Building containerized workflows for RNA-seq data using the BioDepot-workflow-Builder (BwB)

Ling-Hong Hung[1,*] lhhung@uw.edu
Trevor Meiss[1,*] tmeiss@uw.edu
Jayant Keswani[1] jk0805@uw.edu
Yuguang Xiong[2] yuguang.xiong@mssm.edu
Eric Sobie[2] eric.sobie@mssm.edu
Ka Yee Yeung[1,§] kayee@uw.edu

[1]Institute of Technology, University of Washington Tacoma, WA
[2]Ichahn School of Medicine at Mount Sinai, New York, NY

[*]Equal contributors

[§]Corresponding author

January 7, 2017

Abstract

We present BioDepot-workflow-Builder (BwB), a portable and open-source tool for creating bioinformatics workflows with a simple drag-and-drop graphical user interface. The individual components of the workflows are Docker containers which are available from public repositories or provided by the user. The use of software containers ensures that workflows will give identical results across different operating systems and hardware architectures. The use of Docker also allows for individual components to be deployed on the cloud. The modularity and ease of customization and installation of bioinformatics tools using BwB allows for researchers to efficiently test new workflows and compare competing algorithms. Since BwB itself is packaged in a Docker container, the setup is minimal. In particular, users only need to install Docker and have access to a web browser to begin creating and running workflows. As a proof-of-concept case study, we illustrated the feasibility of BwB by developing widgets for the RNA-seq differential expression analysis workflow employed by the NIH BD2K-LINCS Drug Toxicity Signature Generation Center at Mount Sinai. The app and all the containers are available on the BioDepot repository (`https://hub.docker.com/r/biodepot`).

1 Introduction

Analyses for big biomedical data are often computationally intensive, transforming data of many gigabytes in size and requiring many hours to complete. Examples include data produced from high-throughput sequencing technology. In particular, there are many applications of RNA sequencing (RNA-seq) data and no universal analyses pipelines exist for various applications and scenarios [1]. In the alignment

step of RNA-seq data, billions of short cDNA sequences obtained through ultra-high-throughput sequencing are aligned to a reference transcriptome [2]. Many alignment methods and tools have been proposed [3]. A single run of this alignment process can require over 20 gigabytes of RAM and take over a day to complete [4]. Therefore, it is crucial to develop analyses tools for the biomedical community that are computationally efficient.

As computational processes have become an integral part of many scientific studies and have accelerated in complexity, many of the software tools that researchers use have become more intricate with frequently updated third-party dependencies. However, academic tools often lack technical support and the biomedical researcher is faced with the task of installing, running and debugging the software. These issues are compounded by the fact that big data workflows often consist of a series of such tools, where each stage performs a transformation on the data and passes the output to the next stage [5].

Furthermore, academic software is often focused on publishing new methods and/or results, resulting in software with limited modularity and portability. In order to leverage the availability of open source software developed by the computational community, an integrated framework is essential to allow users to easily build workflows consisting of software tools written by different research groups.

Cloud computing has gained popularity in the analyses of biomedical big data [6, 7]. A key advantage of cloud computing is that computational resources can be paid for on demand, providing an alternative to purchasing and maintaining private clusters. In order to reach the largest audience, bioinformatics tools should not only support cloud computing but also support multiple cloud providers, such as Amazon Web Services (AWS) [8], Microsoft Azure [9], and Google Cloud [10].

Additionally, reproducibility is crucial to scientific research [11]. This extends to the research produced using computational tools [12]. The versions of all software dependencies used must be clearly defined, and it is vital that the results obtained from these computational tools are reproducible across all operating systems and hardware architectures.

These issues can all be resolved by using software containers to encapsulate all the dependencies within a single environment. Docker [13], one of the most popular container solutions, is supported by all major cloud providers (AWS, Azure, and Google Cloud) and available on for many operating systems including Windows, Linux and MacOS. The definitions of individual Docker containers include all dependencies and their versions. Users can download, install, and run the entire container with one command. A recent study found that the use of Docker containers had numerous advantages over traditional approaches to reproducing computational scientific work [14]. Another study performed using Docker with bioinformatics workflows found only a minor impact on performance that medium and long running tasks would find negligible [15].

1.1 Our Contributions

While the utilization of Docker solves the problems of installation and reproducibility of bioinformatics tools, the command-line based Docker may be difficult and frustrating for a biomedical researcher with limited programming and technical experience. In order to gain wide adoption in the biomedical community, the user-experience must be intuitive and efficient. In this paper, we present BioDepot-workflow-Builder (BwB) that makes use of a graphical user-interface (GUI), where users can drag and drop widgets onto a canvas to build bioinformatics analyses

pipelines. As an additional feature, the GUI is also wrapped inside a Docker container and can be accessed through most major web browsers. This means that the only requirement for creating and running BioDepot workflows is to install Docker.

As a proof-of-concept, we containerized the standard operating procedures (SOP) for processing RNAseq data developed by the NIH BD2K-LINCS Drug Toxicity Signature (DToxS) Generation Center of the Ichahn School of Medicine at Mount Sinai in New York [16]. The overall goal of DToxS is to use genomic and proteomic high-throughput measurements coupled with medium-throughput experimental measurement of protein states as the basis for computational analysis that integrates network analyses with structural constraints and dynamical models to identify signatures that predict and mitigate toxicity induced by individual drugs.

2 Background and Related Work

Although the initial release of Docker was in 2013, many have already acknowledged the numerous benefits Docker provides to the bioinformatics community and have begun producing repositories of containerized tools. BioContainers is one such repository created by Felipe da Veiga Leprevost, currently harboring over thirty Docker images [17]. BioShaDock is a another community driven registry of Docker-based bioinformatics tools [18]. BioShaDock improves on BioContainers by providing authentication, access control, container meta-data, and search capabilities.

