OpenVSP/OpenMDAO Primer

Mark McMillin
OpenVSP Workshop
Pismo Beach
August 30 – September 1, 2017
Dr. Jason Welstead – TTT/LEAPS funding
Dr. Francisco Capristan – OpenMDAO consulting
Dr. Erik Olson – Aerodynamics validation
Jim Fenbert – Java Plugin developer
Darrell (DJ) Caldwell – LEAPS Development Environment
Contents

◆ Introduction/Objective
◆ OpenMDAO Overview
◆ Geometry Examples
◆ Aerodynamics Example – Work in Progress
◆ Observations
◆ Conclusions/Future Work
Introduce OpenMDAO – I am not an expert!

Flight Optimization System (FLOPS) is being replaced by the Layered and Extensible Aircraft Performance System (LEAPS) Python modules

Experiment with integrating LEAPS modules in an OpenMDAO environment
  - This is exploratory work with regards to how LEAPS may utilize one or both of the OpenMDAO and OpenVSP software packages in the future.
  - Work performed will inform the decision process for LEAPS software development

Provide prototype coded building blocks for future developers
  - Examples of OpenVSP and VSPAero using the OpenMDAO Environment
  - Stimulate interest in other groups to use OpenMDAO to automate OpenVSP

Wrapper and example code walk through – time/interest permitting
Where to Start

✨ [http://openmdao.org/openmdao-1-7-1/](http://openmdao.org/openmdao-1-7-1/)
OpenMDAO Overview

- Problem solving environment/framework
- Open source
- Python object oriented code
- Being developed at Glenn Research Center

Hierarchical Structure
- Problems, Groups, Components
- Drivers and Solvers Computations
Types of Components
- External Code - wrapper
- Independent Variable
- Execute Computation (Single equation)

Anatomy of a Wrapper
- Init function
  - Set up input, output and state variables
  - Specify derivatives type (analytic, finite difference)
  - Provide “call” function to external code
- Solve non-linear
  - Write input file
  - Execute external code
  - Parse output
- Solve linear – provide analytic derivatives

Functions of a Group
- Add components, groups, sub-problems
- Link variables (explicit, implicit)
- Specify Nonlinear solver for convergence loops
  - Requires component to calculate residual of state variable
- Custom such as print results
Problems/SubProblems
- Specify root group
- Specify Driver type
  - Run once - default
  - Optimizer and group with design variable and objective
  - Predetermined runs (parameter sweep, DOE)

Basic Solution Steps
- Define Components and Groups and link variables
- Instantiate a Problem with the root Group
- Perform setup on the Problem to initialize all vectors and data transfers
- Perform run on the Problem

Model results viewing
- View connections (N^2 diagram)
- View model – variable results value table
- Print results - custom
- External code to view results files - custom

Advanced Features...
- Parallel/distributed processing
- Data gathering, file passing
- More....
Environment Requirements

- JAVA
- Python e.g. Anaconda
- OpenMDAO
- OpenVSP
- OpenVSP JAVA Plugin
- LEAPS Modules - optional
- Notepad – optional
- Batch file – optional
- OpenVSPWrapper
- Your problem
Windows Batch File Used to Capture Process

REM -- VSPJavaInterface:
REM -select .vsp3 file
REM -writes State.xml file (kind of like a .des file but includes outputs)
REM -writes OpenVSP3Plugin.vscript file (anglescript, .as) - not used
java -jar VSP_GeomJavaPlugin.jar

REM -- Provides a file handling tag by parsing base .vsp3 name
REM -- from State.xml and writes it to file "BaseName"
python VSP_parseBaseName.py
set /p BASE=<BaseName

REM -- Need to delete old files if they exists??
REM -- Keep original model clean
copy %BASE%.vsp3 %BASE%_VSP_Geom.vsp3
rename State.xml %BASE%_VSP_State.xml

REM -- Run OMDAO test case
python VSP_GeomExamples.py

REM -- Cleanup files...
del OpenVSP3Plugin.vscript
del OpenVSP3PluginCompGeom.csv
del OpenVSP3PluginCompGeom.txt

REM -- Move files to archives...

