General Aviation Synthesis with Python

An Optimization-Based Hybrid Propulsion Aircraft Design Software

Kenneth R. Lyons

- GASPy Development Team
  - Jennifer Bergeson (GRC)
  - Sydney Schnulo (prev. GRC)
  - Eliot Aretskin-Harriton (GRC)
  - Benjamin Margolis (ARC)
- OpenMDAO Development Team (GRC)
- Advisors
  - Jeff Bowles (ARC)
  - Justin Gray (GRC)
  - Rob Falck (GRC)

Outline

Outline

Motivation

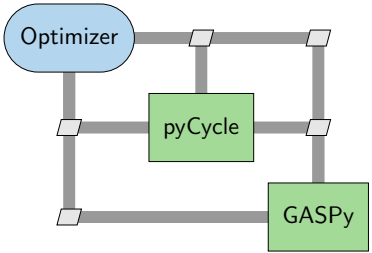
GASP: General Aviation Synthesis Program

- Modernize GASP to meet challenges of tightly-coupled concepts
- Facilitate integration with existing tools to improve subsystem models (e.g. pyCycle, VSPAERO)
- Generalize approach to vehicle analysis and design
- Facilitate knowledge transfer

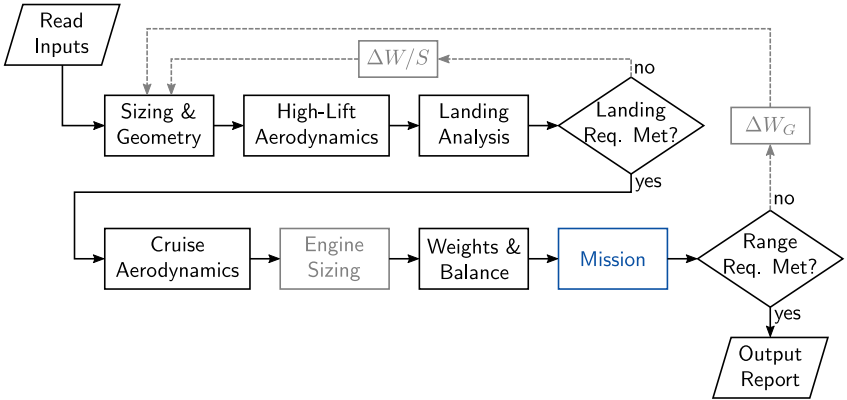
GASPy

Goal: Optimization-Based Design

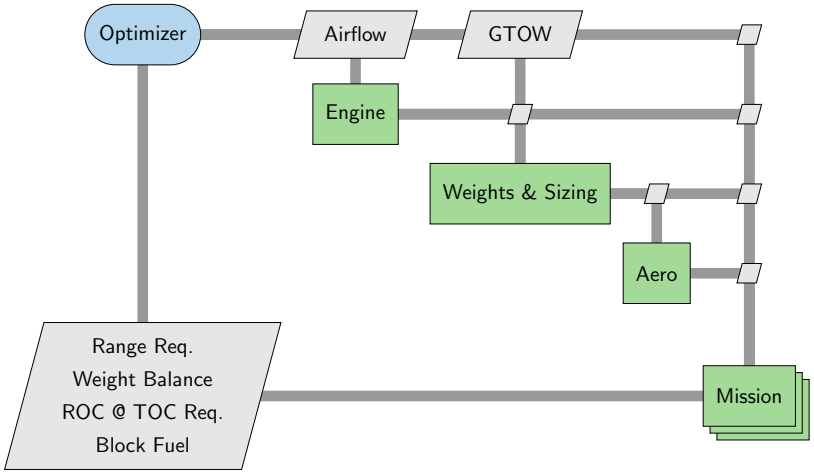
- Checkpoint: Verification
  - Port GASP submodules (weights and sizing, propulsion, aero)
  - Reimplement GASP mission analysis
  - Replicate GASP results with a reference vehicle
- Milestone: New capability
  - Integrate pyCycle-based engine
  - Coupled engine-airframe optimization with electrified engine
- Stretch goal: Optimized electrification



