AIAA SciTech Forum
11–15 & 19–21 January 2021, VIRTUAL EVENT
AIAA Scitech 2021 Forum

10.2514/6.2021-1893

Adjoint Design Optimization for Boundary Layer Ingesting Inlet Guide Vanes with Distorted Inlet Profiles in SU2

Aman uz zaman Baig* and Mark Turner †
University of Cincinnati, Cincinnati, OH, 45221

Adjoint Design Optimization is a gradient-based optimization technique that has the distinct advantage of computing the gradients of an objective function in an efficient manner as compared to gradient-free optimization methods. The open-source multiphysics code SU2 has adjoint optimization capability that has been demonstrated over the last few years. This paper will apply the adjoint optimization capability in SU2 to boundary layer ingesting inlet guide vanes. In order to use the cylindrical inlet profiles and interpolate it to the grid nodes, new interpolation capabilities are introduced in the code. The SU2 code is verified for its flow solution using a commercial flow solver. It is shown that in the unconstrained optimization a reduction of 7.54% in the entropy generation is achieved compared to an improvement of 3.16% when an exit flow angle constraint is applied. Limitations of the current free-form deformation parameterization approach in SU2 are also discussed. The IGV that will be used to demonstrate the adjoint capability in SU2 has been designed as a part of a three-blade row BLI propulsor earlier in the Gas Turbine Simulation Laboratory by using Genetic Algorithm using the inlet profile provided by NASA. This paper will utilize this as a test case for demonstrating adjoint design optimization capability with inlet profiles.

I. Nomenclature

ϕ	=	flow angle as the ratio of radial to axial velocities
α	=	swirl angle
P_T	=	Total Pressure
P_s	=	Static Pressure
V_r	=	Velocity in radial direction
V_z	=	Velocity in axial direction
V_θ	=	Velocity in circumferential direction
IGV	=	Inlet Guide Vanes
BLI	=	Boundary Layer Ingestion

II. Introduction

SU2 is a powerful multiphysics analysis and optimization tool which uses unstructured mesh topologies [1]. It has the capability to compute the gradients of the objective function by using adjoint which is much faster than other contemporary methods available. Adjoint in SU2 is being widely explored for a multitude of cases including unsteady harmonic problems, fluid-structure interaction optimizations, and several others [2–4]. However, adjoint optimizations for Turbomachinery applications are not currently the main strengths of the code which the author wants to address and introduce capabilities that can help achieve that goal.

The authors have been involved with NASA in the design of a BLI Propulsor for a Single Aisle Turbo-Electric Aircraft (STARC-ABL) concept. A BLI optimization case is distinct because the goal is not to only maximize the adiabatic efficiency of the propulsor but to also ideally achieve zero jet losses by minimizing the dissipation of kinetic energy downstream of the exit [5]. In order to achieve that it is important that we start by optimizing each blade row, reduce the losses and the distortion and then carry out a whole system optimization. Work on the optimization of a BLI inlet and the shape of the nacelle was done by Rodriguez et.al.[6] with another non-linear gradient based optimizer

*Graduate Student, Department of Aerospace Engineering, University of Cincinnati, Cincinnati, OH, 45221 and AIAA Student Member.
†Professor, Department of Aerospace Engineering, University of Cincinnati, ML 70, Cincinnati, OH, 45221.

NPSOL developed at Stanford University [7]. This paper will seek to use the gradient-based optimization capability in SU2 to optimize the Inlet Guide Vanes. BLI being the future of the next generation aircrafts, this capability can be of significant importance for carrying out efficient whole system analysis and optimizations for such highly complex systems.

This idea of Boundary Layer Ingestion propulsion first put forth by Smith in 1947 [8]. Smith showed that the aircraft that implemented boundary layer suction on the wing had about a 30% improvement in fuel efficiency and a 7% higher optimum cruise speed. Smith Jr. [9] in 1993 showed that the power saving is the greatest when the propulsor disk loading is high (more number of smaller propulsors), when the wake form factor (or shape factor) is high (flow near separation), and the propulsor design is such that the wake profile tends to be flattened at the exit of the engine (high wake recovery). This work was extended by Hall et al. [5] in 2017 focusing on the type of losses that can occur in a BLI configuration, identifying all the losses distinctly and suggesting a power-analysis based model to compare BLI and non-BLI configurations.

Although there is great promise in BLI technology, researchers are still skeptical of whether the benefits will be able to outweigh the disadvantages or not. The technology has been proven by the Navy [9] [10] [11] [12] but it is for incompressible flow and where there is minimal risk of separation. Compressible flow in air is much more complex. The greatest bottle-neck in the design of a BLI engine is to handle the distortion at the inlet and the tight coupling between the aircraft aerodynamics and the propulsion system presents a challenging design integration problem [13] [6]. The turbomachinery design has to be distortion tolerant and low loss, low drag inlet systems are key technologies to be worked on. Rodriguez [6] in 2009 published a multi-disciplinary design method for the inlet of a BLI engine. The process iterated between a CFD solution of the airframe and the propulsion system to achieve minimum losses at the inlet.

The most recent published work on adjoint optimization for turbomachinery in SU2 at the time of this writing is done by S.Vitale et al [14]. The authors have taken three example cases: an Aachen Turbine, an axial centrifugal turbine, and a mini–organic Rankine cycle radial turbine. After doing a general verification study for all three models by comparing with other codes and some published data, an adjoint optimization was carried out for the Aachen Turbine case using discrete adjoint. For all three models a y+ < 1.0 was maintained at the walls (hub, casing and blade surfaces). The SST $k - w$ model [15] was used for all three cases and all were converged to sufficient tolerances. Free form deformation (FFD) boxes [16] were used for deforming the mesh, with each blade row (or flow domain) having its seperate FFD box. Entropy Generation was introduced as the objective function. It is interesting to note that the original mesh for the Aachen Turbine had a total of 5 million elements but for the optimization they were reduced to only 600,000 elements with slip boundaries at the hub and the casing to reduce the computational cost. This work is significant in demonstrating the capability of adjoint optimization for turbomachinery optimizations in SU2. This thesis will also employ similar techniques in optimizing the IGV of the BLI propulsor.

Serna, in his thesis [17], implemented a parameter-based re-meshing toolchain with SU2 and Paraview [18]. The sensitivities with respect to the grid points were projected to the parameters. An extensive verification of adjoint gradients was carried out and shape optimization of three different cases in turbomachinery was done. Sun et al [19] also implemented a feature-based CAD parameterization by linking CATIA v5 for mesh generation and GMSH for mesh generation with SU2.

There has been a discussion as to what is the best way to parametrize the shape for an adjoint optimization. It is possible to use the airfoil parameters such as the leading edge curvature, chord, sweep, lean and camber for deforming the shape [17]. It is also possible to use a Free-Form deformation box [16] or Hick-Henne Shape Functions [20] to parametrize the geometry externally. Both of these options are built-in SU2. It is also possible to make the mesh points the parameters [21] although that may introduce problems of smoothness and continuity over the surface. Wu et al. [22] realized the most suitable way to analyze a 2D cascade was to use some external parametrization such as the Hick-Henne Bump Functions. In this work an FFD box will be used to change the shape of the 3D geometry. It is expected to tie the parametrization to T-Blade3 [23] in the future.

Adjoint optimization for internal flows and specifically turbomachinery has not been a widely explored area. Certainly there is some time before we can find accurate gradients for 3D multi-blade row realistic problems with mixing interfaces. The problem is made more difficult when we add unsteady effects and turbulence resolution with adjoint equations. However, since this area of research is getting more and more attention lately, we can hope we are moving fast towards that goal. This paper will contribute to research in the domain of adjoint optimization in turbomachinery using the SU2 code. More details regarding the process and results can be found in the author's thesis [24].

