GP2 Code Standardization Policy of the Global Parkinson's Genetics Program

Summary
Overall, the Global Parkinson's Genetics Program (GP2) aims to have a “no surprises” policy where analyses, projects, and manuscripts are coordinated and communicated in an open and transparent manner. GP2 wants the data and code to be used as widely and openly as possible, thus calling for a need to standardize the code across analysis teams. For all GP2 and external non-GP2 members who plan on conducting analyses on GP2 data, GP2 stresses the necessity of having clean, organized, and replicable code open to the scientific community. Code will follow the standardization guidelines, GP2 will be acknowledged, and code will be made available to the community via GitHub.

Introduction
The Global Parkinson’s Genetics Program (GP2) is an international effort aimed at generating significant insight into the genetic basis of Parkinson’s Disease (PD) and democratizing access to both results and data. GP2 is funded by the Aligning Science Across Parkinson’s initiative (ASAP, https://parkinsonsroadmap.org/) and it is part of this program’s strategic objectives to support collaboration, generate resources, and democratize data. Dissemination of GP2 code and data are major goals of GP2 and ASAP encourages the broad, rapid, and open publication of results.

Overview
The aim of this policy is to ensure that:
1. GP2 code and pipelines, from GP2 Consortium-led analyses, are disseminated fully, accurately, and promptly
2. GP2 code and pipelines, from GP2 Consortium-led analyses, are standardized, organized, and made publicly available on GitHub

Code standardization guidelines complement the GP2 publication policy. Please reference this document for GP2 consortium management of Unusual Situations.

The Code Standardization Policy outlines best practices for code standardization that are adopted by the GP2 Steering Committee and GP2-consortium led analyses complementing GP2-consortium led publications. External researchers using GP2 data are strongly encouraged to integrate these practices as well to ensure high calibre analyses and open science. Because of the field’s increasing inclination to use Python/Jupyter Notebooks instead of R, this document concentrates on Python. We acknowledge that many GP2 researchers may use R or other tools and continue to support organized, open, and replicable code in the GP2 community.

As broader motivation, please consider that while cleaning up code is tedious, this is how GP2 will be setting up their code, and you are encouraged to also follow these standards. If it is not clean/clear and can communicate effectively on its own, is it making the impact you really want? Clean code also means you spend less time downstream explaining and reexplaining analyses and concepts, it’s a win-win. Clean code enables replication and transparency. **GP2 is committed to open science, collaboration through code transparency, and inclusivity.**
Below, we describe some basic information and workflows in regards to Python as the primary language, Anaconda as the preferred Python distribution, PEP8 as the Python language structure to follow, storage on the Google Cloud environment, and analyses conducted on the Terra analytical cloud environment. We also stress the importance of having a clear document outlining workflow and rationale, known as a README, for each of the analyses. These are the standards that GP2 will follow, and hope the greater community will as well.

Before We Get Started, a Few Python and General Coding Basics

1. Everything computational for GP2 will likely be done in Terra, NHGRI's The AnVIL or similar cloud based infrastructure
   a. There will be analytical and financial support for projects / groups with logistic / financial constraints
2. Python3+ will be the main language for GP2 analyses; there will be limited support for R and other languages. This leverages the current code bases from the National Institute on Aging’s LNG Data Science Group.
3. To learn Terra, sign up for an AMP-PD account, this is a great getting started point
   a. They have some great “Getting Started” notebooks (where you write and run your code) as well as workspaces and plenty of data to use
   b. The getting started docs at Terra.bio itself are pretty great
4. To learn Python really well in two weeks...
   a. Run through this once a day
   b. Run through this once a week (from our good friends at the NSA via FOIA)
   c. Google Colabs is also a low overhead / low stress place to learn the notebook structure and become familiar with Python
5. Good code is simple, clear and standardized with a grammar, much like a recipe
   a. Google Style Guides are a great place to start
   b. PEP8 is the preferred “grammar” for Python
   c. Visual Studio Code is our preferred editor in GP2 as it includes extensions to stylistically standardize your code, but use whatever you are most comfortable with
6. The support teams’ main language is Python3, using the same tools as us allows us to both share tools with you and support your projects much more efficiently
   a. All internal builds for tools and coding are in Python3
   b. Shared GP2 resources will be in Python3 for the overwhelming majority of projects

How to Download Anaconda/Python for Local Use

1. It’s important to emphasize when we program with Python locally, we are using the Anaconda distribution to ensure a fully-featured Python3 environment
   a. LNG put together a document on how to get set up with Anaconda can be found here
Key Points and Overview

1. You **must** have some method of version control to prevent accidents. Think of this as your electronic lab notebook. The GP2 project lab has its own GitHub for forward-facing pipelines. As part of the general GP2 theme, everything here is public.
   - If not quite ready for the public, **that’s okay!** Make a private repository, and when the code is complete you have the option to push it somewhere public
   - GitHub makes version control and collaboration very simple, especially with their new Desktop interface, more documentation on how to get started [here](#)

2. Putting together a README is **crucial**. Our READMEs are not only how we communicate with others, but are also living documents, constantly changing, but always include information such as:
   - Authors, collaborators, project name, project goal, proposed workflow for analyses, date started, date last updated, paths to working directories, paths to files
   - A free and gorgeous Markdown editor to create READMEs is [stackedit.io](#)

3. The GP2 analysis team works together on experimental proof-of-concept codes using [Google Colabs](#) to make use of the free GPUs, easily share, and easily modify on the local Jupyter environment if need be.