GASP Program Flow

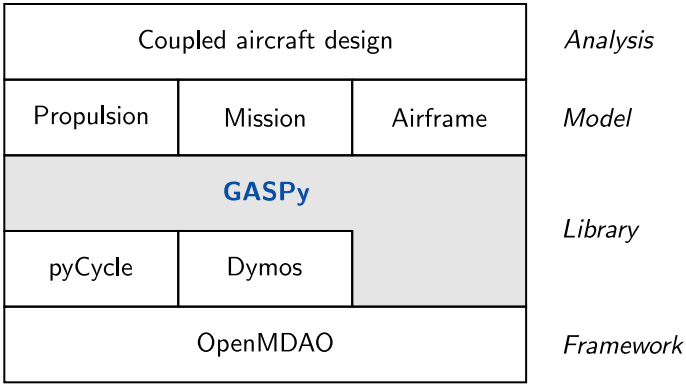


Optimization-Based Approach



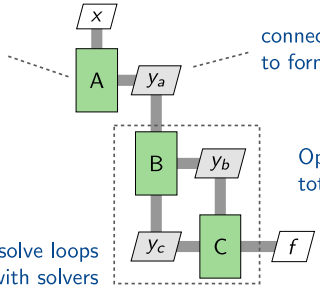
Outline

Software Stack



Open Multidisciplinary Analysis and Optimization

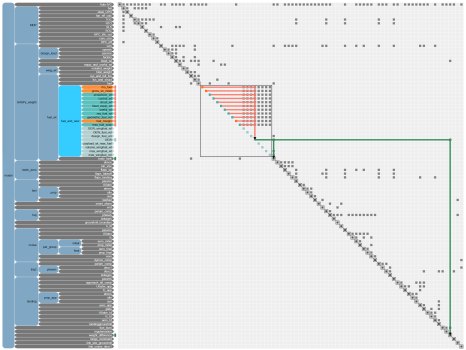
write modular components, optionally with derivatives e.g. $\frac{\partial y_a}{\partial x}$



connect components to form a model

OpenMDAO assembles total derivatives $\frac{df}{dx}$

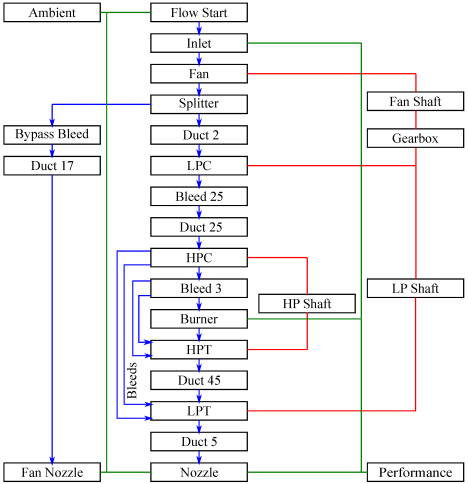
resolve loops with solvers



Gray et al., "OpenMDAO: an open-source framework for multidisciplinary design, analysis, and optimization," 2019.

Hwang and Martins, "A Computational Architecture for Coupling Heterogeneous Numerical Models and Computing Coupled Derivatives,", 2018.

pyCycle

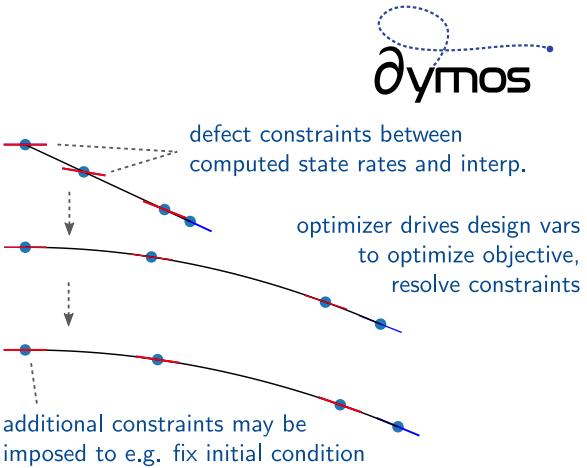


Think *NPSS but with analytic derivatives*

- Cycle analysis tool for constructing an engine model
- Analytic derivatives enable efficient optimization
- Complex design requirements handled through multi-design point (MDP) analysis

Hendricks and Gray, "pyCycle: A Tool for Efficient Optimization of Gas Turbine Engine Cycles," 2019.

