Comprehensive Validation Framework Documentation

Executive Summary

This document provides an in-depth analysis of the validation framework developed to verify the correctness and reliability of the wing optimization system. The validation approach is based on a fundamental principle from aerodynamic theory: **for a planar wing generating a given lift, the minimum induced drag occurs when the lift distribution is elliptical**. This theoretical foundation provides an objective benchmark against which the optimizer's behavior can be validated.

Table of Contents

1. Validation Methodology Overview
2. Theoretical Foundation
3. Code Architecture Analysis
4. Design Variable Configuration
5. Objective Function Formulation
6. Constraint Implementation
7. VSPAERO Analysis Configuration
8. Geometric Control Strategy
9. Optimization Setup
10. Why This Validation is Correct
11. Expected Results
12. Validation Success Criteria
13. Limitations and Considerations

---

1. Validation Methodology Overview

1.1 Core Validation Principle

The validation framework tests whether the optimization system can discover known theoretical optima without being explicitly told what that optimum should be. This is accomplished by:

1. **Setting a specific optimization objective:** Minimize induced drag (Cd_i)
2. **Imposing a fixed constraint:** Maintain constant lift coefficient (Cl ≈ 0.8571)
3. **Allowing geometric freedom:** Let the optimizer modify wing planform shape
4. **Checking convergence:** Verify if the resulting geometry approximates an elliptical planform

1.2 Why This Approach is Valid

This validation methodology is scientifically rigorous because:

Theoretical Certainty: Ludwig Prandtl's lifting-line theory mathematically proves that elliptical lift distribution minimizes induced drag for a given lift. This is not an empirical observation but a proven mathematical result.

Independent Discovery: The optimizer is not programmed with knowledge of elliptical wings. If it independently discovers this shape, it demonstrates that the optimization framework is functioning correctly.

Quantifiable Verification: The convergence toward an elliptical planform can be objectively measured and compared against the theoretical optimum.

Sensitivity Test: This validation tests the optimizer's ability to navigate complex trade-offs between multiple geometric parameters to find a global optimum.

---

2. Theoretical Foundation

2.1 Prandtl's Lifting-Line Theory

The theoretical basis for this validation comes from Ludwig Prandtl's classical lifting-line theory, which states:

For a planar wing with span b and total lift L, the induced drag is minimized when the spanwise lift distribution follows an elliptical form:

```
l(y) = l_0 × sqrt(1 - (2y/b)²)
```

Where:

- $l(y)$ = lift per unit span at spanwise location y
- l_0 = maximum lift per unit span (at wing center)
- y = spanwise coordinate (-b/2 to +b/2)
- b = total wingspan

2.2 Induced Drag Formula

The induced drag coefficient for an elliptical distribution is:

```
Cd_i = Cl² / (π × AR × e)
```

Where:

- Cd_i = Induced drag coefficient
- Cl = Lift coefficient
- AR = Aspect ratio (b^2/S)
- e = Oswald efficiency factor

For a perfect elliptical distribution: e = 1.0

2.3 Non-Elliptical Distributions

Any deviation from elliptical distribution results in $e < 1.0$, which increases induced drag. Common efficiency factors:

- Rectangular wing: e ≈ 0.7-0.8
- Tapered wing: e ≈ 0.8-0.95
- Elliptical wing: e = 1.0 (theoretical maximum)

2.4 Physical Interpretation

Why is elliptical optimal?

Elliptical lift distribution produces a uniform downwash across the entire span. This means:

- Every section of the wing experiences the same induced angle of attack
- Energy is not wasted creating spanwise variations in downwash
- The wing operates at maximum aerodynamic efficiency

Chord Distribution Connection:

To achieve elliptical lift distribution with constant section lift coefficient (Cl_section), the chord must vary elliptically:

```
c(y) = c_0 × sqrt(1 - (2y/b)²)
```

This is the geometric shape the optimizer should discover.

---

3. Code Architecture Analysis

3.1 Overall Structure

The validation code follows the same architectural pattern as the main optimizer but with critical modifications for validation purposes:

```
┌─────────────────────────────────────────┐
│   Optimization Loop (SMPSO)             │
│   - 70 particles, 100 evaluations       │
├─────────────────────────────────────────┤
│   Objective Function                    │
│   - Minimize Cd_i                       │
│   - Enforce Cl ≈ 0.8571                 │
├─────────────────────────────────────────┤
│   VSPAERO Analysis                      │
│   - Custom planform generation          │
│   - AOA = 10°, Re = 600,000           │
├─────────────────────────────────────────┤
│   Planform Geometry Control             │
│   - Root/tip chords, sweep, LE control  │
└─────────────────────────────────────────┘
```