Other containerization efforts provide additional tools to faciliate the construction of workflows. BioBoxes for example, BioBoxes [19] uses an YAML file to define the inputs and outputs of different containers. NGSeasy developed at the NIHR Biomedical Research Centre for Mental Health and Biomedical Research Unit for Dementia in London, UK [20], places all of the next generation sequencing (NGS) tools into one base image and encapsulates each NGSeasy pipeline component in separate containers. Dockstore, developed by the Cancer Genome Collaboratory [21] provides containers for their Toil scheduler [7] which supports the Common Workflow Language (CWL) for defining bioinformatics pipelines from the available component containers.

None of the above repositories provide a GUI for creating and running modular workflows. Seven Bridges, a commercial service for conducting cloud-based bioinformatics analyses [22] does support a GUI where users drag multiple ”apps” onto a canvas and connect them together to define a workflow. However, this is closed-source and requires a paid subscription. The closest free and open-source alternative to BwB is Galaxy, which is a web-based platform that can be used to schedule and create bioinformatics workflows [23]. However, while Galaxy itself is containerized, the components comprising the workflows are not. As a result, Galaxy workflows are not portable and require the platform itself to run.

In the machine learning domain, there is Orange, which is an open-source machine learning and data visualization tool with a drag-and-drop interface to create interactive data analysis workflows [24]. Orange does provide a very limited set of tools for bioinformatics analyses in addition to the the machine learning suite. However, tools are not containerized and are written in Python. As a result, tools written in other languages are not compatible without additional Python bindings and much more work is required to port modules to Orange. Fortunately, Orange does allow for third-party widgets to be added. We used this feature to construct a Docker Remote API Python wrapper (docker-py [25]) and allow the GUI to interact with software containers.

While Orange provides an excellent foundation for adding repositories of tools

and creating workflows, it comes with its own set of dependencies and installation frustrations. BwB containerizes the Orange GUI itself using GUIdock-noVNC. Users can install and run BWB with a single Docker command and then interact with the GUI via a web browser.

3 DToxS RNA-seq data processing SOP

Figure 1 summarizes the steps in the RNA-seq data processing SOP developed by DToxS at Mount Sinai. In particular, we created Docker containers by following the standard operating procedures for sequence alignment [26] and differential expression analysis [27].

The alignment module uses the Burrows-Wheeler Aligner (BWA) [28] to align billions of short cDNA sequence reads to a human reference genome and then obtains read counts of each of the genes expressed in the samples. The BWA software package, written in C, is designed to map sequences against a large reference genome and allows for mismatches and gaps. The sequence reads were generated by DToxS and were formatted as FASTQ files, a common format for nucleotide sequence data. BWA outputs the obtained alignment in the standard SAM (sequence alignment/map) format. A custom script from DToxS translates the SAM files into read counts of genes.

The analysis module uses edgeR, a Bioconductor package written in R [29], to find the genes that were differentially expressed between the samples under different drug treatments. With traditional transcriptomics studies based on microarray technologies, the abundance of a transcript is measured as a florescence intensity. This workflow uses digital gene expression (DGE) data, which measures the abundance of a transcript with a read count, making traditional procedures not directly applicable to DGE. Empirical analysis of DGE data in R (edgeR) determines the differential expression of this data using an overdispersed Poisson (negative binomial) model and an empirical Bayes procedure.

4 Implementation

4.1 Containerizing the SOP

We created Docker containers for each of the alignment stage and the differential expression stage in the DToxS SOP. This was done by creating Dockerfiles that describe the dependencies and commands used to run each stage. The Dockerfiles were then used to build a Docker image, which is stored on the cloud and is publicly available to download and run [30]. Both Docker images start with a base Ubuntu image and then install the necessary dependencies.

4.2 Adding widgets to Orange

We wrote Python scripts to add widgets to the Orange software for each stage of the DToxS workflow. Specifically, we used docker-py [25] to automatically download and run the containers for each of the sequence alignment and differential expression analysis steps. The widgets give visual feedback for the download progress and can also provide progress for the status of the Docker containers. Adding the containers to the Orange GUI also involved creating a new `Directory` widget to notify the DToxS alignment container of the locations of the sequence and reference

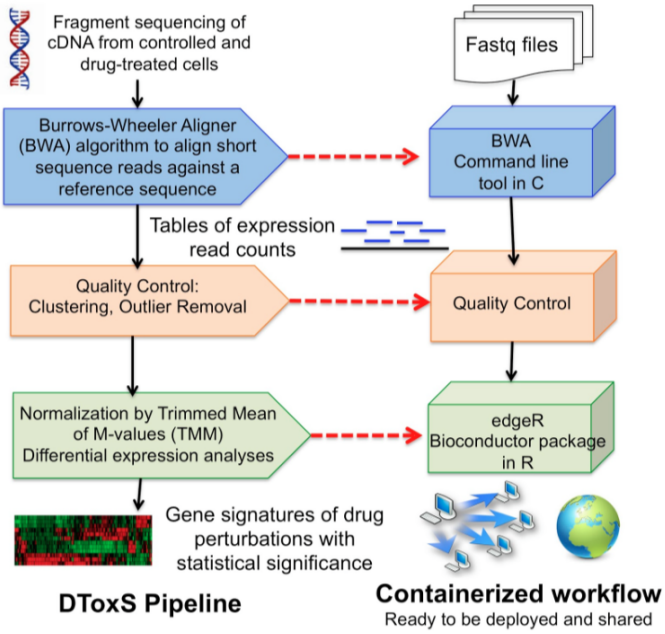


Figure 1: Summary of the RNA-seq stand operating procedure (SOP) from DToxS. There are two main stages in the SOP: the sequence alignment step and the differential expression analyses step. A Docker container is created to correspond to each of these steps.

files. When the DToxS container is run, the sequence and reference directories are mounted inside the container as a shared volume, allowing the container to read and write data.

