Software Design Document
for the
Rotation Curve Modeler (RoCM)

Version 2.0

Robert Moss, Alex Clement

June 23, 2014

Contents

1 Introduction

1.1 Purpose

The Software Design Document (SDD) is intended to describe the software components of the Rotation Curve Modeler (RoCM). The SDD will include details about the software implementation, the data handling, and the overall design.

1.2 Scope

1. Rotation Curve Modeler
   (a) Interact with SOCM (Scholarly Observed Celestial Measurements)
      i. Use the repository of observable galactic data to model hundreds/thousands of different galaxies.
   (b) Interact with RoCS (Rotation Curve Simulation)
      i. RoCS visualizes the spin of star clusters around the center of a galaxy.
      ii. Include scale and legend for the visualization.
   (c) Allow users to locally import and run their own model (via JavaScript code)
      i. Following the defined v(R) input/output standard.
      ii. Observable galactic parameters from SOCM will be available as constants.
      iii. User defined constants will need to be implemented in the user's function.
   (d) Locally import LaTeX equation for each model (optional)
      i. The user can import their own LaTeX equation to be displayed during the data plotting.
      ii. Aids in understanding the behavior of each parameter.
   (e) Dynamic parameter sliders
      i. For every parameter in the individual models, a dynamic slider with user defined ranges can be created.

1.3 Overview

The SDD will contain explicit details about the RoCM software. Distinct acronyms and definitions will be clarified. References for our specified libraries will be included. A system overview of the software architectural design as well as the data handling design will be covered. Finally, the graphical interface design will also be established and talked about.

1.4 Definitions, Acronyms, and Abbreviations

- Application Specific Definitions
  - RoCM - Rotation Curve Modeler
  - SOCM - Scholarly Observed Celestial Measurements
  - RoCS - Rotation Curve Simulator
- Industry Definitions

  - WIT - Wentworth Institute of Technology
  - SRS - Software Requirements Specification
  - JavaScript - A web based programming language.
  - D3 - Data Driven Documents: A JavaScript library for data visualization.
  - Ruby on Rails - A web development framework written in the Ruby programming language.
  - JQuery - A JavaScript library for easy UI development.
  - LaTeX - A document preparation system used widely throughout science and mathematics.
  - SVG - Scalable Vector Graphics: A loss-less graphics format.

- Technical Definitions
  - MoND - Modification of Newtonian Dynamics
  - TeVeS - Tensor-vector-scalar gravity
  - MATLAB - A mathematical programming language (MATrix LABoratory).
  - Mathematica - A mathematical programming language.
  - DB - Database
  - UI - User Interface
  - GUI - Graphical User Interface
  - HTML - HyperText Markup Language
  - div - HTML tag to define a division in a document
  - DOM - Document Object Model. A convention for representing and interacting with objects in HTML.
  - API - Application Programming Interface
  - km/s - Kilometers Per Second
  - kpc - Kiloparsecs

1.5 Reference Material

The list of references below are software documentation that we will be using:

1. Data Driven Documents (D3): http://d3js.org/
2. JQuery documentation: http://api.jquery.com/
3. JQuery UI documentation: http://api.jqueryui.com/
4. MathJax documentation: http://docs.mathjax.org/en/latest/

