Type Ia Supernovae: Models Meet Data

Carlo Graziani

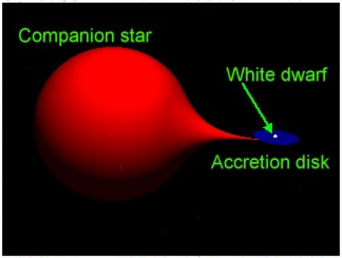
Flash Center For Computational Physics
The University of Chicago

10 August 2011

Outline

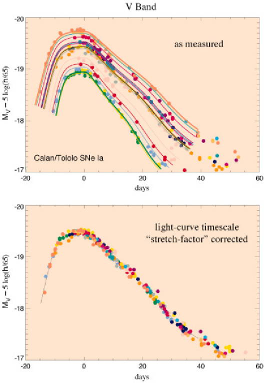
Outline

Type Ia Supernovae

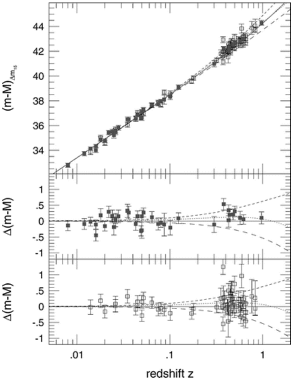


- Type Ia Supernovae (SNIa) are thermonuclear-powered flashes occurring in white dwarf progenitors.
- Standard model: White dwarf accretes matter from a companion star, gradually raising the temperature, simmering for $\sim$ 1000 years, until a fluctuation starts a runaway thermonuclear (C+C, C+O) reaction. Boom.

The Cosmology Connection



Source: Supernova Cosmology Project



Leibundgut 2001

- Acceleration of cosmic expansion (AKA “Dark Energy”) was noticed using SNIa as standard candles.
- Progress on measuring properties of dark energy with SNIa hinges on reducing Hubble diagram scatter — need better SNIa models.

SNIa (Multi) Physics

- Hydrodynamics
  - ...including sub-grid turbulence model for burning...
- Nuclear reaction network
- Simplified nuclear flame propagation
- Self-gravity
- Radiative transfer

SNIa Model Choice

- Two burning modes: Deflagration (reaction-diffusion balance) versus Detonation (shock compression/heating initiates reaction, which supplies "piston" to sustain shock). Initial runaway is a deflagration.
- Spectral line observations require heavy element production (Fe/Ni), which a pure deflagration cannot produce in sufficient abundance.
- $\Longrightarrow$ Require a detonation to be somehow initiated. Easier said than done.
- Two principal mechanisms:
  - Gravitationally-Confined Detonation (GCD)
  - Deflagration-to-Detonation transition (DDT)
- Each of these mechanisms corresponds to a modeling branch in our validation studies.

Outline

Type Ia Supernovae

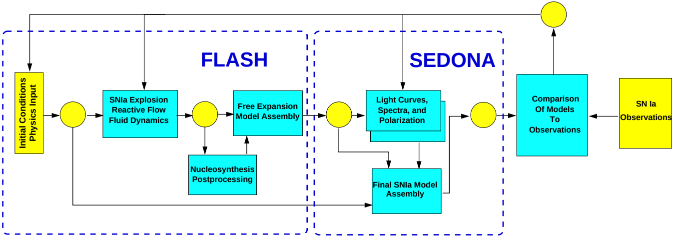
The Simulations

Verification

Validation

Uncertainty Quantification

The Simulations



Simulation Pipeline

- `FLASH` code simulates reactive hydrodynamics, from runaway, through detonation, to ballistic outflow. Also propagates passive particles that record spatial samples of thermal history.
- Nucleosynthetic post-processing is carried out on the particle data. Resulting yields are placed back on the outflow grid, and interpolated.
- `SEDONA` performs radioactive heating and radiative transfer in the outflow, producing SEDs to be compared to observed light curves and spectra.

