

# Logistics of Collaborative Development

What we think we've learned from other projects

Christopher D. Carroll   Alexander M. Kaufman   David C. Low  
Nathan M. Palmer   Matthew N. White

June 2016

- 1 Introduction
- 2 Incentives
- 3 Tools from Software Development
- 4 License Tools
- 5 Conclusion

# Introduction

Pessimistic rephrasing of contribution to open access research code:

- “Come undertake high fixed costs so other strangers (who might be publication competitors) don’t need to!”

Pessimistic rephrasing of contribution to open access research code:

- “Come undertake high fixed costs so other strangers (who might be publication competitors) don't need to!”
- i.e. “Why would anyone ever do this?”

We already know the answer:

- 1 In many other domains ways have been found to make incentives align

We already know the answer:

- ① In many other domains ways have been found to make incentives align
- ② Technology exists such that costs are low enough to be viable
  - Technology in the **economic sense** as well as colloquial sense

## Topics:

- What might appropriate incentives look like?
  - Look at a few successful examples
  - Offer a few tentative ideas



## Topics:

- What might appropriate incentives look like?
  - Look at a few successful examples
  - Offer a few tentative ideas
- What is the enabling technology?
  - Quick overview of technology which supports tentative ideas

## Topics:

- What might appropriate incentives look like?
  - Look at a few successful examples
  - Offer a few tentative ideas
- What is the enabling technology?
  - Quick overview of technology which supports tentative ideas
- Tentative, exploratory, “what we’ve learned”
  - jump in any time!

# Incentives

# Successful Examples

Three Examples:

- Journal of Statistical Software
- AstroPy
- Quant-Econ

For Each:

- What they do
- What they do well
- More info

[Home](#) > [Vol 68 \(2015\)](#)

Established in 1996, the Journal of Statistical Software publishes articles, book reviews, code snippets, and software reviews on the subject of statistical software and algorithms. The contents are freely available on-line. For both articles and code snippets the source code is published along with the paper. Statistical software is the key link between statistical methods and their application in practice. Software that makes this link is the province of the journal, and may be realized as, for instance, tools for large scale computing, database technology, desktop computing, distributed systems, the World Wide Web, reproducible research, archiving and documentation, and embedded systems. We attempt to present research that demonstrates the joint evolution of computational and statistical methods and techniques. Implementations can use languages such as C, C++, S, Fortran, Java, PHP, Python and Ruby or environments such as Mathematica, MATLAB, R, S-PLUS, SAS, Stata, and XLISP-STAT.



### Recent Publications

#### Articles

[CovSel: An R Package for Covariate Selection When Estimating Average Causal Effects](#) [PDF](#)

Jenny Häggström, Emma Persson, Ingeborg Waernbaum, Xavier de Luna

[sms: An R Package for the Construction of Microdata for Geographical Analysis](#) [PDF](#)

Dimitris Kavrouidakis

[Parallel Sequential Monte Carlo for Efficient Density Combination: The DeCo MATLAB Toolbox](#) [PDF](#)

Roberto Casarin, Stefano Grassi, Francesco Ravazzolo, Herman K. van Dijk

[Bayesian Model Averaging Employing Fixed and Flexible Priors: The BMS Package for R](#) [PDF](#)

Stefan Zeugner, Martin Feldkircher

#### OPEN JOURNAL SYSTEMS

##### [Journal Help](#)

#### USER

Username

Password

☐ Remember me

[Login](#)

#### NOTIFICATIONS

- [View](#)
- [Subscribe](#)

#### JOURNAL CONTENT

Search

Search Scope

All

[Search](#)

Figure : image

- What they do:
  - An open-access journal, publishes **papers outlining software**
  - Founded in 1996 with 1 editor in chief
  - Currently 3 editors-in-chief, 58 associate editors, 58 volumes (4.9 IF)
- What they do well:
  - **Academic incentives** for creating useful / used, tested, quality code
    - Currency of academia used appropriately: peer review and citation
  - Published papers and “vignettes” on statistical software posted in CRAN
  - Papers devoted to “writing up” well-structured software

24 APRIL 2014

## Publishing an R package in the Journal of Statistical Software

COMPUTING,  
JOURNALS, JSS, R,  
REFEREEING

12 COMMENTS

I've been an editor of JSS for the last few years, and as a result I tend to get email from people asking me about publishing papers describing R packages in JSS. So for all those wondering, here are some general comments.

