3535 Thursday 12:45

Harnessing cloud computing for high capacity analysis of neuroimaging data from NDAR

Daniel Clark[1], Christian Haselgrove[2], David Kennedy[2], Zhizhong Liu[3],
Michael Milham[1], Petros Petrosyan[4], Carinna Torgerson[3], John Van Horn[3], Cameron Craddock[1,5]

[1]Child Mind Institute, New York, NY, [2] University of Massachuttes Medical School, Worcester, MA, [3]University of Southern California, Los Angeles, CA, [4]UCLA, Los Angeles, CA, [5]Nathan S. Kline Institute for Psychiatric Research, Orangeburg, NY

Introduction

- The National Database for Autism Research (NDAR)[1] hosts a vast collection of neuroimaging datasets that can be processed and utilized to yield significant scientific discoveries.
- This amount of resources necessitates a high-performance computing (HPC) infrastructure, which is not always readily available for researchers in-house.
- Amazon Web Services (AWS) Elastic Compute Cloud (EC2) computing service offers a "pay as you go" model that allows researchers to utilize HPC performance without the up-front captial costs and maintenance of an in-house solution.
- The developers of the Laboratory of Neuro Imaging (LONI) Pipeline, the Neuroimaging Informatics Tools and Resources Clearinghouse (NITRC) Computational Environment (CE) and the Configurable Pipeline for the Analysis of Connectomes (C-PAC) have implemented pipelines in EC2 that interface with NDAR

Methods

LONI Pipeline

- The LONI Pipeline software was extended to include new pipeline modules to access data from the NDAR database, transfer input data out of Amazon S3 (Simple Storage Service), and to load results back into S3[2]
- A pipeline was constructed to extract cortical thickness and subcortical region volume data from structural MRI images in the NDAR database, which included:
  1. Reorient images to standard orientation using FSL's reorient2std module
  2. Extract cortical thickness using FreeSurfer recon-all
  3. Calculate volumes of subcortical regions using FSL's BET and FIRST all
- The resulting pipeline was used to process 780 T1-weighted structural images and return the results to NDAR

Configurable Pipeline For the Analysis of Connectomes (C-PAC)[3]

- C-PAC modules were written in Python to build input data lists by querying NDAR, read input data from S3, write processed results to S3 and write values back to the NDAR database
- New pipelines were created to perform the ANTS cortical thickness[4] procedure and the Preprocessed Connectomes Project's Quality Assessment Protocol (`http://preprocessed-connectomes-project.github.io/quality-assessment-pipeline`)[5]
- The resulting pipelines and modules were used to process several datasets and return the results to NDAR
  1. Cortical extraction from 3,197 T1-weighted structural images
  2. Structural and functional processing for 1,112 datasets from ABIDE
  3. Automated quality asssessment of 1,112 datasets from ABIDE