4. As a general principal, when the collaborative group on a project is larger than 2 or 3 people who work together in the same group, version control is required.

5. Prior to sharing a project/manuscript with the GP2 network, version control and some type of transparent and intelligible coding resource is required.
   - This code will be audited prior to publication approval if GP2 resources in the form of analytics or financial support were used.

Organizing Our Code

1. A way to organize code, [explained here](#)
   - This resource explores making functions, modules, and other neat ways to compartmentalize your work
   - Your code does not need to follow this if you’re not comfortable, but some sort of intuitive organization should be followed
   - Consider your code as part of your manuscript, if it is not possible for a reviewer to understand your manuscript, how will it pass the GP2 open science standards? The same applies to code.
   - While there is no locked in format for code in GP2 please make every effort to be clear.

Code Standardization in Python

1. We follow, as closely as we can, to the Python Enhancement Proposal (PEP8) standards of Python programming. PEP8 is a document that outlines how to best write Python code, and aims to improve the readability and consistency of Python code.
   - A comprehensive but digestible resource can be found [here](#)
   - Most integrated development environments (IDEs) have a plugin to help ensure this standard
An example is Visual Studio Code’s autopep8 plugin.

2. In notebooks, we have a boilerplate template we use to help remind us and users of the motivations behind the projects. By using notebooks, Markdown is your friend to help orient the user while also keeping track of things ran.
   a. Here is our current boilerplate
      i. This can be tailored to your needs
      ii. The general framework should be used for code relating to projects receiving direct support from GP2. This helps with oversight and troubleshooting.

Cloud Platform Best Practices
1. As a part of AMP-PD and likely for GP2, we will be using the Terra platform to write, tweak, and run analysis pipelines.
   a. Terra was designed by the Broad Institute and MIT and leverages commercial cloud infrastructure(s, multiple options), and features a complete interactive Python3 environment in a notebook structure (much like Jupyter)
      i. If unfamiliar with Cloud computing, Google offers an excellent introduction to the lingo here, covering the basics of what is a project, bucket, container, and other key concepts to know moving forward
      ii. Terra-specific examples were put together by the AMP-PD team here
      iii. Some basics on how to move files between the cloud infrastructure can be found in this handy cheat sheet
      iv. Google’s best practices on naming, data storage, and data management can be found here

2. There are 2 key differences between running analyses locally vs. on the cloud.
   a. Locally, you pay for the storage (whether on your own computer or on an institution’s cluster) whereas on the cloud the storage is paid for (ex. By AMP-PD or by GP2)
   b. Locally, you do not pay to run analyses pipelines (never on your own computer, but paying for the cluster is institute specific) whereas on the cloud the users pay for each of the analyses they run (ex. via their PI or institution)

3. The AMP-PD and GP2 analysis teams have written a number of different analysis pipelines (link) that can be tweaked.
   a. Because you pay for the environment you set up and the amount of data you access, one key thing to do is to design and plan out your analyses as best you can
      i. An example of runtime costs that can be used as a guideline can be found here
   b. Because the cloud environment can be updated on the backend, it is important to log and version the buckets you have access to directly refer to them later
   c. Because you pay for the environment you set up and the length of the time you use it for, put it on pause or delete it all together when not in use to only pay for what you used
A Final Comment to Our Friends doing “Wet Work”

Transparency, openness and reproducibility are not just for computational endeavours. We highly recommend tools like protocols.io or using a Rocket Notebook for standardizing lab tasks and experiments. GP2’s Complex Disease-Data Analysis Working Group and Complex Disease-Data and Code Dissemination both would like mandated wet lab practices that mirror our work on the computational end of the spectrum. We are also pushing the sample acquisition and sample naming practices to be standardized online and transparent, even though they are not directly under our scope of work.

Acknowledgements

All manuscripts, preprints, and abstracts/posters that result from use of GP2 data analyses pipelines must acknowledge ASAP and GP2 using the following language:

“Code used in the preparation of this pipeline was obtained from Global Parkinson’s Genetics Program (GP2). GP2 is funded by the Aligning Science Against Parkinson’s (ASAP) initiative and implemented by The Michael J. Fox Foundation for Parkinson’s Research (https://parkinsonsroadmap.org/gp2/). For a complete list of GP2 members see https://parkinsonsroadmap.org/gp2/.”

The Michael J. Fox Foundation, ASAP, and the GP2 Steering Committee maintain the right to modify terms of this agreement, and may do so by posting notice of such modifications on this page. Any modification made is effective immediately upon posting the modification (unless otherwise stated). You should visit this page periodically to review the current use agreement terms.

“Coders’ code of conduct”

Please mirror this conceptual framework for collaborating on coding projects in GP2.