SUPPORTING INFORMATION

A language to simplify computation of differential mobility analyzer response functions

Markus D. Petters

NC State University, Department of Marine Earth and Atmospheric Sciences, Raleigh, NC 27695-8208.

ARTICLE HISTORY
Compiled September 22, 2018

1. Software Components

The software infrastructure supporting this work is composed of several independent open-source projects. Julia is the core programming language and available via the permissive open source MIT license. The Julia DMA language also makes use of a number of external packages that augment Julia, e.g. the PlotlyJS package which implements the Javascript graphics. Jupyter is a platform to display and execute the notebooks. It is available under the permissive open source BSD-3 license. Jupyter can be also used with Python and R code and serves as one of several available interfaces with Julia. The Julia DMA code is available as supplement and also hosted on github.com, which is an online collaborative software development platform now owned and operated by Microsoft. Github is one of several platforms that work with the git distributed revision control system software. Git is free software licensed under the GNU General Public License version 2. Although github.com is a commercial platform, it has traditionally hosted open source projects free of charge. This may change in the future, at which point alternative hosts might be considered. Users can obtain the current repository through github.com anonymously. If they make changes to the code they may upload these changes which are linked to their personal account. They can also propose that the changes are merged into the "master branch" of the code via a pull request, maintain a branched copy of the code, or fork the code. The contributions to the code by each user are clear. The revisions also have a public forum where the changes can be discussed and iteration can take place. The process is visible and traceable by the public via the github.com interface.

The JDL code is licensed under the GNU General Public License version 3. All use,

including commercial applications are encouraged. However, the license expressly forbids application of the software in locked-down devices under conditions where users are unaware that the device is using JDL code and where users do not have the ability to access and/or modify the source code, even if the code was modified by the creator of such a device.

Note that the substantive part of the Julia DMA code only requires the core Julia programming language. However, the visualizations, Jupyter integration, and some of the more complex examples require the presence of the entire software stack described above. Installation of all components can be challenging. Due to the fast pace of open source software development, this software stack can evolve resulting in unanticipated "breakage" of one or more components. For example, an update in the external package package WebIO, which handles communication between Julia and Web-based technologies such as Jupyter, can break functionality of plotting in the notebooks. Worse, such breakage will lead to error messages that may falsely suggest that the entire code is broken. These can be difficult to diagnose for users just starting to learn Julia, or users trying to execute the original software in future years when the underlying web-technologies likely will have changed substantially. This issue is addressed in two ways. First, a virtual machine that contains a snapshot of the entire software stack is permanently archived on zenodo.org (doi:10.5281/zenodo.1432522). The virtual machine and software should, in principle, work indefinitely. Second, the Julia DMA software will be updated by the author to evolve with the existing technologies.

2. Virtual Machine Setup

The machine consists of a virtual hard drive in .vdi format readable using the Oracle VM VirtualBox virtualization software. To run the machine a working installation of the open-source VirtualBox software is required. The exact configuration of the machine may vary with available resources. The configuration shown in Figures 1-3 has been tested on the authors' machine.

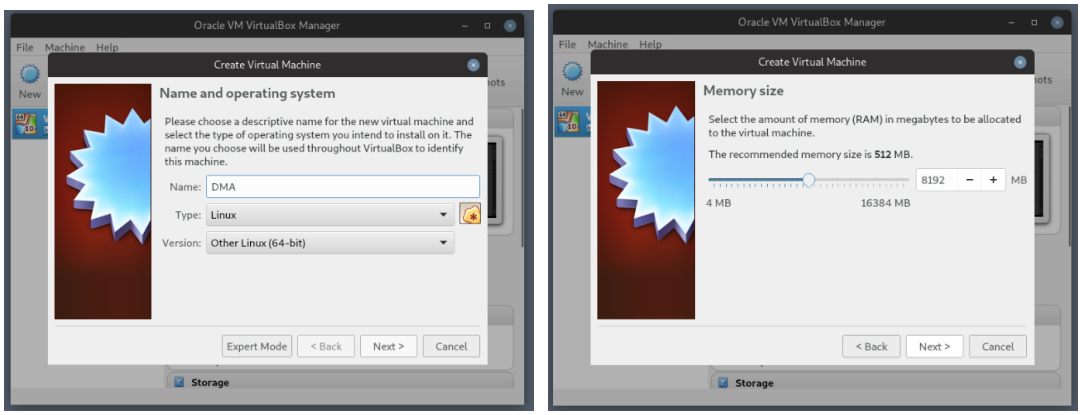

Figure 1.: VirtualBox Setup - Left, step 1: Basic Setup. Right, step 2: Allocate memory.