2 System Overview

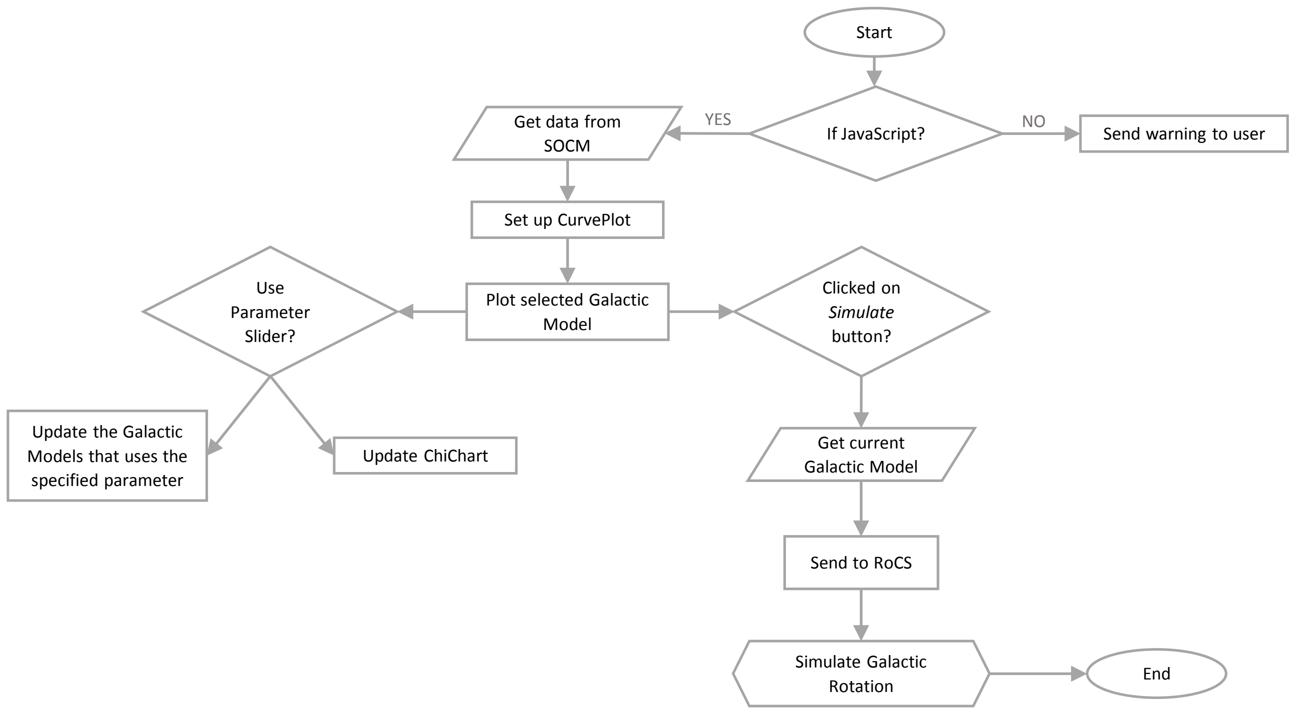
There will be many available functions the users can utilize, catering towards modeling galaxies and visualizing the galactic data. The modular design of the software will allow for additional functions to be easily implemented.

1. A rotation curve plotting tool that overlays different models of the galaxy on top of the observational data. Gives the user the ability to select which curve to plot based on the models available.
2. Value ranged sliders that can update each individual parameter within specific galactic models. Each slider can be dynamically created with user defined ranges. This enables the user to visualize the behavior of each parameter within the each model. Allows for the testing of uncertainty within the galactic parameters.
3. A χ^2 statistical testing module (ChiChart) will be added to calculate the validity of the purposed galactic model. This way, the user can test the strength of the proposed theory.
4. A LaTeX equation viewer for each galactic model. When using the parameter sliders, the formatted equation will help users understand the contribution of the changing parameter in the theory – helping to better understand the behavior of the theory as a whole.
5. RoCS will be able to simulate the spin of the galaxy in order to visualize the dynamics. Provided with a color scale for the speed of the stars around the center of the galaxy, the user will be able to understand the distribution of velocities within the galaxy.
6. The rotationcurve.org website will provide a web interface to the RoCM, SOCM, and RoCS modules. The website back-end will host galaxy research data, namely through SOCM, which will also provide API endpoints for ease of access and extensibility. rotationcurve.org will provide an 'About' section that explains the purpose of the website, and provides the email addresses to contact the contributors.