Finally, another Orange widget was developed to provide information on the Docker processes. This Docker Info widget allows the user to view and remove downloaded Docker images, to view, stop, and remove running Docker containers, and to view basic information such as the Docker version and the hardware it is running on top of. The use of this tool is not necessary for creating a bioinformatics workflow, but can be useful when developing tools and to provide deeper insight for the users.

4.3 Graphical User Interface

Figure 2 shows the BioDepot-workflow-Builder (BwB) GUI developed to allow users to create bioinformatics workflows using individually containerized tools. The available tools are shown in the panel on the left. The workflow constructed on the canvas uses the Docker images developed for the alignment and differential expression analysis stages of the DToxS workflow. The user specifies the directory location of both the FASTQ sequences and the reference transcriptome. These directories are connected to the alignment widget, and then the alignment widget connects directly to the analysis widget. Finally, the analysis widget is connected to a directory to show the location of any output files and to a Data Table widget to show the top 40 differentially expressed genes in the experiment.

For each stage of the workflow, once the necessary inputs are set, the underlying container begins running automatically. The Docker image is also downloaded automatically if it does not already exist on the machine where the container is meant to be run. The user is given satisfying visual feedback for both downloading and run-

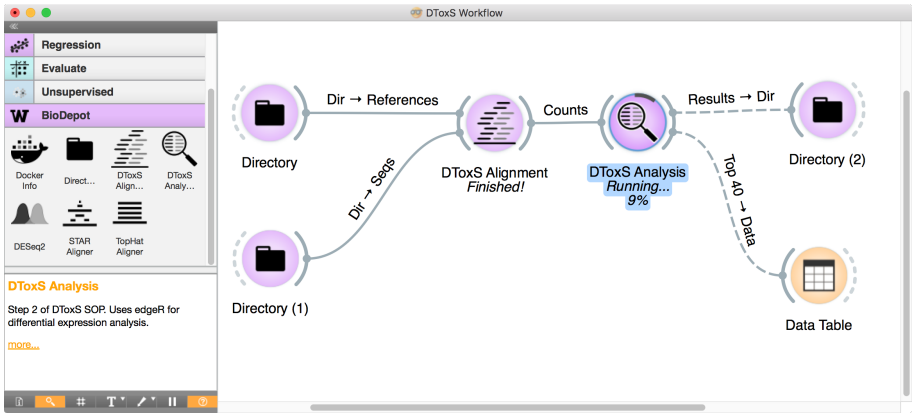


Figure 2: The BioDepot-workflow-Builder (BwB) graphical-user interface. Users can create custom workflows by dragging and dropping widgets onto the canvas. The user then connects the widgets to direct the flow of data.

ning of the containers. The portability of the individual components is demonstrated with the additional sequence alignment and differential expression analysis widgets shown in the left panel.

5 Conclusions

The BioDepot project builds upon many of the ongoing efforts to enhance the reproducibility of computational research using Docker containers. The new BioDepot-workflow-Builder (BWB) aims to provide a user-friendly, modular and open-source graphical user interface to create bioinformatics workflows. Our target audience is a typical biomedical researcher with limited programming and technical training. The use of Docker containers as the foundational technology guarantees that the tools will run identically across various operating systems and also allows for individual components of the workflows to be deployed on the cloud. The containerization of the BioDepot-workflow-Builder (BwB) GUI itself extends the ease of installation and deployment to the fullest extent, as users only need to install Docker and have access to a web browser.

Most importantly, BwB can be used to build different workflows by combining interchangeable and encapsulated widgets, allowing researchers to easily test alternative algorithms and observe how the outputs differ. Each of these widgets call a Docker container to execute software tools that could potentially be written in a different programming language, require different system configurations and/or developed by different research groups.

6 Availability and requirements

- Project name: BioDepot-workflow-Builder (BwB)
- URL: `https://github.com/BioDepot/BioDepot-workflow-builder`
- Contents available for download: Docker Images, Dockerfiles, installation scripts, execution scripts and demo video.
- Operating system(s): Linux, Mac OSX, Microsoft Windows.

- Programming language(s): Python, HTML, JavaScript
- License: MIT License

7 Acknowledgments

Ling-Hong Hung and Ka Yee Yeung are supported by NIH grant U54HL127624. Yuguang Xiong and Eric Sobie are supported by NIH grant U54HG008098. We would also like to thank Microsoft Azure for computing resources and Wes Lloyd for discussions.

References

[1] A. Conesa, P. Madrigal, S. Tarazona, D. Gomez-Cabrero, A. Cervera, A. McPherson, M. W. Szcześniak, D. J. Gaffney, L. L. Elo, X. Zhang, and A. Mortazavi, "A survey of best practices for rna-seq data analysis," *Genome Biology*, vol. 17, no. 1, pp. 1–19, 2016.

[2] A. Mortazavi, B. A. Williams, K. McCue, L. Schaeffer, and B. Wold, "Mapping and quantifying mammalian transcriptomes by rna-seq," *Nature methods*, vol. 5, no. 7, pp. 621–628, 2008.

[3] P. G. Engstrom, T. Steijger, B. Sipos, G. R. Grant, A. Kahles, T. R. Consortium, G. Ratsch, N. Goldman, T. J. Hubbard, J. Harrow, R. Guigo, and P. Bertone, "Systematic evaluation of spliced alignment programs for rna-seq data," *Nat Meth*, vol. 10, pp. 1185–1191, 12 2013.

[4] A. Dobin, C. A. Davis, F. Schlesinger, J. Drenkow, C. Zaleski, S. Jha, P. Batut, M. Chaisson, and T. R. Gingeras, "Star: ultrafast universal rna-seq aligner," *Bioinformatics*, vol. 29, no. 1, pp. 15–21, 2013.

[5] J. Leipzig, "A review of bioinformatic pipeline frameworks," *Briefings in bioinformatics*, p. bbw020, 2016.

[6] L. D. Stein, "The case for cloud computing in genome informatics," *Genome Biology*, vol. 11, no. 5, p. 207, 2010.

