FortranCon 2020

Copernicus Spacecraft Trajectory Design and Optimization Program

Jacob Williams
NASA Johnson Space Center
July 2, 2020
What is Copernicus?

• Copernicus is a generalized, interactive, spacecraft trajectory design and optimization application

• Multiple spacecraft and propulsion systems, integrated GUI and 3D graphics, flexible segment & plugin architecture, selectable mission fidelity (simple to complex)

• Extensive range of missions: impulsive/low/high thrust, multi-body, planet centered/inter-planetary, multi-body transfers/trajectories

• Developed at JSC, and available for use by any NASA employee or U.S. government contractor

• Evolutionary and expandable

• Copernicus can be scaled from a single desktop or laptop computer using the GUI, to computer clusters where no user interaction or graphical feedback are required
Copernicus Models

• Low and high fidelity models in the same tool
  • Mission Segments
  • Integrators/Propagators
  • Optimal Control Theory
  • Parameter Optimization
  • Numerical Differentiation
  • Ephemerides
  • Reference Frames

• Finite Burn Engine Models
• Finite Burn Maneuver Models
• Impulsive Maneuvers
• Lambert Targeting
• State Parameterizations
• Maneuver Parameterizations
• Gravity Assists
• Halo Orbits
• Gravity Models
• Interpolation
• 3D Visualization
• Data Output
• Plugin Interface
• Scripting Interface

• Interactive, flexible architecture
• More than one way to design/optimize a mission

Define the models in the GUI

Many classes of problems can be modeled with the segment concept.
Copernicus History

- Copernicus has been continuously developed for nearly 20 years
- 5 major releases (5.0 is one of the most significant updates we have done)
- 2001-2002: University of Texas at Austin (UT)
- First official release: March 2006
- 2007-Present: Primary Development at JSC
Copernicus Usage

- About 190 licenses issued for Copernicus 4.x since 2014
- About 300 licenses issued in all for all versions since 2006
- Used at NASA (JSC, MSFC, GRC, GSFC, JPL, LaRC, KSC, ARC), numerous government contractors, and universities
- Copernicus has become a workhorse tool for crewed and uncrewed spaceflight mission design at JSC
- What are we using it for...
LCROSS Mission
(Lunar Crater Observation and Sensing Satellite)

• Copernicus was used to construct hundreds of optimal Earth-Lunar flyby-to-Lunar impact trajectories including the separation phase from the original LRO trajectory which was bound for Lunar orbit

• Also used post-launch to examine under/over burns en route

• LCROSS and its Centaur stage impacted the Moon on Oct. 9, 2009
Asteroids, Mars & Outer Planet Studies

ISP Reference Mission 12: Mars Sample Return Mission
[Using low thrust engine and optimal control theory]

ISP Reference Mission 8:
Earth/Venus/Venus/Jupiter/Pluto flyby mission

GTOC-4: 32-Asteroid Intercept with Final Rendezvous (10 years)
Lunar Missions

Earth-Moon Free Return

Lunar Mission With Landing and Stage Disposal

Three-Burn Trans-Earth Injection Maneuver Sequence

TEI-1

TEI-2

TEI-3
Three-Body, Halo Orbits, DRO, NRHO, etc.

• Halo Orbits & Weak Stability Boundary/Ballistic Capture
• Artemis I, II, III
• Human Landing System (HLS)
• Near-Rectilinear Halo Orbits (NRHO)
• Deep Space Gateway / PPE / HALO

Direct and Flyby Transfers to Earth-Moon L1 and L2 Libration Points

Ballistic Transfers from Earth to NRHO

Transfer from NRHO to high lunar orbit

15-Year 9:2 Resonant NRHO
Copernicus Software Development

• Copernicus started in 2001 as Fortran 77/90, Compaq Visual Fortran (Windows only)
• Transitioned to Intel Fortran circa 2007
• Cross platform: Windows, macOS, Linux
• Continuous improvement & modernization, keep up with the latest Fortran standards and tools
  • If Intel supports a feature, we will use it.
  • Copernicus is never finished.
• Some of the tools/libraries we are currently using: Intel Fortran 2019, CMake 3.12.3, Git, Python 3.7, Anaconda 2019.07, MS Visual Studio 2017, VS Code, Qt 5.12.3, OpenSceneGraph 3.6.2, HDF5 1.10.4
Software Architecture

- Formerly, the entire program was Fortran (used the Winteracter library for the GUI)
- Significant refactoring as Fortran 2003+ became available
  - Mostly standard Fortran with some Intel extensions and MKL routines
  - 327 modules, about 218,505 lines of code (not counting 3rd party F77)
- Copernicus is now (v5.0) implemented as a shared library that is called from a Python GUI
  - Core Copernicus code (Fortran) and the GUI (Python) are now completely decoupled
  - Extensive use of C Interoperability – Callbacks to/from Fortran & Python
  - The Copernicus shared library can also be used by other scripts, tools, etc.
- Interactive 3D graphics using OpenSceneGraph and OpenFrames libraries – Fortran interface to C++ code
- Dynamic equations/functions/models user-input:
  - Custom internal function parser written in Fortran
  - User created shared library plugins (DLLs)
  - Eventually: Python code executed by callback.
Copernicus GUI: Example

Interactive graphics: Iterate, pan, zoom, rotate

User-Selectable Themes, Customizable GUI Configuration

Widgets

Embedded Python console
3D Party Fortran Components

- **SNOPT** (optimization)
  - Fortran 77, Waiting for “modern” version
- **SPICELIB** (JPL, solar system ephemeris, geometry, and time)
  - Fortran 77, Recently announced rewriting it in C++
- **JSON-Fortran** (configuration files, data output, data exchange)
- **Bspline-Fortran** (interpolation)
- **SLSQP** (optimization)
- **Hairer** (ODE IVP)
- **FLINT** (ODE IVP)
- And more!

Open source
Conclusions

• Copernicus is an example of an actively developed modern Fortran application with a wide user base at NASA
• Critical software tool for JSC & NASA-wide
• Greatly expanded capabilities and use cases with recent (v5.0) Python GUI and scripting integration

• Copernicus Fortran wish list
  • Better ecosystem & cross-platform tooling, linting, etc.
  • Generic programming, differentiable programming
  • Exception handling
  • Built-in modern string class
  • Dynamic, interactive capability (think Python, Julia, Jupyter)
References


• J. Williams, J. S. Senent, C. Ocampo, R. Mathur, E. C. Davis. “Overview and Software Architecture of the Copernicus Trajectory Design and Optimization System”, 4th International Conference on Astrodynamics Tools and Techniques, 3-6 May 2010, Madrid, Spain


Questions?