Polyglot applications with R and Python

[BARUG Meeting]

Laurent Gautier

DMAC / CBS

November 13th, 2012
Disclaimer

- This is not about:

Only one language?
R and Python
Only one language?
R and Python

Disclaimer

• This is not about:
  • The comparative merits of scripting languages
Disclaimer

- This is not about:
  - The comparative merits of scripting languages
  - Only Python & R
Disclaimer

- This is not about:
  - The comparative merits of scripting languages
  - Only Python & R
- This is about:
Disclaimer

• This is not about:
  • The comparative merits of scripting languages
  • Only Python & R

• This is about:
  • Accessing natively libraries implemented in a different language
Only one language?
R and Python

Disclaimer

• This is not about:
  • The comparative merits of scripting languages
  • Only Python & R

• This is about:
  • Accessing natively libraries implemented in a different language
  • Bridging people and skill sets through a glue language
Disclaimer

- This is not about:
  - The comparative merits of scripting languages
  - Only Python & R
- This is about:
  - Accessing natively libraries implemented in a different language
  - Bridging people and skill sets through a glue language
  - Going from ideas to prototypes faster
Disclaimer

• This is not about:
  • The comparative merits of scripting languages
  • Only Python & R

• This is about:
  • Accessing natively libraries implemented in a different language
  • Bridging people and skill sets through a glue language
  • Going from ideas to prototypes faster
  • Putting some production into research
Disclaimer

- This is not about:
  - The comparative merits of scripting languages
  - Only Python & R

- This is about:
  - Accessing natively libraries implemented in a different language
  - Bridging people and skill sets through a glue language
  - Going from ideas to prototypes faster
  - Putting some production into research
  - Bringing research to production
Preamble

Scenario

- **data people**: Statisticians, data analysts
Preamble

Scenario

- **data people:** Statisticians, data analysts
- Data people have a method $M$
Preamble

Scenario

- **data people**: Statisticians, data analysts
- Data people have a method $M$
- Data people want to work on something new
Preamble

Scenario

- **data people**: Statisticians, data analysts
- Data people have a method $M$
- Data people want to work on something new
- Management wants an application for method $M$
Preamble

Scenario

- **data people**: Statisticians, data analysts
- Data people have a method $M$
- Data people want to work on something new
- Management wants an application for method $M$
- Management wants an application that uses method $M$
Only one language?

Polyglot programs

R and Python

Mapping types
Functions
Evaluation and memory
Building an application
Uniformity and coding standards
Uniformity and coding standards
Uniformity and coding standards
Only one language?  
R and Python  
Polyglot programs

Get the job done
Get the job done

Soloist

Only one language?  
R and Python  

Polyglot programs

• Multi-talented individual
• Documentation
• Teamwork
• Paired-programming
• Use the same tools
• Overlapping skills
Only one language? R and Python

Polyglot programs

Get the job done

Soloist

- Multi-talented individual
- Documentation?
Get the job done

Soloist

- Multi-talented individual
- Documentation?

Teamwork

Overlapping skills?

Paired-programming

Use the same tools?

Polyglot programs

Only one language?
R and Python
Get the job done

Soloist

• Multi-talented individual
• Documentation ?

Teamwork

• Paired-programming
• Use the same tools ?
• Overlapping skills ?
Monolithic development
Monolithic development

- Centralized
- Top-down
- Lot of planning
- Long development, mostly only usable when complete
- Stand in time
Maintainability

Why use **one** unique language?

- A legitimate managerial concern
- In places Java Certifications replaced general programming degrees
- Could good programmers matter more than the language?
- Back to finding a needle in a haystack
Modularity at the heart of UNIX philosophy.

- No branching logic, unless going for shell scripts.
- Shell script no often thought after for applications
- The birth of scripting languages (Sed, Awk, Perl, ...)
• Projects are cross-fields, cross-specialization
• Cost of specification - design - implementation too high
• Especially when the lifespan of the application is too short (or the user base too small).
Example from video games

- Engines (generally in C++)
- Scripting language for the ‘story’ and content
  - Python
  - Lua
  - Proprietary, others, ...
- Large projects (with a lot of money at stake)
- Diverse competences (3D engine $\neq$ story logic)
- When speed of development is more important than speed of execution

This can apply to other industries
- Pipelines in visual effects
- Bioinformatics
<table>
<thead>
<tr>
<th>1</th>
<th>Only one language?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Polyglot programs</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>2</th>
<th>R and Python</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mapping types</td>
<td></td>
</tr>
<tr>
<td>Functions</td>
<td></td>
</tr>
<tr>
<td>Evaluation and memory</td>
<td></td>
</tr>
<tr>
<td>Building an application</td>
<td></td>
</tr>
</tbody>
</table>
R