[7] J. Vivian, A. Rao, F. A. Nothaft, C. Ketchum, J. Armstrong, A. Novak, J. Pfeil, J. Narkizian, A. Deran, A. Musselman-Brown, H. Schmidt, P. Amstutz, B. Craft, M. Goldman, K. Rosenbloom, M. Cline, B. OConnor, M. Hanna, C. Birger, W. J. Kent, D. A. Patterson, A. D. Joseph, J. Zhu, S. Zaranek, G. Getz, D. Haussler, and B. Paten, "Rapid and efficient analysis of 20,000 rna-seq samples with toil," *bioRxiv 062497*, 2016.

[8] `https://aws.amazon.com`.

[9] `https://azure.microsoft.com`.

[10] `https://cloud.google.com`.

[11] B. A. Nosek, G. Alter, G. Banks, D. Borsboom, S. Bowman, S. Breckler, S. Buck, C. Chambers, G. Chin, G. Christensen, *et al.*, "Promoting an open research culture," *Science*, vol. 348, no. 6242, pp. 1422–1425, 2015.

[12] R. D. Peng, "Reproducible research in computational science," *Science*, vol. 334, no. 6060, pp. 1226–1227, 2011.

[13] `https://www.docker.com`.

[14] C. Boettiger, "An introduction to docker for reproducible research," *SIGOPS Oper. Syst. Rev.*, vol. 49, pp. 71–79, Jan. 2015.

[15] P. Di Tommaso, E. Palumbo, M. Chatzou, P. Prieto, M. L. Heuer, and C. Notredame, "The impact of docker containers on the performance of genomic pipelines," *PeerJ*, vol. 3, p. e1273, 9 2015.

[16] `http://research.mssm.edu/pst/DToxS/index.html`.

[17] `https://biocontainers.pro`.

[18] F. Moreews, O. Sallou, H. Mnager, Y. Le bras, C. Monjeaud, C. Blanchet, and O. Collin, "Bioshadock: a community driven bioinformatics shared docker-based tools registry [version 1; referees: 2 approved]," *F1000Research*, vol. 4, no. 1443, 2015.

[19] `http://bioboxes.org`.

[20] A. Folarin, R. Dobson, and S. Newhouse, "Ngseasy: a next generation sequencing pipeline in docker containers [version 1; referees: 3 approved with reservations]," *F1000Research*, vol. 4, no. 997, 2015.

[21] `https://dockstore.org`.

[22] `https://www.sbgenomics.com`.

[23] `https://galaxyproject.org`.

[24] `http://orange.biolab.si`.

[25] `https://docker-py.readthedocs.io`.

[26] `https://martip03.u.hpc.mssm.edu/pdf/DToXS_SOP_CO-3.1_Generation_of_Transcript_Read_Counts.pdf`.

[27] `https://martip03.u.hpc.mssm.edu/pdf/DToXS_SOP_CO-4.1_Identification_of_Differentially_Expressed_Genes.pdf`.

[28] L. H and D. R, "Fast and accurate short read alignment with burrows-wheeler transform," *Bioinformatics*, vol. 25, no. 14, pp. 1754–60, 2009.

[29] M. D. Robinson, D. J. McCarthy, and G. K. Smyth, "edger: A bioconductor package for differential expression analysis of digital gene expression data," *Bioinformatics*, vol. 26, no. 1, pp. 139–140, 2010.

[30] `https://hub.docker.com`.

Biodepot-workflow-builder

USER MANUAL

Latest version available at
https://github.com/BioDepot/BioDepot-workflow-builder

BioDepot can be used to build bioinformatics workflows by combining interchangeable and encapsulated widgets, allowing researchers to easily test other algorithms and observe how the outputs differ. Each of these widgets call a Docker container to execute software tools that could potentially be written in a different programming language, require different system configurations and/or developed by different research groups.

This manual demonstrates the installation and instructions to run orange inside docker.

Docker Image : https://hub.docker.com/r/biodepot/guidock-lite-orange-qt5
Dockerfile : https://github.com/BioDepot/BioDepot-workflow-builder

---

Authors

Ling-Hong Hung*, Trevor Meiss*, Jayant Keswani, Ka Yee Yeung

Institute of Technology
University of Washington
Tacoma, WA, 98402, USA

*Equal contributors.
Corresponding author: Ka Yee Yeung (kayee@uw.edu)

Table of content

1. Docker Installation
    - Linux
    - Mac OS X
    - Windows
2. Running the Orange application
3. On the Cloud
    - AWS
4. Demo

Docker Installation

Linux

Installing Docker

To install docker go to the terminal and follow the steps

1. Update your APT package index.

   ```
   #sudo apt-get update
   ```

2. Install Docker.

   ```
   #sudo apt-get install docker-engine
   ```

3. Start the docker daemon.

   ```
   #sudo service docker start
   ```

4. Verify docker is installed correctly.

   ```
   #sudo docker run hello-world
   ```

The last command downloads a test image and runs it in a container. When the container runs, it prints an informational message. Then, it exits.

For more information please refer to -
https://docs.docker.com/engine/installation/linux/ubuntulinux/

Mac OS X

Installing Docker

Your Mac must be running OS X 10.8 “Mountain Lion” or newer to run Docker software or If you have Mac OS X 10.10.3 Yosemite or newer, consider using Docker for Mac instead. It runs natively on the Mac, so there is no need for a preconfigured Docker QuickStart shell. It uses xhyve for virtualization, instead of VirtualBox. Full install prerequisites are provided in the Docker for Mac topic in Docker for Mac.

To download to the package - https://www.docker.com/products/docker-toolbox

1. Install Docker Toolbox by double-clicking the package or by right-clicking and choosing “Open” from the pop-up menu.
2. The installer launches an introductory dialog, followed by an overview of what’s installed.
3. Press Continue to install the toolbox.
4. The installer presents you with options to customize the standard installation.
5. By default, the standard Docker Toolbox installation:
    - installs binaries for the Docker tools in `/usr/local/bin`
    - makes these binaries available to all users
    - updates any existing VirtualBox installation
6. For now, don’t change any of the defaults.
7. Press Install to perform the standard installation.
8. The system prompts you for your password.
9. Provide your password to continue with the installation.
10. When it completes, the installer provides you with some shortcuts. You can ignore this for now and click Continue.
11. Then click Close to finish the installer.