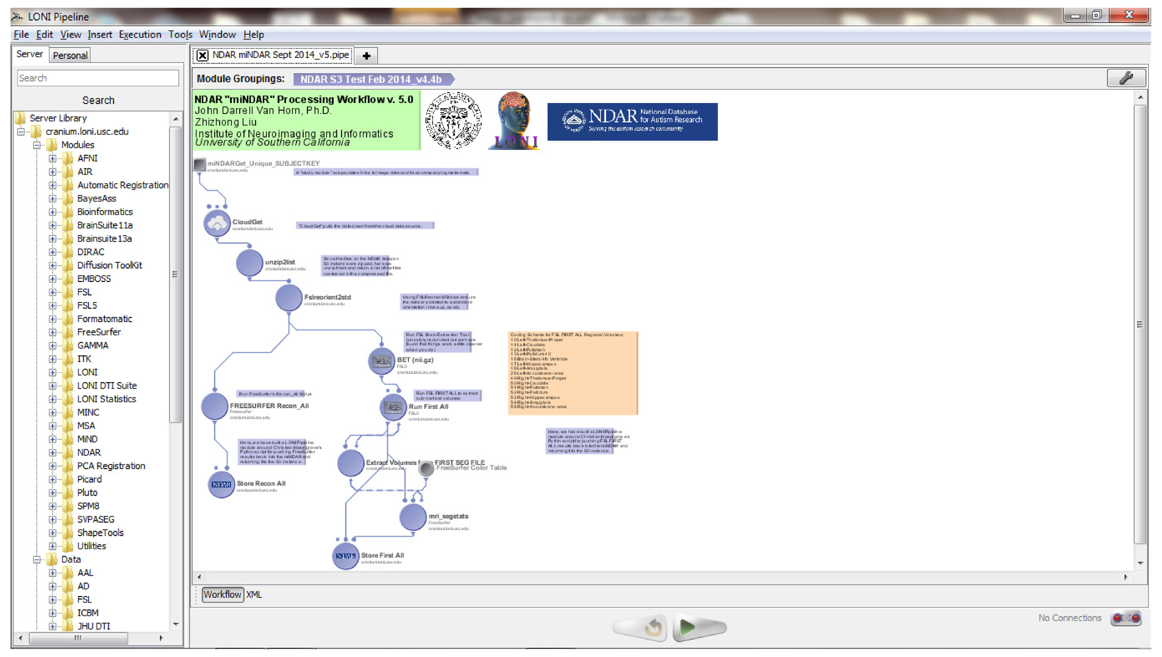

Figure 1 : Graphical layout of the constructed pipeline

Neuroimaging Informatics Tools and Resources Clearinghouse Computational Environment (NITRC-CE)[6]

- The NITRC pipeline processed data using three primary utilities
  1. Extract anatomical and surface-base measures with Freesurfer recon-all
  2. Segment subcortical structures using FSL's FIRST to produce volumetric and mesh outputs
  3. Time series QA measures using the fmriqa generate.pl utility from the BXH/XCEDE Tools suite, including mean intensty, center of mass, per-slice variation, and others
- Python modules were created to query and download data from NDAR as well as to store results back to their database
- The recon-all and FIRST tools processed 986 and 1,247 T1-weighted anatomical scans, respectively; the fmriqa generate QA measures were generated from 1,349 functional scans.

