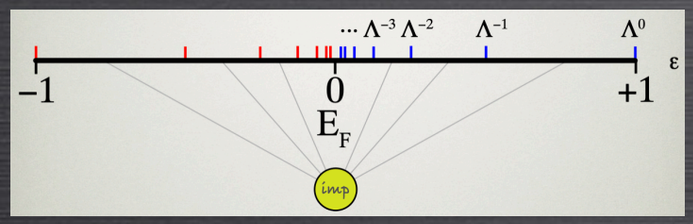
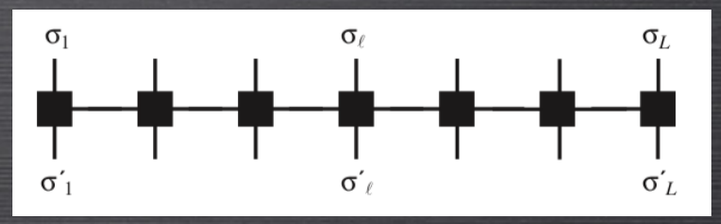
Solvers for quantum impurity problems (with superconducting baths)

Tutorial 1: getting started





Rok Žitko

Jožef Stefan institute, Ljubljana, Slovenia

Faculty of Mathematics and Physics, University of Ljubljana, Slovenia

University of Copenhagen, Oct 2021

Precompiled codes

Singularity containers (OS-level virtualization)

http://f1web.ijs.si/~zitko/containers/singularity_for_CPH/

foss-2020a.sif container (CentOS 7.8 + all libraries)