♦ VSP_GeomProcess.bat

♦ Executes the JAVA standalone version of OpenVSP plugin developed for ModelCenter™
♦ Parses BaseName from state file for file management
♦ Executes OpenMDAO example problem using OpenVSP wrapper
Geometry Examples

♦ Examples used for testing ideas, learning system capabilities and module verification

♦ Verify file parsing
  - myModel.vsp3 file
  - compGeom.csv file
  - massProps.txt file

♦ Experiment with linking capabilities
  - Implicit (automatic)
  - Explicit (manual)

♦ Learn different solution methods
  - Optimization driver
  - Newton solver
♦ All models use the OpenVSP external code component wrapper with forward differenced finite difference derivatives

♦ Model 1
  • Modify section span (state) to achieve desired section planform area (zero residual between current area and desired area)
  • Uses “promote” feature to auto link and raise variable referencing up a level in model hierarchy
  • Uses Newton solver to converge on the solution
  • Verifies Model.vsp3 file parsing

♦ Model 2
  • Modify section span (state) to achieve desired wing surface area (zero residual between current area and desired area)
  • Uses manual variable linking
  • Uses Newton solver to converge on the solution
  • Verifies compGeom.csv file parsing
Four Geometry Examples

♦ Model 3
- Modify section span to achieve section planform area – same as Model 1
- Uses “promote” feature to auto link
- Uses SLSQP optimizer to find the solution
- Uses the ExecComp component to specify objective function
- Uses IndepVarComp to specify design variable
- Limits design variable range

♦ Model 4
- Modify wing x-location to achieve vehicle x-c.g.
- Uses auto variable linking
- Uses Newton solver to converge on the solution
- Verifies MassProps.txt file parsing
Geometry Examples – Print Results

Setup: Checking root problem for potential issues...
No recorders have been specified, so no data will be saved.
The following parameters have no associated unknowns:
residualComp.DesiredHeight
Group '' has the following cycles: [["geometryComp", 'residual']]
The following params are connected to unknowns that are unused]
Setup: Check of root problem complete.

WingK = 14.166028, cgK = 16.999827

Setup: Checking root problem for potential issues...
No recorders have been specified, so no data will be saved.
The following parameters have no associated unknowns:
residualComp.DesiredHeight
Group '' has the following cycles: [["geometryComp", 'residual']]
The following params are connected to unknowns that are unused]
Setup: Check of root problem complete.

Span = 7.636537, Area = 40.000000

Setup: Checking root problem for potential issues...
No recorders have been specified, so no data will be saved.
The following parameters have no associated unknowns:
residualComp.DesiredHeight
Group '' has the following cycles: [["geometryComp", 'residual']]
The following params are connected to unknowns that are unused]
Setup: Check of root problem complete.

Optimization terminated successfully.  (Exit mode 0)
Current function value: 3.07999997e-07
Iterations: 6
Function evaluations: 19
Gradient evaluations: 6
Optimization Complete

Span = 16.000000, Area = 40.000000

9/7/17
McMillin
Geometry Examples – Geometry Model

Baseline Geometry

Wing Placement Found to Achieve Desired C.G.

Wing Span Found to Achieve Desired Section Area

Wing Span Found to Achieve Desired surface Area
# Geometry Examples – View Connections

<table>
<thead>
<tr>
<th>Problem Root Group</th>
<th>Geometry Component</th>
<th>Residual Component</th>
<th>Desired Component</th>
<th>X_Location</th>
<th>X_CG</th>
<th>Desired_CG</th>
<th>X_Location</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Input</td>
<td></td>
<td></td>
<td>Link</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Solver Controlled</td>
<td></td>
<td></td>
<td>State</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Feedback</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **Geometry Component**: X_Location, X_CG
- **Residual Component**: Desired_CG
- **Problem Root Group**: X_Location, X_CG
### Geometry Examples – View Model

<table>
<thead>
<tr>
<th>Source</th>
<th>Units</th>
<th>Value</th>
<th>Units</th>
<th>Target</th>
</tr>
</thead>
<tbody>
<tr>
<td>residualComp.myWing__XSec_1__Span</td>
<td></td>
<td>16.0</td>
<td></td>
<td>geometryComp.myWing__XSec_1__Span</td>
</tr>
<tr>
<td>geometryComp.myWing__XSec_1__Area</td>
<td></td>
<td>40.0</td>
<td></td>
<td>residualComp.myWing__XSec_1__Area</td>
</tr>
<tr>
<td>NO CONNECTION</td>
<td></td>
<td>40.0</td>
<td></td>
<td>residualComp.DesiredArea</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Source</th>
<th>Units</th>
<th>Value</th>
<th>Units</th>
<th>Target</th>
</tr>
</thead>
<tbody>
<tr>
<td>geometryComp.myWing__XSec_1__Area</td>
<td></td>
<td>40.0000000308</td>
<td></td>
<td>objectiveComp.myWing__XSec_1__Area</td>
</tr>
<tr>
<td>designVarComp.myWing__XSec_1__Span</td>
<td></td>
<td>16.0000001232</td>
<td></td>
<td>geometryComp.myWing__XSec_1__Span</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Source</th>
<th>Units</th>
<th>Value</th>
<th>Units</th>
<th>Target</th>
</tr>
</thead>
<tbody>
<tr>
<td>residualComp.myWing__XForm__X_Location</td>
<td></td>
<td>14.1660277778</td>
<td></td>
<td>geometryComp.myWing__XForm__X_Location</td>
</tr>
<tr>
<td>geometryComp.MassProperties__Totals__cgX</td>
<td></td>
<td>16.999827</td>
<td></td>
<td>residualComp.MassProperties__Totals__cgX</td>
</tr>
<tr>
<td>NO CONNECTION</td>
<td></td>
<td>17.0</td>
<td></td>
<td>residualComp.DesiredCG</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Source</th>
<th>Units</th>
<th>Value</th>
<th>Units</th>
<th>Target</th>
</tr>
</thead>
<tbody>
<tr>
<td>residualComp.Span</td>
<td></td>
<td>7.63653673875</td>
<td></td>
<td>geometryComp.myWing__XSec_1__Span</td>
</tr>
<tr>
<td>geometryComp.CompGeom__myWing0__TheoreticalArea</td>
<td></td>
<td>40.0</td>
<td></td>
<td>residualComp.CurrentArea</td>
</tr>
<tr>
<td>NO CONNECTION</td>
<td></td>
<td>40.0</td>
<td></td>
<td>residualComp.DesiredArea</td>
</tr>
</tbody>
</table>
AeroCodeWrapper.py Example – A Work in Progress