The `FLASH` Code

- `FLASH` is a highly modular, extensible, extremely scalable multi-physics hydrodynamics/MHD code.
- Code infrastructure is designed for flexibility of configuration with respect to mesh management, I/O, performance monitoring, physics sub-models.
- *Community* code: All source available; Large user community; High-quality dev team support through mailing lists, tutorials; Many user-developed modules fed back into code.
- Developed and managed by professional scientific programmers: Enforced coding standards; Subversion version control; Outfitted with profiling tools; Regression- and unit-tested nightly with extensive multi-platform test suite; Extensively documented.
- Development driven by in-house science needs.

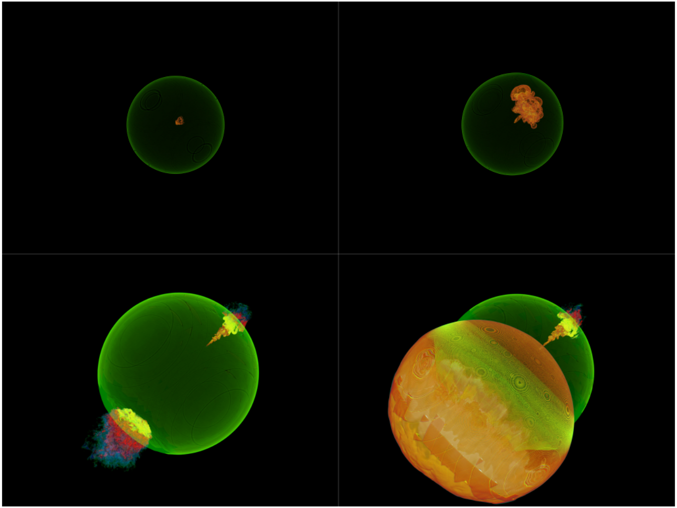
Other Codes

- Nucleosynthesis:
  - `NuCoToRT` Is a particle management/nucleosynthetic reaction network/grid interpolation code based on the `FLASH` framework.
- Radiative Transfer:
  - `SEDONA` is a 3-D Monte Carlo LTE radiation transfer code written and maintained by Dan Kasen (Berkeley).
  - `Phoenix` is a 1-D NLTE code that solves the PDE of radiative transfer.
  - Two very different methods, offering code-to-code verification opportunities.
- Simulation+Data Management:
  - `SMAASH` is a Simulation Management and Analysis System developed by the Flash Center for monitoring simulation health, analyzing data on the fly, and archiving data to long-term storage, while keeping track of past simulations — including science-laden metadata — in a MySQL database with a web interface.

Platforms/Runs

- `FLASH` and `SEDONA` production runs on 16K-32K cores of *Intrepid*, the ANL BG/P 163,840-core machine.
  - 8km resolution `FLASH` runs cost about 100K CPU-Hrs.
  - `SEDONA` runs cost about 400K CPU-Hrs.
- Development, debugging, and I/O co-processing use *Eureka*, *Intrepid*'s auxiliary analysis machine.
- `FLASH` output is typically ~ 2 TB per run.

A GCD Simulation



Jordan et al. 2008

Outline

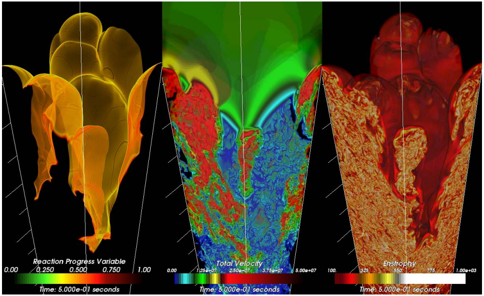
Verification: `FLASH` Test Suite

- Because the FLASH code is in a constant state of production and development, rapid verification of its correctness is a critical requirement.
- Testing is complicated by heterogeneity of supported platforms and applications.
- The FlashTest suite runs (currently 20) unit and (currently 60) regression tests nightly, on a variety of platforms (hardware/compiler combinations).