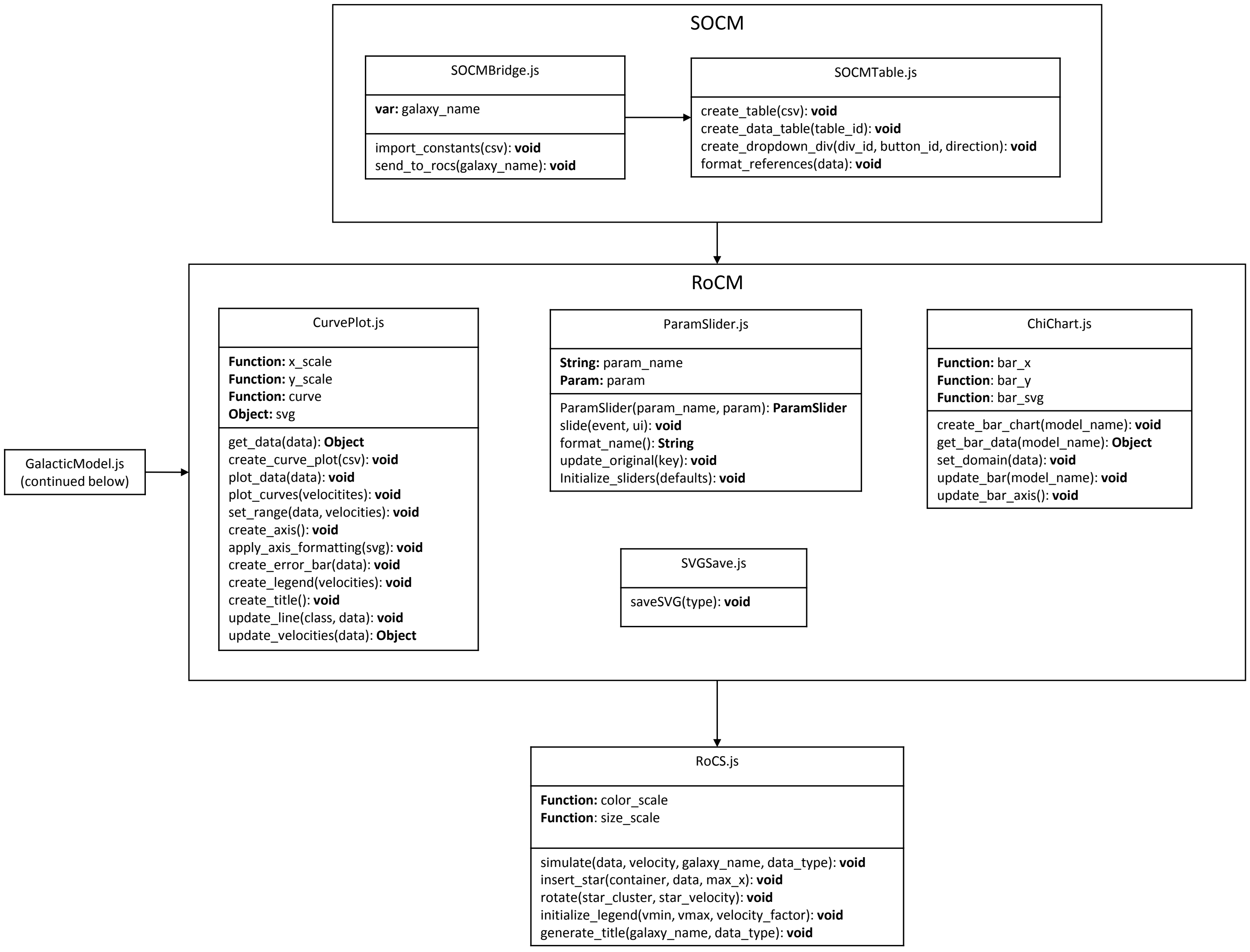
3 System Architecture

The description of the software architectural design and design decomposition will be laid out below.

