HPC Cluster Access and Usage Manual

NUST OpenHPC Cluster

1. Accessing the HPC Cluster

1.1 Determine Your Location

Inside NUST Network:

- You can directly access the HPC cluster without VPN
- Proceed directly to Section 2 (Login Instructions)

Outside NUST Network:

- VPN connection is mandatory
- HPC cluster is not accessible from outside NUST without VPN
- Follow Section 1.2 below

1.2 VPN Access for External Users

Step 1: Request VPN Access

- Send an email to: **asa@sines.nust.edu.pk**
- Subject: "HPC VPN Access Request"
- Include in your email:
  - Full name
  - Department/Affiliation
  - Student/Employee ID
  - Purpose of HPC access

Step 2: Receive VPN Credentials

- ASA will respond with VPN setup instructions
- You will receive:
  - VPN configuration file or credentials
  - Installation guide
  - Connection instructions

Step 3: Connect to VPN

- Follow the instructions provided by ASA carefully
- Install the VPN client as directed
- Connect to NUST VPN before attempting HPC login
- Verify VPN connection is active before proceeding

2. Login Instructions

You can access the HPC cluster using either MobaXterm (recommended) or PuTTY.

2.1 Using MobaXterm (Recommended)

Installation:

1. Download MobaXterm from: https://mobaxterm.mobatek.net
2. Choose the "Home Edition" (Free)
3. Download the "Portable edition" or "Installer edition"
4. Install or extract the application

Login Steps:

1. Launch MobaXterm
2. Click the **Session** button (top-left corner)
3. Select **SSH** from the session types
4. In the SSH session settings:
    - **Remote host:** Enter the HPC master node IP address
    - **Specify username:** Check this box and enter your HPC username
    - **Port:** 22 (default)
5. Click **OK**
6. Enter your password when prompted
7. You are now connected to the HPC cluster

Advantages of MobaXterm:

- Built-in SFTP browser for file transfers
- X11 server included for graphical applications
- Session management and saved connections
- Split-terminal support

2.2 Using PuTTY

Installation:

1. Download PuTTY from: https://www.putty.org
2. Run the installer
3. Complete the installation process

Login Steps:

1. Launch PuTTY
2. In the configuration window:
    - **Host Name (or IP address):** Enter the HPC master node IP

   - **Port:** 22
   - **Connection type:** SSH
3. (Optional) Save the session:
   - Enter a name under "Saved Sessions"
   - Click **Save**
4. Click **Open**
5. If this is your first connection, accept the server's host key
6. Enter your username when prompted
7. Enter your password when prompted
8. You are now connected to the HPC cluster

3. File Transfer Methods

3.1 Using WinSCP (Windows Users)

WinSCP is a free SFTP/SCP client for Windows that provides a graphical interface for file transfers.

Installation:

1. Download WinSCP from: https://winscp.net
2. Run the installer
3. Follow the installation wizard
4. Complete the installation

Configuration and Login:

1. Launch WinSCP
2. In the Login dialog:
   - **File protocol:** SFTP
   - **Host name:** Enter the given HPC master node IP address
   - **Port number:** 22
   - **Username:** Enter your HPC username
   - **Password:** Enter your HPC password
3. (Optional) Click **Save** to save this session for future use
   - Enter a session name
   - Click **OK**
4. Click **Login**
5. If this is your first connection, accept the server's host key

Using WinSCP Interface:

Left Panel (Local Computer):

- Shows your local Windows files and folders
- Navigate to files you want to upload

Right Panel (HPC Cluster):

- Shows your HPC home directory and files
- Navigate to where you want to place files

Transferring Files:

Upload (Local → HPC):

1. Navigate to the file/folder on the left panel (local)
2. Select the file(s) or folder(s)
3. Drag and drop to the right panel (HPC)
    - OR right-click → Upload
    - OR click the "Upload" button in toolbar
4. Monitor transfer progress in the transfer queue

Download (HPC → Local):

1. Navigate to the file/folder on the right panel (HPC)
2. Select the file(s) or folder(s)
3. Drag and drop to the left panel (local)
    - OR right-click → Download
    - OR click the "Download" button in toolbar
4. Monitor transfer progress in the transfer queue

Synchronize Directories:

1. Select a directory on either panel
2. Click **Commands → Synchronize**
3. Choose synchronization direction:
    - Local → Remote (upload changes)
    - Remote → Local (download changes)
    - Both (keep both sides updated)
4. Review changes and click **OK**

Remote File Management:

- **Create new folder:** Right-click in right panel → New → Directory
- **Delete files:** Select file → press Delete key
- **Rename files:** Select file → press F2
- **Edit files:** Right-click file → Edit (opens in default editor)
- **File permissions:** Right-click file → Properties → Change permissions

Important WinSCP Notes:

⚠️ **VPN Required (External Users):** Ensure VPN is connected before launching WinSCP

- ✓ **Resume capability:** WinSCP can resume interrupted transfers
- ✓ **Keep session active:** Enable "Keep Session Active" in advanced settings for long transfers
- ✓ **Queue management:** Multiple transfers are queued automatically

3.2 Using MobaXterm SFTP Browser

If you're using MobaXterm for SSH access, file transfer is built-in.

