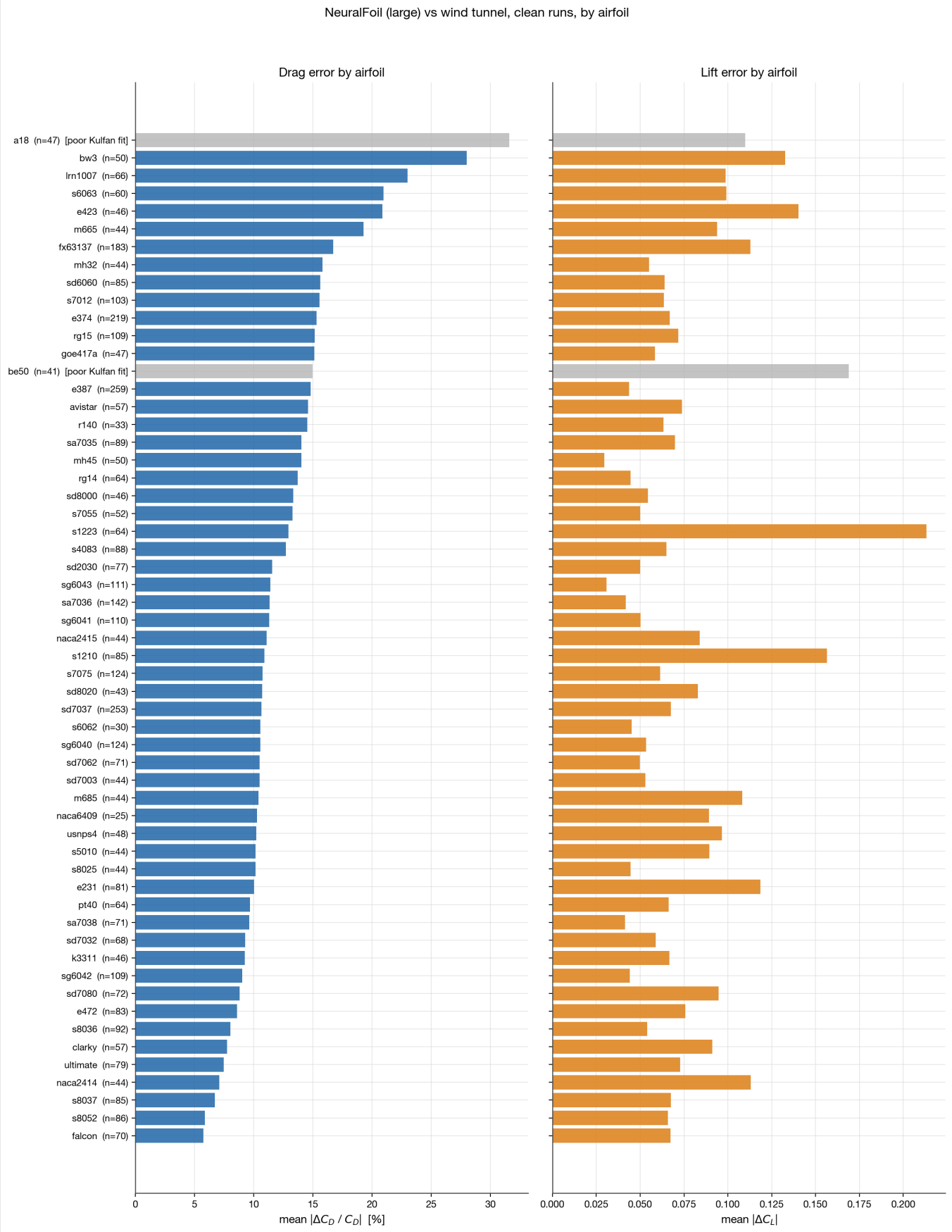


Figure 4. Mean drag and lift error by airfoil, clean runs. Grey bars are the two airfoils excluded from the benchmark because the 17-parameter shape description cannot reproduce them.

Trip strips. Fifteen of the benchmark airfoils were also run with a zigzag trip near the leading edge on both surfaces. A trip forces the boundary layer turbulent early, the way dirt or insects would on a real blade. Against those 885 tripped measurements, NeuralFoil run normally reads 15 percent low on drag, which is exactly what it should do, since nothing told it a trip was there. Run it again with transition forced at the trip positions and the bias drops to 6 percent while the lift error falls by a quarter. On the three airfoils from the 2004 report, where the trip is documented most carefully, it falls from 19 percent to 2 percent and the lift error halves. So NeuralFoil's transition inputs do what they say, which had not been checked against experiment either.

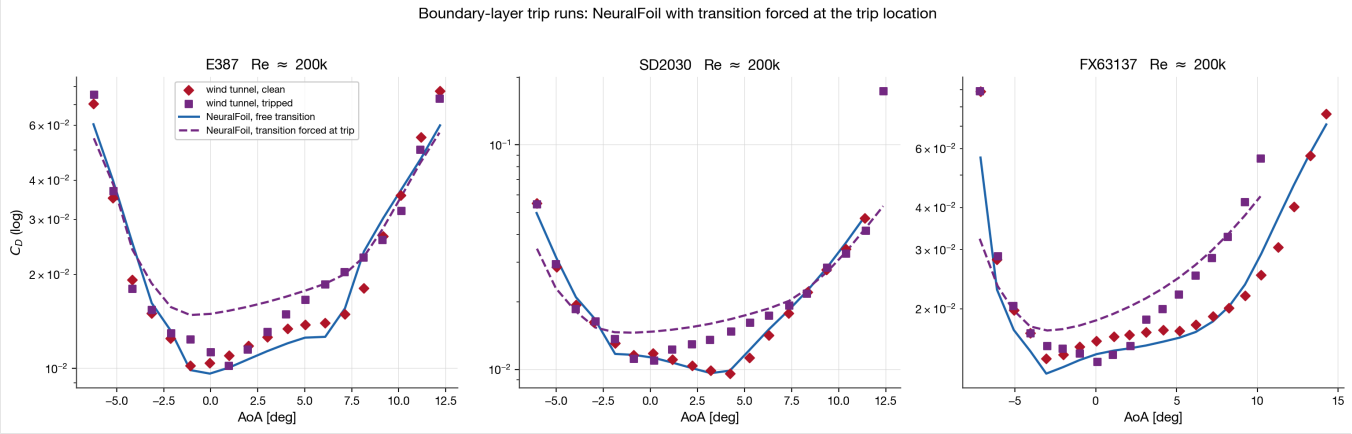


Figure 5. Boundary-layer trip runs near Re = 200k. Diamonds: clean measurement; squares: tripped measurement. Running NeuralFoil with transition forced at the trip location (dashed) reproduces the tripped drag that a free-transition run (solid) misses.

Through the stall. The archive's lift runs sweep out to 15 to 24 degrees, past maximum lift, stepping up and then back down again. That gives 298 clean runs on the 55 benchmark airfoils, at Re = 30,000 to 500,000. Against them, NeuralFoil reads maximum lift 0.06 high on average, about 6 percent. It puts the stall about 1 degree early, and within 1.5 degrees in a typical run. Past the stall its lift error nearly doubles, from 0.08 to 0.15, while its confidence falls from 0.89 to 0.52. So the warning light works there too. Pitching moment is rougher, off by 0.018 against typical values near 0.08. The worst offenders are the S1223, USNPS4, SD7080 and PT40, each read 0.15 to 0.17 high on maximum lift. One thing no steady model can reproduce is stall hysteresis, the gap between the lift going up and the lift coming back down. It's small in a typical run at 0.05, but it passes 0.4 in the worst tenth, and the lift curves show how far the two branches can differ.

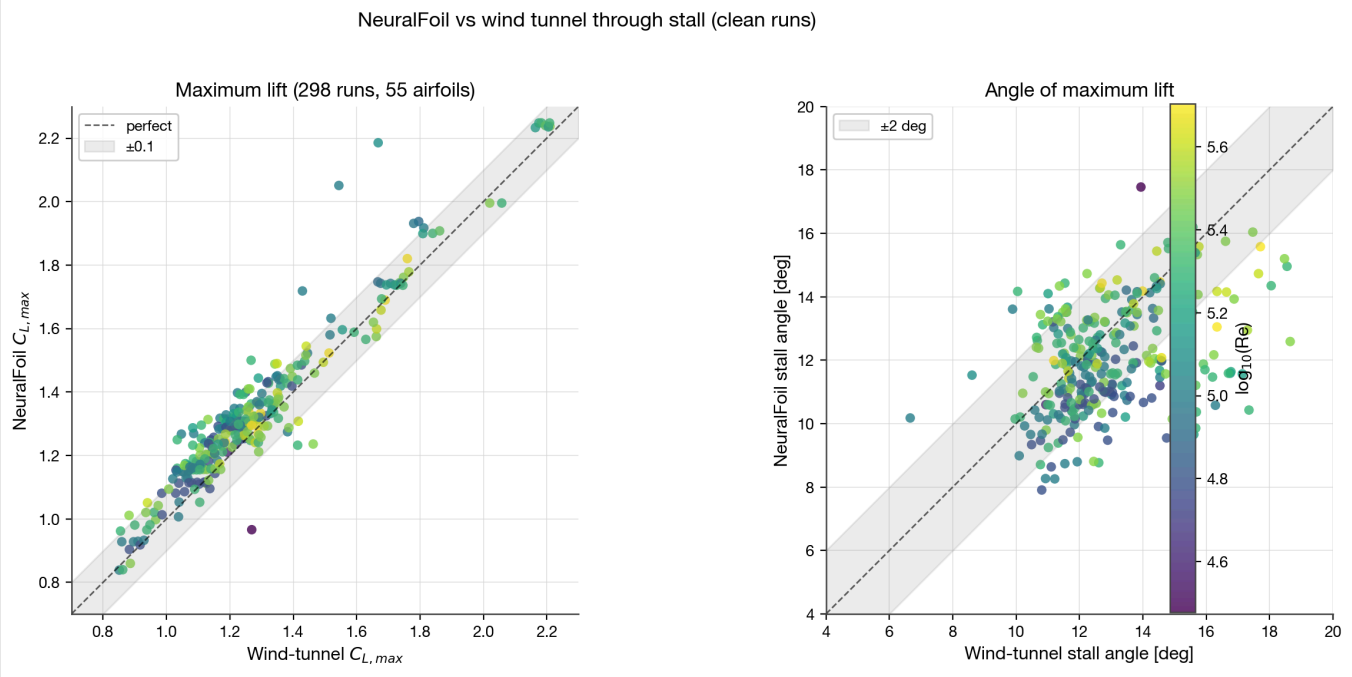


Figure 6. Maximum lift and the angle it occurs at, NeuralFoil versus wind tunnel, for the 298 clean lift sweeps that passed through stall.

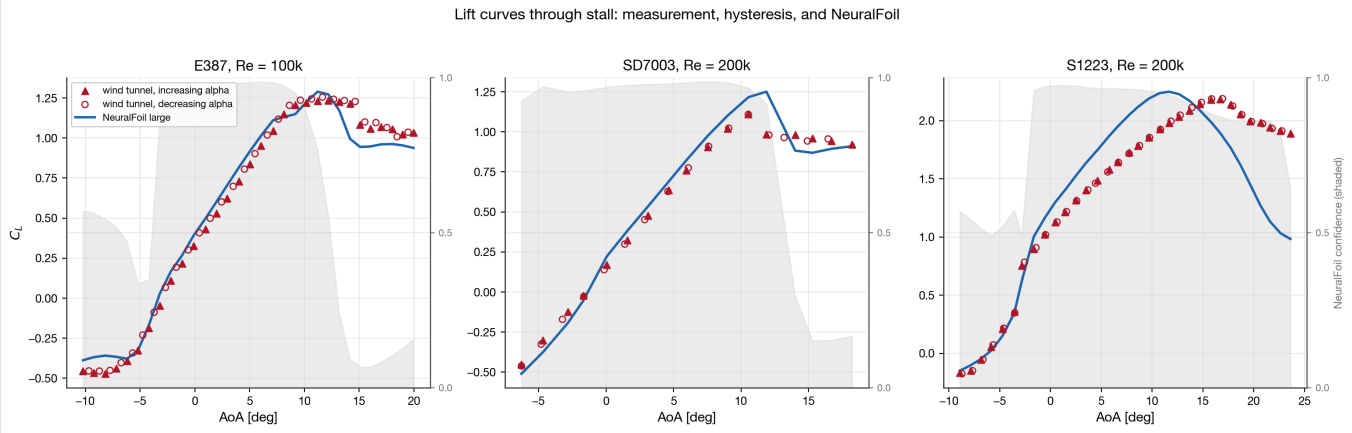


Figure 7. Three lift curves through stall. Triangles: increasing angle; open circles: decreasing angle (the gap is stall hysteresis); line: NeuralFoil; grey band: NeuralFoil's confidence, which collapses past the stall.

What the confidence score actually tells you. Sort the 3,995 attached-flow benchmark points into bins by confidence:

Confidence	Points	Lift error (ΔCL)	Drag error, mean (%)	Drag error, median (%)
below 0.5	222	0.069	32	27
0.5 to 0.7	102	0.085	31	27
0.7 to 0.8	62	0.096	26	20
0.8 to 0.9	157	0.074	21	17
0.9 to 0.95	387	0.082	15	11

Confidence	Points	Lift error (ΔCL)	Drag error, mean (%)	Drag error, median (%)
above 0.95	3,065	0.071	9	7

It's a drag warning and nothing else. Drag error slides steadily from 32 percent down to 9 percent as confidence rises, with r = −0.43 across all points, the same sign in every speed band and at positive and negative angles alike. At the airfoil level, the shapes NeuralFoil is least sure about are the ones it gets most wrong, with a rank correlation of −0.65. Lift error, meanwhile, is flat across every bin, and its correlation with confidence is exactly zero. So when NeuralFoil says it isn't sure, believe it about drag, and therefore about L/D, but not about lift. Section 3.4 puts proper error bars on these correlations and section 3.3 works out what the score is really detecting. Section 3.9 uses the score inside the optimizer.

