New Astronomy 17 (2012) 175–197

Contents lists available at SciVerse ScienceDirect

New Astronomy

journal homepage: www.elsevier.com/locate/newast

# GALACTICUS: A semi-analytic model of galaxy formation

Andrew J. Benson

*California Institute of Technology, MC350-17, 1200 E. California Blvd., Pasadena, CA 91125, USA*

ARTICLE INFO

*Article history:*
Received 11 August 2010
Accepted 15 July 2011
Available online 22 July 2011

Communicated by J. Silk

*Keywords:*
Galaxies
Cosmology
Galactic structure
Galaxy evolution
Galaxy formation

ABSTRACT

We describe a new, free and open source semi-analytic model of galaxy formation, GALACTICUS. The GALACTICUS model was designed to be highly modular to facilitate expansion and the exploration of alternative descriptions of key physical ingredients. We detail the GALACTICUS engine for evolving galaxies through a merging hierarchy of dark matter halos and give details of the specific implementations of physics currently available in GALACTICUS. Finally, we show results from an example model that is in reasonably good agreement with several observational datasets. We use this model to explore numerical convergence and to demonstrate the types of information which can be extracted from GALACTICUS.



## 1. Introduction

The physics of galaxy formation is rich and complex, and has provided a challenge for theorists to understand and explain ever since it became clear that the Universe is filled with galaxies (Shapley and Curtis, 1921). Various theoretical tools have been brought to bear on this problem, ranging from direct analytic reasoning to large scale numerical *N*-body simulations. An intermediate and highly successful approach is that known as "semi-analytic" galaxy formation modeling. In this approach the numerous complex non-linear physics involved are solved using a combination of analytic approximations and empirical calibrations from more detailed, numerical solutions. Historically, such models were first contemplated by White and Rees (1978) and have since been developed further, notably by White and Frenk (1991), Kauffmann et al. (1993), Cole et al. (2000), Hatton et al. (2003), Monaco et al. (2007), and Somerville et al. (2008) among others. Models of this type aim to begin with the initial state of the Universe (specified shortly after the Big Bang) and apply physical principles to determine the properties of galaxies in the Universe at later times, including the present day. Typical properties computed include the mass of stars and gas in each galaxy, broad structural properties (e.g. radii, rotation speeds, geometrical shape etc.), dark matter and black hole contents, and observable quantities such as luminosities, chemical composition etc. As with *N*-body/hydrodynamical simulations, the degree of approximation varies considerably with the complexity of the physics being treated, ranging from precision-calibrated estimates of dark matter merger rates to empirically motivated scaling functions with large parameter uncertainty (e.g. in the case of star formation and feedback).

The primary advantage of the semi-analytic approach is that it is computationally inexpensive compared to *N*-body/hydrodynamical simulations. This facilitates the construction of samples of galaxies orders of magnitude larger than possible with *N*-body techniques and for the rapid exploration of parameter space (Henriques et al., 2009; Benson and Bower, 2010b; Bower et al., 2010) and model space (i.e. adding in new physics and assessing the effects). The primary disadvantage is that they involve a greater degree of approximation. The extent to which this actually matters has not yet been well assessed. Comparison studies of semi-analytic vs. *N*-body/hydro calculations have shown overall quite good agreement (at least on mass scales well above the resolution limit of the simulation) but have been limited to either simplified physics (e.g. hydrodynamics and cooling only; Benson et al., 2001; Yoshida et al., 2002; Helly et al., 2003; Lu et al., 2010) or to simulations of individual galaxies (Stringer et al., 2010).

In this work, we describe a new, free and open source semi-analytic model, GALACTICUS. The GALACTICUS model solves the physics describing how galaxies evolve in a merging hierarchy of dark matter halos in a cold dark matter universe. GALACTICUS has much in common with other semi-analytic models, such as the range of physical processes included and the type of quantities that it can predict, but has some key distinguishing features. In designing GALACTICUS our main goal was to make the code flexible, modular and easily extensible. Much greater priority was placed on making the code easy to use and to modify than on making it fast. We believe that a modular and extensible nature is crucial as galaxy

*E-mail address:* abenson@caltech.edu

1384-1076/$ - see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.newast.2011.07.004

formation is an evolving science. In particular, key design features are:

**Extensible methods for all functions:** Essentially all functions within Galacticus are designed to be extensible, meaning that anyone can write their own version and easily insert it into G alacticus. For example, suppose an improved functional form for the cold dark matter (CDM) halo mass function is derived. A user of Galacticus can simply write a short module conforming to a specified template that computes this mass function and which includes a short directive in the code which explains to Galacticus's build system how to incorporate this module. A recompile of the code will then incorporate the new function. The user is absolved from having to understand the details of the inner workings of the code, instead simply being required to conform to a standard interface.

**Extensible components for tree nodes:** The basic structure in Galacticus is a merger tree, which consists of a set of linked tree nodes which have various properties. Galacticus works by evolving the nodes forward in time subject to a collection of differential equations and other rules. Each node can contain an arbitrary number of *components*. For example, a component may be a dark matter halo, a galactic disk, a black hole etc. Each component may have an arbitrary number of *properties* (some of which may be evolving, others of which can be fixed). Galacticus makes it easy to add additional components. For example, suppose that a user wanted to add a "stellar halo" component (consisting of stars stripped from satellite galaxies). To do this, they would write a module which specifies the following for this component:

- number of properties;
- interfaces to set and get property values and rates of change;
- "Pipes" which allow for flows of mass/energy/etc. from one component to another (and which facilitate interaction between components without the need for knowledge of the specific implementation of any component);
- functions describing the differential equations which govern the evolution of the properties;
- functions describing how the component responds to various events (e.g. the node becoming a satellite, a galaxy-galaxy merger, etc.);
- auxiliary routines for handling outputs etc.

Short directives embedded in this module explain to the Galacticus build system how to incorporate the new component. A recompile will then build the new component into Galacticus. Typically, a new component can be created quickly by copying an existing one and modifying it as necessary. Furthermore, multiple implementations of a component are allowed. For example, Galacticus contains a component which is a Hernquist spheroid. One could add a de Vaucouler's spheroid component and an input parameter then allows one to simply select which implementation will be used in a given run.

**Centralized ODE solver:** Galacticus evolves nodes in merger trees by calling an ODE solver which integrates forward in time to solve for the evolution of the properties of each component in a node. This means that it is not necessary to provide explicit solutions for ODEs (in many cases such solutions are not available anyway) and time-stepping is automatically handled to achieve a specified level of precision. The ODE solver allows for the evolution to be interrupted. A component may trigger an interrupt at any time and may do so for a number of reasons. A typical use is to actually create a component within a given node—for example when gas first begins to cool and inflow in a node a galactic disk component may be created. Other uses include interrupting evolution when a merging event occurs.

To summarize, the Galacticus code is therefore highly modular. Every part of it consists of a simple and well-defined interface into which an alternative implementation of a calculation can easily be added. Similarly, the physical description of galaxies is extremely flexible. Each galaxy has a set of components which can be created/destroyed as needed, each of which has a set of properties. These components are also modular, and any such module can be trivially replaced by another that performs the calculations differently if required. The actual formation and evolution of galaxies is treated by simply defining a set of differential equations for each galaxy. Then, these are all fed into the ODE solver which evolves them to a specified accuracy.

The goal of this article is to describe the technical and physical implementation of Galacticus. Detailed examination of its scientific predictions and their consequences will be deferred to a future work. The remainder of this article is arranged as follows. In Section 2 we describe the technical implementation of Galacticus, while in Section 3 we describe the current specific implementation of galactic components, and in Section 4 the current specific implementation of numerous physical processes and properties. In Section 5 we show the results of an example calculation. Finally, in Section 6 we summarize the information presented in this article. The Galacticus model is available for download from http://sites.google.com/site/galacticusmodel. A full manual documenting both the physics and technical implementation of galacticus can also be found on the same website.

## 2. Technical implementation

In the following section we describe the technical and practical aspects of Galacticus. The Galacticus code is free and open source and was designed to depend only on free and open source compilers and libraries to maximize its portability. In addition to the GNU compilers, Galacticus requires the GNU Scientific Library (GSL) and the FGSL wrapper for GSL, the FoX XML parser and the HDF5 library. Additionally, the analysis codes provided with Galacticus make extensive use of Perl and PDL, although Galacticus output can be just as easily analyzed with other tools.

### 2.1. Running

The behavior of Galacticus is controlled by a set of parameters which are given in a file specified as a command line argument. (Galacticus will in fact provide sensible default values for all parameters if no parameter file is specified, or if some parameters are missing from the file). The parameter file is an XML file, which allows it to be generated easily using a variety of pre-existing XML tools and libraries. Scripts are provided with Galacticus to generate parameter files and run grids of models which span a range of parameter values.

When run, Galacticus proceeds by performing a set of predefined tasks in order until all tasks are done. As with all other aspects of Galacticus, these tasks are modular and extensible, allowing new tasks to be added in as desired. Typically, however, the tasks will consist of some initialization, followed by creation and evolution of one or more merger trees.

### 2.2. Output

Galacticus stores its output in an HDF5 file which allows this output to be viewed and manipulated using a variety of tools in addition to the standard HDF5 C API including:

HDFView A graphical viewer for exploring the contents of HDF5 files;

**HDF5 Command Line Tools**. A set of tools which can be used to extract data from HDF5 files (`h5dump` and `h5ls` are particularly useful);
**C++ and Fortran 90 APIs**. Allow access to and manipulation of data in HDF5 files from within C++ and Fortran90 codes;
H5PY A Python interface to HDF5 files;
**PDL::IO::HDF5**. A Perl interface to HDF5 files.

#### 2.2.1. Output datasets

In addition to the properties of galaxies and their associated dark matter halos in all merger trees, each Galacticus output file stores a full record of all parameter values (either input or default) that were used for the particular run. These can easily be extracted to an XML file suitable for re-input into Galacticus. The output also contains a record of the Galacticus version used for this model, storing the major and minor version numbers, and the revision number along with the time at which the model was run.

Optionally, Galacticus will compute and store volume averaged properties of the entire galaxy population at a set of snapshot times. Currently, the properties stored are star formation rate density, stellar mass density, interstellar medium (ISM) gas density and the density in resolved dark matter halos, all as a function of cosmic time.

Galaxy data can be output at one or more snapshot times, specified as input parameters to Galacticus. At each output time, each merger tree is stored separately within the HDF5 file to facilitate easy access. Each such merger tree group contains all data on a single merger tree, and consists of a collection of datasets each of which lists a property of all nodes in the tree which exist at the output time. Additionally, a weight (in $\mathrm{Mpc}^{-3}$) which should be assigned to this tree (and all nodes in it) to create a volume-averaged sample is stored.

Optionally, a mass accretion history (i.e. mass as a function of time) for the main branch[1] in each merger tree can be output, which lists the mass of the main branch as a function of time.

Finally, Galacticus can output the full structure of merger trees prior to any evolution. This is useful to permit the same trees to be used in other codes, or to examine how galaxy properties depend on merging history for example. If such output is requested the mass of each node in the tree is recorded, along with the cosmic time at which it exists, indices describing relationships between parent, child and sibling nodes and, if desired, quantities such as the virial radius, virial velocity and dark matter scale radius.

Additionally, Galacticus allows for arbitrary additional outputs to be easily implemented.

#### 2.2.2. Post-processing of galaxy properties

Galacticus is provided with a Perl module that allows for easy extraction of datasets from a Galacticus output file together with a straightforward way to implement derived properties (i.e. properties computed by post-processing the output data). Implementations for simple dust-extinction calculations are provided which utilize this framework, together with modules that convert output luminosities into absolute magnitudes in AB or Vega systems. Any such derived properties can be stored back to the Galacticus output file.

### 2.3. Node evolution engine

Galacticus's main task is to evolve galaxies (and their associated dark matter halos) through a complex merging hierarchy. It begins with a pre-constructed dark matter halo merger tree as depicted in the "Stage 1" panel of Fig. 1. In this figure, colored circles represent nodes in the tree (larger circles implying greater mass total mass) and the $y$-axis represents time increasing upwards (such that $t_1 > t_2 > t_3$ etc.). Solid lines connect main progenitor nodes (typically the most massive) to their parents, while dotted lines connect other progenitor nodes (ones that will become subhalos) to their parents. Each node in the tree is given a unique ID number. Galacticus begins with the root node (number 1) and checks to see if it can be evolved forward in time. Since it has some progenitors, it cannot. Therefore, Galacticus steps to the primary progenitor (indicated by the dashed red arrow) and applies the same condition. This eventually leads it to node 4 which has no progenitor and so can be evolved forward in time (indicated by the dotted green arrow).

Once node 4 reaches the time at which its parent, node 3, is defined, node 4 has effectively become node 3 and is promoted, replacing node 3. This is a *node promotion event* in Galacticus parlance. Since it still has no progenitors, Galacticus will continue to evolve node 4 until it reaches node 2 and is promoted to replace it, as indicated in "Stage 2" in Fig. 1. Since node 4 now has progenitors it cannot be evolved and Galacticus steps back to node 6, which it then evolves forward in time, until it is promoted to replace node 5. Following this, node 7 is evolved until it reaches node 6. Since node 7 was not the primary progenitor of node 6, it will become a subhalo within node 6 (subhalos are indicated by dot-dashed circles in Fig. 1). This is termed a *node merger event* in Galacticus parlance, and is shown in "Stage 3" of Fig. 1. Nodes 6 and 7 are then evolved in lockstep until they reach node 4. Galacticus is able to handle nested substructure hierarchies, but the current implementation forces there to be a single level of subhalos. Therefore, both nodes 6 and 7 now become subhalos within node 4.

Nodes 4, 6 and 7 now evolve in lockstep until "Stage 4" of Fig. 1 is reached. Here, subhalo node 6 has reached the center of its host node 4 (due to the action of dynamical friction) and so any galaxies residing in nodes 4 and 6 will merge. This is a *satellite merger event* in Galacticus parlance. After this, nodes 4 and 7 evolve in lockstep until node 4 is promoted to replace node 1 as shown in "Stage 5" of Fig. 1.

Finally, Galacticus descends the remaining branch of the merger tree and evolves it forward in time until "Stage 6" of Fig. 1 is reached. Since all nodes are now at the final time, $t_1$, Galacticus stops evolving them and outputs any properties.

It should be noted that, under this algorithm, at any given point in the calculation different branches of the tree may have been evolved to different times (as in "Stage 5" of Fig. 1 where the left-branch has been evolved to $t_1$, but the right-branch has yet to be evolved at all). This is acceptable providing that galaxies in different branches do not interact. If such interactions were included it would be necessary to impose a synchronicity condition on all branches so that no one branch could evolve significantly past any other. By design, Galacticus makes it easy to add in additional time-stepping criteria such as would be needed to implement such an algorithm.

The evolution of nodes forward in time is carried out by Galacticus's ODE solver, which uses GSL's embedded Runge–Kutta-Fehlberg (4,5) method (`gsl_odeiv_step_rkf45`). However, any number of events are allowed to trigger an "interrupt" which causes ODE solving to cease at a specified time. When an interrupt is generated, a function is specified which will be called to handle the interrupt. The function can manipulate the node being evolved as necessary before passing it back to the ODE solver. Node promotion, node merging and satellite merging events all trigger interrupts. Interrupts are also used, for example, to create/destroy components as required.

[1] "Main branch" is defined by starting from the root node of a tree and repeatedly stepping back to the most massive progenitor of the branch. This does not necessarily select the most massive progenitor at a given time.

