Following Asteroids

The Asteroid Group:

Cole Buhman: BuhmanCole@gmail.com

Michael Hoffmann: mghoffmann@gmail.com

Andrew McMullin: mcmullinboy15@gmail.com

Background and Motivation

NASA provides a RESTful web service called NeoWs (Near Earth Object Web Service), where they publish data on near earth objects (NEOs) every day. This data includes proximity to earth, approximate diameter, velocity relative to earth, whether NASA considers a NEO a potential threat to Earth, and a list of close approaches to Earth for each NEO.

When we discovered this dataset we were excited by the potential insights it could provide and wanted to create visualizations to answer some questions about it. We want to be able to bring this data to the users in a way that they'll understand.

Project Objectives

We would like to create an interactive visualization that helps the user understand the answers to the following questions about near earth objects tracked by NASA.

- Is the frequency of potentially hazardous NEOs changing over time? Are we more likely to share the fate of the dinosaurs than our ancestors were?
- Is the average diameter or velocity of observed NEOs changing over time?
- Is there a positive correlation between NEO diameter or velocity, and closeness to Earth?

Users will benefit from understanding this data by having a more firm grasp of the events occurring in our solar system. This understanding should be available to everyone because it should inform public policy relating to emerging endeavors such as space exploration, asteroid mining, the search for alien life, and defending Earth against fatal collisions.

Data

We are using NASA's NeoWs service, a RESTful API that provides information about asteroids and other near earth objects. The data is in JSON format and is requested using start date and end date URL query parameters with an API key. NASA's endpoint is at https://api.nasa.gov/neo/rest/v1/feed and their instructions for using it can be found at https://api.nasa.gov/#asteroids-neows.

The Data comes from the API in a JSON format that provides a dictionary of days that have a list of Asteroids for that day as it's value.

The key attributes that we will be using for this project are:

- `name : string`
- `id : number`
- `estimated_diameter : number`
- `is_potentially_hazardous_asteroid : true`
- Each NEO has a `close_approach_data` array with an entry for each time the NEO got close to its `orbiting_body`, which we can use to visualize trends in events of close occurrence.
  - `close_approach_data:`
    - `close_approach_date : Date`
    - `relative_velocity: number`
    - `miss_distance : number`
    - `orbiting_body : string`

Data Processing

We expect to have to do substantial cleanup and agglomeration of the data into forms that are usable to answer our questions. The attributes that we plan to get from our data are written above, but they are nested in child objects and some attributes will need to be averaged over the filtered time period because a NEO may be listed in more than one day's report.

Our goal is to retrieve our data using an API instead of downloading the data. This will allow the webpage to show the most recent data from the dataset. We will do this using the json function provided by d3.js, as documented at https://github.com/d3/d3-fetch/blob/v2.0.0/README.md#json.

An edge case presented by using dynamic data retrieval is when the user enters filters that do not yield any data. Our visualizations will need to indicate that no data was found rather than simply not displaying marks and appearing malfunctional.

Must-Have Features

- The user must be able to select a time period and have data about NEOs observed during that period displayed.
- A scatterplot will show the relationship between the mass of NEOs and how close they get to Earth.
- A line chart will show the number of NEOs observed for each day of the selected period.
- A visualization of an idiom we haven't decided on yet will express the closest distance of each NEO to Earth observed during the selected period.

Optional Features

- The project will request data from the NASA API in real-time according to user filters, instead of using local copies of the data. This will allow it to stay up to date as the data changes.
- A visualization will show the relationship between the month of the year and the number of NEOs reported during that month.

Project Schedule

The data for this project comes in the form of lists of near-earth objects (NEOs), grouped by date. Each NEO is listed under at least one day and each NEO is also given a unique name as an identifier. Making that data useful for our objectives will require three discrete pieces of software:

1. An interface that allows the user to filter by selecting a time period and other attribute values, and allows the user to select individual NEOs to view details about them.
2. JavaScript code that uses the filter criteria to send requests to NASA's API, and then filters and agglomerates the data into objects that expose the attributes we are interested in visualizing.
3. HTML and JavaScript/D3 code that takes the filtered data and uses it to update visualization views. As indicated in our prototype diagrams we have decided to display the data in small multiples, so each view will need to be connected to the filtered data and updated as filters and selections are changed and new data is provided.