III. Inlet Interpolation

A. Implementation of Inlet Interpolation in SU2

In BLI applications it is necessary that we give an inlet profile during the simulations. Since SU2 had a rather inconvenient way of taking in inlet profiles, where one had to give the state variable values at each grid node in cartesian format, an initiative was taken to allow SU2 to accept inlet profiles where one can give the swirl angle α,radial flow angle ϕ, total temperature, total pressure, and the turbulence parameters in cylindrical coordinates format - as usually done for turbomachinery. These angles are defined as in equations (1)-(3).

$$\alpha = tan^{-1}\left(\frac{V_\theta}{V_m}\right) \tag{1}$$

$$\phi = tan^{-1}\left(\frac{V_r}{V_z}\right) \tag{2}$$

$$V_m = \sqrt{(V_z)^2 + (V_r)^2} \tag{3}$$

where, V_z, V_r and V_θ are the velocities in z,r and theta directions.

Due to this capability, it will be intuitive to look at the inlet file and easily understand what kind of profile is being given at the inlet. This capability will be critical if the inlet shape is being optimized where the grid nodes are bound to move and will require re-projection of the inlet profile over the grid nodes for every other design iteration. Optimization of the inlet shape is one of the main areas of research in BLI design. This capability can also be used to safely differentiate the inlet nodes as the Akima interpolation is first-order continuous. This capability is now part of the master branch of SU2.

B. The Akima Spline

Akima Interpolation [25] [26] is a piece-wise cubic interpolation function.

For a given set of points x_i, y_i:

$$A(x) \mid_{x\in[x_i,x_{i+1}]} = a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3 = A_i(x) \tag{4}$$

where,

$$a_i = y_i, b_i = A_i^{'}, c_i = \frac{3p_i - 2A_i^{'} - A_{i+1}^{'}}{\Delta x_i}, d_i = \frac{A_i^{'} + A_{i+1}^{'} - 2p_i}{(\Delta x_i)^2} \tag{5}$$

and, p_i is defined by:

$$p_i = \frac{\Delta y_i}{\Delta x_i} \Delta y_i = y_{i+1} - y_i, \Delta x_i = x_{i+1} - x_i \tag{6}$$

Note that the Δy_i and Δx_i are taken as forward differences.

The boundary conditions are derived from the continuity of the sub-spline and its first derivative,

$$A_i(x_i) = y_i, A_i^{'}(x_i) = A_i^{'}, A_i(x_{i+1}) = y_{i+1}, A_i^{'}(x_{i+1}) = A_{i+1}^{'} \tag{7}$$

At the boundaries the following conditions were applied,

$$A_1^{'} = p_1, A_2^{'} = \frac{1}{2}p_1 + \frac{1}{2}p_2, A_n^{'} = p_{n-1}, A_{n-1}^{'} = \frac{1}{2}p_{n-1} + \frac{1}{2}p_{n-2} \tag{8}$$

The program first evaluates the sub-spline and the queries the grid coordinates to interpolate on that grid node. As such this interpolation algorithm needs to run once for every sub-spline.

Akima interpolation has been implemented such that it will be able to extrapolate from data points but will give a warning when doing so.

C. Linear Interpolation

Simple 1D linear interpolation is implemented using forward differences,

$$S_i(x) = y_i + p_i(x_i) \tag{9}$$

where,

$$p_i = \frac{\Delta y_i}{\Delta x_i}, \Delta y_i = y_{i+1} - y_i, \Delta x_i = x_{i+1} - x_i \tag{10}$$

Linear interpolation cannot extrapolate and it will give an error if the grid coordinates radius exceed that what is given in the inlet file. This can easily be rectified by adding another line for that radius. This can also be tuned to achieve zero-order interpolation.

D. Comparison between both splines

The graphs in 1 will compare the interpolation behavior of total pressure and inlet turbulence variable for SA model ν [27]. These are not the original profiles used for the simulations but they are presented here to demonstrate how the interpolation routines will differ in their interpolation behavior.

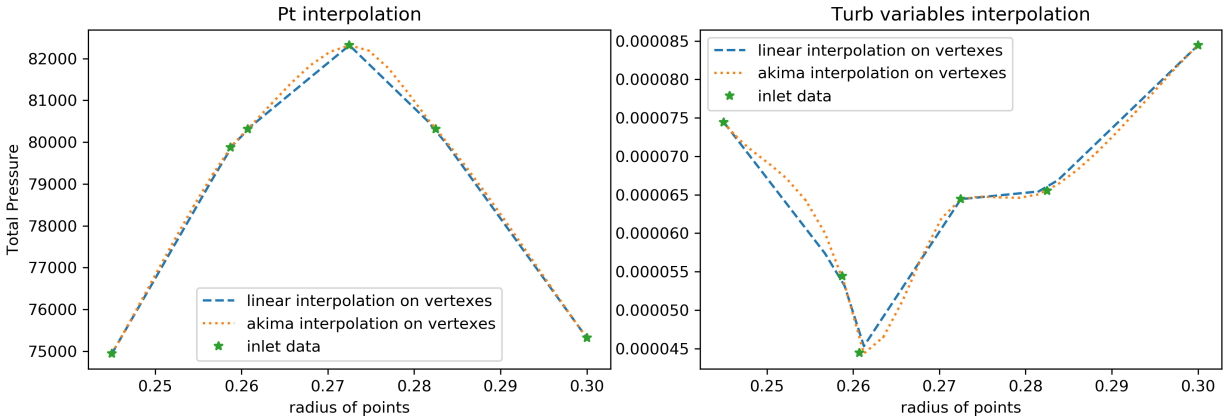

Fig. 1 Akima and Linear Interpolation comparison

If the consecutive data points are fluctuating significantly, Akima interpolation may induce some points of inflexion, albeit less than a third-order cubic spline. It should be noted that the realistic boundary layer data is usually monotonic thus such fluctuations by Akima spline would be rare in practice.

Akima Interpolation should be preferred if the input values are very coarse as it can maintain first-order continuity with wiggles less than that of a third-order cubic spline. If there are any unnecessary wiggles in the Akima spline, the number of data points can be increased to allow for a more defined transition. If the boundary points are to be differentiated then the Akima spline would be an ideal choice.

IV. Flow solution comparison

A. Geometry and grid generation

SU2's native mesh format is a unique .su2 ASCII file format. It can also only accept unstructured grids as it is an unstructured solver.

The geometry was generated using the input files for T-Blade3 using *spancontrol* and *3dbgbinput* files that are particular to this software. These files can be found online at [28]. T-Blade3 then generates a .geomturbo file which can be loaded in Numeca Autogrid which creates a blocked-structured mesh. The mesh is loaded into the commercial solver for the simulation. The mesh is also exported as a plot3D file to Pointwise and the structured grid is converted to an unstructured grid (by merging/linking the repeated, overlapping nodes) and exported to .su2 format. A binary Tecplot

file is selected as the output for SU2. It was observed that with ASCII Tecplot files, the grid coordinates and the results were inaccurate and of lower precision. Similar Tecplot scripts and strategies were used for results from both solvers so the post-processing can be uniform and be done quickly. Section and slice data was exported from Tecplot for plotting with Python. This process is shown schematically in Fig 2.

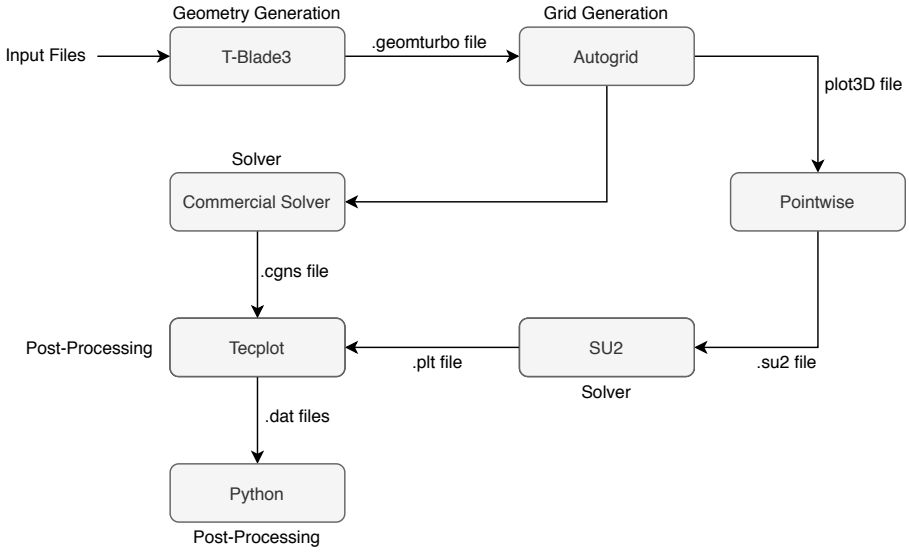

Fig. 2 The complete tool chain for pre- and post-processing that was adopted.