3.2 Key Differences from Main Optimizer

Aspect	Main Optimizer	Validation Code
Objective	Maximize Cl^1.5/Cd	Minimize Cd_i only
Lift Constraint	0.2 ≤ Cl ≤ 0.25	Cl ≈ 0.8571 (tight)
Design Variables	6 (span, area, taper, sweep, twist, incidence)	5 (root_c, tip_c, sweep, angle, strength)
Planform Control	Standard drivers	Custom leading/trailing edge control
Target Result	Maximum endurance	Elliptical planform
Validation Goal	Performance optimization	Framework verification

3.3 Import and Setup

```
from platypus import Problem, Real, SMPSO, PM
import openvsp as vsp
import os, sys, contextlib
from tqdm import tqdm
import csv, math
```

Purpose of Each Import:

- **platypus:** Provides SMPSO optimization algorithm
- **openvsp:** Parametric wing geometry and VSPAERO interface
- **contextlib:** Output suppression for clean execution
- **tqdm:** Progress tracking for long-running optimization
- **csv:** Results logging and data export

---

4. Design Variable Configuration

4.1 Variable Definitions

```
root_c_min, root_c_max = 0.05, 2.0     # Root chord (m)
tip_c_min, tip_c_max = 0.05, 2.0       # Tip chord (m)
SW_min, SW_max = 0.0, 10               # Sweep angle (deg)
angle_min, angle_max = 0, 60           # LE/TE angle (deg)
strength_min, strength_max = 0.5, 1.5  # LE/TE strength
```

4.2 Five Design Variables Explained

Variable 1: Root Chord (root_c)

- Range: 0.05 to 2.0 meters
- Physical meaning: Chord length at wing centerline

- Impact: Controls overall wing area and center section loading
- Wide range allows optimizer full freedom

Variable 2: Tip Chord (tip_c)

- Range: 0.05 to 2.0 meters
- Physical meaning: Chord length at wing tip
- Impact: Determines taper ratio and tip loading
- Critical for planform shape evolution

Variable 3: Sweep Angle (sweep)

- Range: 0 to 10 degrees
- Physical meaning: Leading edge sweep at quarter-chord
- Impact: Affects effective span and induced drag
- Small range prevents excessive sweep

Variable 4: Leading/Trailing Edge Angle (angle)

- Range: 0 to 60 degrees
- Physical meaning: Controls curvature of leading/trailing edges
- Impact: Allows transition from straight to curved planform
- Key parameter for creating elliptical shape

Variable 5: Leading/Trailing Edge Strength (strength)

- Range: 0.5 to 1.5
- Physical meaning: Magnitude of LE/TE curvature
- Impact: Fine-tunes the elliptical approximation
- Works with angle to achieve optimal shape

4.3 Fixed Parameters

Several parameters are fixed to isolate the effect of planform shape:

```
span = 5  # Fixed at 5 meters
aoa = 10.0  # Fixed angle of attack
reynolds = 600000  # Fixed Reynolds number
```

Rationale for Fixed Parameters:

Fixed Span (5m): Eliminates aspect ratio as a variable. Since $Cd_i = Cl^2/(\pi \times AR \times e)$, fixing AR isolates the effect of efficiency factor (e), which is purely determined by lift distribution shape.

Fixed AOA (10°): Ensures all geometries are evaluated at the same flight condition, making Cd_i comparisons meaningful.

Fixed Reynolds: Removes viscous scaling effects, ensuring differences in Cd_i are due to induced drag variations, not parasite drag changes.

4.4 Variable Bounds Reasoning

Why Allow Root > Tip and Tip > Root?

The bounds allow both scenarios (root_c and tip_c both range 0.05-2.0) because:

1. The objective function penalizes tip > root configurations
2. This tests the optimizer's constraint handling
3. In theory, any configuration should be explorable if physically valid

Penalty Implementation:

```
if tip_c > root_c:
    drag = 100000  # Severe penalty
```

This ensures the optimizer learns to avoid physically unrealistic geometries.

---

5. Objective Function Formulation

5.1 Function Structure

```
def weighted_sum_wing_analysis(vars):
    root_c, tip_c, sweep, angle, strength = vars

    if tip_c > root_c:
        # Severe penalty for unrealistic geometry
        drag = 100000
        c1 = Cl_ref
        c2 = Cl_ref
    else:
        # Run actual analysis
        Cl, Cd_i, Cm, l_d = run_vspaero_analysis(...)

        # Calculate penalty for Cl deviation
        Cl_ref = 0.8571
        penalty = 0
        if abs(Cl - Cl_ref) > 0.01:
            penalty = ((abs(Cl - Cl_ref) - 0.01) / 0.01)**2

        # Combined objective
        drag = Cd_i + 0.1 * penalty

        # Constraints
        c1 = Cl - (Cl_ref + 0.01)  # Cl ≤ 0.8671
        c2 = -Cl + (Cl_ref - 0.01)  # Cl ≥ 0.8471

    return [drag], [c1, c2]
```

5.2 Objective Components Explained

Primary Objective: Minimize Cd_i

The core objective is to minimize induced drag coefficient. This is the only aerodynamic quantity being optimized, which makes the validation test focused and unambiguous.