These 3 modules of the project will each need to be at least basically functional in the prototype, which is due November 21st, and they will all need to be fully functional and polished before the final submission is due on December 12th. We plan to divide the work among the three group members, making sure integration is successful by relying on the objectives documented below. These integration criteria will give each team

member specific requirements that their module must fulfill and establish data formats that we plan to use to send data from the filter interface to the request module and then from the request module to the visualization interface.

1. The filter interface will let the user input the filters in the table below and provide ways for the other modules to access the filter values. For the prototype, the filter interface will at least provide the access methods below even if they only return dummy values.

Filter	**Effect**	**Conditions**	**Access Method**
Time Period	A start and end date. Only NEOs who are reported to be near earth on or between these dates will be displayed.	The start date must be before or on the end date. The end date must be no later than the current date.	A function called `filterDate` will take a `Date` as a parameter and return `true` or `false`. This will allow us to extend the filter later with options for day of week, time of year, etc. to provide interesting insights.
Minimum Diameter	Only NEOs whose diameters are greater than or equal to this value will be displayed.	Must be between 0m and a maximum established by the available data- probably some fraction of the diameter of Earth.	A function called `getMinimumDiameter` will return the minimum diameter as a `number`.
Minimum Velocity	Only NEOs whose velocities are greater than or equal to this value will be displayed.	Must be between 0KPH and a maximum established by the available data. The module owner will need to scrape the data to discover a maximum.	A function called getMinimumVelocity will return the minimum velocity as a `number`.
Only Potentially Hazardous	Only NEOs which NASA has indicated are potentially hazardous are displayed.	Must be true or false.	A function called `getOnlyHazardous` will return `true` or `false`.

Table 1: Requirements for the filter interface.

2. The request module will use the access methods above to get filter data with which to make requests to the NASA API, and then provide a JavaScript array of objects with the properties shown in the table below. Each NEO should be represented by exactly one object in the array. Therefore, in cases where a NEO with the same ID appears in multiple days, the data will need to be combined or averaged across those days as appropriate for each attribute's data type. The resulting array of objects will be passed as a parameter to a function called `updateVisualizations`, which the request module should call after it completes the API request and agglomeration and filtering.

Property	**Type**	**Description**
id	number	The id of the NEO. This should be unique for each NEO.
name	string	The name of the NEO.
magnitude	number	The absolute magnitude (h) of the NEO.
potentially_hazardous	boolean	Whether the NEO is potentially hazardous.
close_approaches	array	An array of every close_approach_data value given by the NASA API for the NEO.
closest_approach	number	The closest this NEO has gotten to earth- the minimum `miss_distance` distance of its `close_approaches`.
closest_approach_date	Date	The date that the `closest_approach` occurred.
average_velocity	number	The average of all of the velocities of this NEO's close approaches. This is a relative velocity to Earth.

Table 2: Properties of the objects that the request module needs to create.

3. The visualization interface will display the NEOs given in an array as a parameter to the `updateVisualizations` function. The section below explains which visualizations we may use to show the data resulting from the filters.