- Language for statistics, data analysis, and data visualization
- Unmatched\(^1\) number of libraries for anything having to do with data
- Specialized set of libraries for bioinformatics (Bioconductor)

\(^1\)Almost certainly
Python

- All-purpose scripting language
- Unmatched\(^2\) number of libraries for about anything
- Specialized sets of libraries for bioinformatics (Biopython, and a myriad smaller projects)

\(^2\)May be
Python (continued)

- Machine learning R does not have: PyBrain
- Visualization tools R does not have: Mayavis, Blender
Python is popular in Bioinformatics / DNA sequencing.

- Galaxy pipeline/server framework is in Python
- Some of the internal tools for the SOLiD are written in Python
- Ion Torrent Server is a Django server
- Oxford Nanopore control system is a server running Python
Why use anything else than R?

- Build an application
- Work with very large data
- ‘Just because it can be done in R doesn’t mean you should do it’

---

3 John Dennison, R Meetup presentation
R embedded in Python
rpy2

- Feels like a regular Python library
- Embeds an R process
- Can be thought of as a stateful library
Two main parts:

- Low-level interface
- High-level interface
Low-level interface

- Close to R’s C-API
- Let you do anything safe\(^4\) from that API
- Expose R data structures as Python *builtin* structures

\(^4\)or so is the intent
## Types

<table>
<thead>
<tr>
<th>R</th>
<th>rpy2</th>
<th>Python</th>
</tr>
</thead>
<tbody>
<tr>
<td>numeric</td>
<td><strong>Float</strong> SexpVector</td>
<td>float</td>
</tr>
<tr>
<td>integer</td>
<td><strong>Int</strong> SexpVector</td>
<td>int</td>
</tr>
<tr>
<td>char</td>
<td><strong>Str</strong> SexpVector</td>
<td>str</td>
</tr>
<tr>
<td>logical</td>
<td><strong>Bool</strong> SexpVector</td>
<td>bool</td>
</tr>
<tr>
<td>complex</td>
<td><strong>Complex</strong> SexpVector</td>
<td>complex</td>
</tr>
<tr>
<td>list</td>
<td><strong>List</strong> SexpVector</td>
<td>list</td>
</tr>
<tr>
<td>environment</td>
<td>SexpEnvironment</td>
<td>dict</td>
</tr>
<tr>
<td>function</td>
<td>SexpClosure</td>
<td>function</td>
</tr>
<tr>
<td>S4</td>
<td>SexpS4</td>
<td><strong>object</strong></td>
</tr>
<tr>
<td></td>
<td>SexpLang</td>
<td><strong>object</strong></td>
</tr>
<tr>
<td></td>
<td>SexpExtPtr</td>
<td><strong>object</strong></td>
</tr>
</tbody>
</table>
Vectors and arrays

- C-like: Contiguous blocks of memory
- R objects exposed to Python as sequences or C-like arrays, with or without copy
R

```r
v <- seq(1, 10)
v[1]  # select the first element
w <- v + 1  # add 1 to all elts
```
Only one language?  
R and Python

Mapping types
Functions  
Evaluation and memory  
Building an application

**R**

```r
v <- seq(1, 10)
v[1]  # select the first element
w <- v + 1  # add 1 to all elts
```

**rpy2.rinterface**

```python
import rpy2.rinterface as ri; ri.initr()
v = ri.IntSexpVector(range(1, 11))
v[0]  # select the first element
w = ri.IntSexpVector([x+1 for x in v])
```
Only one language? R and Python

Mapping types
- Functions
- Evaluation and memory
- Building an application

R

```r
v <- seq(1, 10)
v[1]  # select the first element
w <- v + 1  # add 1 to all elts
```

rpy2.rinterface

```python
import rpy2.rinterface as ri; ri.initr()
v = ri.IntSexpVector(range(1, 11))
v[0]  # select the first element
w = ri.IntSexpVector([x+1 for x in v])
```

rpy2.robjects

```python
import rpy2.robjects as ro
v = ro.IntVector(range(1, 11))
v[0]  # select the first element
w = v.ro + 1
```
Missing values

**NaN:**

A numeric data type value representing an undefined or unrepresentable value, especially in floating-point calculations.

- Also used for missing values.
- Is a standard.

**NA:**

- Used for missing values by R.
- Not a standard.