Lift Constraint: Cl ≈ 0.8571

This specific value was chosen because:

1. It's high enough to be representative of actual flight conditions
2. It's achievable at 10° AOA with reasonable geometries
3. It provides sufficient lift for meaningful induced drag comparisons

Tolerance Band: ±0.01

The tight tolerance (Cl = 0.8571 ± 0.01) ensures:

- All geometries generate approximately the same lift
- Differences in Cd_i are due to efficiency variations, not lift variations
- Fair comparison between candidate designs

5.3 Penalty Function Analysis

Quadratic Penalty Structure:

```
penalty = ((abs(Cl - Cl_ref) - 0.01) / 0.01)**2
```

Mathematical Breakdown:

For Cl outside the tolerance band [0.8471, 0.8671]:

- Deviation = abs(Cl - 0.8571) - 0.01
- Normalized deviation = Deviation / 0.01
- Penalty = (Normalized deviation)^2

Example Calculations:

Cl	Deviation	Normalized	Penalty
0.8571	0.0000	0.00	0.0000
0.8671	0.0000	0.00	0.0000
0.8771	0.0100	1.00	1.0000
0.8971	0.0300	3.00	9.0000
0.7571	0.0900	9.00	81.0000

Penalty Weight: 0.1

The penalty is weighted at 10% of induced drag:

```
drag = Cd_i + 0.1 * penalty
```

Rationale:

- Heavy enough to guide optimizer toward feasible region
- Light enough not to dominate the actual Cd_i differences
- Allows small violations while strongly penalizing large ones

Typical Cd_i values: 0.01-0.05 Penalty for Cl = 0.88 (0.02 deviation): 0.1 × 4 = 0.4

This penalty (0.4) is much larger than typical Cd_i differences (0.001-0.005), ensuring feasibility is prioritized.

5.4 Constraint Formulation

Constraint 1: Upper Lift Bound

```
c1 = Cl - (Cl_ref + 0.01)  # Must be ≤ 0
```

Equivalent to: Cl ≤ 0.8671

Constraint 2: Lower Lift Bound

```
c2 = -Cl + (Cl_ref - 0.01)  # Must be ≤ 0
```

Equivalent to: Cl ≥ 0.8471

Dual Enforcement Mechanism:

Both penalty function AND hard constraints enforce the lift requirement:

1. **Penalty:** Soft constraint, guides search direction
2. **Constraints:** Hard constraint, marks solutions as infeasible

This dual approach ensures:

- Smooth objective function landscape (penalty)
- Clear feasibility boundaries (constraints)
- Robust convergence to target Cl

---

6. Constraint Implementation

6.1 Two-Sided Lift Constraint

The validation implements a tight band around the target lift coefficient:

```
0.8471 ≤ Cl ≤ 0.8671
```

Width Analysis:

- Total band width: 0.02 (2% of target value)
- Tolerance: ±0.01 from center
- This is approximately ±1.2% variation

Why This Specific Range?

Aerodynamic Consistency: At 2% variation in Cl, the induced drag formula $Cd_i = Cl^2/(\pi \times AR \times e)$ varies by approximately 4% due to lift alone. This is small enough that efficiency differences (e) dominate the comparison.

Computational Precision: VSPAERO panel method typically has 1-2% accuracy in lift prediction. The ±0.01 tolerance accommodates this numerical precision while still being constraining.

Optimization Feasibility: Too tight (e.g., ±0.001) would make the problem over-constrained and prevent convergence. Too loose (e.g., ±0.05) would allow lift variations to confound the induced drag comparison.

6.2 Geometric Constraint (Implicit)

```
if tip_c > root_c:
    drag = 100000  # Severe penalty
```

Physical Justification:

For subsonic, non-swept wings, having tip chord greater than root chord is aerodynamically unusual because:

1. Structural efficiency: Bending moments are highest at root
2. Stall characteristics: Tip stall before root stall is dangerous
3. Manufacturing: Conventional construction expects larger root

Implementation Choice:

Rather than a hard constraint (marking as infeasible), a severe penalty is used. This allows the optimizer to explore this region briefly but quickly learn to avoid it.

6.3 Constraint Handling in SMPSO

The Platypus SMPSO algorithm handles constraints through:

1. **Feasibility Marking:** Solutions violating constraints are marked as infeasible
2. **Fitness Penalization:** Infeasible solutions receive poor fitness
3. **Selection Pressure:** Only feasible solutions considered for "best" solution
4. **Archive Management:** Leader archive maintains only feasible solutions

Effect on Optimization:

Early iterations: Many particles may be infeasible, exploring the full design space.

Middle iterations: Swarm gravitates toward feasible region through social learning.

Late iterations: Most/all particles remain feasible, refining the optimal design within constraints.