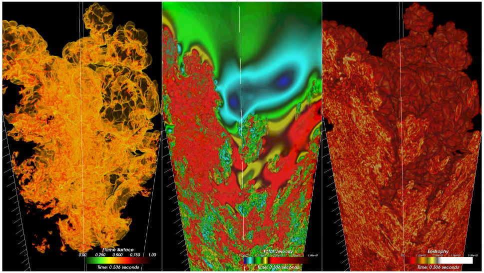
Verification: Turbulent Nuclear Burning

- The deflagration flame width is about 1 cm. Our mesh resolution is 4–8 Km.
- This is a problem because the energy release by the flame is strongly affected by unresolved turbulent flows, which increase the burning rate by increasing the flame area.
- In addition, if the unresolved turbulence is sufficiently strong at scales set by the competition between the flame speed and the RT growth rate, the flame will be torn apart and transition to the "distributed burning" regime. This is a necessary condition for the DDT model.
- We have carried out extensive flame verification simulations, in stars and in columns and open domains with constant gravity, for the purpose of calibrating a phenomenological sub-grid burning model, and to characterize the flame surface at the relevant scales.

Verification: Turbulent Nuclear Burning



$S_{flame} = 30$ Km/s



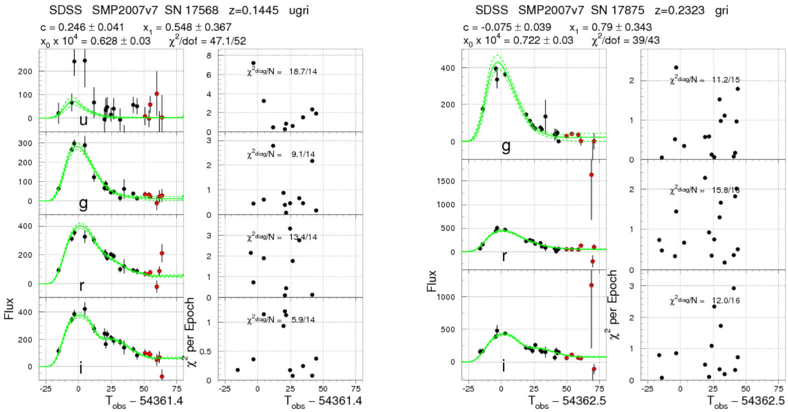
$S_{flame} = 6$ Km/s

Verification: Turbulent Nuclear Burning

- The simulations show that the flame is complex at large scales, but smooth at small scales.
- Transition to distributed burning only occurs at low flame speeds (i.e. low densities, i.e. shallower depth in the star). This results in higher deflagration-phase energy release and more inhomogeneous compositional structure for the DDT model than for the GCD model — an observational signature.

