

# Running Applications on the Hoffman2 Cluster: Case Studies

Doing research on the Hoffman2 cluster

Raffaella D'Auria, PhD

## Learning outcomes:

- ▷ run a batch job
- ▷ Serial, multi-threaded, distributed jobs
- ▷ create matlab stand-alone executables
- ▷ running matlab in batch
- ▷ running abaqus python scripts
- ▷ array jobs
- ▷ array jobs w/ abaqus
- ▷ array jobs w/ R
- ▷ can my workflow be run via an array job
- ▷ jupiter notebooks



# Applications already available on Hoffman2: Python

## Correction to Intro slide no. 54!

```
[rdtest@n9860 ~]$ ls -t $HOME/.local/bin
run_miso.py  compare_miso  summarize_miso  exon_utils  sam_to_bam      miso
filter_events  miso_zip       plot.py        pe_utils    index_gff      f2py
test_miso     run_events_analysis.py  sashimi_plot  miso_pack   module_availability
[rdtest@n9860 ~]$ which miso
miso: Command not found.

[rdtest@login3 ~]$ echo "export PATH=$HOME/.local/bin:$PATH" >> $HOME/.bash_profile
[rdtest@login3 ~]$ tail -n1 $HOME/.bash_profile
export PATH=/u/home/r/rdtest/.local/bin
[rdtest@login3 ~]$ . /u/home/r/rdtest/.bash_profile
[rdtest@login3 ~]$ which miso
~/./local/bin/miso
```

# Running a batch job

Why?

# Running a batch job

The screenshot shows a web browser window with the following details:

- Address Bar:** https://www.hoffman2.idre.ucla.edu
- Toolbar:** Includes back, forward, search, and other standard browser icons.
- Header:** The page title is "Running a batch job". The header also includes the UCLA IDRE logo and a "Report Typos and Errors" link.
- Content Area:**
  - UCLA Hoffman2 Cluster User Guide:** A sidebar on the left contains links to "About", "User Support", "FAQ", "Getting started", "News", "Security policy", "Status", "How to acknowledge the Hoffman2 cluster program", and "How to use this site".
  - Search:** A "Search this website" input field.
  - Navigation Tabs:** A horizontal menu with tabs: ACCESS, COMPUTING (circled in green), DATA STORAGE, FILE TRANSFER, and SOFTWARE.
  - Main Content:** The "Home" page is displayed. It includes a breadcrumb trail ("You are here: Home"), a main heading "Home", a welcome message, and sections for "Recent Announcements" (mentioning Free Classes and Stata 15) and "Free Classes".

# Running a batch job

The screenshot shows a web browser displaying the 'Computing' page of the idre Hoffman2 Cluster User Guide. The URL in the address bar is <https://www.hoffman2.idre.ucla.edu/computing/>. The page has a dark blue header with the idre UCLA logo and navigation links for ACCESS, COMPUTING, DATA STORAGE, FILE TRANSFER, and SOFTWARE. A search bar is also present. The main content area shows the 'Computing' page with a sidebar containing links like About, User Support, FAQ, Getting started, News, Security policy, Status, How to acknowledge the Hoffman2 cluster program, and How to use this site. The main content area displays the 'Computing' page with sections for General information, Interactive jobs, and Batch Jobs. The 'Batch Jobs' section is highlighted with a green circle. The 'Interactive jobs' section is also highlighted with a green circle.

https://www.hoffman2.idre.ucla.edu/computing/

120% C ☆ Search

Report Typos and Errors

idre UCLA

Hoffman2 Cluster User Guide

Search this website

ACCESS COMPUTING DATA STORAGE FILE TRANSFER SOFTWARE

About

User Support

FAQ

Getting started

News

Security policy

Status

How to acknowledge the Hoffman2 cluster program

How to use this site

You are here: Home / Computing

## Computing

Contents [hide]

General information

Interactive jobs

Batch Jobs

GPU Computing

Additional Information

How to ...

### General information

- Job Queues and Policy

### Interactive jobs

- How to Get an Interactive Session

### Batch Jobs

- Submitting a Job
- Commonly-Used UGE Commands

Apps on H2: Case Studies

R. D'Auria, PhD

# Running a batch job

```
[rdtest@login3 ~]$ cp /u/local/apps/submit_scripts/submit_job.sh ./
[rdtest@login3 ~]$ ls -lat submit_job.sh
-rwxr--r-- 1 rdtest systems 575 Oct 11 14:05 submit_job.sh
[rdtest@login3 ~]$ █
```

If `submit_job.sh` had not been executable we should have issued:

```
[rdtest@login3 ~]$ chmod u+x submit_job.sh
```

To submit the job:

```
[rdtest@login3 ~]$ qsub submit_job.sh
Your job 865688 ("submit_job.sh") has been submitted
[rdtest@login3 ~]$ █
```

# Checking the status of a job in the queue

```
[rdtest@login3 ~]$ myjob
job-ID prior name user state submit/start at queue slots ja-task-ID
-----
865688 10.000000 submit_job rdtest r 10/11/2017 14:24:34 c2_smp.q@n9860 1
[rdtest@login3 ~]$
```

If `submit_job.sh` had been pending its “state” would have been “qw”

```
[rdtest@login3 ~]$ qstat -s p | head -n 6
job-ID prior name user state submit/start at queue slots ja-task-ID
-----
861033 5.75286 mayo_cruce caroartc qw 10/10/2017 22:59:11 4
862844 2.87643 QRLOGIN caroartc qw 10/11/2017 07:20:43 4
121156 1.91801 roms_TEST_ hfrenzel qw 07/14/2017 11:24:01 96
740935 1.91771 in.ThreeD_ tjones19 qw 09/27/2017 12:37:24 24
```

# Anatomy of a batch job script

Shell that will be used

```
#!/bin/bash  
#$ -cwd  
# error = Merged with joblog  
#$ -o joblog.$JOB_ID  
#$ -j y  
#$ -l h_rt=1:00:00,h_data=1G  
# Email address to notify  
#$ -M $USER@mail  
# Notify when  
#$ -m bea
```

Submission script  
preamble

#\$ scheduler commands

```
# echo job info on joblog:  
echo "Job $JOB_ID started on: " `hostname -s`  
echo "Job $JOB_ID started on: " `date`  
echo "
```

```
# load the job environment:  
. /u/local/Modules/default/init/modules.sh  
module load <APP>  
module li  
echo "
```

```
# run <APP>  
echo '<execution command>'  
<execution command>
```

# comments

```
# echo job info on joblog:  
echo "Job $JOB_ID ended on: " `hostname -s`  
echo "Job $JOB_ID ended on: " `date`  
echo "
```

# Anatomy of a batch job script

```
[rdtest@login3 ~]$ myjob
```

You do not have any active SGE jobs currently.

```
[rdtest@login3 ~]$ cat joblog.865688
```

```
Job 865688 started on: n9860
```

```
Job 865688 started on: Wed Oct 11 14:24:35 PDT 2017
```

The 'gcc/4.9.3' module is being loaded

```
pwd; sleep 500
```

```
/u/home/r/rdtest
```

```
Job 865688 ended on: n9860
```

```
Job 865688 ended on: Wed Oct 11 14:32:55 PDT 2017
```