---

7. VSPAERO Analysis Configuration

7.1 Fixed Simulation Parameters

```
span = 5  # meters
aoa = 10.0  # degrees
mach = 0.15  # Mach number
reynolds = 600000  # Reynolds number based on chord
```

Air Properties (Standard Sea Level):

```
rho = 1.2256  # kg/m³ (standard atmosphere)
```

7.2 Reference Quantities

```
vsp.SetDoubleAnalysisInput(analysis_name, "Sref", [10])  # m²
vsp.SetDoubleAnalysisInput(analysis_name, "bref", [10])  # m
vsp.SetDoubleAnalysisInput(analysis_name, "cref", [1])   # m
```

Critical Insight: Fixed Reference Values

The reference values are fixed at Sref=10 m^2, bref=10 m, cref=1 m regardless of actual wing geometry. This is intentional and correct because:

Coefficient Definition:

```
Cl = L / (0.5 × ρ × V² × Sref)
Cd_i = D_i / (0.5 × ρ × V² × Sref)
```

Why Fixed References Work:

If we used actual wing area as Sref, then:

- Larger wings would have smaller Cl for same lift force
- Smaller wings would have larger Cl for same lift force
- Comparison would be meaningless

By fixing Sref = 10 m^2, we ensure:

- All wings are compared on equal footing
- Cl and Cd_i are consistent across different geometries
- The constraint Cl ≈ 0.8571 has consistent physical meaning

Verification:

```
Actual span = 5 m (fixed)
bref = 10 m (reference)
Sref = 10 m² (reference)

Implied reference AR = bref² / Sref = 100 / 10 = 10
```

This is a reasonable aspect ratio, ensuring the reference frame is physically sensible.

7.3 Mesh Quality Settings

```
vsp.SetParmVal(wing_id, "SectTess_U", "XSec_1", 10)  # Spanwise panels
vsp.SetParmVal(wing_id, "Tess_W", "Shape", 20)       # Chordwise panels
vsp.SetParmVal(wing_id, "LECluster", "WingGeom", 0.2)  # LE clustering
vsp.SetParmVal(wing_id, "TECluster", "WingGeom", 0.2)  # TE clustering
```

Panel Count:

- Spanwise: 10 panels per section
- Chordwise: 20 panels
- Total: Approximately 200 panels per half-wing

Clustering:

- 20% of panels clustered at leading edge
- 20% of panels clustered at trailing edge
- Captures high-gradient flow regions

Trade-off Analysis:

Aspect	Low Mesh	Current Mesh	High Mesh
Panels	50-100	200	500-1000
Accuracy	±5%	±2%	±1%
Time/run	3-5 sec	5-10 sec	15-30 sec
Total time	1 hour	2-3 hours	6-10 hours

The current mesh (200 panels) balances accuracy and computational cost for validation purposes.

7.4 Airfoil Selection

```
vsp.ReadFileAirfoil(root_id, "NACA0012.dat")
vsp.ReadFileAirfoil(tip_id, "NACA0012.dat")
```

NACA 0012 Characteristics:

- **Symmetric airfoil:** No camber (mean line is straight)
- **12% thick:** Maximum thickness = 12% of chord
- **Zero Cl at α=0:** Requires angle of attack for lift

Why NACA 0012 for Validation?

1. **Symmetry:** Eliminates camber effects on lift distribution
2. **Well-documented:** Extensive wind tunnel data available for verification
3. **Numerical stability:** Smooth contour prevents convergence issues
4. **Constant section Cl:** At fixed α, all sections have same Cl_section

The last point is crucial: with constant section lift coefficient, achieving elliptical total lift distribution requires elliptical chord distribution, which is exactly what we want to verify.

7.5 Analysis Type

```
vsp.SetIntAnalysisInput(analysis_name, "GeomSet", [0])  # Panel method
```

Panel Method (GeomSet = 0):

- 3D surface panel method
- Includes viscous boundary layer corrections
- More accurate than vortex lattice for this validation
- Properly captures induced drag variations

7.6 No Symmetry Plane

```
# X-Z symmetry is NOT enabled (commented out)
# vsp.SetIntAnalysisInput(analysis_name, "Symmetry", [1])
```

Why Full Wing Analysis?

The validation analyzes the full wing (both halves) rather than using symmetry because:

1. **Complete visualization:** Allows verification of spanwise symmetry in results
2. **Debugging:** Easier to detect asymmetric issues
3. **Comprehensive validation:** Confirms optimizer handles full geometry correctly

Trade-off:

- Computational cost: 2× longer per evaluation

- Benefit: More thorough validation, easier result interpretation

For validation purposes, the extra time is justified.