To run a Docker container:

- create a new (or start an existing) Docker Engine host running
- switch your environment to your new VM
- use the docker client to create, load, and manage containers

Once you create a machine, you can reuse it as often as you like. Like any Virtual Box VM, it maintains its configuration between uses.

1. Open the Launchpad and locate the Docker Quickstart Terminal icon.
2. Click the icon to launch a Docker Quickstart Terminal window.
3. The terminal does a number of things to set up Docker Quickstart Terminal for you.
4. Click your mouse in the terminal window to make it active.
5. The prompt is traditionally a `$` dollar sign. You type commands into the *command line* which is the area after the prompt. Your cursor is indicated by a highlighted area or a `|` that appears in the command line. After typing a command, always press RETURN.
6. Type the `docker run hello-world` command and press RETURN.

The last command downloads a test image and runs it in a container. When the container runs, it prints an informational message. Then, it exits.

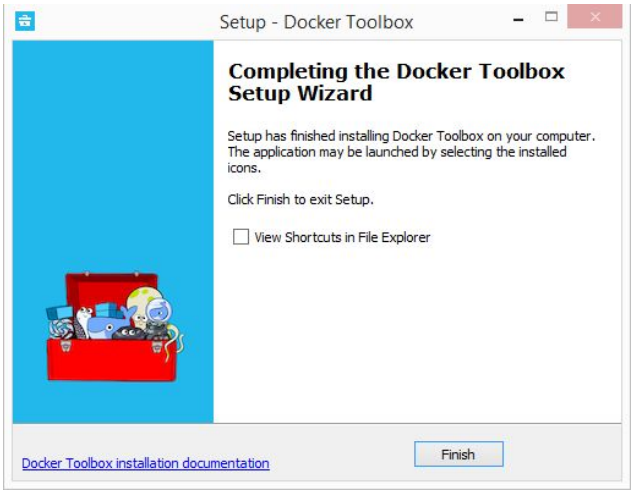
Windows

Installing Docker

If you have a newer system, specifically 64bit Windows 10 Pro, with Enterprise and Education (1511 November update, Build 10586 or later), consider using Docker for Windows instead. It runs natively on the Windows, so there is no need for a preconfigured Docker QuickStart shell. It also uses Hyper-V for virtualization.

To download to the package - https://www.docker.com/products/docker-toolbox

1. Go to folder where the installation file is saved and run the installation file.
2. Click the installer link to download.
3. Install Docker Toolbox by double-clicking the installer.
4. The installer launches the “Setup - Docker Toolbox” dialog.
5. If Windows security dialog prompts you to allow the program to make a change, choose Yes. The system displays the Setup - Docker Toolbox for Windows Wizard.
6. Press Next to accept all the defaults and then Install.
7. Accept all the installer defaults. The installer takes a few minutes to install all the components:
8. When notified by Windows Security the installer will make changes, make sure you allow the installer to make the necessary changes.
9. When it completes, the installer reports it was successful:

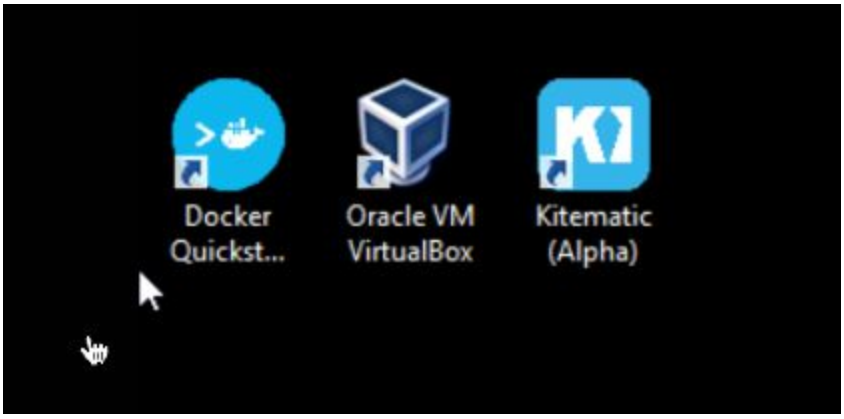


10. Uncheck “View Shortcuts in File Explorer” and press Finish.

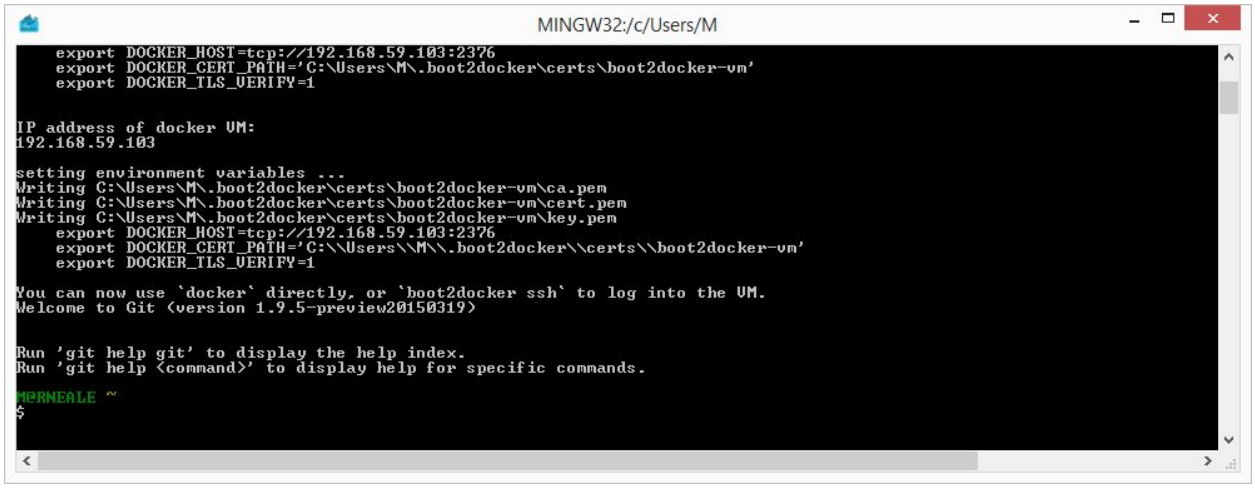
To run a Docker container:

The installer places Docker Toolbox and VirtualBox in your Applications folder. In this step, you start Docker Toolbox and run a simple Docker command.