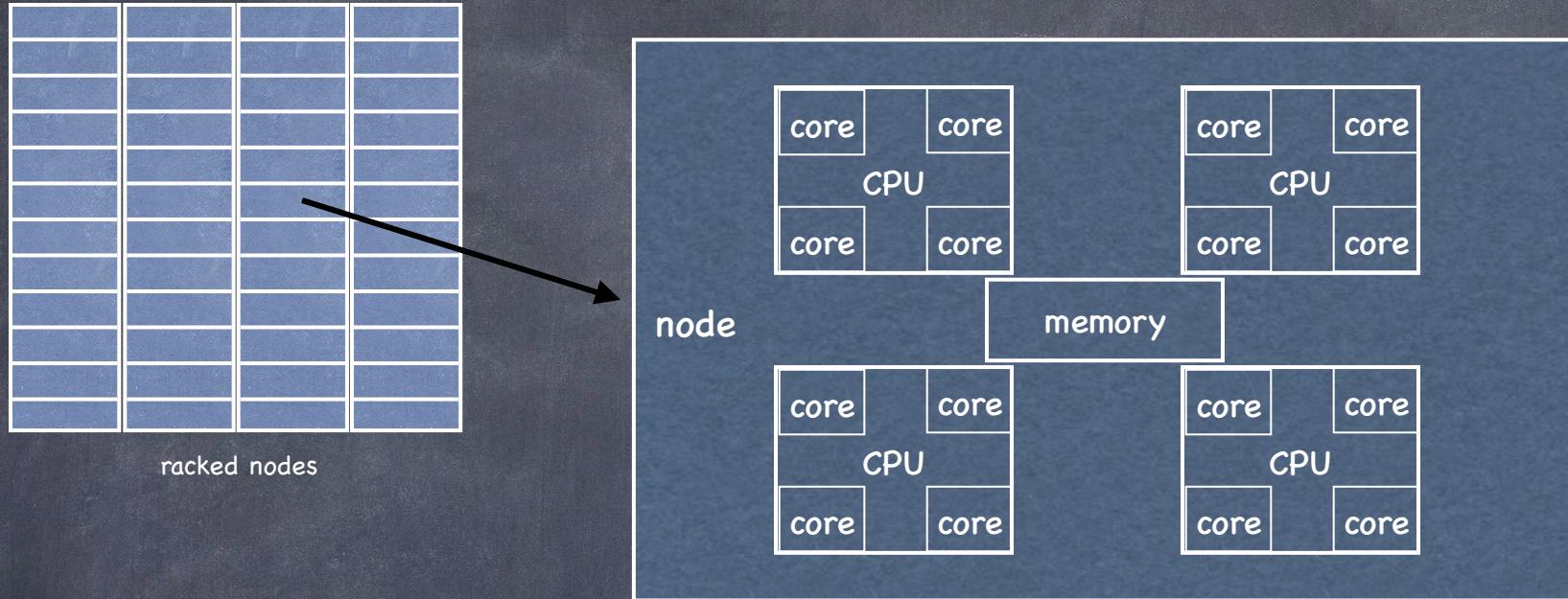
```
[rdtest@login3 ~]$
```

# Running a batch job

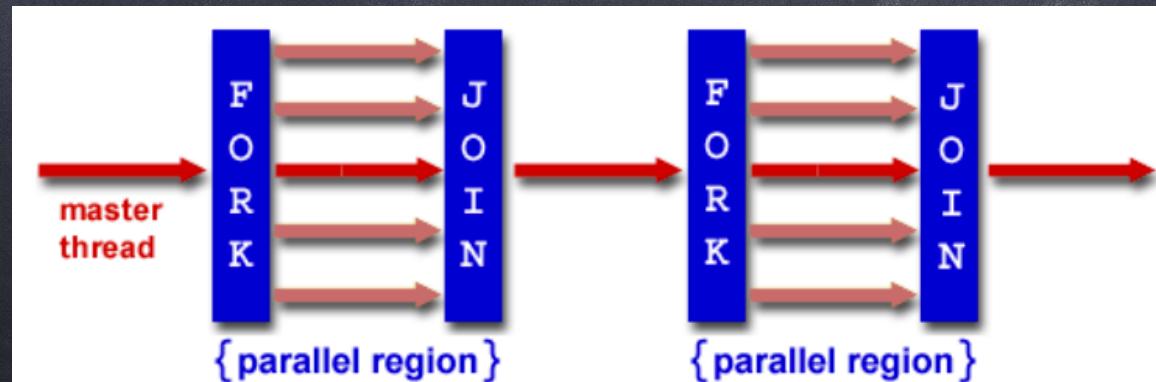
You can also use our own queue scripts, such as:

- job.q (serial jobs)
- openmp.q (multiple threads/shared memory jobs)
- intelmpi.q (distributed memory jobs w/ intelmpi)
- openmpi.q (distributed memory jobs w/ openmpi)
- parallel.q (distributed memory jobs DIY)
- <APP>.q:
  - stata.q. (allows serial/multiple threads)
  - R.q (serial only version of R set)
  - etc.
- jobarray.q. (embarrassingly parallel jobs)

# Shared Memory jobs



- Use more than one core (slot in queue jargon)
- All cores are from the same node (server/computer)
- The program spawns threads each of which runs on a separate core



# Shared Memory jobs

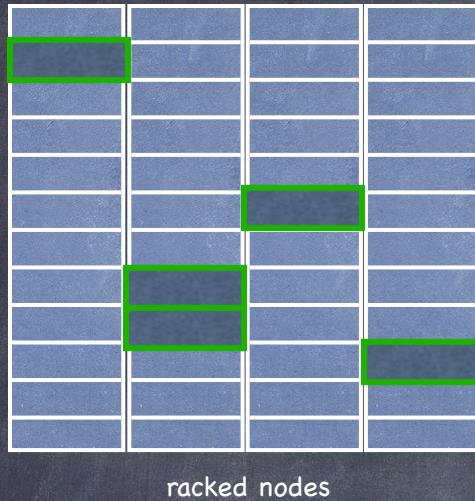
You may be running in shared memory jobs b/c:

- You link to multi-threaded libraries (MKL, blas, lapak, etc.)
- You have explicitly included threads/openmp chunks of code
- You are running an app, developed by a third party, which spawns threads
  - You should accordingly request multiple cores
    - Either with openmp.q
    - Or generalize your submission script preamble:

```
#!/bin/bash
#$ -cwd
# error = Merged with joblog
#$ -o joblog.$JOB_ID
#$ -j y
#$ -l h_rt=1:00:00,h_data=1G
#$ -pe shared 6
# Email address to notify
#$ -M $USER@mail
# Notify when
#$ -m bea
```

```
#!/bin/bash
#$ -cwd
# error = Merged with joblog
#$ -o joblog.$JOB_ID
#$ -j y
#$ -l h_rt=1:00:00,h_data=1G,exclusive
# Email address to notify
#$ -M $USER@mail
# Notify when
#$ -m bea
```