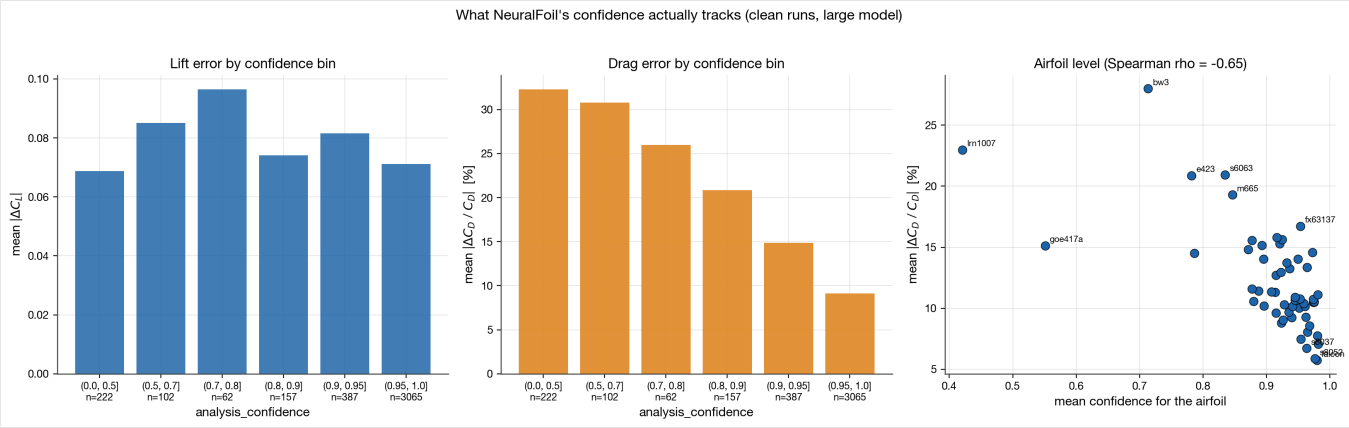


Figure 8. What NeuralFoil's confidence score tracks in the UIUC data. Left: lift error is flat across confidence bins. Middle: drag error falls from 32% to 9% as confidence rises. Right: the same relationship at the airfoil level (section 3.4 shows this panel does not replicate in the Princeton data, and why).

3.2 The same test in a second wind tunnel

Testing against one lab can't separate the model's error from that lab's error. So everything in section 3.1 was run again on the Princeton data: 54 airfoils, 68 wind-tunnel models, 4,702 clean measurements at Re = 60,000 to 300,000. Different decade, different tunnel, different instruments, different people building the models. Here's what came back, with NeuralFoil run on each model's measured shape:

Re	Points	Lift error (ΔCL)	Drag error, mean (%)	Drag error, median (%)	Drag bias (%)	Confidence
60,000	759	0.101	17	13	+12	0.96
100,000	1,018	0.092	12	8	+3	0.95
150,000	810	0.082	10	7	+1	0.95
200,000	1,157	0.081	9	6	−2	0.95

Re	Points	Lift error (ΔCL)	Drag error, mean (%)	Drag error, median (%)	Drag bias (%)	Confidence
300,000	958	0.077	9	7	−4	0.96
All	**4,702**	**0.086**	**11**	**8**	**+1**	**0.96**

The UIUC numbers hold up. Drag is off by 11 percent on average and 8 percent in a typical case, against 12 and 8 at UIUC. It degrades toward Re = 60,000 the same way, and the bias flips sign the same way. That flip runs from 12 percent high at Re = 60,000 to 4 percent low at 300,000, next to 13 percent high and 7 percent low at UIUC. L/D is read 15 percent high with a typical error of 15 percent and an average error of 21 percent, which is the same three numbers as the UIUC set. Lift is a bit worse, 0.086 against 0.072, and leans the same way, about 0.04 to 0.05 high in both tunnels. `xxlarge` is again no better. Dropping n_crit to 7 kills the over-prediction at Re = 60,000 but adds a 5 percent under-prediction everywhere else, and 11 doubles the error.

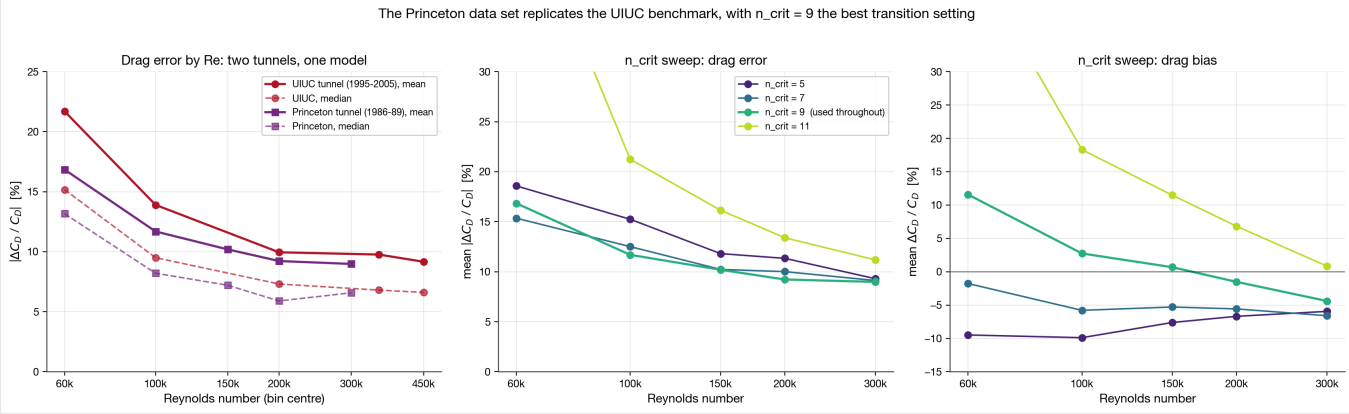


Figure 9. Left: NeuralFoil's drag error by Reynolds number against the UIUC tunnel (red) and the Princeton tunnel (purple), mean and median. Middle and right: sweeping the transition parameter n_crit on the Princeton data; 9, the value used throughout, gives the lowest error and the smallest bias.

The two archives disagree by about as much as NeuralFoil disagrees with either one. Fifteen airfoils were measured in both. Each Princeton polar is assigned to exactly one UIUC polar: nearest Reynolds number within 15%, then largest angle overlap and stable provenance fields as deterministic tie-breaks. This gives 87 polar pairs and 1,241 unique matched points:

Re	Points	Airfoils	UIUC vs Princeton, drag (%)	NeuralFoil vs UIUC (%)	NeuralFoil vs Princeton (%)	UIUC vs Princeton, lift (ΔCL)
60,000	185	11	16	19	16	0.053
100,000	288	15	14	12	11	0.046
150,000	161	8	11	9	10	0.037
200,000	332	15	9	8	7	0.044

Re	Points	Airfoils	UIUC vs Princeton, drag (%)	NeuralFoil vs UIUC (%)	NeuralFoil vs Princeton (%)	UIUC vs Princeton, lift (ΔCL)
300,000	275	13	6	7	7	0.040
All	**1,241**	**15**	**10.9**	**10.5**	**9.7**	**0.044**

The two archives differ on drag by 10.9 percent, with an airfoil-cluster bootstrap interval of 9.1 to 12.9 percent and UIUC reading 4.8 percent higher. NeuralFoil differs from each of them by 9.7 to 10.5 percent on those same points. On 52 percent of them it lands closer to the Princeton value than the UIUC measurement does. And from Re = 200,000 up, all three numbers collapse to 6 to 8 percent. That is the level at which these two archives stop agreeing, but it is a different quantity from the repeatability of either tunnel. The physical models, recorded geometries, dates and data-reduction pipelines also differ. Jewel Barlow, who directs the Glenn L. Martin Wind Tunnel, put within-facility repeatability at a fraction of 1 percent for a model rigid enough that its surface varies by a small fraction of a millimetre, and said the differences between facilities are much larger, because of the number of details that cannot be duplicated from one to the next. The Princeton repeat runs below agree to 1.5 and 2.0 percent at Re = 300,000, so the 7 percent at that speed is a statement about two archives and not about one tunnel. A model that beat 7 percent could still be demonstrated against either archive on its own. What could not be settled is which of the two it should be scored against. Lift is a different story. The two archives agree with each other at 0.044, better than NeuralFoil agrees with either at 0.069 and 0.080, so the lift error is the model's.

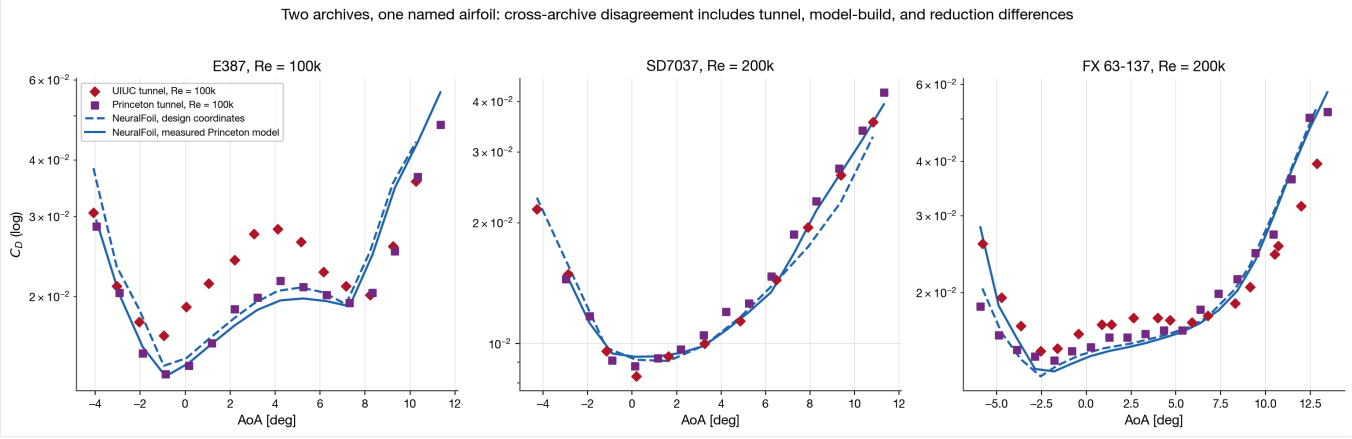


Figure 10. The same airfoil in two wind tunnels. Diamonds: UIUC; squares: Princeton; dashed: NeuralFoil on the design coordinates; solid: NeuralFoil on the measured shape of the Princeton model. For the E387 at Re = 100k the two experiments differ by more than either differs from the model.

How tight is that limit? The Princeton archive holds two models that were mounted and run a second time in the same tunnel by the same builder, the E387A and the SD7003. Across 95 matched points those repeats disagree with themselves by 3.7 percent in drag and 0.009 in lift. So there are three numbers stacked on top of each other: 3.7 percent is what one experiment does to itself, 10.9 percent is what these two archives do to each other, and 11.7 percent is what NeuralFoil does against

them. The model error is three times the repeatability, so it is not hiding inside the measurement noise. It is also no larger than the disagreement between the two archives. That is useful external context, not a pure tunnel-reproducibility ceiling. The floor is not flat either: the repeat runs agree to 1.5 and 2.0 percent at Re = 300,000 and to 4.7 and 8.3 percent at Re = 100,000, the same Reynolds trend the model errors follow.

One thing that 3.7 percent is not. It is repeatability, which is precision: the same model, the same tunnel, the same builder, mounted and measured twice. It says nothing about whether the balance was reading the right number in the first place. A calibration offset repeats perfectly every time, so it would sit inside all three of these figures and never show up in any of them. Accuracy and precision are different quantities and only the second one can be recovered from repeat runs. Read 3.7 percent as a floor on the random part of the experimental error, not as a bound on the whole of it.

It is not the instrument either. Asked what a tunnel should manage, Barlow gave a fraction of 1 percent on force and moment measurements for a rigid model, commonly a small fraction of that. The Princeton repeats sit above that everywhere they can be checked. What 3.7 percent measures is not the balance. It is what happens when a hand-built model is taken off the mount and put back on, at Reynolds numbers where forces are sensitive to small differences in surface detail and that sensitivity interacts with the freestream turbulence. For scoring a model against an archive that is the more useful of the two figures, because anyone using the archive inherits the remount and the surface finish along with the numbers.

Build error hides inside the model error. For the 56 models with both sets of coordinates, the built shape differs from the drawing by 0.22 percent of chord in a typical case and up to 0.68 percent for the E387B. That's exactly the range assumed for the build study in section 2.5. Running NeuralFoil on the measured shape instead of the drawing drops the drag error from 12.8 to 11.1 percent and the lift error from 0.091 to 0.081, and it helps 46 of the 56 models. The shape change on its own moves NeuralFoil's drag by 5 percent and its lift by 0.03, which is about half the size of what error remains. For the worst-built model, the E387B, drag error falls from 13 to 8 percent. So part of what looks like model error in every benchmark of this kind, section 3.1 included, is really the wind-tunnel model not being the airfoil it was meant to be.

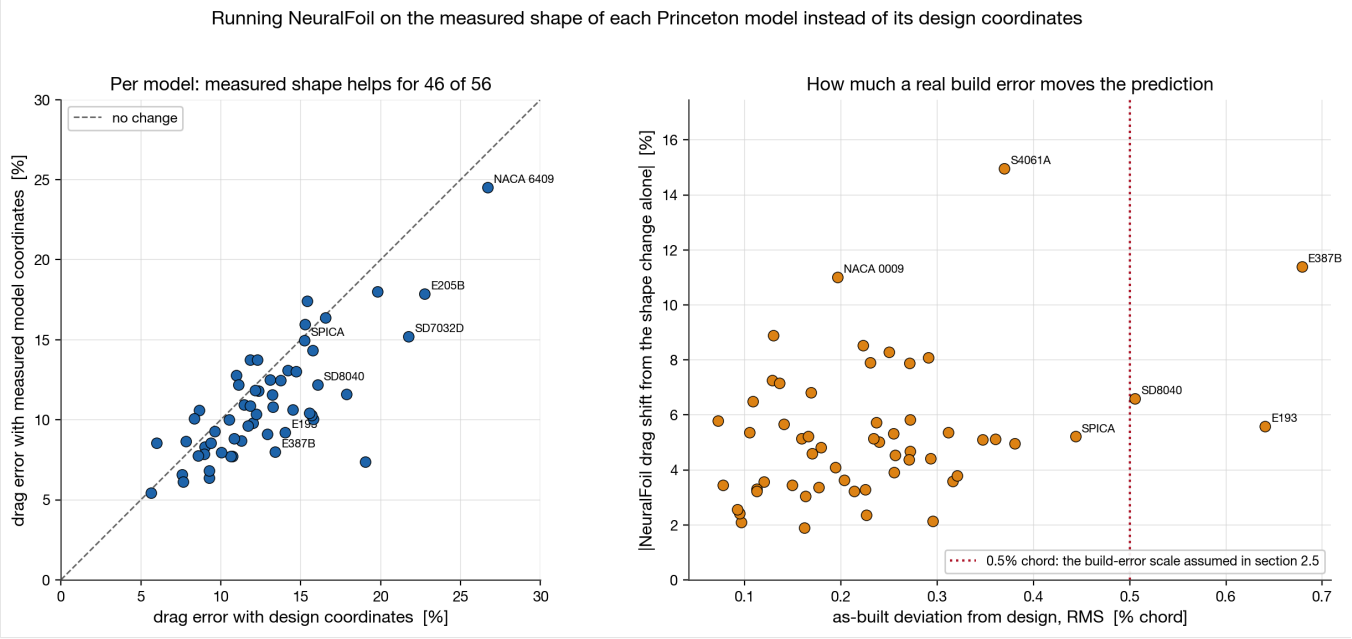


Figure 11. Left: drag error per Princeton model with design versus measured coordinates; points below the line improved. Right: the as-built deviation of each model against how much that deviation alone moves NeuralFoil's drag; the dotted line is the 0.5% chord build-error scale used in section 2.5.