- Trajectory optimization framework supporting implicit and explicit methods
  - GASPy currently using collocation exclusively
- ODE evaluated only at the collocation nodes
- Ideal for optimizing without a controller model



Falck et al., "dymos: A Python package for optimal control of multidisciplinary systems," 2021.
Paris, Riehl, and Sjauw, "Enhanced Procedures for Direct Trajectory Optimization Using Nonlinear Programming and Implicit Integration," 2006.

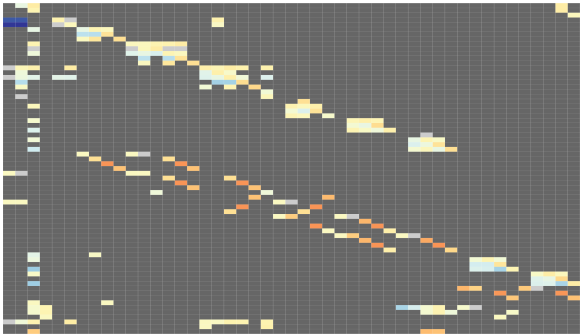
SNOPT

Sparse Nonlinear Optimizer

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \; f(x)$$

$$\text{subject to } l \leq \begin{pmatrix} x \\ c(x) \\ Ax \end{pmatrix} \leq u$$

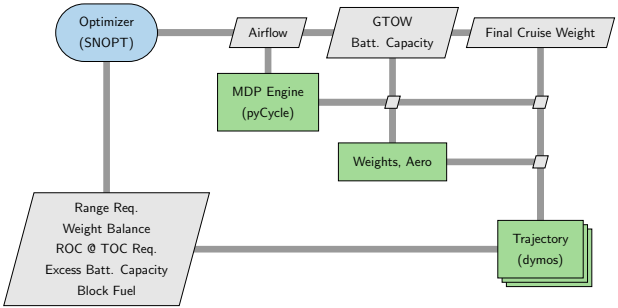
- Sequential quadratic programming (SQP)
- Can make use of (user-provided) exact derivatives
- Suited to *sparse* problems



Gill, Murray, and Saunders, "SNOPT: An SQP Algorithm for Large-Scale Constrained Optimization," 2005.

Model Construction

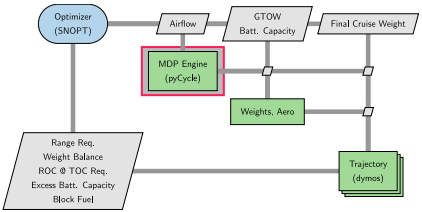
Putting it all together

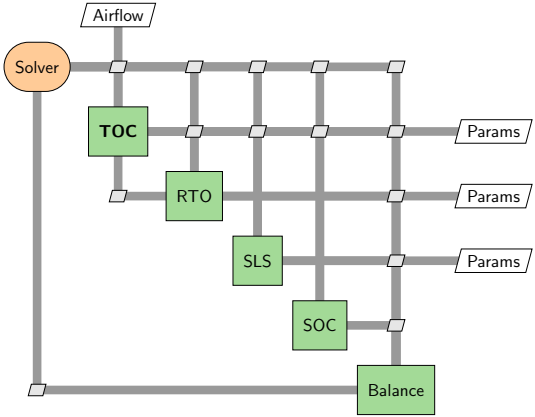


- Electrified pyCycle engine
- GASP-based weights and sizing, aero
- GASP-based mission implemented with dymos

Model Construction: MDP Engine

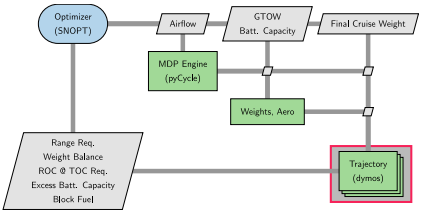
- Design points: top of climb, rolling takeoff, sea-level static, start-of-cruise