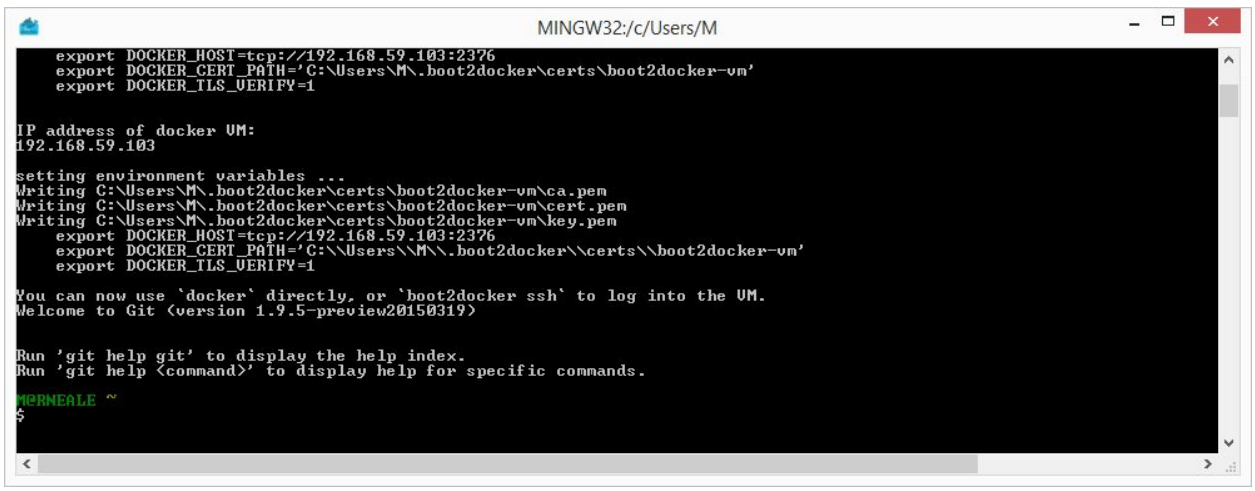
1. On your Desktop, find the Docker Toolbox icon.



2. Click the icon to launch a Docker Toolbox terminal.
3. If the system displays a User Account Control prompt to allow VirtualBox to make changes to your computer. Choose Yes.
4. The terminal does several things to set up Docker Toolbox for you. When it is done, the terminal displays the `$` prompt.



5. The terminal runs a special `bash` environment instead of the standard Windows command prompt. The `bash` environment is required by Docker.
6. Make the terminal active by click your mouse next to the `$` prompt.



7. The prompt is traditionally a `$` dollar sign. You type commands into the *command line* which is the area after the prompt. Your cursor is indicated by a highlighted area or a `|` that appears in the command line. After typing a command, always press RETURN.
8. Type the `docker run hello-world` command and press RETURN.

The last command downloads a test image and runs it in a container. When the container runs, it prints an informational message. Then, it exits.

Running the Orange application

After you have installed docker on your machine, launch docker and type the following commands on Docker terminal to install and run the Orange application (same across all the platforms)

1. Downloading the image
   `# docker pull biodepot/guidock-lite-orange-qt5:biowidgets`

```
➜ ~ docker pull biodepot/guidock-lite-orange-qt5
Using default tag: latest
latest: Pulling from biodepot/guidock-lite-orange-qt5

2f12ed5a7535: Pull complete
7285a81a1125: Pull complete
2beb8d49cf93: Pull complete
b303b1bb2d71: Pull complete
fb3777a1ba1a: Pull complete
496dc30b7448: Pull complete
f6c2142d45a0: Pull complete
0b209c20cee8: Pull complete
a0159e67f44f: Pull complete
9c56635862d6: Pull complete
90c2b4b10280: Pull complete
5f1eda116f31: Pull complete
c6e46097f989: Pull complete
7ae8b433a716: Pull complete
5bfca92b026f: Pull complete
3eaf7df56ef6: Pull complete
ab4b95b4c5c1: Pull complete
0a459696e8d3: Pull complete
08a4e138318c: Downloading  44.3 MB/141.4 MB
1aa01a2b566c: Download complete
b7c24266317a: Download complete
a6daaad548ae: Downloading 3.769 MB/56.26 MB
14cf6e060447: Download complete
83a43f1741f4: Waiting
ae4f04fc151c: Waiting
```

2. Running the application
   `# docker run -it -v /var/run/docker.sock:/var/run/docker.sock -p 6080:6080 biodepot/guidock-lite-orange-qt5:biowidgets`

Once the container is running you can open any of the browser to interact with it by using the default alpine port 6080 or you can check the port and IP at which the service is running by using docker command – `# docker ps`

Type the URL http://dockerMachineIPAddress:6080 on the browser and Orange window should popup.

```
Last login: Sat Nov 26 22:53:41 on ttys000
➜  ~ docker ps
CONTAINER ID        IMAGE                                COMMAND                  CREATED             STA
TUS                 PORTS                                NAMES
0f48cc7f132a        biodepot/guidock-lite-orange-qt5     "/root/startup.sh"       24 seconds ago      Up
23 seconds          0.0.0.0:6080->6080/tcp, 6800/tcp     awesome_yonath
```