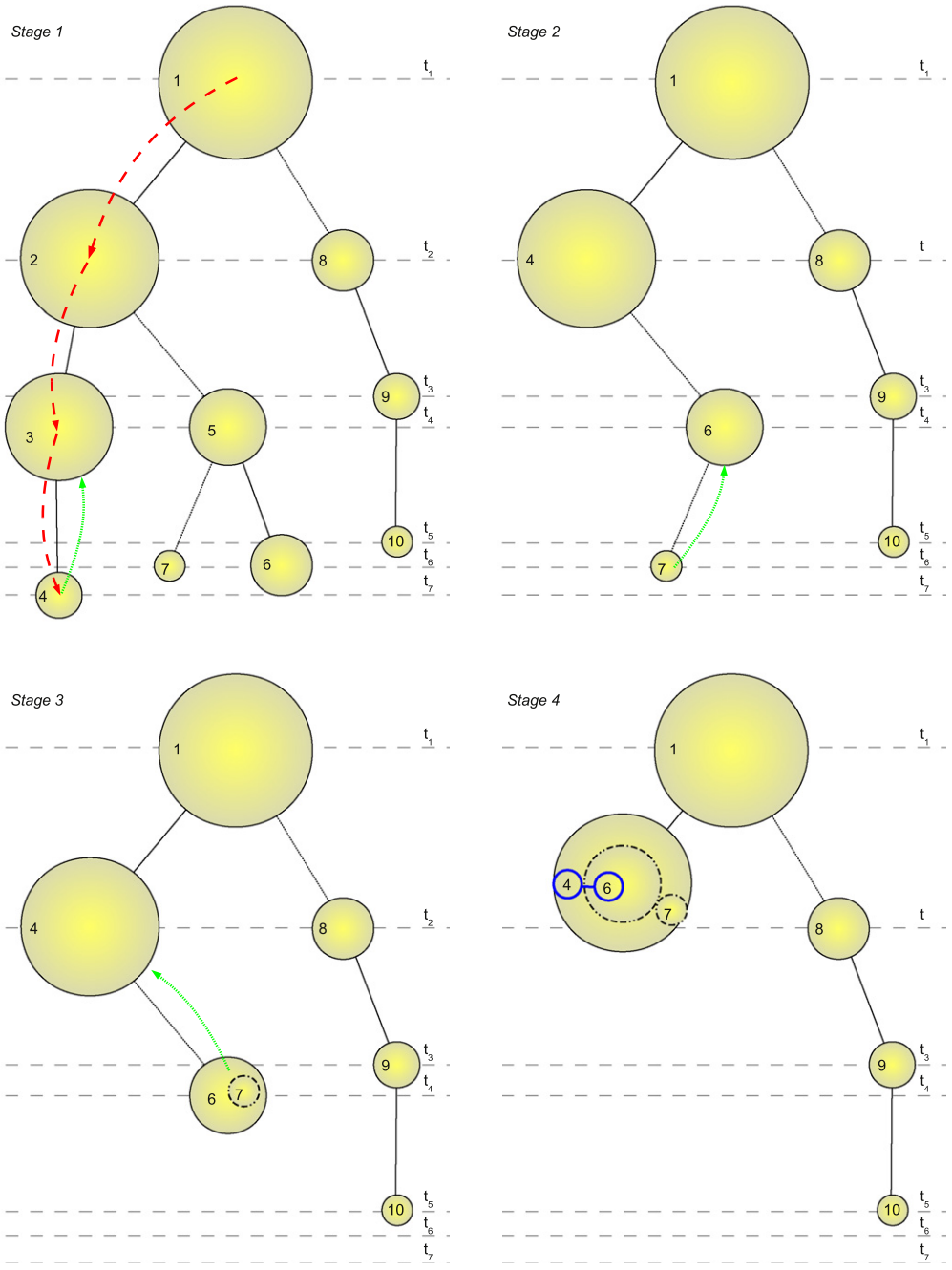


**Fig. 1.** A schematic representation of the evolution of a merger tree within Galacticus. Colored circles represent nodes in the merger tree (with larger circles indicating greater mass). Solid circles are isolated halos, while dashed circles represent subhalos inside a larger halo. Time progresses from bottom to top (i.e. $t_1 > t_2 > t_3 \ldots$). Red dashed arrows indicate Galacticus stepping backward through the merger tree to find a halo which can be evolved. Green dotted arrows indicate a halo being evolved forward in time. Finally, linked blue circles indicate a merger between a subhalo and its host halo and any galaxies that they contain. The evolution of the merger tree is depicted at six key stages, each of which is discussed in the text. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

## 3. Implemented components

In this section we describe the currently implemented node components available within Galacticus. We emphasize that a key feature of Galacticus is the ability to easily add new components, or replace the current components with alternative implementations. In several cases we include "null" implementations which effectively switch off a given component. Throughout, input parameters are identified using the form [`inputParameter`]. For each component we describe the following:

**Properties**. Any variables (e.g. mass, angular momentum, etc.) that describe the component;
**Initialization**. How the component properties are initialized prior to any evolution of the merger tree;
**Differential Evolution**. The equations governing the smooth evolution of the component;
**Event Evolution**. How the component changes in response to one of the following events:

**Node mergers**. Triggered whenever one dark matter halo becomes a subhalo within a large halo;

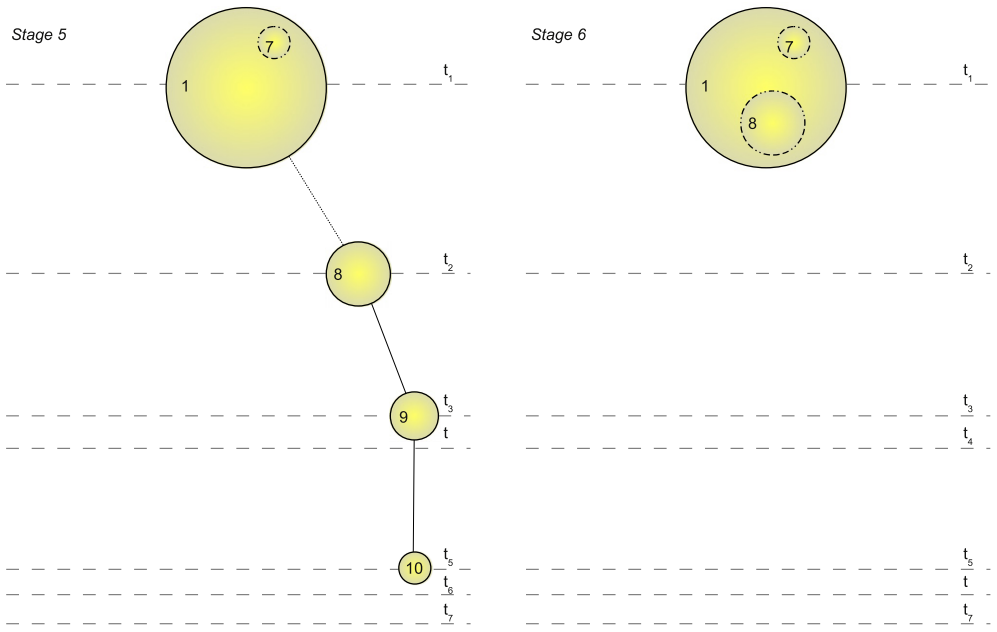


**Fig. 1** (*continued*)

**Satellite merging**. Triggered whenever a subhalo merges with its host halo (or potentially another subhalo);
**Node promotion**. Triggered when the main progenitor of a given node in the merger tree has evolved to the time at which its parent node is defined, and so must be promoted to become that node.

### 3.1. (Supermassive) black hole

#### 3.1.1. "Null" implementation

The null black hole implementation defines the same properties as all other black hole implementations, but implements dummy functions for all black hole properties. It can be used to effectively switch off black holes. Of course, this is not safe if any of the other active components expect to get or set black hole properties (or if they rely on a sensible implementation of black hole evolution).

#### 3.1.2. "Standard" implementation

*Properties:* The standard black hole implementation defines the following properties:

- The mass of the black hole: $M_\bullet$;
- The spin of the black hole, $j_\bullet$.

*Initialization:* Black holes are not initialized, they are created (with a seed mass given by [`blackHoleSeedMass`] and zero spin) as needed.

*Differential Evolution:* The mass and spin evolve as:

$$\dot{M}_\bullet = (1-\epsilon_{\rm radiation})\dot{M}_0, \tag{1}$$

$$\dot{j}_\bullet = \dot{j}(M_\bullet, j_\bullet, \dot{M}_0), \tag{2}$$

where $\dot{M}_0$ is the rest mass accretion rate, $\epsilon_{\rm radiation}$ is the radiative efficiency of the accretion flow feeding the black hole and $\dot{j}(M_\bullet, j_\bullet, \dot{M}_0)$ is the spin-up function of that accretion flow (see Section 4.3). The rest mass accretion rate is computed assuming Bondi–Hoyle–Lyttleton accretion (see, for example, Edgar, 2004) from the spheroid gas reservoir (with an assumed temperature of [`bondiHoyleAccretionTemperatureSpheroid`]) enhanced by a factor of [`bondiHoyleAccretionEnhancementSpheroid`] and from the host halo (with whatever temperature the hot halo temperature profile specifies; see Section 4.11) enhanced by a factor of [`bondiHoyleAccretionEnhancementHotHalo`]. The rest mass accretion rate is removed (as a mass sink; see Section 3.4.2) from the spheroid component. The black hole is assumed to cause feedback in two ways:

**Radio-mode**. Any jet power from the black hole-accretion disk system (see Section 4.3) is included in the hot halo heating rate providing that the halo is in the slow cooling regime (i.e. if the cooling radius is smaller than the virial radius; see, for example, Benson and Bower, 2010a);
**Quasar-mode**. A mechanical wind luminosity of (Ostriker et al., 2010)

$$L_{\rm wind} = \epsilon_{\bullet,{\rm wind}}\dot{M}_0 c^2, \tag{3}$$

where $\epsilon_{\bullet,{\rm wind}}$ = [`blackHoleWindEfficiency`] is the black hole wind efficiency, is added to the gas component of the spheroid (which, presumably, will respond with an outflow for example) if and only if the wind pressure (at the spheroid characteristic radius) is less than the typical thermal pressure in the spheroid gas (Ciotti et al., 2009), i.e.

$$P_{\rm wind} < P_{\rm ISM},$$
$$\frac{1}{2}\rho_{\rm wind}V_{\rm wind}^2 < \frac{3k_{\rm B}T_{\rm ISM}\langle\rho_{\rm ISM}\rangle}{2m_{\rm H}}. \tag{4}$$

Since $\Omega r^2 \rho_{\rm wind} V_{\rm wind}^3 = L_{\rm wind}$ where $\Omega$ is the solid angle of the wind flow, this can be rearranged to give $\langle\rho_{\rm ISM}\rangle > \rho_{\rm wind,critical}$ where

$$\rho_{{\rm wind},critical} = \frac{2m_{\rm H}L_{\rm wind}}{3\Omega r^2 V_{\rm wind} k_{\rm B} T_{\rm ISM}}. \tag{5}$$

This critical wind density is computed at the characteristic radius of the spheroid, $r_{\rm spheroid}$, assuming $V_{\rm wind} = 10^4$ km/s, $T_{\rm ISM} = 10^4$ K and $\Omega = \pi$, and the interstellar medium (ISM) density is approximated by

$$\langle \rho_{\rm ISM} \rangle = \frac{3 M_{\rm gas,spheroid}}{4\pi} r_{\rm spheroid}^{-3}. \quad (6)$$

For numerical ease, the fraction, $f_{\rm wind}$, of the wind luminosity added to the spheroid is adjusted smoothly through the $\rho_{\rm ISM} \approx \rho_{\rm wind,critical}$ region according to

$$f_{\rm wind} = \begin{cases} 0 & \text{if } x < 0, \\ 3x^2 - 2x^3 & \text{if } 0 \leqslant x \leqslant 1, \\ 1 & \text{if } x > 1, \end{cases} \quad (7)$$

where $x = \rho_{\rm ISM}/\rho_{\rm wind,critical} - 1/2$.

*Event Evolution → Node mergers:* None.
*Event Evolution → Satellite merging:* The black holes in the two merging galaxies are instantaneously merged. Properties are computed using the selected black hole binary merger method (see Section 4.24).
*Event Evolution → Node promotion:* None.

### 3.2. Hot halo

#### 3.2.1. "Null" implementation

The null hot halo implements dummy functions for all hot halo properties. It can be used to effectively switch off hot halos. Of course, this is not safe if any of the other active components expect to get or set hot halo properties (or if they rely on a sensible implementation of hot halo evolution).

#### 3.2.2. "Standard" implementation

*Properties:* The standard hot halo implementation defines the following properties:

- The mass of gas which failed to accrete into the hot halo[2]: $M_{\rm failed}$;
- The mass of gas in the hot halo: $M_{\rm hot}$;
- The angular momentum of the gas in the hot halo, $J_{\rm hot}$;
- The mass(es) of heavy elements in gas in the hot halo, $M_{Z,\rm hot}$;
- The mass of gas from outflows in the hot halo: $M_{\rm outflowed}$;
- The angular momentum of the outflowed gas in the hot halo, $J_{\rm outflowed}$;
- The mass(es) of heavy elements in outflowed gas, $M_{Z,\rm outflowed}$,

and the following pipes:

**Energy Input**. Energy sent through this pipe is added to the hot halo and used to offset the cooling rate (see below).
**Cooling Gas**. The net cooling rate of gas mass (and metal content and angular momentum) is sent through this pipe. Any component may claim this pipe and connect to it, allowing it to receive the cooling gas. For example, in the current implementation, the exponential disk component (see Section 3.3.2) claims and connects to this pipe.
**Gas Outflow**. Galactic components that wish to expel gas due to an outflow can send that mass (plus metals and angular momentum) through this pipe, where it will be received into the hot halo component.
**Gas Mass Sink**. Removes gas (and proportionate amounts of angular momentum and elements) from the hot gas halo.

*Initialization:* At initialization, any nodes with no progenitors are assigned a hot halo mass, and failed accreted mass as dictated by the baryonic accretion method (see Section 4.1) and angular momentum based on the accreted mass and the halo spin parameter.

*Differential Evolution:* In the standard hot halo implementation the hot gas mass and heavy element mass(es) evolves as:

$$\dot{M}_{\rm failed} = \dot{M}_{\rm failed\ accretion}, \quad (8)$$

$$\dot{M}_{\rm hot} = \dot{M}_{\rm accretion} - \dot{M}_{\rm cooling} + \dot{M}_{\rm outflow,return}, \quad (9)$$

$$\dot{M}_{Z,\rm hot} = -\dot{M}_{\rm cooling} \frac{M_{Z,\rm hot}}{M_{\rm hot}} + \dot{M}_{Z,\rm outflow,return}, \quad (10)$$

where $\dot{M}_{\rm accretion}$ is the rate of growth of the hot component due to accretion from the intergalactic medium (IGM), $\dot{M}_{\rm failed\ accretion}$ is the rate of failed accretion from the intergalactic medium (IGM) (these may include a component due to transfer of mass from the failed to hot reservoirs) and $\dot{M}_{\rm cooling}$ is the rate of mass loss from the hot halo due to cooling (see Section 4.5.2) minus any heating rate defined as

$$\dot{M}_{\rm heating} = \dot{E}{\rm input}/V_{\rm virial}^2, \quad (11)$$

where $\dot{E}$ is the rate at which energy is being sent through the "energy input" pipe and $V_{\rm virial}$ is the virial velocity of the halo.[3] The angular momentum of the hot gas evolves as:

$$\dot{J}_{\rm hot} = \dot{M}_{\rm accretion} \frac{\dot{J}_{\rm node}}{\dot{M}_{\rm node}} - \dot{M}_{\rm cooling} r_{\rm cool} V_{\rm rotate} + \dot{J}_{\rm outflow,return}, \quad (12)$$

where $\dot{M}_{\rm node}$ and $\dot{J}_{\rm node}$ are defined in Section 3.5, $r_{\rm cool}$ is the cooling radius (see Section 4.5.3) and $V_{\rm rotate}$ is the effective rotation speed of the halo for its angular momentum (see Section 4.6.1). For the outflowed components:

$$\dot{M}_{\rm outflowed} = -\dot{M}_{\rm outflow,return} + \dot{M}_{\rm outflows}, \quad (13)$$

$$\dot{M}_{Z,\rm outflowed} = -\dot{M}_{Z,\rm outflow,return} + \dot{M}_{Z,\rm outflows} \quad (14)$$

and:

$$\dot{J}_{\rm outflowed} = -\dot{J}_{\rm outflow,return} + \dot{J}_{\rm outflows}. \quad (15)$$

In the above

$$\dot{M}|\dot{M}_Z|\dot{J}_{\rm outflow,return} = \alpha_{\rm outflow\ return\ rate} \frac{M|M_Z|J_{\rm outflowed}}{\tau_{\rm dynamical,halo}}, \quad (16)$$

where $\alpha_{\rm outflow\ return\ rate}$ = [`hotHaloOutflowReturnRate`] is an input parameter controlling the rate at which gas flows from the outflowed to hot reservoirs, and $\dot{M}|\dot{M}_Z|\dot{J}_{\rm outflows}$ are the net rates of outflow from any components in the node.

*Event Evolution → Node mergers:* If the [`starveSatellites`] parameter is true, then any hot halo properties of the minor node are added to those of the major node and the hot halo component is removed from the minor node. Additionally in this case, any material outflowed from the satellite galaxy to its hot halo is transferred to the hot halo of the host dark matter halo after each timestep.
*Event Evolution → Satellite merging:* If the [`starveSatellites`] parameter is false, then any hot halo properties of the satellite node are added to those of the host node and the hot halo component is removed from the satellite node.
*Event Evolution → Node promotion:* Any hot halo properties of the parent node are added to those of the node prior to promotion.

[2] By "failed to accrete" we mean any mass which would accrete into the halo in the absence of baryonic physics, such as pressure and heating. This provides a way to accrete this mass later if, for example, the dark matter halo potential well deepens and becomes more effective at accreting mass.

[3] The net cooling rate is never allowed to drop below zero. If the mass heating rate exceeds the mass cooling rate then the excess energy is not used.

### 3.3. Galactic disk

#### 3.3.1. "Null" implementation

The null disk implementation implements dummy functions for all disk properties. It can be used to effectively switch off disks. Of course, this is not safe if any of the other active components expect to get or set disk properties (or if they rely on a sensible implementation of disk evolution).

#### 3.3.2. "Exponential" implementation

This implementation assumes a disk with an exponential surface density profile in which stars trace gas.

*Properties:* The exponential galactic disk implementation defines the following properties:

- the mass of gas in the disk: $M_{\rm disk,gas}$;
- the masses of elements in the gaseous disk: $M_{Z,\rm disk,gas}$;
- the mass of stars in the disk: $M_{\rm disk,stars}$;
- the masses of elements in the stellar disk: $M_{Z,\rm disk,stars}$;
- the luminosities (in multiple bands) of the stellar disk: $L_{\rm disk,stars}$;
- the angular momentum of the disk, $J_{\rm disk}$;
- the radial scale length of the disk, $R_{\rm disk}$;
- the circular velocity of the disk at $R_{\rm disk}$, $V_{\rm disk}$.

*Initialization:* No initialization is performed—disks are created as needed.

*Differential Evolution:* In the exponential galactic disk implementation the gas mass evolves as:

$$\dot{M}_{\rm disk,gas} = \dot{M}_{\rm cooling} - \dot{M}_{\rm outflow,disk} - \dot{M}_{\rm stars,disk} - M_{\rm disk,gas}/\tau_{\rm bar}, \quad (17)$$

where the rate of change of stellar mass is

$$\dot{M}_{\rm disk,stars} = \Psi - \dot{R} - M_{\rm disk,stars}/\tau_{\rm bar}, \quad (18)$$

with

$$\Psi = \frac{M_{\rm disk,gas}}{\tau_{\rm disk,star\ formation}}, \quad (19)$$

where $\tau_{\rm disk,star\ formation}$ is the star formation timescale (see Section 4.17), $\dot{R}$ is the rate of mass recycling from stars and $\tau_{\rm bar}$ is a bar instability timescale (see Section 4.7). The mass removed from the disk by the bar instability mechanism is added to the active spheroid component. Element abundances (including total metals) evolve according to:

$$\dot{M}_{Z,\rm disk,gas} = \dot{M}_{Z,\rm cooling} - \dot{M}_{Z,\rm outflow,disk} - \dot{M}_{Z,\rm stars,disk} + \dot{y} \quad (20)$$

and

$$\dot{M}_{Z,\rm stars,disk} = \Psi \frac{M_{Z,\rm disk,gas}}{M_{\rm disk,gas}} - \dot{R}_Z, \quad (21)$$

where $\dot{y}$ is the rate of element yield from stars and $\dot{R}_Z$ is the rate of element recycling. Recycling rates and yields are discussed in Section 4.12 and Section 4.18. The angular momentum evolves as:

$$\dot{J}_{\rm disk} = \dot{J}_{cooling} - \left[\dot{M}_{\rm outflow,disk} + \frac{M_{\rm disk,gas} + M_{\rm disk,stars}}{\tau_{\rm bar}}\right] \times \frac{J_{\rm disk}}{M_{\rm disk,gas} + M_{\rm disk,stars}}. \quad (22)$$

The outflow rate, $\dot{M}_{\rm outflow,disk}$, is computed for the current star formation rate and gas properties by the stellar properties subsystem (see Section 4.18).

