<table>
<thead>
<tr>
<th>Quick Links</th>
</tr>
</thead>
<tbody>
<tr>
<td>HPC Home</td>
</tr>
<tr>
<td>Getting an account</td>
</tr>
<tr>
<td>Getting started on Prince</td>
</tr>
<tr>
<td>Prince How-to Articles</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Logging in</th>
</tr>
</thead>
<tbody>
<tr>
<td>Windows</td>
</tr>
<tr>
<td>Mac/Linux</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Clusters and Storage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prince (HPC)</td>
</tr>
<tr>
<td>Dumbo (Hadoop)</td>
</tr>
<tr>
<td>Brooklyn (OpenStack)</td>
</tr>
<tr>
<td>Dalmata (NYU Abu Dhabi)</td>
</tr>
</tbody>
</table>
Transferri
g data
to/from
the
clusters
Transferri
g data
to/from
Prince
cluster
using
Globus
Submittin
g jobs
with
sbatch
Available
software
Licensed
Software
Available
on the
HPC
Cluster
Building
Software
Slurm
Tutorial
Tutorials
FAQs
Scratch
Area
Cleanup
Programming for
Biologist
s
Acknowle
dge
Statement
Research
Gallery
HPC
People
Logging In

To set up a reusable tunnel, follow the instructions here.

```bash
# via a previously-created and started tunnel:
ssh prince

# create a one-off tunnel:
ssh -L 8023:prince:22 NetID@hpc.nyu.edu
# use it (in another terminal window)
ssh -X -p 8023 localhost
```

Finding Software (modules)

```bash
# list the available modules
module avail

# ok, that list was too long. I want matlab.
# Show me modules starting with 'm'
module avail m

# check what modules I currently have loaded
module list

# unload all of them
module purge

# load matlab
module load matlab/2014a
```

Interactive session
# start an interactive batch setting with
# default limits (1 CPU for 1 hour walltime)
qsub -X -I

# I need 4 CPUs, and for 2 hours:
qsub -X -I -l nodes=1:ppn=4 -l walltime=2:00:00

# I need more than 4 hours, and don't mind waiting
# a while for it to start (the fast-response
# interactive queue has a limit of 4 hours, so must
# use the standard queue)
qsub -X -I -l walltime=8:00:00 -q s48

Transferring files to/from the cluster

These examples assume you have an ssh tunnel running!

# copy from my workstation to my $HOME on Prince:
scp file1 file2 file3 prince:

# copy whole directories from my workstation
# to $HOME on Prince:
scp -r dir1/ dir2/ prince:

# copy, keeping permissions and modification times
scp -p file1 file2 prince:

# copy from my workstation to my $SCRATCH on
# Prince: (note the '\')
scp file1 file2 file3 prince:\$SCRATCH

# A better way:
# replicate the current directory from my workstation
# as $SCRATCH/my_data on Prince - only sending files that
# are not already there, or are there but not the same:
rsync -a . prince:\$SCRATCH/my_data

# the same, more noisily so I can see what was transferred:
rsync -av . prince:\$SCRATCH/my_data

Sample PBS job scripts
6-hour, 1-CPU job, named the same as the script

```bash
#!/bin/bash
#PBS -l mem=4GB
#PBS -l walltime=6:00:00

cd $SCRATCH/my_run_dir
./model.exe > model_out.txt
```

6-hour, 8-CPU OpenMP (threaded), named the same as the script

```bash
#!/bin/bash
#PBS -l mem=4GB
#PBS -l walltime=6:00:00
#PBS -l nodes=1:ppn=8

cd $SCRATCH/my_run_dir
# optional but not necessary:
# export OMP_NUM_THREADS=$PBS_NUM_PPN
./my_program
```

Running an MPI job

Pure MPI (1 CPU per MPI task), 4 MPI tasks, total memory 40GB (10GB/task)

```bash
#!/bin/bash
#PBS -l mem=40GB
#PBS -l walltime=6:00:00
#PBS -l procs=4

module load openmpi/intel/1.6.5

cd $SCRATCH/my_run_dir
# run one MPI task on each CPU
# (equivalent to mpiexec -np $PBS_NP ./my_program):
mpiexec .my_program
```
#!/bin/bash
#PBS -l mem=40GB
#PBS -l walltime=6:00:00
#PBS -l nodes=4:ppn=8

module load openmpi/intel/1.6.5

cd $SCRATCH/my_run_dir
export OMP_NUM_THREADS=$PBS_NUM_PPN
mpiexec --bynode -np $PBS_NUM_NODES ./my_program