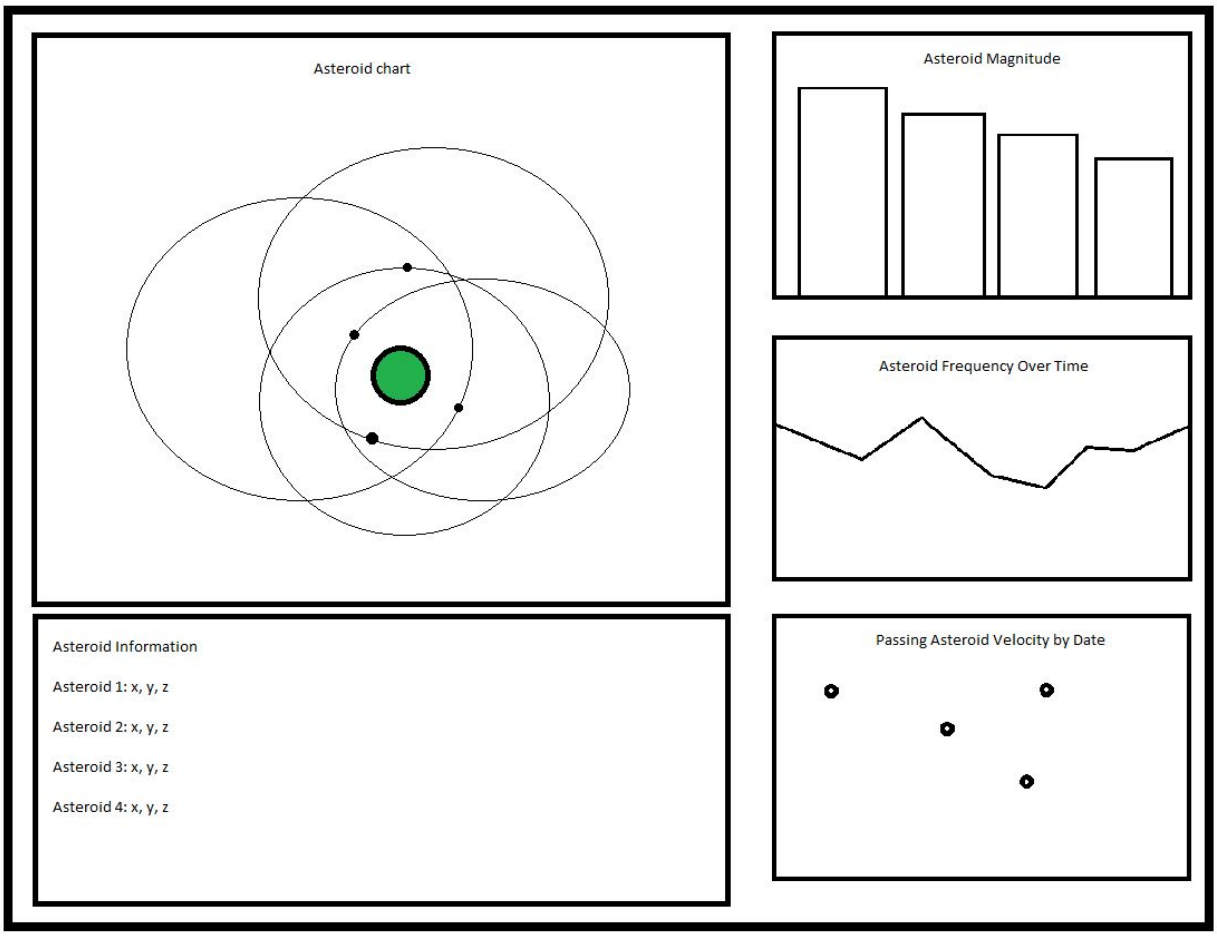
   This module also needs to let the user select individual NEOs from an overview of the filtered data and display their attributes in a details view.

Visualization and Design

We want to use small multiples to show relationships between attributes of NEOs filtered by the user, as detailed in our Project Objectives above, and also an overview-details design to let the user explore individual NEOs. The most prominent visualization will show the nearest approach distance to Earth of each NEO in the filtered data, which will help the user discover trends over time for their selected period.