The grid that was used for the comparison has a total number of 964681 points and 925120 elements. The convergence criteria was to bring the density residual to E-9. This was deemed sufficient for both solvers. The grid is shown in Fig 3.

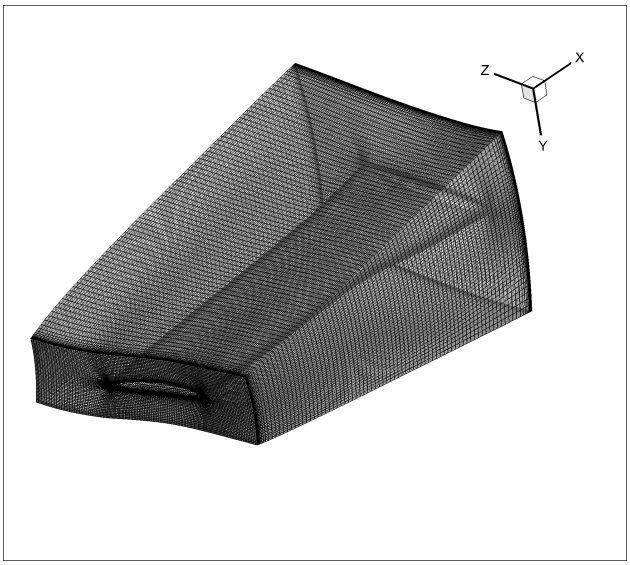

Fig. 3 The volume mesh for the IGV case.

B. Inlet Results

The applied inlet profile was a 1D profile with a variation in total pressure and the ϕ angle. The results after applying the inlet profile in SU2 compared with the commercial solver, called 'FT' here, are shown in 4.

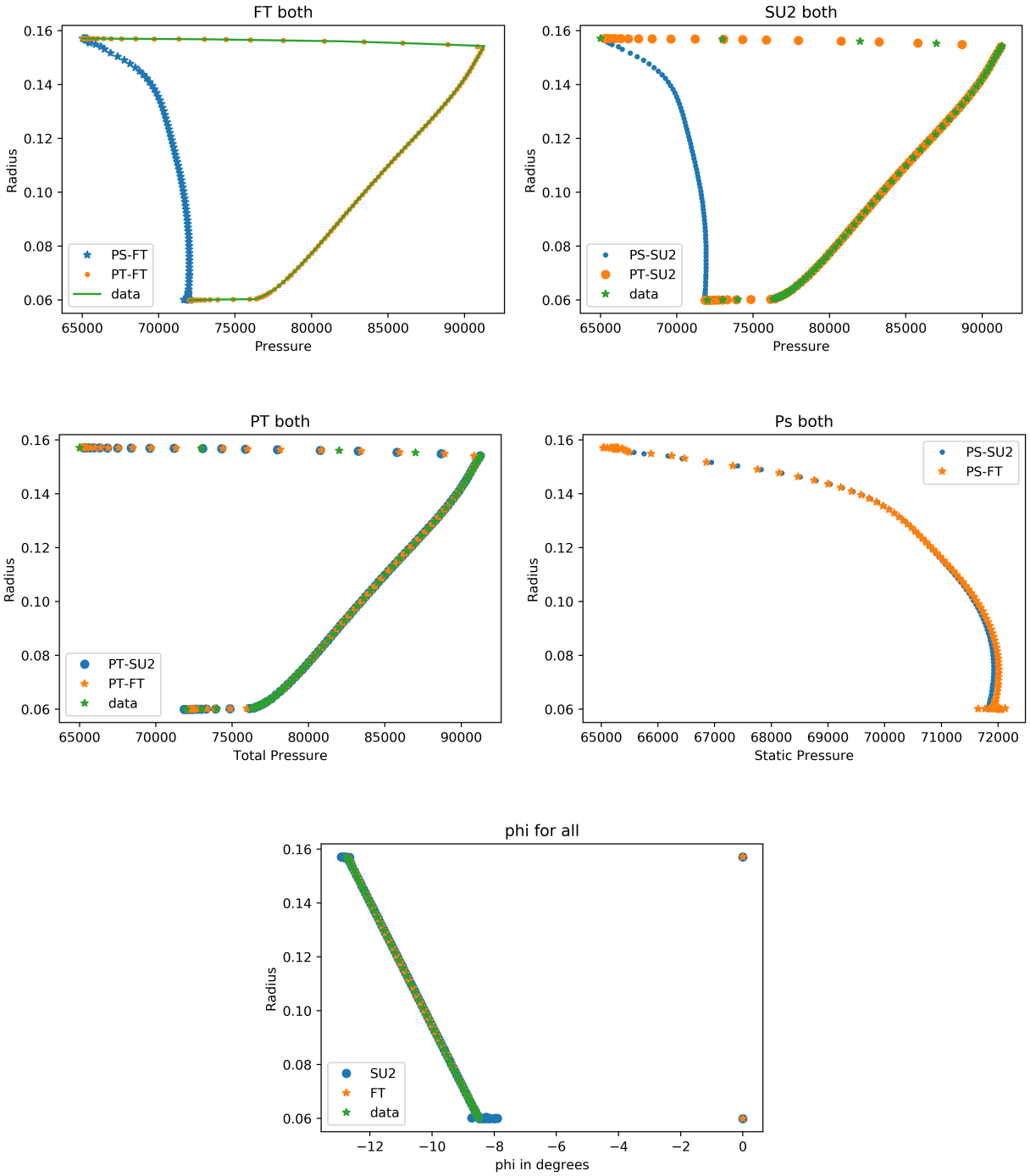

Fig. 4 Inlet Profile plots

C. Blade Results

Similar to the way the inlet was analyzed, the distribution of state variables over the blade was also analyzed. Because the velocities are zero on the blade surfaces due to the no-slip condition, the total and static quantities were the same over the blade and thus only plots of the static quantities are shown. Entropy is also calculated over the blade for both solvers. The flow direction is +z.

The 12 blade sections that were analyzed are shown in Figs 5.

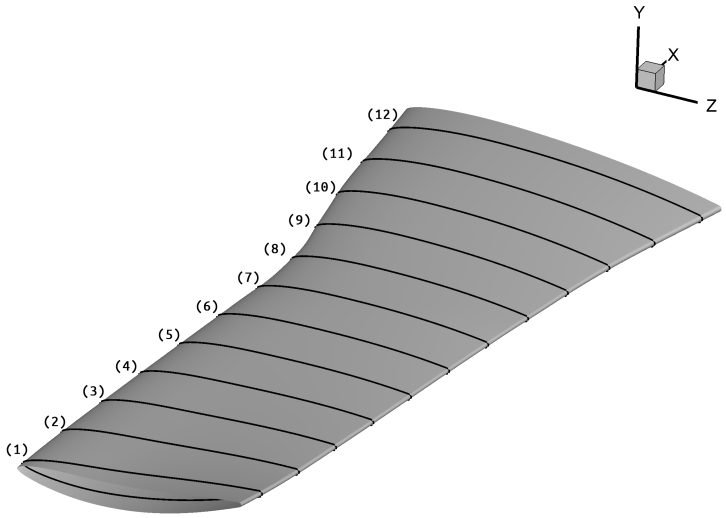

Fig. 5 Blade sections corresponding to the plots.

Results for section (11) can be seen in 6. Results for other sections were also very similar, they are not shown here to keep it concise.

The results are closely matching for both solvers as can be seen in the plots. To visualize the pressure and temperature distribution on the blade one can also look at the contour plots as shown in 7.

It was concluded after this comparison on the blade that the results are satisfactorily matching between both the solvers.