- Dummy mission component
- Atmospheric model component
- VSPAERO wrapper component
- This is a non-iterative $N^2$ diagram

### Preliminary results values

<table>
<thead>
<tr>
<th>Source</th>
<th>Units</th>
<th>Value</th>
<th>Units</th>
<th>Target</th>
</tr>
</thead>
<tbody>
<tr>
<td>missionComp.CruiseMach</td>
<td></td>
<td>0.75</td>
<td>atmComp.mach</td>
<td></td>
</tr>
<tr>
<td>missionComp.CruiseMach</td>
<td></td>
<td>0.75</td>
<td>atmComp.mach</td>
<td></td>
</tr>
<tr>
<td>missionComp.CruiseAoA</td>
<td></td>
<td>1.5</td>
<td>atmComp.AoA</td>
<td></td>
</tr>
<tr>
<td>missionComp.CruiseAoA</td>
<td></td>
<td>36000.0</td>
<td>atmComp.all</td>
<td></td>
</tr>
<tr>
<td>atmComp.ReRef</td>
<td></td>
<td>0.002797668716</td>
<td>atmComp.ReRef</td>
<td></td>
</tr>
<tr>
<td>atmComp.ReRef</td>
<td></td>
<td>430.352547189</td>
<td>atmComp.ReRef</td>
<td></td>
</tr>
<tr>
<td>atmComp ReRef</td>
<td></td>
<td>328150204.09</td>
<td>atmComp.ReRef</td>
<td></td>
</tr>
<tr>
<td>NO CONNECTION</td>
<td></td>
<td>0.75</td>
<td>missionComp.mach</td>
<td></td>
</tr>
<tr>
<td>NO CONNECTION</td>
<td></td>
<td>1.5</td>
<td>missionComp.AoA</td>
<td></td>
</tr>
<tr>
<td>NO CONNECTION</td>
<td></td>
<td>36000.0</td>
<td>missionComp.all</td>
<td></td>
</tr>
<tr>
<td>NO CONNECTION</td>
<td></td>
<td>10.0</td>
<td>atmComp.ReRef</td>
<td></td>
</tr>
<tr>
<td>NO CONNECTION</td>
<td></td>
<td>1000</td>
<td>atmComp.Sref</td>
<td></td>
</tr>
</tbody>
</table>

### Preliminary results Cp

- Delta-Cp
  - 0.65421
  - 0.17059
  - 0.04874
  - 0.07391
  - 0.19617
  - 0.31842
  - 2.98401
Automated Parameter Sweeps are useful in understanding the design space for debugging models. This can save significant time in eliminating a poorly posed problem. Example trying to size a wing for lift = weight in a non-convergent design space.
Using previous VSPAERO verification case to ensure wrapper is working properly
Observations

- Use linkage viewer after setup and before run
  - Data dictionary would be good for auto linking
- Initialization and Component/Group execution order matter and is controllable
- Duplicated JAVA write script to provide filename data management, auxiliary file computation timing and set control
- Stackoverflow.com is a valuable resource
- Symmetry - Full model vs half model issues can be confusing
  - Planform vs section area
- Multi-panel wings not handled yet (in plugin but not wrapper)
- Subsurface capability not handled yet (in plugin but not wrapper)
- Verification/unit testing should be built in
- Units and tolerances are a headache
IMHO OpenMDAO is easy, free and capable

Develop multi-discipline models
- Include geometry
- Include aerodynamics
- Include mass models
- Farther out
  - Include mission analysis
  - Include structural analysis

Perform a demonstration
- LEAPS milestone
- Possible design function – size wing for lift=weight at cruise with static margin and tail volume constraints

Generate documentation

Archive code into LEAPS GIT revision control environment