Proposal for Production of a Lattice QCD Data Visualization Code Base

1 Project Goal

The goal for this project is to produce a code base that will provide all members of the Lattice QCD (LQCD) group within FRIB the ability to perform basic LQCD data analyses and to visualize the analyses data in a clear, efficient and robust way. As a byproduct of the data analysis, it will also standardize our data formatting across our group members.

2 Learning Outcomes

The major motivation for this project is to provide our group with some skills in producing a code base in a group environment. This project will also be useful for helping and sharing ideas and concepts of coding between group members. Since this code base will be written in python (see Sec. 4), utilizing some packages, we will gain the ability to use python and the packages required for the project.

3 Members and Time-frame

The development will be performed by Jack Dragos, Jangho Kim, Giovanni Pederiva, Sam Liao, Matt Rizik and Mathias Vege (through zoom co-ordination).

The allocated time for development will be in our coding group meetings, typically held every Friday 10:30 am for approximately 1.5 hours.

UPDATE meeting 28th September, 2018: changed meetings to be allocated for UPDATE meeting 28th September, 2018s on the project and to plan for the proceeding week of work. This means that the coding will be done between meetings.

A time frame review will be performed in 3 months time (7th December) to review progress, discuss any alterations in design and to determine if the project will be continued.

This proposal was produced in the first meeting, the following meetings are planned to undertake the following:

- Meeting to understand bokeh and pyvis python packages for plotting (see Sec. 4.3).
- Meeting to discuss data structure and IO.
- Meeting to decide on delegating sections of code to group members.

4 Implementation

4.1 Code Structure/Graph

Referring to fig. 1, data is main class type for reading in data and all our types of data (e.g. two-point correlators, flowed opeartors etc...) is inherited from Data. A "pyvis settings" class will then take in a Data object instance which will control the GUI interface for data visualization. IO will be done in pyvis, which will link to IO within the Data class.

4.2 Data IO Standardization

The Data class and each child subclass will have its own IO function. Along with this, the "pyvis settings" class will have its own IO class, which will link to the IO of the Data objects used in the figures.

All storage of Data objects will be in hdf5 for efficiency and usability. The data stored for pyvis will (most likely) be in JSON formatting for human readability.

UPDATE meeting 28th September, 2018: xarray data type has been addopted, this means netCDF (built on top of hdf5) is addpoted for binary data storage.

4.3 Language and Packages

This project will be written in Python version 3.6, with no planned backwards compatibility with version 2.7.

The project will use the following packages.

- numpy: Basic array and number manipulation
- pandas: Primary method for storing data in tabulated form. Provides inbuilt IO functions as well.
- bokeh: Back end for plotting, used in substitute for matplotlib to provide web based plot display and greater formatting and functionality.
- holoviews Package built on bokeh used for greater functionality for plotting.

4.4 Debugging, Testing and Test Sets

Since all types of data computed from CHROMA will be child classes of a master Data class, sample data for this class will be produced to be used for debugging Data and "pyvis settings" classes. We will need some raw CHROMA data to test IO for all the child classes of Data.

5 Roles/Delegation

This is to be decided in (the planned) 3 weeks time, after we have done some preliminary testing on pyvis, bokeh and discussed standardized IO data formatting.

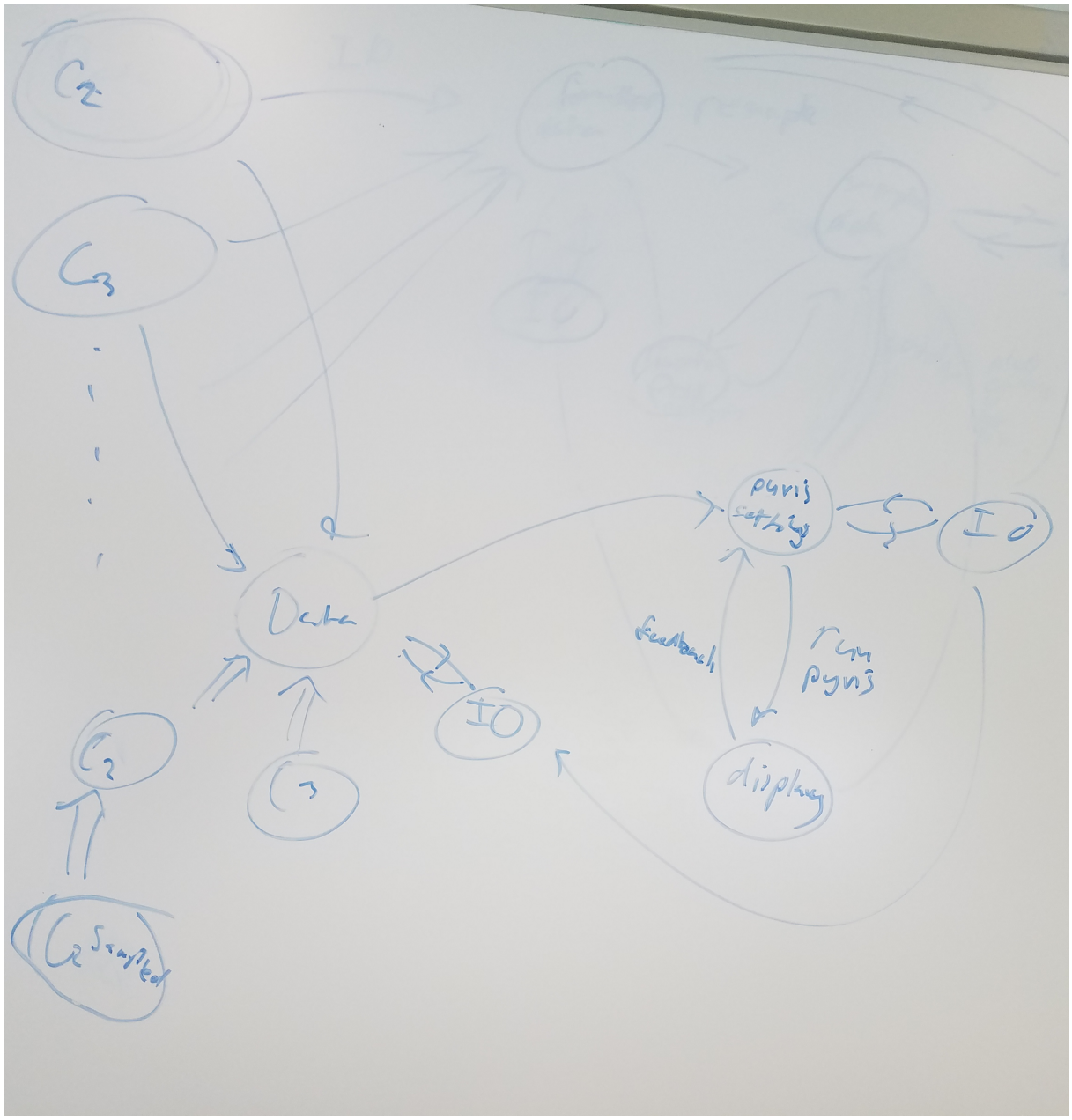

Figure 1: Code schematic for code base.

UPDATE meeting 28th September, 2018:

- Matt and Jack are delegated to the "data" part of the code.
- Giovanni and Sam will be working on the "visualisation" part of the code.
- Jangho has agreed to provide support and possible testing/debugging with the project.