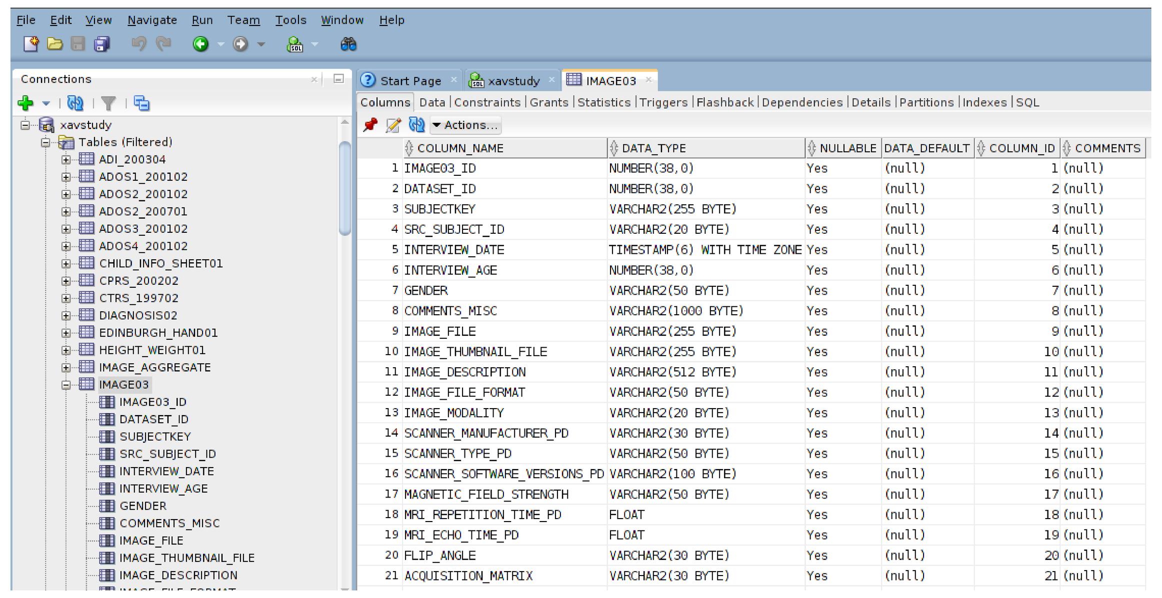

Figure 2 : miNDAR database

Interacting with the NDAR database

- Launched an AWS-hosted miNDAR database by querying NDAR website for the data of interest (e.g. from a particular study)
- Built a subject list by querying the database for subjects of interest to pass to our pipeline
- Launch an AWS EC2 HPC cluster using Starcluster (`http://star.mit.edu/cluster/`)
- Log into the cluster and submit a Sun Grid Engine job using our pipeline software and the subject list
- The pipeline software will process the data, store raw outputs in an AWS S3 bucket and insert S3 filepaths and output measures into miNDAR database

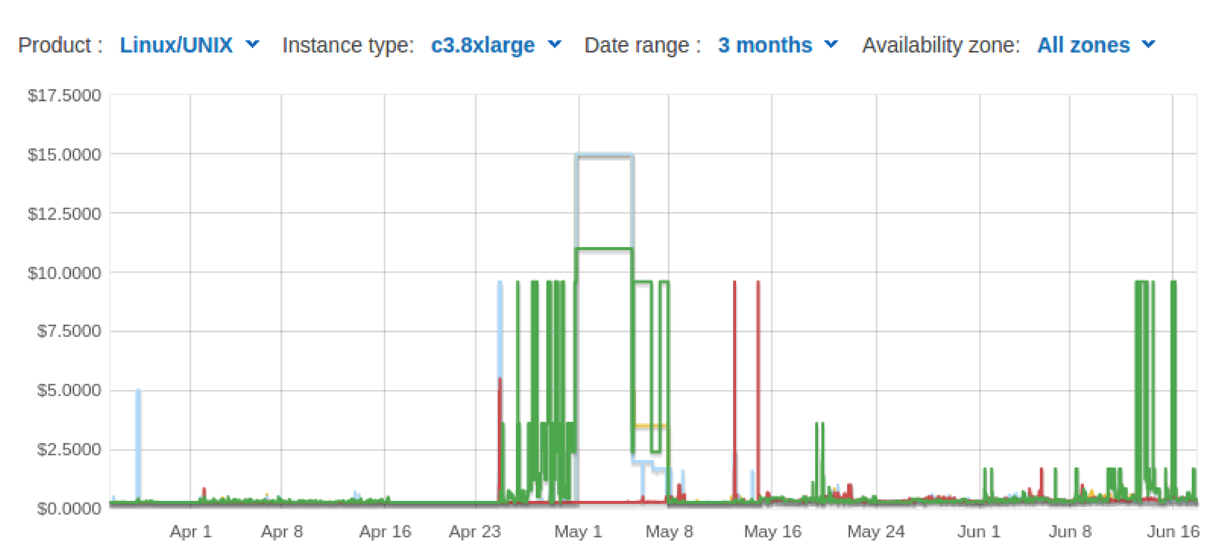

Figure 3 : Spot price history for the c3.8xlarge instance in the us-east-1 region on AWS across the four different availability zones for the past three months

EC2 Spot Pricing

- In addition to the "pay as you go" compute pricing model, AWS offers users to bid on instances that are not being used
- As opposed to paying an "on-demand" fixed price, the hourly charge for the instances will vary over time based on what people in different regions are willing to pay
- If the spot market price goes above what the user bids, their instance will be terminated by Amazon and all of their processes will be stopped and their data lost
- Typically, one can expect to pay one-eighth of the hourly price of a standard on-demand instance when using spot-pricing
- AWS offers cloud solutions based on geographic region, where the price for services vary
- Some regions fluxuate on spot price more than others, but there are ways to minimize process interruptions and lost data in how one configures their cluster and pipelines

Results

Results cont.

Costs and run time models

- Models assume that users upload their input data to the cloud, run their pipelines and (optionally) downloads their outputs as they become available.
- Outputs are stored on an AWS Elastic Block Store (EBS) hard drive mounted to the master node and is NFS-shared across all of the slave nodes.
- Users can also upload their processed outputs to a cloud storage solution, like AWS Simple Storage Service (S3) directly from the cluster - this is the approach that was taken in processing data for NDAR.
- Storing results in AWS S3 avoids costly download time and provides for a viable solution for backing up data; however this does incur additional storage costs.