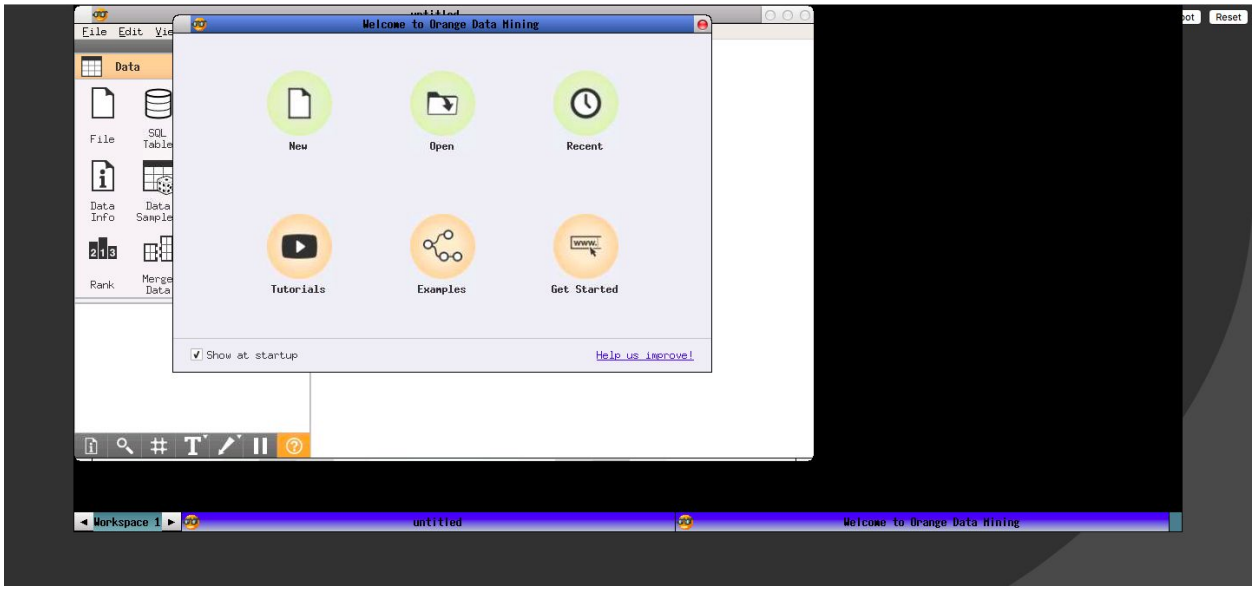


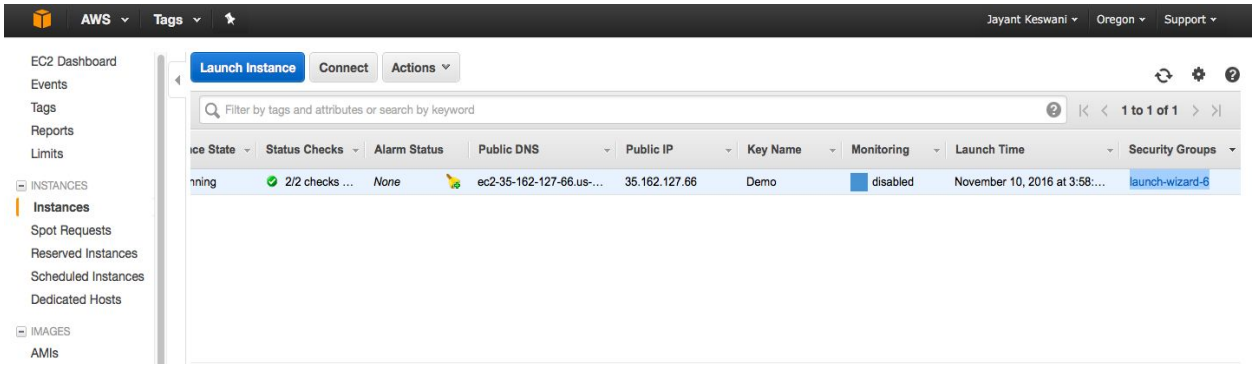
Fig: Orange running in browser

On The Cloud

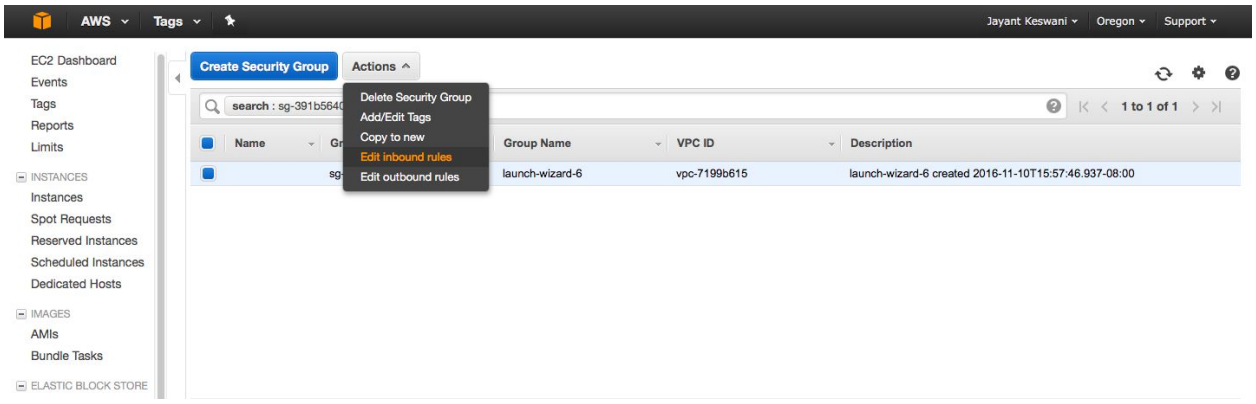
On the cloud, GUIdock-lite-orange can also be run on any cloud instances. Please refer to the Linux and Windows to install GUIdock-lite-orange on the cloud corresponds with the operating system installed for the cloud instance.