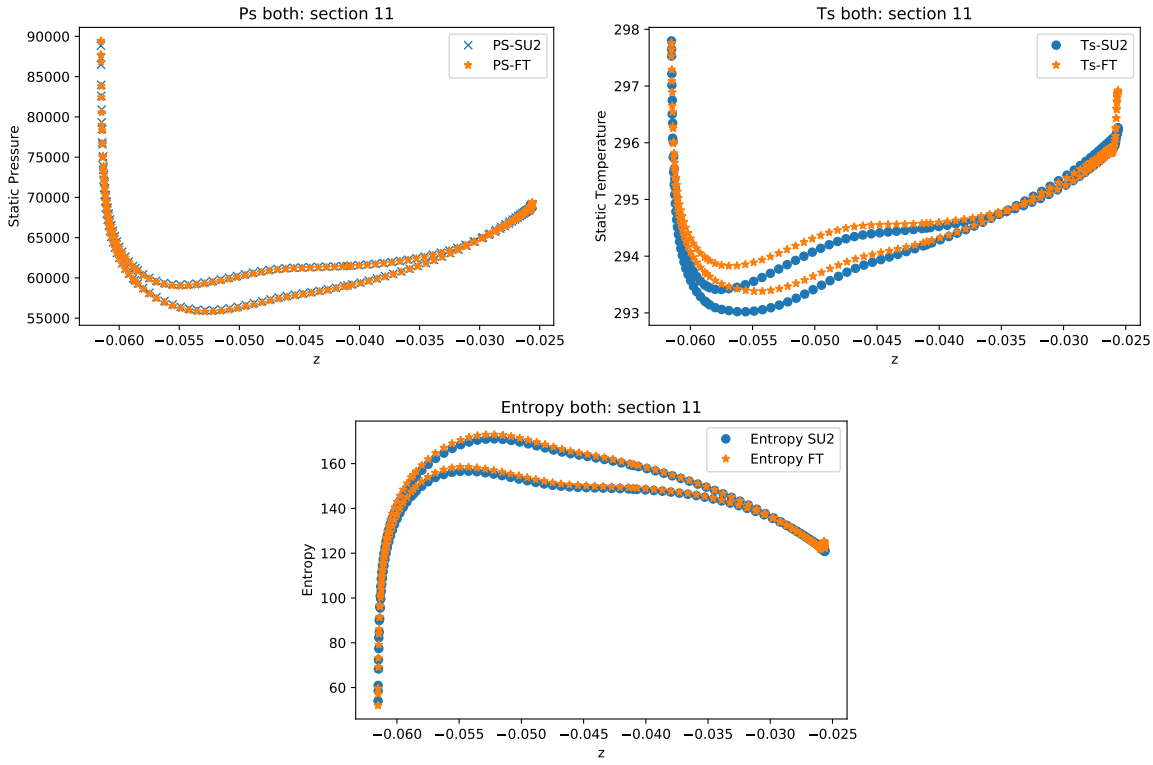

Fig. 6 Section (11) results for blade.

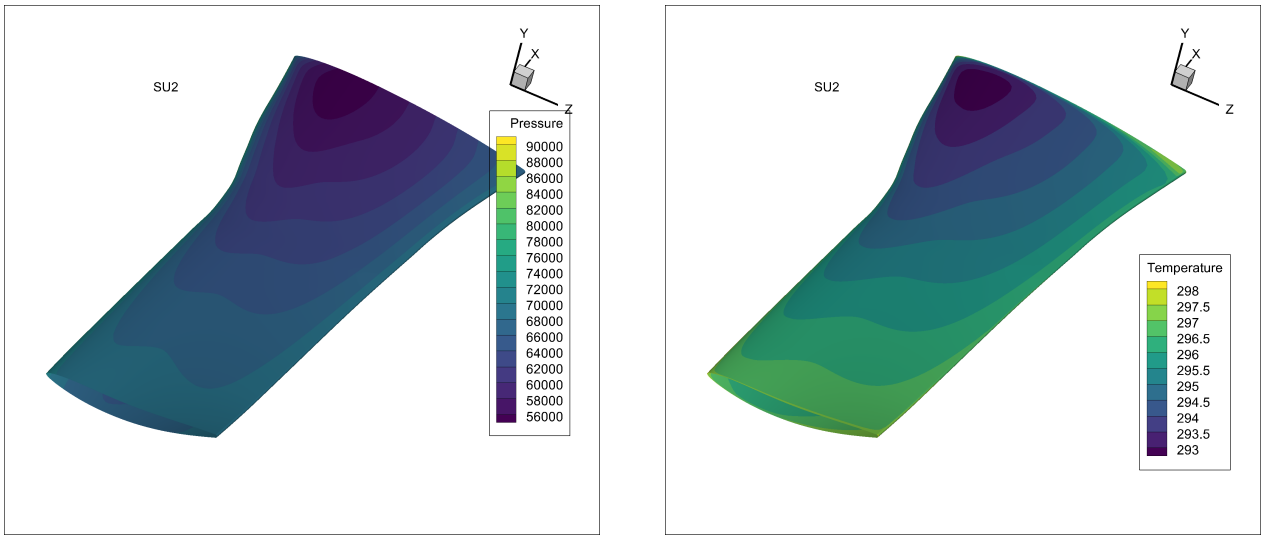

Fig. 7 Blade Contour plots for the top surface for SU2.

D. Outlet Results

The Radial Equilibrium boundary condition was used at the outlet. Simple Radial Equilibrium is defined as,

$$\frac{\partial p}{\partial r} = \rho \frac{(v_\theta)^2}{r} \tag{11}$$

where,

r was taken as the mid-span radius for both solvers.

Non-reflecting boundary conditions were used which can be used in SU2 by using `MARKER_GILES`. The outlet profiles were compared in detail to make sure the correct boundary conditions have been applied.

The outlet plane was divided into 5 sections as shown in Fig 8.

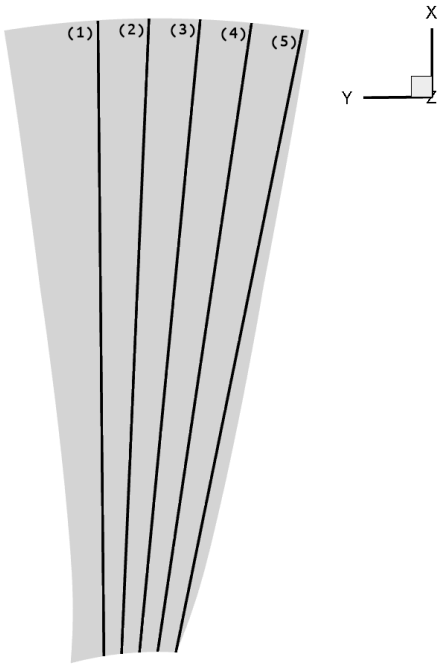

Fig. 8 Outlet sections corresponding to the plots.

Results for section 1 are shown in Fig 9 and Fig 10. Results for other sections were also very similar, they are not shown here to keep it concise. It can be seen that the results are closely matching for both solvers. However, the static pressure plot shows that there is a difference in how the boundary conditions are applied for both solvers. This points out that the implementation non-reflecting boundary conditions in SU2 for 3D cases may be imperfect. An outlet further downstream may alleviate this issue.

