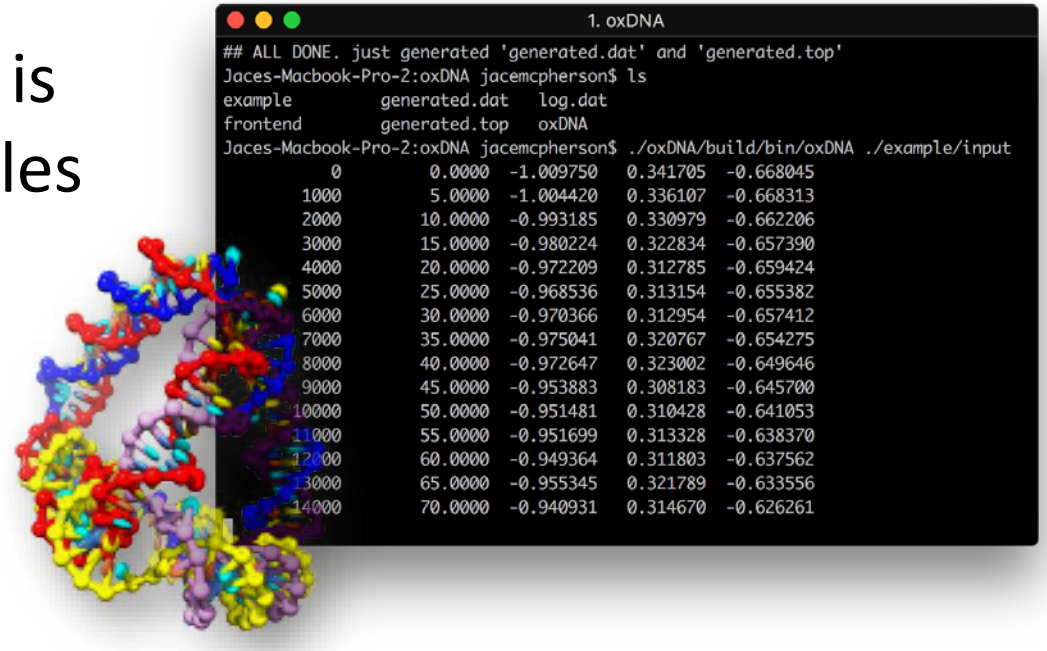
WebDNA Project Proposal

Jace McPherson, Salvador Sanchez, Zack Fravel,
David Darling, Jonathon Raney

November 6, 2017

Premise: Here's the Problem…

- oxDNA is a widely used DNA simulation software.
- It's clunky, and very inaccessible to most DNA researchers.
- Analysis of the oxDNA output is the main time-consumer for DNA related research.
- Because it lacks a UI, file management is also an issue. oxDNA produces many files and it's up to the user to figure out how to organize them.
- Difficult visualization of simulation results



Objective: How can we fix this?

- We plan to wrap oxDNA functionality with a website
- All input data, simulation visualization, and data analysis should take place in our controlled environment
- Guide the configuration experience with a clean, sensible UI
- Allow advanced users to download their results and perform offline analysis, if necessary.

Design Goals

1. Provide simple user interface for generating input data to the oxDNA simulation environment.
2. Allow users to modify the execution parameters of the simulation
3. Provide a visualizer to show the DNA system state at different points in time.
4. Provide a visual *analysis platform* so that the user can easily perform analysis with no code experience necessary.