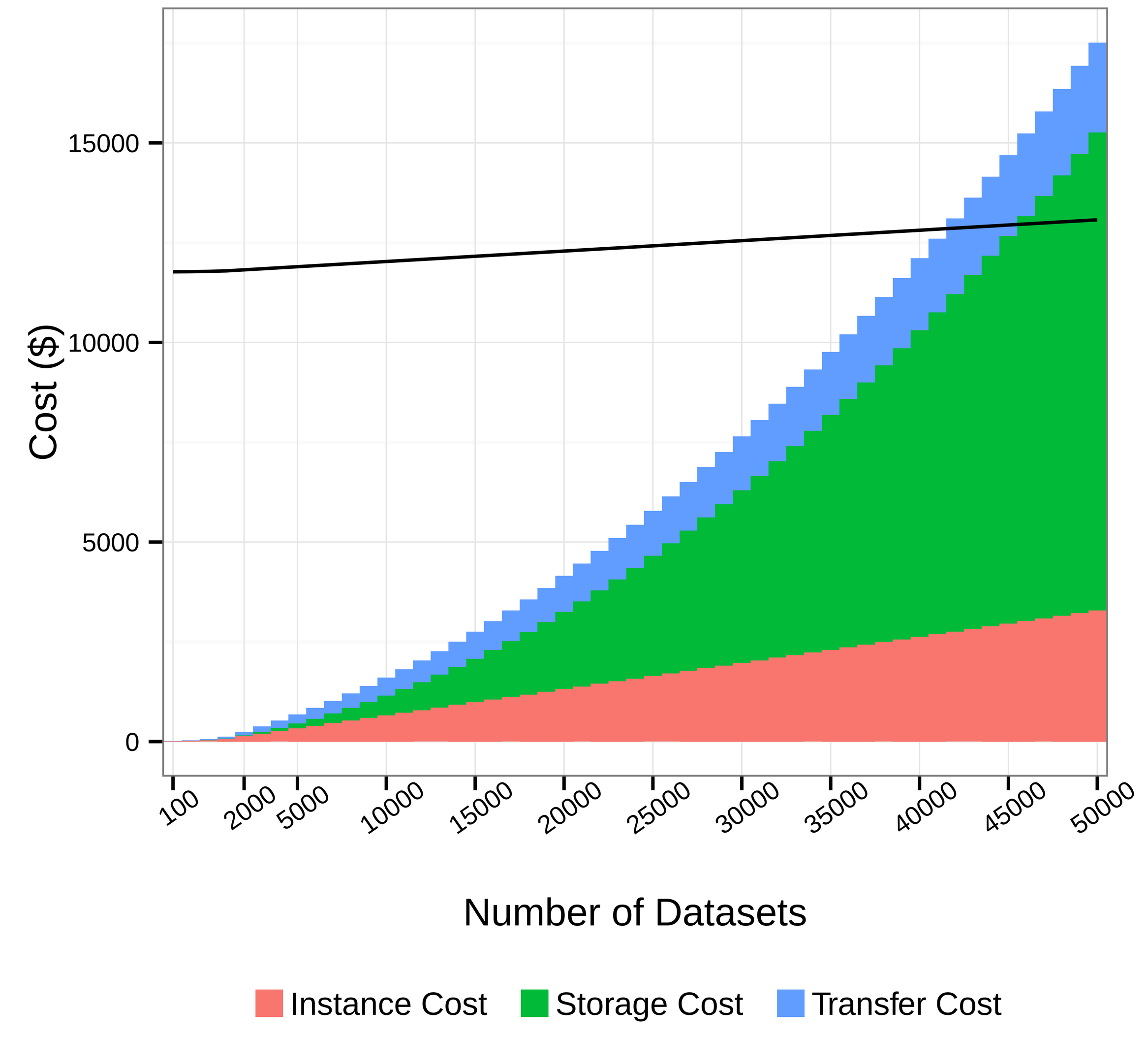

Figure 4 : AWS EC2 costs, grouped by cost type, for a typical C-PAC pipeline for different sized datasets versus owning and maintaining own server; costs are based on using the cluster configuration shown in Table 1 for the cloud, and a single compute node locally

