README (October 13, 2022)

This folder contains Julia code to replicate the simulation exercises and empirical calculations in “Stratification Trees for Adaptive Randomization in Randomized Controlled Trials”. The code was last run in Julia 1.6 and R 4.2.0 on the Acropolis Computing Cluster at the University of Chicago. The Acropolis cluster consists of 32 compute nodes. The compute nodes are Dell M630s. Each compute node consists of “Intel(R) Xeon(R) CPU E5-2690 v4 @ 2.60GHz” CPUs with a total of 28 cores or 56 computational threads and 250GB of RAM. The folder contains files which are designed to be run on a personal computer (however see below for caveats), as well as files designed to run the simulations on a computing cluster. On a computing cluster, computation times range between a few hours (for Model 1 with with a pilot sample size of 100) to a few days (for Model 3 with a pilot sample size of 1500).

The Julia and R packages required to run the files are:
Julia: Distributions, CSV, DataFrames, StatsBase, Random, RCall, DelimitedFiles
R: sandwich

The paper uses replication data from Karlan and Wood (2017) which is available in its original form from Innovations for Poverty Action under a CC0 1.0 Public Domain License (https://dataverse.harvard.edu/dataset.xhtml?persistentId=doi%3A10.7910/DVN/RDHJQZ). To obtain the dataset, click “Access Dataset” and fill out the registration form, including a brief description of its intended use.

The folder contains the following files:

- NodeGenetic.jl: Contains the code to run the evolutionary algorithm
- NodeSims.jl: Contains other general functions
- NodeRunSims.jl: Contains a function designed to run the main simulation specifications on a personal computer
- Node RunAppSims.jl: Contains a function designed to run the application-based simulation specification on a personal computer
- NodeApp.jl: Contains functions to reproduce the empirical application results (Figures 6 and 7 in the paper)
- data_full.csv: Contains the data to replicate Figure 6
- data_left.csv: Contains the data to replicate the left subtree in Figure 7
- data_right.csv: Contains the data to replicate the right subtree in Figure 7
- model_app.csv: Contains the data used in the application-based simulation
- Rtest.R: Contains R functions used to run OLS and compute robust standard errors
- Simulation Results: This folder contains copies of the simulation output used to create Tables 1, 2, and 3 in the paper.
- Batch Job Version: This folder contains the necessary files to run the simulations in parallel on a computing cluster (see below)
    - NodeRunSimsBatch.jl: Contains a function designed to run the main simulation specifications on a computing cluster
    - NodeRunAppSimsBatch.jl: Contains a function designed to run the application-

based simulation specification on a computing cluster

- load_sims.jl: This file runs the simulation for a given sample size, DGP, depth and number of iterations per job
- load_app_sims.jl: This file run the application-based simulation specification for a given sample size, depth and number of iterations per job
- runbatch_acropolis.sh: This is a PBS job script which loads either load_sims.jl or load_app_sims.jl
- compile_results.py: This is a python 3 script which compiles all of the csv files produced by NodeRunSimsBatch.jl or NodeRunAppSimsBatch.jl to produce the final simulation results for a given sample size, DGP and depth. The dependencies required to run this file are numpy and csv.

NOTE: Although the programs in this folder can in principle run the simulation designs presented in the paper, this is not a practical solution. In reality, the simulations were run in parallel on a computing cluster. As a result, attempting to recreate the simulations from the paper using NodeRunSims.jl or NodeRunAppSims.jl will 1. be unreasonably slow and 2. not perfectly reproduce the results. Versions of NodeRunSims.jl and NodeRunAppSims.jl designed to be executed in parallel are available in the "Batch Job Version" folder.

Instructions to run batch job:

1. Write a job loading script designed to run load_sims.jl with 300 jobs indexed from 0-299 (see runbatch_acropolis.sh for an example)
2. Edit load_sims.jl to run the desired design (pilot sample size, main sample size, DGP, depth)
3. Run the job script, this will produce 4 CSV files labeled results_val.csv, results_cov.csv, results_len.csv and results_pow.csv
4. Run the python script compile_results.py in the same folder as the CSV files to consolidate these files into one CSV file which contains the final output for that simulation design

References:

Karlan, Dean and Daniel H Wood (2017), "The effect of effectiveness: Donor response to aid effectiveness in a direct mail fundraising experiment." Journal of Behavioral and Experimental Economics, 66, 1–8.

Karlan, Dean; Wood, Daniel H., 2017, "The Effect of Effectiveness: Donor Response to Aid Effectiveness in a Direct Mail Fundraising Experiment", https://doi.org/10.7910/DVN/RDHJQZ, Harvard Dataverse, V1, UNF:6:/4inN17v5fO0z1k4w8dspw== [fileUNF]