---

8. Geometric Control Strategy

8.1 Leading and Trailing Edge Parameters

The validation code uses OpenVSP's leading/trailing edge control parameters to enable curved planforms:

```
# Root section (straight edges)
vsp.SetParmVal(wing_id, "OutLEMode", "XSec_0", 1)
vsp.SetParmVal(wing_id, "OutLESweep", "XSec_0", 0)
vsp.SetParmVal(wing_id, "OutLEStrength", "XSec_0", root_c)
vsp.SetParmVal(wing_id, "OutTEMode", "XSec_0", 1)
vsp.SetParmVal(wing_id, "OutTESweep", "XSec_0", 0)
vsp.SetParmVal(wing_id, "OutTEStrength", "XSec_0", root_c)

# Tip section (curved edges controlled by optimizer)
vsp.SetParmVal(wing_id, "InLEMode", "XSec_1", 1)
vsp.SetParmVal(wing_id, "InLESweep", "XSec_1", angle)
vsp.SetParmVal(wing_id, "InLEStrength", "XSec_1", strength)
vsp.SetParmVal(wing_id, "InTEMode", "XSec_1", 1)
vsp.SetParmVal(wing_id, "InTESweep", "XSec_1", -angle)
vsp.SetParmVal(wing_id, "InTEStrength", "XSec_1", strength)
```

8.2 Parameter Meanings

Mode = 1: Enables custom control of edge shape (rather than automatic)

Sweep: Angle of the edge tangent (controls curvature direction)

Strength: Magnitude of curvature (controls how much the edge curves)

8.3 Geometric Effect

Root Section (Center):

- Straight leading and trailing edges
- Forms the center of the planform
- Provides stable geometric reference

Tip Section:

- Curved leading edge with angle and strength
- Curved trailing edge with opposite angle (maintains symmetry)
- Creates smooth transition from root to tip

Combined Effect: By varying angle (0-60°) and strength (0.5-1.5), the optimizer can create planforms ranging from:

- Rectangular (angle=0, any strength)
- Tapered (small angle, low strength)
- Elliptical (optimal angle and strength)
- Highly curved (large angle, high strength)

8.4 Origin Location Adjustment

```
vsp.SetParmVal(wing_id, "X_Rel_Location", "XForm", -root_c/2)
```

Purpose: Shifts the wing so that the quarter-chord point of the root section is at the origin.

Why This Matters:

- Consistent moment reference point across all geometries
- Proper pitching moment calculations
- Standard aerodynamic reference convention

---

9. Optimization Setup

9.1 Algorithm Configuration

```
swarm_size = 70
algorithm = SMPSO(
    problem,
    swarm_size=70,
    leader_size=70,
    mutation=PM(probability=1.0/6.0, distribution_index=5)
)
max_evaluations = 100
```

9.2 Parameter Justification

Swarm Size = 70:

- 5 design variables
- Rule of thumb: 10-15 × n_vars
- 70 ≈ 14 × 5 (adequate for this dimensionality)
- Larger than main optimizer (50) for more thorough exploration

Max Evaluations = 100:

- Total function calls: 70 × 100 = 7,000
- At 5-10 seconds per call: 10-20 hours total
- Sufficient for convergence on this problem

Leader Size = 70:

- Archive size equals swarm size
- Standard practice for single-objective problems
- All non-dominated solutions retained

Mutation Probability = 1/6 ≈ 0.167:

- Rule of thumb: 1/n_vars
- But n_vars = 5, so 1/5 = 0.2
- Using 1/6 is slightly more conservative
- Each variable has ~17% chance of mutation per iteration

Distribution Index = 5:

- Lower value (5 vs 20) means larger mutations
- Increases exploration capability
- Helps escape local minima
- Appropriate for validation where thoroughness matters more than speed

9.3 Progress Tracking

```
progress_bar = tqdm(
    total=swarm_size*(max_evaluations/step),
    desc="VSPAERO Runs",
    unit="run"
)
```

Total runs: 70 × 100 = 7,000 VSPAERO evaluations

Display: Shows completion percentage, elapsed time, remaining time, and runs per second

9.4 Result Logging

Two CSV files track different aspects:

aero_results.csv: Every VSPAERO call

```
Iteration,Cl,Cdi,Cm,L/D
1,0.8234,0.0145,-0.0123,6.84
2,0.8456,0.0138,-0.0109,7.12
...
```

wingopt_results.csv: Best feasible solution per iteration

```
iteration,objective,root_c,tip_c,sweep,angle,strength
1,-0.0145,1.234,0.876,2.3,15.6,1.12
10,-0.0138,1.189,0.923,1.8,18.4,1.08
...
```

---

10. Why This Validation is Correct

10.1 Theoretical Soundness

Mathematical Foundation:

The validation is based on Prandtl's lifting-line theory, which provides a closed-form solution:

```
Cd_i_min = Cl² / (π × AR)  (for elliptical distribution, e=1)
```

Any other distribution gives:

```
Cd_i = Cl² / (π × AR × e)  where e < 1
```

Therefore, minimizing Cd_i at fixed Cl and AR is mathematically equivalent to maximizing e, which occurs uniquely for elliptical distribution.