JSS prefers to publish papers about packages where the package is on [CRAN](#) and has been there long enough to have matured (i.e., obvious bugs ironed out and a few active users). This is partly because we have so many submissions that it helps to filter some out and this approach provides some basic quality checks. So I suggest you begin by developing the package for CRAN. This is a preference rather than a requirement, and it is not stated anywhere in the JSS rules. A paper describing a package that has only recently been put on CRAN will still be considered, but the probability of it getting through the reviewing process is smaller.

Figure : image

# Would You Like to Know More?

- Main webpage: <http://www.jstatsoft.org/>
- Open Journal Systems: <http://pkp.sfu.ca/ojs/>
- **Excellent** overviews by founder:
  - Presentation: “JSS: Past, Present Future”  
[http://gifi.stat.ucla.edu/janspubs/2014/notes/deleeuw\\_mullen\\_U\\_14.pdf](http://gifi.stat.ucla.edu/janspubs/2014/notes/deleeuw_mullen_U_14.pdf)
  - Interview: “JSS and its Success”  
<http://archive.sciencewatch.com/inter/jou/2011/11decJofStatSoft/>
- Excellent overview of mature academic software review process:  
<http://robjhyndman.com/hyndsight/jss-rpackages/>





A Community Python Library for Astronomy

The AstroPy Project is a community effort to develop a single core package for Astronomy in Python and foster interoperability between Python astronomy packages.

[Current Documentation](#)

[Other Docs](#) ▼

Current Version: 1.0.6

Please remember to [acknowledge](#) the use of AstroPy!

## Install AstroPy



OS X



Linux



Windows

[</> Source](#)

[Developer](#)

There are a number of options for installing the astropy package on Linux. AstroPy can be installed on some Linux distributions using the built-in package manager (apt-get, yum, etc.), and is also included by default in the [Anaconda Python](#)

- What they do:
  - Community-driven effort to provide single core “Astronomy” package
- What they do well:
  - Use modern tools: git, Python
  - Many details of organizational development
    - core tools, affiliated packages
    - community of users / developers
    - IP issues
    - Cool logo

# Would You Like to Know More?

- Main webpage: <http://www.astropy.org/>
  - About: <http://www.astropy.org/about.html>
  - Detailed About: <http://docs.astropy.org/en/stable/overview.html>
  - Original Vision:  
<http://docs.astropy.org/en/stable/development/vision.html>
- Introduction paper: “Astropy: A community Python package for astronomy”  
<http://www.aanda.org/articles/aa/abs/2013/10/aa22068-13/aa22068-13.html>



## Quantitative Economics

Thomas J. Sargent

John Stachurski

[Home](#)

[P](#)

This website presents a series of lectures on quantitative economic modelling, designed and written by [Thomas J. Sargent](#) and [John Stachurski](#). The primary programming languages are Python and Julia. You can send feedback to the authors via our web forum [quantecon](#) or [webmaster@quant-econ.net](mailto:webmaster@quant-econ.net).



[About this site](#)



[Choose Python](#)



[Choose Julia](#)

- What they do:
  - Community-driven effort to provide single core “Astronomy” package
- What they do well:
  - Use modern tools: git, Python
  - Many details of organizational development
    - core tools, affiliated packages
    - community of users / developers
    - IP issues
    - Cool logo

# Would You Like to Know More?

- Main webpage:
  - Lectures: <http://quant-econ.net/>
  - Organization: <http://quantecon.org/>
  - Github: <https://github.com/QuantEcon/>
- Developers

# Tentative Ideas

- If you build it, they will come only works in the movies.
- How to encourage collaborative development?

