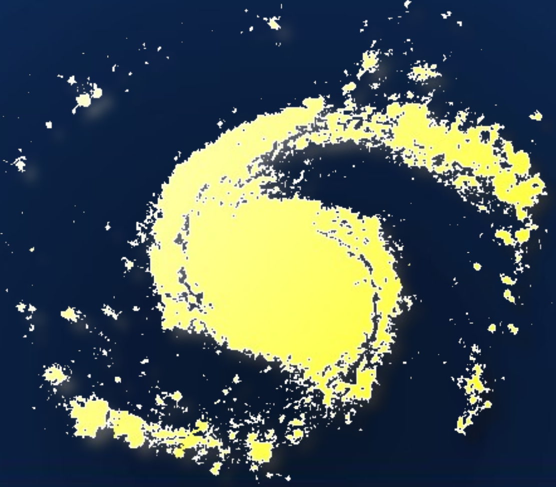
Advancing Galaxy Formation Modeling

Andrew Benson

California Institute of Technology



Advancing Galaxy Formation Codes

- Why a new code?
  - Adding in new features (e.g. self-consistent reionization, noninstantaneous recycling, new star formation rules) to existing models can be challenging
- How?
  - Create a code which is modular by design, isolating assumptions so that they don't have consequences throughout the code.

GALACTICUS

sites.google.com/site/galacticusmodel

Design Features

- Open source (compiles with GNU compilers)
- Modular design
  - Each function can have multiple implementations, selected by input parameter.
  - "Node" can have arbitrary number of components (e.g. DM halo, disk, spheroid), all with multiple implementations
- Combination of smooth (ODE) evolution and instantaneous events (e.g. mergers)

Design Features

- Well documented
- Promotes a standard format for merger tree data
  - w[illegible]caltech.edu/galacticus/MergerTreeFileFormat.pdf
- Parallelized
  - OpenMP
  - MPI (soon...)
  - Currently simple, but allows for expansion

Source code

Binaries

Cloud (Amazon EC2)

External Tools

- GNU Scientific Library/`FGSL`
  - ODE solver; integration; other numerics
- `FoX` library
  - Read/write XML files
- `FSPS`
  - Population synthesis
- `Cloudy`
  - Cooling times

Modularity

- New implementation of function easily added:
  - Write a module containing the function
  - Add directives indicating that this function is for, e.g., disk star formation timescale calculations
  - Recompile – build system automatically finds this new module and works out how to compile it into the code
- Modules are self-contained and independent
- Self-initializing and recursive

Node Components

- Component could be, e.g. disk (exponential)
- Stores various types of data:
  - Properties – evolved within ODE system
  - Data – internal data, not evolved
  - Histories – records of past/future history (e.g. star formation history)
- Allows for multiple components of each type

Node Components

- Defining a component:
  - Set of ODEs giving rates of change of properties (can access properties of other components/nodes as needed)
  - Responses to events (merging, becoming satellite etc.)
  - Specify properties to be output

Node Evolution

- Repeatedly walk tree – find nodes that to evolve:
  - Cannot evolve if have children
  - Can't evolve beyond their satellites
  - Limit on timestep
  - Arbitrary other factors can be included
- Evolve nodes forward
- Stops when no more nodes to evolve

Node Evolution

- All component properties fed into ODE solver
- Evaluate derivatives – evolve forward in time
- No need for fixed timesteps or analytic solutions
  - Makes implementing, for example, Kennicutt-Schmidt law trivial (just add new star formation timescale function)
- Evolution can be interrupted as needed (e.g. when galaxy merges)

Node Evolution

- Component creation:
  - Nodes begin with only basic component (mass, time)
  - If accretion from IGM occurs, stop and create a hot halo component
  - If cooling occurs, stop and create a disk component
  - Components can be destroyed as needed also

Advantages

- Modularity makes it highly flexible:
  - Add new star formation rule in 5 minutes
  - Change in cooling model confined to few modules which compute cooling time and rate
- Unified ODE solver makes new features simple:
  - Timestepping handled automatically
  - No need for analytic solutions
  - Implemented noninstantaneous recycling in one afternoon rather than two months!

Disadvantages

- Slower
  - Wasn't designed for speed, but for simplicity
- Missing features (planned for addition):
  - Ram pressure/tidal stripping
  - Self-consistent reionization
  - Satellite orbits/disk heating
  - etc.