nrgljubljana/ two versions of NRG code
tensor/ DMRG code for Richardson model
ctqmc/ CT-HYB QMC code

Containers

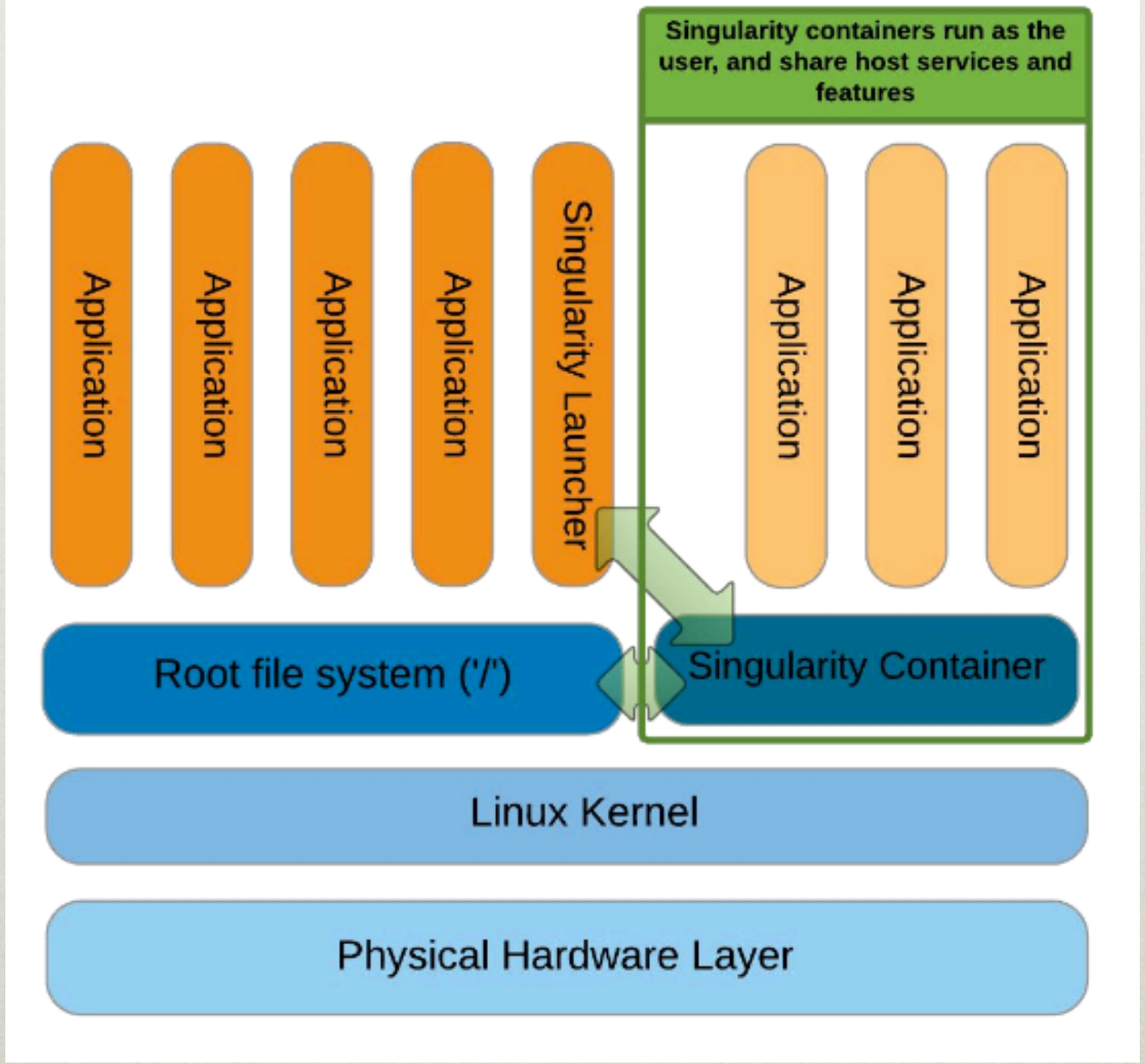
- = operating-system-level virtualisation
- Mechanism for encapsulating software with **all** dependencies (entire operating system, except the kernel): user can "swap out" the entire OS
- Applications **fully independent** of the underlying infrastructure
- Containers looks like a real computer from the point of view of programs running inside them