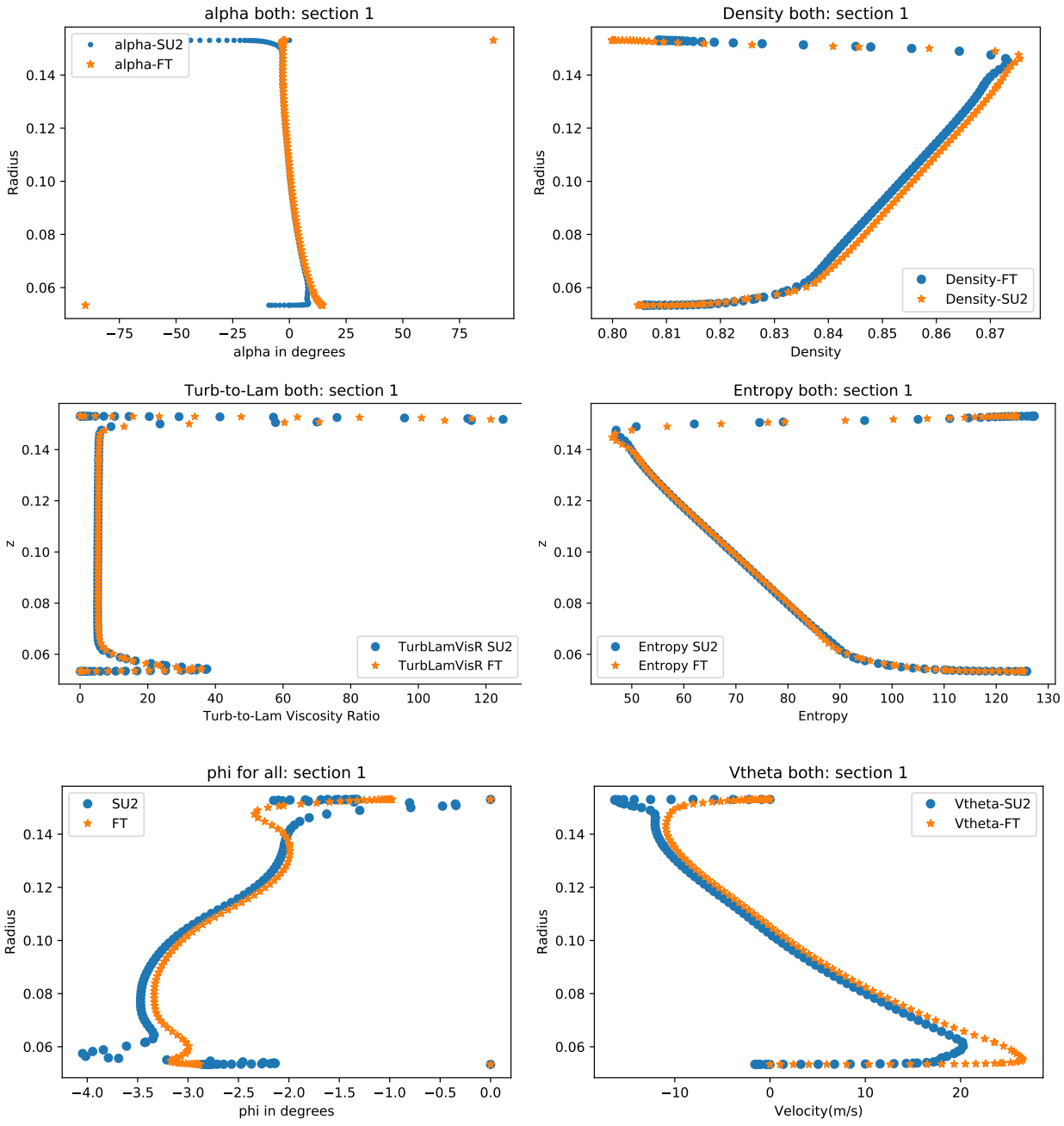

Fig. 9 Outlet Plane Results: section 1.

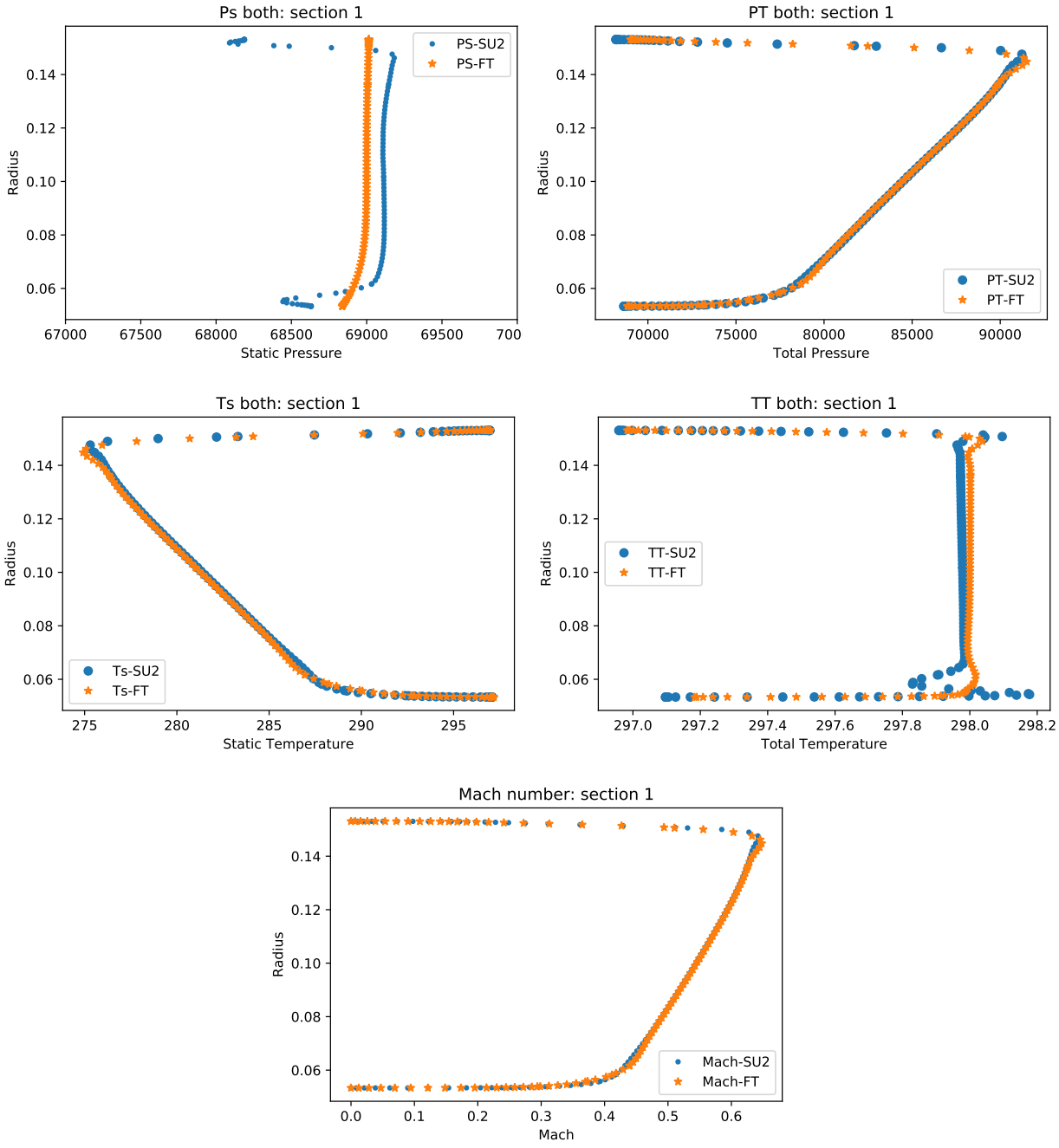

Fig. 10 Outlet Plane Results 2: section 1.

E. Mass Flow and Loss comparison

Table 1 shows the mass flow $\dot{m}$, total pressure, static pressure and static temperature mass-flow averaged averaged values at the inlet and outlet faces in both solvers.

	Inlet		**Outlet**	
	Commercial Solver	**SU2**	**Commercial Solver**	**SU2**
$\dot{m}$ **[kg/s]**	10.24	10.249	10.24	10.2351
Pt [Pa]	86258.87	86222.93	85924.41	85883.56
Ps [Pa]	69563.25	69222.13	68992.66	69000.52
Ts [K]	297.95	297.96	280.01	280.04
TT [K]	298.00	298.00	298.01	298.01

Table 1 Mass-flow averaged quantities at inlet and outlet.

In order to quantify the loss, the turbine loss coefficient is used. Since the IGVs are accelerating the flow, a turbine loss coefficient appears most appropriate.

It is defined as,

$$\zeta = \frac{P_{t1} - P_{t2}}{P_{t2} - P_{s2}} \tag{12}$$

The loss coefficient for both solvers are given in Table 2.