Amazon AWS

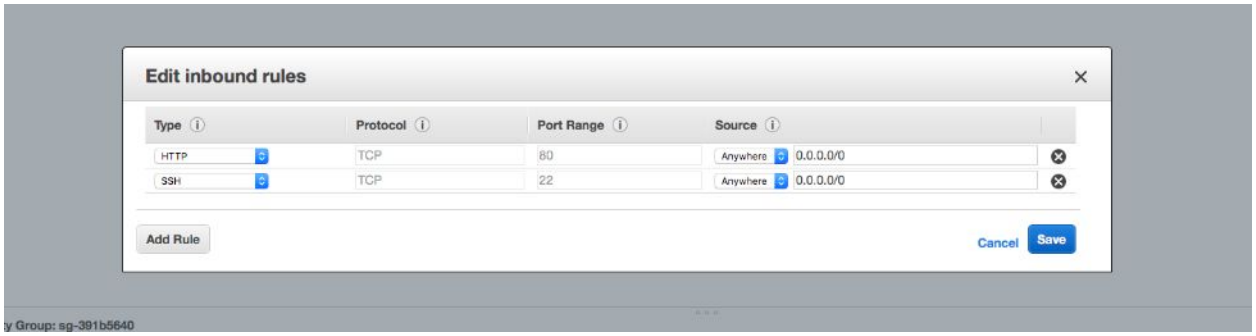
1. Login to your console and create a new EC2 instance of ubuntu (Here we are using ubuntu you can choose operating system of your choice)
2. Select the configuration and click on “Review and Launch”
3. You will be prompted to associate a ssh key pair with the instance, you can use an existing key pair or create a new one. The key will be downloaded onto the computer which will be later used to ssh into the machine.
4. Once the instance is running select your instance and scroll right for security groups.



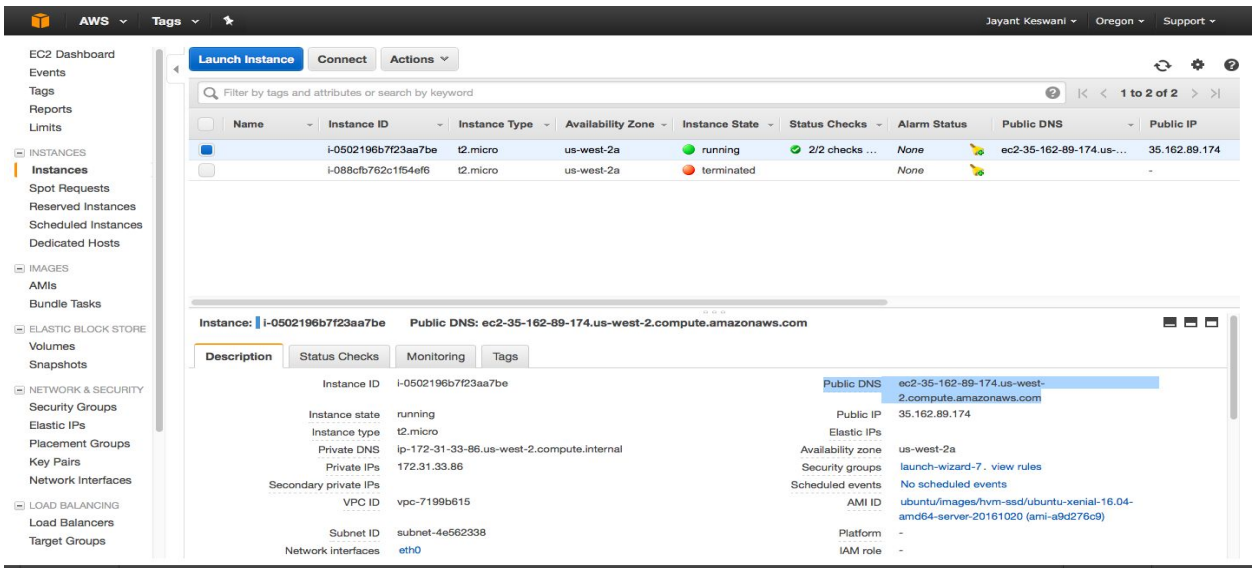
5. From “Actions” button select “Edit inbound rules”



6. Add a new http rule for port 6080 to access the GUI from the container



7. Copy the public dns of the instance



8. SSH into the instance by typing the following command into the terminal.
   (Type the commands in the directory where the ssh key of AWS instance was downloaded)
   ```
   # chmod 400 demo.pen (demo.pen is name of the key)
   # ssh -i demo.pen ubuntu@public-dns-of-aws-instance
   ```

9. After you are logged in type these commands to install docker on the instance
   ```
   # sudo apt-get update
   # sudo apt-get install docker.io
   # sudo docker run -it -p 6080:6080 -v ${PWD}:/local
   guidock-lite-orange
   ```

10. Use your favorite browser and go to the address to access the container
    ```
    public-dns-of-aws-instance:6080
    public-ip-of aws-instance:6080
    ```
11. Select the cytoscape application from the start menu

Firewall Configurations

Please refer to the following documentations for more information:

- Google Cloud Platform: https://cloud.google.com/compute/docs/networking
- Amazon Web Services: http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/authorizing-access-to-an-instance.html
- Microsoft Azure: https://azure.microsoft.com/en-us/documentation/articles/virtual-machines-windows-classic-setup-endpoints/

On Distributed Docker Platform

GUIdock-lite-orange can also be run on a distributed Docker Platform (e.g. Google Container, Docker Swarm, or Amazon EC2 Container Service). GUIdock-lite-orange is not designed for distributed system, hence, running multiple container does not increase the container's performance --instead each single containers serve as an independent container. Accessing the container can be performed using the container's public IP address from a web browser.

Demo

Here is the link to the -- demo