Differences with respect to true virtualization

- Virtualisation emulates physical hardware, runs a full unmodified operating system.
  Low-overhead on modern CPUs, but start-up time significant.
  More flexible (Windows VM on Linux host.)
  Examples: vmware, parallels, virtualbox.
- Software level virtualisation, no hardware and kernel emulation.
  Zero overhead, almost zero start-up time.
  Less flexible (Linux on Linux).

Singularity

- Solution for running **containers** in **high-performance computing** (HPC) environment
- Developed by Lawrence Berkeley Laboratory (LBL), since 2015.
- Free, cross-platform, open-source
- Available on many large HPC installations


Singularity containers run as the user, and share host services and features
Application
Application
Application
Application
Application
Application
Application
Application
Singularity Launcher
Root file system ('/')
Singularity Container
Linux Kernel
Physical Hardware Layer

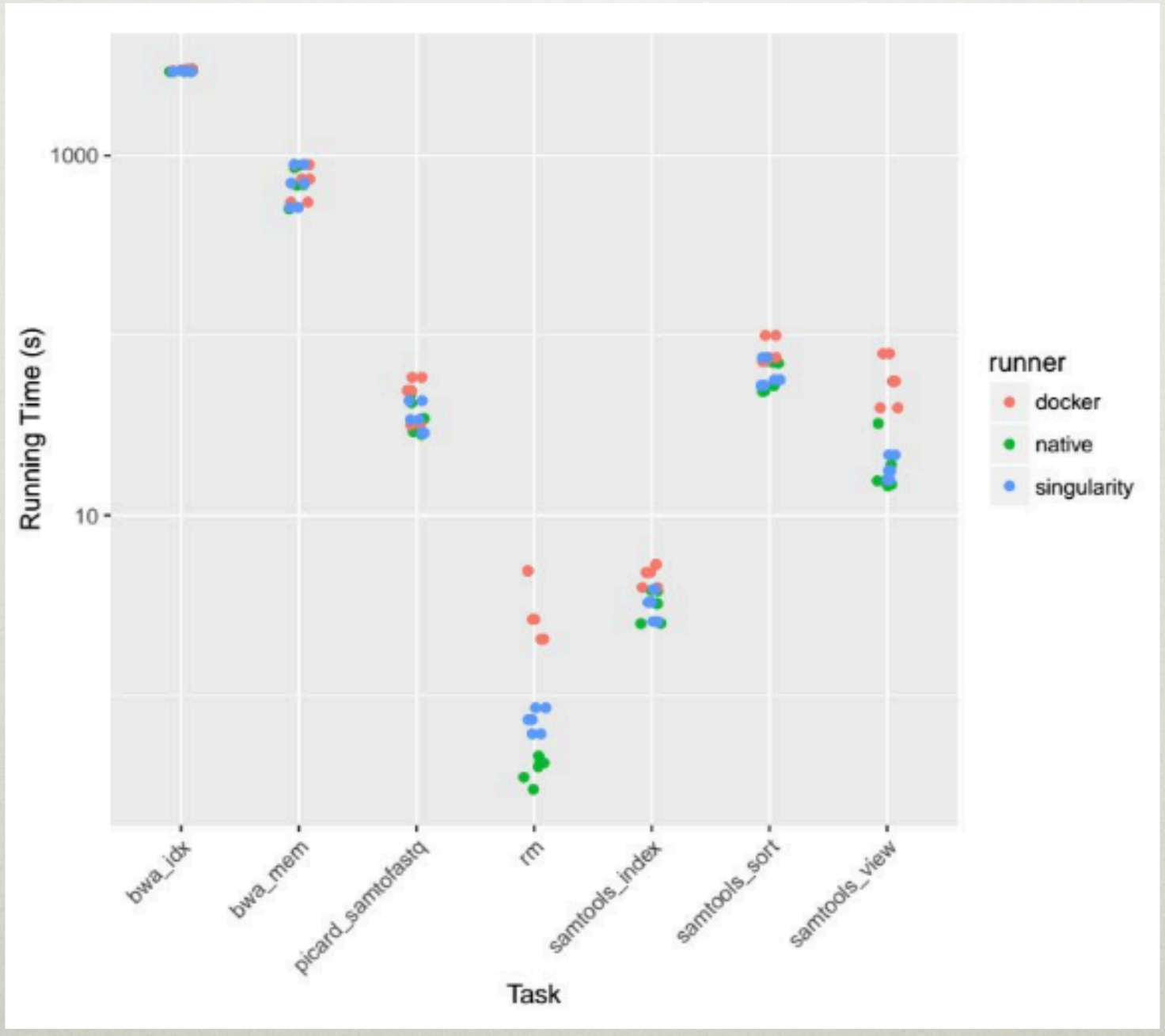
User management

- Same user inside the container as on the host.
- Home on the host maps on home in the container.

Filesystem access

- Access to host filesystem controlled by a simple config file.
- Home directory mapped by default.

Overhead?


1000
10
Running Time (s)
runner
docker
native
singularity
bwa_idx
bwa_mem
picard_samtofastq
rm
samtools_index
samtools_sort
samtools_view
Task

Workflow

- Build image
  `sudo singularity build [image] [definition]`
- Run image
  `singularity shell [image]`
  `singularity exec [image] [command]`
  `singularity run [image]`
  `./image`

Reproducibility

- Fully encapsulated computer codes, with all dependencies (even the OS itself!).
- Single image file to be archived.
- Container image files are not modified during their use. [but see next slide!]
- Can be digitally signed.

Contact me on rok.zitko@ijs.si in case of trouble.

curl & wget
(download tools)

curl -O [URL]

or

wget [URL]

Installing curl (but note that you can use wget instead!)

