Automated ISS Flight Utilities

Jan Tuzlić Offermann

NASA Johnson Space Center

During my internship at NASA Johnson Space Center, I worked in the Space Radiation Analysis Group (SRAG), where I was tasked with a number of projects focused on the automation of tasks and activities related to the operation of the International Space Station (ISS). As I worked on a number of projects, I have written short sections below to give a description for each, followed by more general remarks on the internship experience.

My first project is titled “General Exposure Representation EVADOSE”, also known as “GEnEVADOSE”. This project involved the design and development of a C++/ ROOT framework focused on radiation exposure for extravehicular activity (EVA) planning for the ISS. The utility helps mission managers plan EVAs by displaying information on the cumulative radiation doses that crew will receive during an EVA as a function of the egress time and duration of the activity. SRAG uses a utility called EVADOSE, employing a model of the space radiation environment in low Earth orbit to predict these doses, as while outside the ISS the astronauts will have less shielding from charged particles such as electrons and protons. However, EVADOSE output is cumbersome to work with, and prior to GEnEVADOSE, querying data and producing graphs of ISS trajectories and cumulative doses versus egress time required manual work in Microsoft Excel. GEnEVADOSE automates all this work, reading in EVADOSE output file(s) along with a plaintext file input by the user providing input parameters. GEnEVADOSE will output a text file containing all the necessary dosimetry for each proposed EVA egress time, for each specified EVADOSE file. It also plots cumulative dose versus egress time and the ISS trajectory, and displays all of this information in an auto-generated presentation made in LaTeX. New features have also been added, such as best-case scenarios (egress times corresponding to the least dose), interpolated curves for trajectories, and the ability to query any time in the EVADES output. As mentioned above, GEnEVADOSE makes extensive use of ROOT version 6, the data analysis framework developed at the European Organization for Nuclear Research (CERN), and the code is written to the C++11 standard (as are the other projects).

My second project is the Automated Mission Reference Exposure Utility (AMREU). Unlike GEnEVADOSE, AMREU is a combination of three frameworks written in both Python and C++, also making use of ROOT (and PyROOT). Run as a

combination of daily and weekly cron jobs, these macros query the SRAG database system to determine the active ISS missions, and query minute-by-minute radiation dose information from ISS-TEPC (Tissue Equivalent Proportional Counter), one of the radiation detectors onboard the ISS. Using this information, AMREU creates a corrected data set of daily radiation doses, addressing situations where TEPC may be offline or locked up by correcting doses for days with less than 95% live time (the total amount time the instrument acquires data) by averaging the past 7 days. As not all errors may be automatically detectable, AMREU also allows for manual corrections, checking an updated plaintext file each time it runs. With the corrected data, AMREU generates cumulative dose plots for each mission, and uses a Python script to generate a flight note file (.docx format) containing these plots, as well as information sections to be filled in and modified by the space weather environment officers with information specific to the week. AMREU is set up to run without requiring any user input, and it automatically archives old flight notes and information files for missions that are no longer active.

My other projects involve cleaning up a large data set from the Charged Particle Directional Spectrometer (CPDS), joining together many different data sets in order to clean up information in SRAG SQL databases, and developing other automated utilities for displaying information on active solar regions, that may be used by the space weather environment officers to monitor solar activity.

I consulted my mentor Dr. Ryan Rios and Dr. Kerry Lee for project requirements and added features, and ROOT developer Edmond Offermann for advice on using the ROOT library. I also received advice and feedback from Dr. Janet Barzilla of SRAG, who tested my code. Besides these inputs, I worked independently, writing all of the code by myself. The code for all these projects is documented throughout, and I have attempted to write it in a modular format. Assuming that ROOT is updated accordingly, these codes are also Y2038-compliant (and Y10K-compliant). This allows the code to be easily referenced, modified and possibly repurposed for non-ISS missions in the future, should the necessary inputs exist.

These projects have taught me a lot about coding and software design – I have become a much more skilled C++ programmer and ROOT user, and I also learned to code in Python and PyROOT (and its advantages and disadvantages compared to C++/

ROOT). Furthermore, I have learned about space radiation and radiation modeling, topics that greatly interest me as I pursue a degree in physics. Working alongside experimental physicists like Dr. Rios, I have developed a greater understanding and appreciation for experimental science, something I have always leaned towards but to which I lacked significant exposure. My work in SRAG has also given me the invaluable opportunity to witness the work environment for physicists at NASA, and what a career in academia may look like at a government laboratory such as NASA Johnson Space Center. As I continue my studies and look forward to graduate school and a future career, this experience at NASA has given me a meaningful and enjoyable opportunity to put my skills to use and see what my future career path might hold.

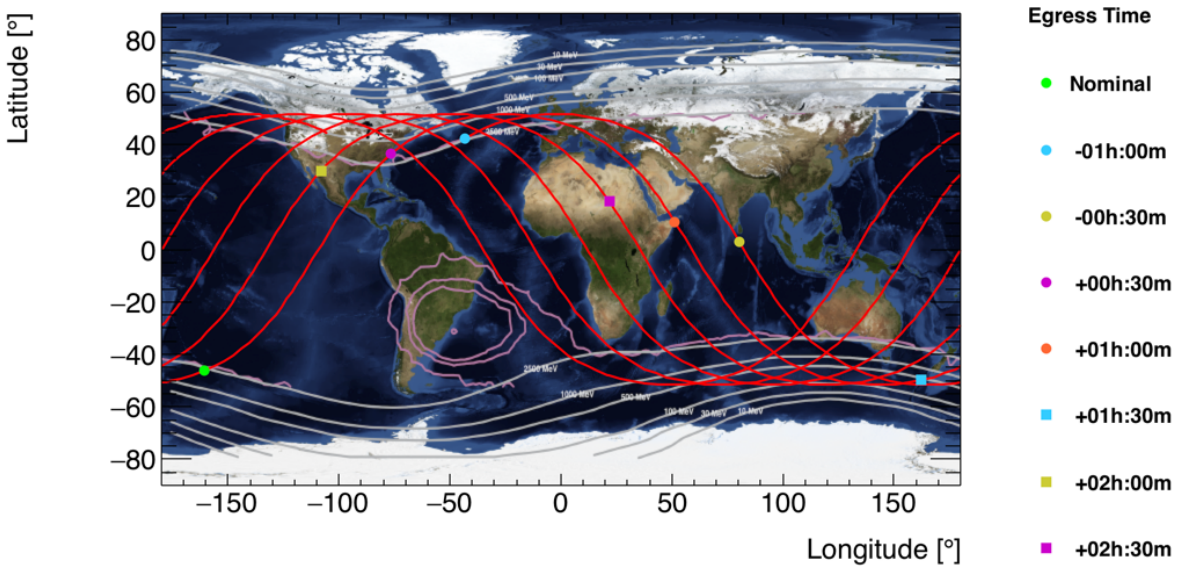

Figure 1. An example trajectory plot from GEnEVADOSE for a (fictitious) EVA with multiple proposed adjustments to the nominal egress time.