# Distributed memory jobs



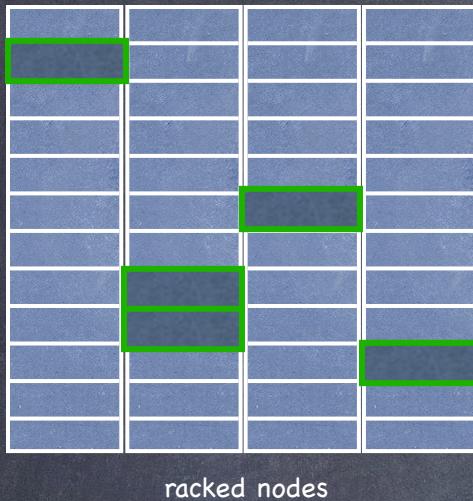
- Each parallel worker has its own private memory space
- All communications is done with message passing (MPI)
- Your code contains parallel instructions

As you need multiple cores possibly coming from multiple nodes

- Either use intelmpi.q/openmpi.q/parallel.q
- Or generalize your job script preamble

```
#!/bin/bash
#$ -cwd
# error = Merged with joblog
#$ -o joblog.$JOB_ID
#$ -j y
#$ -l h_rt=1:00:00,h_data=1G
#$ -pe dc* 24
# Email address to notify
#$ -M $USER@mail
# Notify when
#$ -m bea
```

# Hybrid distributed memory/shared memory jobs



- Message passing between the nodes
- Shared memory within each node
- Used by:
  - comsol.q (sorry no public licenses)
  - gaussian.q

As you need multiple cores possibly coming from multiple nodes  
→ generalize your job script preamble

```
#!/bin/bash
#$ -cwd
# error = Merged with joblog
#$ -o joblog.$JOB_ID
#$ -j y
#$ -l h_rt=1:00:00,h_data=1G,exclusive
#$ -pe node 3
# Email address to notify
#$ -M $USER@mail
# Notify when
#$ -m bea
```

# Matlab stand-alone executables

Matlab is a very hot commodity (limited licenses/highly coveted):

- Interactive work checks out a license for the entire time of the session
  - \* Please never forget a matlab desktop open
- Batch matlab jobs that run a matlab script also check out a license for the entire duration of the job
- Matlab scripts can be compiled with the matlab compiler!
  - Matlab stand-alone executables
  - \* Licenses are checked out only during compilation!
  - \* You can run as many copies of your stand-alone as you will: no license limit!

# Matlab stand-alone executables

The screenshot shows a web browser window with the URL <https://www.hoffman2.idre.ucla.edu>. The page title is "Matlab stand-alone executables". The header includes the UCLA and idre logos, a search bar, and navigation icons. A sidebar on the left lists various user support links. The main content area displays the "Home" page with sections for recent announcements and free classes.

Report Typos and Errors

UCLA  
Hoffman2 Cluster User Guide

Search this website

ACCESS COMPUTING DATA STORAGE FILE TRANSFER SOFTWARE

About [About](#)  
User Support  
FAQ  
Getting started  
News  
Security policy  
Status  
How to acknowledge the Hoffman2 cluster program  
How to use this site

You are here: [Home](#)

## Home

Welcome to the Hoffman2 Cluster User Guide. To look up usage information, click on one of the navigation tabs above or the sidebar items on the left.

### Recent Announcements

**Free Classes** IDRE Research Technology Group is pleased to present Fall Quarter classes. There is no charge or fee to attend these classes. See [IDRE Fall 2017 HPC Classes](#)

**Stata 15** A new version of Stata has been installed. See [Stata 15 available](#) for further information.

# Matlab stand-alone executables

https://www.hoffman2.idre.ucla.edu

matab

Report Typos and Errors

UCLA Hoffman2 Cluster User Guide

ACCESS COMPUTING DATA STORAGE FILE TRANSFER SOFTWARE

You are here: Home

## Home

Welcome to the Hoffman2 Cluster User Guide. To look up usage information, click on one of the navigation tabs above or the sidebar items on the left.

### Recent Announcements

**Free Classes** IDRE Research Technology Group is pleased to present Fall Quarter classes. There is no charge or fee to attend these classes. See [IDRE Fall 2017 HPC Classes](#)

**Stata 15** A new version of Stata has been installed. See [Stata 15 available](#) for further information.

About User Support FAQ Getting started News Security policy Status How to acknowledge the Hoffman2 cluster program How to use this site

# Matlab stand-alone executables

https://www.hoffman2.idre.ucla.edu/matlab\_toolboxes/ 120% Search Report Typos and Errors

**idre**

**UCLA**  
Hoffman2 Cluster User Guide

Search this website

ACCESS COMPUTING DATA STORAGE FILE TRANSFER SOFTWARE

You are here: Home / MATLAB

## MATLAB

Matlab is a multi-purpose numerical computing environment.

Contents [hide]

- General information
- Interactive use
  - Invoking matlab
  - Using the Distributed Computing Toolbox
    - Using resources local to one computational node
    - Using distributed resources across more than one computational node
    - Create a cluster configuration profile
    - Use the distributed cluster profile
  - Creating matlab standalone executables with the matlab compiler
  - Running matlab standalone executables
    - Running matlab standalone executables for matlab version 9.1 (R2016b)
- Batch use
  - How to run serial or multi-threaded MATLAB jobs using the Queue Scripts
  - How to run MATLAB jobs that contain parallel instructions
  - How to run MATLAB using UGE commands
- Some toolboxes
- Running Mapreduce using Matlab Parallel Pool

# How to generate matlab stand-alone executables

```
[rdtest@login1 ~]$ qrsh -l h_data=3g,h_rt=1:00:00
[rdtest@n9860 ~]$ module load matlab
[rdtest@n9860 ~]$ cat myInverseMatrix.m
% Compute the inverse of a N-by-N random matrix

X = rand(4)
Y = inv(X)
X*Y % check
[rdtest@n9860 ~]$ mcc -m myInverseMatrix.m
[rdtest@n9860 ~]$ ls -lt
total 124
-rwxr--r-- 1 rdtest systems 92207 Oct 11 18:06 myInverseMatrix
-rwxr--r-- 1 rdtest systems  883 Oct 11 18:06 run_myInverseMatrix.sh
-rw-r--r-- 1 rdtest systems  3801 Oct 11 18:06 readme.txt
-rw-r--r-- 1 rdtest systems    12 Oct 11 18:06 requiredMCRProducts.txt
-rw-r--r-- 1 rdtest systems 1277 Oct 11 18:06 mccExcludedFiles.log
-rw-r--r-- 1 rdtest systems   369 Oct 11 18:00 myInverseMatrix.m
[rdtest@n9860 ~]$ ./myInverseMatrix
```

# How to run matlab stand-alone executables

`matlab.q`

Runs single or multi-processor in two steps: compile and execute

`mcc.q`

Use the MATLAB compiler to create a stand-alone executable. If you are using `mcc.q` interactively, it will ask you if you want the executable produced by `mcc` to use a single processor or not. If you are using `mcc.q` in command line mode, to create an executable that will run on a single processor specify this argument:

`-R -singleCompThread`

`matexe.q`

Run a MATLAB stand-alone executable created with `mcc`

# How to generate matlab stand-alone executables w/ explicit parallel instructions

```
[rdtest@login1 ~]$ qrsh -l h_data=3g,h_rt=1:00:00
```

```
[rdtest@n9860 ~]$ module load matlab/9.1_MCR
```

```
[rdtest@n9860 ~]$ cat myparpoolexample.m
```

```
n=4;  
p = parpool('local',n);  
  
tic  
ticBytes(gcp);  
n = 1024;  
A = zeros(n);  
parfor (i = 1:n)  
    A(i,:) = (1:n) .* sin(i*2*pi/1024);  
end  
tocBytes(gcp)  
toc  
  
delete(p)
```

```
[rdtest@n9860 ~]$ mcc -m -R -singleCompThread myparpoolexample.m
```

```
[rdtest@n9860 ~]$ ./myparpoolexample
```