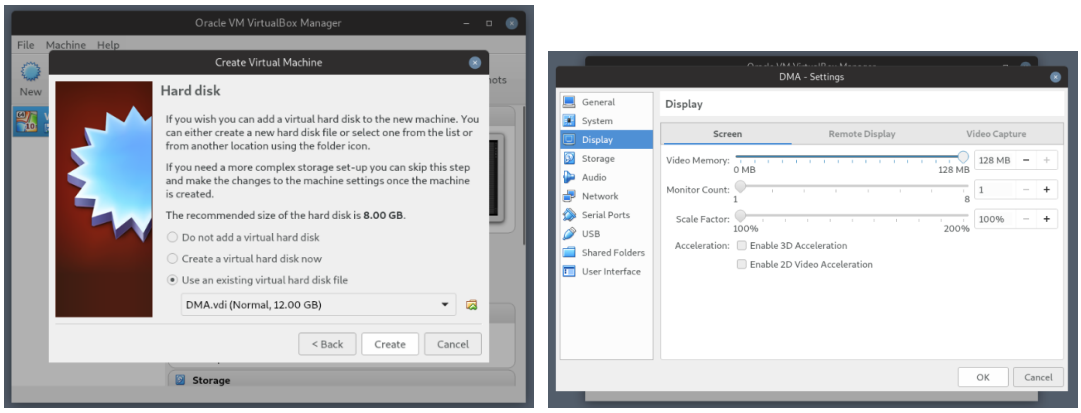

Figure 2.: VirtualBox Setup - Left, step 3: Add downloaded virtual hard disk. Right, step 4: Allocate video memory.

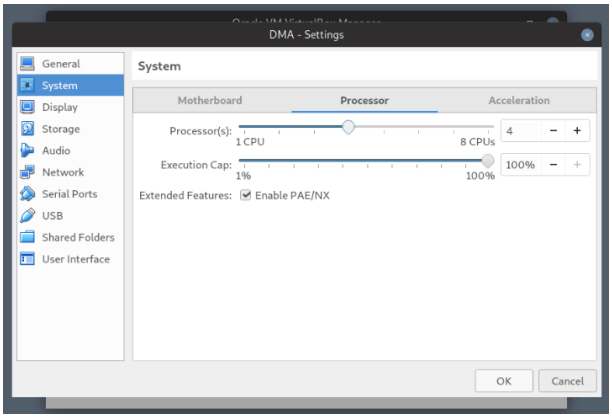

Figure 3.: VirtualBox Setup - Step 5: allocate number of CPUs.

3. Virtual Machine Setup

The virtual machine is running Debian GNU/Linux v. 9.5 as operating system and uses the XFCE window manager. These were selected based on software stability/maturity and to minimize software overhead associated with the virtual machine itself (e.g. requiring emulation of 3D graphics). The access credentials to the virtual machine are

```
user: dmauser
password: dmauser
root password: DMA
```

After logging on, start the Terminal Emulator located on the Desktop. In the terminal start the julia interpreter

```
dmauser@DMA:~/Desktop/DifferentialMobilityAnalyzers/docs$ julia
```

Then start the notebook server using the "using IJulia" and "notebook(detached = true)" commands.

```
   _       _ _(_)_     |  A fresh approach to technical computing
  (_)     | (_) (_)    |  Documentation: https://docs.julialang.org
   _ _   _| |_  __ _   |  Type "?help" for help.
  | | | | | | |/ _` |  |
  | | |_| | | | (_| |  | Version 0.6.4 (2018-07-09 19:09 UTC)
 _/ |\__'_|_|_|\__'_|  |  Official http://julialang.org/ release
|__/                   |  x86_64-pc-linux-gnu

julia> using IJulia

julia> notebook(detached = true)
```

Finally, navigate to the folder "Desktop/DifferentialMobilityAnalyzers/docs" and load any of the notebooks. All notebooks have been executed once on the virtual machine and are therefore "trusted", meaning that the javascript graphics display. Note, however, that the notebook must be executed sequentially for the first time after loading, as the computational state of the notebook is not available to the freshly started interpreter. Figure 4 shows a screenshot of virtual machine after following all the steps above. Note that the virtual machine has PartMC installed. Original archives of source codes for the julia language, PartMC, and this work are in the directory "Desktop/Source Codes". External julia packages are located in the directly "/home/dmauser/.julia/v0.6/".