3.1 Architectural Design

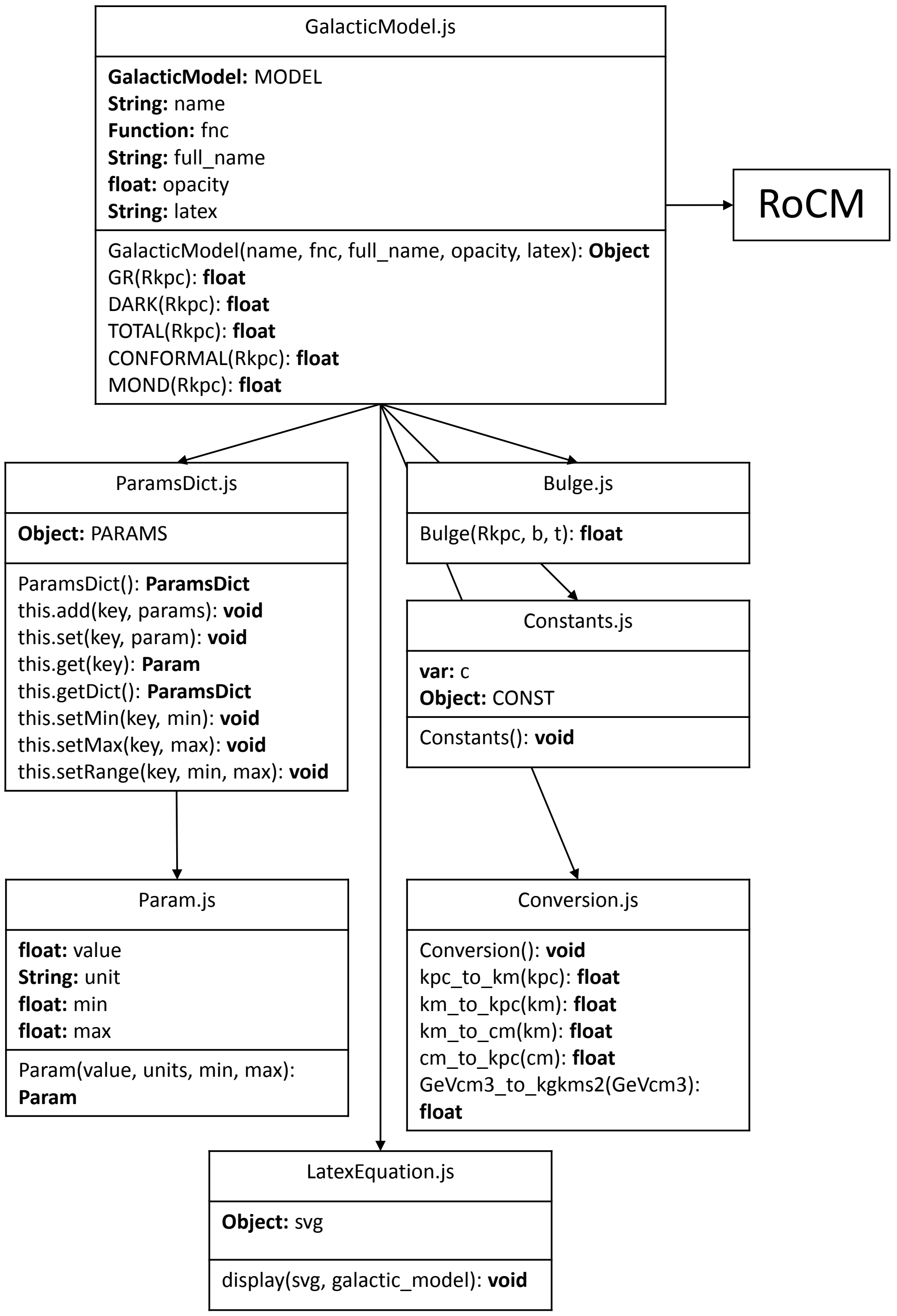

Start
If JavaScript?
YES
NO
Send warning to user
Get data from SOCM
Set up CurvePlot
Plot selected Galactic Model
Use Parameter Slider?
Update the Galactic Models that uses the specified parameter
Update ChiChart
Clicked on *Simulate* button?
Get current Galactic Model
Send to RoCS
Simulate Galactic Rotation
End

3.2 Decomposition Description



SOCM
SOCMBridge.js
var: galaxy_name
import_constants(csv): void
send_to_rocs(galaxy_name): void
SOCMTable.js
create_table(csv): void
create_data_table(table_id): void
create_dropdown_div(div_id, button_id, direction): void
format_references(data): void
RoCM
CurvePlot.js
Function: x_scale
Function: y_scale
Function: curve
Object: svg
get_data(data): Object
create_curve_plot(csv): void
plot_data(data): void
plot_curves(velocitites): void
set_range(data, velocities): void
create_axis(): void
apply_axis_formatting(svg): void
create_error_bar(data): void
create_legend(velocities): void
create_title(): void
update_line(class, data): void
update_velocities(data): Object
ParamSlider.js
String: param_name
Param: param
ParamSlider(param_name, param): ParamSlider
slide(event, ui): void
format_name(): String
update_original(key): void
Initialize_sliders(defaults): void
ChiChart.js
Function: bar_x
Function: bar_y
Function: bar_svg
create_bar_chart(model_name): void
get_bar_data(model_name): Object
set_domain(data): void
update_bar(model_name): void
update_bar_axis(): void
SVGSave.js
saveSVG(type): void
GalacticModel.js
(continued below)
RoCS.js
Function: color_scale
Function: size_scale
simulate(data, velocity, galaxy_name, data_type): void
insert_star(container, data, max_x): void
rotate(star_cluster, star_velocity): void
initialize_legend(vmin, vmax, velocity_factor): void
generate_title(galaxy_name, data_type): void

3.3 Decomposition Description Continued...



3.4 Design Rationale