ICM heating/X-ray emission
Multi-level hierarchy
Black hole merging timescales/kicks
~~H_2-based star formation~~
Resolved disks
~~Compton/H_2 cooling~~
Deterministic spins/concentrations

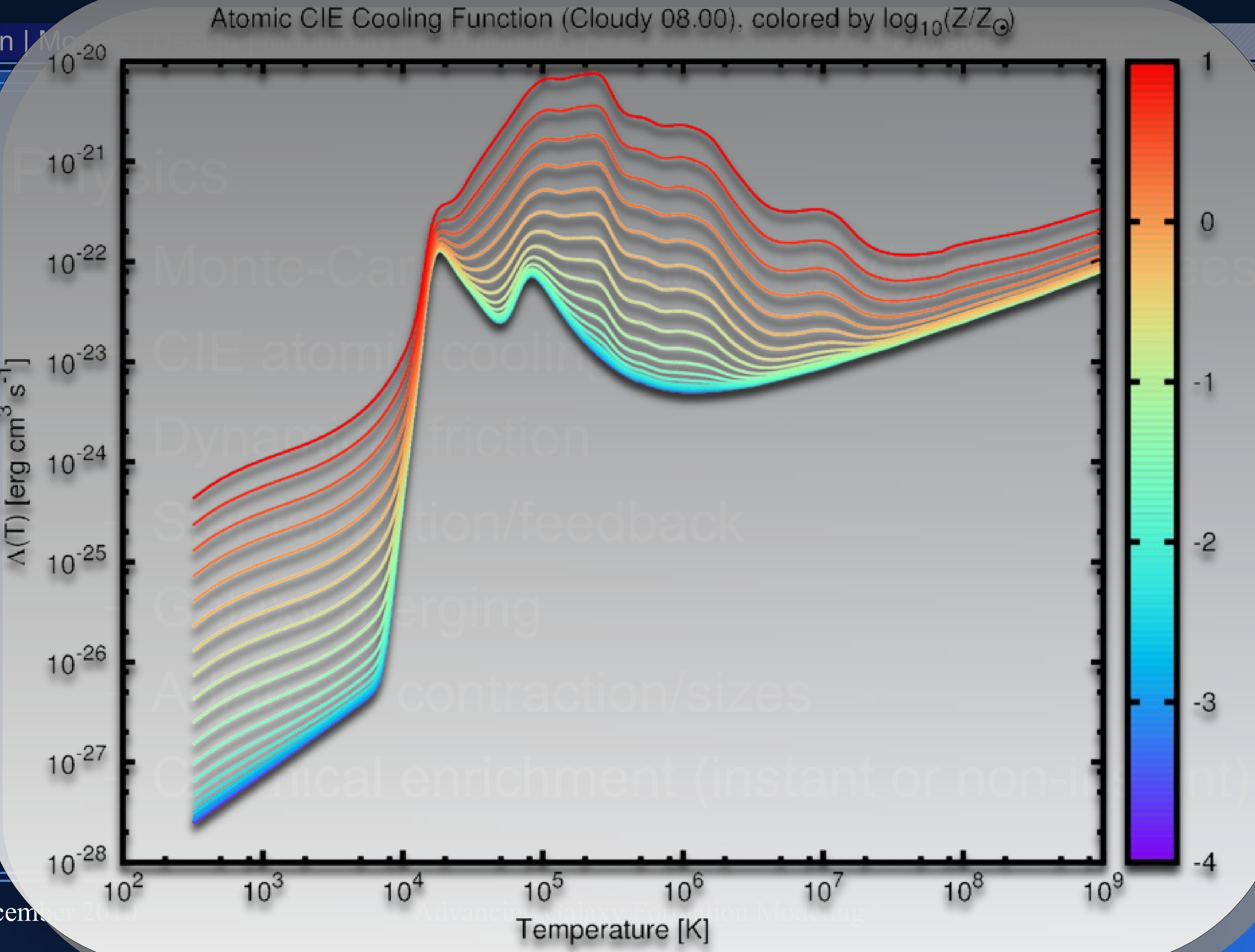
Current Feature List

- Components
  - DM profile [isothermal/ NFW]
  - Hot halo
  - Disk [exponential]
  - Spheroid [Hernquist]
  - Black holes

Tracks mass and spin.
Spin from mergers and accretion.
Accretion spin-up using Benson & Babul formula
Jet power from Benson & Babul also.

Current Feature List

- Physics
  - Monte-Carlo merger trees
  - CIE atomic cooling
  - Dynamical friction
  - Star formation/feedback
  - Galaxy merging
  - Adiabatic contraction/sizes
  - Chemical enrichment (instant or non-instant)



Current Feature List

- Physics *(cont.)*:
  - Disk instabilities
  - Black hole merging
  - AGN feedback
  - Stellar population synthesis (with arbitrary IMF)

Summary

GALACTICUS

sites.google.com/site/galacticusmodel