# How to run matlab stand-alone executables when parallel instructions are explicitly used

/u/local/apps/submit\_scripts/**matlab\_compile\_and\_submit.sh**

- Generates and submits a batch job that builds and runs a matlab standalone application out of one or more matlab functions
- Matlab standalone executables support the use of the Distributed Computing Toolox. The maximum number of parallel workers supported on hoffman2 is 16.
- If any part of your matlab code includes a parfor loop you will need to include the following lines:

%before the parfor loop, for example for 5 workers and the local profile:

```
p = parpool('local',5);
```

%after the parfor loop:

```
delete(p)
```

# How to run matlab stand-alone executables when parallel instructions are explicitly used

```
[rdtest@n9860 ~]$ cp /u/local/apps/submit_scripts/matlab_compile_and_submit.sh ./
[rdtest@n9860 ~]$ ls -l matlab_compile_and_submit.sh
-rwxr--r-- 1 rdtest systems 4783 Oct 11 18:54 matlab_compile_and_submit.sh
[rdtest@n9860 ~]$ ./matlab_compile_and_submit.sh
```

Usage:

```
./matlab_compile_and_submit.sh [-t time in hours]
[ -s number of processes ] [-m memory per process (in GB)]
[-p parallel environment: 1 for shared 2 for distributed]
[-f main matlab function] [-f matlab function 2] ... [-f matlab function n]
[-ns (to build a submission script without submitting the job)]
[ --help ]
```

```
[rdtest@n9860 ~/matlab]$ ./matlab_compile_and_submit.sh -t 1 -s 4 -m 2 -p 1 -f
myparpoolexample.m
```

Running myparpoolexample on 4 processes each with 2 GB of memory for 1 hour

```
[rdtest@n9860 ~/matlab]$ █
```

# Abaqus python scripts

```
[rdtest@login1 ~]$ qrsh -l h_data=3g,h_rt=1:00:00  
[rdtest@n9860 ~]$ module load abaqus  
[rdtest@n9860 ~]$ abaqus cae &
```

This will open the Abaqus desktop from which you can do pre/post processing.

If you already have your abaqus python script:

```
[rdtest@login1 ~]$ qrsh -l h_data=3g,h_rt=1:00:00  
[rdtest@n9860 ~]$ module load abaqus  
[rdtest@n9860 ~]$ abaqus cae noGUI=test.py -- <input 1> <input 2> ... >> test.output
```

# Job Arrays

https://www.hoffman2.idre.ucla.edu

| c | | | | → | ↓ | ⌂ | ⌂ | ⌂



Report Typos and Errors

**UCLA**  
Hoffman2 Cluster User Guide

Search this website

ACCESS COMPUTING DATA STORAGE FILE TRANSFER SOFTWARE

You are here: Home

## Home

Welcome to the Hoffman2 Cluster User Guide. To look up usage information, click on one of the navigation tabs above or the sidebar items on the left.

### Recent Announcements

**Free Classes** IDRE Research Technology Group is pleased to present Fall Quarter classes. There is no charge or fee to attend these classes. See [IDRE Fall 2017 HPC Classes](#)

**Stata 15** A new version of Stata has been installed. See [Stata 15 available](#) for further information.

About User Support FAQ Getting started News Security policy Status How to acknowledge the Hoffman2 cluster program How to use this site

# Job Arrays

https://www.hoffman2.idre.ucla.edu

| c |  |  |  | → | ↓ | ⌂ | ⌄ | ⌄

  
Report  
Typos  
and  
Errors

**UCLA**  
Hoffman2 Cluster User Guide

array job

ACCESS COMPUTING DATA STORAGE FILE TRANSFER SOFTWARE

You are here: [Home](#)

## Home

Welcome to the Hoffman2 Cluster User Guide. To look up usage information, click on one of the navigation tabs above or the sidebar items on the left.

### Recent Announcements

**Free Classes** IDRE Research Technology Group is pleased to present Fall Quarter classes. There is no charge or fee to attend these classes. See [IDRE Fall 2017 HPC Classes](#)

**Stata 15** A new version of Stata has been installed. See [Stata 15 available](#) for further information.

About   
User Support  
FAQ  
Getting started  
News  
Security policy  
Status  
How to acknowledge the Hoffman2 cluster program  
How to use this site

# Job Arrays

https://www.hoffman2.idre.ucla.edu/computing/job\_arrays/ 120% Search Report Typo and Errors

**idre**  
**UCLA**  
Hoffman2 Cluster User Guide

Search this website

ACCESS COMPUTING DATA STORAGE FILE TRANSFER SOFTWARE

You are here: Home / Computing / Running an Array of Jobs Using UGE

## Running an Array of Jobs Using UGE

Contents [hide]

UGE Job Arrays  
File: hellotask.sh  
File: data1.in  
File: data2.in  
Using the Queue Script to Submit a Job Array

### UGE Job Arrays

The Univa Grid Engine (UGE) has a job array option, which lets you submit multiple jobs with a single command file. It is convenient to use job arrays when you need to conduct a parametric study to optimize the various parameters in an experiment. Computing the optimum solution involves repeated execution of same program over a range of input values and this is exactly what the job array option lets you do.

To run using job arrays, you need to insert an instruction of the form:

`#$ -t lower-upper:interval`

into your UGE command file.

# Array jobs: submit multiple jobs with a single command file

- can your job be broken down in a series of tasks that are totally independent one from the other?
  - generally not if you are predicting tomorrow's weather...
  - great if you are doing a parametric study

```
#!/bin/sh
echo "Task id is $SGE_TASK_ID"
if [ -e $HOME/JOBARRAY/data$SGE_TASK_ID.in ]; then
while read file
do
echo "hello $file"
done < $HOME/JOBARRAY/data$SGE_TASK_ID.in
fi
```

# Array jobs: submit multiple jobs with a single command file

```
[rdtest@login1 ~]$ mkdir $HOME/JOBARRAY
[rdtest@login1 ~]$ cd $HOME/JOBARRAY
[rdtest@login1 ~/JOBARRAY]$ cp /u/local/apps/submit_scripts/array_jobs/hellotask.sh ./
[rdtest@login1 ~/JOBARRAY]$ cp /u/local/apps/submit_scripts/array_jobs/data*.in ./
[rdtest@login1 ~/JOBARRAY]$ jobarray.q
```

Enter to continue.

```
jobarray Serial UGE job script
Functions (acceptable abbreviations are shown in CAPS)
Menu:      Display this menu
Info:      Display help information
Build:     Build a UGE .cmd file for jobarray
Submit:    Submit a UGE .cmd file for execution
Status:    Display the status of UGE jobs for rdtest
SYsstat:   Display the status of UGE jobs for the system
Hold:      Hold a UGE job
RElease:   Release a UGE job that is held
RESet:     Reset the priority of a UGE job
Cancel:    Cancel a UGE job
Quit:     Exit this script
```