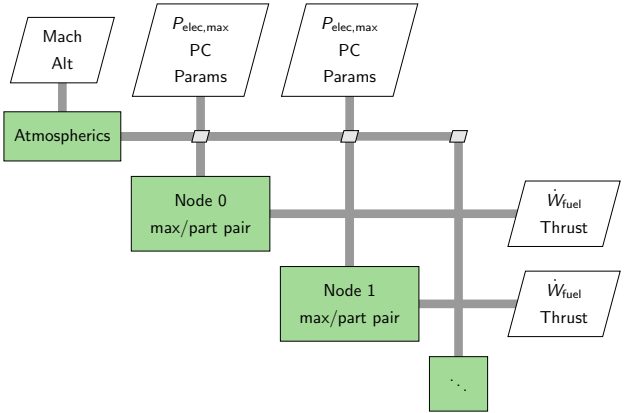




Model Construction: Trajectory

- Each node in the trajectory runs a pair of pyCycle models





Outline

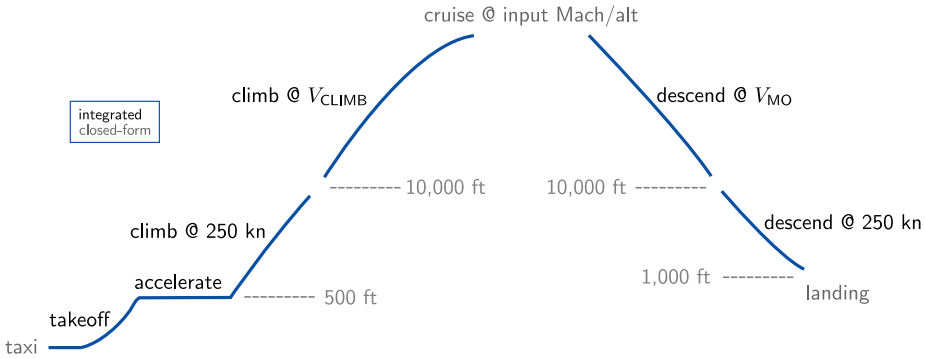
Reference Vehicle

- GASPy built up from GASP 737 MAX 8 inputs
- I/O from MAX 8 case used as a basis for testing

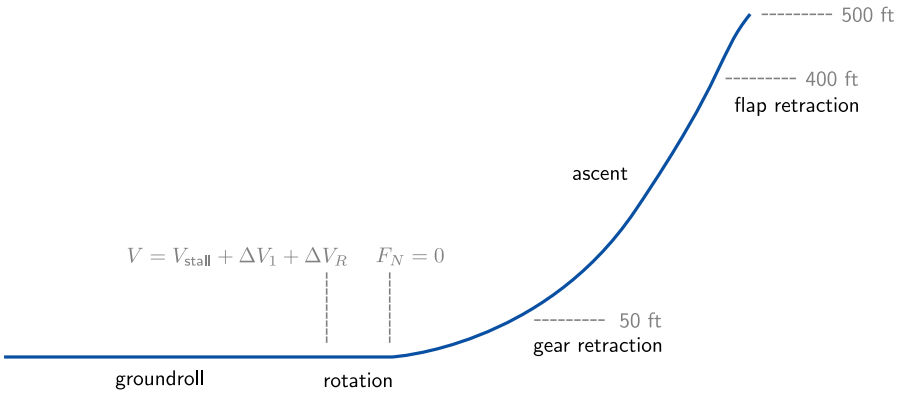
pjs2005 from Hampshire, UK, rotated by the uploader, CC BY-SA 2.0, via Wikimedia Commons

GTOW	175,400 lb
Cruise Mach	0.8
Cruise alt	37,500 ft
Payload	36,000 lb
SLS thrust	28,690 lbf