*Event Evolution → Node mergers:* None.

*Event Evolution → Satellite merging:* Disks may be destroyed (or, potentially, created or otherwise modified) as the result of a satellite merging event, as dictated by the selected merger remnant mass movement method (see Section 4.9.1).

*Event Evolution → Node promotion:* None.

### 3.4. Galactic spheroid

#### 3.4.1. "Null" implementation

The null spheroid implements dummy functions for all spheroid properties. It can be used to effectively switch off spheroids. Of course, this is not safe if any of the other active components expect to get or set spheroid properties (or if they rely on a sensible implementation of spheroid evolution).

#### 3.4.2. "Hernquist" implementation

This implementation assumes a Hernquist profile (Hernquist, 1990) for the spheroidal component of a galaxy in which stars trace gas.

*Properties:* The Hernquist galactic spheroid implementation defines the following properties:

- the mass of gas in the spheroid: $M_{\rm spheroid,\ gas}$;
- the masses of elements in the gaseous spheroid: $M_{Z,\rm spheroid,\ gas}$;
- the mass of stars in the spheroid: $M_{\rm spheroid,\ stars}$;
- the masses of elements in the stellar spheroid: $M_{Z,\rm spheroid,\ stars}$;
- the luminosities (in multiple bands) of the stellar spheroid: $L_{\rm spheroid,stars}$;
- the pseudo-angular momentum[4] of the spheroid, $J_{\rm spheroid}$;
- the radial scale length of the spheroid, $r_{\rm spheroid}$;
- the circular velocity of the spheroid at $r_{\rm spheroid}$, $V_{\rm spheroid}$.and the following pipes:

**Energy Input**. Energy sent through this pipe is added to the gas of the spheroid and will result in an outflow (see below).

**Gas Mass Sink**. Removes gas (and proportionate amounts of angular momentum and elements) from the spheroid gas.

*Initialization:* No initialization is performed—spheroids are created as needed.

*Differential Evolution:* In the Hernquist galactic spheroid implementation the gas mass evolves as[5]:

$$\dot{M}_{\rm spheroid,gas} = -\dot{M}_{\rm outflow,spheroid} - \dot{M}_{\rm stars,spheroid}, \quad (23)$$

where the rate of change of stellar mass is

$$\dot{M}_{\rm stars,spheroid} = \Psi - \dot{R}, \quad (24)$$

with

$$\Psi = \frac{M_{\rm spheroid,gas}}{\tau_{\rm spheroid,star\ formation}}, \quad (25)$$

with $\tau_{\rm spheroid,star\ formation}$ being the star formation timescale (see Section 4.17) and $\dot{R}$ is the rate of mass recycling from stars. Element abundances (including total metals) evolve according to:

$$\dot{M}_{Z,\rm spheroid,gas} = -\dot{M}_{Z,\rm outflow,spheroid} - \dot{M}_{Z,\rm stars,spheroid} + \dot{y} \quad (26)$$

and

[4] Effectively the angular momentum that the spheroid would have, were it rotationally supported rather than pressure supported.

[5] There may be an additional contribution to the mass and angular momentum rates of change in the spheroid due to material transferred from the disk component via the bar instability mechanism (see Section 3.3.2). This is not included here as it is not intrinsic to this specific spheroid implementation—it is handled explicitly by the disk component and so applies equally to any spheroid component implementation.

$$\dot{M}_{Z,\mathrm{stars,spheroid}} = \Psi \frac{M_{Z,\mathrm{spheroid,gas}}}{M_{\mathrm{spheroid,gas}}} - \dot{R}_Z, \tag{27}$$

where $\dot{y}$ is the rate of element yield from stars and $\dot{R}_Z$ is the rate of element recycling. Recycling rates and yields are discussed in Section 4.12 and Section 4.18. The angular momentum evolves as:

$$\dot{J}_{\mathrm{spheroid}} = \dot{M}_{\mathrm{outflow,spheroid}} \frac{J_{\mathrm{spheroid}}}{M_{\mathrm{spheroid,gas}} + M_{\mathrm{spheroid,stars}}}. \tag{28}$$

The outflow rate, $\dot{M}_{\mathrm{outflow,disk}}$, is computed for the current star formation rate and gas properties by the stellar properties subsystem (see Section 4.18), with an additional contribution given by

$$\dot{M}_{\mathrm{outflow,spheroid}} = \beta_{\mathrm{spheroid,energy}} \frac{\dot{E}_{\mathrm{gas,spheroid}}}{V^2_{\mathrm{spheroid}}}, \tag{29}$$

where $\beta_{\mathrm{spheroid,energy}}$ is set by the [`spheroidEnergeticOutflowMassRate`] input parameter, and $\dot{E}_{\mathrm{gas,spheroid}}$ is any input energy sent through the "Energy Input" pipe.

*Event Evolution → Node mergers:* None.
*Event Evolution → Satellite merging:* Spheroids may be created as the result of a satellite merging event, as dictated by the selected merger remnant mass movement method (see Section 4.9.1).
*Event Evolution → Node promotion:* None.

### 3.5. Basic properties

Basic properties are the total mass of a node and the cosmic time at which it currently exists.

#### 3.5.1. "Simple" implementation

*Properties:* The simple basic properties implementation defines the following properties:

- The total mass of the node: $M_{\mathrm{node}}$;
- The time at which the node is defined: $t_{\mathrm{node}}$;
- The time at which the node was last an isolated halo (i.e. not a subhalo).

*Initialization:* All basic properties are required to be initialized by the merger tree construction routine (see Section 4.14).
*Differential Evolution:* Properties are evolved according to:

$$\dot{M}_{\mathrm{node}} = \begin{cases} \frac{M_{\mathrm{node,parent}} - M_{\mathrm{node}}}{t_{\mathrm{node,parent}} - t_{\mathrm{node}}} & \text{if primary progenitor} \\ 0 & \text{otherwise,} \end{cases} \tag{30}$$

$$\dot{t}_{\mathrm{node}} = 1, \tag{31}$$

where the "parent" subscript indicates a property of the parent node in the merger tree.

*Event Evolution → Node mergers:* None.
*Event Evolution → Satellite merging:* None.
*Event Evolution → Node promotion:* $M_{\mathrm{node}}$ is updated to the node mass of the parent prior to promotion.

### 3.6. Satellite node orbit

This component tracks the orbital properties of subhalos.

#### 3.6.1. "Simple" implementation

*Properties:* The simple satellite orbit implementation defines the following properties:

- The time until the satellite will merge with its host: $t_{\mathrm{satellite,merge}}$.

*Initialization:* None.
*Differential Evolution:* Properties are evolved according to:

$$\dot{t}_{\mathrm{satellite,merge}} = -1. \tag{32}$$

*Event Evolution → Node mergers:* The component is created and the time to merging is assigned a value (see Section 4.22.1).
*Event Evolution → Satellite merging:* None.
*Event Evolution → Node promotion:* Not applicable (component only exists for satellite nodes).

### 3.7. Dark matter halo spin

This component stores and tracks the spin parameters of dark matter halos.

#### 3.7.1. "Null" implementation

The null spin component implements dummy functions for all spin properties. It can be used to effectively switch off spins. Of course, this is not safe if any of the other active components expect to get or set spin properties (or if they rely on a sensible implementation of spin evolution).

#### 3.7.2. "Random" implementation

*Properties:* The random dark matter halo spin implementation defines the following properties:

- the spin parameter of the halo: $\lambda$.

*Initialization:* The spin parameter of each node, if not already assigned, is selected at random from a distribution of spin parameters. This value is assigned to the earliest progenitor of the halo traced along its primary branch. The value is then propagated forward along the primary branch until the node mass exceeds that of the node for which the spin was selected by a factor of [`randomSpinResetMassFactor`], at which point a new spin is selected at random, and the process repeated until the end of the branch is reached, in a manner similar to the algorithm used by Cole et al. (2000).
*Differential Evolution:* The spin parameter does not evolve.

*Event Evolution → Node mergers:* None
*Event Evolution → Satellite merging:* None.
*Event Evolution → Node promotion:* The spin is updated to equal that of the parent node. (The two will differ only if this is a case where the new halo node was sufficiently more massive than the node for which a spin was last selected that a new spin value was chosen.)

### 3.8. Dark matter profile

This component stores dynamic properties associated with dark matter halo density profiles.

#### 3.8.1. "Null" implementation

The null profile implements dummy functions for all profile properties. It can be used to effectively switch off profiles. Of course, this is not safe if any of the other active components expect to get or set profile properties (or if they rely on a sensible implementation of profile evolution).

#### 3.8.2. "Scale" implementation

*Properties:* The scale dark matter profile implementation defines the following properties:

- the scale length of the density profile.

*Initialization:* The scale length of each node, if not already assigned, is assigned using the concentration parameter function (see Section 4.6.2), but is not allowed to drop below [darkMatterProfileMinimumConcentration], such that the scale length is equal to the virial radius divided by that concentration. The value is propagated in both directions along the primary child branch from the node.

*Differential Evolution:* The scale radius does not evolve.

*Event Evolution → Node mergers:* None.
*Event Evolution → Satellite merging:* None.
*Event Evolution → Node promotion:* None.

## 4. Specific physical implementation

In this section we describe specific implementations of the numerous physical processes that are currently available in Galacticus. In each subsection we briefly describe what physical process/property is being discussed and then describe the specific implementations available within Galacticus.

### 4.1. Accretion of gas into halos

The accretion rate of gas from the intergalactic medium (IGM) into a dark matter halo is expected to depend on (at least) the rate at which that halo's mass is growing, the depth of its potential well and the thermodynamical properties of the accreting gas. Galacticus implements the following calculations of gas accretion from the intergalactic medium (IGM).

*Simple Method:* Currently the only option, this method sets the accretion rate of baryons into a halo to be:

$$\dot{M}_{\rm accretion} = \begin{cases} \frac{\Omega_{\rm b}}{\Omega_0}\dot{M}_{\rm halo} & \text{if } V_{\rm virial} > V_{\rm reionization}, \\ & \text{or } z > z_{\rm reionization}, \\ 0 & \text{otherwise}, \end{cases} \quad (33)$$

where $\dot{M}_{\rm halo}$ is the total rate of growth of the node mass, $z_{\rm reionization}$ = [reionizationSuppressionRedshift] is the redshift at which the Universe is reionized and $V_{\rm reionization}$ = [reionizationSuppressionVelocity] is the virial velocity below which accretion is suppressed after reionization. Setting $V_{\rm reionization}$ to zero will effectively switch off the effects of reionization on the accretion of baryons. This algorithm attempts to offer a simple prescription for the effects of reionization and has been explored by multiple authors (e.g. Benson et al., 2002). In particular, Cooper et al. (2011) show that it produces results in good agreement with more elaborate treatments of reionization. For halos below the accretion threshold, any accretion rate that would have otherwise occurred is instead placed into the "failed" accretion rate. For halos which can accrete, and which have some mass in their "failed" reservoir, that mass will be added to the regular accretion rate at a rate equal to the mass of the "failed" reservoir times the specific growth rate of the halo.

### 4.2. Background cosmology

The background cosmology describes the evolution of an isotropic, homogeneous Universe within which galaxy formation calculations are carried out. For the purposes of Galacticus, the background cosmology is used to relate expansion factor/redshift to cosmic time and to compute the density of various components (e.g. dark matter, dark energy, etc.) at different epochs.

*Matter + Lambda:* In this implementation, cosmological relations are computed assuming a universe that contains only collisionless matter and a cosmological constant.

### 4.3. Circumnuclear accretion disks

Circumnuclear accretion disks surrounding supermassive black holes at the centers of galaxies influence the evolution of both the black hole (via accretion rates of mass and angular momentum and possibly by extracting rotational energy from the black hole) and the surrounding galaxy if they lead to energetic outflows (e.g. jets) from the nuclear region. Current implementations of accretion disks are as follows and are selected via [accretionDisksMethod].

*Shakura–Sunyaev Geometrically Thin, Radiatively Efficient Disks:* This implementation assumes that accretion disks are always described by a radiatively efficient, geometrically thin accretion disk as described by Shakura and Sunyaev (1973). The radiative efficiency of the flow is computed assuming that material falls into the black hole without further energy loss from the innermost stable circular orbit (ISCO), while the spin-up rate of the black hole is computed assuming that the material enters the black hole with the specific angular momentum of the innermost stable circular orbit (ISCO) (i.e. there are no torques on the material once it begins to fall in from the innermost stable circular orbit (ISCO); Bardeen, 1970). For these thin disks, jet power is computed, using the expressions from Meier (2001) (his Eqs. (4) and (5)).

*Advection Dominated, Geometrically Thick, Radiatively Inefficient Flows (ADAFs):* This implementation assumes that accretion is via an advection dominated accretion flow (Narayan and Yi, 1994) which is radiatively inefficient and geometrically thick. The radiative efficiency of the flow, which will be zero for a pure ADAF, can be set via the input parameter [adafRadiativeEfficiency]. The spin up rate of the black hole and the jet power produced as material accretes into the black hole are computed using the method of Benson and Babul (2009). The energy of the accreted material can be set equal to the energy at infinity (as expected for a pure ADAF) or the energy at the innermost stable circular orbit (ISCO) by use of the [adafEnergyOption] parameter. The ADAF structure is controlled by the adiabatic index, $\gamma$, and viscosity parameter, $\alpha$, which are specified via the [adafAdiabaticIndex] and [adafViscosityOption] input parameters respectively. [adafViscosityOption] may be set to "fit", in which case the fitting function for $\alpha$ as a function of black hole spin from Benson and Babul (2009) will be used.

*"Switched" Disks*: This method allows for accretion disks to switch between radiatively efficient (Shakura–Sunyaev) and inefficient (ADAF) modes. Which mode is used is determine by the accretion rate onto the disk:

- Radiatively efficient accretion if $\dot{M}/\dot{M}_{\rm Eddington} >$ [accretionRateThinDiskMinimum] and $\dot{M}/\dot{M}_{\rm Eddington} <$ [accretionRateThinDiskMaximum];
- Radiatively inefficient accretion otherwise.

### 4.4. Cold dark matter structure formation

A variety of functions are used to describe structure formation in cold dark matter dominated universes. These are described below.

#### 4.4.1. Primordial power spectrum

The functional form of the primordial dark matter power spectrum. The power spectrum is computed from the specified primordial power spectrum and the transfer function (see Section 4.4.2) and normalized to a value of $\sigma_8$ specified by [sigma_8].

(*Running*)*Power Law Spectrum:* This method implements a primordial power spectrum of the form:

$$P(k) \propto k^{n_{\rm eff}(k)}, \tag{34}$$

where

$$n_{\rm eff}(k) = n_{\rm s} + \frac{{\rm d}n}{{\rm d}\ln k}\frac{k}{k_{\rm ref}}, \tag{35}$$

where $n_{\rm s}$ = [powerSpectrumIndex] is the power spectrum index at wavenumber $k_{\rm ref}$ = [powerSpectrumReferenceWavenumber] and ${\rm d}n/{\rm d}\ln k$ = [powerSpectrumRunning] describes the running of this index with wavenumber.

#### 4.4.2. Transfer function

The functional form of the cold dark matter transfer function is selected by [transferFunctionMethod]. The power spectrum is computed from the specified transfer function and the primordial power spectrum (see Section 4.4.1) and normalized to a value of $\sigma_8$ specified by [sigma_8].

*BBKS:* This method uses the fitting function of Bardeen et al. (1986) to compute the cold dark matter (CDM) transfer function.

*Eisenstein& Hu:* This method uses the fitting function of Eisenstein and Hu (1999) to compute the cold dark matter (CDM) transfer function. It requires that the effective number of neutrino species be specified via the [effectiveNumberNeutrinos] parameter and summed mass of all neutrino species (in eV) be specified via the [summedNeutrinoMasses] parameter.

CMBFast: This method uses the CMBFast code to compute the cold dark matter (CDM) transfer function. It requires that the mass fraction of helium in the early Universe be specified via the [Y_He] parameter. CMBFast will be downloaded and run if the transfer function needs to be computed. The transfer function will then be stored in a file for future reference. An example of a transfer function computed in this way is shown in Fig. 2.

*File:* This method reads a tabulated transfer function from an XML file, interpolating between tabulated points.

#### 4.4.3. Linear growth function

The function describing the amplitude of linear perturbations.

*Simple:* This method calculates the growth of linear perturbations using standard perturbation theory in a Universe consisting of collisionless matter and a cosmological constant.

#### 4.4.4. Critical overdensity

The method used to compute the critical linear overdensity at which overdense regions virialize.

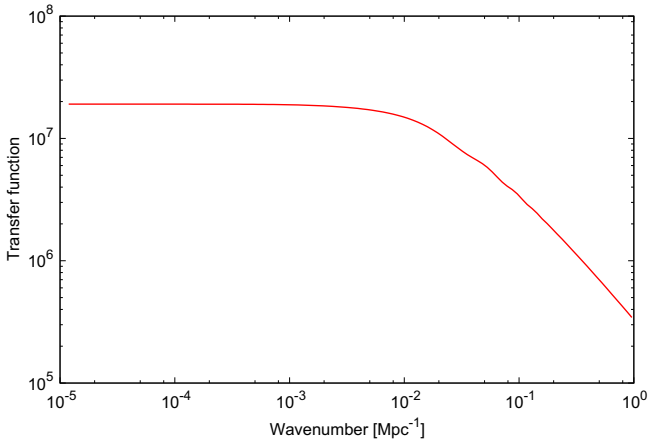


**Fig. 2.** The transfer function, $T(k)$, computed automatically by Galacticus by downloading, compiling and running CMBFast. The transfer function is computed for a cosmological model with $\Omega_{\rm M} = 0.25$, $\Omega_\Lambda = 0.75$, $\Omega_{\rm b} = 0.045$ and $H_0 = 75$ km/s/Mpc.