Trip strips again. Thirty-four tripped runs on 18 models, 1,744 points, mostly a single strip on the upper surface somewhere between 20 and 70 percent of chord. That's a different situation from the leading-edge trips in section 3.1. NeuralFoil run normally shows no drag bias here, because a strip that far back sits near where the flow would have gone turbulent anyway. Forcing transition at the strip still helps, dropping drag error from 13.5 to 11.8 percent and lift error from 0.084 to 0.071. For the strips closest to the leading edge, at 30 percent of chord or less, drag error falls from 15 to 13 percent.

Through the stall again. Against the 114 clean lift sweeps at Re = 55,000 and up that passed the stall, on 35 airfoils, NeuralFoil reads maximum lift 0.11 high on average. That's worse than the 0.06 of the UIUC set. It places the stall within 1.4 degrees in a typical run. Down at the very slowest speeds, 20,000 to 50,000 on the MB253515, it finds no stall at all before 20 degrees while the measurement stalls at 12 to 15.

The confidence score again. At the extremes it repeats: drag error is 30 percent below a confidence of 0.5 and 10 percent above 0.95, and lift error stays flat at r = −0.03. Two things don't repeat. The correlation with drag error is weaker, r = −0.26 against −0.43, and the middle bins sit flat at 22 to 25 percent instead of stepping down. And at the airfoil level, the rank correlation between an airfoil's average confidence and its average drag error is −0.08 and not significant, against −0.65 in the UIUC set. Section 3.4 explains why.

3.3 So whose error is it, XFoil's or the network's?

XFoil converged at 8,814 of the 9,130 clean conditions, or 96.5 percent. NeuralFoil needs no solver run, so its error against the tunnels is reported on all 9,130 points: that column is the 11.7 percent

section 3.4 carries. The two comparisons that involve XFoil only exist where XFoil converged, so the three-way head-to-head is made on that subset and gets its own column. The two NeuralFoil columns differ because the points XFoil dropped are the hard ones, which is the subject of the third paragraph below.

Tunnel	Re	Clean points	XFoil converged	NeuralFoil vs tunnel, all points (%)	NeuralFoil vs tunnel, converged (%)	XFoil vs tunnel (%)	NeuralFoil vs XFoil (%)
UIUC	60,000	565	543	21.7	20.5	22.0	3.3
UIUC	100,000	1,105	1,056	13.9	12.5	14.9	3.9
UIUC	200,000	1,183	1,125	10.0	9.4	10.8	3.6
UIUC	300,000 to 400,000	1,187	1,107	9.8	9.3	9.9	2.5
UIUC	400,000 to 500,000	388	377	9.2	8.9	9.2	1.7
Princeton	60,000	759	744	16.8	16.7	17.3	2.9
Princeton	100,000	1,018	1,002	11.7	11.7	12.3	2.6
Princeton	150,000	810	796	10.2	9.7	10.1	2.3
Princeton	200,000	1,157	1,131	9.2	9.1	9.4	2.3
Princeton	300,000	958	933	9.0	8.8	9.2	2.3
Both	**all**	**9,130**	**8,814**	**11.7**	**11.2**	**12.1**	**2.8**

The last three columns are the only ones that may be compared with each other, because only they are computed on the same points.

It's XFoil's. The network differs from the solver it copies by 2.8 percent in drag on average, 1.7 percent in a typical case, and by 0.011 in lift. That's about a quarter of its disagreement with the tunnels. The two signed errors track each other at $r = 0.95$; the centered simple regression of NeuralFoil error on XFoil error has $R^2 = 0.895$. In every speed band of both tunnels the network's own share stays under 4 percent, including at Re = 60,000 where the total error runs 17 to 22 percent. So the Re = 60,000 over-prediction, the bias flipping sign with speed, and the high-camber lift error in section 3.1 are all XFoil's behaviour, copied faithfully. XFoil's own lift error against the tunnels is 0.082, next to NeuralFoil's 0.079.

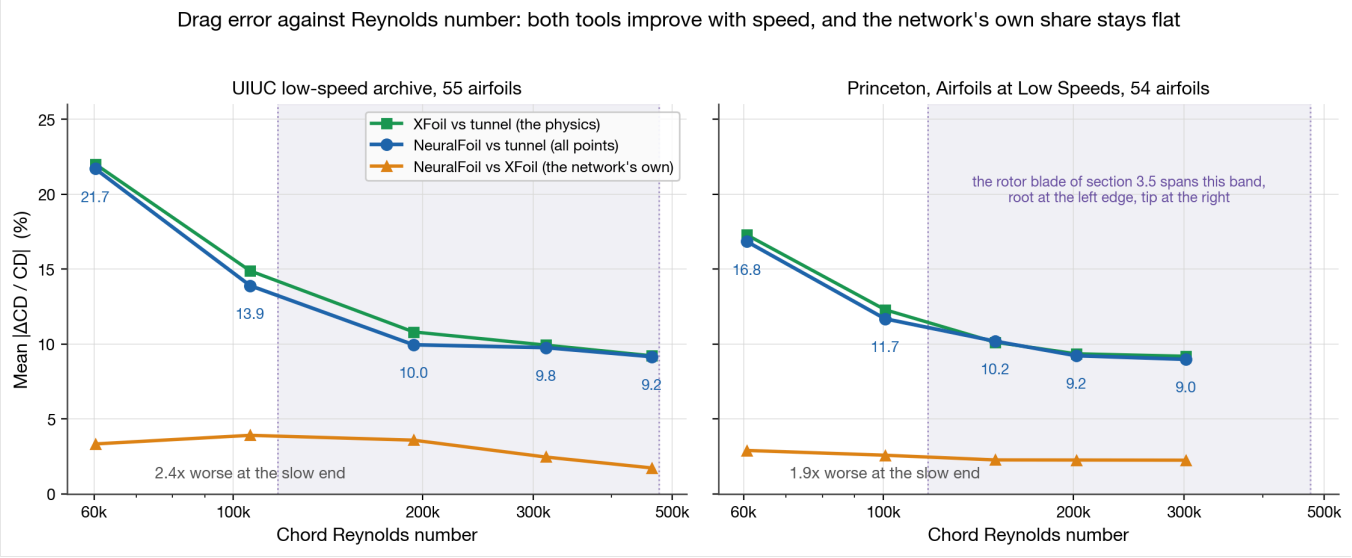


Figure 12. Mean drag error against Reynolds number in both archives, with XFoil's error and the network's own emulation error separated. Both tools improve steadily as the air speeds up, by a factor of 2.4 in the UIUC set and 1.9 in the Princeton set, while the part contributed by the network stays flat and small at every speed. The shaded band is the Reynolds range the rotor blade of section 3.5 spans, root at its left edge and tip at its right.

The Reynolds trend is the finding, not the table. Both tools get steadily better as the air speeds up, by a factor of 2.4 across the UIUC set and 1.9 across the Princeton set, while the part the network adds stays flat and small at every speed. That trend is not an abstraction: it is a position along a blade. The rotor of section 3.5 spans 118,000 at its root cut to 475,000 at its tip, which is most of this axis, so the inboard third of that blade is the least trustworthy part of it and the outboard two thirds are the best. Section 3.5 puts numbers on what that costs.

And NeuralFoil is slightly closer to reality than XFoil is. On 55 percent of points the network lands nearer the measurement. Comparing the two on the points where both ran, its average drag error is lower in every band: 11.2 against 12.1 percent overall, and 12.5 against 14.9 at Re = 100,000 in the UIUC tunnel. That makes sense. Fit a network to hundreds of thousands of XFoil polars and you smooth out XFoil's point-to-point scatter, and some of that scatter is error. The 3.5 percent of conditions where XFoil wouldn't converge are worth a look too. NeuralFoil's average drag error there is 25 percent and its average confidence 0.57, against 11 percent and 0.95 where XFoil did converge. XFoil converges at only 56 percent of the conditions where NeuralFoil's confidence is below 0.5, and 99 percent where it's above 0.95.

What the confidence score is really detecting. Sort the converged attached-flow points by confidence and show all three errors at once:

Confidence	Points	NeuralFoil vs tunnel (%)	XFoil vs tunnel (%)	NeuralFoil vs XFoil (%)	Lift, NeuralFoil vs XFoil (ΔCL)
below 0.5	158	32	41	10	0.034
0.5 to 0.7	98	28	34	10	0.029

Confidence	Points	NeuralFoil vs tunnel (%)	XFoil vs tunnel (%)	NeuralFoil vs XFoil (%)	Lift, NeuralFoil vs XFoil (ΔCL)
0.7 to 0.8	79	26	32	10	0.030
0.8 to 0.9	213	21	27	8	0.021
0.9 to 0.95	631	15	17	5	0.017
above 0.95	6,742	9	10	2	0.009

The score was trained to report how unsure the network is about XFoil, and it does that job: the copying error climbs from 2 to 10 percent as confidence drops. But look closer. It tracks XFoil's error against the tunnel more tightly, at r = −0.40, than it tracks the network's error against XFoil at r = −0.33, or against the tunnel at r = −0.33. Low confidence marks the conditions where XFoil's answer is shaky or missing altogether, meaning separation bubbles and post-stall flow. Those are the same conditions where XFoil's physics is wrong. So the score warns about the physics as well as the copying, which is more than it was built to do, and it's the reason it works as a stand-in for real drag error. Section 3.9 relies on that.

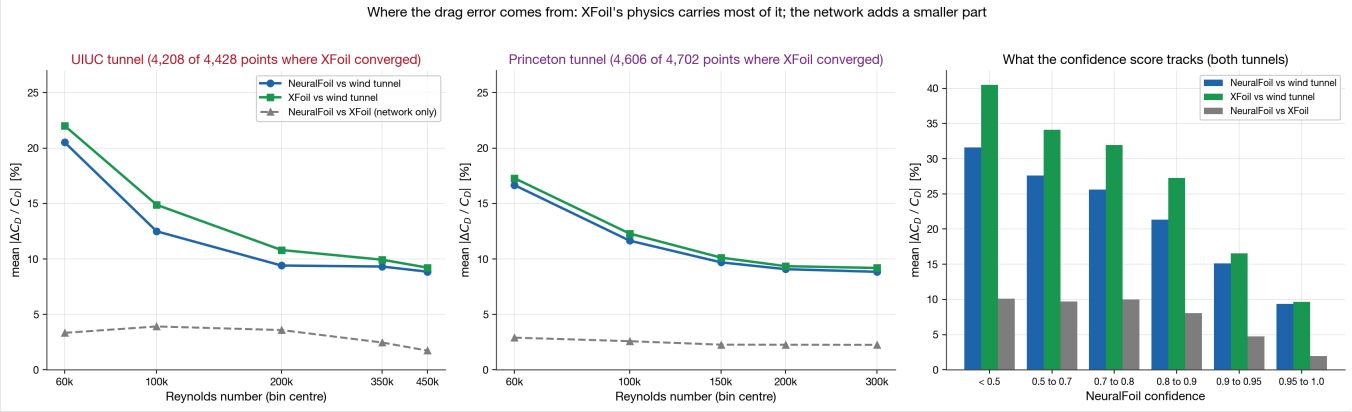


Figure 13. Where the drag error comes from. Left and middle: mean drag error by Reynolds number in each tunnel for NeuralFoil versus tunnel (blue), XFoil versus tunnel (green) and NeuralFoil versus XFoil (grey, the network's own contribution), on the points where XFoil converged. Right: the same three errors by confidence bin, both tunnels pooled; the score tracks XFoil's error against experiment at least as closely as the network's error against XFoil.

Does the error follow the length of the laminar run? Nikhil Khobragade (IIT Madras) proposed this test after the n_crit result in section 3.2, where dropping n_crit to 7 helps at Re = 60,000 and hurts everywhere else. That implies the flow wants more freestream turbulence at low speed than at high speed. One reading is that a tunnel running slower genuinely has a higher turbulence intensity, since intensity is a fluctuation divided by a mean speed. The other is that e^N is being asked for something it does not carry: it predicts where transition starts, not how long a separation bubble runs or when it bursts, and at these speeds the bubble is most of the drag. On the second reading the drag error should grow with the length of the laminar run, which XFoil reports at every point as a transition location on each surface.

It does not grow with it. It shrinks.

Laminar run length and drag error are both governed by Reynolds number, and run length is close to a restatement of angle of attack, so the test holds log(Re) and |alpha| fixed and clusters the bootstrap on the airfoil. Against the suction-side laminar run the partial rank correlation is −0.17 pooled and negative in every Reynolds band but the highest:

Re band	Points	Partial rho	95% interval
40,000 to 80,000	1,287	−0.28	−0.35 to −0.21
80,000 to 130,000	1,954	−0.26	−0.32 to −0.19
130,000 to 180,000	1,089	−0.21	−0.30 to −0.14
180,000 to 250,000	2,067	−0.18	−0.24 to −0.11
250,000 to 550,000	2,417	−0.02	−0.10 to 0.05
all	8,814	−0.17	−0.21 to −0.12

A longer laminar run goes with a smaller drag error, and the effect is largest at exactly the Reynolds numbers where the bubble argument said the error should be largest. Quartiles say the same without a correlation: in the 40,000 to 80,000 band the mean drag error runs 32.0, 18.3, 12.9 and 13.9 percent from the shortest-run quartile to the longest. Absolute error in drag counts falls the same way, 0.0106 to 0.0026, so it is not an artefact of dividing by a small drag.