GASP Design Mission



GASP Design Mission: Takeoff



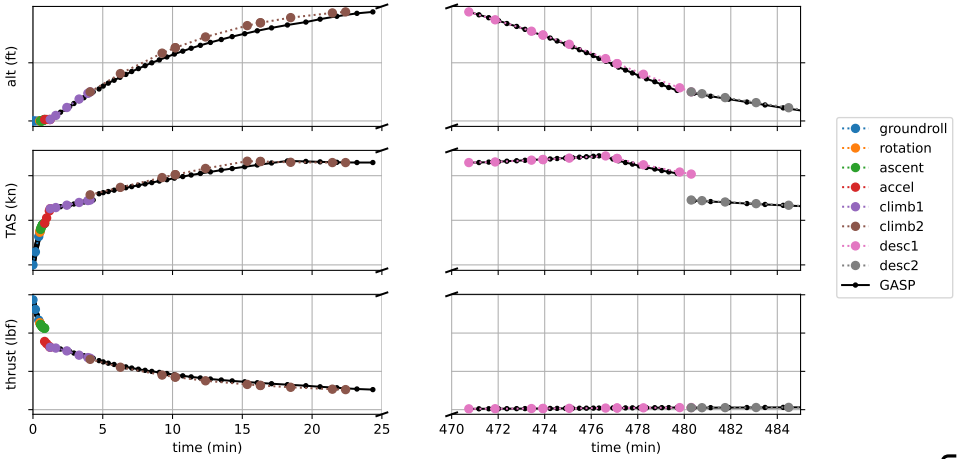
Outline

Reference Case Results

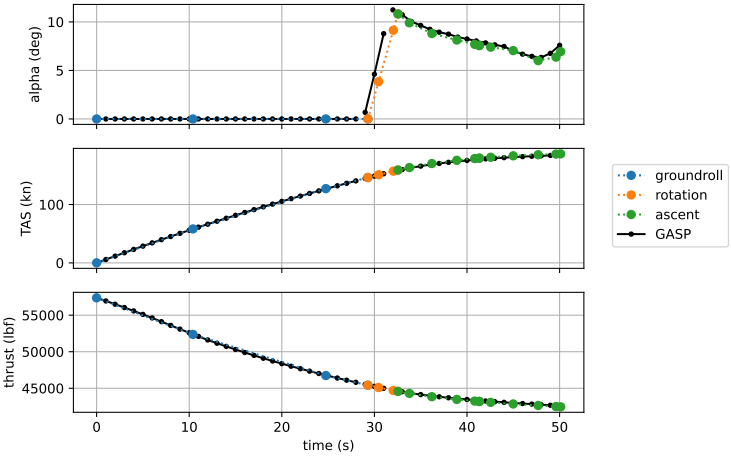
- "Vehicle closure"
  - Variable GTOW
  - Fixed engine size
  - Match GASP range

	GASP	GASPy	error
GTOW (lb)	175,400	174,562	-0.48%
OEW (lb)	96,348	95,805	-0.56%
Block fuel (lb)	43,050	42,756	-0.68%
Flight time (hr)	8.128	8.110	-0.22%
Cruise L/D	18.61	18.58	-0.16%

Reference Case Results: Whole Trajectory



Reference Case Results: Takeoff



Engine Deck Sizing

- Variable SLS airflow $\left(F(\mathrm{Ma}, h, T_4) = \frac{\dot{m}_{\mathrm{SLS}}}{\dot{m}_{\mathrm{SLS,ref}}} F_{\mathrm{ref}}(\mathrm{Ma}, h, T_4)\right)$
- Rate-of-climb at top-of-climb constrained to ≥ 300 fpm
- Total range always 3,675 nmi

	GASP	vary GTOW	vary airflow	vary both
GTOW (lb)	175,400	174,562	175,400	173,977
SLS thrust (lbf)	28,690	28,690	29,435	28,162
ROC @ TOC (fpm)	339.9	341.6	400.0	300.0*
Cruise TSFC (lb/lbf/h)	0.5487	0.5478	0.5470	0.5485
Cruise range (nmi)	3,417	3,429	3,437	3,421
Cruise fuel (lb)	33,866	33,969	34,225	33,785
Block fuel (lb)	43,050	42,756	42,936	42,645

* Active constraint

Engine Deck Sizing

- Variable SLS airflow $\left(F(\mathrm{Ma}, h, T_4) = \frac{\dot{m}_{\mathrm{SLS}}}{\dot{m}_{\mathrm{SLS,ref}}} F_{\mathrm{ref}}(\mathrm{Ma}, h, T_4)\right)$
- Rate-of-climb at top-of-climb constrained to ≥ 300 fpm
- Total range always 3,675 nmi

