Databases and Ontologies

SoS Notebook: An Interactive Multi-Language Data Analysis Environment

Bo Peng1,*, Gao Wang2, Jun Ma1, Man Chong Leong3, Chris Wakefield1, James Melott1, Yulun Chiu1, Di Du1, and John N. Weinstein1

1Department of Bioinformatics and Computational Biology, The University of Texas, MD Anderson Cancer Center, 1400 Pressler St, Unit 1410, Houston, TX, 77030, 2Department of Human Genetics, The University of Chicago, 920 E 58th St, Suite 410, Chicago, IL, 60637, 3Department of Statistics, Rice University, 6100 Main St, Houston, TX 77005

*To whom correspondence should be addressed.

Associate Editor: XXXXXXX
Received on XXXXX; revised on XXXXX; accepted on XXXXX

Abstract
Motivation: Complex bioinformatic data analysis workflows involving multiple scripts in different languages can be difficult to consolidate, share, and reproduce. An environment that streamlines the entire processes of data collection, analysis, visualization and reporting of such multi-language analyses is currently lacking.

Results: We developed Script of Scripts (SoS) Notebook, a web-based notebook environment that allows the use of multiple scripting language in a single notebook, with data flowing freely within and across languages. SoS Notebook enables researchers to perform sophisticated bioinformatic analysis using the most suitable tools for different parts of the workflow, without the limitations of a particular language or complications of cross-language communications.

Availability: SoS Notebook is hosted at http://vatlab.github.io/SoS/ and is distributed under a BSD license.
Contact: bpeng@mdanderson.org

1 Introduction

Due to complications and limitations of cross-language interfaces (e.g. the R and MATLAB interfaces for Python), most bioinformaticians write separate scripts when tools in different languages are needed for parts of the analysis. Adding to all of the difficulties in interfacing between languages and producing reports from multiple scripts, it is challenging to keep track of separate script files, share fragmented workflows with colleagues, and archive the workflows properly for future reference.

A multi-language data analysis environment provides a unified platform on which workflows composed in mixed languages can be easily executed, shared, and reproduced. A few such environments under active development include RStudio Notebook, Apache Zeppelin, and Beaker Notebook (now BeakerX). RStudio has a clear root in R with limited support for other languages; Zeppelin is designed for data analytics and visualization in large-scale data exploration; Beaker Notebook has now been replaced by BeakerX which supports only Java-based languages. None of those environments support SAS or MATLAB, and that limits their applications in the bioinformatics and biostatistics community.

With the goal of developing a versatile environment for daily bioinformatic data analysis, we developed a multi-language notebook environment powered by a Python3-based workflow engine entitled Script of Scripts (SoS), hence its name SoS Notebook.

2 Features

SoS Notebook consists of a new kernel and a number of front-end extensions of the web-based Jupyter Notebook platform. The SoS kernel acts as a proxy to the SoS workflow engine and a hub between Jupyter and more than 60 existing Jupyter kernels. It executes scripts in multiple languages and coordinates data exchange among the kernels. The SoS front-end extends the single-kernel notebook interface of Jupyter to a multi-language notebook interface with global and cell-level language selectors and a multi-purpose side panel. Using a set of “magics” (special commands prefixed with %) and keyboard shortcuts, SoS Notebook provides an interactive notebook environment with explicit and automatic kernel switch and data exchange, line-by-line code execution, and other useful features such as preview of variables and files.
A more powerful data exchange method is provided to exchange variables among kernels for supported languages (Bash, JavaScript, Julia, MATLAB, Octave, Python2 and 3, R, and SAS). Because of large differences in datatypes among scripting languages, SoS transfers variables through the creation of independent homonymous variables of the most similar datatypes in the destination language. For example, although 3 and c(3, 5) are both numeric arrays in R, they are transferred to Python as integer 3 and numpy array([3, 5]) respectively. Similarly, Julia’s Char and str types are both converted to and from Python as str, while Python DataFrame is converted to data.frame in R, dataset in SAS, table in MATLAB, data.frame in Octave, and nested dictionaries in JavaScript. Data exchange is performed automatically for variables with names starting with sos and explicitly with magics %with and %use. Magic %expand interpolates cell contents with variables in the SoS kernel, allowing the composition of scripts in different languages using shared notebook variables (e.g. filenames, parameters). Magic %capture captures cell outputs as SoS variables so results from one kernel can be passed to and used by other kernels. Along with the main notebook, a side panel is provided to perform a variety of nonpermanent actions, such as executing scratch commands, showing results of line-by-line execution, previewing variables and files, and showing the table of contents of the notebook.

An SoS notebook consists of markdown cells and code cells with content executed by either SoS (based on Python 3) or one of the sub-kernels. Cell kernels can be changed either interactively or using magics %with and %use. Magic %expand interpolates cell contents with variables in the SoS kernel, allowing the composition of scripts in different languages using shared notebook variables (e.g. filenames, parameters). Magic %capture captures cell outputs as SoS variables so results from one kernel can be passed to and used by other kernels. Along with the main notebook, a side panel is provided to perform a variety of nonpermanent actions, such as executing scratch commands, showing results of line-by-line execution, previewing variables and files, and showing the table of contents of the notebook.