# Tentative Ideas

- Scholarships, fellowships for participation.
- Journal of Statistical Software approach, with modern tools
  - Essential to have “deliverables” which can be cited, e.g. for performance reviews
  - Aim to capture much of the “lost effort” associated with computational research, while at the same time improving code production
- Finally, incentives created by influencing norms
  - the force of personality approach
  - students given assignments
  - journals encourage submissions to fulfill reproducibility
  - graduate research hunger



# Tools from Software Development

# Technology Improvement: Efficient Code Output

- Automated Documentation
  - Inline
  - Online (Sphinx)
- Automated Testing
  - Unit testing
  - Peer review
- Version Control
  - Automatically archive code
  - Collaborative workflows (tracking issues, decentralized review)
  - “Freeze” scientific publications for reference
- Scientific Notebooks
  - Combine code, math, descriptions, interaction (“vignettes”)

Types of documentation:

- system-style
- on-line (API)

# Automated Documentation

```
def utility(c, gamma):  
    """  
    Return constant relative risk aversion (CRRA) utility of consumption "c"  
    given risk aversion parameter "gamma."  
  
    Parameters  
    -----  
    c: float  
        Consumption value.  
    gamma: float  
        Risk aversion, gamma != 1.  
  
    Returns  
    -----  
    u: float  
        Utility.  
  
    Notes  
    -----  
    gamma cannot equal 1. This constitutes natural log utility; np.log  
    should be used instead.  
    """  
    u = c**((1.0 - gamma) / (1.0 - gamma)) # Find the utility value of c given gamma  
    return u                               # Return the utility value
```

# Automated Documentation

HARK 0.97 documentation »next | modules | index

## Table Of Contents

Welcome to HARK's documentation!  
Indices and tables

### Next topic

<no title>

### This Page

Show Source

### Quick search

Go

## Welcome to HARK's documentation!

### Contents:

<a href="#">HARKutilities</a>	General purpose / miscellaneous functions.
<a href="#">HARKsimulation</a>	Functions for generating simulated data and shocks.
<a href="#">HARKparallel</a>	Early version of multithreading in HARK.
<a href="#">HARKinterpolation</a>	Custom interpolation methods for representing approximations to functions.
<a href="#">HARKestimation</a>	Functions for estimating structural models, including optimization methods and bootstrapping tools.
<a href="#">HARKcore</a>	High-level functions and classes for solving a wide variety of economic models.
<a href="#">ConsIndShockModel</a>	Classes to solve canonical consumption-savings models with idiosyncratic shocks to income.

Figure : Sphinx Docs

Unit tests: small tests at simple functional levels

- doctests
- unittest
- Automatically run tests, examine test coverage
- Broader acceptance protocols: determining when a piece of code is acceptable

# Automatic Testing: Quant-Econ Example

## Quantitative Economics (Python)

A code library for quantitative economic modeling in Python

Library Website: [http://quantecon.org/python\\_index.html](http://quantecon.org/python_index.html)

### Installation

See the [library website](#) for instructions

#### Build and Coverage Status:

build passing coverage 87%

#### ReadTheDocs Status:

docs latest

Figure : Quant Econ Build Status

What is needed for automated testing?

# Doctest Example

```
def utility(c, gamma):  
    """  
    Return CRRA utility of consumption "c" given risk aversion parameter "gamma."  
  
    ...(excluded for brevity)...  
  
    Tests  
    ----  
    Test a value which should pass:  
    >>> utility(1.0, 2.0)  
    -1.0  
  
    Test a value which should fail:  
    >>> utility(1.0, 1.0)  
    Traceback (most recent call last):  
    ...  
    ZeroDivisionError: float division by zero  
    """  
    return( c**(1.0 - gamma) / (1.0 - gamma) )
```

Figure : Doctest Example



# Version Control

Two parts: **version control** system (eg. Git) and **repository hosting** system (eg. Github, Stash, Bitbucket, etc...)