*Spherical Collapse (Matter + Cosmological Constant):* This method calculates critical overdensity using a spherical top-hat collapse model assuming a Universe which contains collisionless matter and a cosmological constant (see, for example, Percival (2005)).

#### 4.4.5. Virial density contrast

The method used to compute the mean density contrast of virialized dark matter halos is specified via [virialDensityContrastMethod].

*Bryan & Norman Fitting Function:* This method calculates virial density contrast using the fitting functions given by Bryan and Norman (1998). As such, it is valid only for $\Omega_\Lambda = 0$ or $\Omega_M + \Omega_\Lambda = 1$ cosmologies and will abort on other cosmologies.

*Spherical Collapse (Matter + Cosmological Constant):* This method calculates virial density contrast using a spherical top-hat collapse model assuming a Universe which contains collisionless matter and a cosmological constant (see, for example, Percival (2005)).

#### 4.4.6. Halo mass function

The dark matter halo mass function (i.e. the number of halos per unit volume per unit mass interval). Which mass function to use is specified by [haloMassFunctionMethod].

*Press-Schechter:* This method uses the functional form proposed by Press and Schechter (1974) to compute the halo mass function.

*Sheth-Tormen:* This method uses the functional form proposed by Sheth et al. (2001) to compute the halo mass function.

*Tinker:* This method uses the functional form proposed by Tinker et al. (2010) to compute the halo mass function. The mass function is computed at the appropriate virial overdensity (see Section 4.4.5).

### 4.5. Cooling of gas inside halos

The cooling of gas within dark matter halos is controlled by a number of different algorithms which will be described below.

#### 4.5.1. Cooling function

The cooling function of gas, $\Lambda(\rho, T, \mathbf{Z})$, where $\rho$ is gas density, $T$ is temperature and $\mathbf{Z}$ is a vector of elemental abundances, is selected by [coolingFunctionMethod]. Multiple such methods may be specified and are cumulative (i.e. the net cooling function is the sum over all specified cooling functions).

*Atomic Collisional Ionization Equilibrium Using* Cloudy: This method computes the cooling function using the Cloudy code and under the assumption of collisional ionization equilibrium with no molecular contribution. Abundances are Solar, except for zero metallicity calculations which use Cloudy's "primordial" metallicity. The helium abundance for non-zero metallicity is scaled linearly with metallicity between primordial and Solar values. The Cloudy code will be downloaded and run to compute the cooling function as needed, which will then be stored for future use. As this process is slow, a precomputed table is provided with Galacticus. If metallicities outside the range tabulated in this file are required it will be regenerated with an appropriate range. Fig. 3 shows an example of a cooling function computed automatically by Galacticus using Cloudy,

*Collisional Ionization Equilibrium From File:* This method the cooling function is read from a file. The cooling function is assumed to be computed under conditions of collisional ionization equilibrium and therefore to scale as $\rho^2$.

*CMB Compton Cooling:* This method computes the cooling function due to Compton scattering off of cosmic microwave background (CMB) photons:

$$\Lambda = \frac{4\sigma_{\rm T} a k_{\rm B} n_{\rm e}}{m_{\rm e} c} T_{\rm CMB}^4 (T - T_{\rm CMB}), \tag{36}$$

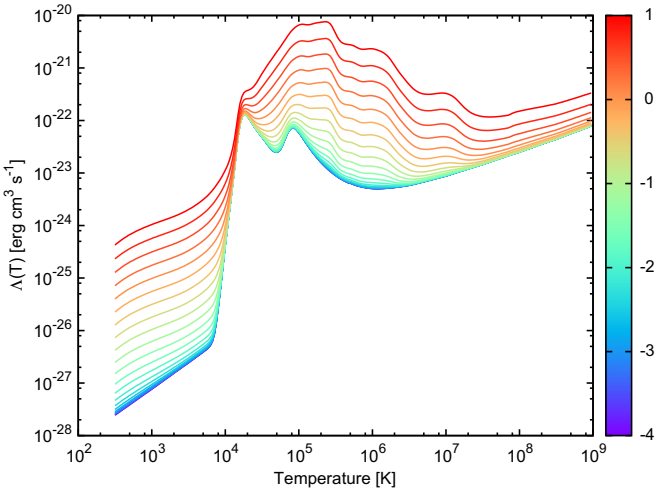


**Fig. 3.** The cooling function for atomic gas in collisional ionization equilibrium computed using Cloudy 08.00 (automatically downloaded, compiled and run by Galacticus) as a function of gas temperature. The color scale indicates the gas metallicity. Solar abundance ratios are assumed, except for zero metallicity calculations which use Cloudy's "primordial" metallicity.

where $\sigma_T$ is the Thompson cross-section, $a$ is the radiation constant, $k_B$ is Boltzmann's constant, $n_e$ is the number density of electrons, $m_e$ is the electron mass, c is the speed of light, $T_{CMB}$ is the cosmic microwave background (CMB) temperature at the current cosmic epoch and $T$ is the temperature of the gas. The electron density is computed from the selected ionization state method (see Section 4.13).

#### 4.5.2. Cooling rate

The algorithm used to compute the rate at which gas drops out of the hot halo due to cooling.

*White & Frenk:* This method computes the cooling rate using the expression given by White and Frenk (1991), namely

$$\dot{M}_{\rm cool} = 4\pi r_{\rm cool}^2 \rho(r_{\rm cool}) \dot{r}_{\rm cool}, \tag{37}$$

where $r_{\rm cool}$ is the cooling radius (see Section 4.5.3) in the hot halo and $\rho(r)$ is the density profile of the hot halo (see Section 4.10).

#### 4.5.3. Cooling radius

The algorithm used to compute the cooling radius in hot halo gas.

*Simple:* This method computes the cooling radius by seeking the radius at which the time available for cooling (see Section 4.5.5) equals the cooling time (see Section 4.5.4). The growth rate is determined consistently based on the slope of the density profile, the density dependence of the cooling function and the rate at which the time available for cooling is increasing. This method assumes that the cooling time is a monotonic function of radius.

#### 4.5.4. Cooling time

The algorithm used to compute the time taken for gas to cool (i.e. the cooling time).

*Simple:* This method assumes that the cooling time is simply

$$t_{\rm cool} = \frac{N}{2} \frac{k_B T n_{\rm tot}}{\Lambda}, \tag{38}$$

where $N$ = [`coolingTimeSimpleDegreesOfFreedom`] is the number of degrees of freedom in the cooling gas which has temperature $T$ and total particle number density $n_{\rm tot}$ and $\Lambda$ is the cooling function (see Section 4.5.1).

#### 4.5.5. Time available for cooling

The method used to determine the time available for cooling (i.e. the time for which gas in a halo has been able to cool).

*White & Frenk:* This method assumes that the time available for cooling is equal to

$$t_{\rm available} = \exp[f \ln t_{\rm Universe} + (1-f) \ln t_{\rm dynamical}], \tag{39}$$

where $f$ = [`coolingTimeAvailableAgeFactor`] is an interpolating factor, $t_{\rm Universe}$ is the age of the Universe and $t_{\rm dynamical}$ is the dynamical time in the halo. The original White and Frenk (1991) algorithm corresponds to $f = 1$.

### 4.6. Dark matter halos

Several algorithms are used to implement dark matter halos. These are described below.

#### 4.6.1. Density profile

The method used to compute density profiles of dark matter halos is selected via the [`darkMatterProfileMethod`] parameter.

*Isothermal:* Under this method the density profile is given by:

$$\rho_{\rm node}(r) \propto r^{-2}, \tag{40}$$

normalized such that the total mass of the node is enclosed with the virial radius.

*NFW:* Under this method the Navarro–Frenk–White (NFW) density profile (Navarro et al., 1997) is used

$$\rho_{\rm node}(r) \propto \left(\frac{r}{r_s}\right)^{-1} \left[1+\frac{r}{r_s}\right]^{-2}, \tag{41}$$

normalized such that the total mass of the node is enclosed with the virial radius and with scale length $r_s = r_{\rm virial}/c$ where $c$ is the halo concentration (see Section 4.6.2).

#### 4.6.2. Dark matter density profile concentration

The method used to compute the concentration of dark matter halos.

*Gao2008:* Under this method the concentration is computed using a fitting function from Gao et al. (2008):

$$\log_{10} c = A \log_{10} M_{\rm halo} + B. \tag{42}$$

The parameters are a function of expansion factor, $a$. We use the following fits to the Gao et al. (2008) results:

$$A = -0.14 \exp\left[-\left(\frac{\log_{10} a + 0.05}{0.35}\right)^2\right], \tag{43}$$

$$B = 2.646 \exp\left[-\left(\frac{\log_{10} a}{0.50}\right)^2\right]. \tag{44}$$

#### 4.6.3. Spin parameter distribution

The method used to compute the distribution of dark matter halo spin parameters is selected via the [`haloSpinDistributionMethod`] parameter.

*Lognormal:* Under this method the spin is drawn from a lognormal distribution.

*Bett2007:* Under this method the spin is drawn from the distribution found by Bett et al. (2007). The $\lambda_0$ and $\alpha$ parameter of Bett et al.'s distribution are set by the [spinDistributionBett2007Lambda0] and [spinDistributionBett2007Alpha] input parameters.

### 4.7. Disk stability/bar formation

The method uses to compute the bar instability timescale for galactic disks is selected by [barInstabilityMethod].

*Efstathiou, Lake*& Negroponte: This method uses the stability criterion of Efstathiou et al. (1982) to estimate when disks are unstable to bar formation:

$$\epsilon\left(\equiv\frac{V_{\rm peak}}{\sqrt{GM_{\rm disk}/r_{\rm disk}}}\right)<\epsilon_{\rm c} \tag{45}$$

for stability, where $V_{\rm peak}$ is the peak velocity in the rotation curve (computed here assuming an isolated exponential disk), $M_{\rm disk}$ is the mass of the disk and $r_{\rm disk}$ is its scale length (assuming an exponential disk). The value of $\epsilon_{\rm c}$ is linearly interpolated in the disk gas fraction between values for purely stellar and gaseous disks as specified by [stabilityThresholdStellar] and [stabilityThresholdGaseous] respectively. For disks which are judged to be unstable, the timescale for bar formation is taken to be

$$t_{\rm bar}=t_{\rm disk}\frac{\epsilon_{\rm c}-\epsilon_{\rm iso}}{\epsilon_{\rm c}-\epsilon}, \tag{46}$$

where $\epsilon_{\rm iso}$ is the value of $\epsilon$ for an isolated disk and $t_{\rm disk}$ is the disk dynamical time, defined as $r/V$ (where $V$ is the circular velocity) at one scale length. This form gives an infinite timescale at the stability threshold, reducing to a dynamical time for highly unstable disks.

### 4.8. Galactic structure

The algorithm to be used when solving for galactic structure (specifically, finding radii of galactic components) is specified via [galacticStructureRadiusSolverMethod].

*Simple:* This method determines the sizes of galactic components by assuming that their self-gravity is negligible (i.e. that the gravitational potential well is dominated by dark matter) and that, therefore, baryons do not modify the dark matter density profile. The radius of a given component is then found by solving

$$j=\sqrt{GM_{\rm DM}(r)r}, \tag{47}$$

where $j$ is the specific angular momentum of the component (at whatever point in the profile is to be solved for), $r$ is radius and $M(r)$ is the mass of dark matter within radius $r$.

*Adiabatic:* This method takes into account the baryonic self-gravity of all galactic components when solving for structure and additionally accounts for back reaction of the baryons on the dark matter density profile using the adiabatic contraction algorithm of Gnedin et al. (2004). The parameters $A$ and $\omega$ of that model are specified via input parameters [adiabaticContractionGnedinA] and [adiabaticContractionGnedinOmega], respectively. Solution proceeds via an iterative procedure to find equilibrium radii for all galactic components in a consistently contracted halo. The method used follows that described by Benson and Bower (2010b).

### 4.9. Galaxy merging

The process of merging two galaxies currently involves two algorithms: one which decides how the merger causes mass components from both galaxies to move and one which determines the size of the remnant galaxy.

#### 4.9.1. Mass movements

The movement of mass elements in the merging galaxies. In the following, $M_1$ and $M_2$ are the baryonic masses of the satellite and central galaxies respectively that are about to merge.

*Simple:* This method implements mass movements according to:

- If $M_1 > f_{\rm major}M_2$ then all mass from both satellite and central galaxies moves to the spheroid component of the central galaxy;
- Otherwise: Gas from the satellite moves to the component of the central specified by the [minorMergerGasMovesTo] parameter (either "disk" or "spheroid"), stars from the satellite move to the spheroid of the central and mass in the central does not move.

Here, $f_{\rm major}$ = [majorMergerMassRatio] is the mass ratio above which a merger is considered to be "major".

#### 4.9.2. Remnant sizes

The method used to calculate the sizes of merger remnants.

*Null:* This is a null method which does nothing at all. It is useful, for example, when running Galacticus to study dark matter only (i.e. when no galaxy properties are computed).

Cole et al. (2000)*:* This method uses the algorithm of Cole et al. (2000) to compute merger remnant spheroid sizes. Specifically

$$\frac{(M_1+M_2)^2}{r_{\rm new}}=\frac{M_1^2}{r_1}+\frac{M_2^2}{r_2}+\frac{f_{\rm orbit}}{c}\frac{M_1M_2}{r_1+r_2}, \tag{48}$$

where $M_1$ and $M_2$ are the baryonic masses of the merging galaxies and $r_1$ and $r_2$ are their half mass radii, $r_{\rm new}$ is the half mass radius of the spheroidal component of the remnant galaxy and $c$ is a constant which depends on the distribution of the mass. For a Hernquist spheroid $c = 0.40$ can be found by numerical integration while for an exponential disk $c = 0.49$. For simplicity a value of $c = 0.5$ is adopted for all components. The parameter $f_{\rm orbit}$ = [mergerRemnantSizeOrbitalEnergy] depends on the orbital parameters of the galaxy pair. For example, a value of $f_{\rm orbit} = 1$ corresponds to point mass galaxies in circular orbits about their center of mass.

### 4.10. Hot halo density profile

The hot halo density profile is used, for example, in calculations of cooling rates.

*Cored Isothermal:* This method adopts a spherically symmetric cored-isothermal density profile for the hot halo. Specifically,

$$\rho_{\rm hot\ halo}(r)\propto\left[r^2+r_{\rm core}^2\right]^{-1}, \tag{49}$$

where the core radius, $r_{\rm core}$, is set to be a fixed fraction of the virial radius, that fraction being given by the input parameter [isothermalCoreRadiusOverVirialRadius]. The profile is normalized such that the current mass in the hot gas profile is contained within the virial radius.

### 4.11. Hot halo temperature profile

The hot halo temperature profile is used, for example, in calculations of cooling rates.

*Virial Temperature:* This method assumes an isothermal halo with a temperature equal to the virial temperature of the halo.

### 4.12. Initial mass functions

The stellar initial mass function (IMF) subsystem within Galacticus supports multiple IMFs and extensible algorithms to select which initial mass function (IMF) to use based on the physical conditions of star formation.

#### 4.12.1. Initial mass function selection

The method to use for selecting which initial mass function (IMF) to use.

*Fixed:* This method uses a fixed initial mass function (IMF) irrespective of physical conditions. The initial mass function (IMF) to use is specified by the [`imfSelectionFixed`] parameter (e.g. setting this parameter to `Salpeter` selects the Salpeter initial mass function (IMF)).

#### 4.12.2. Initial mass functions

A variety of different initial mass function (IMF) s are available. Each initial mass function (IMF) supplies a recycled fraction and metal yield for use in the instantaneous recycling approximation. These can be set via the parameters `imf{imfName}RecycledInstantaneous` and `imf{imfName}YieldInstantaneous` where {`imfName`} is the name of the initial mass function (IMF).

*Chabrier:* The `Chabrier` initial mass function (IMF) is defined by (Chabrier, 2001):

$$\phi(M) \propto \begin{cases} M^{-1} \mathrm{e}^{-[\log_{10}(\frac{M}{M_\mathrm{c}})/\sigma_\mathrm{c}]^2/2} & \text{for } 0.1\,\mathrm{M}_\odot < M < 1\,\mathrm{M}_\odot, \\ M^{-2.3} & \text{for } 1\,\mathrm{M}_\odot < M < 125\,\mathrm{M}_\odot, \\ 0 & \text{otherwise,} \end{cases} \tag{50}$$

where $\sigma_\mathrm{c} = 0.69$ and $M_\mathrm{c} = 0.08\,\mathrm{M}_\odot$.