One warning about the variable, because it nearly fooled this paper. The bubble sits on the suction side, and below zero lift the suction side is the lower surface. Run the identical test against the *upper* surface regardless of the sign of lift, and a positive correlation appears at negative incidence, +0.19 in the 40,000 to 130,000 band, which is precisely the result the hypothesis predicts. Choose the surface by the sign of lift instead and the same points give −0.22. The disagreement is confined to the alpha window from −3 to 0 degrees, where 336 of 599 points sit below zero lift; from 0 degrees upward the two definitions agree to three decimal places. The positive version of this finding is a property of the variable, not of the flow.

So the bubble reading of the n_crit result is not supported here, and the evidence points the other way.

Nor is the other reading. Khobragade, asked directly how much the turbulence intensity of a low-speed tunnel moves between 10 and 30 m/s, answered that it stays approximately the same, with a minor decrease that can be neglected over that range. Put a number on what the n_crit trend would need instead. Mack's correlation, n_crit = −8.43 − 2.4 ln(Tu), puts n_crit = 9 at a turbulence intensity of 0.070 percent and n_crit = 7 at 0.161 percent. Going from one to the other as the tunnel slows asks the intensity to rise by a factor of 2.3. A negligible drift cannot deliver a doubling, so the best-fit n_crit is not tracking the turbulence of the tunnel.

That leaves both candidate explanations dead and the observation standing. At Re = 60,000 on the Princeton set, n_crit = 9 over-predicts drag by 11.6 percent and n_crit = 7 brings the bias to −1.8; by Re = 300,000 the ordering has gone. Whatever n_crit is absorbing down there, it is neither the length of the laminar run nor the freestream turbulence, and this paper cannot name it. What it can say is that a single n_crit calibrated at one Reynolds number should not be trusted at another, and that the fitted value is a free parameter soaking up a drag error rather than a measurement of the flow.

What the negative correlation means in its own right is a further question this paper does not answer. A short laminar run marks conditions where the flow has already gone turbulent or separated near the leading edge, and the solver is worst there. That is a statement about where XFoil struggles, not about the bubble.

3.4 The spread, and how sure the numbers are

One number hides the shape. A mean of 11.7 percent reads as though most predictions land near 11.7 percent. They don't. Across all 9,130 clean points the drag error is strongly right-skewed, and the lift error with it:

Statistic	Drag error (%)	Lift error (ΔCL)
25th percentile	3.6	0.035
Median	7.8	0.069
75th percentile	14.8	0.110
90th percentile	26.6	0.154
Mean	11.7	0.079

Three predictions in five land within 10 percent of the measurement and five in six within 20 percent, but one in thirteen is more than 30 percent out. The mean sits above the 60th percentile, which is what a long tail does to an average. Quote the mean alone and you overstate the typical case and understate the bad one. A design that needs a margin should take it from the tail.

The signed error is a different distribution again: the average is +0.9 percent, the median −1.5 percent, and the middle 90 percent of points runs from −21.6 to +30.8 percent. So the drag error is nearly unbiased on average and rarely small in any individual case, which is exactly the combination that makes a single headline number misleading.

And the spread itself is predictable. The fitted model later in this section is not a fit to the mean. It is a Gamma model, shape 1.39, so it carries its own spread: given a confidence score, a Reynolds number, an angle and a shape, it returns both the expected error and the 80th and 95th percentile bands around it, and those bands cover 80 and 95 percent of held-out points. That is what makes it usable as a design margin rather than as a scoreboard.

Honest error bars. Resample airfoils instead of points and the 95 percent intervals get two to five times wider. Every headline result still survives:

Statistic	UIUC	Princeton	Both tunnels
Mean drag error (%)	12.3 [11.3, 13.4]	11.1 [10.3, 12.0]	11.7 [11.1, 12.3]
Median drag error (%)	8.1 [7.4, 8.8]	7.5 [6.8, 8.2]	7.8 [7.3, 8.3]
Drag bias (%)	+0.4 [−1.4, +2.1]	+1.3 [0.1, +2.6]	+0.9 [−0.3, +2.0]
Mean lift error (ΔCL)	0.072 [0.064, 0.081]	0.086 [0.080, 0.092]	0.079 [0.073, 0.085]
L/D bias (%)	+15.1 [12.4, 18.0]	+15.0 [12.1, 18.1]	+15.1 [13.0, 17.4]
r(confidence, drag error)	−0.43 [−0.50, −0.35]	−0.27 [−0.33, −0.20]	−0.37 [−0.43, −0.31]
r(confidence, lift error)	0.00 [−0.07, 0.07]	−0.03 [−0.07, 0.02]	0.01 [−0.04, 0.05]
Airfoil-level rank correlation	−0.65 [−0.80, −0.46]	−0.10 [−0.34, 0.14]	−0.41 [−0.59, −0.22]
Drag error, confidence below 0.5 (%)	32 [26, 41]	30 [24, 39]	32 [27, 38]
Drag error, confidence above 0.95 (%)	9.1 [8.3, 10.0]	9.6 [8.9, 10.4]	9.4 [8.9, 9.9]

Brackets are the airfoil-resampled 95 percent intervals. Read them and a few things become clear. The overall drag bias can't be told apart from zero in either tunnel. The L/D over-prediction can. The drag correlation with confidence is firmly negative in both tunnels, and the lift correlation includes zero in both. Only one statistic really fails to repeat, and that's the airfoil-level rank correlation, whose Princeton interval straddles zero.

Why that one doesn't repeat. Split the confidence-versus-drag-error link by level and the answer shows up:

Level	UIUC	Princeton
All points	−0.43	−0.27
Between airfoils (airfoil averages)	−0.69	−0.07
Within airfoils	−0.40	−0.28
Between polars (polar averages)	−0.37	−0.07
Within a polar (along the angle sweep)	−0.45	−0.31

Level	UIUC	Princeton
Share of confidence spread between airfoils	26%	7%

In the UIUC set the confidence score moves both from airfoil to airfoil and along each polar, and both movements predict drag error. In the Princeton set, 93 percent of the score's spread sits inside airfoils rather than between them, because nearly every Princeton model sits above 0.95. There's simply nothing left between airfoils to correlate. What remains lives entirely along the angle sweep, where confidence sags toward the stall while drag error climbs. So the point-level result is real in both sets. The airfoil-level one is a feature of the UIUC sample, which happens to include shapes NeuralFoil is unsure about, and is not a general property of the score itself.

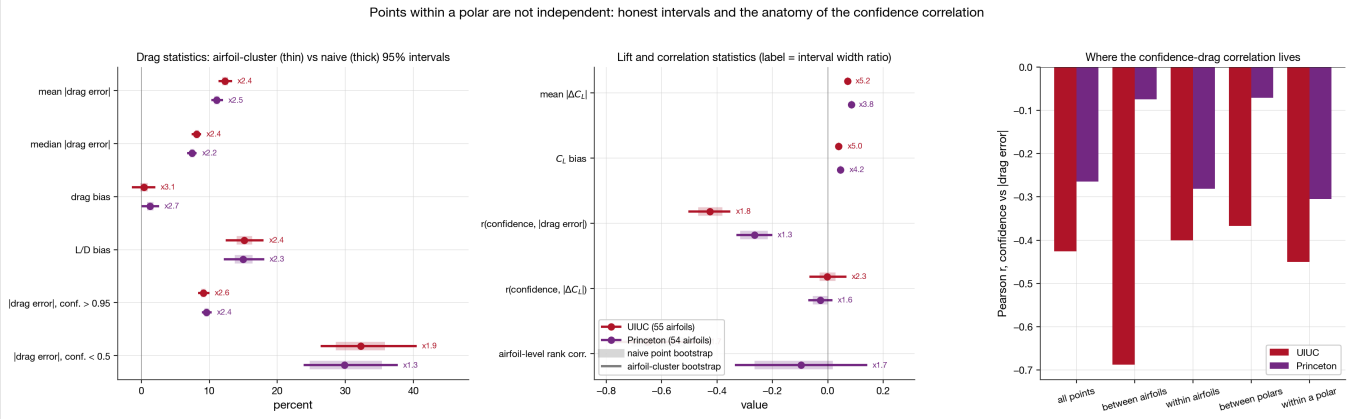


Figure 14. Honest uncertainty. Left and middle: point estimates with airfoil-cluster bootstrap intervals (thin) and naive point-bootstrap intervals (thick), with the width ratio labelled; clustered intervals are two to five times wider, and every headline result survives. Right: the confidence-versus-drag-error correlation split by level; in the Princeton data it lives entirely within polars, which is why the airfoil-level result does not replicate there.

A fitted error model. Take the 8,211 attached-flow points from both tunnels and fit expected drag error against confidence, Reynolds number, angle of attack, camber and thickness. Each decade of (1 minus confidence) multiplies the expected error by 1.87. Each decade of Reynolds number multiplies it by 0.43. Each percent of camber multiplies it by 1.035. Thickness does nothing at all.

Out of sample, with whole airfoils held out, the model predicts the drag error with a mean absolute error of 0.074. A flat constant gets 0.087 and the six-bin confidence table gets 0.079, so the fit beats the constant by 14 percent and the table by 7. Its rank correlation with what actually happened is 0.39. Its 80th and 95th percentile bands cover 80 and 95 percent of held-out points, which is what they're supposed to do. Train it on UIUC alone and test on Princeton and it scores 0.071 against 0.085 for a constant; go the other way and it's 0.077 against 0.089. Its held-out calibration tracks the diagonal from a predicted 6 percent, where the truth was 7, up to a predicted 29 percent, where the truth was 29.

Here's what that means in practice. Take an ordinary 3 percent camber section at 4 degrees. The model expects 7 percent drag error at confidence 0.98 and Re = 200,000, with an 80th percentile of 11 percent. Slow it to Re = 60,000 and that becomes 10 percent. Drop confidence to 0.90 at Re =

100,000 and it's 13 percent. Drop it to 0.16 at Re = 200,000, which is exactly where the unconstrained optimizer of section 3.9 wants to go, and it's 18 percent. The coefficients ship with the code, so anyone can run the model without this paper.

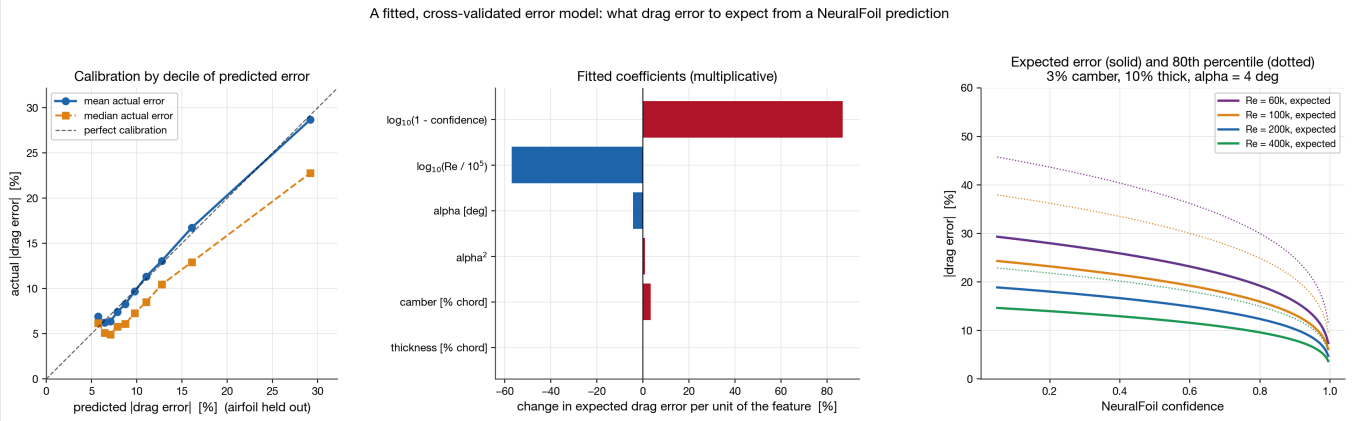


Figure 15. The fitted error model. Left: held-out calibration by decile of predicted drag error, whole airfoils held out. Middle: multiplicative effect of each feature. Right: expected drag error (solid) and its 80th percentile (dotted) against confidence at four Reynolds numbers for a representative section.

3.5 What the drag error does to a rotor, and then to an aircraft

A benchmark number is only worth having if it survives the rest of a design calculation. Bharath Govindarajan put the question directly: an 11 percent drag error carries into rotor torque and rotor power, and there is a real possibility that it builds up over the course of the calculation. In a design loop it becomes battery weight, then payload, then take-off weight. This section runs that chain and measures it.

The rotor has to sit where the benchmark sits. Blade sections must operate in the Reynolds range the two tunnels actually cover, or the measured error does not apply to them. That constraint is harder to satisfy than it sounds, because Reynolds number at a blade station is chord times speed while blade loading is thrust divided by chord. Push one up and the other falls. A rotor whose blade genuinely lives near Re = 400,000 with a sensibly loaded blade belongs on an aircraft of twenty kilograms, not on a hobby quadcopter. The sizing sweep in `data/rotor_sizing_sweep.csv` covers 48 rotors across four radii, four tip speeds and three chords at a fixed disk loading. Twenty-eight of them keep every blade station inside the benchmark's range, and only two of those reach Re = 400,000 at 75 percent span. Both pay for it. The best figure of merit anywhere in the sweep is 0.78, at Re = 273,000; the two rotors that reach 400,000 manage 0.57 and 0.70.

The rotor used here is a four-rotor lift configuration, radius 0.50 m, two blades, chord tapering from 99 to 71 mm, linear washout, turning at 1,910 rpm for a tip speed of 100 m/s and a tip Mach number of 0.29. The section is the E387, which is in the benchmark set and measured in both tunnels, so its error is known rather than assumed. Trimmed to hover it carries 38.5 N per rotor, 15.7 kg all-up, a disk loading of 49 N/m2 and a figure of merit of 0.57.