	Commercial Solver	**SU2**	**|Difference|**
zeta	0.01975	0.02010	1.77%

Table 2 Loss coefficients for both solvers.

SU2 is predicting a slightly higher loss and that could be attributed to the reflections that are occurring at the outlet boundary causing additional losses. The overall results appear to be in good agreement for both solvers under reasonable tolerances.

V. Adjoint Optimization

A. Shape Optimization overview in SU2

SU2 has the capability to calculate the discrete and continuous adjoint solutions from the flow solutions. This feature is enabled through the python *scipy* optimization library with a couple of independent python scripts that drive the SU2 C++ executables. This makes a reliable and modular architecture for running shape optimization problems.

The optimization runs in the following sequence (for discrete adjoint using AD):

- The flow solution is calculated using `SU2_CFD`
- The adjoint solution is calculated using `SU2_CFD_AD`
- The sensitivities of the design parameters are achieved by projecting the volume sensitivities on the design variables by running `SU2_DOT_AD`
- The mesh is then deformed using `SU2_DEF` where the deformation value is determined by multiplying the sensitivities with the relaxation factors
- The shape optimization script then starts to move in the direction as determined by the adjoint sensitivities until it turns out to be incorrect and the gradient needs re-evaluation
- This process continues until the convergence criteria for the minimum gradient is satisfied

For this optimization process, a free-form deformation box was used to parameterize the shape of the blade and part of the hub and the casing were also allowed to deform. This allows more flexibility in design and mesh deformation and allows the optimizer to achieve more design iterations without failing to deform the mesh. Sequential Least Squares Programming (SLSQP) in python was used as the optimizer which can be called directly from the shape optimization scripts in SU2.

B. Verification of adjoint gradients

One of the important steps for running a gradient-based optimization is to verify the functional gradients with respect to the design variables. This can be done via finite differences by taking the forward difference in the following manner and comparing it with the adjoint gradients.

$$\frac{dJ}{d(DV)} \approx \frac{(J_{i+1} - J_i)}{\delta DV} \tag{13}$$

where the smaller the δDV, the more close the gradients are to the accurate adjoint gradients. This process can have a high uncertainty as a large deformation step can cause the gradients to be inaccurate. There will be truncation and rounding off errors as well although they can be minimized by using central differences. This process also assumes that the sensitivities are changing linearly with the design parameters deformation which may not be the case especially if there are a large number of design parameters as in this case. Additionally to verify the adjoint gradients via finite differences is very time consuming and the solution time scales proportionally with the number of design parameters present. The more the design parameters, the more the possible deformed designs need to be run individually for a flow solution. In our case, one flow solution takes about 24 hours to run on 40 cores. For 352 DV's this would be 352 full days.

Another reason for uncertainty in this process are the small values of the adjoint gradients. If the gradients are small,

$$\delta J = gradient * \delta DV \tag{14}$$

the δJ can be very small. If we try to increase δDV then as discussed before, the results will be inaccurate. This means the results have to converge to extremely high precision if the gradients are small for the finite differences to be valid. This was the most significant issue in verifying the adjoint gradients for this case as the adjoint gradients were of the order of almost 1E-4. The reference length of the geometry was of the order 1E-3. This would mean that the δDV needs to be almost 1E-6 for the gradients to be accurate, as only small deformations are desired. This means that our δJ needs to be accurate up to at least 10 significant figures which is why the flow solution was run much longer and up to 120,000 iterations during the optimization to get a very precise value of the objective function otherwise the optimizer would have been misguided if it was going in the correct direction but due to lack of convergence it calculated an inaccurate objective function value. However, for finite differences, this approach also failed as the residuals started to oscillate at about 8 significant figures of accuracy.

With larger steps of $\delta DV = 1E - 3$, it was possible to get some calculable change in δJ but the errors were very large from 10% to 50%. As we reduce the δDV, the error starts reducing but the gradient can no longer be evaluated as

the objective function values are almost the same (negligible mesh deformation). Too small of a δDV can introduce round-off errors as well. A total of 7 DVs were analyzed and only 1 of them had an error of less than 1% with 1E-3 of deformation. All of them, however, were pointing in the correct gradient direction. One way to reduce the error can be to use a higher order finite difference, but since the objective function value oscillates so much, the value at every δDV will be a guess or a rough average which will cause significant error. It is also possible to use a coarser or a finer mesh which may have difference convergence properties. Due to the impracticality of this approach for this case, the finite difference verification effort was stopped after showing gradients in the correct direction and verifying to the extent presented. It was realized that if the adjoint optimization proceeded as expected intuitively, that is, by reducing the blade thickness to reduce the entropy generation, the adjoint solver and the gradients can be partially *verified* albeit in a crude way. It will be observed in the next section that it was indeed the case and the optimizer attempted to flatten the blade to reduce the objective function.

C. Unconstrained Optimization Results

The optimization results are as shown in table 3:

NIT	FC	OBJFUN	GNORM
1	1	6.221928E-04	5.981149E-06
2	2	5.887454E-04	5.175949E-06
3	4	5.752168E-04	4.899583E-06

Table 3 Optimizer Iterations for the unconstrained optimization.

The optimizer here was able to reduce the entropy generation by 7.54% which is a significant improvement relative to the former unconstrained optimization. The solution diverged after three design iterations.

Fig 11 shows the mesh at the final design iteration. It can be seen that though there are no negative volumes near the hub, the skewness of the mesh is high and that may lead to unreliable results. This begs the question of using more sophisticated mesh deforming or re-meshing strategies that can be implemented.

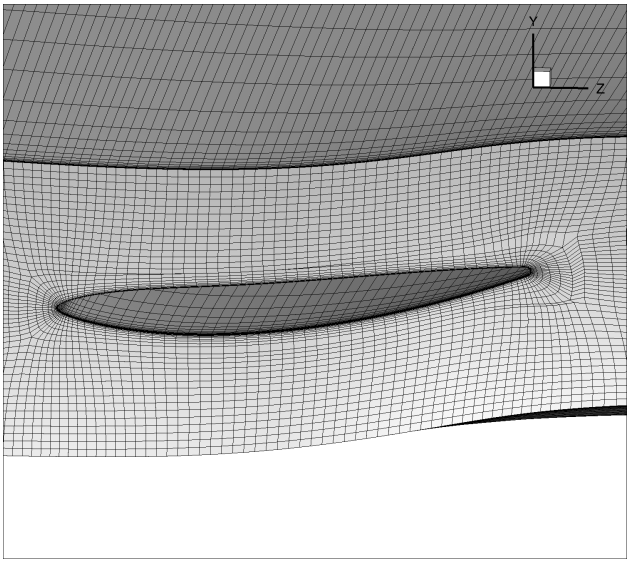

Fig. 11 The mesh at the hub wall at the final design iteration. No negative volumes, but low mesh quality.

The change in the blade shape in the three designs is shown in Fig 12.

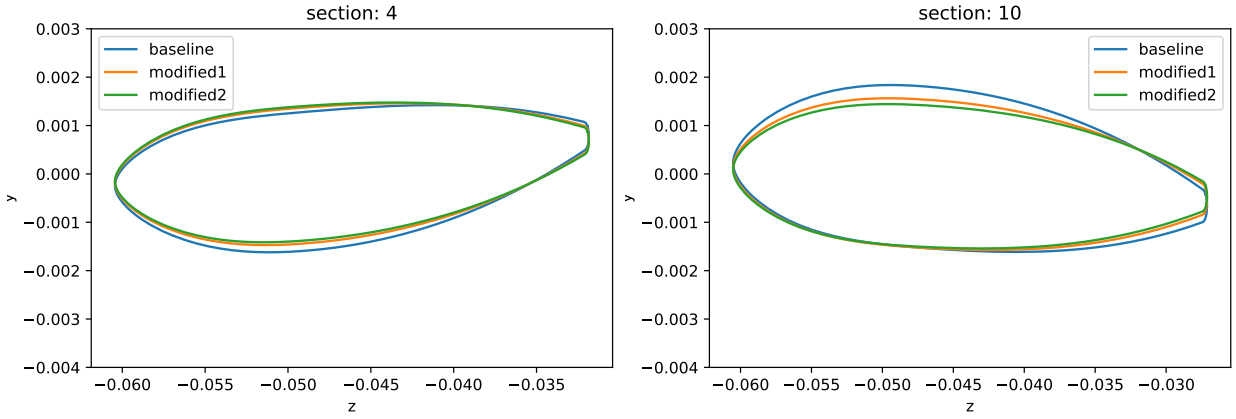

Fig. 12 Blade deformation for the unconstrained optimization.