Command: b

```
Enter the name of the Job array program or script to be executed
<or quit>: hellotask.sh
[...]
```

# Array jobs

```
[rdtest@login1 ~] $ head nn 35 hello task.sh.cmd
```

```
#!/bin/csh -f
# hellotask.sh.cmd
#
# UGE job for hellotask.sh built Wed Oct 11 20:51:21 PDT 2017
#
# The following items pertain to this script
# Use current working directory
#$ -cwd
# input          = /dev/null
# output         = /u/home/r/rdtest/matlab/ARRAYJOBS/hellotask.sh.joblog.$JOB_ID.$TASK_ID
#$ -o /u/home/r/rdtest/matlab/ARRAYJOBS/hellotask.sh.joblog.$JOB_ID.$TASK_ID
# error          = Merged with joblog
#$ -j y
# The following items pertain to the user program
# user program   = /u/home/r/rdtest/matlab/ARRAYJOBS/hellotask.sh
# arguments      =
# program input  = Specified by user program
# program output = Specified by user program
# Resources requested
#
#$ -l h_data=1024M,h_rt=1:00:00
# #
# Name of application for log
#$ -v QQAPP=job
# Email address to notify
#$ -M rdtest@mail
# Notify at beginning and end of job
#$ -m a
# Job is not rerunable
#$ -r n
#
# Job array indexes
#$ -t 1-2:1
[...]
```

# Array jobs

```
[rdtest@login1 ~/JOBARRAY]$ ls -lt
total 24
-rw-r--r-- 1 rdtest systems 547 Oct 11 20:52 hellotask.sh.joblog.868129.2
-rw-r--r-- 1 rdtest systems 149 Oct 11 20:52 hellotask.sh.output.868129.2
-rw-r--r-- 1 rdtest systems 544 Oct 11 20:52 hellotask.sh.joblog.868129.1
-rw-r--r-- 1 rdtest systems 149 Oct 11 20:52 hellotask.sh.output.868129.1
-rwxr--r-- 1 rdtest systems 3165 Oct 11 20:51 hellotask.sh.cmd
-rw-r--r-- 1 rdtest systems 13 Oct 11 20:50 data2.in
-rw-r--r-- 1 rdtest systems 13 Oct 11 20:50 data1.in
-rwxr--r-- 1 rdtest systems 183 Oct 11 20:50 hellotask.sh
[rdtest@n9860 ~/JOBARRAY]$ head -n3 hellotask.sh.output.868129.1
"Task id is 1"
"hello first"
"hello second"
```

If your job can be broken in 100s, or 1000s or 10,000s independent tasks you are in business!!!

# Array jobs R example

```
[rdtest@login3 R]$ cp /u/local/apps/submit_scripts/array_jobs/R/test_array.R ./
[rdtest@login3 R]$ cp /u/local/apps/submit_scripts/array_jobs/R/submit_R_array.sh./
[rdtest@login3 R]$ cat test_array.R
```

```
options(echo=TRUE) # if you want see commands in output file
args <- commandArgs(trailingOnly = TRUE)
print(args)
n <- length(args)
print(n)
suff <- args[n]
outputfile <- paste("myout", ".", suff, sep="")
write(args, file=outputfile, ncolumns = n,
      append = FALSE, sep = " ")
Sys.sleep(120)
```

# Array jobs R example

```
[rdtest@login3 R]$ cat submit_R_array.sh
```

```
#!/bin/bash
#$ -cwd
#$ -o test.joblog.$JOB_ID.$TASK_ID
#$ -j y
# Resources requested:
#$ -l h_data=1g,h_rt=2:00:00
# Email address to notify
#$ -M $USER@mail
#$ -m bea
#$ -t 1-3:1

echo ""
echo "test started on:  "` hostname -s `"
echo "test started at:  "` date `"
echo ""

#
. /u/local/Modules/default/init/modules.sh
module load R

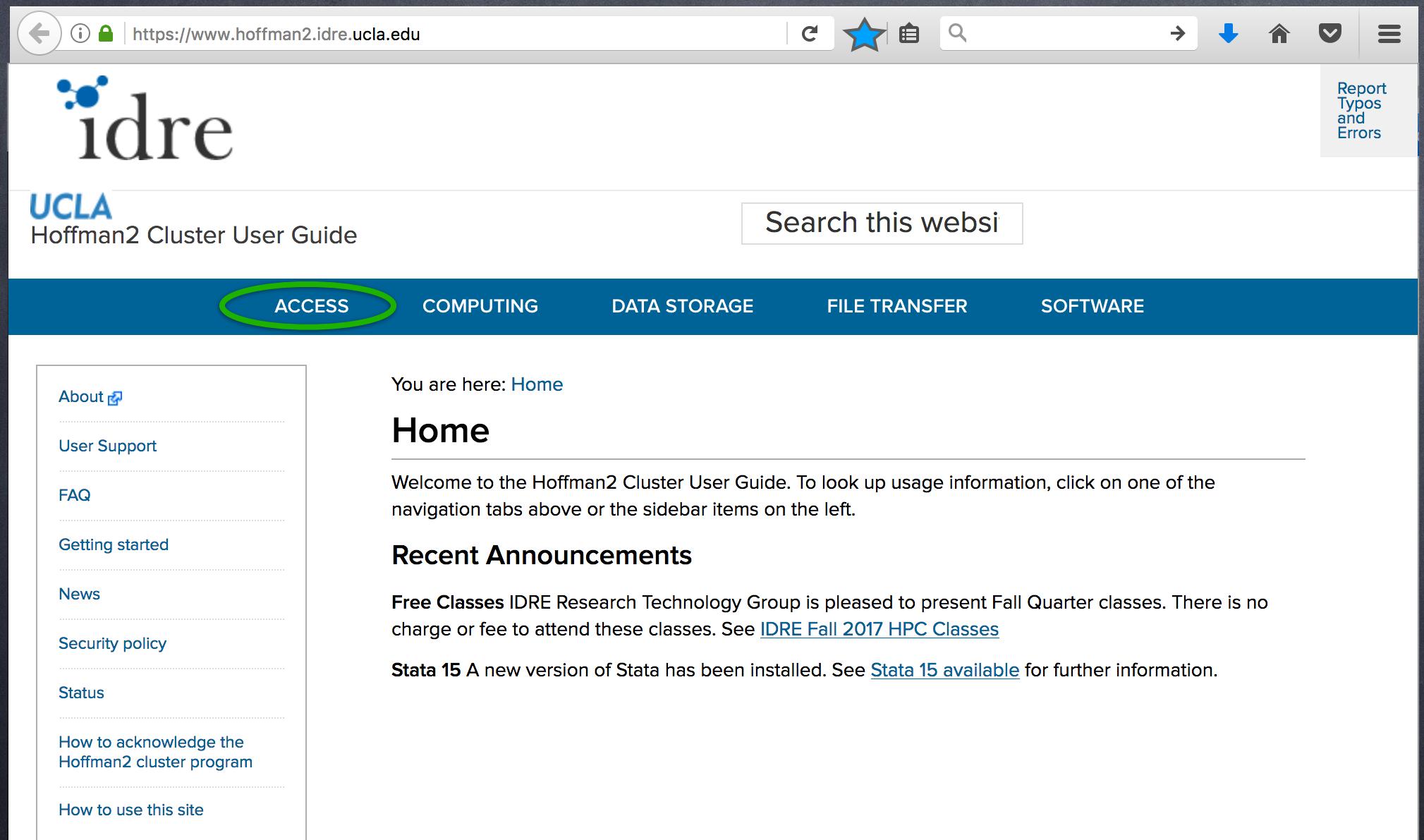
echo "Task id is \$SGE_TASK_ID"

#
# Run the user program
#
R CMD BATCH --no-save --no-restore "--args \$SGE_TASK_ID" ./test_array.R \
test_array.out.${JOB_ID}.${SGE_TASK_ID}
```

