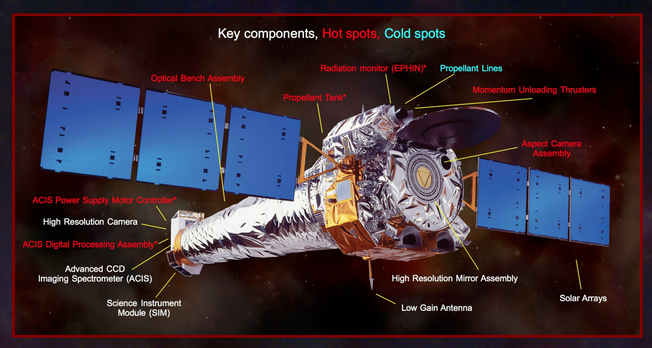
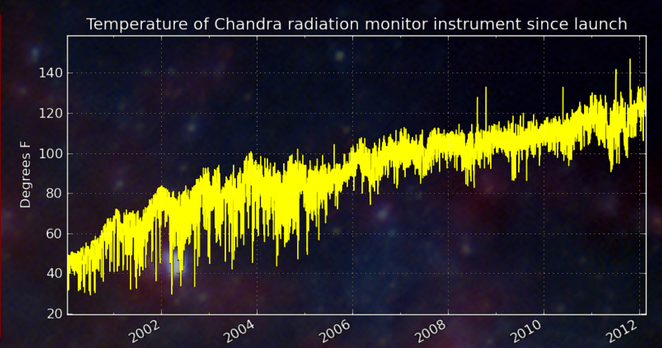
Behind the curtain: a look at the thermal models used for Chandra planning

Tom Aldcroft (Smithsonian Astrophysical Observatory / Chandra X-ray Center)
Matthew Dahmer (Northrop Grumman / Flight Operations Team)

The Chandra Science and Flight Operations Teams face a continual challenge in scheduling peer-reviewed and TOO observations while maintaining a safe thermal environment for all spacecraft components. Due to degradation of the silverized insulation that protects Chandra from the extremes of space, overall spacecraft temperatures have been increasing steadily since launch. The heating of different components depends on pitch angle, so staying within thermal limits is a delicate balancing act. A key part of the observation planning process is predicting on-board temperatures using the Xija thermal modeling framework which allows time-series modeling using pluggable components written in Python. In this poster we describe Xija and how it is used to make your Chandra observations happen.

The Challenge: climate change on Chandra





- Ionizing particle radiation environment is worse than expected and degrades the silverized teflon insulation
- No repair is possible since Chandra is in a high-Earth elliptical orbit
- Different parts of the spacecraft get too warm or cold depending on the relative “pitch angle” to the Sun
- Mission observing schedule is carefully planned to prevent overheating (or freezing)
- *This process requires accurate predictive models of spacecraft temperatures*
- Components marked with * above have such models
- Xija has reduced model development time dramatically, from many months down to days in some cases

Xija – Modular and extensible time series modeling framework in Python

- Solve linear first order coupled different equations:

$$\frac{d\vec{Y}}{dt} = \tilde{A}(t, \vec{Y}, \vec{p}) \cdot \vec{Y} + \vec{B}(t, \vec{Y}, \vec{p})$$

- The A coupling matrix and B vector can both depend on time, the output data values Y and the model parameters p
- Handle time series up to ~10^6 elements long
- Choice of minimization algorithms (fast versus robust)
- Handle models with up to ~100 parameters

- Modular and extensible
- Model definition via Python code or static JSON data structure
- Interactive and iterative model development and fitting
- Switch between predicting nodes or using truth (training) data
- Key integration steps coded in C for speed
- GUI interface for model fitting
- Fitting engine provided by **Sherpa** (*http://cxc.harvard.edu/contrib/sherpa*)

Fitting a model using the GUI fit tool

- ***Fit*** model and possibly ***stop*** before fit has completed
- ***Save*** the model and current fit parameters
- ***Add*** a plot (as defined by ModelComponent classes)