- Pitfall when passing data to C without copy/checks
- Applies to any C libraries (includes rpy2)
Functions

R functions can be called as if they were Python functions

```python
import rpy2.robjects as ro

f = ro.r("function(x, y) { 2 * (x + y) }")

f(1, 2)
```

- conversion on-the-fly
- translated signatures (dot-to-underscore)
Packages and modules

R

Namespaces attached to the search path

```r
> searchpaths()
[1] 
```

```
[2] 
```
```
[3] 
```
```
[4] 
```
```
[5] 
```
```
[6] 
```
```
[7] 
```
```
[8] 
```
```
[9] 
```
```
```
```
```
```
```
```
```
```
```
```
```

Python

Python modules as namespaces

```python
import os
os.path.basename('/path/to/a/file')
```
R packages (almost) as Python modules

```python
from rpy2.robjects.packages import importr
stats = importr('stats')
# PCA !
pc = stats.prcomp(m)
```
R scripts as modules!

```python
from rpy2.robjects.packages import SignatureTranslatedAnonymousPackage

# R code in a file as a package
with open('rflib.R') as f:
    code = ''.join(f.readlines())
    rf = SignatureTranslatedAnonymousPackage(code, "rf")

imp = rf.get_importance(dataf, response)
```
R environments

- Associate symbols to objects
- Exposed as Python dictionaries (key - value)

```r
env <- new.env()
assign('x', 123, envir = env)

y <- 456
```

```python
import rpy2.robjects as ro

env = ro.Environment()
env['x'] = 123

ro.globalenv['y'] = 456
```
R and callback functions

Common R idiom

```
# m: matrix of numerical values
f <- function(x) sum(x[x > 0])
res <- apply(m, 1, f)
```

How to do that with rpy2?

```
import rpy2.interactive as r
import rpy2.rinterface as ri
r_code = "function(x) sum(x[x > 0])"

tmp = ri.parse(r_code)

eval = r.packages.base.eval
r_func = eval(tmp)

r.base.apply(m, 1, r_func)
```

```python
def tmp(x):
    gnr = [elt for elt in x if elt > 0]
    return sum(gnr)

r_func = ri.rternalize(tmp)
r.base.apply(m, 1, r_func)
```
R and callback functions

Common R idiom

```r
# m: matrix of numerical values
f <- function(x) sum(x[x > 0])
res <- apply(m, 1, f)
```

How to do that with rpy2?

```python
import rpy2.interactive as r
import rpy2.rinterface as ri
r_code = ""
    function(x)
        sum(x[x > 0])
    ""
 tmp = ri.parse(r_code)
eval = r.packages.base.eval
r_func = eval(tmp)
r.base.apply(m, 1, r_func)
```
R and callback functions

Common R idiom

```r
# m: matrix of numerical values
f <- function(x) sum(x[x > 0])
res <- apply(m, 1, f)
```

How to do that with rpy2?

```python
import rpy2.interactive as r
import rpy2.rinterface as ri
def tmp(x):
    gnr = elt for elt in x \n        if elt > 0
    return sum(gnr)
r_func = ri.rternalize(tmp)
r.base.apply(m, 1, r_func)
```
Evaluation strategies

<table>
<thead>
<tr>
<th>R</th>
<th>Python</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Pass-by-value / Call-by-value</td>
<td>• Pass-by-reference</td>
</tr>
<tr>
<td>• Modifying an object locally is always safe</td>
<td>• Explicit request if copy</td>
</tr>
<tr>
<td>• Unnecessary copies</td>
<td></td>
</tr>
</tbody>
</table>

Rpy2 exposes R as if it was pass-by-reference
**Python**

```python
from rpy2.robjects.vectors import IntVector

def f(x):
    x[0] = 123
v = ro.IntVector(range(1, 11))
f(v)
```

**R**

```r
f <- function(x) {
    x[0] = 123
    return(x)
}
v = seq(1, 11)
v = f(v)
```
Memory management and garbage collection
Memory management and garbage collection

**R**
- Tracing GC (check for reachability)
- *R_PreciousList*
## Memory management and garbage collection

<table>
<thead>
<tr>
<th>Language</th>
<th>Features</th>
</tr>
</thead>
</table>
| **R**    | - Tracing GC (check for reachability)  
           - `R_PreciousList` |
| **Python** | - Reference counting  
               - Tracing GC |
Memory management and garbage collection