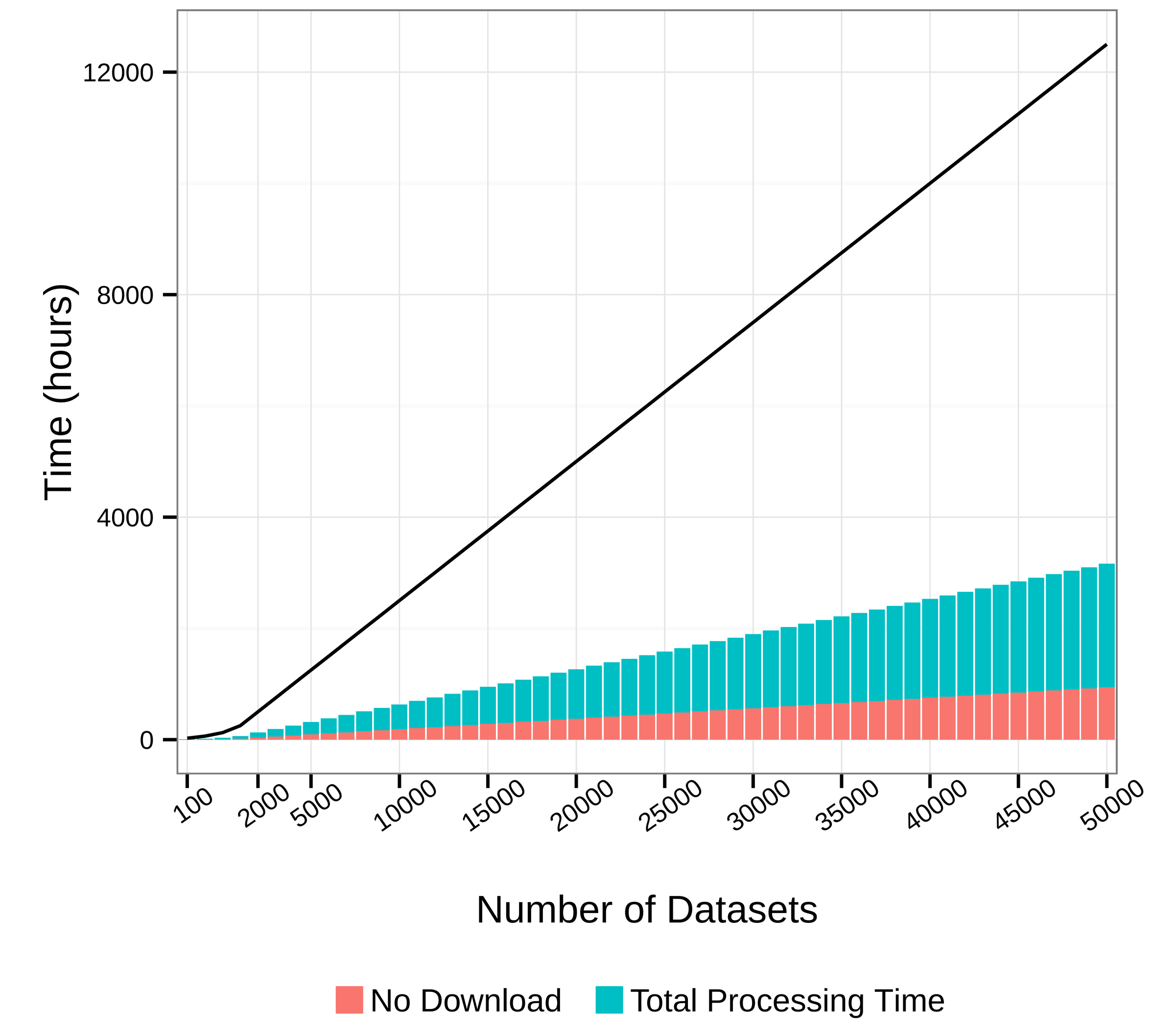

Figure 6 : Run times using the C-PAC pipeline, grouped by downloading vs non-downoading output data, for different sized datasets versus running locally; times are based on using the cluster configuration shown in Table 1 for the cloud, and a single compute node locally

