GCAT Developer Guide

Version 0.1. September 15, 2015

# Get to Know GCAT

## Documentation

GCAT’s manual contains a good description of the software and is maintained up to date. It can be accessed by a link from any GCAT server, e.g. <http://gcat-pub.glbrc.org/resources/GCAT_users_manual.pdf>

There is also a peer-reviewed paper on GCAT:

Bukhman, Yury V., Nathan W. DiPiazza, Jeff Piotrowski, Jason Shao, Adam G. W. Halstead, Minh Duc Bui, Enhai Xie, and Trey K. Sato. “Modeling Microbial Growth Curves with GCAT.” *BioEnergy Research* 8, no. 3 (February 8, 2015): 1022–30. doi:10.1007/s12155-015-9584-3. See <http://link.springer.com/article/10.1007%2Fs12155-015-9584-3> or ask Yury for a reprint.

## GCAT Servers

Public: <http://gcat-pub.glbrc.org/>

GLBRC internal: <http://gcat.glbrc.org/>

Test: <http://gcat-test.glbrc.org/>

## GCAT project on GitHub

GCAT project page: <https://github.com/ybukhman/GCAT>

Milestone View: <https://github.com/ybukhman/GCAT/milestones>

Clone URL: <https://github.com/ybukhman/GCAT.git>

## Code architecture

Yury has various documents that describe the structure of the R back end and the Rails front end.

# Learn the tools

## Git

Book: <http://git-scm.com/book/en/v2>

Cheat Sheet: <https://training.github.com/kit/downloads/github-git-cheat-sheet.pdf>

Git on Windows: <http://blogs.msdn.com/b/kristol/archive/2013/07/02/the-git-command-line-101-for-windows-users.aspx>

## R

GCAT’s back end is an R package named “GCAT”. GCAT makes significant use of S4 classes and methods, although it is not consistently implemented in S4 throughout.

A GCAT developer must become familiar with the following:

* An “official” introduction to R: <https://cran.r-project.org/doc/manuals/r-release/R-intro.html>
* Hadley Wickham’s book on R packages, <http://r-pkgs.had.co.nz/> (need pretty much all of it)
* The chapter on S4 classes from Hadley Wickham’s “Advanced R” book, <http://adv-r.had.co.nz/OO-essentials.html#s4>.

Additional/alternative reading:

* For other introductions to S4, look at Bioconductor course materials, e.g. <http://www.bioconductor.org/help/course-materials/2010/AdvancedR/S4InBioconductor.pdf> .
* We have a couple of books on R programming at the office
* There are many other resources under “Documentation” at the R site, <https://www.r-project.org/>

## Rails

Overview

The front end of the GCAT app is built with Ruby on Rails. Ruby is a general-purpose scripting language, and Rails is a web framework, written in Ruby. Normally, this platform is used to build database-backed web apps, but in the case of GCAT, there is no database, and only Rails’ presentation and routing logic are used. GCAT is written with Ruby version 1.9.3, and Rails version 3.

Learning about Ruby on Rails

1. It can be useful to learn Ruby first. This is easier than trying to understand a new language and a web platform at the same time.
   1. There is a Lynda.com class on Ruby that is pretty good, though dated: <http://www.lynda.com/Ruby-tutorials/essential-training/47905-2.html>
   2. Darin has several Ruby books available, which may also be helpful.
2. There are many resources available for learning Rails.
   1. Hartl’s *Ruby on Rails Tutorial* is a very popular resource, and is available in both book form, and online (for free) at <https://www.railstutorial.org/book>. I believe you can use this tutorial without knowing any Ruby first, if you prefer to do so.
   2. Darin has several Rails books available (including an older version of the Hartl Tutorial)
   3. There is a Lynda.com Rails course, as well, but we haven’t used it before: <http://www.lynda.com/Ruby-Rails-tutorials/Ruby-Rails-4-Essential-Training/139989-2.html>
   4. There are MANY resources online for learning Ruby on Rails – if something isn’t working for you, try something else.
3. The Ruby Version Manager (RVM) is a good tool for maintaining Ruby and Rails installations on your computer: https://rvm.io/.

# Set up your development environment

## GLBRC Development VM

This VM comes pre-configured with all the necessary tools. Download the VM from file.glbrc.org under \shared\Bioinformatics Applications\VMWare image.

Use git to download GCAT project from GitHub.

## Windows

If not using GLBRC Development VM, follow these instructions to set up GCAT development environment on Windows

1. Install R from <https://cran.r-project.org/mirrors.html> . Do not uncheck the 32-bit version when installing, as this is what GCAT on Windows needs!

2. Install Rstudio (<https://www.rstudio.com/>)

3. Install GCAT's dependencies. See DESCRIPTION file. As of today, dependencies include pheatmap and gplots. Also install testthat. To do this, run the following command in an R session: **install.packages(c("pheatmap","gplots","testthat"))**

4. Install GCAT R package, e.g. using Rstudio's "Build & Reload" command. One can also do this from the Windows command prompt: run **R CMD INSTALL GCAT** in the top level GCAT/R folder.