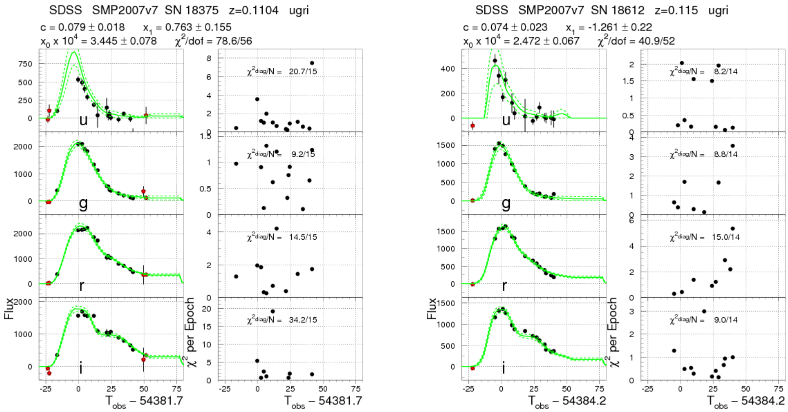
Outline

Validation: SDSS-II Supernova Data



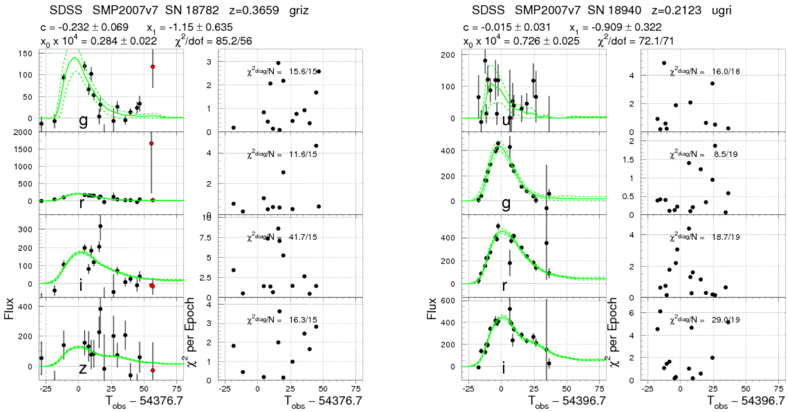
- Sample of 133 nearby, high S/N SNIa events observed by the SDSS-II Supernova Search Team.
- Green lines are the SALT2 "model" — actually a trained summary of all data.

Validation: SDSS-II Supernova Data



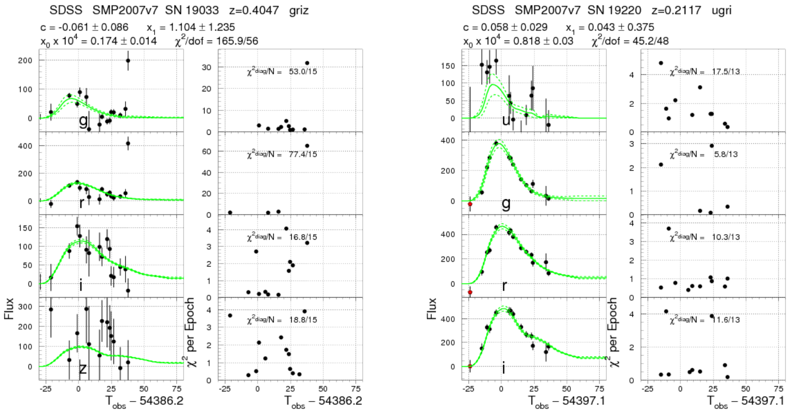
- Sample of 133 nearby, high S/N SNIa events observed by the SDSS-II Supernova Search Team.
- Green lines are the SALT2 “model” — actually a trained summary of all data.

Validation: SDSS-II Supernova Data



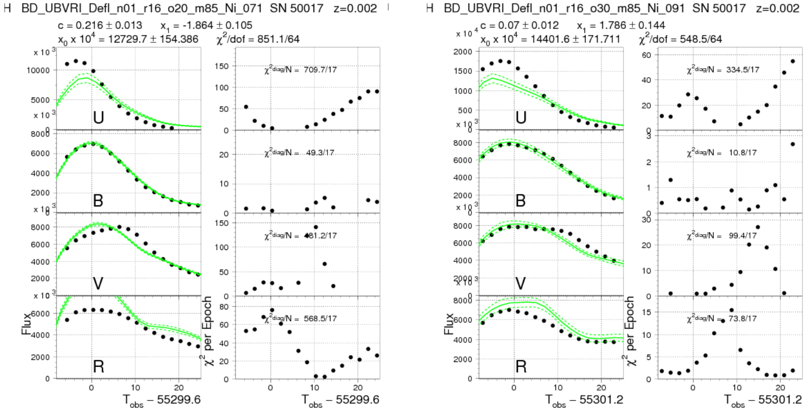
- Sample of 133 nearby, high S/N SNIa events observed by the SDSS-II Supernova Search Team.
- Green lines are the SALT2 "model" — actually a trained summary of all data.