I promise, these will be clearer in a second…

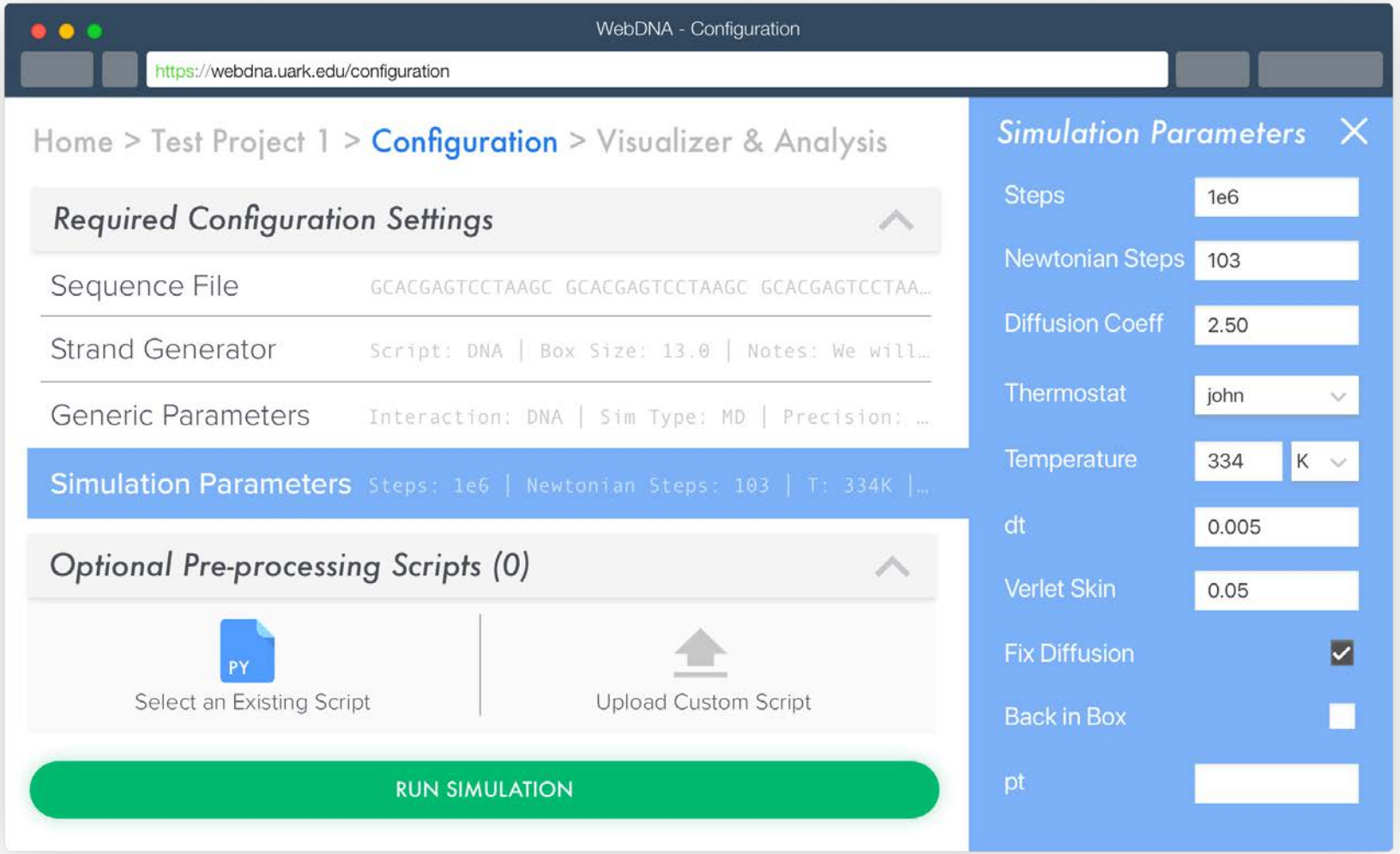
Simulation Configuration

Required Settings

- Normally set up with Python scripts in a clunky workflow
- Now, reduced to simple series of steps with *same functionality*