R

- Tracing GC (check for reachability)
- \textit{R\_PreciousList}

Python

- Reference counting
- Tracing GC

- Bridge different memory models
- Intermediate reference counting of R objects exposed
- That part could become very generic.
R objects exposed to R

```python
import rpy2.rinterface as ri
ri.initr()
baseenv = ri.baseenv
letters = baseenv.get('letters')
```
import rpy2.rinterface as ri
ri.initr()
base = ri.baseenv
letters = base['letters']
>>> letters = base['letters']
>>> letters.rid  # varies
123456
>>> letters.__sexp_refcount__
1
>>> letters2 = base['letters']
>>> letters2.__sexp_refcount__
2
>>> letters.__sexp_refcount__
2
>>> letters_2.rid  # same R ID
123456
<table>
<thead>
<tr>
<th>Exceptions</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>RRuntimeError:</strong> error while evaluating R code</td>
</tr>
<tr>
<td><strong>KeyError:</strong> symbol not found in an environment</td>
</tr>
<tr>
<td><strong>ValueError:</strong> invalid value passed to an rpy2 function</td>
</tr>
</tbody>
</table>
Only one language? R and Python

Mapping types
Functions
Evaluation and memory
Building an application

Performances

```r
function(x) {
    total = 0;
    for (elt in x) {
        total <- total + elt
    }
}
```

<table>
<thead>
<tr>
<th>Function</th>
<th>Sequence</th>
<th>Speedup</th>
</tr>
</thead>
<tbody>
<tr>
<td>R</td>
<td></td>
<td>1.00</td>
</tr>
<tr>
<td>R compiled</td>
<td></td>
<td>6.52</td>
</tr>
<tr>
<td>R builtin</td>
<td></td>
<td>329.29</td>
</tr>
<tr>
<td>pure python</td>
<td>FloatVector</td>
<td>0.51</td>
</tr>
<tr>
<td>builtin python</td>
<td>FloatVector</td>
<td>0.54</td>
</tr>
<tr>
<td>pure python</td>
<td>SexpVector</td>
<td>7.45</td>
</tr>
<tr>
<td>builtin python</td>
<td>SexpVector</td>
<td>20.92</td>
</tr>
<tr>
<td>builtin python</td>
<td>array.array</td>
<td>53.62</td>
</tr>
<tr>
<td>builtin python</td>
<td>list</td>
<td>90.47</td>
</tr>
</tbody>
</table>

*R through rpy2 can be faster than R*
Let’s build a web application

- Why do that?
  - Allow access to computing resources
  - Use the UI of the browser
  - Good example

- Micro web framework: Flask
Hello world with Flask

```python
from flask import Flask
app = Flask(__name__)

@app.route('/'

def hello_world():
    return 'Hello World!'

if __name__ == '__main__':
    app.run()

python hello.py
```
Importance of variables with random forest

1. Data in a CSV file
2. Use R to compute a random forest and compute importance of variables
3. Make a pretty plot with *ggplot2*
Only one language?

R and Python

Mapping types
Functions
Evaluation and memory
Building an application

data

dataf <- read.csv("/some/data/file.csv")
response <- 'var_name'

## importance of variables
library(randomForest)
get_importance <-
function (dataf, response) {
  fmla <- formula(paste(response, '˜ .'))
  dataf_rf <- randomForest(fmla, data = dataf,
                          keep.forest = FALSE,
                          importance = TRUE)
  imp <- importance(dataf_rf, type = 1)
  imp <- as.data.frame(imp[order(imp[,1]), , drop=FALSE])
  return(imp)
}
imp <- get_importance(dataf, response)

## plot
library(ggplot2)
get_plot <-
function (imp) {
  rn <- rownames(imp)
  rn <- factor(rn, levels=rn, ordered=TRUE)
  imp <- cbind(as.data.frame(imp), rn = rn)
  p = ggplot(imp) +
  geom_bar(aes(y = '%IncMSE', x = rn, fill = '%IncMSE')) +
  scale_x_discrete("Variables")
  return(p)
}
p <- get_plot(imp)
print(p)
Only one language?  
**R and Python**

## data
```
dataf <- read.csv("/some/data/file.csv")
response <- 'var_name'
```

## importance of variables
```
library(randomForest)
get_importance <- \textbf{function}(dataf, response) {
    fmla <- formula(paste(response, "˜ ."))
dataf_rf <- randomForest(fmla, data = dataf,  
    keep.forest = \textbf{FALSE},
    importance = \textbf{TRUE})
    imp <- \textbf{importance}(dataf_rf, type = 1)
    imp <- as.data.frame(imp[order(imp[,1]), , drop=\textbf{FALSE}])
    \textbf{return}(imp)
}
```
```
imp <- get_importance(dataf, response)
```