A more powerful data exchange method is provided to exchange variables among kernels for supported languages (Bash, JavaScript, Julia, MATLAB, Octave, Python2 and 3, R, and SAS). Because of large differences in datatypes among scripting languages, SoS transfers variables through the creation of independent homonymous variables of the most similar datatypes in the destination language. For example, although 3 and c(3, 5) are both numeric arrays in R, they are transferred to Python as integer 3 and numpy array([3, 5]) respectively. Similarly, Julia’s Char and str types are both converted to and from Python as str, while Python DataFrame is converted to data.frame in R, dataset in SAS, table in MATLAB, data.frame in Octave, and nested dictionaries in JavaScript. Data exchange is performed automatically for variables with names starting with sos and explicitly with magics %with and %use. Magic %expand interpolates cell contents with variables in the SoS kernel, allowing the composition of scripts in different languages using shared notebook variables (e.g. filenames, parameters). Magic %capture captures cell outputs as SoS variables so results from one kernel can be passed to and used by other kernels. Along with the main notebook, a side panel is provided to perform a variety of nonpermanent actions, such as executing scratch commands, showing results of line-by-line execution, previewing variables and files, and showing the table of contents of the notebook.

A more powerful data exchange method is provided to exchange variables among kernels for supported languages (Bash, JavaScript, Julia, MATLAB, Octave, Python2 and 3, R, and SAS). Because of large differences in datatypes among scripting languages, SoS transfers variables through the creation of independent homonymous variables of the most similar datatypes in the destination language. For example, although 3 and c(3, 5) are both numeric arrays in R, they are transferred to Python as integer 3 and numpy array([3, 5]) respectively. Similarly, Julia’s Char and str types are both converted to and from Python as str, while Python DataFrame is converted to data.frame in R, dataset in SAS, table in MATLAB, data.frame in Octave, and nested dictionaries in JavaScript. Data exchange is performed automatically for variables with names starting with sos and explicitly with magics %with and %use. Magic %expand interpolates cell contents with variables in the SoS kernel, allowing the composition of scripts in different languages using shared notebook variables (e.g. filenames, parameters). Magic %capture captures cell outputs as SoS variables so results from one kernel can be passed to and used by other kernels. Along with the main notebook, a side panel is provided to perform a variety of nonpermanent actions, such as executing scratch commands, showing results of line-by-line execution, previewing variables and files, and showing the table of contents of the notebook.

A more powerful data exchange method is provided to exchange variables among kernels for supported languages (Bash, JavaScript, Julia, MATLAB, Octave, Python2 and 3, R, and SAS). Because of large differences in datatypes among scripting languages, SoS transfers variables through the creation of independent homonymous variables of the most similar datatypes in the destination language. For example, although 3 and c(3, 5) are both numeric arrays in R, they are transferred to Python as integer 3 and numpy array([3, 5]) respectively. Similarly, Julia’s Char and str types are both converted to and from Python as str, while Python DataFrame is converted to data.frame in R, dataset in SAS, table in MATLAB, data.frame in Octave, and nested dictionaries in JavaScript. Data exchange is performed automatically for variables with names starting with sos and explicitly with magics %with and %use. Magic %expand interpolates cell contents with variables in the SoS kernel, allowing the composition of scripts in different languages using shared notebook variables (e.g. filenames, parameters). Magic %capture captures cell outputs as SoS variables so results from one kernel can be passed to and used by other kernels. Along with the main notebook, a side panel is provided to perform a variety of nonpermanent actions, such as executing scratch commands, showing results of line-by-line execution, previewing variables and files, and showing the table of contents of the notebook.

A more powerful data exchange method is provided to exchange variables among kernels for supported languages (Bash, JavaScript, Julia, MATLAB, Octave, Python2 and 3, R, and SAS). Because of large differences in datatypes among scripting languages, SoS transfers variables through the creation of independent homonymous variables of the most similar datatypes in the destination language. For example, although 3 and c(3, 5) are both numeric arrays in R, they are transferred to Python as integer 3 and numpy array([3, 5]) respectively. Similarly, Julia’s Char and str types are both converted to and from Python as str, while Python DataFrame is converted to data.frame in R, dataset in SAS, table in MATLAB, data.frame in Octave, and nested dictionaries in JavaScript. Data exchange is performed automatically for variables with names starting with sos and explicitly with magics %with and %use. Magic %expand interpolates cell contents with variables in the SoS kernel, allowing the composition of scripts in different languages using shared notebook variables (e.g. filenames, parameters). Magic %capture captures cell outputs as SoS variables so results from one kernel can be passed to and used by other kernels. Along with the main notebook, a side panel is provided to perform a variety of nonpermanent actions, such as executing scratch commands, showing results of line-by-line execution, previewing variables and files, and showing the table of contents of the notebook.

A more powerful data exchange method is provided to exchange variables among kernels for supported languages (Bash, JavaScript, Julia, MATLAB, Octave, Python2 and 3, R, and SAS). Because of large differences in datatypes among scripting languages, SoS transfers variables through the creation of independent homonymous variables of the most similar datatypes in the destination language. For example, although 3 and c(3, 5) are both numeric arrays in R, they are transferred to Python as integer 3 and numpy array([3, 5]) respectively. Similarly, Julia’s Char and str types are both converted to and from Python as str, while Python DataFrame is converted to data.frame in R, dataset in SAS, table in MATLAB, data.frame in Octave, and nested dictionaries in JavaScript. Data exchange is performed automatically for variables with names starting with sos and explicitly with magics %with and %use. Magic %expand interpolates cell contents with variables in the SoS kernel, allowing the composition of scripts in different languages using shared notebook variables (e.g. filenames, parameters). Magic %capture captures cell outputs as SoS variables so results from one kernel can be passed to and used by other kernels. Along with the main notebook, a side panel is provided to perform a variety of nonpermanent actions, such as executing scratch commands, showing results of line-by-line execution, previewing variables and files, and showing the table of contents of the notebook.