Optional Settings

- Allows users to run built-in and/or custom scripts on their input



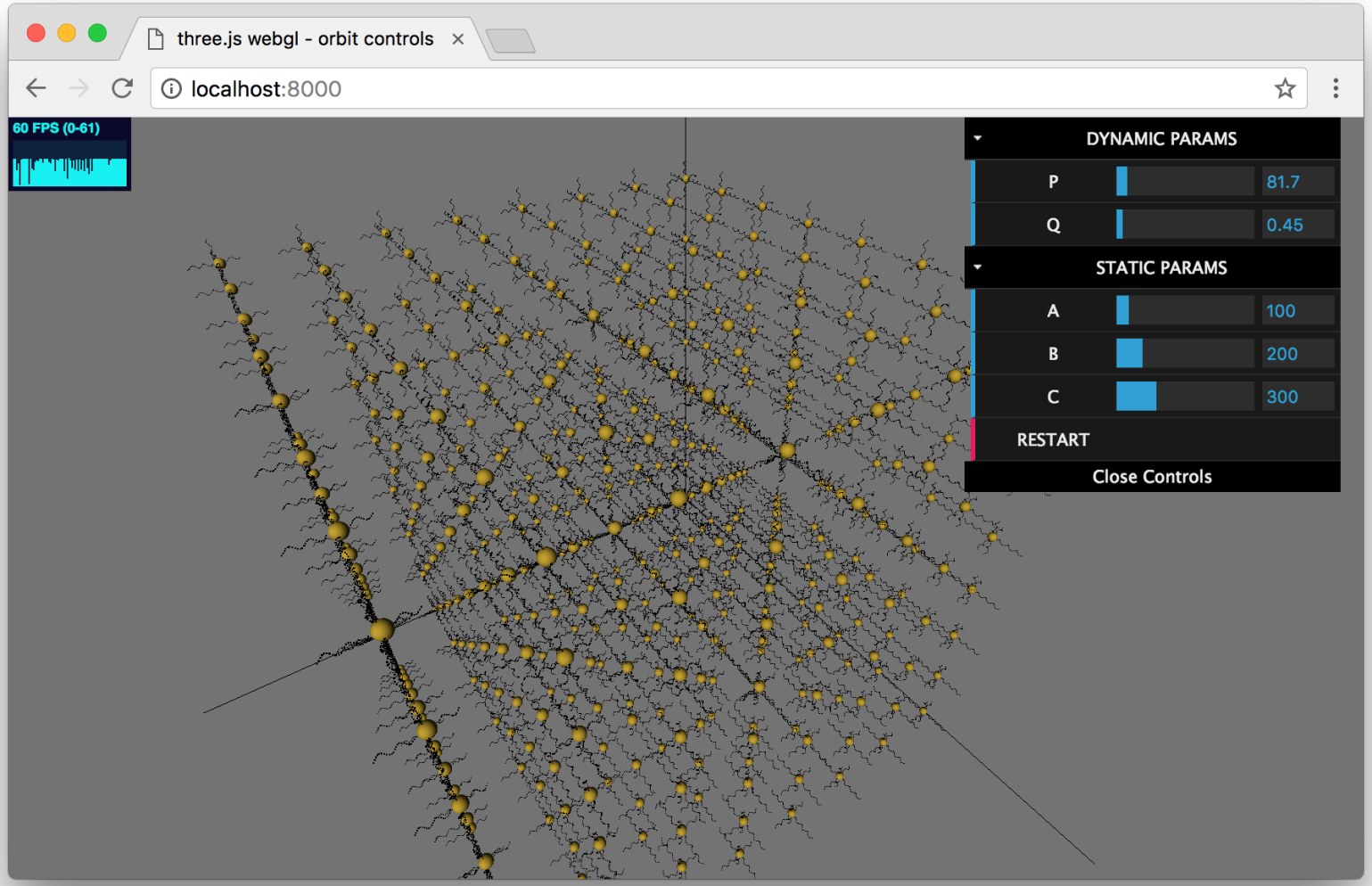
Simulation Visualization

Current Visualizer

- Simple nanoparticle visualizer already implemented
- We can use the current visualizer as a base for our more flexible oxDNA simulation visualizer.