Validation: SDSS-II Supernova Data



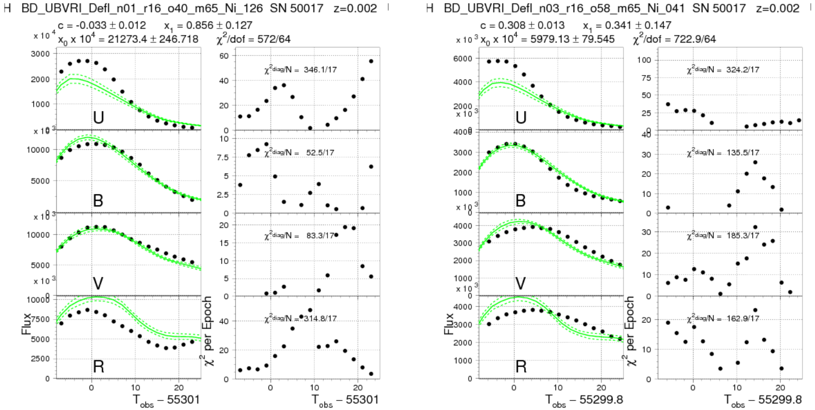
- Sample of 133 nearby, high S/N SNIa events observed by the SDSS-II Supernova Search Team.
- Green lines are the SALT2 "model" — actually a trained summary of all data.

Validation: Models Meet Data



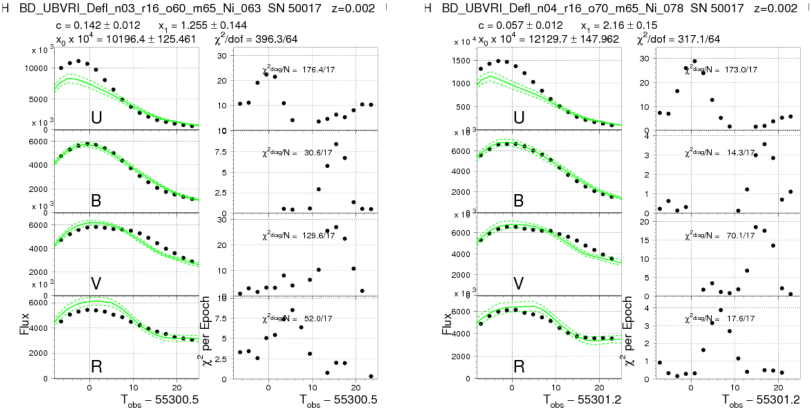
- Dots are now simulation output "data" fit to SALT2 "model"
- Fits indicate considerable model error for early simulations.

Validation: Models Meet Data



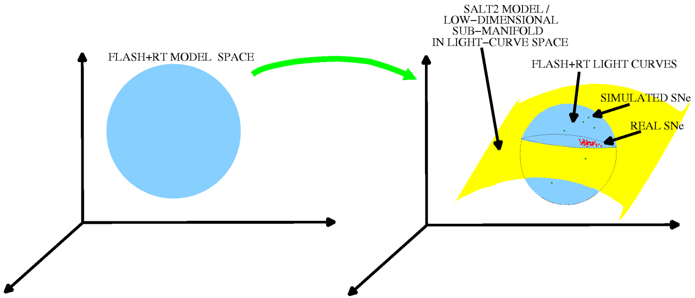
- Dots are now simulation output “data” fit to SALT2 “model”
- Fits indicate considerable model error for early simulations.

Validation: Models Meet Data



- Dots are now simulation output “data” fit to SALT2 “model”
- Fits indicate considerable model error for early simulations.

Validation: Models Meet Data (Cont'd)



- Physics model parameter space is 5-10 dimensional.
- SNIa correlations indicate that data somehow lives on a 2-dimensional sub-manifold of the (typically) 50-100 dimensional space of light curves.
- We choose fit quality scores (like χ^2_{SALT2}) designed to help guide our models to the right ZIP code — i.e. both towards the SALT2 sub-manifold *and* towards the region populated by real SNIa.