	GASP	vary GTOW	vary airflow	vary both
GTOW (lb)	175,400	174,562	175,400	173,977
SLS thrust (lbf)	28,690	28,690	29,435	28,162
ROC @ TOC (fpm)	339.9	341.6	400.0	300.0*
Cruise TSFC (lb/lbf/h)	0.5487	0.5478	0.5470	0.5485
Cruise range (nmi)	3,417	3,429	3,437	3,421
Cruise fuel (lb)	33,866	33,969	34,225	33,785
Block fuel (lb)	43,050	42,756	42,936	42,645

* Active constraint

Engine Deck Sizing

- Variable SLS airflow $\left(F(\mathrm{Ma}, h, T_4) = \frac{\dot{m}_{\mathrm{SLS}}}{\dot{m}_{\mathrm{SLS,ref}}} F_{\mathrm{ref}}(\mathrm{Ma}, h, T_4)\right)$
- Rate-of-climb at top-of-climb constrained to ≥ 300 fpm
- Total range always 3,675 nmi

	GASP	vary GTOW	vary airflow	vary both
GTOW (lb)	175,400	174,562	175,400	173,977
SLS thrust (lbf)	28,690	28,690	29,435	28,162
ROC @ TOC (fpm)	339.9	341.6	400.0	300.0*
Cruise TSFC (lb/lbf/h)	0.5487	0.5478	0.5470	0.5485
Cruise range (nmi)	3,417	3,429	3,437	3,421
Cruise fuel (lb)	33,866	33,969	34,225	33,785
Block fuel (lb)	43,050	42,756	42,936	42,645

* Active constraint

Electrification with pyCycle

- Assist added as constant power applied to LP shaft
- Constant power for a given phase: $P_{\text{part}} = P_{\text{max}} \left(\frac{\text{PC}}{30} - \frac{2}{3} \right)$
- Preliminary automatic battery sizing with augmentation system weight estimation

Phase	PC	P_{max} (hp)	P_{part} (hp)
Taxi	21	0	0
Takeoff	50	1,341	1,341
Accel	47	1,341	1,206.9
Climb	47	1,341	1,206.9
Cruise	$\sim$	0	0
Descent	21	0	0
Landing	21	0	0

Cruise PC determined by solver for $T = D$

pyCycle Engine Sizing and Optimization Progress

- Variable GTOW, fixed MDP engine
  - pyCycle enabled for all phases *except* descent
  - Run on Skylake node with 42 processes took $\sim$ 25 minutes

pyCycle Engine Sizing and Optimization Progress

- Variable GTOW, fixed MDP engine
  - pyCycle enabled for all phases *except* descent
  - Run on Skylake node with 42 processes took $\sim$ 25 minutes
- Fixed GTOW, sized MDP engine (variable $\dot{m}_{\text{TOC}}$)
  - pyCycle enabled for second climb & cruise only
  - Engine sized up dramatically (860 fpm @ TOC)
  - Engine weight model may not be sufficient

pyCycle Engine Sizing and Optimization Progress

- Variable GTOW, fixed MDP engine
  - pyCycle enabled for all phases *except* descent
  - Run on Skylake node with 42 processes took $\sim$ 25 minutes
- Fixed GTOW, sized MDP engine (variable $\dot{m}_{\text{TOC}}$)
  - pyCycle enabled for second climb & cruise only
  - Engine sized up dramatically (860 fpm @ TOC)
  - Engine weight model may not be sufficient
- Variable GTOW, sized MDP engine
  - Still having some convergence issues
  - Enabling pyCycle for second climb segment only is working

GASPy

Future Work

- Performance
  - Improve robustness with and without pyCycle
  - Improve augmentation system integration

Future Work

- Performance
  - Improve robustness with and without pyCycle
  - Improve augmentation system integration
- GASP feature parity
  - Develop simplified user interface
  - Test additional aircraft and configurations
  - Reserve mission, OEI takeoff, other climb modes

GASPy

Future Work

- Performance
  - Improve robustness with and without pyCycle
  - Improve augmentation system integration
- GASP feature parity
  - Develop simplified user interface
  - Test additional aircraft and configurations
  - Reserve mission, OEI takeoff, other climb modes
- Potential future tool integrations
  - Engine weight estimation
  - Higher-fidelity aerodynamics

GASPy