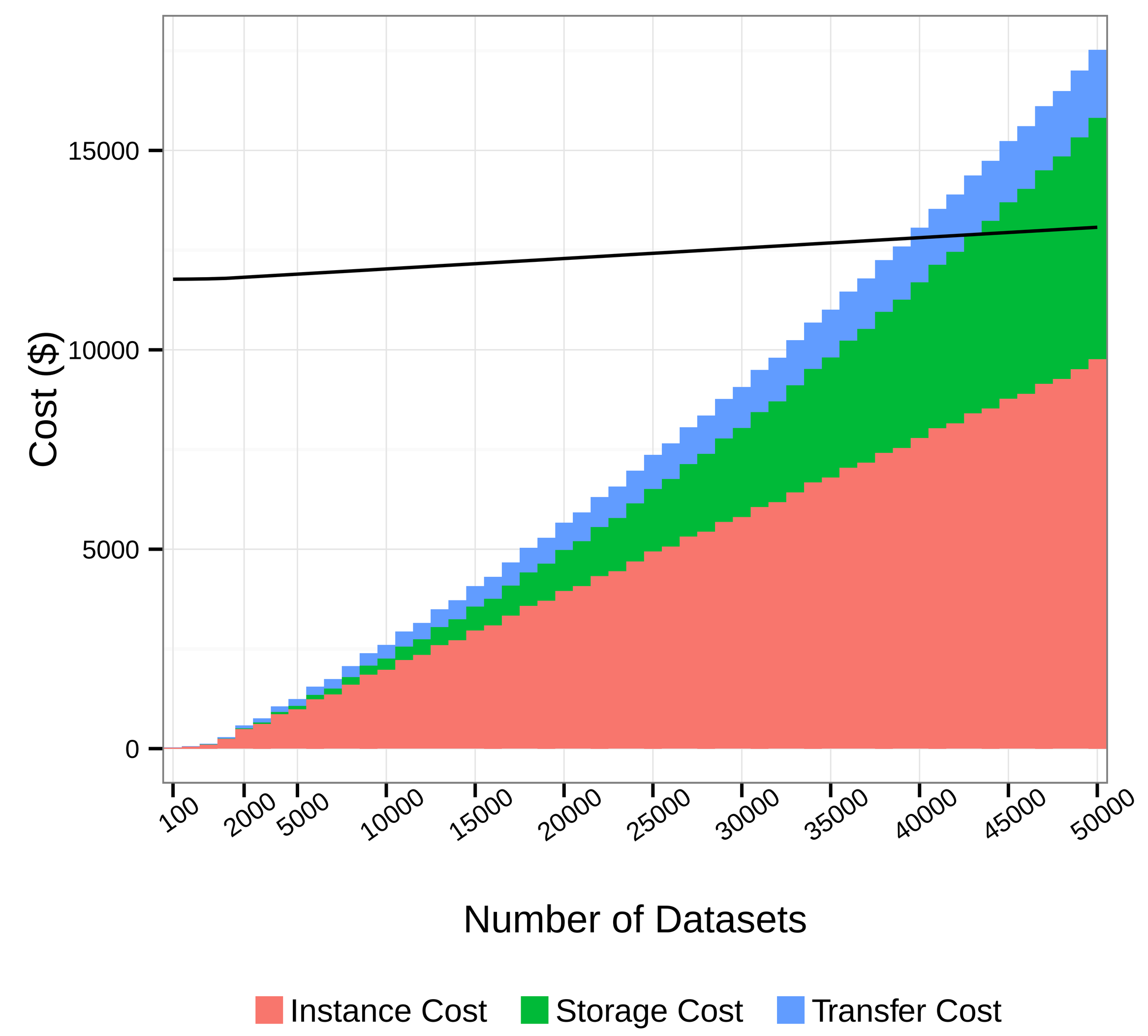

Figure 5 : AWS EC2 costs, grouped by cost type, for a typical Freesurfer pipeline for different sized datasets versus owning and maintaining own server; costs are based on using the cluster configuration shown in Table 1 for the cloud, and a single compute node locally