*Kennicutt:* The `Kennicutt` initial mass function (IMF) is defined by (Kennicutt, 1983):

$$\phi(M) \propto \begin{cases} M^{-1.25} & \text{for } 0.10\,\mathrm{M}_\odot < M < 1.00\,\mathrm{M}_\odot, \\ M^{-2.00} & \text{for } 1.00\,\mathrm{M}_\odot < M < 2.00\,\mathrm{M}_\odot, \\ M^{-2.30} & \text{for } 2.00\,\mathrm{M}_\odot < M < 125\,\mathrm{M}_\odot, \\ 0 & \text{otherwise.} \end{cases} \tag{51}$$

*Kroupa:* The `Kroupa` initial mass function (IMF) is defined by (Kroupa, 2001):

$$\phi(M) \propto \begin{cases} M^{-0.3} & \text{for } 0.01\,\mathrm{M}_\odot < M < 0.08\,\mathrm{M}_\odot, \\ M^{-1.8} & \text{for } 0.08\,\mathrm{M}_\odot < M < 0.5\,\mathrm{M}_\odot, \\ M^{-2.7} & \text{for } 0.5\,\mathrm{M}_\odot < M < 1\,\mathrm{M}_\odot, \\ M^{-2.3} & \text{for } 1\,\mathrm{M}_\odot < M < 125\,\mathrm{M}_\odot \\ 0 & \text{otherwise.} \end{cases} \tag{52}$$

*Miller-Scalo:* The `Miller-Scalo` initial mass function (IMF) is defined by (Miller and Scalo, 1979):

$$\phi(M) \propto \begin{cases} M^{-1.25} & \text{for } 0.10\,\mathrm{M}_\odot < M < 1.00\,\mathrm{M}_\odot, \\ M^{-2.00} & \text{for } 1.00\,\mathrm{M}_\odot < M < 2.00\,\mathrm{M}_\odot, \\ M^{-2.30} & \text{for } 2.00\,\mathrm{M}_\odot < M < 10.0\,\mathrm{M}_\odot, \\ M^{-3.30} & \text{for } 10.0\,\mathrm{M}_\odot < M < 125\,\mathrm{M}_\odot. \\ 0 & \text{otherwise.} \end{cases} \tag{53}$$

*Salpeter:* The `Salpeter` initial mass function (IMF) is defined by (Salpeter, 1955):

$$\phi(M) \propto \begin{cases} M^{-2.35} & \text{for } 0.1\,\mathrm{M}_\odot < M < 125\,\mathrm{M}_\odot, \\ 0 & \text{otherwise.} \end{cases} \tag{54}$$

*Scalo:* The `Scalo` initial mass function (IMF) is defined by (Scalo, 1986):

$$\phi(M) \propto \begin{cases} M^{+1.60} & \text{for } 0.10\,\mathrm{M}_\odot < M < 0.18\,\mathrm{M}_\odot, \\ M^{-1.01} & \text{for } 0.18\,\mathrm{M}_\odot < M < 0.42\,\mathrm{M}_\odot, \\ M^{-2.75} & \text{for } 0.42\,\mathrm{M}_\odot < M < 0.62\,\mathrm{M}_\odot, \\ M^{-2.08} & \text{for } 0.62\,\mathrm{M}_\odot < M < 1.18\,\mathrm{M}_\odot, \\ M^{-3.50} & \text{for } 1.18\,\mathrm{M}_\odot < M < 3.50\,\mathrm{M}_\odot, \\ M^{-2.63} & \text{for } 3.50\,\mathrm{M}_\odot < M < 125\,\mathrm{M}_\odot, \\ 0 & \text{otherwise.} \end{cases} \tag{55}$$

### 4.13. Ionization state

The ionization state of gas, including, for example, the electron density, as a function of the gas density, composition and temperature.

*Atomic Collisional Ionization Equilibrium Using* Cloudy: This method computes the ionization state using the Cloudy code and under the assumption of collisional ionization equilibrium with no molecular contribution. Abundances are Solar, except for zero metallicity calculations which use Cloudy's "primordial" metallicity. The helium abundance for non-zero metallicity is scaled linearly with metallicity between primordial and Solar values. The cloudy code will be downloaded and run to compute the cooling function as needed, which will then be stored for future use. As this process is slow, a precomputed table is provided with Galacticus. If metallicities outside the range tabulated in this file are required it will be regenerated with an appropriate range.

*Collisional Ionization Equilibrium From File:* In this method the ionization state is read from a file. The ionization state is assumed to be computed under conditions of collisional ionization equilibrium and therefore densities of all species scale as the total density, $\rho$.

### 4.14. Merger tree construction

Merger trees are "constructed[6]" using a method specified via [`mergerTreeConstructMethod`].

*Read From File:* This method reads merger tree structures from an HDF5 file.

*Build:* This method first creates a distribution of tree root halo masses at a specified final redshift and then builds a merger tree using the selected build algorithm (see Section 4.16). The root halo masses are selected to lie within a user specified range, with a specified average number of trees per decade of halo mass. The distribution of halo masses is such that the mass of the $i$th halo is

$$M_{\mathrm{halo,i}} = \exp\left[\ln(M_{\mathrm{halo,min}}) + \ln(M_{\mathrm{halo,max}}/M_{\mathrm{halo,min}}) x_i^{1+\alpha}\right]. \tag{56}$$

[6] By "construct" we mean any process of creating a representation of a merger tree within Galacticus.

Here, $x_i$ is a number between 0 and 1 and $\alpha$ is an input parameter that controls the relative number of low and high mass tree produced. The distribution of $x$ is controlled by an input parameter with options:

`uniform` $x$ is distributed uniformly between 0 and 1;
`quasi` $x$ is distributed using a quasi-random sequence.

### 4.15. Merger tree branching

The method to be used for computing branching probabilities in merger trees when trees are constructed using Monte Carlo techniques.

*Modified Press-Scheduler:* This method uses the algorithm of Parkinson et al. (2008) to compute branching ratios. The parameters $G_0$, $\gamma_1$ and $\gamma_2$ of their algorithm are specified by the input parameters [`modifiedPressSchechterG0`], [`modifiedPressSchechterGamma1`] and [`modifiedPressSchechterGamma2`] respectively. Additionally, the parameter [`modifiedPressSchechterFirstOrderAccuracy`] limits the step in the critical linear theory overdensity for collapse, $\delta_{\rm crit}$, so that it never exceeds [`modifiedPressSchechterFirstOrderAccuracy`] times $\sqrt{2[\sigma^2(M_2/2)-\sigma^2(M_2)]}$, where $M_2$ is the mass of the halo being considered for branching and $\sigma(M)$ is the cold dark matter (CDM) mass variance computed by filtering the power spectrum using top-hat spheres. This ensures that the first order expansion of the merging rate that is assumed is accurate. Progenitor mass functions computed by this branching algorithm when used in the Cole et al. (2000) merger tree building algorithm are shown in Fig. 4 where they are compared with equivalent mass functions measured from the Millennium Simulation (Springel et al., 2005). Clearly there is excellent agreement with the $N$-body results.

### 4.16. Merger tree building

The method to be used for building merger trees.

Cole et al. (2000) *Algorithm:* This method uses the algorithm described by Cole et al. (2000), with a branching probability method selected via the `treeBranchingMethod` parameter. This action of this algorithm is controlled by the following parameters:

[`mergerTreeBuildCole2000MergeProbability`] The maximum probability for a binary merger allowed in a single timestep. This allows the probability to be kept small, such the probability for multiple mergers within a single timestep is small.
[`mergerTreeBuildCole2000AccretionLimit`] The maximum fractional change in mass due to sub-resolution accretion allowed in any given timestep when building the tree.
[`mergerTreeBuildCole2000MassResolution`] The minimum halo mass that the algorithm will follow. Mass accretion below this scale is treated as smooth accretion and branches are truncated once they fall below this mass.

*Smooth Accretion:* This method builds a branchless merger tree with a smooth accretion history using the fitting formula of Wechsler et al. (2002). The tree is defined by a final mass at a specified redshift and is continued back in time by decreasing the halo mass by a specified factor at each new node until a specified mass resolution is reached. The fitting formula of Wechsler et al. (2002) has one free parameter, the formation redshift. The formation redshift can either be computed automatically using the method of Bullock et al. (2001), or specified by an input parameter.

### 4.17. Star formation timescales

The methods for computing star formation timescales in disks and spheroids.

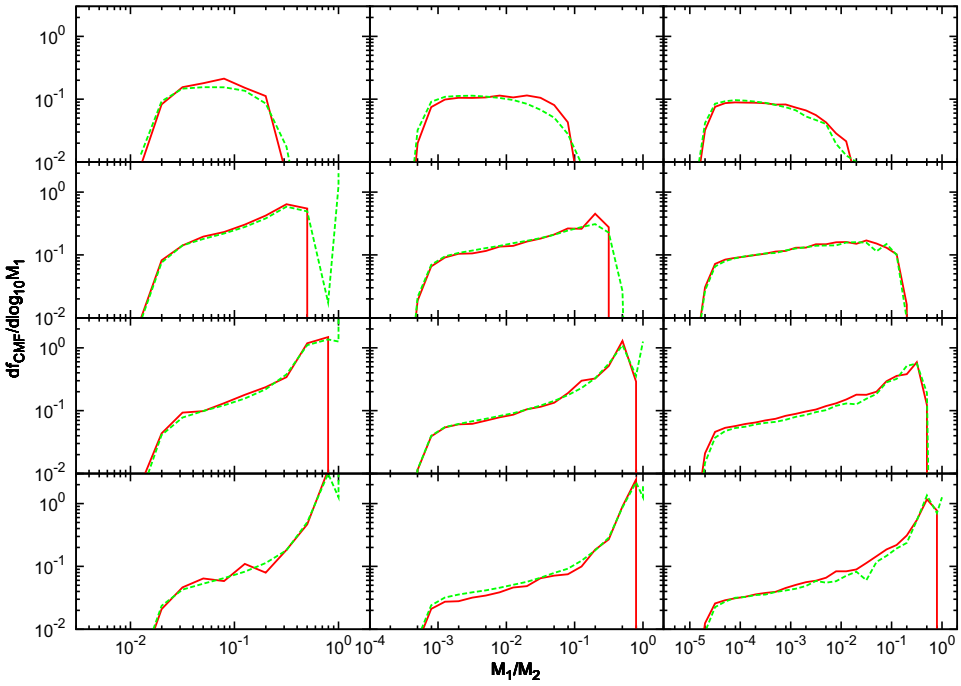


**Fig. 4.** Progenitor mass functions at redshifts $z = 0.5$, 1, 2 and 4 (bottom to top) for halos of mass $10^{12\pm0.151}$, $10^{13.5\pm0.151}$ and $10^{15\pm0.151}$ $h^{-1}$ $M_\odot$ (left to right) at $z = 0$ are shown. Here, $M_1$ is the mass of the progenitor halo and $M_2$ the mass of the $z = 0$ halo. Green lines were measured from the Millennium Simulation (Springel et al., 2005) by Parkinson et al. (2008), while red lines are computed using Galacticus's merger tree building routines (with the Parkinson et al. (2008) branching algorithm and the Cole et al. (2000) tree building algorithm). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

*Power Law:* This method (as used by Cole et al. (2000) for example) computes the star formation timescale to be:

$$\tau_\star = \epsilon_\star^{-1}\tau_{\rm dynamical}\left(\frac{V}{200\,{\rm km/s}}\right)^{\alpha_\star}, \quad (57)$$

where $\epsilon_\star$ = [starFormation{Component}Efficiency] and $\alpha_\star$ = [starFormation{Component} VelocityExponent] are input parameters (with {Component} being a placeholder for Disk or Spheroid), $\tau_{\rm dynamical} \equiv r/V$ is the dynamical timescale of the component and $r$ and $V$ are the characteristic radius and velocity respectively of the component. The timescale is not allowed to fall below a minimum value specified by [starFormation{Component}MinimumTimescale].

### 4.18. Stellar population properties

The algorithm for determining stellar population properties—essentially the rates of change of stellar and gas mass and abundances given a star formation rate and fuel abundances (and perhaps a historical record of star formation in the component)—is specified by [stellarPopulationPropertiesMethod].

*Instantaneous:* This method uses the instantaneous recycling approximation. Specifically, given a star formation rate $\phi$, this method assumes a rate of increase of stellar mass of $\dot{M}_\star = (1-R)\phi$, and a corresponding rate of decrease in fuel mass. The rate of change of the metal content of stars follows from the fuel metallicity, while that of the fuel changes according to

$$\dot{M}_{{\rm fuel},Z} = -(1-R)Z_{\rm fuel}\phi + y\phi. \quad (58)$$

In the above $R$ is the instantaneous recycled fraction and $y$ is the yield, both of which are supplied by the initial mass function (IMF) subsystem (see Section 4.12). The rate of energy input from the stellar population is computed assuming that the canonical amount of energy from a single stellar population (as defined by the feedbackEnergyInputAtInfinityCanonical) is input instantaneously.

*Non-instantaneous:* This method assumes fully non-instantaneous recycling and metal enrichment. Recycling and metal production rates from simple stellar populations are computed, for any given initial mass function (IMF), from stellar evolution models. The rates of change are then:

$$\dot{M}_\star = \phi - \int_0^t \phi(t')\dot{R}(t-t';Z_{\rm fuel}[t'])\,{\rm d}t', \quad (59)$$

$$\dot{M}_{\rm fuel} = -\phi + \int_0^t \phi(t')\dot{R}(t-t';Z_{\rm fuel}[t])\,{\rm d}t', \quad (60)$$

$$\dot{M}_{\star,Z} = Z_{\rm fuel}\phi - \int_0^t \phi(t')Z_{\rm fuel}(t')\times\dot{R}(t-t';Z_{\rm fuel}[t'])\,{\rm d}t', \quad (61)$$

$$\dot{M}_{{\rm fuel},Z} = -Z_{\rm fuel}\phi + \int_0^t \phi(t')\{Z_{\rm fuel}(t')\dot{R}(t-t';Z_{\rm fuel}[t']) + \dot{p}(t-t';Z_{\rm fuel}[t'])\}\,{\rm d}t', \quad (62)$$

where $\dot{R}(t;Z)$ and $\dot{p}(t;Z)$ are the recycling and metal yield rates respectively from a stellar population of age $t$ and metallicity $Z$. The energy input rate is computed self-consistently from the star formation history.

### 4.19. Stellar population spectra

Stellar population spectra are used to construct integrated spectra of galaxies.

*Conroy, White & Gunn:* This method uses v2.1 of the FSPS code of Conroy et al. (2009) to compute stellar spectra. If necessary, the FSPS code will be downloaded, patched[7] and compiled and run to generate spectra. These tabulations are then stored to file for later retrieval.

*File:* This method reads stellar population spectra from an HDF5 file.

### 4.20. Stellar population spectra post-processing

Stellar population spectra are post-processed (to handle, for example, absorption by the intergalactic medium (IGM)) by any number of the following algorithms.

Meiksin (2006) *IGM Attenuation:* This method post-processes spectra through absorption by the intergalactic medium (IGM) using the results of Meiksin (2006).

Madau (1995) *IGM Attenuation:* This method post-processes spectra through absorption by the intergalactic medium (IGM) using the results of Madau (1995).

*Null Method:* This method performs no post-processing.

### 4.21. Stellar astrophysics

Various properties related to stellar astrophysics are required by Galacticus. The following documents their implementation.

#### 4.21.1. Basics

This subset of properties include recycled mass, metal yield and lifetime.

*File:* This method reads properties of individual stars of different initial mass and metallicity from an XML file and interpolates in them. The stars can be irregularly spaced in the plane of initial mass and metallicity.

#### 4.21.2. Stellar winds

Energy input to the interstellar medium (ISM) from stellar winds is used in calculations of feedback efficiency.

Leitherer et al. (1992)*:* This method uses the fitting formulae of Leitherer et al. (1992) to compute stellar wind energy input from the luminosity and effective temperature of a star (see Section 4.21.3).

#### 4.21.3. Stellar tracks

The method used to compute stellar evolutionary tracks.

*File:* This method luminosities and effective temperatures of stars are computed from a tabulated set of stellar tracks. Galacticus is supplied with a suitable file constructed from the tracks of Bertelli et al. (2008) and Bertelli et al. (2009).

#### 4.21.4. Supernovae type Ia

Properties of Type Ia supernovae, including the cumulative number occurring and metal yield.

Nagashima et al. (2005) *Prescription:* This method uses the prescriptions from Nagashima et al. (2005) to compute the numbers and yields of Type Ia supernovae.

#### 4.21.5. Population III supernovae

Properties of Population III specific supernovae, in particular the energy released.

*Heger & Woosley (2002)*: This method computes the energies of pair instability supernovae from the results of Heger and Woosley (2002).

[7] Patching is not to fix any bugs in FSPS, but merely to insert code for reading a tabulated initial mass function (IMF) output by Galacticus.

#### 4.21.6. Stellar feedback

Aspects of stellar feedback, such as the energy input to the surrounding interstellar medium (ISM).

*Standard:* This method assumes that the cumulative energy input from a stellar population is equal to the total number of (Type II and Type Ia) supernovae multiplied by a specified energy per supernovae (SNe) plus any Population III-specific supernovae energy plus the integrated energy input from stellar winds. The number of Type II supernovae (SNe) is computed automatically from the initial mass function (IMF) and a specified minimum mass required for a Type II supernova.

### 4.22. Substructure and merging

Substructures and merging of nodes/substructures is controlled by several algorithms which are described below:

#### 4.22.1. Merging timescales

The method used to compute merging timescales of substructures is specified via the [`satelliteMergingMethod`] parameter.

*Dynamical Friction: Lacey & Cole:* This method computes merging timescales using the dynamical friction calculation of Lacey and Cole (1993). Timescales are multiplied by the value of the [`mergingTimescaleMultiplier`] input parameter.

*Dynamical Friction: Jiang (2008):* This method computes merging timescales using the dynamical friction calibration of Jiang et al. (2008).

*Dynamical Friction: Boylan-Kolchin (2008):* This method computes merging timescales using the dynamical friction calibration of Boylan-Kolchin et al. (2008).

#### 4.22.2. Virial orbits

The algorithm to be used to determine orbital parameters of substructures when they first enter the virial radius of their host.

Benson (2005)*:* This method selects orbital parameters randomly from the distribution given by Benson (2005).

#### 4.22.3. Node merging

The algorithm to be used to process nodes when they become substructures.

*Single Level Hierarchy:* This method maintains a single level hierarchy of substructure, i.e. it tracks only substructures, not sub-substructures or deeper levels. When a node first becomes a satellite it is appended to the list of satellites associated with its host halo. If the node contains its own satellites they will be detached from the node and appended to the list of satellites of the new host (and assigned new merging times; see Section 4.22.1).

### 4.23. Supernovae feedback models

The supernovae feedback driven outflow rate for disks and spheroids.

*Power Law:* This method assumes an outflow rate of:

$$\dot{M}_{\rm outflow} = \left(\frac{V_{\rm outflow}}{V}\right)^{\alpha_{\rm outflow}} \frac{\dot{E}}{E_{\rm canonical}}, \tag{63}$$

where $V_{\rm outflow}$ = [{`component`}`OutflowVelocity`] and $\alpha_{\rm outflow}$ = [{`component`}`OutflowExponent`] are input parameters, $V$ is the characteristic velocity of the component, $\dot{E}$ is the rate of energy input from stellar populations and $E_{\rm canonical}$ is the total energy input by a canonical stellar population normalized to 1 $M_\odot$ after infinite time.