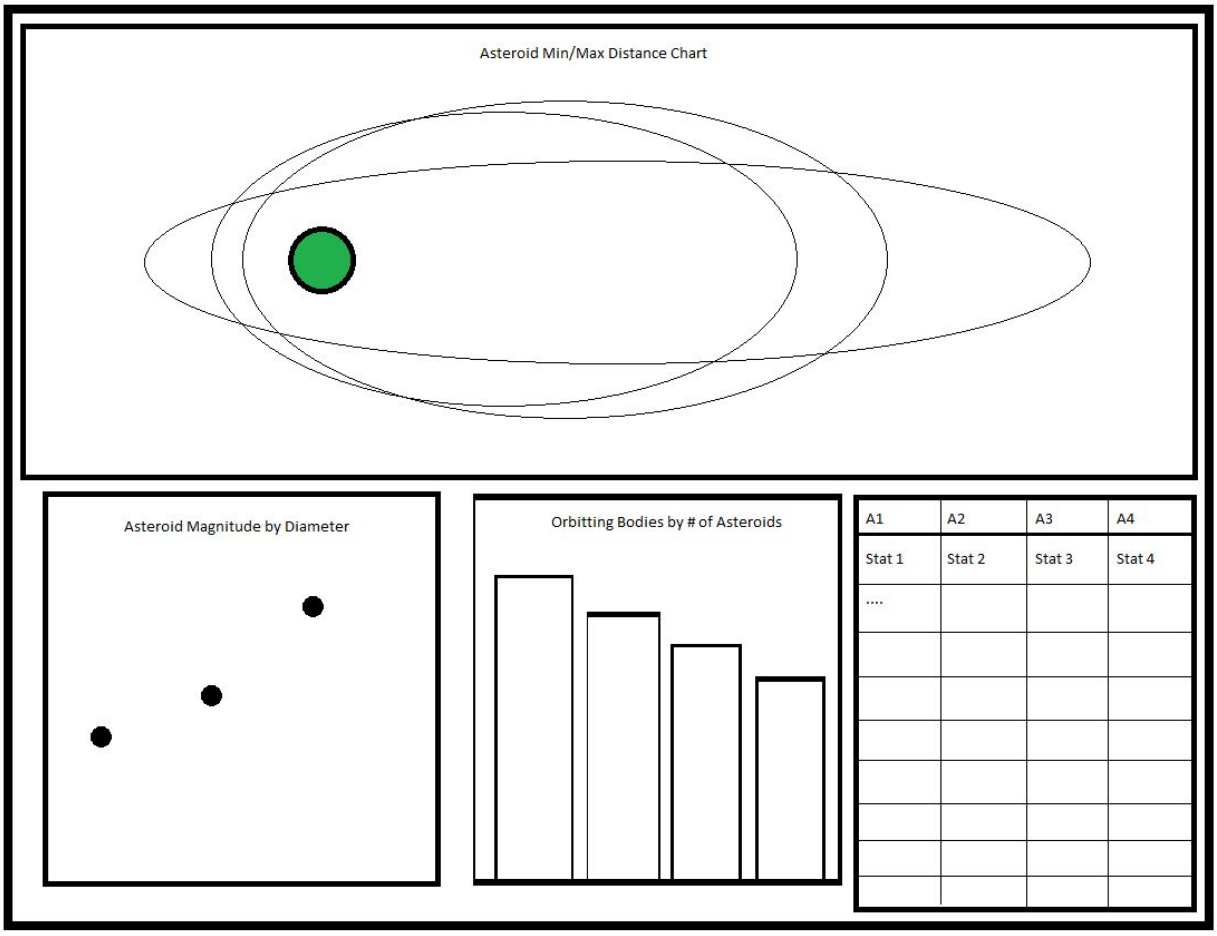
We would like to make this view aesthetically pleasing by mimicking 3-dimensional orbital patterns around an image of Earth, but more research and experimentation is needed to come up with a way to do this using only 1-dimensional proximity data without violating the expressiveness principle. One potential alternative is to use a radial bar chart or another idiom to indicate close approach distances from earth, but this may not scale well if the user's filter criteria select more than dozens of data points. We want this view to be the focus of the visualization because large differences in distance to Earth will lead the user's attention to closer approaches. This will invite them to explore individual NEOs and proximity trends, and emphasize the frequency of our potentially hazardous encounters, which may be unexpectedly high.

Prototype 1:



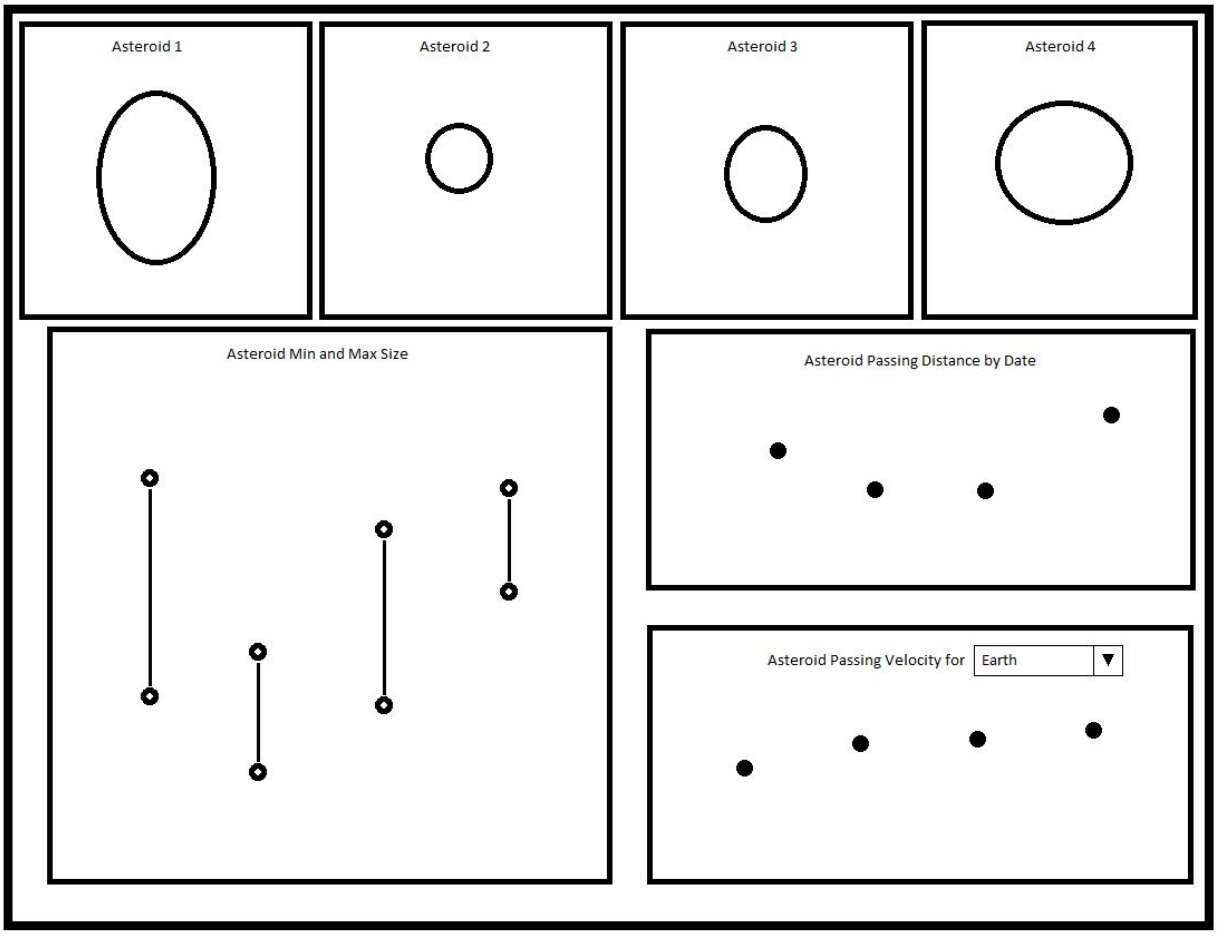
This prototype shows a chart for real-world context as well as several attributes and an info panel for individual asteroid data.

Prototype 2:



This prototype has an adjusted chart to focus on the minimum and maximum distance from Earth. It also shows a couple attributes and a table rather than an info panel.