The design is highly modular. The justification for this is in case other features need to be implemented into RoCM or RoCS. The data retrieval portion of RoCM will be handled by SOCM (and SOCMBridge for data parsing). Splitting up the plotting functions with CurvePlot.js and ChiChart.js will encapsulate the D3.js plotting functionality. Each model will be their own GalacticModel.js, with their own implemented velocity function. When using the global PARAMS object from the ParamsDict.js, the user will be able to use the parameters that come from SOCM within their custom galactic models.

4 Data Design

****See SOCM Software Requirements Specification****

5 Component Design

5.1 SOCMBridge

This component will take care of all data transfer from SOCM. The data will be formatted by the specified functionalities (for example: CurvePlot will format the incoming data to plot using D3.js).

5.2 SOCMTable

The formatted drop-down table generation will happen within this component. It will receive it's data from SOCMBridge, and create a dynamic table accordingly.

5.3 CurvePlot

This component will handle all D3.js plotting of the rotation curves. It comprises of two scale function objects for conversion from a value on a plot, to an screen coordinate. The curve function object will define the shape and inputs of the velocity curve. The svg object will be used throughout in order to draw the svg elements onto an HTML svg tag. The processing of data coming from SOCM will be handled in the get_data(data) function. Each component of the graph will be generated within their own function (Ex: legend, axes, title, etc.). Update functions for the velocity curves will be provided.

5.4 ParamSlider

The defined object, ParamSlider, will have two class variables, a string of the parameter name, and a Param object. The slider has a dynamic slide function which only updates the galactic models that use the changed parameter.