NIT	FC	OBJFUN	GNORM
1	1	6.221451E-04	5.925290E-06
2	2	6.025055E-04	5.879590E-06
3	3	6.024721E-04	5.879121E-06

Table 4 Constrained Optimization run.

D. Constrained Optimization

The objective function in the constrained optimization was the same as in the unconstrained optimization, that is, to reduce the entropy generation. However, a constraint was applied this time where the outlet ϕ angle was constrained to be less than -2°. This constraint was taken to demonstrate the capability with a constraint rather than a physical constraint useful for design. This is defined in the config file as follows:

`OPT_CONSTRAINT = (FLOW_ANGLE_OUT < -2.0) * 1E-3`,

where the angle is multiplied with the scaling factor. The ϕ angle was defined in *FlowAngleOut* in SU2 as $atan(V_r/V_z)$.

An optimization scaling factor was also given to scale down the mesh deformation. This was defined as,

`OPT_RELAX_FACTOR = 1E-3`

This factor was carefully chosen after some manual design iterations to make sure no negative volumes were introduced in the first few design iterations. This factor was not the same as in unconstrained optimization since the adjoint solution for *FlowAngleOut* was calculated as well, the gradients and mesh deformation values were different and needed different relaxation factors. This factor is found to be most reliable and robust in scaling mesh deformation in the config file.

The change in the optimization direction because of the constraint was evident. The optimization progress is shown as in table 4. It can be seen that the objective function value has reduced as expected and the gradient value is also falling as it is approaching the minimum.

The reduction in entropy generation with the constraint applied was 3.16%. This was significantly less than the unconstrained optimization. This is expected, as when a constraint is applied the design space for a problem shrinks and limits how much the parameters can change. The change in the flow angle is shown in table 5. It can be seen that the constraint is satisfied in the first design iteration.

However, this is a mass-averaged value of the ϕ angle and can be higher or lesser than -2° in some spanwise locations. To check this a plot for the outlet sections as shown in Fig 13 was plotted for ϕ variation in the radial direction for the final design. It can be seen that the ϕ has skewed towards -2° in the constrained optimization. This has resulted in the mass-flow averaged ϕ to be slightly less than -2° but a significant portion of the span has an angle higher than the constraint. The mass-averaged flow angle out was also calculated in Tecplot from the outlet plane results to verify that the average being calculated by SU2 is correct.

Design Iteration	Unconstrained Optimization(°)	Constrained Optimization(°)
1	-1.7488572075	-1.752913895
2	-1.8469497044	-2.0021237641
3	-1.8954953081	-2.0024126407

Table 5 ϕ difference in optimization.

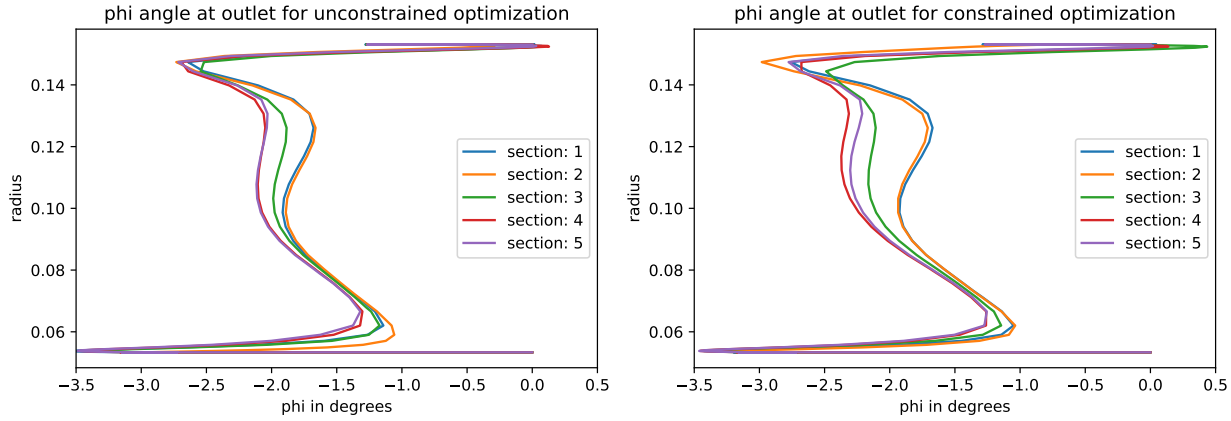

Fig. 13 ϕ variation for unconstrained (left) and constrained (right) optimization at the outlet plane.

The Fig 14 shows the change in shape in the constrained and the unconstrained optimization of the blade. It can be seen that the optimizer chooses to move in a different direction while reducing the objective function and satisfying the constraint simultaneously.

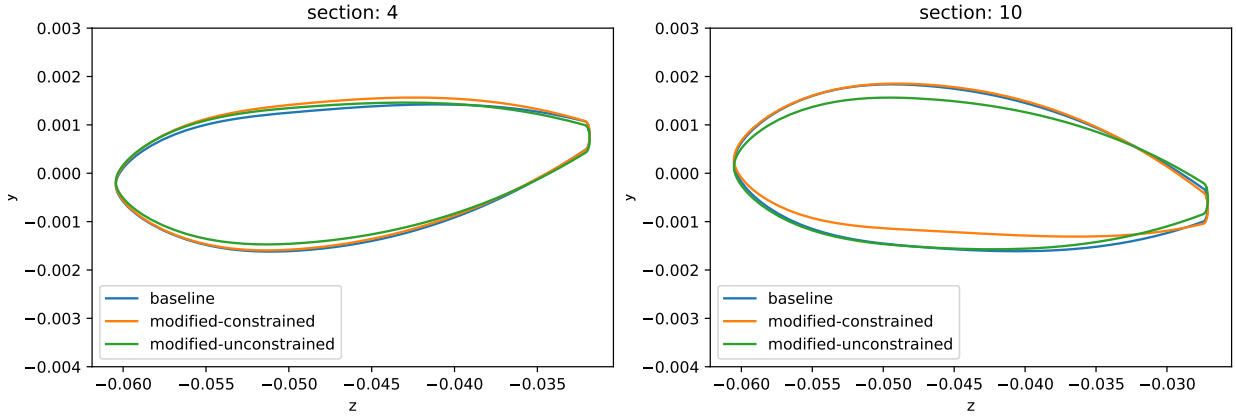

Fig. 14 Blade deformation for constrained and unconstrained optimization.

VI. Conclusion

In this paper a new inlet interpolation capability added by the author was discussed and it was verified through the flow solution analysis at the inlet face that it is being correctly applied. The flow solution in SU2 was then verified against a commercial software through various line and contour plots at the inlet and outlet face, blade surface and other iso-contours and constant-span plots, not all of which have been discussed in this paper to keep it concise. This work led us to verify SU2 as a reliable flow solver for single blade row stationary turbomachinery cases.

The adjoint optimization capability in SU2 was then used to optimize the blade shape. In the unconstrained optimization a reduction of 7.54% was achieved in the entropy generation compared to a reduction of 3.16% with an exit flow angle constrained applied. However, a deterioration in mesh quality was also observed as the design iterations proceeded. After three design iterations the mesh deformation routine created negative volumes at the hub and casing which limited the change that the design could undergo. A better mesh deformation or parameterization scheme if implemented in SU2 can help alleviate this issue. It was also noted that the constraint is applied a mass-flow averaged value and not to every part of the span.