# Array jobs in R

```
[rdtest@login3 R]$ chmod u+x submit_R_array.sh
[rdtest@login3 R]$ qsub submit_R_array.sh
Your job-array 868598.1-3:1 ("submit_R_array.sh") has been submitted
[rdtest@login3 R]$ ls -lt
total 32
-rw-r--r-- 1 rdtest systems 1093 Oct 11 21:40 test_array.out.868598.3
-rw-r--r-- 1 rdtest systems 1093 Oct 11 21:40 test_array.out.868598.2
-rw-r--r-- 1 rdtest systems 1093 Oct 11 21:40 test_array.out.868598.1
-rw-r--r-- 1 rdtest systems    2 Oct 11 21:38 myout.3
-rw-r--r-- 1 rdtest systems    2 Oct 11 21:38 myout.1
-rw-r--r-- 1 rdtest systems    2 Oct 11 21:38 myout.2
-rw-r--r-- 1 rdtest systems   94 Oct 11 21:38 test.joblog.868598.3
-rw-r--r-- 1 rdtest systems   94 Oct 11 21:38 test.joblog.868598.2
-rw-r--r-- 1 rdtest systems   94 Oct 11 21:38 test.joblog.868598.1
-rwxr--r-- 1 rdtest systems  529 Oct 11 21:29 submit_R_array.sh
-rw-r--r-- 1 rdtest systems  295 Oct 11 21:28 test_array.R
```

# Jupyter notebooks



The screenshot shows a web browser window with the URL <https://www.hoffman2.idre.ucla.edu>. The page title is "Jupyter notebooks". The header includes a back button, a lock icon, and the URL. To the right are icons for refresh, star, search, forward, download, home, email, and menu. Below the header is the UCLA idre logo and a "Report Typos and Errors" link. On the left, there's a sidebar with links like "About", "User Support", "FAQ", "Getting started", "News", "Security policy", "Status", "How to acknowledge the Hoffman2 cluster program", and "How to use this site". The main content area shows the "Home" page with the breadcrumb "You are here: Home". A green oval highlights the "ACCESS" tab in the navigation bar. Other tabs include "COMPUTING", "DATA STORAGE", "FILE TRANSFER", and "SOFTWARE". The main content area has a heading "Recent Announcements" with two entries: "Free Classes" and "Stata 15".

Report Typos and Errors

UCLA

Hoffman2 Cluster User Guide

Search this website

ACCESS COMPUTING DATA STORAGE FILE TRANSFER SOFTWARE

About 

User Support

FAQ

Getting started

News

Security policy

Status

How to acknowledge the Hoffman2 cluster program

How to use this site

You are here: Home

## Home

Welcome to the Hoffman2 Cluster User Guide. To look up usage information, click on one of the navigation tabs above or the sidebar items on the left.

### Recent Announcements

**Free Classes** IDRE Research Technology Group is pleased to present Fall Quarter classes. There is no charge or fee to attend these classes. See [IDRE Fall 2017 HPC Classes](#)

**Stata 15** A new version of Stata has been installed. See [Stata 15 available](#) for further information.

# Jupyter notebooks

https://www.hoffman2.idre.ucla.edu/access/

Report Typos and Errors

## idre

### UCLA Hoffman2 Cluster User Guide

Search this website

ACCESS COMPUTING DATA STORAGE FILE TRANSFER SOFTWARE

You are here: Home / Accessing the Hoffman2 Cluster

## Accessing the Hoffman2 Cluster

Contents [hide]

- [Using SSH](#)
  - [Linux and Mac](#)
  - [Windows](#)
- [Opening GUI Applications](#)
- [Connecting via the NX client](#)
- [Connecting via a Jupyter notebook](#)
- [Additional Topics](#)

You can connect to the Hoffman2 cluster using the SSH protocol. For command line-type operations, that do not require opening of graphical interfaces, using SSH will suffice. To allow on your local computer opening of graphical interfaces (GUI applications) running on the cluster you will need to allow forwarding of the X Window System over SSH. NX is an alternative way to connect to the cluster that allows opening of a remote desktop on the cluster (with access to graphical interfaces). You can also connect to the Hoffman2 cluster via ipython notebooks.