## plot
```
library(ggplot2)
get_plot <- \textbf{function}(imp) {
    rn <- rownames(imp)
    rn <- factor(rn, levels=rn, ordered=\textbf{TRUE})
    imp <- \textbf{cbind}(as.data.frame(imp), rn = rn)
    p = ggplot(imp) +
    \textbf{geom_bar}(aes(y = '%IncMSE',
        x = rn,
        fill = '%IncMSE')) +
    \textbf{scale_x_discrete}("Variables")
    \textbf{return}(p)
}
```
```
p <- get_plot(imp)
print(p)
```
R library

```r
get_dataframe <- function(filename) {
  return(read.csv(filename))
}

## importance of variables
library(randomForest)
get_importance <- function(dataf, response) {
  fmla <- formula(paste(response, '~ .'))
  dataf_rf <- randomForest(fmla, data = dataf,
                            keep.forest = FALSE, 
                            importance = TRUE)
  imp <- importance(dataf_rf, type = 1)
  imp <- as.data.frame(imp[order(imp[,1]), , drop=FALSE])
  return(imp)
}
```
R library

```r
library(ggplot2)

get_plot <- function(imp) {
  rn <- rownames(imp)
  rn <- factor(rn, levels=rn, ordered=TRUE)
  imp <- cbind(as.data.frame(imp), rn = rn)
  p = ggplot(imp) +
    geom_bar(aes(y = `%IncMSE`,
                 x = rn,
                 fill = `%IncMSE`)) +
    scale_x_discrete("Variables")
  return(p)
}

make_PNGplot <- function(imp, dir) {
  filename <- tempfile(tmpdir = dir, fileext = '.png')
  p <- get_plot(imp)
  png(filename)
  print(p)
  dev.off()
  return(basename(filename))
}
```
Only one language ?
R and Python
Mapping types
Functions
Evaluation and memory
Building an application

Python application

```python
import os
from flask import Flask, render_template, flash
from flask import url_for, send_from_directory
from flask import request
from werkzeug import secure_filename
from rpy2.robjects.packages import SignatureTranslatedAnonymousPackage

UPLOAD_FOLDER = '/tmp'

# R code as a package
with open('rflib.R') as f:
    code = ''.join(f.readlines())
rf = SignatureTranslatedAnonymousPackage(code, "rf")
```
Only one language?  
R and Python

Mapping types  
Functions  
Evaluation and memory  
Building an application

15 # create application
16 app = Flask(__name__)
17 app.secret_key = 'change this !!!'
18 app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER
19
20 # serve files
21 @app.route('/files/<filename>')</def files(filename):
23 return send_from_directory(UPLOAD_FOLDER, filename)
# create application
app = Flask(__name__)
app.secret_key = 'change this !!!'
app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER

# serve files
@app.route('/files/<filename>')</def files(filename):
    return send_from_directory(UPLOAD_FOLDER, filename)

def plot(dataf, response):
    # compute importance of variables
    imp = rf.get_importance(dataf, response)
    # plot into a file
    plot_fn = rf.make_PNGplot(imp, UPLOAD_FOLDER)[0]
    return url_for('files', filename = plot_fn)
# main function

```python
@app.route('/', methods=['GET', 'POST'])
def index():
    plot_url = None
    # test if data posted
    if request.method == 'POST':
        f = request.files['data']
        response = request.form['response']
        # test is file 'data' uploaded
        if f:
            # save the uploaded file
            filename = secure_filename(f.filename)
            f.save(os.path.join(app.config['UPLOAD_FOLDER'], filename))
            # get R data.frame from the file
            dataf = rf.get_dataframe(f.filename)
            # check if response variable is present
            if response in dataf.names:
                plot_url = plot(dataf, response)
            else:
                flash('No such response variable', category='error')
        else:
            flash('Invalid file extension', category='error')
    #
    return render_template('index.html', plot_url=plot_url)
```
<table>
<thead>
<tr>
<th>Only one language?</th>
<th>Mapping types</th>
</tr>
</thead>
<tbody>
<tr>
<td>R and Python</td>
<td>Functions</td>
</tr>
<tr>
<td></td>
<td>Evaluation and memory</td>
</tr>
<tr>
<td></td>
<td>Building an application</td>
</tr>
</tbody>
</table>

Showtime...
Next steps

- Generic library to bridge R to anything with a C API
- Julia and R (hopefully end of 2012)