### 4.24. Supermassive black holes binary mergers

The method to be used for computing the effects of binary mergers of supermassive black holes.

Rezzolla et al. (2008)*:* This method uses the fitting function of Rezzolla et al. (2008) to compute the spin of the black hole resulting from a binary merger. The mass of the resulting black hole is assumed to equal the sum of the mass of the initial black holes (i.e. there is negligible energy loss through gravitational waves).

## 5. Example calculations

In this section we show results from an example Galacticus model. The parameters of this model are listed in Table 1. It should be noted that the values of many of these parameters were chosen on the basis of previous experience with similar models—we have not performed an extensive search of the available parameter space, either manually or in an automated manner. As such, we do not claim that this example model represents the best match to observational data that can be obtained with the standard implementation of Galacticus—indeed it is certainly not so, although it does achieve reasonably good matches to many $z \approx 0$ observational datasets. Instead, this example model merely serves to illustrate the type of predictions which can be extracted from Galacticus. A detailed search of the available parameter space will be presented in a future paper.

### 5.1. Numerical results

The numerical solution of ODEs in Galacticus is controlled by various parameters which affect the accuracy of solution. In this subsection we explore the numerical convergence of Galacticus with respect to these results. Additionally, we examine the time taken to evolve merger trees in the example model.

#### 5.1.1. Convergence

To test the numerical convergence in Galacticus we run a single $10^{12}$ $M_\odot$ merger tree to $z = 0$ as our base model. We repeat the calculation for the same merger tree with altered numerical parameters as will be described below. The results of this exercise are shown in Fig. 5, in which we plot galaxies from this merger tree in the plane of total stellar mass and stellar mass-weighted scale length. The base model is shown by filled black circles. Galaxies in the comparison models which have a correspondent in the base model are indicated by open circles connected by a line to the base model galaxy. Where a base model galaxy is missing from the comparison model it is marked with a cross, while in cases where a comparison model galaxy is not present in the base model it is indicated by a star. The parameters adjusted in the three comparison models are:

[`odeToleranceRelative`] *Base model value:* 0.01; *Comparison model value:* 0.001; *Color:* red. This parameter specifies the accuracy requested from the ODE solver[8]—it is required to keep fractional errors in evolved quantities below this value during any given time step.

[`timestepHostRelative`] *Base model value:* 0.1; *Comparison model value:* 0.01; *Color:* green. This parameter limits the time interval by which satellite nodes may be evolved beyond the time at which their host halo is currently located. Specifically,

[8] We typically use a relatively large tolerance. Since the approximations made by semi-analytic models are not expected to be valid to high precision a higher tolerance is not generally warranted. Higher tolerances can be adopted if necessary of course.

**Table 1**
Values of parameters used in the example model. Parameters selecting between different implementations of physical processes or components are only listed where more than one non-null implementation currently exists within Galacticus.

| Parameter | Value | Reference |
|---|---|---|
| [H_0] | 70.2 km/s | Section 4.2; (Komatsu et al., 2010) |
| [Omega_0] | 0.2725 | Section 4.2; (Komatsu et al., 2010) |
| [Omega_DE] | 0.7275 | Section 4.2; (Komatsu et al., 2010) |
| [Omega_b] | 0.0455 | Section 4.2; (Komatsu et al., 2010) |
| [T_CMB] | 2.72548 K | Section 4.2; (Komatsu et al., 2010) |
| [accretionDisksMethod] | ADAF | Section 4.3 |
| [adafAdiabaticIndex] | 1.444 | Section 4.3 |
| [adafEnergyOption] | Pure ADAF | Section 4.3 |
| [adafRadiativeEfficiency] | 0.01 | Section 4.3 |
| [adafViscosityOption] | Fit | Section 4.3 |
| [adiabaticContractionGnedinA] | 0.8 | Section 4.8 |
| [adiabaticContractionGnedinOmega] | 0.77 | Section 4.8 |
| [barInstabilityMethod] | ELN | Section 4.7 |
| [blackHoleSeedMass] | 100 | Section 3.1.2 |
| [blackHoleWindEfficiency] | 0.001 | Section 3.1.2 |
| [bondiHoyleAccretionEnhancementHotHalo] | 1 | Section 3.1.2 |
| [bondiHoyleAccretionEnhancementSpheroid] | 1 | Section 3.1.2 |
| [bondiHoyleAccretionTemperatureSpheroid] | 100 | Section 3.1.2 |
| [coolingFunctionMethod] | Atomic CIE Cloudy | Section 4.5.1 |
| [coolingTimeAvailableAgeFactor] | 0 | Section 4.5.5 |
| [coolingTimeSimpleDegreesOfFreedom] | 3 | Section 4.5.4 |
| [darkMatterProfileMethod] | NFW | Section 4.6.1 |
| [darkMatterProfileMinimumConcentration] | 4 | Section 3.8.2 |
| [diskOutflowExponent] | 2 | Section 4.23 |
| [diskOutflowVelocity] | 200 km/s | Section 4.23 |
| [effectiveNumberNeutrinos] | 4.34 | Section 4.4.2 |
| [galacticStructureRadiusSolverMethod] | Adiabatic | Section 4.8 |
| [haloMassFunctionMethod] | Tinker2008 | Section 4.4.6 |
| [haloSpinDistributionMethod] | Bett2007 | Section 4.6.3 |
| [hotHaloOutflowReturnRate] | 1.26 | Section 3.2.2 |
| [imfSalpeterRecycledInstantaneous] | 0.39 | Section 4.12.2 |
| [imfSalpeterYieldInstantaneous] | 0.02 | Section 4.12.2 |
| [imfSelectionFixed] | Salpeter | Section 4.12.1 |
| [isothermalCoreRadiusOverVirialRadius] | 0.1 | Section 4.10 |
| [majorMergerMassRatio] | 0.1 | Section 4.9.1 |
| [mergerRemnantSizeOrbitalEnergy] | 1 | Section 4.9.2 |
| [mergerTreeBuildCole2000AccretionLimit] | 0.1 | Section 4.16 |
| [mergerTreeBuildCole2000MassResolution] | $5 \times 10^9\,M_\odot$ | Section 4.16 |
| [mergerTreeBuildCole2000MergeProbability] | 0.1 | Section 4.16 |
| [mergerTreeConstructMethod] | Build | Section 4.14 |
| [minorMergerGasMovesTo] | Spheroid | Section 4.9.1 |
| [modifiedPressSchechterFirstOrderAccuracy] | 0.1 | Section 4.15 |
| [modifiedPressSchechterG0] | 0.57 | Section 4.15 |
| [modifiedPressSchechterGamma1] | 0.38 | Section 4.15 |
| [modifiedPressSchechterGamma2] | −0.01 | Section 4.15 |
| [powerSpectrumIndex] | 0.961 | Section 4.4.1; (Komatsu et al., 2010) |
| [powerSpectrumReferenceWavenumber] | 1 $\mathrm{Mpc}^{-1}$ | Section 4.4.1; (Komatsu et al., 2010) |
| [powerSpectrumRunning] | 0 | Section 4.4.1; (Komatsu et al., 2010) |
| [randomSpinResetMassFactor] | 2 | Section 3.7.2 |
| [reionizationSuppressionRedshift] | 9 | Section 4.1 |
| [reionizationSuppressionVelocity] | 30 km/s | Section 4.1 |
| [satelliteMergingMethod] | Jiang2008 | Section 4.22.1 |
| [sigma_8] | 0.807 | Section 4.4.1 & Section 4.4.2 |
| [spheroidEnergeticOutflowMassRate] | 1 | Section 3.4.2 |
| [spheroidOutflowExponent] | 2 | Section 4.23 |
| [spheroidOutflowVelocity] | 50 km/s | Section 4.23 |
| [spinDistributionBett2007Alpha] | 2.509 | Section 4.6.3 |
| [spinDistributionBett2007Lambda0] | 0.04326 | Section 4.6.3 |
| [stabilityThresholdGaseous] | 0.9 | Section 4.7 |
| [stabilityThresholdStellar] | 1.1 | Section 4.7 |
| [starFormationDiskEfficiency] | 0.01 | Section 4.17 |
| [starFormationDiskMinimumTimescale] | 0.001 Gyr | Section 4.17 |
| [starFormationDiskVelocityExponent] | −1.5 | Section 4.17 |
| [starFormationSpheroidEfficiency] | 0.1 | Section 4.17 |
| [starFormationSpheroidMinimumTimescale] | 0.001 Gyr | Section 4.17 |
| [starveSatellites] | True | Section 3.2.2 |
| [stellarPopulationPropertiesMethod] | Instantaneous | Section 4.18 |
| [summedNeutrinoMasses] | 0 | Section 4.4.2 |
| [transferFunctionMethod] | Eisenstein + Hu | Section 4.4.2 |
| [virialDensityContrastMethod] | Spherical top hat | Section 4.4.5 |

no satellite node is allowed to evolve beyond [timestepHostRelative] times the cosmological expansion timescales, $H^{-1}(t)$, at the time of the host halo, before it must wait for the host node to catch up.

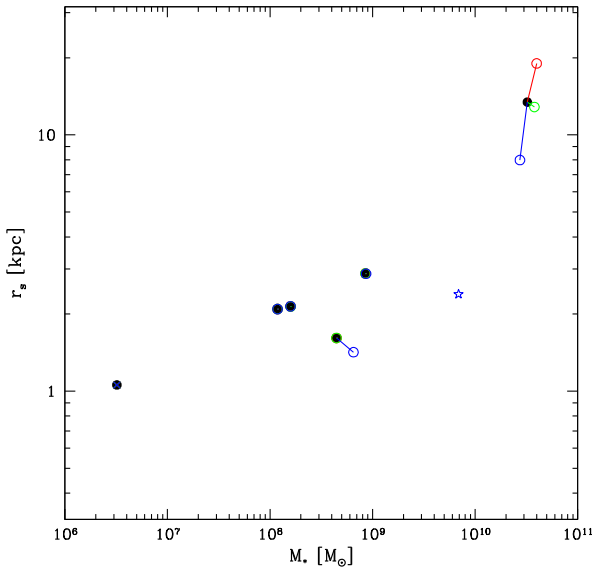


**Fig. 5.** Demonstration of convergence in galaxy stellar mass and stellar mass-weighted scale length. A single merger tree, with a final halo mass of $10^{12}\,M_\odot$ was run. Each point in the figure corresponds to a galaxy existing within that halo at $z = 0$. Black points correspond to our standard set of numerical parameters as defined in the text. Red points show a run with increased tolerance in the ODE solver. Green and blue points show runs with increased tolerance in time stepping relative to a node's host and to itself respectively. Where the same galaxy exists in the original model, the galaxy is represented by an open circle connected to the original model galaxy by a line. If a galaxy no longer exists in a modified model it is shown by a cross, while if a galaxy exists in the modified model that was not present in the original model it is shown by a star. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

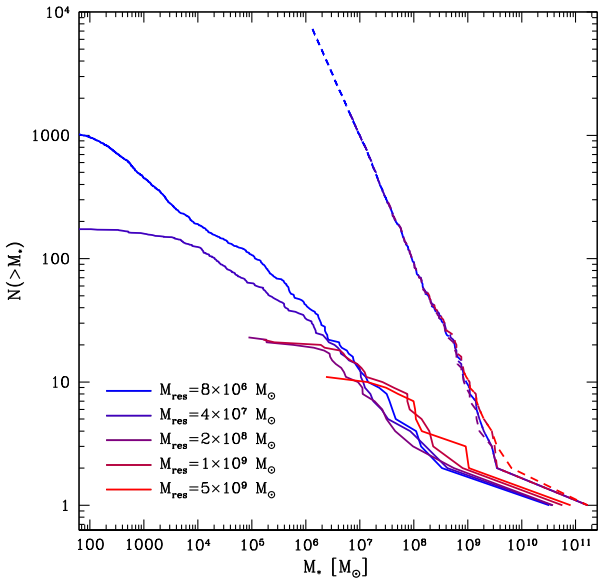


**Fig. 6.** Tests of convergence in the cumulative stellar mass function of galaxies formed in a merger tree with a final halo mass of $10^{12}\,M_\odot$ at $z = 0$. Solid lines show the stellar mass function, while dashed lines indicate the (sub) halo mass function scaled by a factor of $\Omega_b/\Omega_0$. The difference in slope between the two sets of mass functions is a consequence of the supernovae (SNe)-driven feedback included in this model. It can be seen that the (sub) halo mass function is well converged, while the galaxy stellar mass function is not. Note that black holes were not included in calculations used for this particular convergence study.

[`timestepSimpleRelative`] *Base model value:* 0.1; *Comparison model value:* 0.01; *Color:* blue. This parameter, specified as a fraction of the cosmological expansion timescale, $H^{-1}(t)$, at the time of the node, limits the time step over which any node may be evolved before evolution is paused and any necessary post-processing (such as transferring gas driven out of a satellite galaxy) is performed.

The results of this convergence study, as shown in Fig. 5, lead to several insights. First, the properties of half of the galaxies in this merger tree are almost perfectly stable to the specific changes in the numerical parameters that we have explored, indicating a high level of convergence. The other three galaxies show significant changes in their properties however. We begin by considering the most massive galaxy. When the ODE solver tolerance is improved (red symbols), this central galaxy gains slightly in mass and becomes significantly larger. By tracing the merging history of this galaxy in Galacticus, we can identify the cause of this as a merger with the lowest mass galaxy that was present in the base model. In the base model calculation, this lowest mass galaxy (located at a mass of around $3 \times 10^6\,M_\odot$ in Fig. 5) did not merge with the central. However, the improved ODE tolerance resulted in a merging event being triggered which then significantly altered the further evolution of the central galaxy. Conversely, when the [`timestepSimpleRelative`] parameter is reduced (blue symbols) we find a new galaxy appearing (indicated by the blue star). Here the opposite has happened—a galaxy which did merge in the base model failed to merge in the model with better timestep tolerance. This galaxy originally merged into the most massive galaxy in the tree. Consequently, that galaxy is now somewhat less massive and significantly smaller that in the base model. Finally, in this same model, the galaxy originally at $4.5 \times 10^8\,M_\odot$ is increased in mass and slightly smaller. Once again examining the history of this galaxy in Galacticus, we find that this change occurs because the galaxy in question has a disk which sits close to the stability threshold for bar formation (see Section 4.7). Under our current implementation of this process, the timescale for mass transfer from disk to spheroid due to the bar instability is finite at the stability threshold, but infinite below that threshold. As such, properties of galaxies close to this threshold will be sensitive to small changes in numerical parameters in Galacticus. Fundamentally, this represents a limitation of the physical model adopted for the bar instability—clearly a more physically motivated model is desirable (Athanassoula, 2008). While these effects make for substantial changes in the properties of the galaxy population of this particular merger tree, we find much less change when considering statistical samples of galaxies. Nevertheless, it is important to test for convergence in the quantities of interest for any particular study performed with Galacticus.

Fig. 6 shows another example of convergence. We take the same merger tree and re-simulate it while increasing the resolution of the tree branches.[9] Each line in Fig. 6 represents a factor 5 decrease in the minimum mass resolved in merger tree branches. Solid lines show the cumulative stellar mass function of galaxies, while dashed lines show the cumulative mass function of dark matter (sub) halos scaled by a factor $\Omega_b/\Omega_0$. Clearly the (sub) halo mass function is well converged. The galaxy mass function, however, is not. For example, the stellar mass of the most massive galaxy changes by a factor of around 3 from the highest to lowest

[9] Since our trees are generated by a Monte Carlo method simply changing the resolution and rebuilding the tree would result in a very different tree (a single different branching would lead to an entirely different branching path). To make a fair comparison between the different resolutions, we therefore create a tree at the highest resolution considered and then prune away branches below the required resolution threshold.

resolution runs. Of course, the results shown span a range of 625 in merger tree mass resolution. This resolution dependence is expected—as lower mass halos are resolved more gas gets locked up in low mass galaxies leaving less to form the massive galaxy. Convergence should only be reached once a physical suppression scale is reached. In the case of cold dark matter this will be a baryonic scale (since any cut-off in the cold dark matter (CDM) power spectrum is expected to occur at very low masses) associated with a truncation in the cooling function or with the temperature of the intergalactic medium (IGM) and photoheating by an ionizing background. Even at the highest resolution shown, full convergence across the whole mass function is not reached. However, for resolutions of $2 \times 10^8\,M_\odot$ and better the mass of the most massive (central) galaxy is converged to better than 50% which may be sufficient for many applications. Convergence in tree mass resolution will depend on the details of the implemented baryonic physics, and also on what galaxy samples/properties are being studied. This clearly demonstrates that a convergence test of the type carried out here should always be performed to ensure that results are not affected by the limited resolution of merger trees.

### 5.1.2. Timing

Galacticus was designed with simplicity, modularity and flexibility as key design principles. Execution speed was not a high priority but, nevertheless, we have given significant attention to optimizing key areas of the code. To explore the time taken to run typical calculations we have evolved sets of merger trees of different masses in our example model and computed the mean time taken to evolve each tree. The results are shown in Fig. 7, where we show results for the full model, the same model but with black hole physics switched off and the same model but with all baryonic physics switched off.