5.5 ChiChart

This component will generate and update the χ^2 statistical testing chart with the current model vs. the observed data. The χ^2 test is formulated by: $\frac{(O-E)^2}{E}$ where O is the observed data and E is the expected result from the model.

5.6 SVGSave

This component will take care of converting the CurvePlot svg element into a downloadable format and save it as an SVG file. The importance of a Scalable Vector Graphic file is due to the loss-less format of saving the image as a set of object vectors as opposed to individual pixels.

5.7 GalacticModel

The GalacticModel object will contain all of the equations for modeling a galaxy. The object has member variables of the model's name, the equation function (in the form of v(R)), the full formatted model name (for the CurvePlot legend), the line opacity, and the LaTeX equation. Four models will be included: General Relativity (GR), Λ-Cold Dark Matter (DARK), GR + DARK (TOTAL), Conformal Gravity (CONFORMAL), and Modification of Newtonian Dynamics (MOND). The models can be used throughout the code with the global MODEL object (Ex: cg = MODEL[“CONFORMAL”] will get the Conformal Gravity equation function and can be used by cg(10))

5.8 Bulge

This object formulate the bulge contribution of a galactic model. The inner bulge of a galaxy has to be calculated separately from the thin disk.

5.9 ParamsDict

The dictionary of Param objects will contain the key (parameter name) and value (Param object) pairs. Additional helper function will be provided in order to implement a standard dictionary. The global PARAMS object can be used in user defined galactic models to access the parameters retrieved from SOCM (Ex: R0 = PARAMS[“R0”] will get the value of R0 from SOCM).

5.10 Param

The Param object will define each individual parameter retrieved from SOCM. It has member variables of the float value, the units, and the range of values for the ParamSlider (denoted by min and max).

5.11 Constants

The component will define a global CONST object that contains constant values to be used in any GalacticModel. To access the CONST value, use this syntax CONST[“c”] (this will get the speed of light in kilometers).

5.12 Conversion

In order to convert units within RoCM, a custom Conversion object will be provided. Only necessary conversions will be implemented. Conversions from kiloparsecs (kpc), kilometers (km), and centimeters (cm) will be available. For the dark matter model, the conversion from $\frac{GeV}{cm^3}$ to $\frac{kg}{km*s^2}$ will be available.

5.13 LatexEquation

This component will handle the conversion from LaTeX code to formatted text via the MathJax library.

5.14 RoCS

The Rotation Curve Simulation will handle all rotation visualizations of the galaxy. A color_scale will be used to display the speed of the star with a color association. The size_scale function will convert the size of the stars relative to their position from the center (for a better visualization). The simulate function will handle the calls to generate and rotate the stars.

6 Human Interface Design

6.1 Overview of User Interface

The GUI will be very minimal, in order to not over complicate the software. A drop-down table will be present to access the SOCM data. From there the user can plot or download that raw data. The Rotation Curve Modeler section will house the CurvePlot functionality and enable the user to *Simulate* the galaxy's rotation or *Save to SVG* the SVG element of the graph. A parameter table will be included to show the current selected galaxy's parameters.

The Sliders section will contains the parameter sliders and the ChiChart. The sliders will be created dynamically by the user, with an additional + button.

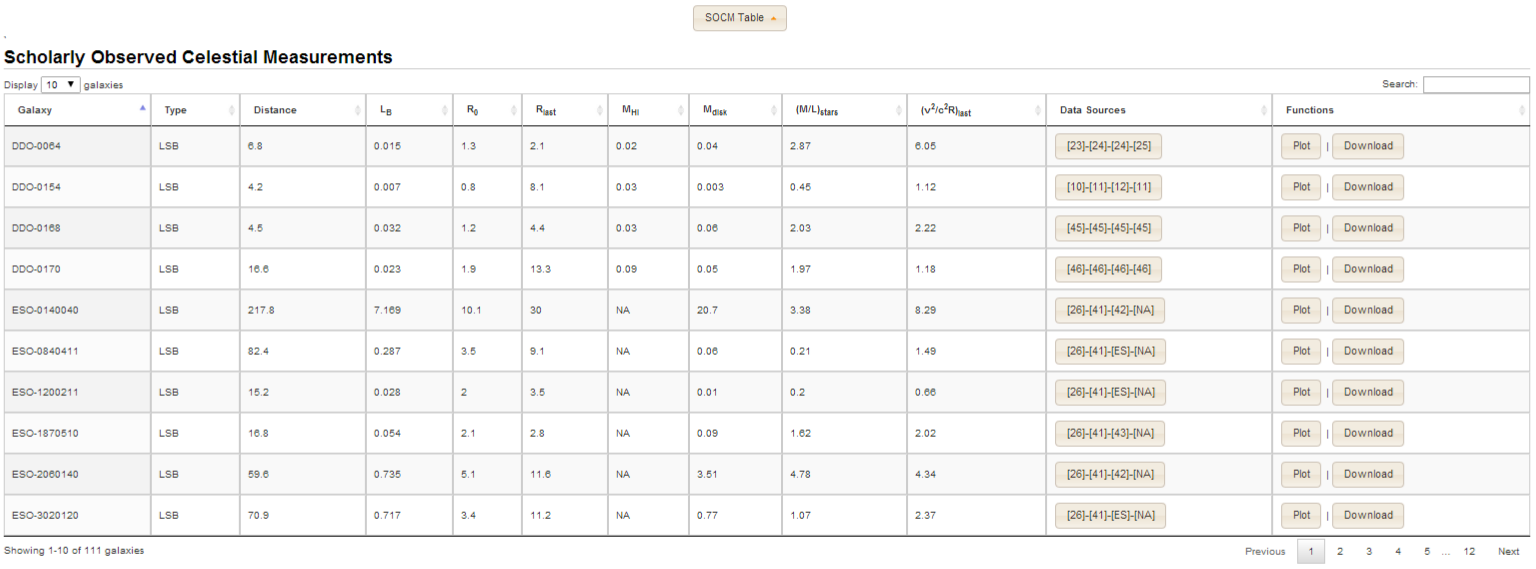
The LaTeX equation section of the web page will show the individual equations for each galactic model.