On ubuntu:
sudo apt update
sudo apt install curl

Approach:

- Obtain Linux with a running Singularity installation: either install Singularity from a package (directly on RedHat, via "alien" tool on Ubuntu), compile Singularity yourself, or get a VM with Ubuntu+Singularity using vagrant
- Obtain container with CentOS + numerical libraries
- Obtain the codes

Install Singularity (RedHat)

sudo yum install singularity

install Singularity (using RedHat packages on Ubuntu)

sudo apt-get install alien

Luckily for those not wanting to compile from source, most of the binaries in the (well maintained) EPEL EL8 singularity 3.8.x rpm are static linked, so it is one of the cleaner conversions using alien:

```
$ grep PRETTY_NAME /etc/os-release
PRETTY_NAME="Ubuntu 20.04.2 LTS"
$ alien --version
alien version 8.95
# WARNING:  singularity version numbers subject to change, browse the yum
repo to get up-to-date URL
wget https://dl.fedoraproject.org/pub/epel/8/Everything/x86_64/Packages/s/
singularity-3.8.0-1.el8.x86_64.rpm
$ sudo alien -d singularity-3.8.0-1.el8.x86_64.rpm
...
$ sudo dpkg -i ./singularity_3.8.0-2_amd64.deb
...
$ singularity --version
singularity version 3.8.0-1.el8
$
```

installing Vagrant (on ubuntu)

https://linuxize.com/post/how-to-install-vagrant-on-ubuntu-20-04/

```
sudo apt update
sudo apt install virtualbox
curl -O https://releases.hashicorp.com/vagrant/2.2.9/vagrant_2.2.9_x86_64.deb
sudo apt install ./vagrant_2.2.9_x86_64.deb
vagrant --version
```

INSTALLING VAGRANT (WINDOWS)
INSTALLATION GUIDE ON SINGULARITY WEB PAGE PAGE

https://sylabs.io/guides/3.8/admin-guide/installation.html#installation-on-linux

Scroll down to the following section and install all the programs:

Windows

Install the following programs:

- Git for Windows
- VirtualBox for Windows
- Vagrant for Windows
- Vagrant Manager for Windows

Sometimes one gets the following output when running the container:

UID: readonly variable
[...]

In some cases the container still works.

To be fixed!!!

