수학 및 엔지니어링 프로그래밍의 새로운 대안

Altair Korea

원종성 연구원
Compose in the Math & Systems suite

Compose

- Math Engine
- Development Tools
- Visualization
- CAE Integration

Activate

- Signal Based
- Physical Based
- Co-Simulation

Embed

- Diagram to Code
- Visual RTOS
- Statecharts
- Interactive HIL
Compose is a high level, matrix-based language as well as an interactive & unified programming environment for all types of math.
solidThinking Compose

IDE
(Integrated Development Environment)

OML
Octave
Matlab

python
SciPy

HW-Integration

Design the Difference with Digital twin
Compose : a multi-language environment

• A complete IDE
  • Edition, Execution and Debugging of scripts
  • Plotting and session management

• Multiple Languages
  • OML, Tcl & Python scripting
  • Bridges with OML

• Tight integration with HyperWorks
  • CAE Readers
  • Function Registration in HyperGraph, HyperStudy, Matrix Browser, OptiStruct, MotionView..

• Unique Business Model
  • Flexibility
  • Affordability
IDE and Octave/Matlab Compatibility
Compose for Edition / Execution / Debugging

• Integrated Development Environment
Octave/Matlab Compatibility

• Same syntax

```matlab
n = length(t);
rand('seed', 2017);
x = A * exp(-b*t) .* sin(omega*t) .* normrnd(1, 0.25, 1, n);
plot(t, x, 'linewidth', 3);
xlabel('time');
ylabel('amplitude');
title('Damped sinusoid test data with noise');
```

• Functionality organized in libraries

• All part of the single product, no separate toolboxes

• 2017.2 release covers more than 500 functions

• Future releases will expand coverage
Octave/Matlab Compatibility
Python Interface
Compose-Python Interface

- **Python scripts in Compose**
  - OML script creates and executes a system command to execute python (outside Compose)
  - The bridge where a main script in OML will call Python scripts

- **Bridge between OML and Python (2-ways)**
  - Call OML from Python
  - Call Python from OML and pass data
  - Access OML CAE readers from Python
  - Use Python extensions in thre OML-Python bridge

```python
from matplotlib.pyplot import figure, show
from numpy import arange, sin, pi

#--- Time

T = arange(0.0, 1.0, 0.01)

fig = figure(1)

ax1 = fig.add_subplot(211)

ax1.plot(T, sin(2*pi*T))

#--- To put grid lines

ax1.grid(True)

#--- To set y-axis limit

ax1.set_ylim([-2, 2])

#--- To label y-axis

ax1.set_ylabel('Amplitude')

#--- To label x-axis

ax1.set_xlabel('Time (sec)')

#--- To give title

fig.suptitle('A sine wave')
```

![Graph of a sine wave]
Bridge between OML and Python

- Utilizing Python Functions by Interfacing with Python

- From OML, a series of commands are available to execute and exchange data with the Python workspace.
  - `evalpythonfile`: Evaluate a Python file
  - `evalpythonscript`: Evaluate a Python script
  - `exporttopython`: Export a variable from OML to Python
  - `getpythonvar`: Retrieve Python variables in OML

- Similarly, commands have been added to Python
  - `EvalOmlFile`: Evaluate an OML file
  - `EvalOmlScript`: Evaluate an OML script
Python Extensions

• Compose includes Python 3.4 and extensions such as numpy, scipy or matplotlib.
• However, there are many other powerful Python libraries (modules) available.

• It is possible to extend Python in Compose with the pip command

  `<Compose install path>\python\python3.4\win64\python.exe -m pip install <package-name>`

  And then import the package in Python command window.
HW - Integration
Interfacing with HyperWorks applications

- Registering OML function enables using Compose OML functions for
  - HyperGraph curves and notes,
  - HyperStudy responses,
  - HyperView responses and MotionView responses
  - HyperMesh Matrix Browser

- OptiStruct responses (DRESP3) can we defined as OML function
Interfacing with HyperWorks applications
CAE Readers & HW integration

• Dedicated utility to read CAE files and create OML commands to extract data
CAE Readers & HW integration

• Use CAE readers to directly extract data and process in Compose
Neuber Plasticity Correction

• **Usage**
  
  • Linear analysis that exceeds the yield stress in small and specific regions
  
  • The surrounding areas will withstand part of the stress

• **Input**

  • Element stress output (.strs file – OPTI output)
Neuber Plasticity Correction

- Output
  - .hwascii file with corrected element stresses to be plotted in HyperView or to be used elsewhere
Why Compose?

• Best of multiple worlds
  • Leverages past investment in Matlab
  • CAE readers from HyperWorks
  • 2017 release supports Python, as well; Julia integration next

• Superior business model
  • Available under HWUs
  • Accessible to everybody at your organization
  • One module for various applications

• Non-disruptive adoption
  • Can co-exist with Matlab
http://learn.solidThinking.com/
Thank you!

support@altair.co.kr

blog.altair.co.kr