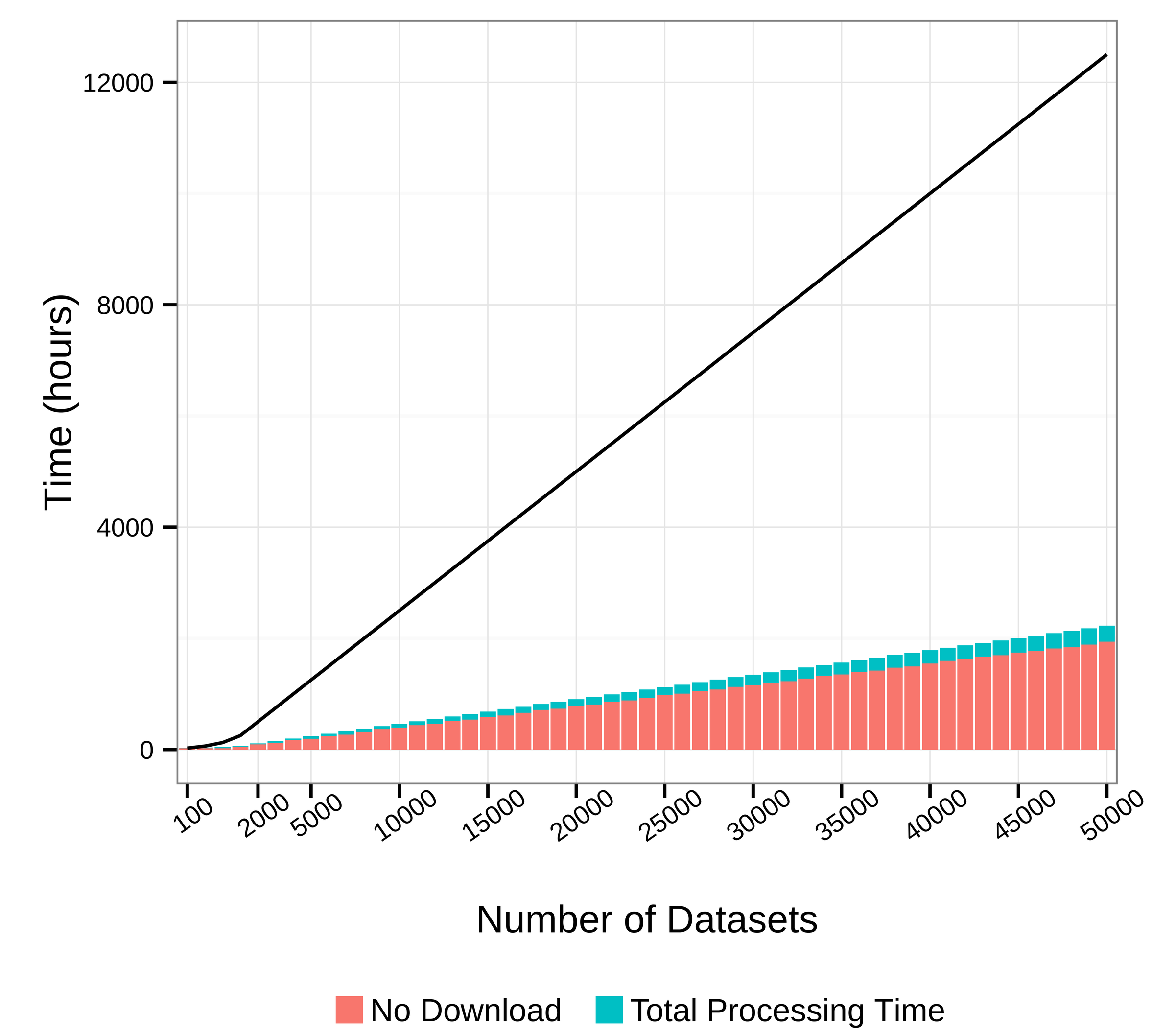

Figure 7 : Run times using the Freesurfer pipeline, grouped by downloading vs non-downoading output data, for different sized datasets versus running locally

cameron.craddock@childmind.org