Quantity	Value
Hover power, one rotor	300.2 W
Induced power	189.4 W, 63.1 percent
Profile power	110.8 W, 36.9 percent
Reynolds number, root cut to tip	117,500 to 475,500
Reynolds number at 75 percent span	405,700
Blade stations inside the benchmark	24 of 24

The verification gate. Govindarajan quoted about 60 percent induced power in hover, with profile taking the rest. This rotor gives 63.1 percent. That agreement is the check that the rotor is representative at all, and it comes before any of the numbers below. It also costs something: rotors that maximise figure of merit run at 85 percent induced, and the sweep shows a 60 percent split belongs to a rotor working harder against its own profile drag. Since the profile share is the only part a drag error can touch, and the whole question is what a drag error does, the split Govindarajan named is the right place to answer it.

Is the solver any good? Checked against measured propellers. Govindarajan's second reply was blunt: validate the tools before putting them inside a sizing loop. Until this paragraph the blade-element solver had been checked against nothing. The check uses the UIUC Propeller Database, from the same group that ran both airfoil archives. Deters designed and 3D-printed a set of propellers so that their geometry could be specified, which means their airfoil section is known exactly: the SDA1075, 12 percent thick, over the whole blade, with coordinates published in the paper. Two blade families, the DA4002 with c/R 0.18 at four pitches and the DA4022 with c/R 0.23 with two, three and four blades, each at 5 and 9 inch diameter, with static thrust and power measured across an RPM sweep: 242 measured points on 14 propellers. Each one is built in the solver from its measured chord and pitch distribution, the section is looked up in NeuralFoil exactly as the rotor above looks up the E387, and it is run at every measured RPM. Nothing is tuned. One honest detail on the section: the SDA1075 fits its 17-number Kulfan description to 0.13 percent of chord, because its trailing edge was thickened to 1.1 percent of chord so the 5 inch blades could be printed. That is coarser than every Princeton model, which all fit to better than 0.07 percent, but it sits inside the UIUC spread, whose median is 0.07 and whose worst is 0.23. NeuralFoil sees that fitted shape.

Two things came out. The first was a defect in the solver. The solver is this paper's own: a hover blade-element momentum model that balances each annulus's blade-element thrust against momentum thrust, with Prandtl tip and root loss and no swirl. Its root-finder searched for the induced velocity below a quarter of the local rotational speed. That limit was a bracket for the root-finder, not physics. A lightly twisted rotor never reaches it (the design rotor above sat at 0.977 of it), but a fixed-pitch propeller root at 36 degrees does. Past the limit an element fell back to zero inflow and sat fully stalled, with no warning. The first repair only moved the limit to 0.95 of the rotational

speed, which still left elements falling back on the XFoil runs. The solver now solves for the inflow angle instead of the induced velocity, and takes its bracket from the element: at zero inflow a lifting section makes more thrust than momentum asks for, and once the inflow angle has carried it 10 degrees below zero incidence it makes negative thrust, so the balance lies between. There is no velocity limit left, and a solve that cannot balance fails loudly.

Solving it properly exposed a second thing. NeuralFoil's SDA1075 polar has a small dip at stall, and where an element sits on it, momentum balances at up to three inflows. The old root-finder chose between them by accident, according to where its bracket happened to end. The solver now scans the bracket and takes the lowest-inflow balance, the one the flow reaches first as the propeller spins up from rest, which is the convention of Ning's CCBlade (Ning, 2014). That choice was made on 45 elements across 27 of the 261 points, all on 9 inch propellers of 6.75 inch pitch and steeper. It moves no point by more than 2.2 percent in thrust, and the design rotor above not at all, to within a microwatt.

The second is the result. Inside a window of Re 40,000 to 98,000 at 75 percent span, over 87 points on the seven 9 inch propellers, the solver lands within 1 percent of measured thrust on average and 4 percent high on power, with mean absolute errors of 9 and 13 percent. Govindarajan asked whether that gap between a small mean and a large mean absolute error is the unsteadiness of the data showing through. It is not. Each measured point sits within 0.6 percent of thrust and 0.8 percent of power, on average, of a smooth curve through its own propeller's sweep, so the balance is not scattering by anything like 9 percent. The error belongs to the propellers: 81 percent of the thrust error's variance and 97 percent of the power error's is the difference between one propeller's offset and the next, and inside a single propeller's sweep the prediction wanders by only 3.6 and 1.6 percent. The two families miss in opposite directions, which is why the bias is small and the mean absolute error is not:

Propeller	Blades	Thrust error	Power error
DA4002 9x2.85	2	−3 percent	−2 percent
DA4002 9x4.76	2	+1 percent	+7 percent
DA4002 9x6.75	2	−8 percent	−14 percent
DA4002 9x8.95	2	−20 percent	−20 percent
DA4022 9x6.75	2	+5 percent	+13 percent
DA4022 9x6.75	3	+10 percent	+16 percent
DA4022 9x6.75	4	+12 percent	+17 percent

Mean signed error, predicted over measured, inside the window. Geometry is not what drives this: rerunning the DA4002 on its drawing instead of the built article moves thrust by 3 percent.

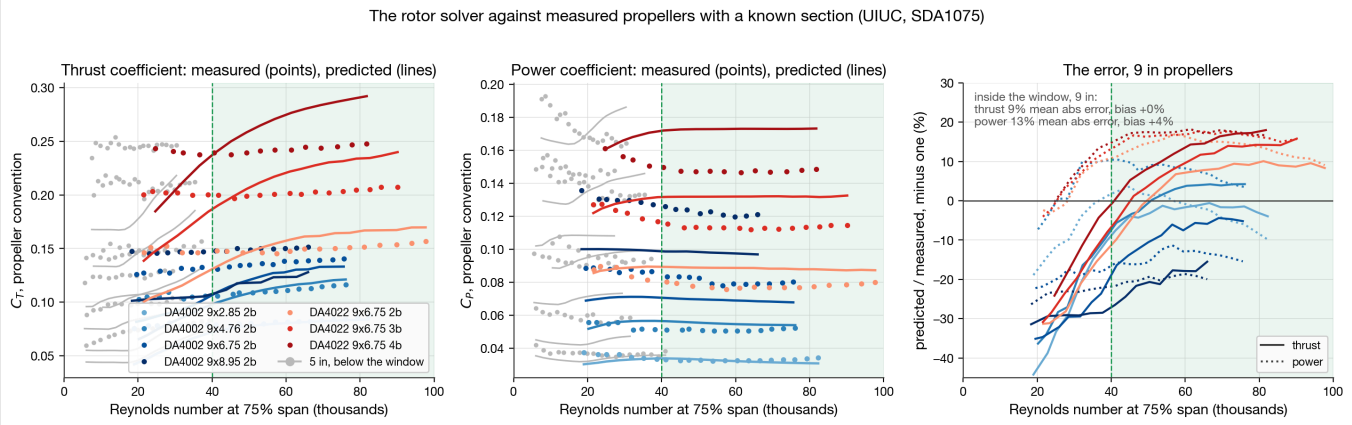


Figure 19. The rotor solver against 14 measured UIUC propellers whose section is known exactly (SDA1075). Left and centre: thrust and power coefficients against Reynolds number at 75 percent span, measured as points and predicted as lines, the 5 inch propellers in grey; the shaded band is the Re 40,000 to 98,000 window inside which the section polar is trustworthy. Right: the error of every 9 inch propeller, thrust solid and power dotted. The two blade families miss in opposite directions, and below Re 40,000 the comparison is dominated by the section polar rather than the solver.

The blade-count series is the cleanest test, because the section and the pitch are identical across it and only the number of blades changes, so the section error is held fixed and what moves is the inflow and tip-loss model. Measured thrust rises by a factor of 1.35 from two blades to three and 1.21 from three to four; the solver says 1.41 and 1.25. It over-rewards solidity by 3 to 4 percent per added blade, and its over-prediction of the DA4022 grows with blade count for the same reason.

So is it the solver, or the section? Run the whole test again with XFoil filling the section table instead of NeuralFoil, on the same Kulfan geometry the network sees, and the answer barely moves. Inside the window the thrust bias goes from +0.5 to +0.4 percent and the power bias from +3.8 to +4.7; the mean absolute errors go from 8.7 and 12.7 percent to 9.0 and 13.7. Point by point across the same 87 points the two section models disagree about thrust by a median of 1.2 percent. So almost none of that 9 and 13 percent scatter is the network's emulation of XFoil, which is consistent with the 2.8 percent emulation error of section 3.3 and is the one candidate this paper can rule out. What the swap cannot separate is the rest: the blade-element model, the measured blade geometry, and XFoil's own physics all survive it, because both section tables carry the same physics.

Two limits carry into everything that follows. These propellers reach Re 98,000 at 75 percent span; the rotor above sits at 118,000 to 475,000. Nothing in the database reaches it, so this validates the solver below its operating point, in the harder regime, not at it. And below Re 40,000 the comparison stops being about the solver: every propeller under-predicts thrust by 14 to 37 percent there, the measured thrust barely moves with RPM, and NeuralFoil's SDA1075 polar stalls at CL 0.78 near 6 degrees at Re 40,000 then recovers to 1.26 at 12 degrees by 60,000. That early stall sits below the 60,000 floor of the benchmark, so this paper has no measurement to say whether it is right. The 5 inch propellers never leave that regime; they are in the figure and out of the numbers.

One more thing to say plainly. At the top of their sweeps six of these seven propellers are 77 to 89 percent induced power; only the steepest-pitched, the DA4002 9x8.95, comes down to 54 percent,

and it is also the worst predicted. The rotor above is 63 percent. So this data tests the thrust and inflow side of the solver hard, and the profile-power side, which is the channel the drag error travels through in the rest of this section, rather less.

Can a static thrust stand check profile power at all? Fitting C_P = κ C_T^1.5/√2 + σ Cd0/8 across the four DA4002 pitches fails: the intercept goes negative, because the pitch swings the 75 percent station from 4 to 16 degrees and neither κ nor Cd0 holds still. Govindarajan suggested two simpler routes, and `profile_power_check.py` tries both on the seven 9 inch propellers inside the window. The first is to take the lowest-thrust point and treat all of its power as profile. The lowest thrust in the database is the DA4002 9x2.85, and it is not close to zero: C_T/σ is 0.14, and its measured figure of merit is 0.57, which says without any rotor model that at least 57 percent of its power is ideal induced power. Read as all profile, its power gives a blade drag coefficient of 0.103, five times what the section can plausibly have. Taking the ideal induced power out first turns the same point into a model-free upper bound of 0.044, or 0.035 with a typical κ of 1.15. His second route is the section's zero-lift drag, 0.021 at this propeller's Reynolds number from NeuralFoil and from XFoil alike, and within 2 percent of the section's minimum drag there, so it serves as the floor. That brackets the effective blade drag between 0.021 and 0.044, and the solver's own power-weighted blade drag, 0.031, sits inside. The check passes, but the bracket spans a factor of two, far wider than the 11 percent drag error it would need to resolve. Carrying the zero-lift drag to the steeper propellers shows where it stops working: the κ it leaves behind climbs from 1.39 at the lowest pitch to 2.50 at the steepest, which no rotor can have, because at 16 degrees incidence NeuralFoil puts the section's drag at about ten times its zero-lift value. One detail matters if anyone repeats this: with σ counted over the blade alone, from 30 percent span, σ Cd0/8 understates the profile power of a uniform Cd0 by about a quarter, so the integral here is done on the measured chord. The answer, then, is that this database can bound profile power but cannot measure it to the precision the question needs. A propeller run near zero thrust would close most of the gap, and nothing in the database was run there. Short of that it takes CFD or a measured blade drag, and this paper records it as a limit.

The blade does not inherit the headline number. Averaged over the blade, the fitted error model of section 3.4 expects 7.4 percent drag error, not 11.7 percent. The blade runs at NeuralFoil confidence 0.94 to 0.98, at moderate angles, and at Reynolds numbers from 118,000 up. Those are the good conditions in this benchmark. The error is not spread evenly along the span: the root station is the worst at 12.3 percent, it falls to a minimum of 6.3 percent near mid-span, and it climbs back to 8.6 percent at the tip, where confidence drops as the section unloads. The inboard climb is the Reynolds trend of section 3.2 showing up as a position along a blade, which is the practical form of that finding: the least trustworthy part of a rotor blade is the part nearest the hub.

Correlated or independent decides the answer. Every blade element is the same airfoil, so an error in that airfoil's drag is not independent noise from element to element. If the model is 10 percent low on this section, it is 10 percent low everywhere along the span. The opposite assumption, independent errors, lets them partly cancel. Both were run, 1,000 Monte Carlo draws each, with the error drawn from the fitted Gamma model at each station's own conditions and the rotor re-trimmed

to the same thrust every time. The signed cases give the sign of each draw from the measured bias at that Reynolds number; the symmetric cases flip a coin instead.

Case	Mean power error (%)	Standard deviation (%)	5th to 95th percentile (%)
Correlated, signed	−1.88	3.34	−7.96 to +3.52
Correlated, symmetric	−0.19	3.58	−6.27 to +5.55
Independent, signed	−2.44	0.97	−4.04 to −1.03
Independent, symmetric	−0.32	1.12	−2.23 to +1.37

The correlated spread is three and a half times the independent one. That factor is the whole disagreement, and it is why the assumption has to be stated rather than left implicit. The correlated case is the honest one here, because the error source is a shared modelling assumption about a single airfoil, not measurement noise sprinkled along a blade.

It checks out against arithmetic. If the error is fully correlated, the fractional power error should be roughly the profile share times the drag error: 0.369 times 7.4 percent, or 2.7 percent. The Monte Carlo gives a mean absolute power error of 2.9 percent in that case. The two agree, which says the rotor model is doing what the algebra says it should and not something else.

So at fixed weight the answer to the first half of Govindarajan's question is that the error is diluted, not amplified. A 7.4 percent section drag error arrives as a 2.9 percent hover power error, because two thirds of hover power is induced and no drag error can touch it.