Fig. 7 shows that the time taken to evolve a tree scales close to, but somewhat faster than, linearly with halo mass. Since we use a fixed mass resolution in these merger trees, the number of nodes into which a tree is resolved will scale approximately with the mass of the halo, so this result is to be expected. While a Milky Way-mass halo can be evolved in around 2 s, a high mass cluster takes around 1.25 h. This is not as fast as some other semi-analytic models, but, as we stated above, speed is not our primary concern, with accuracy, simplicity and modularity taken precedence. Excluding black holes from the calculation significantly reduces run time by a factor of 2 to 2.5. This is because black hole properties (particularly spin) can evolve on very short timescales making evolution of the ODE system computationally expensive. This is worsened by the fact that many black holes are close to the equilibrium spin at which spin-up by accretion and spin down from powering jets cancel, forcing small timesteps to maintain this balance. Finally, switching off all baryonic physics (which leaves tree building, and evolution of the dark matter halos including dynamical friction, and which can be useful for

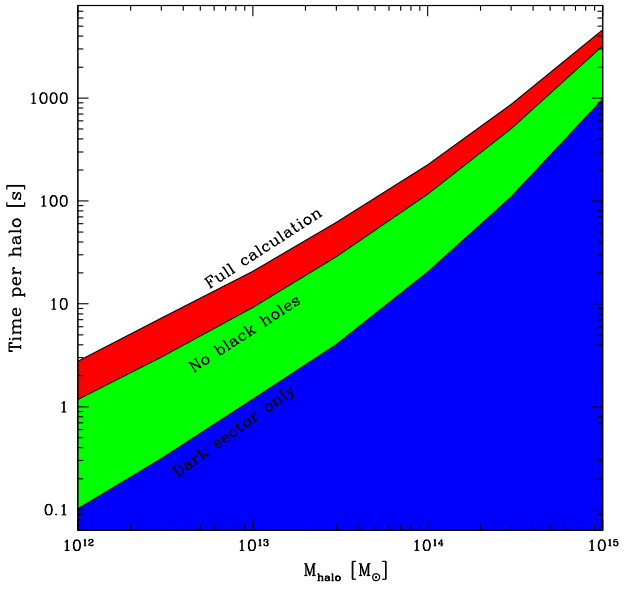


**Fig. 7.** Execution time per merger tree. (Run on a 2.7 GHz AMD Opteron processor and compiled using standard optimizations.) The red region indicates the average time taken to evolve a tree in the example model to $z = 0$. The green region indicates the time taken to run the same tree with black hole physics switched off. Finally, the blue region shows the time taken to evolve each tree when only dark sector physics are included (i.e. all baryonic physics is switched off). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

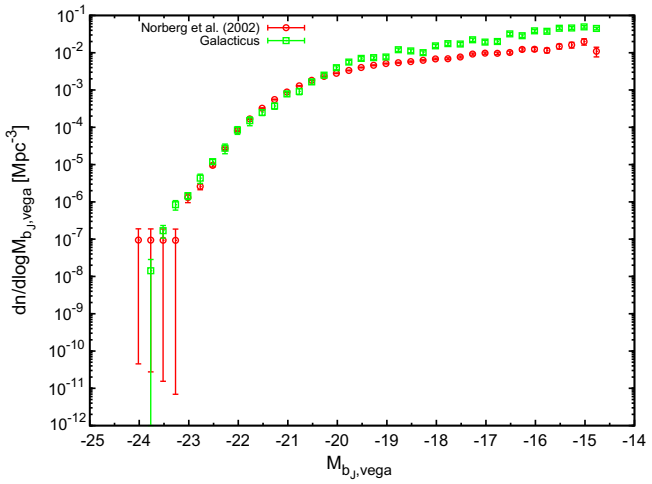


**Fig. 8.** The $z = 0$ $b_J$-band luminosity function. Red points indicate the observed luminosity function from the 2dFGRS (Norberg et al., 2002), while green points show results from our example Galacticus model after including the effects of dust-extinction using the model of Ferrara et al. (1999). Error bars on the model points indicate the uncertainty due to the finite number of merger trees computed. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

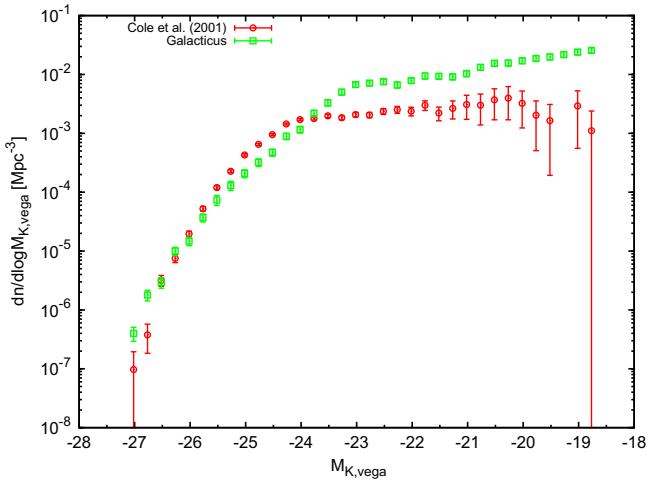


**Fig. 9.** The $z = 0$ K-band luminosity function from the example model. Red points indicate data from the 2dFGRS + 2MASS (Cole et al., 2001), while green points show results from our example Galacticus model after including the effects of dust-extinction using the model of Ferrara et al. (1999). Error bars on the model points indicate the uncertainty due to the finite number of merger trees computed. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

exploring the properties of dark matter models) reduces run time by a further factor of around 10.

### 5.2. Galaxy properties

In this final subsection we compare several observational datasets with their equivalents as predicted by the example Galacticus model. We reiterate that this example model is not intended to represent the best fit to observational data attainable with Galacticus—a full search of the Galacticus parameter space is a significant undertaking which will be explored in a future paper. Instead, the results in this subsection are intended to illustrate that types of quantities which can be computed and to show that they are, at least, in broad agreement with current observational constraints. We give a brief description and discussion of each result. Much of the underlying physics behind the model results has been previously discussed by other authors, as mentioned below. In a few cases, we defer further study to future papers which will explore some of the key physical drivers in greater detail.

Figs. 8 and 9 show $b_J$ and K-band luminosity functions at $z = 0$ derived from the 2dFGRS and 2MASS galaxy surveys, with observational determinations shown by red points and results from the example Galacticus model show by green points. Dust extinction is included in the model galaxy magnitudes using the results of Ferrara et al. (1999). The match to the bright end cut off in the luminosity functions is quite good, although the K-band prediction undershoots the knee of the observed luminosity function. However, the predicted faint end slope is significantly steeper than those observed (particularly in the K-band). This is a well known problem in semi-analytic models (e.g. Benson et al. (2003)) and is closely linked to the inclusion of supernovae (SNe)-feedback in such models. Stronger feedback would help to reconcile this discrepancy.

A key ingredient in producing the sharp cut off at the bright end of the galaxy luminosity function is the inclusion of active galactic nuclei (AGN) feedback in the example Galacticus model (Croton et al., 2006; Bower et al., 2006). As such, it is crucial that predicted black hole properties be reasonable. Fig. 10 shows the relation between black hole mass and spheroid stellar mass from observations (red points) and from the example Galacticus model (green points with error bars; indicating the mean and 1-$\sigma$ dispersion at each spheroid mass). The normalization and slope of the model

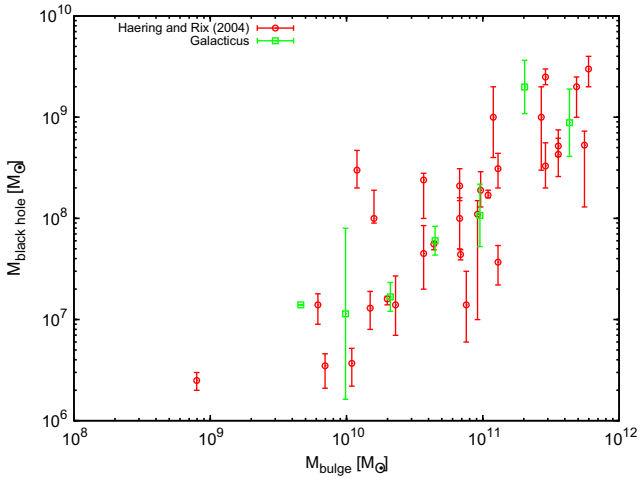


**Fig. 10.** The relation between supermassive black hole mass and galaxy bulge stellar mass. Data are taken from Häring and Rix (2004) and are shown by red points. Green points with error bars show the mean and 1-$\sigma$ dispersion from our example Galacticus model. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

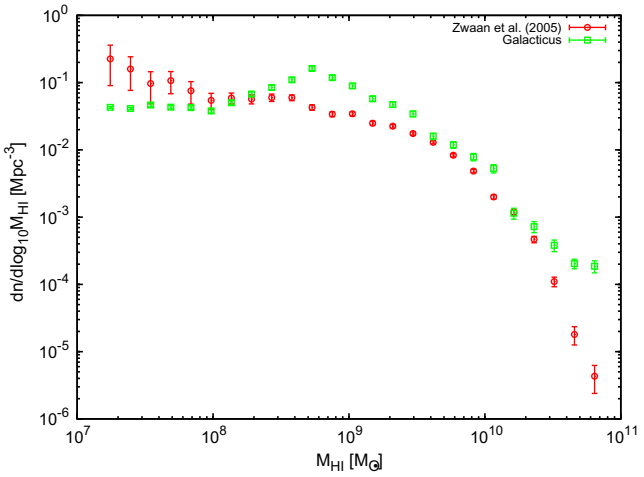


**Fig. 11.** The $z = 0$ HI gas mass function. The green points show the mass function from the example Galacticus model assuming a constant factor of 0.54 to convert from total gas mass to HI mass (Power et al., 2010). Red points indicate data from Zwaan et al. (2005). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

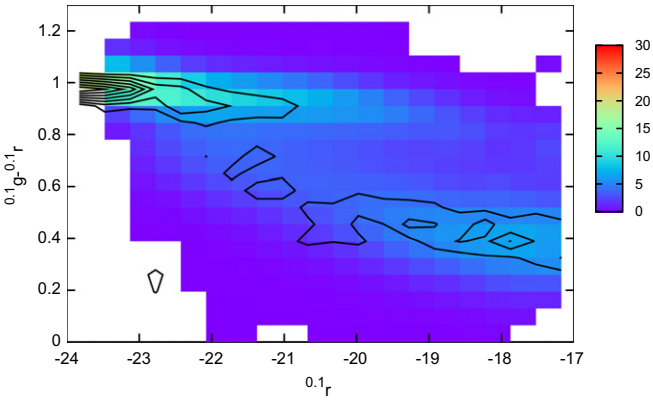


**Fig. 12.** $^{0.1}g-^{0.1}r$ differential color distribution (normalized to unit area under the surface) for galaxies at $z = 0.1$ as a function g-band absolute magnitude. The color scale shows data from the SDSS (Weinmann et al., 2006), while the black contours show the distribution from the example Galacticus model. Ten contour levels are shown, linearly spaced between the minimum and maximum of the color scale. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

relation is in good agreement with that which is observed. The model displays little scatter in black hole mass at fixed stellar mass, except for at the highest masses (partially due to the limited number of such high mass spheroids in our model sample) and around $10^{10}\,M_\odot$ spheroid mass. In the model, this increase in scatter below $10^{10}\,M_\odot$ occurs because the regulating mechanism of active galactic nuclei (AGN) feedback (which controls the relation between black hole and spheroid mass) breaks down.

Fig. 11 explores the gas content of galaxies in the example model, by comparing the model HI mass function with that observed by Zwaan et al. (2005). While the overall normalization is approximately correct the shape of the predicted function is incorrect and, in particular, does not truncate sharply enough at high masses. Similar results were found and considered in greater detail by Power et al. (2010).

A key result from the SDSS has been a clear demonstration of the dichotomy of the galaxy population, with most galaxies being part of a star forming "blue cloud" or a quiescent "red sequence". Fig. 12 shows the SDSS color-magnitude diagram (colored shading)

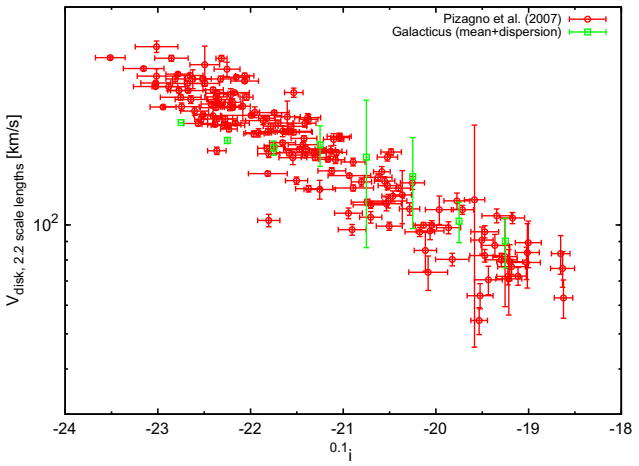


**Fig. 13.** The i-band Tully-Fisher relation from the SDSS (Pizagno et al., 2007) is shown by red points, while results from the example Galacticus model are indicated by green points. Model points show the mean disk circular velocity at each magnitude with error bars indicating the 1-$\sigma$ dispersion in model galaxies. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

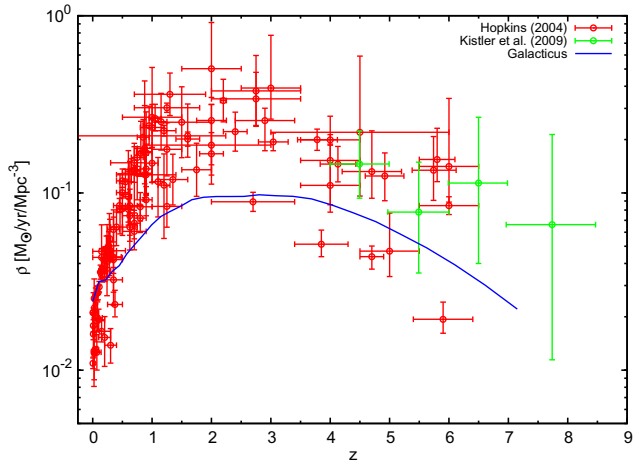


**Fig. 14.** The star formation rate per unit comoving volume in the Universe as a function of redshift. Red points show observational estimates from a variety of sources as compiled by Hopkins (2004) while green points show the star formation rate inferred from gamma ray bursts by Kistler et al. (2009). The blue line shows the result from the example Galacticus model. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

derived by Weinmann et al. (2006) which clearly shows these two populations. The black contours show the color-magnitude diagram from the example Galacticus model. This clearly shows the same two populations of galaxies, with approximately the correct median colors and transition luminosity. In the example model, active galactic nuclei (AGN) feedback plays a key role in establishing this division, as has been found by other authors (Croton et al., 2006; Bower et al., 2006; see also Font et al., 2008).

The structural and dynamical state of model galaxies is explored in Fig. 13, in which we compare results from the example model to the i-band Tully–Fisher relation measured by Pizagno et al. (2007). The model is in moderately good agreement with the data—the slope at lower luminosities is approximately correct, but flattens at higher luminosities such that the model velocities are too low. Additionally, the dispersion in model galaxy velocities at intermediate luminosities is too large. The Tully–Fisher relation encodes significant constraints on the process of galaxy formation.

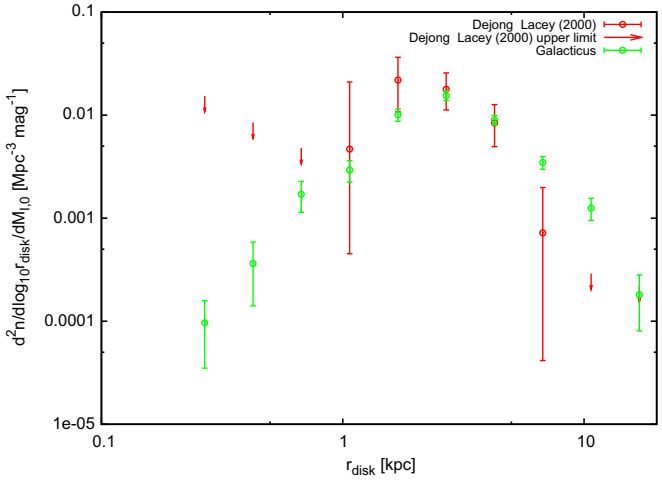


**Fig. 15.** Distribution of disk scale lengths for galaxies at $z = 0$ selected by face-on I-band absolute magnitude, $-20 < M_{I,0} < -19$. Red circles show data from de Jong and Lacy (2000) with upper limits indicated by red triangles. Green points show results from our example Galacticus model. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

As such, we intend to explore this aspect of the Galacticus model in greater detail in a future paper.

The volume averaged star formation rate density in the example model is shown in Fig. 14 and is compared with a compilation of observational data. Clearly the model under-predicts the star formation rate at all redshifts above $z \approx 0.25$. This is a key failing of the example model, and therefore represents a key constraint for future parameter spaace studies with Galacticus. The origins of this underestimate of the star formation rate are not clear without further investigation, but could plausibly be due to over-zealous active galactic nuclei (AGN) feedback at intermediate redshifts or simply a breakdown in the assumed scalings of star formation timescales (which are currently based on simple scaling relations rather than any physical model).

Finally, we show in Fig. 15 the sizes of galactic disks for galaxies in a narrow range of face-on I-band magnitude, $-20 < M_{I,0} < -19$, and compared to the observational determination of de Jong and Lacy (2000). The peak of the distribution in the example model is in approximately the correct location, but the model predicts somewhat too many very large disks compared to the data (although, given the large errors on the data in this region, the difference between model and data is not very significant). Sizes of galaxies are a key property since they directly affect galactic dynamical timescales and, therefore, star formation rates.

## 6. Summary

We have described a new, free and open source semi-analytic model of galaxy formation, Galacticus, which can be downloaded from http://sites.google.com/site/galacticusmodel. A full manual documenting both the physics and technical implementation of Galacticus can also be found on the same website.[10] In particular in this article, we have focused on the technical and practical implementation of Galacticus and have detailed the numerous physical ingredients which make up the model. Additionally, we have explored basic properties of a specific example model, demonstrating numerical convergence and illustrating the type of quantities which Galacticus can compute and compare against

[10] This article specifically describes the current development version of Galacticus, v0.9.0. The current stable version, v0.0.1, contains almost all of the same features.

observational data. Galacticus contains the majority of the physics incorporated into other major semi-analytic models (see Benson (2010) for a review). Missing physics, notably tidal and ram pressure mass loss from satellite galaxies (Benson et al., 2002; Lanzoni et al., 2005; Font et al., 2008; Henriques and Thomas, 2010) and the ability to use information on subhalos from *N*-body merger trees,[11] is already being incorporated into the development version of galacticus.

The Galacticus model can be employed for a wide variety of calculations related to galaxy formation studies, ranging from dark matter phenomenology, through observationally testable statistical properties of galaxy samples to the construction of mock galaxy samples. Its flexibility makes it easily adaptable to new problems and accepting of new physical prescriptions.