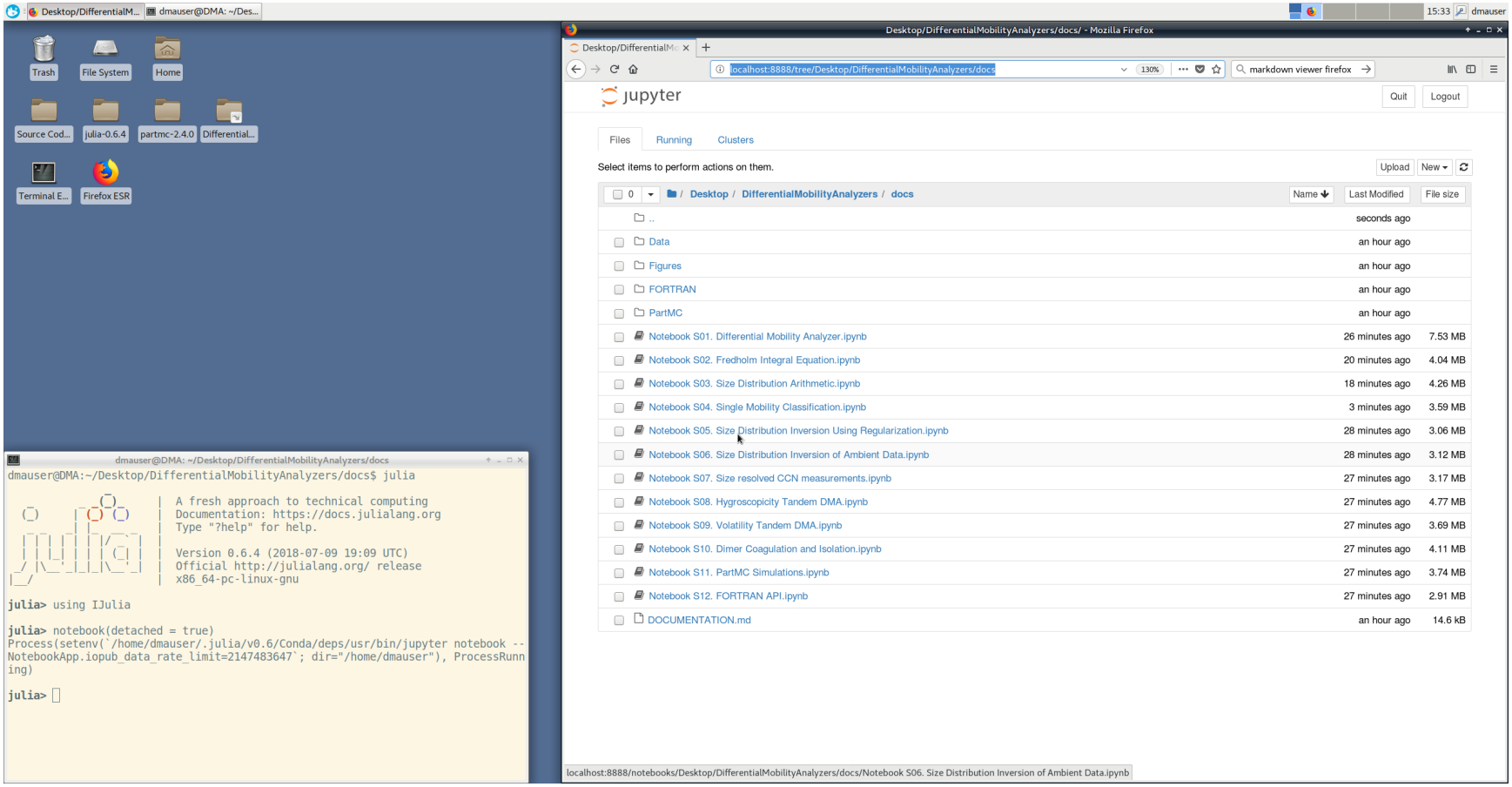

Figure 4.: Screenshot of virtual machine with julia and notebook server running.