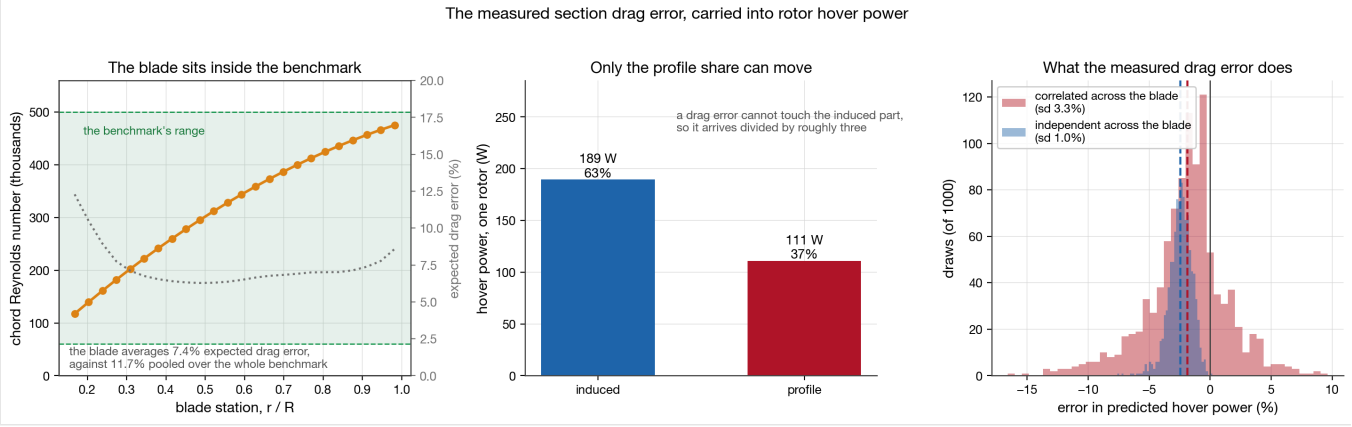


Figure 18. Left: every blade station of the design rotor sits inside the benchmark's Reynolds range, and the fitted error model's expectation (dotted) is worst on the inboard blade. Centre: only the 37 percent of hover power that is profile power can respond to a drag error. Right: 1,000 Monte Carlo draws of the measured error through the rotor, with the error correlated along the blade and independent along it.

And that part generalises. Push the same fully correlated 11.7 percent drag error through every one of the 28 in-envelope rotors in the sizing sweep and the hover power error runs from 1.5 to 5.8

percent, a spread of nearly four. So the propagated number is a property of the rotor at least as much as of the surrogate, and quoting one without the other means little. What does generalise is the mechanism. Across all 28 rotors the power error is 0.93 times the profile share of hover power times the section drag error, with a standard deviation of 0.009 on that coefficient. The profile share is what varies, from 14 to 52 percent, and it is the only thing a designer needs to look up to carry a section drag error into a rotor:

```
hover power error  ≈  0.93 × (profile share of hover power) × (section drag error)
```

The 0.93 rather than 1.00 is the trim: raising drag changes the inflow and the collective slightly, and the rotor recovers a little of the loss. Two cautions before anyone uses it. The coefficient was measured with a deterministic +11.7 percent perturbation, and the response is mildly convex, so an error of the same size in the other direction bites a little harder. And running the whole distribution rather than perturbing by its mean adds roughly a tenth: the rule puts this rotor at 2.5 percent where the Monte Carlo gives 2.7 with symmetric signs and 2.9 with the measured ones. It is a design rule of thumb, not a replacement for the draw. What it is good for is the thing the section number alone cannot do: it says which rotors should worry about a surrogate's drag error and which should not, and the answer is set by one quantity a designer already knows. It is also why an answer to "does the error build up" that does not name the rotor is not an answer.

The weight loop is where it grows. Hover power sets the energy the battery must hold, energy sets battery mass, battery mass sets take-off weight, and weight sets the thrust that sets hover power. Closing that loop for a 2.5 kg payload, a structure fraction of 0.40, 45 minutes of hover, a 180 Wh/kg pack at 85 percent usable and 85 percent drivetrain efficiency gives a take-off mass of 15.71 kg, converging in about fifty iterations. Running it again on the true drag rather than the predicted drag moves the whole aircraft:

Drag error carried in	Take-off mass	Miss, in grams	Hover power error, fixed mass	Hover power error, after closure	Take-off weight error
None, what the model predicts	15.71 kg	0	0	0	0
−19.1 percent, 5th percentile	18.28 kg	+2,573 g	8.0 percent	18.2 percent	14.1 percent
−4.5 percent, median	16.20 kg	+492 g	1.7 percent	4.1 percent	3.0 percent
+10.2 percent, 95th percentile	14.76 kg	−954 g	3.6 percent	9.0 percent	6.5 percent

Read the last two columns together and the compounding Govindarajan warned about is visible and measured. Holding the aircraft's weight fixed, a 19 percent drag error is an 8 percent power error. Let the battery grow to feed that power and the aircraft grow to carry the battery, and the same drag

error becomes an 18 percent power error. The loop multiplies the power error by 2.3 to 2.5 times. That is the build-up, and it is real.

The amplification factor. The ratio of percent error in take-off weight to percent error in section drag is 0.74 at the 5th percentile, 0.68 at the median and 0.63 at the 95th. It is below one at every point of the distribution, and remarkably steady across it. The error compounds inside the loop and still arrives at the aircraft smaller than it started, because payload and structure make up 60 percent of the aircraft and neither of them cares about drag.

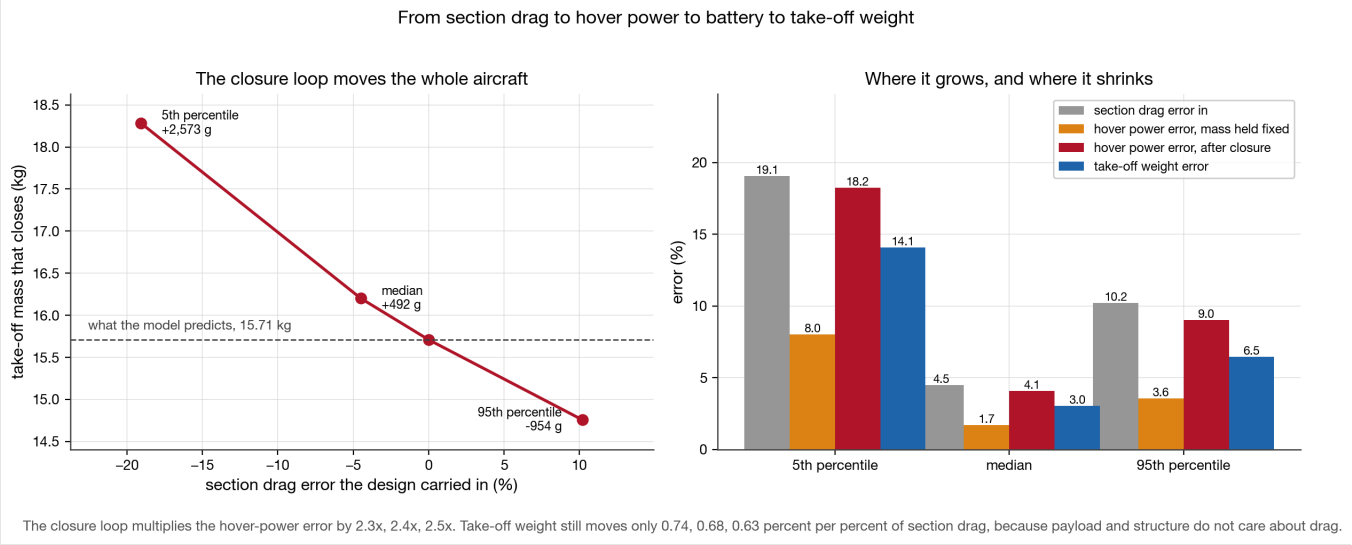


Figure 20. Left: the take-off mass that closes the design loop, against the section drag error carried into it. Right: the same three cases as a chain. The closure loop roughly doubles the hover-power error, and take-off weight still moves about two thirds of a percent per percent of section drag.

Both halves of that sentence matter. A designer who reads the amplification factor alone and concludes that the drag error is harmless has missed that the power error doubled on the way. A designer who reads only the doubling and panics has missed that take-off weight is the quantity being sized. What the chain actually says is: budget roughly two thirds of a percent of take-off weight for every percent of uncertainty in section drag, and expect the power number in the middle of the calculation to be about twice as wrong as the aerodynamics alone would suggest.

In grams, for this aircraft, the 90 percent interval on the drag error is worth about three and a half kilograms of take-off mass, from 14.76 to 18.28 kg on a 15.71 kg design. That is the number a design review would want, and it comes from a wind tunnel rather than from a rule of thumb.

Hover was the mild case. Govindarajan's third point was that the profile fraction rises with airspeed, which makes hover the friendliest place to ask this question. It does, and it is. The same rotor was run in level forward flight, trimmed at each speed against an airframe drag area of 0.05 m2, with the blade elements integrated over azimuth as well as span and the inflow taken from Glauert's formula. Shaft power falls from 291 W in hover to a minimum of 216 W at 12 m/s and climbs again as airframe drag takes over, which is the familiar power bucket and is the first sign the model is behaving.

Speed	μ	Induced (W)	Profile (W)	Propulsive (W)	Shaft (W)	Profile share	Power error from 11.7 percent
0 (hover)	0	173	119	0	291	40.7%	4.47%
6	0.06	113	129	2	243	52.9%	5.87%
12	0.12	59	144	13	216	66.6%	7.49%
14	0.14	49	147	21	217	67.8%	7.64%
20	0.20	23	151	61	235	64.1%	7.19%

The profile share climbs from 41 percent in hover to 68 percent at 14 m/s, and the same fixed drag error costs 7.6 percent of shaft power there against 4.5 percent in hover. Forward flight is **1.7 times worse**, and the rule holds through all of it: the coefficient drifts only from 0.938 to 0.963 across the whole speed range.

There is an unhappy coincidence in those two columns. Shaft power is lowest at 12 m/s and the profile share peaks at 14. The speed this aircraft would cruise at, because it is the cheapest, is within two metres per second of the speed where a surrogate drag error does the most damage. Sizing an endurance mission at the best-endurance speed puts the design exactly where this uncertainty matters most.

Two limits on that. Above $\mu \approx 0.2$ a uniform-inflow model with no cyclic pitch and no flapping stops being defensible, and the sweep stops there rather than producing numbers it cannot stand behind. And the forward-flight solver and the hover solver are not the same model: run at zero speed, the forward-flight one gives 291 W against the hover solver's 300, and a profile share of 40.7 percent against 36.9. Two standard treatments of the same rotor, three percent apart on power and four points apart on the split. Every forward-flight number above is a comparison within one of them, which is why that gap does not propagate into the 1.7.

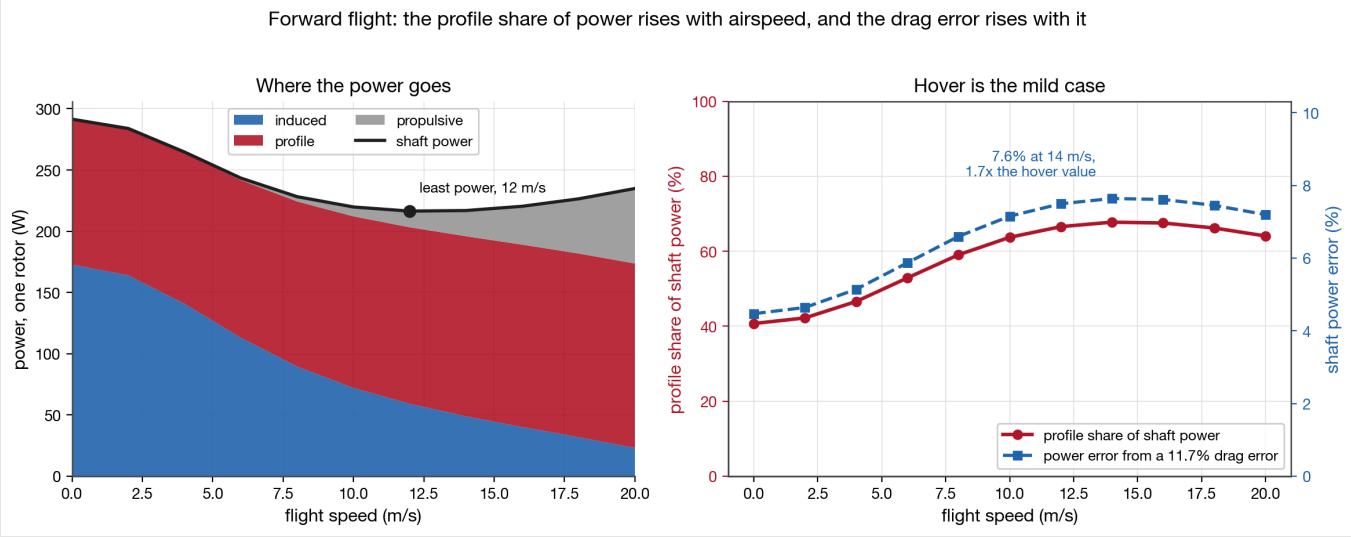


Figure 21. The same rotor in level forward flight. Left: shaft power and its three parts against speed, showing the power bucket with its minimum at 12 m/s. Right: the profile share of shaft power climbs from 41 percent in hover to 68 percent at 14 m/s, and the power error caused by a fixed 11.7 percent section drag error climbs with it, to 1.7 times the hover value.

3.6 Peak performance against a wing that holds up everywhere

Airfoil	Goal	Peak L/D	Worst-case L/D	Max thickness
A (single-point)	best L/D at one condition	232.9 †	7.1	13%
B (robust)	best worst-case L/D	105.1	**38.2**	9%

Airfoil B gives up about 55 percent of A's headline peak and gets back a **5.4 times higher worst case**. A is a spike that collapses the moment conditions move. B is a broad plateau that works nearly everywhere.

† A's peak comes from a condition where NeuralFoil reports almost no confidence, so 233 is an optimistic ceiling and not a real number. The fitted model in section 3.4 puts the expected drag error there near 18 percent, and section 3.1 shows L/D reads 15 percent high even where confidence is good. Airfoil A sits on a razor-thin peak.