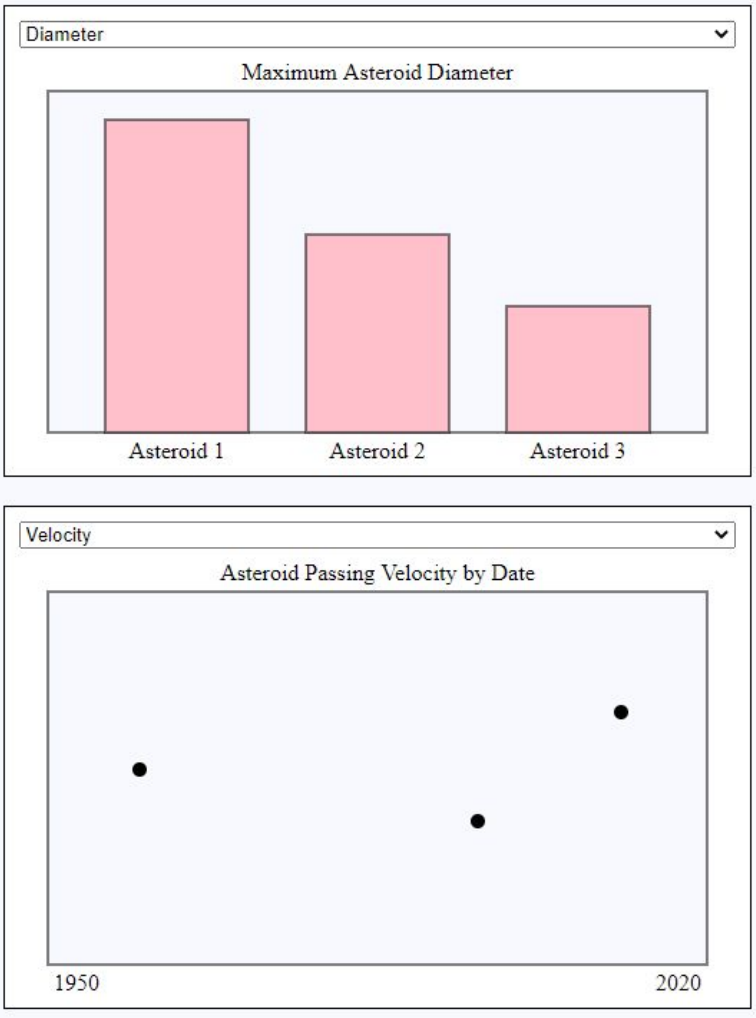
Prototype 3:



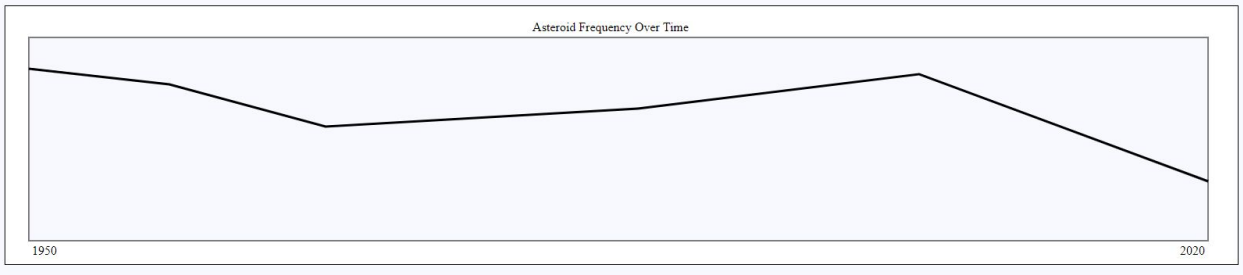
This prototype shows the asteroids as circles with diameters relative to their real world diameters. It also shows this information in a chart for a comparison on a uniform scale and it has two attribute charts, one of which is interactive.

Final Design, Overall Layout:

Final Design Attribute Charts:



Final Design Footer:



Asteroid 1	Asteroid 2	Asteroid 3
ID: X	ID: X	ID: X
Magnitude: X	Magnitude: X	Magnitude: X
Diameter: X	Diameter: X	Diameter: X
Passing Velocity: X	Passing Velocity: X	Passing Velocity: X

The final design incorporates the asteroid display from the first prototype since works well as an overview, with some of the other charts showing details of the selected NEOs and some serving as small multiples for the filtered data. Each detail chart will use axes scaled to the filtered data and use highlighting to show which values correspond with which asteroid.

The design also has two charts which can be adjusted to fit different attributes. The first one is for static data like diameter, while the second one shows data relative to the time the asteroid passed by Earth. This way users can easily see how asteroids compare to each other while also getting an idea of any trends over time.

The footer contains an info panel for summaries of individual asteroids as well as a frequency chart that can be brushed for a summary of the data over a time period. The info panel will make it easier to access exact figures without hovering over the charts, as well as showing IDs and any important categorical data that won’t be in the bar charts.