# Jupyter notebooks

https://www.hoffman2.idre.ucla.edu/access/

Report Typos and Errors

idre

UCLA

Hoffman2 Cluster User Guide

Search this website

ACCESS COMPUTING DATA STORAGE FILE TRANSFER SOFTWARE

You are here: Home / Accessing the Hoffman2 Cluster

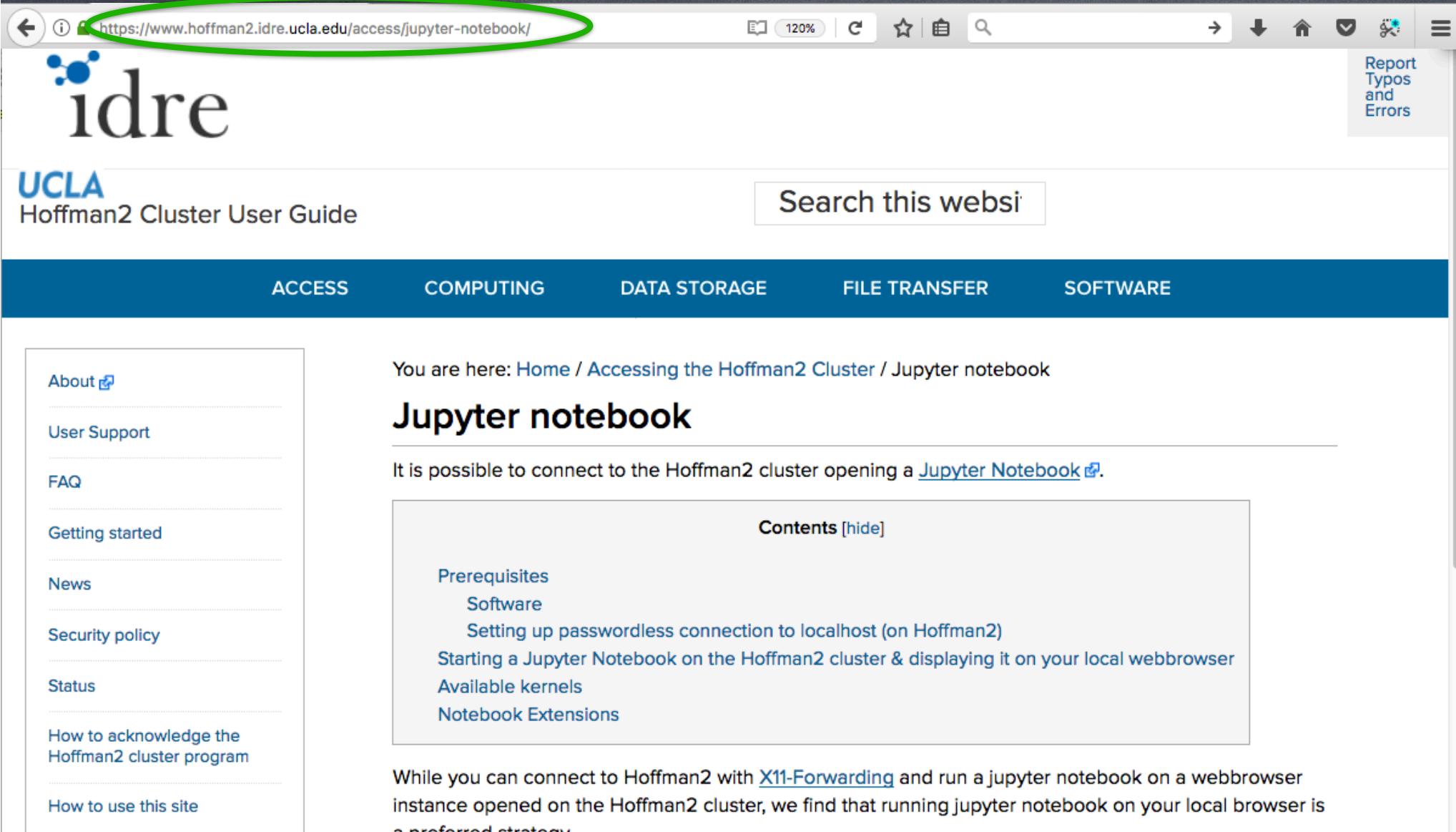
## Accessing the Hoffman2 Cluster

Contents [hide]

- [Using SSH](#)
  - [Linux and Mac](#)
  - [Windows](#)
- [Opening GUI Applications](#)
- [Connecting via the NX client](#)
- [Connecting via a Jupyter notebook](#)
- [Additional Topics](#)

You can connect to the Hoffman2 cluster using the SSH protocol. For command line-type operations, that do not require opening of graphical interfaces, using SSH will suffice. To allow on your local computer opening of graphical interfaces (GUI applications) running on the cluster you will need to allow forwarding of the X Window System over SSH. NX is an alternative way to connect to the cluster that allows opening of a remote desktop on the cluster (with access to graphical interfaces). You can also connect to the Hoffman2 cluster via ipython notebooks.

# Jupyter notebooks



The screenshot shows a web browser window with the URL <https://www.hoffman2.idre.ucla.edu/access/jupyter-notebook/> highlighted by a green oval. The page content is from the UCLA Hoffman2 Cluster User Guide, specifically the Jupyter notebook section. The page includes a navigation bar with links for ACCESS, COMPUTING, DATA STORAGE, FILE TRANSFER, and SOFTWARE. A sidebar on the left contains links for About, User Support, FAQ, Getting started, News, Security policy, Status, How to acknowledge the Hoffman2 cluster program, and How to use this site. The main content area displays the Jupyter notebook section, which explains how to connect to the cluster using a Jupyter Notebook and provides a list of prerequisites and software components.

https://www.hoffman2.idre.ucla.edu/access/jupyter-notebook/

Report Typos and Errors

idre

UCLA Hoffman2 Cluster User Guide

Search this website

ACCESS COMPUTING DATA STORAGE FILE TRANSFER SOFTWARE

About [About](#)

User Support

FAQ

Getting started

News

Security policy

Status

How to acknowledge the Hoffman2 cluster program

How to use this site

You are here: Home / Accessing the Hoffman2 Cluster / Jupyter notebook

## Jupyter notebook

It is possible to connect to the Hoffman2 cluster opening a [Jupyter Notebook](#).

Contents [hide]

Prerequisites

- Software
- Setting up passwordless connection to localhost (on Hoffman2)
- Starting a Jupyter Notebook on the Hoffman2 cluster & displaying it on your local webbrowser
- Available kernels
- Notebook Extensions

While you can connect to Hoffman2 with [X11-Forwarding](#) and run a jupyter notebook on a webbrowser instance opened on the Hoffman2 cluster, we find that running jupyter notebook on your local browser is a preferred strategy.

To run an jupyter notebook on the Hoffman2 cluster while displaying it on your local browser please follow the steps outlined here.

# Running R on the Hoffman2 cluster

- **for interactive use:**

- ssh -X <youruserid>@hoffman2.idre.ucla.edu
- qrsh -l h\_data=1g,h\_rt=2:00:00 -pe shared 8
- export OMP\_NUM\_THREADS=8
- module load R/3.3.3
- R

- **for batch use:**

- ssh <youruserid>@hoffman2.idre.ucla.edu
- R333.q

- **for notebook use:**

- you will need:
  - a terminal (cygwin if on windows)
  - a local installation of python (version 2.6 or greater)
  - a web browser
- navigate to:
  - <https://www.hoffman2.idre.ucla.edu/access/jupyter-notebook/>
  - download the script: [h2jupynb](#)
- run:
  - python h2jupynb
  - run R from your web browser...

# Running python on the Hoffman2 cluster

- **for interactive use:**

- ssh -X <youruserid>@hoffman2.idre.ucla.edu
- qrsh -l h\_data=1g,h\_rt=2:00:00 -pe shared 8
- export OMP\_NUM\_THREADS=8
- module load python/2.7.13
- python

- **for batch use:**

- ssh <youruserid>@hoffman2.idre.ucla.edu
- Write a submission script or use multithread.q

- **for notebook use:**

- python h2jupynb

# Jupyter notebooks

```
dauria@alba:~> jupyter-notebook/h2jupynb --help
```

Usage:

```
h2jupynb [-u <Hoffman2 user name>] [-t <time in hours>] [-m <memory in GB>]  
[-s <number of slots>] [-v <python-version>] [-p <port>] [-d <dir>]
```

If no arguments are given to this script it is assumed that:

your Hoffman2 user name is the same as on your client machine

the time duration for your session is of 2 hours

the memory per slot for your session is of 1GB

the number of slots for your session is of 1

the python version for your notebook is 2.7.13

the port on which the server is started is 8789

the starting directory on Hoffman2 is your \$HOME

python versions currently available are 2.7.13 or 3.6.1

python versions: 2.7.3 or 3.4 are available but deprecated.

# Jupyter notebooks

```
dauria@alba:~> jupyter-notebook/h2jupynb -u rdtest
```

Your Hoffman2 user name is rdtest

The time in hours is 2

The memory in GB per slots is 1

The number of slots is 1

The version of python for the notebook is 2.7.13

The port is 9360

The directory on Hoffman2 is \$HOME

rdtest@hoffman2.idre.ucla.edu's password: <- enter password here

Last login: Sat Oct 7 17:05:00 2017 from cpe-76-91-252-105.socal.res.rr.com

Welcome to the Hoffman2 Cluster!

[...]

```
qrsh -N JUPYNB -l i,h_rt=2:00:00,h_data=1g -pe dc\* 1
```

```
module load python/2.7.13
```

```
echo HOSTNAME=`hostname`
```

```
[rdtest@login4 ~]$ qrsh -N JUPYNB -l i,h_rt=2:00:00,h_data=1g -pe dc\* 1
```

# Jupyter notebooks

```
module load python/2.7.13  
echo HOSTNAME=`hostname`  
[rdtest@n2188 ~]$ module load python/2.7.13
```

[...]

```
[rdtest@n2188 ~]$ echo HOSTNAME=`hostname`
```

```
HOSTNAME=n2188
```

```
[rdtest@n2188 ~]$ jupyter notebook --port=9360
```

[...]

