Optimization of Turbine Engine Cycle Analysis with Analytic Derivatives

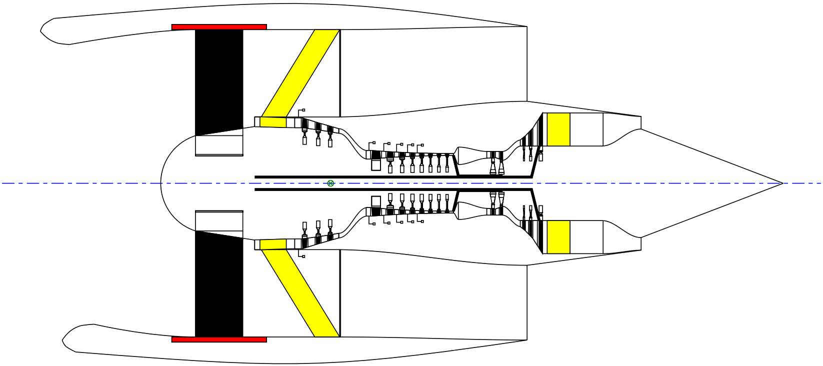
Tristan Hearn, Eric Hendricks, Jeffrey Chin, Justin Gray, Kenneth T. Moore

NASA Glenn Research Center, Cleveland, OH

June 16th, 2016

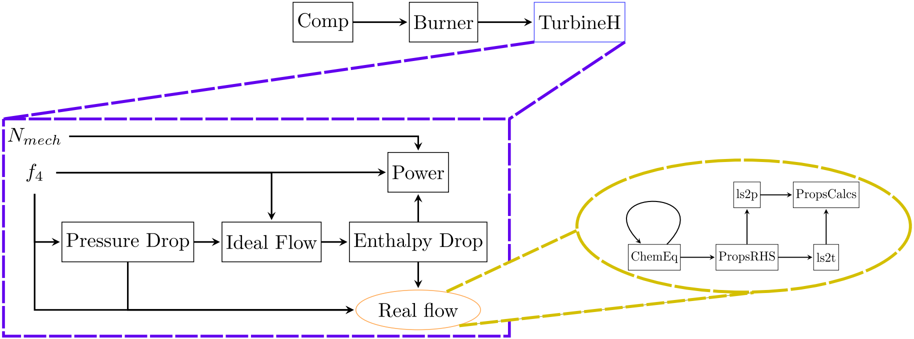
Faster engine cycle optimization

- Optimization of a separate flow turbofan design was performed with analytic derivatives using the cycle analysis code Pycycle
- Computation cost on average was 1/3 that of an optimization performed on an NPSS implementation, with finite-difference derivatives



Pycycle Overview

Pycycle is a 1D cycle modeling tool similar to NPSS, but with an extra level of decomposition[1]

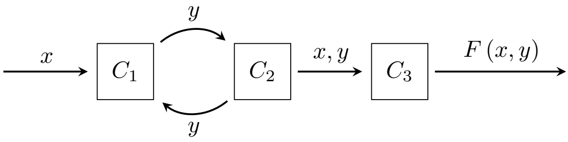


This allows for the implementation of analytic derivatives

[1] Justin S. Gray et al. "Thermodynamics For Gas Turbine Cycles With Analytic Derivatives in OpenMDAO". . In: *2016 AIAA SciTech Conference*. American Institute of Aeronautics and Astronautics, Jan. 2016.

Analytic derivatives within OpenMDAO

OpenMDAO computes coupled derivatives for complex multidisciplinary models automatically



Forward:

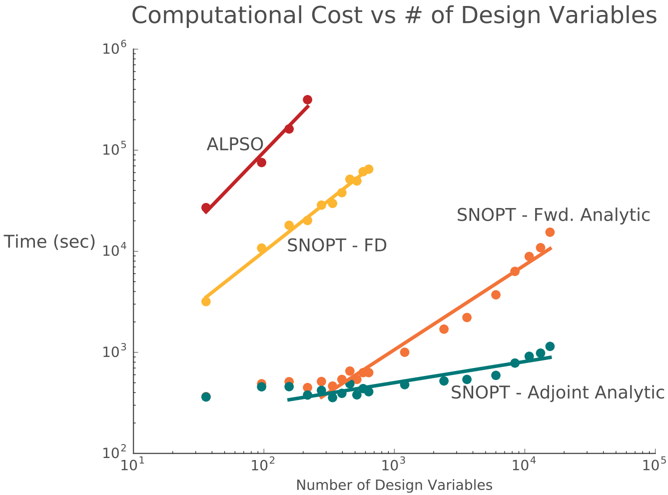
$$\underbrace{\frac{d\mathbf{F}}{d\mathbf{x}_i}}_{m\times 1} = \underbrace{\frac{\partial \mathbf{F}}{\partial \mathbf{x}_i}}_{m\times 1} - \underbrace{\frac{\partial \mathbf{F}}{\partial \mathbf{y}}}_{m\times n} \underbrace{\left(\frac{\partial \mathbf{R}}{\partial \mathbf{y}}\right)^{-1} \frac{\partial \mathbf{R}}{\partial \mathbf{x}_i}}_{n\times 1} \tag{1}$$