5. Install python 2 - not sure if this is really necessary, but it won't hurt

6. Install enca: download from <https://www.john.geek.nz/2010/02/enca-binary-compiled-for-32-bit-windows/>

7. Install Rails from Railsinstaller.org, <http://railsinstaller.org/en>. GCAT needs an older distribution with Ruby 1.9.3 and Rails 3. Railsinstaller site provides a link to that distribution: <https://s3.amazonaws.com/railsinstaller/Windows/railsinstaller-2.2.5.exe>

8. Run **bundle install** in GCAT's Rails folder

9. Run **rails s** in GCAT's Rails folder

10. Open http://localhost:3000/ in a web browser

## Linux

If not using GLBRC Development VM, follow these instructions to set up GCAT development environment on Linux

### Install enca

Use your Linux package manager, e.g. on CentOS run **sudo yum install enca**

### Install R

R is available in Linux package repositories. However, you usually need to set things up in order to get the latest stable version. To see instructions for your Linux distribution, open a mirror under <https://cran.r-project.org/mirrors.html> and navigate to “[Download R for Linux](https://cran.rstudio.com/bin/linux/)”.

For example, the following notes are for CentOS.

Instructions: open <https://cran.rstudio.com/>, go to any mirror - Download R for Linux - redhat - README  
  
The Fedora RPMs for R have been ported to RHEL by the project Extra Packages for Enterprise Linux (EPEL). EPEL instructions: <http://fedoraproject.org/wiki/EPEL>  
  
In CentOS, you can install EPEL by running **yum install epel-release**. The epel-release package is included in the CentOS Extras repository, enabled by default.  
  
Then run **sudo yum install R** to install the R meta-package

### Install required add-on R packages

From an R session, install R packages pheatmap, gplots, testthat:

**> install.packages(c("pheatmap","gplots","testthat"))**

(Note: package "methods" comes with the core distribution and does not need to be installed).

### Install RStudio

See <https://www.rstudio.com/>

### Install and configure Ruby on Rails

Need Ruby 1.9.3 and Rails 3.

For Linux, the best tool overall for using and maintaining Ruby on Rails installations is called RVM (**R**uby **V**ersion **M**anager). Installing RVM will allow you to switch between different versions of the software on the fly, without worrying about conflicts or un-installing versions not currently in use. Information about RVM (including installation instructions) is at <https://rvm.io/>.

### Install and run GCAT

Install GCAT’s R package, e.g. using RStudio's "Build & Reload" command. You can also do this from the Linux shell: run **R CMD INSTALL GCAT** in the top level GCAT/R folder.

Run **bundle install** in GCAT's Rails folder

Run **rails s** in GCAT's Rails folder

Open http://localhost:3000/ in a web browser

# Run GCAT on your computer

## Run GCAT Rails server

Open a terminal in GCAT’s Rails folder and run the following commands:

**bundle install**

**rails s**

Note: you only need to run **bundle install** once

## Run GCAT from an R session

When developing the R back-end, it is useful to be able to run GCAT from an R session, without the web interface. This is accomplished by calling gcat.analysis.main(). Here is some example code.

INPUT.DIR = "H:/Projects/GCAT\_B05/code/git/GCAT/Testing/test\_input\_files/default\_examples"

INPUT.FILE = "single - YPDAFEXglucoseTests\_2-25-10.csv"

OUTPUT.DIR = "H:/Projects/GCAT\_B05/B05.21 GCAT 6/GCAT 6.1/3 AUC/temp"

# Run GCAT

library(GCAT)

setwd(INPUT.DIR)

time.input=1/3600

wells = gcat.analysis.main(file.list = INPUT.FILE, single.plate = T,

layout.file = NULL,

out.dir = OUTPUT.DIR, graphic.dir = OUTPUT.DIR,

add.constant = 0, blank.value = 0.04, start.index = 1,

growth.cutoff = 0.05,

use.linear.param=F, use.loess=F, smooth.param=0.1,

points.to.remove = integer(), remove.jumps = F,

time.input=time.input,

silent = F, verbose = T, return.fit = T, overview.jpgs = T)

This code reads in an example dataset from INPUT.DIR/INPUT.FILE, performs all modeling calculations and puts the output files into OUTPUT.DIR. Additionally, variable wells contains an array of well objects or, if GCAT failed, an error message.

# Contribute to GCAT

Use Git’s “Integration-Manager Workflow”. The general idea is that student programmers should use their own forks. New issues should be worked on in branches. Once the work on an issue is completed, the student should submit a pull request. This will give Yury and/or Darin a chance to review the student’s code before merging it with the master. For details, see <http://git-scm.com/book/en/v2/Distributed-Git-Distributed-Workflows#Integration-Manager-Workflow> and <http://git-scm.com/book/en/v2/Distributed-Git-Contributing-to-a-Project>.