Added Features

- Scrubbing through simulation over time



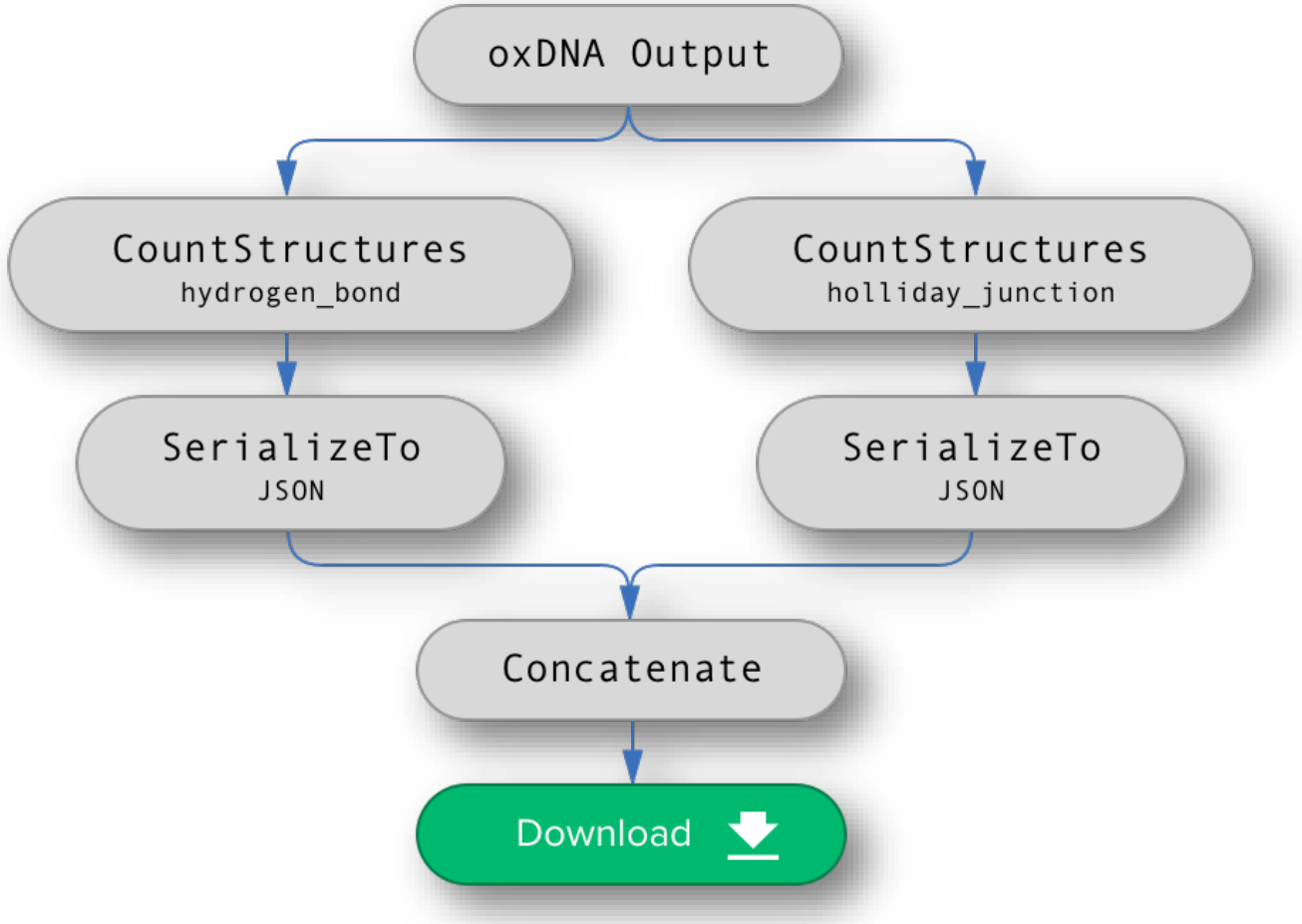
Analysis Editor

Current Analysis Pipeline

- NONE! Researchers have to write their own analysis code to extract meaningful data from simulations

Our Analysis Solution

- Create a drag-and-drop style interface for users to pipe their simulation data through a series of analysis steps
- Allow downloads of data at *each of the steps*, including the final output, for full customizability.



Technologies

Server

Django Python Server

- Listens for HTTPS requests
- Serves web pages using the Django HTML renderer

PostgreSQL DBMS

- Widely used SQL extension

Client

Django Python Renderer

- Renders HTML, CSS, and Javascript using logic written in Python

Three.JS

- 3D Renderer for browsers (using WebGL)