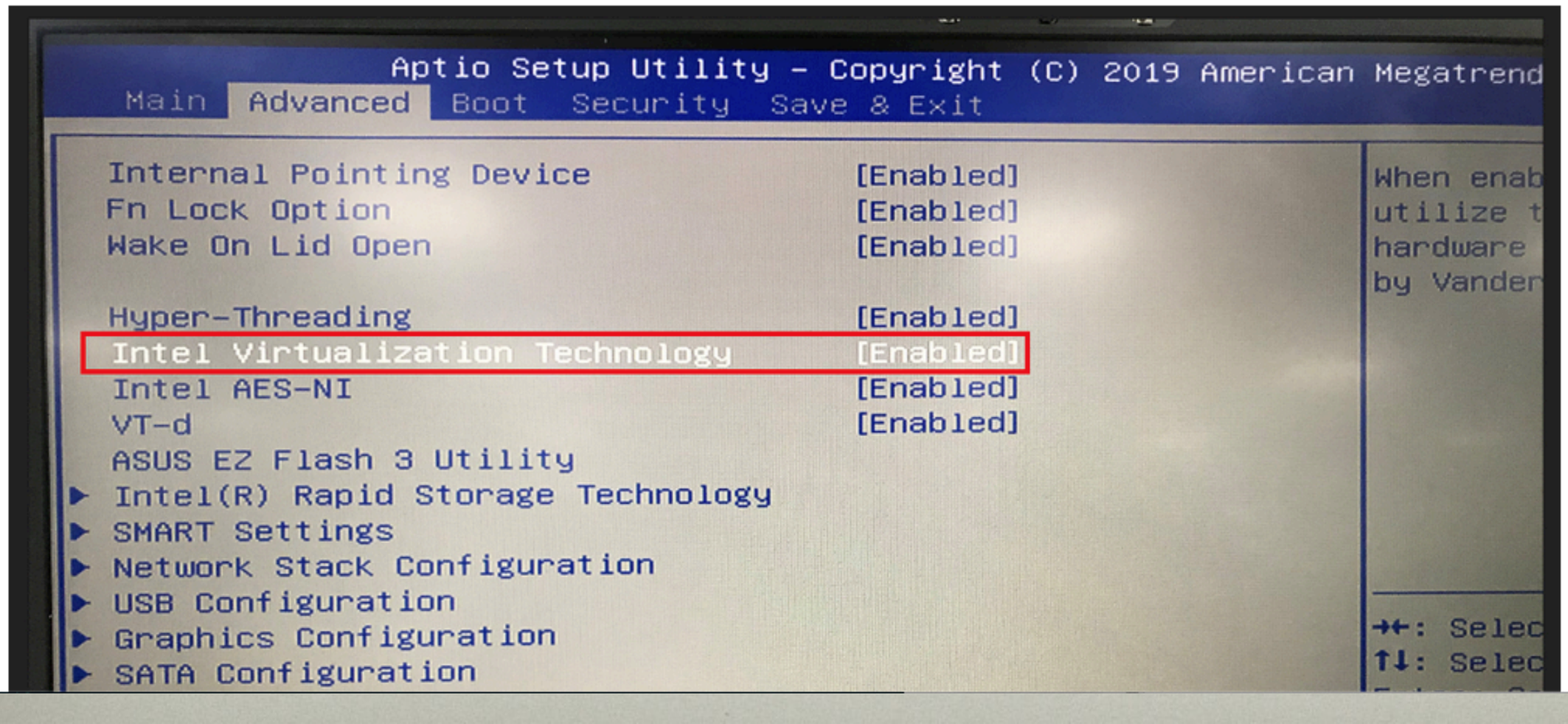
Virtualization support

If you get error messages about VT-x being disabled, do the following:

[Notebook] How to check if my laptop's processor(CPU) supports Intel® Virtualization Technology(VT-x)?

How to enable or disable Intel® Virtualization Technology (VT-x)?

With a VT-x supported Intel® Processor and a VT-x supported BIOS, VT-x can be enabled or disabled in BIOS. (The default of this item is **Enabled** in BIOS)



Vagrant (MacOS)

Install brew:

https:/ /brew.sh/



brew install vagrant

Virtual machine with Ubuntu & Singularity

```
vagrant init sylabs/singularity-ce-3.8-ubuntu-bionic64
vagrant up
vagrant ssh
```

At this point you will be inside the virtual machine with working singularity.

```
singularity --version
singularity-ce version 3.8.0
```

Green text is the expected output!

QUICK START ON LINUX

Once you have singularity working, proceed with these steps:

```
mkdir course
cd course

wget http://f1web.ijs.si/~zitko/containers/
singularity_for_CPH/foss-2020a.sif

chmod a+x foss-2020a.sif

./foss-2020a.sif cat /etc/redhat-release
CentOS Linux release 7.8.2003 (Core)
```

Green text is the expected output!

```
curl -O http://f1web.ijs.si/~zitko/
containers/singularity_for_CPH/
nrgljubljana.tar.gz

tar zxf nrgljubljana.tar.gz
```

```
curl -O http://f1web.ijs.si/~zitko/
containers/singularity_for_CPH/
nrg_quick_test.tar.gz

tar zxf nrg_quick_test.tar.gz
```

```
./foss-2020a.sif bash
```