Real example: ACIS Digital Processor Assembly
- Handles overall instrument control and processing
- Could exceed temperature qualification limit : BAD
- Loss of this unit would cripple Chandra

Heating sources include:
- Sun (depends on sun pitch angle)
- Science Instrument Module translation position
- Instrument electronic power draw
- Nearby thermostatic heater element
- Thermal coupling to rest of SIM

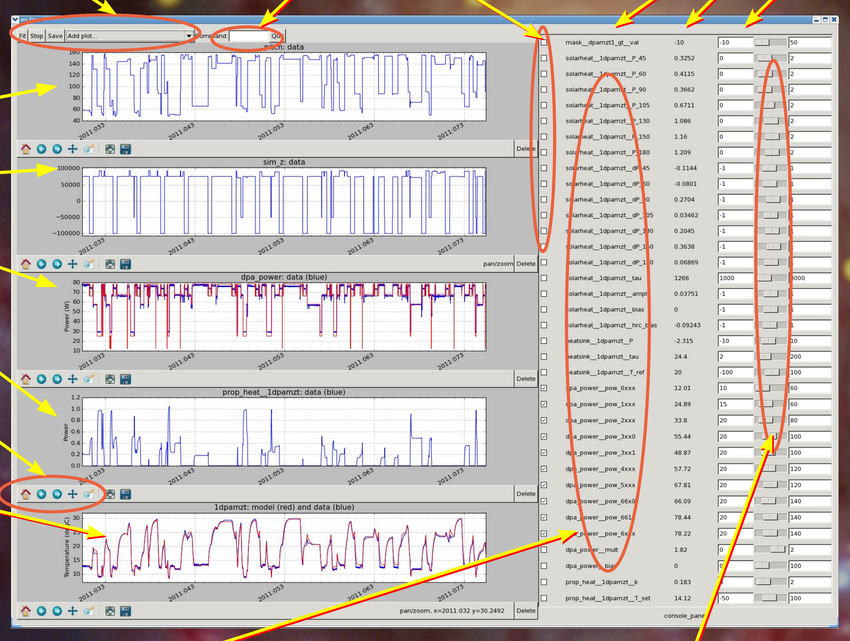
Plots are linked in time axis to allow investigation of poor model performance in certain regions

Input data from plots 1, 2, and 3 are used to compute the proportional heater power (plot 4) and the ACIS DPA temperature (plot 5)

The match between actual temperature (blue) and prediction (red) is good despite the complex variations

Freeze or thaw multiple parameters using glob syntax or with checkboxes

Parameter name, value, min and max (for fitting)



LOTS of parameters! But it is not so ridiculous because many are not tightly coupled. Most importantly, it works.

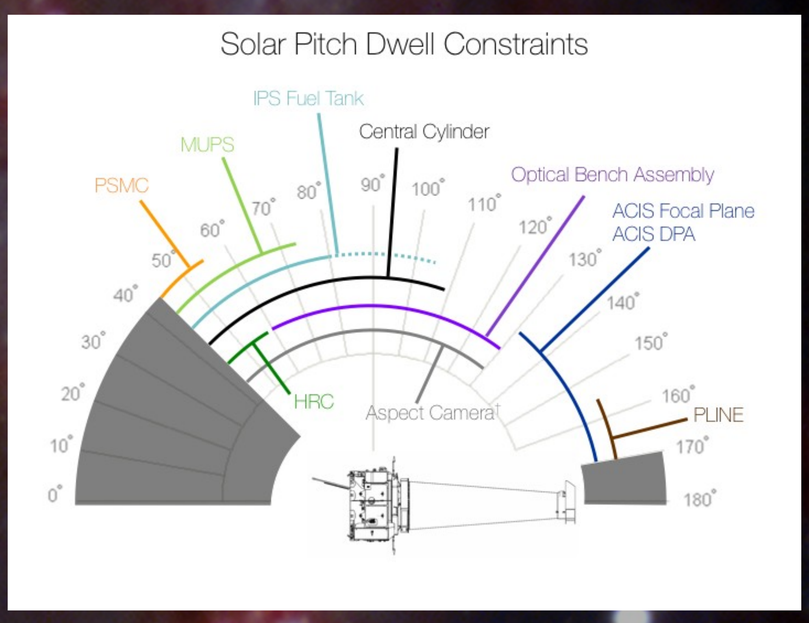
Sliders adjust parameter value and model prediction updates in real time. Useful for understanding parameter sensitivity.