No Assumptions Required:

This result doesn't depend on:

- Specific airfoil shape (works for any section)
- Reynolds number (within inviscid approximation)
- Angle of attack (as long as Cl is constant)
- Sweep angle (for small sweep)

10.2 Independent Discovery Principle

The validation is structured so the optimizer:

1. **Has no knowledge** of elliptical wings or their properties
2. **Only knows** it should minimize Cd_i while maintaining Cl
3. **Must discover** the optimal shape through numerical exploration
4. **Can be objectively verified** by comparing result to theory

This "blind" discovery is the gold standard for validation because:

- It cannot be a result of programming bias
- It tests the fundamental optimization capability
- It provides quantitative convergence metrics

10.3 Sensitivity and Specificity

Sensitivity: If the optimizer is working correctly, it WILL find the elliptical optimum (or very close to it).

Specificity: If the optimizer is broken (wrong constraints, bad objective, numerical errors), it will NOT find the elliptical optimum.

False Positives: Virtually impossible - randomly finding elliptical shape has probability ≈ 0

False Negatives: Also unlikely - theory guarantees the optimum exists and is unique

10.4 Quantifiable Metrics

The validation provides multiple quantifiable metrics:

Objective Function Value:

- Expected $Cd_i \approx Cl^2/(\pi \times AR) = 0.8571^2/(\pi \times 10) \approx 0.0233$
- Actual Cd_i from optimization
- Difference indicates optimization quality

Efficiency Factor:

```
e = Cl² / (π × AR × Cd_i)
```

- Theoretical maximum: e = 1.0
- Achieved value: e_actual
- Efficiency: (e_actual / 1.0) × 100%

Geometric Metrics:

- Chord distribution c(y) vs elliptical c_elliptical(y)
- RMS deviation from perfect ellipse
- Aspect ratio of best-fit ellipse

10.5 Robustness to Parameter Choices

The validation is robust because the theoretical result holds across wide ranges:

Lift Coefficient: Any positive Cl gives same conclusion (elliptical is best)

- Chosen Cl = 0.8571 is arbitrary but reasonable
- Could use 0.3, 0.5, 1.2 - result would be same

Angle of Attack: As long as Cl is constant, α doesn't matter

- Chosen α = 10° provides good Cl at reasonable angle
- Could use 5° or 15° with equivalent results

Reynolds Number: Induced drag is inviscid phenomenon

- Chosen Re = 600,000 is representative
- Theory holds for any Re (within VSPAERO's validity range)

Span: Absolute size doesn't affect optimal shape

- Chosen span = 5m is convenient
- Could use 2m or 10m - planform shape would be identical

10.6 Cross-Validation Opportunities

The validation can be independently verified through:

1. **Analytical Calculation:** Compare VSPAERO Cd_i to formula $Cl^2/(\pi \times AR \times e)$
2. **Literature Data:** Compare to published elliptical wing measurements
3. **Geometric Analysis:** Measure chord distribution and fit to ellipse equation
4. **Convergence Study:** Run with different swarm sizes/iterations, verify consistency
5. **Alternative Solvers:** Run same geometry in different CFD code (FlightStream, XFLR5)

---

11. Expected Results

11.1 Convergence Behavior

Phase 1: Exploration (Iterations 0-20)

- Objective function highly variable
- Many infeasible solutions (Cl violations)
- Wide range of planform shapes explored
- Cd_i values: 0.025-0.100+ (highly inefficient shapes)

Phase 2: Feasible Region Discovery (Iterations 20-40)

- Swarm gravitates toward Cl ≈ 0.8571
- Objective function variance decreases
- Rectangular and simple tapered planforms appear
- Cd_i values: 0.025-0.040

Phase 3: Shape Refinement (Iterations 40-70)

- Leading/trailing edge curvature increases
- Planforms become more elliptical
- Efficiency improves steadily
- Cd_i values: 0.024-0.028

Phase 4: Fine Tuning (Iterations 70-100)

- Minor adjustments to angle and strength
- Convergence to near-optimal shape
- Minimal improvement in objective
- Cd_i values: 0.0233-0.0236 (near theoretical minimum)

11.2 Expected Optimal Parameters

Based on theory and numerical approximations:

Root Chord: ~1.3-1.5 m

- Center of ellipse has maximum chord
- Exact value depends on how OpenVSP interpolates

Tip Chord: ~0.3-0.5 m

- Ellipse has non-zero chord at tip
- Should be approximately 25-35% of root chord

Sweep: ~0-2 degrees

- Minimal sweep is optimal for low-speed wings
- Small sweep may emerge for stability reasons

Angle: ~20-35 degrees

- Creates smooth curvature from root to tip
- Approximates elliptical arc

Strength: ~1.0-1.2

- Moderate strength smooths the transition
- Too low: piecewise linear, too high: overshoots

11.3 Expected Cd_i Value

Theoretical Minimum:

```
Cd_i_min = Cl² / (π × AR)
Cd_i_min = 0.8571² / (π × 10)
Cd_i_min = 0.7346 / 31.416
Cd_i_min = 0.02338
```

Expected Optimization Result:

```
Cd_i_optimized ≈ 0.0234-0.0238
```

Achievable Efficiency:

```
e = Cl² / (π × AR × Cd_i)
e = 0.7346 / (31.416 × 0.0236)
e ≈ 0.99
```

Efficiency: 99% of theoretical maximum

Why Not 100%?