<http://localhost:9360/?token=b13b6ba1a7406688721224e6398b0bfd693ce042444eec69>

```
ssh -4 -t -Y rdtest@hoffman2.idre.ucla.edu -L 9360:localhost:9360 ssh -t -Y n2188 -L 9360:localhost:9360
```

Pseudo-terminal will not be allocated because stdin is not a terminal.

rdtest@hoffman2.idre.ucla.edu's password: **<- enter password here**

Warning: No xauth data; using fake authentication data for X11 forwarding.

Pseudo-terminal will not be allocated because stdin is not a terminal.

Warning: no access to tty (Bad file descriptor).

Thus no job control in this shell.

TUNNEL

Succesfully opened notebook!

Kill this process to end your notebook connection.

# Jupyter notebooks

A screenshot of a Jupyter Notebook interface. At the top, there's a browser-style header with tabs for "Jupyter notebook" and "Home". Below the header is a search bar and a toolbar with various icons. On the left, there's a sidebar titled "jupyter" with tabs for "Files", "Running", "Clusters", and "Nbextensions". The main area shows a file browser with a list of directories and files. A context menu is open on the right side, listing options like "Text File", "Folder", "Terminal", "Notebooks", "Python 2", "Python 3", and "R".

localhost:9360/tree?token=b13b6ba1a7406688721224e6398b0bfd693ce042444eec69

Search

jupyter

Logout

Files Running Clusters Nbextensions

Select items to perform actions on them.

Upload New

- Text File
- Folder
- Terminal

Notebooks

Python 2

Python 3

R

- abaqus\_plugins
- ado
- C
- comsol
- Desktop
- Documents
- Downloads
- gnu
- gsrc
- gui\_tmp
- intel
- IPython-notebook-extensions
- JOBARRAY

localhost:7114/notebooks/jupyter\_R\_example\_approx\_of\_pi.ipynb

jupyter R example approx of pi Last Checkpoint: 04/07/2017 (autosaved)

File Edit View Insert Cell Kernel Help

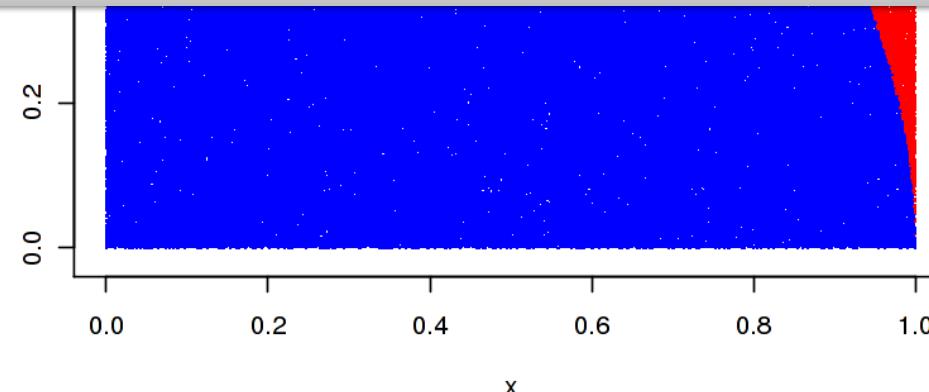
CellToolbar

```
In [2]: n <- 9000000
x <- runif(n); y <- runif(n)
inside <- x^2+y^2 <= 1
pi <- 4*sum(inside)/n
idx <- sample.int(trunc(n/(log(n))), replace=FALSE)
x <- x[idx]; y <- y[idx]; inside <- inside[idx]
plot(x,y, col=ifelse(inside,"blue","red"), cex = 0.5, pch = ".", main = sprintf("Bootstrap approx of pi", n, pi))
print(pi)
```

[1] 3.141733

**Bootstrap approx of pi**

You interact with your localhost while the heavy lifting is done on the cluster!!!



localhost:7432/notebooks/plot\_w\_jupyter\_nb.ipynb

x01 status ATSWiki jira db.ats IDRE Support Monitoring PyPI vSphere Web Client Coursera Box H2 User Guide

# jupyter plot\_w\_jupyter\_nb (autosaved)

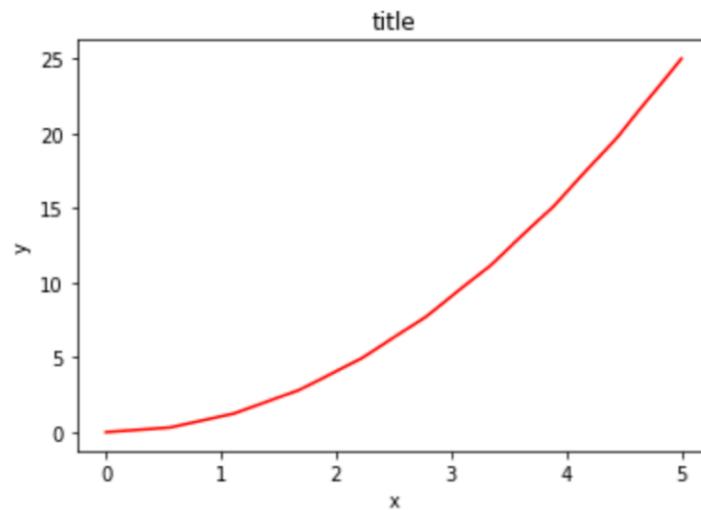
Logout

File Edit View Insert Cell Kernel LaTeX\_envs Help Python 2

Code CellToolbar

```
In [1]: %pylab inline
from pylab import *
import matplotlib.pyplot as plt
from pylab import *
x = linspace(0, 5, 10)
y = x ** 2
figure()
plot(x, y, 'r')
xlabel('x')
ylabel('y')
title('title')
show()
```

Populating the interactive namespace from numpy and matplotlib



```
In [ ]:
```

# Jupyter notebook

Warning: No xauth data; using fake authentication data for X11 forwarding.

Pseudo-terminal will not be allocated because stdin is not a terminal.

Warning: no access to tty (Bad file descriptor).

Thus no job control in this shell.

TUNNEL

Succesfully opened notebook!

Kill this process to end your notebook connection.

**^C** Traceback (most recent call last):

File "jupyter-notebook/h2jupynb", line 242, in <module>  
    time.sleep(timeinhours\*3600)

KeyboardInterrupt

Killed by signal 2.

The screenshot shows a Jupyter Notebook interface running in a web browser. The title bar indicates the window is titled 'Jupyter notebook' and the tab is 'plot\_w\_jupyter\_nb.ipynb'. A modal dialog box is open in the center, displaying the message 'Connection failed'. The message states: 'A connection to the notebook server could not be established. The notebook will continue trying to reconnect. Check your network connection or notebook server configuration.' There is an 'OK' button at the bottom right of the dialog. In the background, the notebook interface shows a code cell labeled 'In [1]' containing Python code related to plotting a parabola. The status bar at the top right shows 'Not Connected' and 'Python 2'.