- Git: basic command-line system for managing archiving and workflow

```
localhost:~/workspace/solvingmicrodsop$ git status
On branch master
Your branch is up-to-date with 'origin/master'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   Python/Exploring-Estimation-Module.ipynb
        modified:   Python/Histogram_of_Beta_Bootstrap_Sample.png
        modified:   Python/Histogram_of_Rho_Bootstrap_Sample.png
        modified:   Python/params_json_replication.json

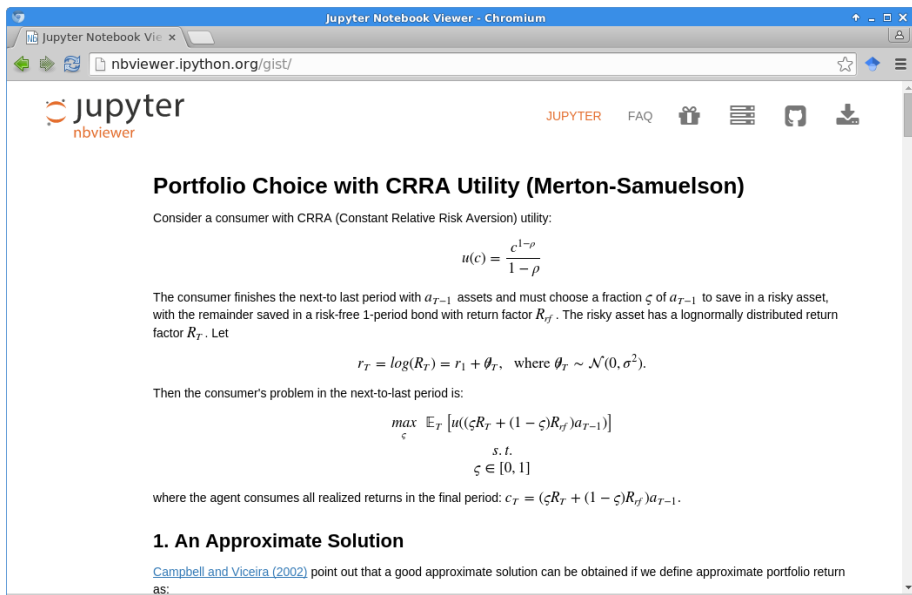
no changes added to commit (use "git add" and/or "git commit -a")
localhost:~/workspace/solvingmicrodsop$
```

- Github/Stash/Bitbucket/etc: centralized repository hosting with web interface

# Scientific Notebooks: Jupyter

- Recall “de-facto non-transparent” problem
- Simply reviewing scientific code can take time
- Solution: scientific notebooks combining code, math, visualization, interaction
- Must be:
  - low cost for researcher end-user
  - easy to use, share
  - archivable

# Scientific Notebooks: Jupyter



Jupyter Notebook Viewer - Chromium

Jupyter Notebook Vie x

nbviewer.ipython.org/gist/

**jupyter**  
nbviewer

JUPYTER FAQ

**Portfolio Choice with CRRA Utility (Merton-Samuelson)**

Consider a consumer with CRRA (Constant Relative Risk Aversion) utility:

$$u(c) = \frac{c^{1-\rho}}{1-\rho}$$

The consumer finishes the next-to last period with  $a_{T-1}$  assets and must choose a fraction  $\zeta$  of  $a_{T-1}$  to save in a risky asset, with the remainder saved in a risk-free 1-period bond with return factor  $R_{rf}$ . The risky asset has a lognormally distributed return factor  $R_T$ . Let

$$r_T = \log(R_T) = r_1 + \theta_T, \text{ where } \theta_T \sim \mathcal{N}(0, \sigma^2).$$

Then the consumer's problem in the next-to-last period is:

$$\begin{aligned} \max_{\zeta} \quad & \mathbb{E}_T [u((\zeta R_T + (1 - \zeta)R_{rf})a_{T-1})] \\ \text{s.t.} \quad & \zeta \in [0, 1] \end{aligned}$$

where the agent consumes all realized returns in the final period:  $c_T = (\zeta R_T + (1 - \zeta)R_{rf})a_{T-1}$ .

**1. An Approximate Solution**

[Campbell and Viceira \(2002\)](#) point out that a good approximate solution can be obtained if we define approximate portfolio return as:

Example of easy-to-create code “vignettes:”

- Create Jupyter notebook
- Host on Github/Bitbucket/etc as Gist, scientific archive
- Post to NBViewer

# License Tools

# Open Source and Related License

- Technological improvement under broad definition. . .
- Will only extremely briefly review these

From excellent US DoD “Lessons Learned” document [\[link\]](#)

# Open Source License Landscape

- Public domain
- Permissive
  - MIT, BSD
  - Apache
- “Goldilocks”
- Copyleft

See also license proliferation project [\[link\]](#)



# Open Source License Landscape

- Public domain
- Permissive
  - MIT, BSD
  - **Apache** <- FSF notes
- “Goldilocks”
- Copyleft

See also license proliferation project [\[link\]](#)

# Conclusion

Specific roles:

- Contributor:
  - direct contributor (low and high levels)
  - indirect contributor: ideas and wishlist, and others interested
- Incentivize-er:
  - assigning HW projects, translate and contribute projects
  - brainstorm about a “Journal of Economic Software” or equivalent
  - brainstorm about performance incentives related to above