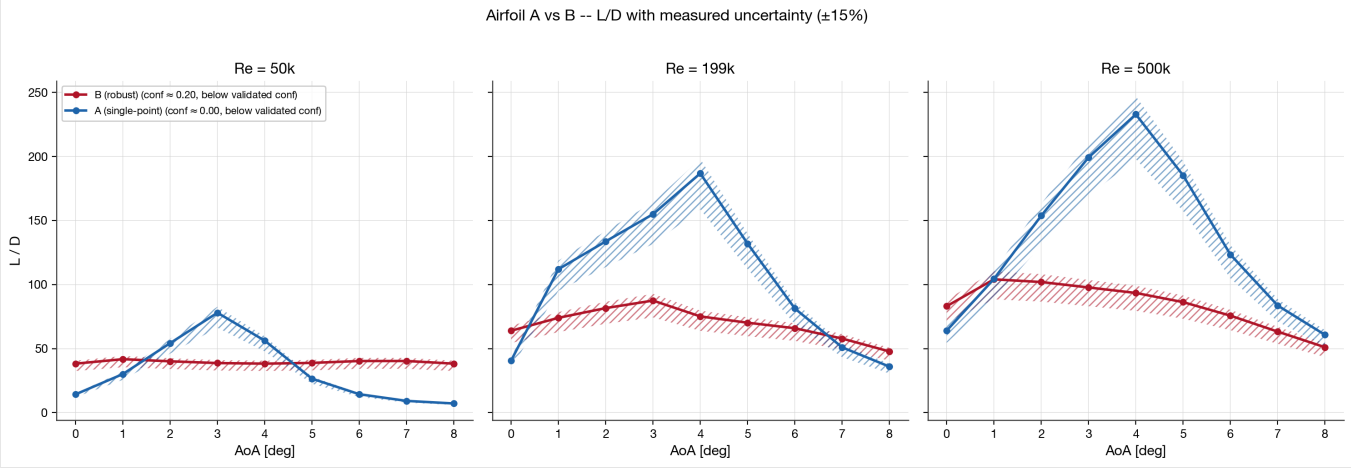


Figure 16. Lift-to-drag ratio versus angle of attack for airfoil A (single-point) and B (robust) at three Reynolds numbers. Hatched bands show the measured 15% surrogate uncertainty; both airfoils sit below NeuralFoil's validated confidence range, so the band is a lower bound.

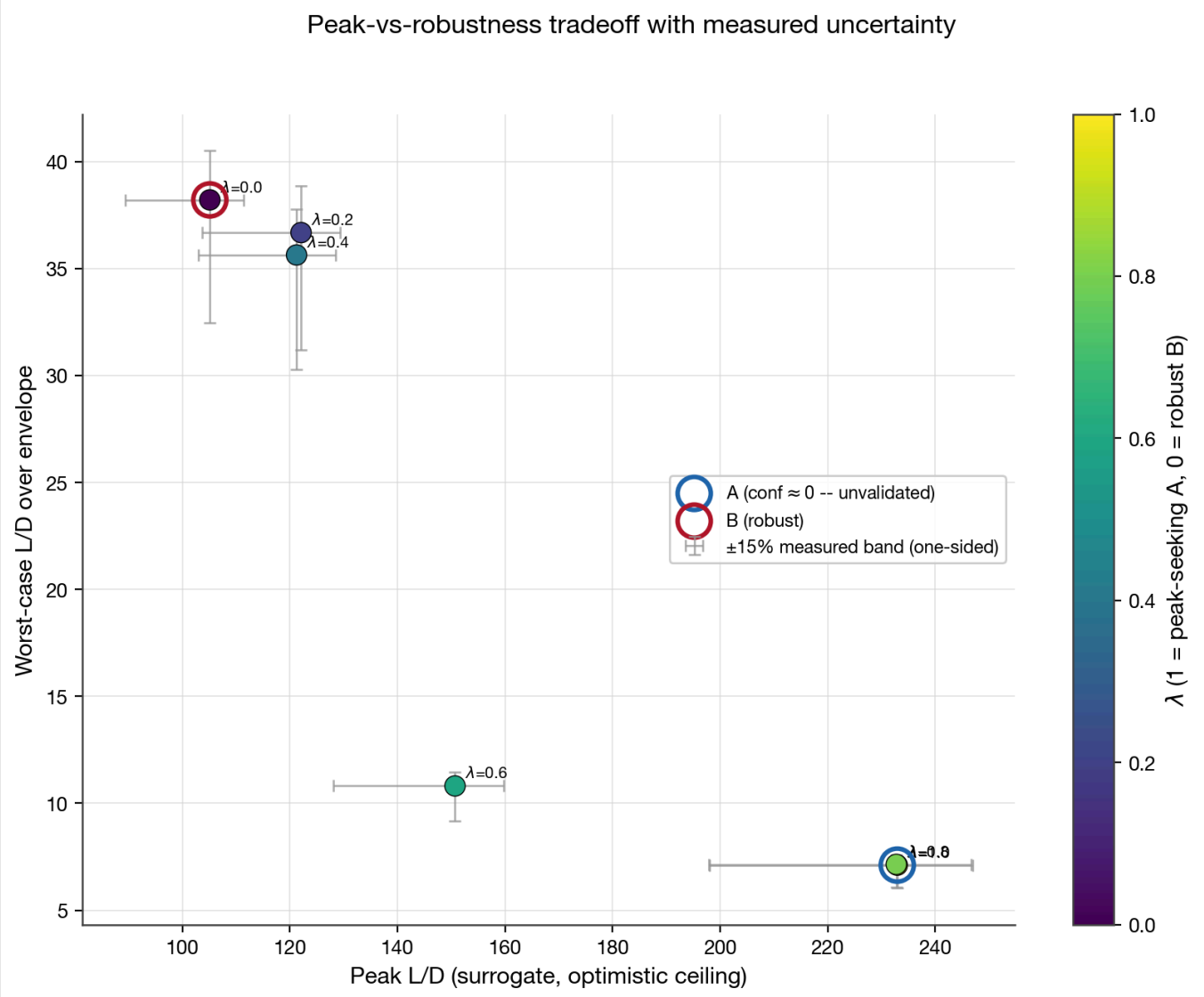


Figure 17. Peak versus worst-case L/D across the design family. Error bars are one-sided (true values trend downward at low Re). Airfoil A's peak near 233 is produced where NeuralFoil reports near-zero confidence.

3.7 Four goals at once

Setting	Worst-case L/D	Stall angle	Max thickness	Noise proxy
Efficiency-first	**38.5**	9.0 degrees	8.0%	15.1×10^{-3}
Balanced	33.9	10.0 degrees	10.4%	11.9×10^{-3}
Community-first	17.8	**11.5 degrees**	**13.7%**	$\mathbf{11.1 \times 10^{-3}}$

Structural depth and slow-speed efficiency pull hard against each other. Making the section thick enough for a deep spar, 13.7 percent, costs about 54 percent of its worst-case L/D. What it buys is a 2.5 degree bigger margin before the stall and a thinner boundary layer at the trailing edge.

3.8 Does the robust wing survive being built badly?

Design	On paper, worst-case L/D	As built, average	As built, worst 5%
B, robust to conditions only	38.0	**19.4**	**9.7**
B, robust to conditions and build error	30.5	19.2	8.7

The perturbations are truncated at ±2.5σ and accepted only when both designs retain at least 0.4 percent chord local thickness. Forty paired valid draws (mean 0.565 percent chord RMS error) required 145 attempts; every accepted shape is reported. Under that declared physical population, the build-robust design gives up substantial on-paper performance and does not improve the held-out mean, fifth percentile or minimum. This is a null/negative result. The earlier twofold claim was produced by scoring self-crossing geometries and is withdrawn.

3.9 Putting the measured confidence inside the optimizer

Section 3.1 showed the confidence score is a reliable warning about drag error. Section 3.3 showed the warning is mostly about XFoil's physics. And drag is what carries the L/D error. So put the warning inside the optimizer.

Alongside the worst-case L/D, the optimizer gets a second reward for landing where NeuralFoil is confident, controlled by one dial called `w_conf`. The grid, 5 by 5, and the three starting shapes match the robust airfoil B from section 3.6. The thickness range is a little wider, 8 to 16 percent instead of 9 to 13, and each dial setting starts from the previous one.

w_conf	Worst-case L/D	Mean confidence	Lowest confidence	Drag error expected there (section 3.1)
0 (trust the model blindly)	38.5	0.16	0.01	about 32%
0.5	37.7	0.96	0.89	about 9%
1	37.7	0.96	0.90	about 9%
2	37.5	0.96	0.92	about 9%
4	36.7	0.97	0.93	about 9%
8	35.2	0.98	0.94	about 9%

With the dial off, the optimizer finds airfoil B again: a predicted worst-case L/D of 38.5 against B's 38.2, at a mean confidence of 0.16 against B's 0.16. At that confidence the benchmark says NeuralFoil's drag is wrong by about a third on average. So 38.5 is a number the model itself won't stand behind. The optimizer went there because, as far as it can tell, a whole family of shapes gives nearly the same worst-case L/D. Some sit inside the region where the model was checked and some sit well outside it. With nothing to break the tie, it picked one outside.

A small nudge on the dial, `w_conf` = 0.5, breaks the tie. The design moves to a mean confidence of 0.96, where the measured drag error is about 9 percent, and the predicted worst-case L/D drops 2 percent, from 38.5 to 37.7. Push harder and you pay steadily more for a little more confidence: at `w_conf` = 8 the predicted L/D is 35.2. The shapes themselves barely change, and the two extremes are shown in the figure.

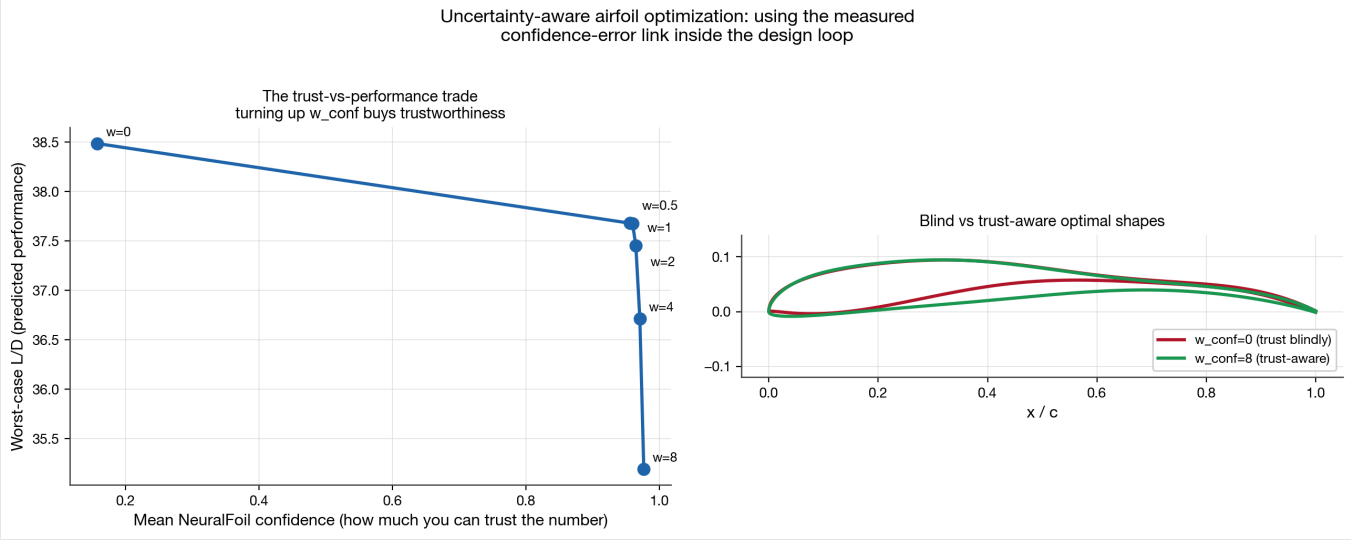


Figure 22. The confidence-aware optimizer. Left: worst-case L/D against mean confidence as the weight w_conf is turned up; the first step, from confidence 0.16 to 0.96, costs 2% of predicted L/D. Right: the two extreme shapes.

Two cautions. That "expected drag error" column is a population statistic from 94 other airfoils, not a measurement of these designs, and none of them have been tested. And as section 1.4 says plainly,

penalizing a surrogate's uncertainty inside an optimizer isn't new; trust-region methods and Kriging-based search rules have done it for decades. What's specific here is that the penalty has a measured scale. The confidence score has been calibrated against 9,130 wind-tunnel points, including the split that shows it flags XFoil's physics error and not just the network's. And the experiment shows something worth knowing on its own: trusting the surrogate blindly lands you, by default, in the region where it's known to be least reliable, and leaving that region is nearly free.

---

4. Discussion

Know when to trust the stand-in, and know what to trust it about. From Re = 100,000 up, NeuralFoil's drag lands within 7 to 10 percent of measurement and its lift within a few hundredths for ordinary shapes. That holds in two separate tunnels. From Re = 200,000 up its drag error is no bigger than the disagreement between those tunnels, so at those speeds it has reached the limit of what the experiments can check. Below Re = 100,000, for very cambered high-lift shapes, and for L/D in general, it turns optimistic.

The split changes what all that means. NeuralFoil is a faithful copy of XFoil, within 3 percent on drag and 0.01 on lift, and it's marginally closer to reality than XFoil itself. The 11 to 12 percent drag error, the Re = 60,000 over-prediction and the high-camber lift error are XFoil's physics, inherited whole. So anyone who wants better slow-speed drag than this won't get it from a bigger network trained on more XFoil. They'll need better physics to train on, or a correction from experiment.

A benchmark number is worth what it does downstream. Eleven point seven percent is a scoreboard entry until someone asks what it costs. Section 3.5 follows it: through a hovering rotor, where two thirds of the power is induced and untouchable, so a 7.4 percent section drag error over the blade arrives as a 2.9 percent power error; then through a design loop, where battery mass feeds back into weight and weight back into power, and that same error becomes 15 percent of hover power at the tail of the distribution but only 12 percent of take-off weight. The chain both dilutes and compounds, in different places, and neither effect is visible from the section number alone. The solver those ratios come out of was checked against 14 measured propellers with a known section before any of this was run. Inside Re 40,000 to 98,000 it lands within a percent of measured thrust on average and 4 percent high on power, with 9 and 13 percent scatter, and nothing in that database reaches the Reynolds numbers this rotor runs at, so the check sits below the operating point rather than on it. And hover is the friendly case. Fly the same rotor forward and the profile share of shaft power climbs from 41 to 68 percent, so the same section error costs 1.7 times as much at cruise as it does hovering. Uncertainty quantification earns its keep exactly here: it says how much of what is unknown early survives to the end. For this aircraft the answer is about two thirds of a percent of take-off weight per percent of section drag, and a three kilogram spread on a 15.7 kg design across the 90 percent interval of the measured error.