- Discrete panel mesh approximates smooth ellipse

- Numerical convergence tolerances
- Optimizer stopping before absolute optimum
- OpenVSP's geometric parameterization limitations

Achieving 99% efficiency is excellent validation success.

11.4 Geometric Comparison

Elliptical Chord Distribution:

```
c(y) = c_0 × sqrt(1 - (2y/b)²)
```

For b = 5m (semi-span = 2.5m):

```
c(0) = c_0 = ~1.4m (center)
c(1.25m) = 0.866 × c_0 = ~1.2m
c(2.5m) = 0 (theory), ~0.4m (practical)
```

Optimized Wing Chord Distribution: Should closely match these values when measured at corresponding spanwise stations.

Quantitative Metric: Root-mean-square deviation from perfect ellipse:

```
RMS = sqrt(Σ(c_actual - c_ellipse)² / n) < 0.05m
```

RMS < 0.05m (5 cm) indicates excellent agreement.

---

12. Validation Success Criteria

12.1 Primary Criterion: Induced Drag

Threshold: $Cd_i \leq 0.0240$

Justification:

- Theoretical minimum: 0.02338
- Acceptable numerical error: ±2.5%
- $0.02338 \times 1.025 = 0.02396 \approx 0.0240$

Success: If $Cd_i \leq 0.0240$, optimizer has found near-optimal solution

12.2 Secondary Criterion: Efficiency Factor

Threshold: $e \geq 0.97$

Justification:

```
e = Cl² / (π × AR × Cd_i)
e = 0.7346 / (31.416 × 0.0240) = 0.974
```

Success: If $e \geq 0.97$, the lift distribution is highly elliptical

12.3 Tertiary Criterion: Geometric Shape

Visual Inspection: Planform should visually resemble an ellipse

Quantitative Measures:

1. Taper ratio = tip_c / root_c ≈ 0.25-0.35 (elliptical ratio)
2. Maximum chord at centerline (symmetric)
3. Smooth curvature (no discontinuities)

RMS Deviation: < 0.05m from perfect elliptical chord distribution

12.4 Convergence Criterion

Objective Function Stability: Last 10 iterations should show < 1% variation in best objective value

Example:

```
Iteration 90: Cd_i = 0.02355
Iteration 91: Cd_i = 0.02352
...
Iteration 100: Cd_i = 0.02349

Variation = (0.02355 - 0.02349) / 0.02349 = 0.26% ✓
```

Success: If variation < 1%, optimization has converged

12.5 Constraint Satisfaction

Lift Constraint: Final design must satisfy: $0.8471 \leq Cl \leq 0.8671$

Success: If best solution is marked "feasible" by optimizer

12.6 Repeatability

Multiple Runs: Run validation 3-5 times with different random seeds

Success: All runs converge to similar Cd_i (within ±3%) and similar planform shapes

Expected Variation:

- Cd_i: 0.0234-0.0238 across runs
- Geometric parameters: ±10% variation
- Final efficiency: e = 0.97-0.99

12.7 Overall Validation Verdict

PASS Criteria:

- Primary criterion met (Cd_i ≤ 0.0240): ✓
- Secondary criterion met (e ≥ 0.97): ✓
- Convergence achieved: ✓
- Constraints satisfied: ✓
- Planform shape qualitatively elliptical: ✓

If ALL criteria met: Optimization framework is VALIDATED

If ANY criterion fails: Framework has issues requiring investigation

---

13. Limitations and Considerations

13.1 Numerical Limitations

Panel Method Accuracy:

- VSPAERO panel method assumes inviscid flow with boundary layer corrections
- Induced drag calculation accurate to ±2-3%
- Mesh resolution limits geometric fidelity

Optimization Convergence:

- SMPSO is stochastic - doesn't guarantee global optimum
- Finite evaluation budget may stop before absolute best
- Local optima possible in high-dimensional space

Geometric Parameterization:

- OpenVSP's LE/TE control has limited expressiveness
- Cannot represent perfect mathematical ellipse
- Approximation quality depends on parameter ranges

13.2 Theoretical Assumptions

Lifting-Line Theory Assumptions:

1. **High aspect ratio:** Theory most accurate for AR > 6
   - Validation uses AR = 10 ✓
2. **Small angles:** Assumes small angle approximation
   - $\alpha = 10°$ is borderline but acceptable
3. **Planar wake:** Assumes wake doesn't roll up immediately
   - Panel method enforces this ✓
4. **Incompressible flow:** M = 0.15 is sufficiently low ✓

13.3 Practical Considerations

Computational Cost:

- 7,000 evaluations × 5-10 sec = 10-20 hours
- Long runtime limits parameter exploration
- Cannot easily run sensitivity studies

Non-Uniqueness:

- Multiple geometric configurations might achieve near-optimal Cd_i
- Different (angle, strength) combinations could approximate ellipse equally
- Optimizer may find different solutions on different runs

Edge Effects:

- Wing tips have finite thickness (tip chord > 0)
- True ellipse has zero chord at tip
- Practical constraints prevent perfect theoretical match

13.4 Validation Scope

What This Validates:

✓ Optimizer can minimize a single objective ✓ Constraint handling works correctly ✓ VSPAERO integration is functional ✓ Geometric parameterization is controllable ✓ Convergence mechanisms are effective

What This Does NOT Validate:

✗ Multi-objective optimization capability ✗ Complex constraint handling (stability, structures) ✗ High-dimensional problems (10+ variables) ✗ Different flight conditions (transonic, high-α) ✗ FlightStream integration

13.5 Potential Issues

Issue 1: Premature Convergence

- **Symptom:** Cd_i stops improving after 30-40 iterations
- **Cause:** Swarm collapses to local optimum
- **Solution:** Increase mutation rate or swarm size

Issue 2: Constraint Violation

- **Symptom:** Best solution has Cl outside [0.8471, 0.8671]
- **Cause:** Penalty weight too low or constraints improperly formulated
- **Solution:** Increase penalty multiplier from 0.1 to 0.2-0.5

Issue 3: Non-Elliptical Result

- **Symptom:** Cd_i ≈ 0.028-0.030 (e ≈ 0.83-0.87)
- **Cause:** Geometric parameterization insufficient
- **Solution:** Add more design variables or use different parameterization

Issue 4: Numerical Instability

- **Symptom:** VSPAERO crashes or returns invalid results

- **Cause:** Extreme geometric configurations
- **Solution:** Tighten variable bounds or add geometric constraints

13.6 Interpretation Guidelines

Strong Validation (Cd_i < 0.0235, e > 0.98):

- Optimization framework is highly reliable
- Proceed confidently to main optimization tasks
- Framework can handle complex objectives

Moderate Validation (Cd_i = 0.0235-0.0245, e = 0.95-0.98):

- Optimization framework is functional
- May need tuning for complex problems
- Acceptable for most practical applications

Weak Validation (Cd_i = 0.0245-0.0260, e = 0.90-0.95):

- Optimization framework has issues
- Investigate algorithm parameters, constraints
- Framework may struggle with difficult problems

Failed Validation (Cd_i > 0.0260, e < 0.90):

- Optimization framework is not working correctly
- Critical bugs in objective, constraints, or integration
- Do not proceed to main optimization until fixed

13.7 Extensions and Future Work

Validation Enhancement:

1. **Multi-Section Wings:** Test with 3-5 spanwise sections for finer resolution
2. **Twist Optimization:** Add washout as variable to test 3D optimization
3. **Different AR:** Validate at AR = 6, 8, 12 to verify robustness
4. **Different Airfoils:** Test with NACA 2412, ClarkY to verify airfoil independence
5. **Viscous Effects:** Compare panel method results to RANS CFD

Alternative Validation Approaches:

1. **Known Optimum:** Optimize a function with known mathematical solution
2. **Benchmark Problems:** Use standard optimization test problems
3. **Comparison Study:** Compare SMPSO to genetic algorithm, gradient-based optimizer
4. **Experimental Validation:** Build and test physical wing in wind tunnel

---

14. Conclusion

14.1 Validation Framework Strengths

Theoretically Grounded: Based on proven aerodynamic theory with 100 years of validation

Objective Benchmark: Clear, quantitative success criteria independent of subjective judgment

Comprehensive Testing: Exercises all critical components - geometry, analysis, optimization, constraints

Self-Contained: Does not require external reference data or complicated setup

Repeatable: Can be run multiple times to verify statistical consistency

14.2 Why This Approach is Scientifically Rigorous

The validation satisfies key scientific principles:

1. **Falsifiability:** Clear criteria for pass/fail
2. **Reproducibility:** Can be independently verified
3. **Theoretical Foundation:** Based on fundamental physics
4. **Quantitative Metrics:** Objective numerical measures
5. **Independence:** Optimizer discovers result without being told

14.3 Confidence in Main Optimizer

If validation passes, we can be confident that:

- The optimization algorithm (SMPSO) functions correctly
- VSPAERO integration is reliable and accurate
- Constraint handling works as intended
- Geometric parameterization is adequate
- Result logging and tracking are accurate
- The framework can find global optima in complex spaces

This confidence transfers directly to the main endurance optimization problem, which has similar structure but different objective and constraints.

14.4 Final Assessment

The validation framework is **scientifically sound, computationally rigorous, and practically useful**. It provides strong evidence that the optimization system works correctly by demonstrating that it can independently discover a known theoretical optimum (elliptical planform for minimum induced drag) without being explicitly programmed with that knowledge.

This is the gold standard for validation in optimization: showing that the system discovers known optimal solutions through its own computational search, rather than being hard-coded to find them.

End of Validation Framework Documentation