Validation: Models Meet Data (Cont'd)

- We use the SALT2 "model" errors as weights in χ^2, since they represent quantitatively the relative extent to which the model is "well trained" in various bands and epochs.
- We do *not* sweat the horrible values of χ^2 at this point, since at this stage we are dealing with an optimization problem, rather than with a strictly statistical problem.
- When (if?) we locate regions of physical parameter space that can produce light curves within shouting distance of the actual data, we will discard the SALT2 model and attempt to fit individual SNIa light curves using more-or-less standard statistical methods.

Outline

Uncertainty Quantification: Objectives

- This is discovery science. Unlike many engineering/reliability applications, where the main goal is prediction, we are more interested in inference than in prediction:
    - Model Choice — we want to accumulate evidence favoring some classes of models and excluding others;
    - Calibration (AKA parameter estimation) — We'd like to be able to measure, as well as possible, model parameters such as progenitor mass, composition, ignition modes etc. for individual SNIa. The distributions of such parameters across SNIa have intrinsic scientific interest.
    - We'd like a more reliable, lower-scatter, physics-based luminosity estimator, to probe the properties of Dark Energy.

Response Surface Emulation

- Model evaluations are too expensive to perform densely over a 5-10-dimensional parameter space (as an MCMC invocation, for example).
- Over the unsampled parameter space, the response surface will have to be approximated by an *emulator* — an interpolation “trained” using available simulations at sampled parameter values.
- An emulation technique of great promise (but with substantial challenges) is Gaussian Process modeling (Kennedy & O’Hagan 2001, Higdon 2004). A random function with a GP distribution is chosen as the response surface approximation — a “fuzzy” interpolant.
- In principle, a carefully-chosen GP emulator can supply an accounting of interpolation error in the overall uncertainty budget.

A Refinement: Model Fidelity

- One usually has the option of running one's computational model at lower — and cheaper — levels of model fidelity.
- For example, in SNIa simulations:
  - Lower mesh resolution;
  - 2-D instead of 3-D;
  - Cheaper physics/surrogate models.
- This opens the possibility of establishing a *Fidelity Hierarchy*, wherein we probe model parameter space using abundant sprays of cheaper, lower-fidelity models runs. Discrepancies from the high-fidelity model are calibrated using less-frequent high-fidelity runs at carefully-chosen parameter settings.
- Some results exist (Kennedy & O'Hagan 2000, Qian & Wu 2008, Cumming & Goldstein 2009). Some generalization desirable, e.g. non-stationary kernels, fidelity levels not strictly ordered by informativeness.

Model Fidelity Hierarchy

- Different levels of hierarchy denoted by $\Phi = 1, \ldots, \Phi_{max}$.
- Parameter settings $\boldsymbol{\theta}_\mu^{(\Phi)}, \mu = 1, \ldots, N_\Phi$.
- Code outputs $y^{(\Phi)}(\boldsymbol{\theta}_\mu^{(\Phi)}, \mathbf{x}_i), i = 1, \ldots, N_{out}$.
- Unknown true parameter setting $\boldsymbol{\theta}_T$, corresponding to measurements $\mathbf{y}_{meas}$.

Fidelity Hierarchy GP Model

$$\begin{aligned} y^{(\infty)}(\boldsymbol{\theta}, \mathbf{x}) &= y^{(\Phi_{max})}(\boldsymbol{\theta}, \mathbf{x}) + \delta(\mathbf{x}); \\ y^{(\Phi)}(\boldsymbol{\theta}, \mathbf{x}) &= y^{(\Phi_{max})}(\boldsymbol{\theta}, \mathbf{x}) + \Delta^{(\Phi)}(\boldsymbol{\theta}, \mathbf{x}), \\ & \qquad\qquad\qquad\qquad \Phi \neq \Phi_{max}, \end{aligned}$$