During fitting the plots and sliders update to track the fit engine (but without getting in the way).

Short term scheduling

Every two weeks the Flight Operations Team Mission Planning group creates the detailed schedule of observations and commanding for the coming two weeks.

- Science targets are drawn from the Long Term Schedule.
- ***Planned schedule must not violate any thermal or spacecraft constraints***.
- As shown below, each of the affected systems have separate pitch regions and power states where they heat up, allowing a balance of heating and cooling time for each constraint to be built into the observation schedule.
- Schedule development is done iteratively by using the FOT Maneuver Constraint Checker (MCC) tool in conjunction with the OR Viewer, a graphical scheduling tool.
- MCC (written in Matlab) directly calls the Python Xija model framework to generate a predicted temperature profile given a proposed schedule. When the schedule meets all constraints it can be released for load review.
- The proposed schedule is then independently reviewed by teams with expertise for each subsystem. This includes the FOT thermal engineer.

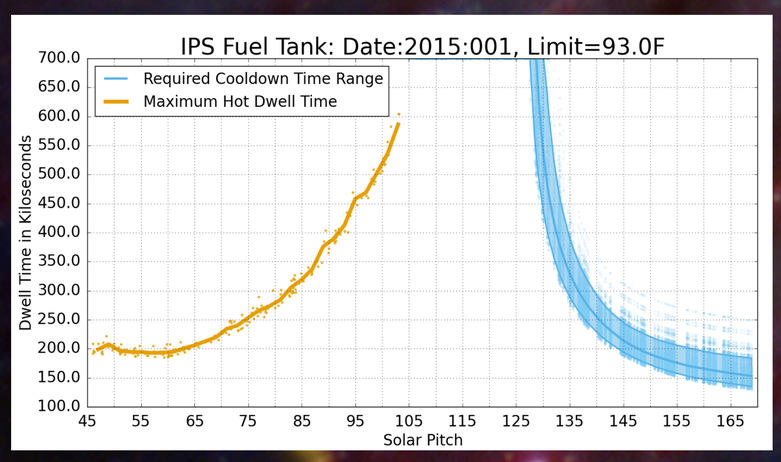


Long term scheduling: Pyger

A key task of the Science Operations Team Mission Planning group is laying out the long term schedule for a year of targets after the annual peer review. The set of targets in each two-week bin must allow meeting the thermal constraints when the detailed short term schedule is created.

An input to this process is the Pyger tool, which is used to determine the average balance of “hot” and “cool” time for each thermal constraint for a two-week bin.

This tool simulates thousands of dwells at all observable pitch values using past observations as initial conditions for each individual constraint. The subset of these simulations which result in temperatures that reach the specified thermal limit for each constraint are used to characterize the maximum hot dwell time, represented by the individual orange datapoints in the example Pyger plot. These hot dwell cases are then used to seed cooldown simulations to calculate the cooling time at various cooling attitudes required to balance the hot time. Since this cooling time can depend on a number of factors such as nearby temperatures, power states, and pitch, there is usually some variability in the balance of cooling time, represented by the blue datapoints. The three blue lines represent the 10, 50, and 90 percentile curve fits through the cooling simulation data.



Acknowledgements
- TLA is funded by NASA contract NAS8-39073.
- Discussions with the ACIS ops team and the Chandra Thermal Modeling Working Group were quite helpful.
- Many open source packages made this work possible, including NumPy, Matplotlib, Sherpa, PyGTK, PyTables, and of course Python itself.