Adjoint:

$$\underbrace{\frac{d\mathbf{F}_i}{d\mathbf{x}}}_{1\times k} = \underbrace{\frac{\partial \mathbf{F}_i}{\partial \mathbf{x}}}_{1\times k} - \underbrace{\left(\left(\frac{\partial \mathbf{R}}{\partial \mathbf{y}}^T\right)^{-1} \frac{\partial \mathbf{F}_i}{\partial \mathbf{y}}^T\right)^T}_{1\times n} \underbrace{\frac{\partial \mathbf{R}}{\partial \mathbf{x}}}_{n\times k}, \tag{2}$$

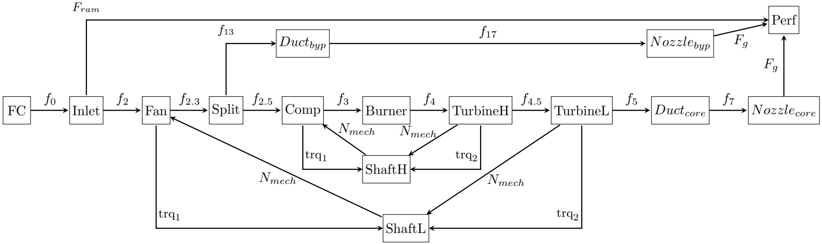
Analytic derivative benefits

Analytic derivatives provide significant computational savings for gradient based optimization



Turbofan model structure

A separate flow turbofan model was built in both Pycycle and NPSS and optimized in OpenMDAO



$$
\begin{array}{lccl}
\text{Minimize:} & & & \\
 & & \text{TSFC} & \\
\text{With respect to:} & & & \\
 & 1 \leq & \text{FPR} & \leq 2 \\
 & 1 \leq & \text{CPR} & \leq 30 \\
 & 1 \leq & \text{BPR} & \leq 12 \\
 & 1 \leq & \text{W} & \leq 2000 \ \frac{\mathrm{lbm}}{\mathrm{s}} \\
\text{Such That:} & & & \\
 & & \mathrm{OPR} & = 30 \\
 & & F_n & = 25,000 \ \mathrm{lbf} \\
 & & T_4 & \leq 3000^\circ \ \text{R}
\end{array}
$$

Flight condition: 35,000 ft, 0.8 MN

Resulting designs

Pycycle and NPSS based optimizations drove towards the same answer

	Baseline	Optimized (Pycycle)	Optimized (NPSS)
FPR	1.5	2.0	2.0
CPR	10.3	15.0	15.0
BPR	5.0	12.0	12.0
W	500.0	1069.2	1032.40
TSFC	0.612	0.331	0.320

Mass flow and TSFC vary between codes due to a thermodynamic discrepancy

Tolerances obtained

- Both internal solver tolerances were set to 10^{-5}
- Pycycle converged to much tighter tolerances overall

	Pycycle	NPSS
Max. constraint violation	$3.5 \cdot 10^{-15}$	$1.2 \cdot 10^{-3}$
$\text{ShaftL}_{\text{net pwr.}}$	$1.64 \cdot 10^{-6}$	-0.022
$\text{ShaftH}_{\text{net pwr.}}$	$6.11 \cdot 10^{-8}$	$2.826 \cdot 10^{-6}$

Optimization performance metrics

Analytic Derivatives give fewer iterations and lower wall time on average

	Pycycle	NPSS				
FD step size	-	10^{-5}	10^{-4}	$0.99 \cdot 10^{-3}$	10^{-3}	$1.01 \cdot 10^{-3}$
SNOPT iterations	44	120	58	721	11	98
Run time (s)	3753	30912	12796	131581	1071	18788

- NPSS optimizations were highly sensitive to step size
- Difference in compute cost is primarily due to the difference in the cost of computing derivatives
- Tight tolerance requires more iterations for each FD step

Conclusions

- Results suggest analytic derivatives are suitable for optimization of engine cycle analysis
- Optimizations performed using engine cycle analysis outperform analyses performed using finite-difference derivatives
- Access to analytic adjoint derivatives will enable more ambitious MDO problems (propulsion-airframe, propulsion-mission, etc.)

Acknowledgments

- TAC Transformational Tools and Technologies Project
- Thomas Lavelle, NASA GRC
- Christopher Snyder, NASA GRC