This paper demonstrated the feasibility and capability of the open-source code SU2 for flow solution and adjoint optimization for stationary turbomachinery cases. It was seen that generally the code works really well but it does have some limitations when it comes to applying non-reflecting boundary conditions at the outlet or mesh deformation beyond a few design iterations. Suggestions on how these limitations could be overcome were given. This paper did show that even with a relatively coarse mesh one could still achieve significant gains from adjoint optimization in SU2. In the future, a more realistic problem with multiple objectives and constraints, a higher mesh density and with enough computational resources, can be run on SU2 to determine how well it performs for more advanced cases.

Acknowledgments

This project is part of the optimization study that the Gas Turbine Simulation Laboratory at the University of Cincinnati carried out for the Boundary Layer Ingestion Propulsor project from NASA. The authors would also like to acknowledge the funding received from NASA for this project. Additionally, the authors would also like to acknowledge that this research was made possible in part through research cyber infrastructure resources and services provided by the Advanced Research Computing (ARC) center at the University of Cincinnati.

References

[1] Economon, T. D., Palacios, F., Copeland, S. R., Lukaczyk, T. W., and Alonso, J. J., "SU2: An open-source suite for multiphysics simulation and design," Vol. 54, No. 3, 2015, pp. 828–846.

[2] Rubino, A., Pini, M., Colonna, P., Albring, T., Nimmagadda, S., Economon, T., and Alonso, J., "Adjoint-based fluid dynamic design optimization in quasi-periodic unsteady flow problems using a harmonic balance method," Vol. 372, 2018, pp. 220–235.

[3] Venkatesan-Crome, C., Sanchez, R., and Palacios, R., "AERODYNAMIC OPTIMIZATION USING FSI COUPLED ADJOINTS IN SU2," Vol. 3, No. 4, 2018, p. 5.

[4] Kumar, D., Raisee, M., and Lacor, C., "Combination of Polynomial Chaos with Adjoint Formulations for Optimization Under Uncertainties," *Uncertainty Management for Robust Industrial Design in Aeronautics*, Springer, 2019, pp. 567–582.

[5] Hall, D. K., Huang, A. C., Uranga, A., Greitzer, E. M., Drela, M., and Sato, S., "Boundary Layer Ingestion Propulsion Benefit for Transport Aircraft," *Journal of Propulsion and Power*, Vol. 33, No. 5, 2017, pp. 1118–1129.

[6] Rodriguez, D. L., "Multidisciplinary Optimization Method for Designing Boundary-Layer-Ingesting Inlets," *Journal of Aircraft*, Vol. 46, No. 3, 2009, pp. 883–894.

[7] Philip, E., Murray, W., Saunders, M. A., and Wright, M. H., "Users Guide for NPSOL 5.0: A Fortan package for non-linear programming," 2001.

[8] AMO Smith, "The Jet Airplane Utilizing Boundary Layer Air for Propulsion," *Journal of the Aeronautical Sciences*, Vol. 14, No. 2, 1947, pp. 97–109.

[9] Smith, L. H., "Wake Ingestion Propulsion Benefit," *Journal of Propulsion and Power*, Vol. 9, No. 1, 1993, pp. 74–82.

[10] Wislicenus, G. F., "Hydrodynamics and Propulsion of Submerged Bodies," *ARS Journal*, Vol. 30, No. 12, 1960, pp. 1140–1148.

[11] Evanbar, M., and Thurston, S., "Efficiency of a Propulsor on a Body of Revolution-Inducting Boundary-Layer Fluid." *Journal of Aircraft*, Vol. 3, No. 3, 1966, pp. 270–277.

[12] Brandau, J. H., "Performance of Waterjet Propulsion Systems-A Review of the State-of-the-Art," *Journal of Hydronautics*, Vol. 2, No. 2, 1968, pp. 61–73.

[13] Hardin, L., Tillman, G., Sharma, O., Berton, J., and Arend, D., "Aircraft System Study of Boundary Layer Ingesting Propulsion," *48th AIAA/ASME/SAE/ASEE Joint Propulsion Conference & Exhibit*, Joint Propulsion Conferences, American Institute of Aeronautics and Astronautics, 2012. https://doi.org/10.2514/6.2012-3993, URL https://doi.org/10.2514/6.2012-3993.

[14] Vitale, S., Pini, M., and Colonna, P., "Multistage Turbomachinery Design Using the Discrete Adjoint Method Within the Open-Source Software SU2," *Journal of Propulsion and Power*, Vol. 36, No. 3, 2020, pp. 465–478.

[15] Menter, F., "Zonal Two Equation Kw Turbulence Models for Aerodynamic Flows," *23rd Fluid Dynamics, Plasmadynamics, and Lasers Conference*, 1993, p. 2906.

[16] Samareh, J., "Aerodynamic Shape Optimization Based on Free-Form Deformation," *10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, Multidisciplinary Analysis Optimization Conferences, American Institute of Aeronautics and Astronautics, 2004. https://doi.org/10.2514/6.2004-4630, URL https://doi.org/10.2514/6.2004-4630.

[17] Pablo Garrido de la Serna, "Adjoint-based 3D Shape Optimization for Turbomachinery Applications," 2019. URL http://resolver.tudelft.nl/uuid:d8d5dfcf-a5ac-4b54-bac5-9f78fb15f786, delft University of Technology.

[18] Agromayor, R., and Anand, N., "ParaBlade documentation," , 2019.

[19] Sun, L., Yao, W., Robinson, T., Marques, S., and Armstrong, C., "A Framework of gradient-based shape optimization using feature-based CAD parameterization," *AIAA Scitech 2020 Forum*, AIAA SciTech Forum, American Institute of Aeronautics and Astronautics, 2020. https://doi.org/10.2514/6.2020-0889, URL https://doi.org/10.2514/6.2020-0889.

[20] Reuther, J., *Aerodynamic shape optimization using control theory*, 1996. URL https://ntrs.nasa.gov/citations/19960029105, iD: 19960029105.

[21] Jameson, A., "Optimum aerodynamic design using CFD and control theory," *12th Computational Fluid Dynamics Conference*, Fluid Dynamics and Co-located Conferences, American Institute of Aeronautics and Astronautics, 1995. https://doi.org/10.2514/6.1995-1729, URL https://doi.org/10.2514/6.1995-1729.

[22] Wu, H.-Y., Yang, S., Liu, F., and Tsai, H.-M., "Comparisons of Three Geometric Representations of Airfoils for Aerodynamic Optimization," *16th AIAA Computational Fluid Dynamics Conference*, Fluid Dynamics and Co-located Conferences, American Institute of Aeronautics and Astronautics, 2003. https://doi.org/10.2514/6.2003-4095, URL https://doi.org/10.2514/6.2003-4095.

[23] Siddappaji, K., Turner, M. G., and Merchant, A., "General Capability of Parametric 3D Blade Design Tool for Turbomachinery," *GT2012*, Volume 8: Turbomachinery, Parts A, B, and C, 2012, pp. 2331–2344. https://doi.org/10.1115/GT2012-69756, URL https://doi.org/10.1115/GT2012-69756.

[24] Aman uz zaman Baig, "Adjoint Design Optimization for Boundary Layer Ingesting Inlet Guide Vanes with Distorted Inlet Profiles in SU2," 2020. University of Cincinnati.

[25] Akima, H., "A new method of interpolation and smooth curve fitting based on local procedures," Vol. 17, No. 4, 1970, pp. 589–602.

[26] Fedorov, "Numerics Interpolation Chapter," , 2010. URL https://phys.au.dk/~fedorov/Numeric/15/Book/interpolation.pdf.

[27] SPALART, P., and ALLMARAS, S., "A one-equation turbulence model for aerodynamic flows," *30th Aerospace Sciences Meeting and Exhibit*, Aerospace Sciences Meetings, American Institute of Aeronautics and Astronautics, 1992. https://doi.org/10.2514/6.1992-439, URL https://doi.org/10.2514/6.1992-439.

[28] "Case 7-28-8 IGV files," , 2020. URL https://github.com/GTSL-UC/T-Blade3/tree/master/inputs/OpenCSM/new_inputs/Case7-28-8/IGV.