The confidence score, meanwhile, is more useful than it was designed to be. It flags the conditions where XFoil's answer is shaky, and those are the same conditions where XFoil's physics is wrong. A designer can watch it as a live warning light for drag and L/D. Better still, run the fitted model from section 3.4, then send only the doubtful, high-value designs off for slower analysis or a real tunnel. If the blade will fly rough or dirty, use the transition inputs, because they work. And when a design finally gets built, measure the built shape and run it again. A few tenths of a percent of chord in build error moves the drag prediction by about 5 percent.

Two lessons here generalize. Points inside one polar are not independent, and treating them as if they were understates the uncertainty by a factor of two to five. And a correlation measured at the point level can come from completely different structures underneath. The UIUC and Princeton sets reach the same conclusion about the confidence score for different reasons, and only the breakdown in section 3.4 shows that the airfoil-level version belonged to one sample.

The design results come with a qualification. Designing against changing operating conditions is still worth doing. Designing against build error, on this small ensemble, is not: once the invalid geometries were thrown out the advantage went with them. The confidence-aware optimizer fares better, and shows that staying inside NeuralFoil's checked region costs almost nothing in predicted performance.

---

5. Limitations

- **Most of this is a validation and framework study, not a discovery.** The shape description, the surrogate and the base optimization methods were all built by other people, and as section 1.4 sets out, robust design and uncertainty-penalized optimization are established fields. What's new is the two-tunnel benchmark, the XFoil split, the experimental noise floor, the build-error separation, the honest statistics and error model, and the measured meaning of the confidence score. Any specific optimized shape still needs higher-fidelity or physical confirmation.
- **Every L/D number here is an estimate**, with a typical measured error of 15 percent and an average over-prediction of 15 percent. Trust the relative conclusions, like robust beating peak, not the raw values.
- **The sample is 94 airfoils, not 9,130 independent points** (section 2.6), and it covers their plain form only. Flapped and gurney-flap versions and UIUC Volume 5 aren't in it. Two UIUC airfoils were dropped because the 17-number description couldn't reproduce them. The Princeton data are older, that tunnel's turbulence level isn't recorded, and the two archives differ by 10.9 percent in drag on matched points. That number includes more than tunnel repeatability and does not bound every result in this paper. Where only design coordinates exist, which is all of the UIUC set, part of every reported error belongs to whoever built the model. The Princeton comparison puts that part at roughly 2 percentage points of drag error. Part belongs to the shape description too. These are pipeline errors, not pure network errors.

- **The XFoil split covers converged points only**, 96.5 percent of them. XFoil's failures cluster where NeuralFoil's confidence is low, so the split is least complete exactly where the errors are biggest.
- **The fitted error model is descriptive.** It explains a modest share of the point-to-point variation, with a rank correlation of 0.39 out of sample. Its value is in the calibrated average and the percentile bands, not in nailing individual points.
- **The "expected drag error" attached to the optimized designs is a population statistic**, not a measurement of those designs.
- **The laminar-run test is a null result and is reported as one.** Drag error does not grow with laminar run length; it falls, and section 3.3 gives no mechanism for that. The test also uses XFoil's own predicted transition location rather than a measured one, so it asks whether XFoil errs more where XFoil itself puts transition early. A solver that places transition in the wrong spot would not reveal that here, and nothing in either archive measures where transition actually happened. Ruling out turbulence intensity as the alternative rests on one expert's estimate of how a low-speed tunnel behaves, not on a measurement in either facility, since neither archive recorded one. Intensity is also only one parameter of a turbulent flow, and Barlow notes that its length scales matter too. Mack's correlation takes intensity alone, so what is ruled out here is the intensity of the freestream and not everything about its turbulence.
- **The 3.7 percent floor is precision, not accuracy.** Repeat runs bound the random part of a tunnel measurement and nothing else. A systematic error cannot appear in them, because a miscalibrated balance repeats its own offset exactly. Neither archive publishes a calibration uncertainty budget, so the total experimental error at any point is larger than 3.7 percent by an amount this paper cannot quantify.
- **Unsteadiness and measurement scatter are not separated here.** Lift and drag at these Reynolds numbers are genuinely unsteady, so a single time-averaged number may not be the right target in the first place. The 3.7 percent repeat-run figure contains both effects mixed together: real run-to-run flow unsteadiness, and ordinary measurement scatter. Neither archive is time-resolved, so this paper cannot tell them apart and does not try. Separating them needs measurements nobody has published at these speeds.
- **The rotor solver has been checked below its operating point, not at it.** Blade-element momentum theory is itself a model with its own error. Against seven measured propellers with a known section, inside Re 40,000 to 98,000, it lands within 1 percent of thrust on average and 4 percent high on power, with 9 and 13 percent scatter and the two blade families missing in opposite directions. No propeller in that database reaches the Re 118,000 to 475,000 this rotor runs at, and six of the seven are 77 to 89 percent induced against this rotor's 63, so the profile-power channel the drag error travels through is the less exercised half. The absolute power numbers in section 3.5 should still not be read as a prediction of what this rotor would draw on a test stand; the ratios between perturbed and unperturbed runs are what carry weight. The rotor is also idealised: no hub, no blade-root fittings, no interference between rotors, and a figure of merit of 0.57 that BEMT most likely flatters.
- **Forward flight is a coarser model than hover.** It uses uniform inflow, no cyclic pitch, no blade flapping, and a bluff-body treatment of the reverse-flow disk, and it stops at $\mu = 0.2$ for

that reason. Run at zero speed it disagrees with the hover solver by 3 percent on power and 4 points on the induced-profile split, so forward-flight results are only quoted as ratios within that solver. Climb, descent, manoeuvre and gusts are not covered at all.

- **The error model was fitted to a wind tunnel, not to a blade.** The Gamma model draws on two-dimensional, non-rotating residuals. Rotational effects on the boundary layer, radial flow and centrifugal pumping among them, are not in it, and they are known to matter most on the inboard blade, which is also where this study says the expected error is largest.
- **The weight closure is one aircraft with one set of assumptions.** Payload, structure fraction, endurance and pack energy were fixed at plausible values, and the amplification factor depends on them. A design with a larger payload fraction would dilute the error further; one that is nearly all battery would not.
- **The measurements are two-dimensional and steady.** No three-dimensional wing, no unsteady gusts, and no rotating blade. The rotor study in section 3.5 propagates the measured two-dimensional error through a rotor model; it does not measure a rotor.
- **None of the optimized airfoils have been physically tested.**

---

6. Conclusion

This paper tests NeuralFoil against 9,130 wind-tunnel measurements on 94 airfoils in two separate tunnels below Re = 500,000, runs XFoil at every one of those conditions, and puts an honest error bar on every number.

The stand-in is accurate on drag from Re = 100,000 up, optimistic below that and for high-lift shapes, and the same in both labs. From Re = 200,000 up its drag error is no bigger than the disagreement between those labs. Almost all of its error is XFoil's, inherited faithfully; the network itself adds about 3 percent on drag and 0.01 on lift, and lands slightly closer to reality than the solver it copies. Its confidence score is a usable warning about drag error, and about XFoil's physics error, but not about lift, and the airfoil-level version of that result belongs to one sample. A fitted model turns confidence, Reynolds number, angle and camber into an expected drag error that carries over between tunnels. The transition inputs reproduce trip-strip measurements. Running the model on a wing's measured shape instead of its drawing removes about a sixth of the drag error. And fed back into the optimizer with its measured meaning, the confidence score moves designs out of the least reliable region for about 2 percent of predicted performance.

And the number does not stay a scoreboard entry. Carried through a hovering rotor whose blade sits entirely inside the benchmark's Reynolds range, and then through a design loop that closes battery mass against take-off weight, a percent of section drag error arrives as 0.63 percent of take-off weight, after the loop has roughly doubled the hover-power error along the way. The error is diluted by the two thirds of hover power that is induced, compounded by the battery feeding back into the weight, and diluted again by a payload and a structure that do not care about drag. The rotor solver itself was checked against measured propellers with a known section, and lands within a percent of

thrust and 4 percent of power on average, below the Reynolds numbers this rotor runs at rather than at them. The propagated number belongs to the rotor as much as to the surrogate, and across 28 rotors spanning a factor of four in propagated error the mechanism is the same one: power error is 0.93 times the profile share of shaft power times the section drag error. That holds in forward flight too, where the profile share climbs from 41 percent in hover to 68 percent at 14 m/s and the same drag error costs 1.7 times as much. The speed this aircraft would cruise at for endurance is within two metres per second of the speed where the surrogate's error does the most damage. On this 15.7 kg aircraft the 90 percent interval of the measured drag error is worth about three and a half kilograms of take-off mass.

Four things would push this further. A third tunnel, with documented turbulence and flapped configurations. A measured rotor or propeller in the Re 100,000 to 500,000 range with a published section, so the solver can be checked where this rotor actually runs. No public database has one. A training set for the surrogate that includes experimental or higher-fidelity slow-speed data, since the split here shows that more XFoil will not help. And a real wind-tunnel test of 3D-printed robust and peak-tuned airfoils, with their built shapes measured, which would turn this validation study into an original experimental result.

Author Contributions

The author designed and directed this study: choosing the question, setting its goals and scope, and reviewing and interpreting every result. AI tools were used for parts of the code implementation, figure generation and drafting, under the author's direction.

Acknowledgments

All experimental data used in this work are from public sources and are cited below.

References

1. Sharpe, P. D., and Hansman, R. J. (2025). *NeuralFoil: An airfoil aerodynamics analysis tool using physics-informed machine learning*. arXiv:2503.16323.
2. Selig, M. S., and McGranahan, B. D. (2004). *Wind Tunnel Aerodynamic Tests of Six Airfoils for Use on Small Wind Turbines*. NREL/SR-500-34515.
3. McGhee, R. J., Walker, B. S., and Millard, B. F. (1988). *Experimental Results for the Eppler 387 Airfoil at Low Reynolds Numbers in the Langley Low-Turbulence Pressure Tunnel*. NASA TM-4062.

4. Kulfan, B. M. (2008). Universal parametric geometry representation method. *Journal of Aircraft*, 45(1), 142-158.
5. Drela, M. (1989). XFOIL: An analysis and design system for low Reynolds number airfoils. In *Low Reynolds Number Aerodynamics* (pp. 1-12). Springer.
6. Sharpe, P. D. (2021). *AeroSandbox: A differentiable framework for aircraft design optimization*. MIT master's thesis and software.
7. Selig, M. S., Guglielmo, J. J., Broeren, A. P., and Giguère, P. (1995). *Summary of Low-Speed Airfoil Data, Volume 1*. SoarTech Publications.
8. Selig, M. S., Lyon, C. A., Giguère, P., Ninham, C. P., and Guglielmo, J. J. (1996). *Summary of Low-Speed Airfoil Data, Volume 2*. SoarTech Publications.
9. Lyon, C. A., Broeren, A. P., Giguère, P., Gopalarathnam, A., and Selig, M. S. (1998). *Summary of Low-Speed Airfoil Data, Volume 3*. SoarTech Publications. Tabulated data for Volumes 1 to 4 from the UIUC Low-Speed Airfoil Tests archive, https://m-selig.ae.illinois.edu/pd.html
10. Selig, M. S., Donovan, J. F., and Fraser, D. B. (1989). *Airfoils at Low Speeds*. Soartech 8, SoarTech Publications, Virginia Beach. Tabulated data from https://m-selig.ae.illinois.edu/uiuc_lsat.html
11. Selig, M. S. (2003). *Low Reynolds Number Airfoil Design Lecture Notes*. VKI Lecture Series, Low Reynolds Number Aerodynamics on Aircraft Including Applications in Emerging UAV Technology, von Karman Institute for Fluid Dynamics, RTO/AVT-VKI-104.
12. Selig, M. S., Deters, R. W., and Williamson, G. A. (2011). Wind tunnel testing airfoils at low Reynolds numbers. *49th AIAA Aerospace Sciences Meeting*, AIAA 2011-875.
13. Demie, A. B., Ancha, V. R., and Kahsay, M. B. (2026). Multi-model assessment and experimental validation of a custom high-camber airfoil for wind-lens technology application. *Wind*, 6(2), 28.
14. Li, J., Du, X., and Martins, J. R. R. A. (2022). Machine learning in aerodynamic shape optimization. *Progress in Aerospace Sciences*, 134, 100849.
15. Drela, M. (1998). Pros and cons of airfoil optimization. In *Frontiers of Computational Fluid Dynamics 1998* (pp. 363-381). World Scientific.
16. Li, W., Huyse, L., and Padula, S. (2002). Robust airfoil optimization to achieve drag reduction over a range of Mach numbers. *Structural and Multidisciplinary Optimization*, 24(1), 38-50.
17. Alexandrov, N. M., Dennis, J. E., Lewis, R. M., and Torczon, V. (1998). A trust-region framework for managing the use of approximation models in optimization. *Structural Optimization*, 15(1), 16-23.
18. Jones, D. R., Schonlau, M., and Welch, W. J. (1998). Efficient global optimization of expensive black-box functions. *Journal of Global Optimization*, 13(4), 455-492.
19. Queipo, N. V., Haftka, R. T., Shyy, W., Goel, T., Vaidyanathan, R., and Tucker, P. K. (2005). Surrogate-based analysis and optimization. *Progress in Aerospace Sciences*, 41(1), 1-28.
20. Forrester, A. I. J., Sobester, A., and Keane, A. J. (2008). *Engineering Design via Surrogate Modelling: A Practical Guide*. Wiley.
21. Yang, Y., Li, R., Zhang, Y., and Chen, H. (2026). *Uncertainty-aware data-based method for fast and reliable shape optimization*. arXiv:2601.21956.

22. Ning, S. A. (2014). A simple solution method for the blade element momentum equations with guaranteed convergence. *Wind Energy*, 17(9), 1327-1345.

Reproducibility: all code, configuration files, extracted data as CSV, the fitted error model as JSON, and every figure are in the project repository. Software versions are pinned and random seeds are fixed. Every study writes its data to disk before making any plots, and a test suite pins every number quoted in this paper to those data files.