Lastly, the About section of the web page will contain information of each component and a tutorial for users on how to operate the software.

For the Rotation Curve Simulation web page, the simulated galactic rotation will be centered with a color scale or the star velocities. A slider for the speed of the spin will enable the user to set the rotation speed accordingly.

6.2 Screen Images

6.2.1 Scholarly Observed Celestial Measurements Table (SOCM Table)



Scholarly Observed Celestial Measurements

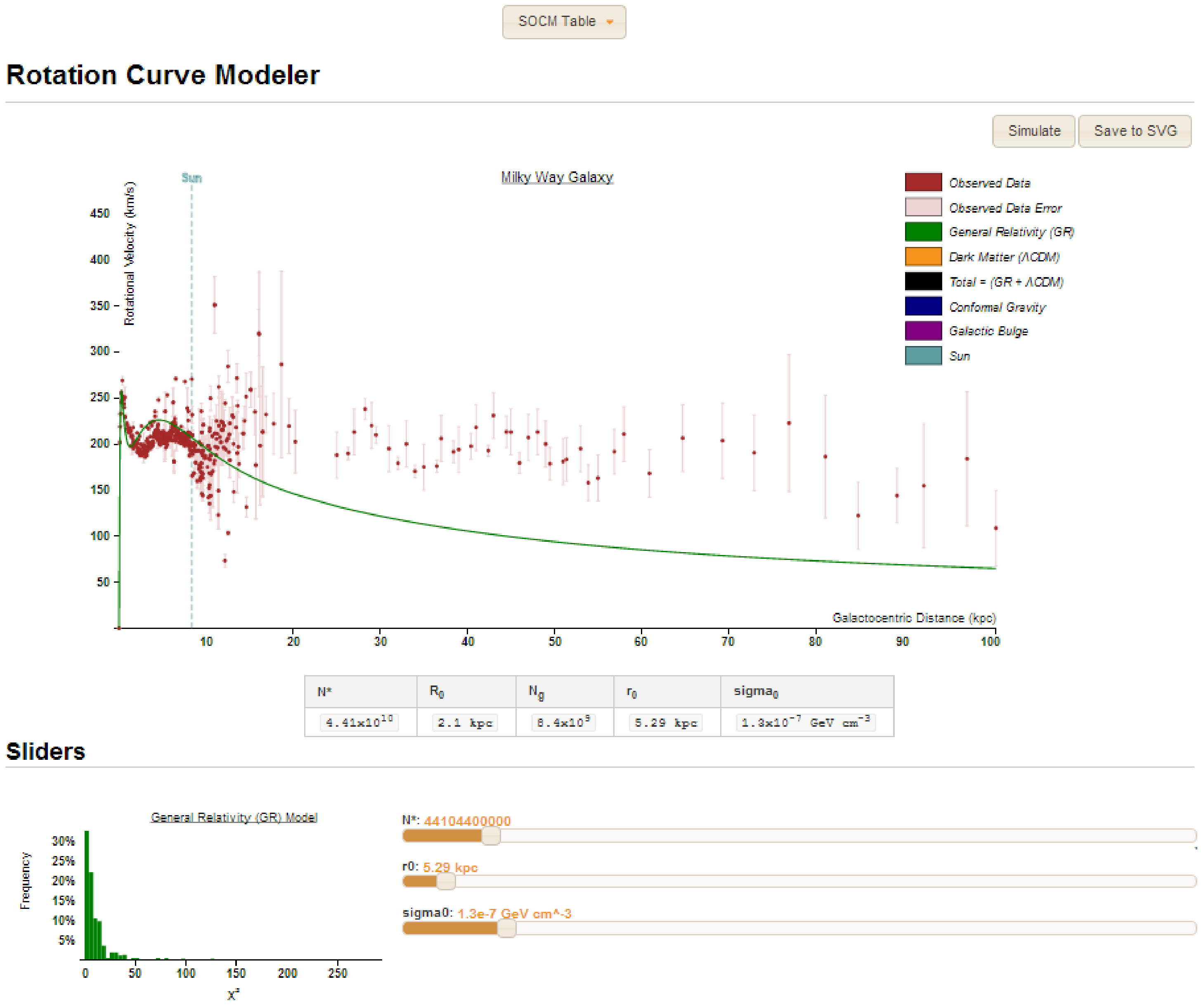
Display 10 galaxies

Search:

Galaxy	Type	Distance	L_B	R_0	R_{last}	M_{HI}	M_{disk}	$(M/L)_{stars}$	$(v^2/c^2R)_{last}$	Data Sources	Functions
DDO-0064	LSB	6.8	0.015	1.3	2.1	0.02	0.04	2.87	6.05	[23]-[24]-[24]-[25]	Plot | Download
DDO-0154	LSB	4.2	0.007	0.8	8.1	0.03	0.003	0.45	1.12	[10]-[11]-[12]-[11]	Plot | Download
DDO-0168	LSB	4.5	0.032	1.2	4.4	0.03	0.06	2.03	2.22	[45]-[45]-[45]-[45]	Plot | Download
DDO-0170	LSB	16.6	0.023	1.9	13.3	0.09	0.05	1.97	1.18	[46]-[46]-[46]-[46]	Plot | Download
ESO-0140040	LSB	217.8	7.169	10.1	30	NA	20.7	3.38	8.29	[26]-[41]-[42]-[NA]	Plot | Download
ESO-0840411	LSB	82.4	0.287	3.5	9.1	NA	0.06	0.21	1.49	[26]-[41]-[ES]-[NA]	Plot | Download
ESO-1200211	LSB	15.2	0.028	2	3.5	NA	0.01	0.2	0.66	[26]-[41]-[ES]-[NA]	Plot | Download
ESO-1870510	LSB	16.8	0.054	2.1	2.8	NA	0.09	1.62	2.02	[26]-[41]-[43]-[NA]	Plot | Download
ESO-2060140	LSB	59.6	0.735	5.1	11.6	NA	3.51	4.78	4.34	[26]-[41]-[42]-[NA]	Plot | Download
ESO-3020120	LSB	70.9	0.717	3.4	11.2	NA	0.77	1.07	2.37	[26]-[41]-[ES]-[NA]	Plot | Download

Showing 1-10 of 111 galaxies

6.2.2 Rotation Curve Modeler (RoCM) Web Page



Rotation Curve Modeler

N^*	R_0	N_g	r_0	sigma_0
4.41×10^{10}	2.1 kpc	8.4×10^{9}	5.29 kpc	1.3×10^{-7} GeV cm^{-3}