Then inside the container:

```
. nrgljubljana/2020.09/vars.hs
cd nrg_quick_example/
nrg
```

```
[zitko@spinon course]$ ./foss-2020a.sif bash
Singularity> . nrgljubljana/2020.09/vars.hs
NRG ROOT: /home/zitko/course/nrgljubljana/2020.09
Singularity> cd nrg_quick_test/
Singularity> nrg
Parallelization using MPI: Running on 1 processors.

NRG Ljubljana - (c) rok.zitko@ijs.si
Timestamp: Fri Sep 25 10:43:52 2020
Compiled on Sep 25 2020 at 10:52:08

[OpenMP] Max. number of threads: 1
[OpenMP] Number of processors: 32
[OpenMP] Dynamic thread adjustment: 0
[OpenMP] Nested parallelism: 0

Using Intel MKL library 2020.0.1
Processor optimization: Intel(R) Advanced Vector Extensions 2 (Intel(R) AVX2) enabled processors
max_threads=1 blas_max_threads=1 dynamic=1


workdir=/work/zitko//noio03
```

Note 1: notice the space after the dot in ". nrgljubljana/2020.09/vars.hs".

Note 2: "vars.hs" is a typo that I plan to fix (so it might be "vars.sh" by now that you are reading this).

```
 ***** [NRG] Iteration 39/39 (scale 0.000001) *****
[1] xi(38)=1.3758612068999665e-06 xi_scaled(38)=0.9999999999818112 zeta(39)=0
Stats: nr=29 min=1 max=139
Safeguard: keep additional 2 states
Emax=8.005383940866716 nrkept=269 nrkeptmult=626
Total energy=-2.8920226655456518  Egs=-0.814855799238554
Kept: 269 out of 982, ratio=0.274
Memory used: 317 MB Time elapsed: 1.308 s

** Iteration completed.



Total energy: -2.8920226655456518
TKW=1.5087767171083757e-05
Master done. Terminating slave processes.
Master exiting.

Peak usage: 317 MB

Timing report [0]
=============
         All: 1.310 s
        diag: 0.827 s
    recalc f: 0.024 s
       Other: 0.459 s
Singularity>
```

As of 6. 10. 2021

Index of /~zitko/containers/singularity_for_CPH

Name	Last modified	Size	Description
Parent Directory		-	
1_build	11-Sep-2020 09:33	86	
2_shell	21-Feb-2020 08:30	42	
3_mk_foss2020a	13-Sep-2020 15:26	81	
NOTES	14-Sep-2020 10:04	63	
Singularity.foss-2020a	17-Sep-2021 12:25	2.9K	
centos7.7-Python3	13-Sep-2020 15:25	896	
centos7.7.sif	13-Sep-2020 15:32	671M	
ctqmc.tar.gz	06-Oct-2021 19:00	258K	
foss-2020a.sif	17-Sep-2021 19:44	2.8G	
nrg_quick_test.tar.gz	06-Oct-2021 19:00	10K	
nrgljubljana.tar.gz	06-Oct-2021 19:00	14M	
tensor.tar.gz	06-Oct-2021 19:00	2.7M	

As of 6. 10. 2021

Index of /~zitko/courses/2021_CPH

Name	Last modified	Size	Description
Parent Directory		-	
2021_solvers_L1.pdf	05-Oct-2021 18:41	14M	
2021_solvers_L2.pdf	05-Oct-2021 18:41	12M	
2021_solvers_T1.pdf	05-Oct-2021 18:41	2.5M	
2021_solvers_T2.pdf	05-Oct-2021 18:41	3.9M	
2021_solvers_T3.pdf	05-Oct-2021 18:41	5.0M	
Vagrant.README	06-Oct-2021 19:16	562	
download	06-Oct-2021 19:04	546	
tutorial_2021_v1.tar.gz	06-Oct-2021 18:24	3.1M	

Getting "nrginit" to work

- Does Mathematica run from command line? (Type "math" to test)
- Copy "nrginit" and *.m files somewhere in your $PATH (e.g. create $HOME/bin, put files there, add $HOME/bin to your $PATH).

Linux: do this on the host.
Windows, MacOS: do this on the host or inside the Linux VM.