Galacticus requires only free and open source libraries and compilers to run and so is portable and practical to use. In designing Galacticus we placed an emphasis on simplicity, flexibility and extensibility to make it simple to use and maintain and to facilitate future development as required in the rapidly changing field of galaxy formation. We therefore hope that Galacticus will prove to be a valuable tool for a significant time, and we are already endeavoring to expand and improve upon this initial version.

## Acknowledgment

The author acknowledges the support of the Gordon & Betty Moore Foundation.

## References

Athanassoula, E., 2008. MNRAS 390 (October), L69 <http://adsabs.harvard.edu/abs/2008MNRAS.390L..69A>.
Bardeen, J.M., 1970. Nature 226 (April), 64 <http://adsabs.harvard.edu/abs/1970Natur.226...64B>.
Bardeen, J.M., Bond, J.R., Kaiser, N., Szalay, A.S., 1986. Astrophys. J. 304 (May), 15 <http://adsabs.harvard.edu/abs/1986ApJ...304...15B>.
Benson, A.J., 2005. MNRAS 358, 551 <http://adsabs.harvard.edu/abs/2005MNRAS.358..551B>.
Benson, A.J., 2010. Galaxy formation theory, June. <http://adsabs.harvard.edu/abs/2010arXiv1006.5394B>.
Benson, A.J., Babul, A., 2009. MNRAS 397 (August), 1302 <http://adsabs.harvard.edu/abs/2009MNRAS.397.1302B>.
Benson, A.J., Bower, R., 2010a. Cold mode accretion in galaxy formation, April. <http://adsabs.harvard.edu/abs/2010arXiv1004.1162B>.
Benson, A.J., Bower, R.G., 2010b. Galaxy formation spanning cosmic history, February. <http://adsabs.harvard.edu/abs/2010arXiv1003.0011B>.
Benson, A.J., Pearce, F.R., Frenk, C.S., Baugh, C.M., Jenkins, A., 2001. MNRAS 320, 261 <http://adsabs.harvard.edu/abs/2001MNRAS.320..261B>.
Benson, A.J., Lacey, C.G., Baugh, C.M., Cole, S., Frenk, C.S., 2002. MNRAS 333 (June), 156 <http://adsabs.harvard.edu/abs/2002MNRAS.333..156B>.
Benson, A.J., Bower, R.G., Frenk, C.S., Lacey, C.G., Baugh, C.M., Cole, S., 2003. Astrophys. J. 599 (December), 38 <http://adsabs.harvard.edu/abs/2003ApJ...599...38B>.
Bertelli, G., Girardi, L., Marigo, P., Nasi, E., 2008. Astron. Astrophys. 484 (June), 815 <http://adsabs.harvard.edu/abs/2008A%26A...484..815B>.
Bertelli, G., Nasi, E., Girardi, L., Marigo, P., 2009. Astron. Astrophys. 508 (December), 355 <http://adsabs.harvard.edu/abs/2009A%26A...508..355B>.
Bett, P., Eke, V., Frenk, C.S., Jenkins, A., Helly, J., Navarro, J., 2007. MNRAS 376 (March), 215 <http://adsabs.harvard.edu/abs/2007MNRAS.376..215B>.
Bower, R.G., Benson, A.J., Malbon, R., Helly, J.C., Frenk, C.S., Baugh, C.M., Cole, S., Lacey, C.G., 2006. MNRAS 370 (August), 645 <http://adsabs.harvard.edu/abs/2006MNRAS.370..645B>.
Bower, R.G., Vernon, I., Goldstein, M., Benson, A.J., Lacey, C.G., Baugh, C.M., Cole, S., Frenk, C.S., 2010. <http://adsabs.harvard.edu/doi/10.1111/j.1365-2966.2010.16991.x>.
Boylan-Kolchin, M., Ma, C., Quataert, E., 2008. MNRAS 383, 93 <http://adsabs.harvard.edu/abs/2008MNRAS.383...93B>.
Bryan, G.L., Norman, M.L., 1998. Astrophys. J. 495 (March), 80 <http://adsabs.harvard.edu/abs/1998ApJ...495...80B>.
Bullock, J.S., Kolatt, T.S., Sigad, Y., Somerville, R.S., Kravtsov, A.V., Klypin, A.A., Primack, J.R., Dekel, A., 2001. MNRAS 321 (March), 559 <http://adsabs.harvard.edu/abs/2001MNRAS.321..559B>.
Chabrier, G., 2001. Astrophys. J. 554 (June), 1274 <http://adsabs.harvard.edu/abs/2001ApJ...554.1274C>.
Ciotti, L., Ostriker, J.P., Proga, D., 2009. The Astrophysical Journal 699 (July), 89 <http://adsabs.harvard.edu/abs/2009ApJ...699...89C>.
Cole, S., Lacey, C.G., Baugh, C.M., Frenk, C.S., 2000. MNRAS 319 (November), 168 <http://adsabs.harvard.edu/abs/2000MNRAS.319..168C>.
Cole, S., Norberg, P., Baugh, C.M., Frenk, C.S., Bland-Hawthorn, J., Bridges, T., Cannon, R., Colless, M., Collins, C., Couch, W., Cross, N., Dalton, G., Propris, R.D., Driver, S.P., 2001. MNRAS 326 (September), 255 <http://adsabs.harvard.edu/abs/2001MNRAS.326..255C>.
Conroy, C., Gunn, J.E., White, M., 2009. Astrophys. J. 699 (July), 486 <http://adsabs.harvard.edu/abs/2009ApJ...699..486C>.
Cooper, A., De Lucia, G., Helly, J., Helmi, A., Li, Y.-S., McCarthy, I., Navarro, J., Springel, V., Starkenburg, E., Wang, J., 2011, MNRAS, in press.
Croton, D.J., Springel, V., White, S.D.M., Lucia, G.D., Frenk, C.S., Gao, L., Jenkins, A., Kauffmann, G., Navarro, J.F., Yoshida, N., 2006. MNRAS 365, 11 <http://adsabs.harvard.edu/abs/2006MNRAS.365...11C>.
de Jong, R.S., Lacey, C., 2000. Astrophys. J. 545, 781 <http://adsabs.harvard.edu/abs/2000ApJ...545..781D>.
Edgar, R., 2004. New Astron. Rev. 48 (10), 843.
Efstathiou, G., Lake, G., Negroponte, J., 1982. MNRAS 199, 1069 <http://adsabs.harvard.edu/abs/1982MNRAS.199.1069E>.
Eisenstein, D.J., Hu, W., 1999. Astrophys. J. 511 (January), 5 <http://adsabs.harvard.edu/abs/1999ApJ...511....5E>.
Ferrara, A., Bianchi, S., Cimatti, A., Giovanardi, C., 1999. Astrophys. J. Suppl. Ser. 123 (August), 437 <http://adsabs.harvard.edu/abs/1999ApJS..123..437F>.
Font, A.S., Bower, R.G., McCarthy, I.G., Benson, A.J., Frenk, C.S., Helly, J.C., Lacey, C.G., Baugh, C.M., Cole, S., 2008. MNRAS 389 (October), 1619 <http://adsabs.harvard.edu/abs/2008MNRAS.389.1619F>.
Gao, L., Navarro, J.F., Cole, S., Frenk, C.S., White, S.D.M., Springel, V., Jenkins, A., Neto, A.F., 2008. MNRAS 387 (June), 536 <http://adsabs.harvard.edu/abs/2008MNRAS. 387..536G>.
Gnedin, O.Y., Kravtsov, A.V., Klypin, A.A., Nagai, D., 2004. Astrophys. J. 616 (November), 16 <http://adsabs.harvard.edu/abs/2004ApJ...616...16G>.
Häring, N., Rix, H., 2004. Astrophys. J. 604 (April), L89 <http://adsabs.harvard.edu/abs/2004ApJ...604L..89H>.
Hatton, S., Devriendt, J.E.G., Ninin, S., Bouchet, F.R., Guiderdoni, B., Vibert, D., 2003. MNRAS 343 (July), 75 <http://adsabs.harvard.edu/abs/2003MNRAS.343...75H>.
Heger, A., Woosley, S.E., 2002. Astrophys. J. 567 (March), 532 http://adsabs.harvard.edu/abs/2002ApJ...567..532H.
Helly, J.C., Cole, S., Frenk, C.S., Baugh, C.M., Benson, A., Lacey, C., Pearce, F.R., 2003. MNRAS 338 (February), 913 http://adsabs.harvard.edu/abs/2003MNR AS .338..913H.
Henriques, B.M.B., Thomas, P.A., 2010. MNRAS 403 (April), 768 <http://adsabs.harvard.edu/abs/2010MNRAS.403..768H>.
Henriques, B.M.B., Thomas, P.A., Oliver, S., Roseboom, I., 2009. MNRAS 396 (June), 535 <http://adsabs.harvard.edu/abs/2009MNRAS.396..535H>.
Hernquist, L., 1990. Astrophys. J. 356 (June), 359 <http://adsabs.harvard.edu/abs/1990ApJ...356..359H>.
Hopkins, A.M., 2004. Astrophys. J. 615 (November), 209 <http://adsabs.harvard.edu/abs/2004ApJ...615..209H>.
Jiang, C.Y., Jing, Y.P., Faltenbacher, A., Lin, W.P., Li, C., 2008. ApJ 675 (March), 1095 <http://adsabs.harvard.edu/abs/2008ApJ...675.1095J>.
Kauffmann, G., White, S.D.M., Guiderdoni, B., 1993. MNRAS 264 (September), 201 <http://adsabs.harvard.edu/abs/1993MNRAS.264..201K>.
Kennicutt, R.C., 1983. Astrophys. J. 272 (September), 54 <http://adsabs.harvard.edu/abs/1983ApJ...272...54K>.
Kistler, M.D., Yuksel, H., Beacom, J.F., Hopkins, A.M., Wyithe, J.S.B., 2009. The star formation rate in the reionization era as indicated by gamma-ray bursts. 0906.0590, June. <http://arxiv.org/abs/0906.0590>.
Komatsu, E., Smith, K.M., Dunkley, J., Bennett, C.L., Gold, B., Hinshaw, G., Jarosik, N., Larson, D., Nolta, M.R., Page, L., Spergel, D.N., Halpern, M., Hill, R.S., Kogut, A., Limon, M., Meyer, S.S., Odegard, N., Tucker, G.S., Weiland, J.L., Wollack, E., Wright, E.L., 2010. Seven-Year wilkinson microwave anisotropy probe (WMAP) observations: Cosmological interpretation, January. <http://adsabs.harvard.edu/abs/2010arXiv1001.4538K>.
Kroupa, P., 2001. MNRAS 322 (April), 231 <http://adsabs.harvard.edu/abs/2001MNRAS.322..231K>.
Lacey, C., Cole, S., 1993. MNRAS 262 (June), 627 <http://adsabs.harvard.edu/abs/1993MNRAS.262..627L>.
Lanzoni, B., Guiderdoni, B., Mamon, G.A., Devriendt, J., Hatton, S., 2005. MNRAS 361 (August), 369 <http://adsabs.harvard.edu/abs/2005MNRAS.361..369L>.
Leitherer, C., Robert, C., Drissen, L., 1992. ApJ 401 (December), 596 <http://adsabs.harvard.edu/abs/1992ApJ...401..596L>.
Lu, Y., Kereš, D., Katz, N., Mo, H.J., Fardal, M., Weinberg, M.D., 2010. On the algorithms of radiative cooling in semi-analytic models, August. <http://adsabs.harvard.edu/abs/2010arXiv1008.1075L>.
Madau, P., 1995. Astrophys. J. 441 (March), 18 <http://adsabs.harvard.edu/abs/1995ApJ...441...18M>.
Meier, D.L., 2001. Astrophys. J. 548 (February), L9 <http://adsabs.harvard.edu/abs/2001ApJ...548L...9M>.
Meiksin, A., 2006. MNRAS 365 (February), 807 <http://adsabs.harvard.edu/abs/2006MNRAS.365..807M>.
Miller, G.E., Scalo, J.M., 1979. Astrophys. J. Suppl. Ser. 41 (January), 513 <http://adsabs.harvard.edu/abs/1979ApJS...41..513M>.

[11] Galacticus can utilize simple merger trees from *N*-body simulations—software is provided, for example, to convert trees from the Millennium Simulation into Galacticus's format—but not those with subhalo information.

Monaco, P., Fontanot, F., Taffoni, G., 2007. MNRAS 375 (March), 1189 <http://adsabs.harvard.edu/abs/2007MNRAS.375.1189M>.
Nagashima, M., Lacey, C.G., Baugh, C.M., Frenk, C.S., Cole, S., 2005. MNRAS 358 (April), 1247 <http://adsabs.harvard.edu/abs/2005MNRAS.358.1247N>.
Narayan, R., Yi, I., 1994. Astrophys. J. 428 (June), L13 <http://adsabs.harvard.edu/abs/1994ApJ...428L..13N>.
Navarro, J.F., Frenk, C.S., White, S.D.M., 1997. ApJ 490 (December), 493 <http://adsabs.harvard.edu/abs/1997ApJ...490..493N>.
Norberg, P., Cole, S., Baugh, C.M., Frenk, C.S., Baldry, I., Bland-Hawthorn, J., Bridges, T., Cannon, R., Colless, M., Collins, C., Couch, W., Cross, N.J.G., Dalton, G., Propris, R.D., Driver, S.P., Efstathiou, G., Ellis, R.S., Glazebrook, K., Jackson, C., Lahav, O., Lewis, I., Lumsden, S., Maddox, S., Madgwick, D., Peacock, J.A., Peterson, B.A., Sutherland, W., Taylor, K., 2002. Mon. Notices Royal Astron. Soc. 336 (November), 907 <http://adsabs.harvard.edu/abs/2002MNRAS.336..907N>.
Ostriker, J.P., Choi, E., Ciotti, L., Novak, G.S., Proga, D., 2010. Momentum driving: which physical processes dominate AGN feedback? April. http://adsabs.harvard.edu/abs/2010arXiv1004.2923O>.
Parkinson, H., Cole, S., Helly, J., 2008. MNRAS 383 (January), 557 <http://adsabs.harvard.edu/abs/2008MNRAS.383..557P>.
Percival, W.J., 2005. Astron. Astrophys. 443 (December), 819 <http://adsabs.harvard.edu/abs/2005A%26A...443..819P>.
Pizagno, J., Prada, F., Weinberg, D.H., Rix, H., Pogge, R.W., Grebel, E.K., Harbeck, D., Blanton, M., Brinkmann, J., Gunn, J.E., 2007. Astron. J. 134, 945 <http://adsabs.harvard.edu/abs/2007AJ....134..945P>.
Power, C., Baugh, C.M., Lacey, C.G., 2010. MNRAS 406 (July), 43 <http://adsabs.harvard.edu/abs/2010MNRAS.406...43P>.
Press, W.H., Schechter, P., 1974. Astrophys. J. 187 (February), 425 <http://adsabs.harvard.edu/abs/1974ApJ...187..425P>.
Rezzolla, L., Barausse, E., Dorband, E.N., Pollney, D., Reisswig, C., Seiler, J., Husa, S., 2008. Phys. Rev. D 78 (August), 44002 <http://adsabs.harvard.edu/abs/2008PhRvD..78d4002R>.
Salpeter, E.E., 1955. Astrophys. J. 121 (January), 161 <http://adsabs.harvard.edu/abs/1955ApJ...121..161S>.
Scalo, J.M., 1986. Fundam. Cosmic Phys. 11 (May), 1 <http://adsabs.harvard.edu/abs/1986FCPh...11....1S>.
Shakura, N.I., Sunyaev, R.A., 1973. Astron. Astrophys. 24, 337 <http://ukads.nottingham.ac.uk/abs/1973A%26A....24..337S>.
Shapley, H., Curtis, H.D., 1921. Tech. Rep. 2.
Sheth, R.K., Mo, H.J., Tormen, G., 2001. MNRAS 323 (May), 1 <http://adsabs.harvard.edu/abs/2001MNRAS.323....1S>.
Somerville, R.S., Hopkins, P.F., Cox, T.J., Robertson, B.E., Hernquist, L., 2008. MNRAS 391 (December), 481 <http://adsabs.harvard.edu/abs/2008MNRAS.391..481S>.
Springel, V., White, S.D.M., Jenkins, A., Frenk, C.S., Yoshida, N., Gao, L., Navarro, J., Thacker, R., Croton, D., Helly, J., Peacock, J.A., Cole, S., Thomas, P., Couchman, H., Evrard, A., Colberg, J., Pearce, F., 2005. Nature 435 (7042), 629 http://dx.doi.org/10.1038/nature03597.
Stringer, M.J., Brooks, A.M., Benson, A.J., Governato, F., 2010. Analytic and numerical realisations of a disk galaxy, January. http://adsabs.harvard.edu/abs/2010arXiv1001.0594S>.
Tinker, J.L., Robertson, B.E., Kravtsov, A.V., Klypin, A., Warren, M.S., Yepes, G., Gottlober, S., 2010. The large scale bias of dark matter halos: numerical calibration and model tests, January. <http://adsabs.harvard.edu/abs/2010arXiv1001.3162T>.
Wechsler, R.H., Bullock, J.S., Primack, J.R., Kravtsov, A.V., Dekel, A., 2002. Astrophys. J. 568 (March), 52 <http://adsabs.harvard.edu/abs/2002ApJ...568...52W>.
Weinmann, S.M., van den Bosch, F.C., Yang, X., Mo, H.J., 2006. MNRAS 366 (February), 2 <http://adsabs.harvard.edu/abs/2006MNRAS.366....2W>.
White, S.D.M., Frenk, C.S., 1991. Astrophys. J. 379 (September), 52 <http://adsabs.harvard.edu/abs/1991ApJ...379...52W>.
White, S.D.M., Rees, M.J., 1978. MNRAS 183 (May), 341 <http://adsabs.harvard.edu/abs/1978MNRAS.183..341W>.
Yoshida, N., Stoehr, F., Springel, V., White, S.D.M., 2002. MNRAS 335 (September), 762 <http://adsabs.harvard.edu/abs/2002MNRAS.335..762Y>.
Zwaan, M.A., Meyer, M.J., Staveley-Smith, L., Webster, R.L., 2005. MNRAS 359 (May), L30 <http://adsabs.harvard.edu/abs/2005MNRAS.359L..30Z>.