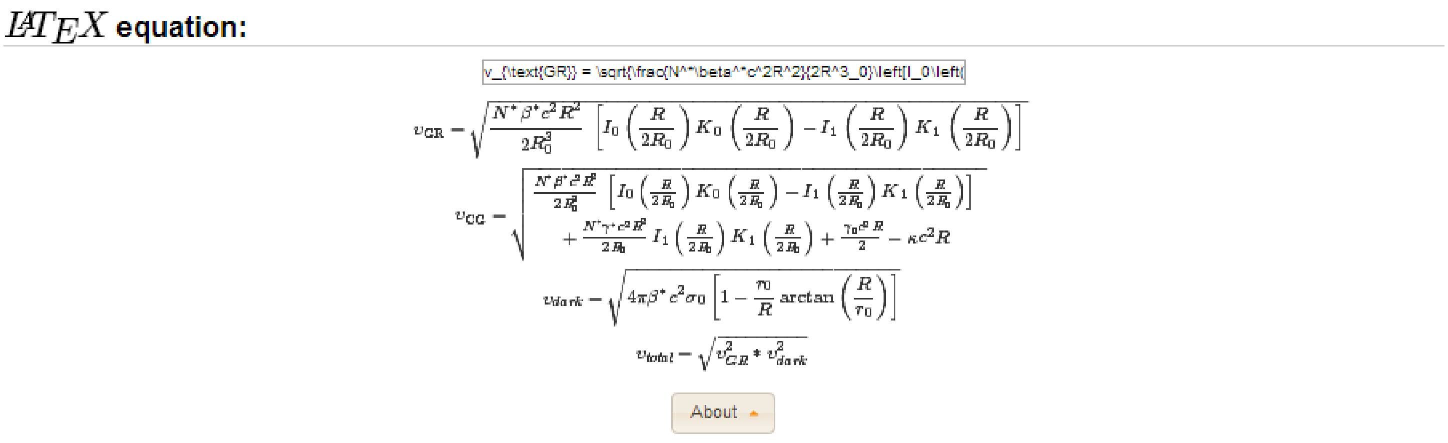
Sliders

N*: 44104400000

r0: 5.29 kpc

sigma0: 1.3e-7 GeV cm^-3

6.2.3 Latex Equation Viewer



LaTeX equation:

```
v_{\text{GR}} = \sqrt{\frac{N^*\beta^*c^2R^2}{2R^3_0}\left[I_0\left(
```

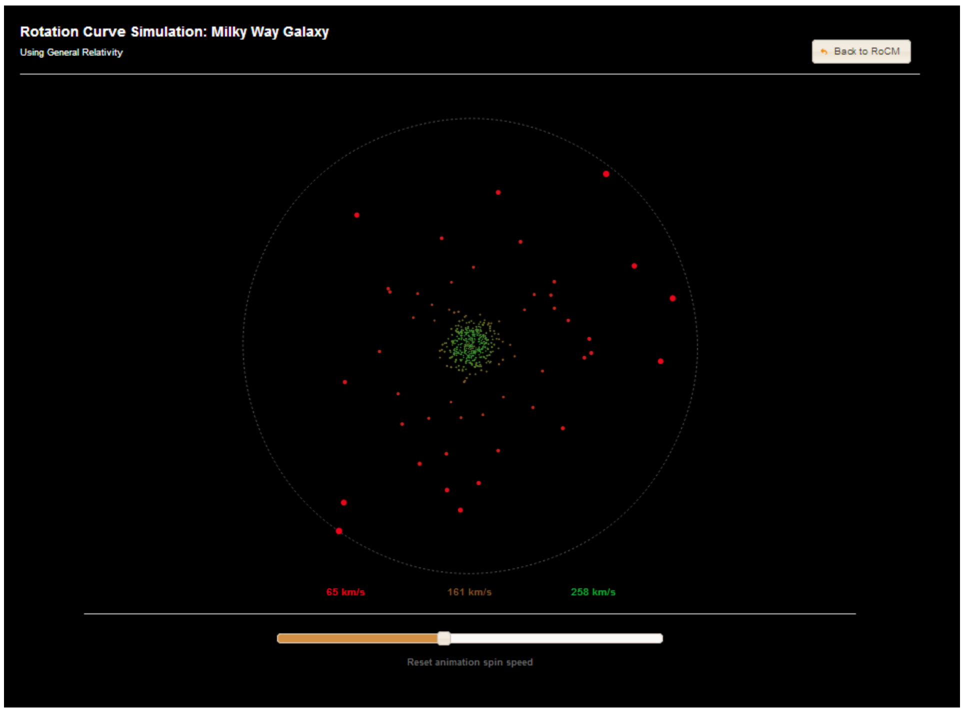
$$v_{\text{GR}} = \sqrt{\frac{N^*\beta^*c^2R^2}{2R_0^3}\left[I_0\left(\frac{R}{2R_0}\right)K_0\left(\frac{R}{2R_0}\right) - I_1\left(\frac{R}{2R_0}\right)K_1\left(\frac{R}{2R_0}\right)\right]}$$

$$v_{\text{CG}} = \sqrt{\begin{array}{c}\frac{N^*\beta^*c^2R^2}{2R_0^3}\left[I_0\left(\frac{R}{2R_0}\right)K_0\left(\frac{R}{2R_0}\right) - I_1\left(\frac{R}{2R_0}\right)K_1\left(\frac{R}{2R_0}\right)\right] \\ + \frac{N^*\gamma^*c^2R^2}{2R_0}I_1\left(\frac{R}{2R_0}\right)K_1\left(\frac{R}{2R_0}\right) + \frac{\gamma_0c^2R}{2} - \kappa c^2R\end{array}}$$

$$v_{dark} = \sqrt{4\pi\beta^*c^2\sigma_0\left[1 - \frac{r_0}{R}\arctan\left(\frac{R}{r_0}\right)\right]}$$

$$v_{total} = \sqrt{v_{GR}^2 + v_{dark}^2}$$

6.2.4 Rotation Curve Simulation (RoCS) Web Page


Rotation Curve Simulation: Milky Way Galaxy
Using General Relativity
Back to RoCM
65 km/s
161 km/s
258 km/s
Reset animation spin speed