- $\delta(\mathbf{x}_i)$ is the *Global Discrepancy Term* (Higdon 2004).
- $\Delta^{(\Phi)}(\boldsymbol{\theta}, \mathbf{x})$ are *Inter-Fidelity Level Discrepancy* terms.

$$\begin{aligned} y^{(\Phi_{max})} &\sim GP\left(\mu^{(\Phi_{max})}, k^{(\Phi_{max})}(\boldsymbol{\theta}, \mathbf{x}; \boldsymbol{\theta}', \mathbf{x}')\right) \\ \Delta^{(\Phi)} &\sim GP\left(\mu^{(\Phi)}, k^{(\Phi)}(\boldsymbol{\theta}, \mathbf{x}; \boldsymbol{\theta}', \mathbf{x}')\right) \\ & \qquad\qquad\qquad\qquad \Phi \neq \Phi_{max} \\ \delta &\sim GP\left(\mu^{(\infty)}, k^{(\infty)}(\mathbf{x}; \mathbf{x}')\right) \end{aligned}$$

Adaptive Numerical Experimental Design

- At what parameter settings should simulations be performed?
- An *a priori* design, such as latin hypercube, is unlikely to capture important features in a complex-structure response surface.
- An *adaptive*, iterative scheme, which learns from previous emulator/data comparison to predict new, useful simulation parameter settings seems crucial, especially as the parameter space gets large.
- The scheme must resolve a tension — the *Exploration-Exploitation Tradeoff* — between two important objectives: Undestanding the response surface everywhere, and locating regions where it most resembles the experimental data.

Adaptive Numerical Experimental Design, Cont'd

- Efforts to-date have been focused on exploration — good characterization of response surface to be emulated, ignoring the data (e.g. Gramacy & Lee 2009, Cumming & Goldstein 2009).
- For consideration of the full exploration/exploitation tension one has to look to the literature on *physical* experimental design.
- A particularly promising development: Loredo & Chernoff (2004) show that physical measurements currently "in the can" can be used to calculate the expected information gain — negative Shannon entropy, measured in bits of information — from a future measurement with selected experimental parameters, and so choose those parameters so as to maximize that information.

Adaptive Numerical Experimental Design, Cont'd

This development translates immediately over from physical to numerical experimental design. Moreover, GP structure of emulator results in feasibly-computable information measures.

$$
\begin{aligned}
\mathrm{Info}(\boldsymbol{\theta}_+) &= H\left[\pi(\mathbf{y}_+|\boldsymbol{\theta}_+, \mathbf{Y}, \boldsymbol{\Theta})\right] \\
&\quad - \int d\boldsymbol{\theta}_T \, \pi(\boldsymbol{\theta}_T|\mathbf{y}_{meas}, \mathbf{Y}, \boldsymbol{\Theta}) H\left[\pi(\mathbf{y}_+|\boldsymbol{\theta}_+, \mathbf{Y}, \mathbf{y}, \boldsymbol{\Theta}, \boldsymbol{\theta}_T)\right].
\end{aligned}
$$

- $H[\pi(\cdot)]$ is entropy of PDF π; $\boldsymbol{\theta}_+$ is proposed new parameter point; $(\mathbf{Y}, \boldsymbol{\Theta})$ is the current numerical design; $\mathbf{y}_{meas}$ are the observations.
- The first term embodies exploration (by itself, it yields Maxent sampling). The second term embodies exploitation, rewarding smaller predictive uncertainty near best-fit parameter point.

Adaptive Numerical Experimental Design, Cont'd

- When all simulations are equal cost, the use of this measure is simple: for a large set of proposed future simulations, obtain the number of bits of information expected from each one based on the current design, rank-order them, skim off the highest few, and dispatch them to the available computing resources.
- When simulation costs differ for different fidelity levels, criterion is no longer bits, but bits/CPU-Hr. The simulations most highly ranked by this criterion are the ones to be dispatched to the hardware at the beginning of each new "Simulation-Inference-Design" cycle.