Accessing SFTP Browser:

1. Connect to HPC via SSH in MobaXterm (see Section 2.1)
2. The left sidebar automatically shows the SFTP browser
3. Navigate through remote directories

Transferring Files:

- **Upload:** Drag files from Windows Explorer to MobaXterm SFTP sidebar
- **Download:** Drag files from SFTP sidebar to Windows Explorer
- **Right-click options:** Upload, download, delete, rename, etc.

3.3 Command-Line File Transfer (Linux/Mac Users)

SCP (Secure Copy):

```
# Upload file to HPC
scp local_file.txt username@hpc_master_ip:/remote/path/
# Upload directory to HPC
scp -r local_directory/ username@hpc_master_ip:/remote/path/
# Download file from HPC
scp username@hpc_master_ip:/remote/path/file.txt ./
# Download directory from HPC
scp -r username@hpc_master_ip:/remote/path/directory/ ./
```

SFTP (Interactive Session):

```
# Connect to HPC
sftp username@hpc_master_ip
# SFTP commands once connected:
ls                          # List remote files
```

```
lls                          # List local files
cd /remote/path              # Change remote directory
lcd /local/path              # Change local directory
put local_file.txt           # Upload file
get remote_file.txt          # Download file
put -r local_directory/      # Upload directory
get -r remote_directory/     # Download directory
exit                         # Close SFTP session
```

3.4 File Transfer Best Practices

✓ **Compress large files:** Use tar/gzip before transferring large datasets

```
# On local machine before upload:
tar -czf dataset.tar.gz dataset/
# On HPC after download
tar -xzf dataset.tar.gz
```

✓ **Verify transfers:** Check file sizes and checksums after large transfers

```
# On both local and remote:
ls -lh filename
md5sum filename
```

✓ **Use appropriate tool:**

- Small files (<100MB): Any method works
- Large files (>1GB): WinSCP or command-line with compression
- Many small files: Compress into archive first

✓ **Organize your data:**

- Create project-specific directories on HPC
- Keep input data separate from output results
- Use descriptive filenames

⚠️ **Security reminder:** Never transfer sensitive credentials or API keys in plain text file

4. HPC Cluster Architecture

3.1 Cluster Specifications

Platform: OpenHPC-based cluster **Job Scheduler:** SLURM (Simple Linux Utility for Resource Management)

Node Configuration:

- **1 Master Node:**
  - Manages job scheduling and resource allocation
  - Hosts SLURM controller
  - **DO NOT run compute jobs on this node**
  - Master node name:

```
master
```

- **4 Compute Nodes:**
  - Execute submitted jobs
  - All computational work must run on these nodes
  - Accessed automatically through SLURM
  - Compute node names are:

```
compute1
compute2
compute3
compute4
```

3.2 Important Rules

⚠ **CRITICAL:** Jobs submitted or executed directly on the master node will be **automatically killed**

✓ **Correct approach:** Submit all jobs through SLURM, which allocates them to compute nodes

✓ **Access method:** You do not SSH directly to compute nodes; SLURM handles resource allocation

5. Working with SLURM

4.1 Basic Workflow

1. Prepare your job script or command
2. Submit the job using SLURM commands
3. Monitor job status
4. Retrieve results when job completes

4.2 Job Types

Batch Jobs:

- Non-interactive jobs submitted via script
- Suitable for long-running computations
- Submitted using `sbatch`

Interactive/Serial Jobs:

- Real-time interactive sessions
- Useful for testing and debugging
- Submitted using `srun` or `salloc`

ANNEX: SLURM Command Reference

a. Cluster Information Commands

Command	Description	Example
`sinfo`	Display node and partition status	`sinfo`
`sinfo -N`	Show node-specific information	`sinfo -N`
`sinfo -N -l`	Detailed node information	`sinfo -N -l`
`scontrol show partition`	Show partition details	`scontrol show partition`
`scontrol show node`	Show all node details	`scontrol show node`

b. Job Monitoring Commands

Command	Description	Example
`squeue`	View all jobs in queue	`squeue`
`squeue -u $USER`	View your jobs only	`squeue -u $USER`
`squeue -j <job_id>`	View specific job	`squeue -j 12345`
`sacct`	View job accounting information	`sacct`
`sacct -j <job_id>`	View details of specific job	`sacct -j 12345`
`scontrol show job <job_id>`	Detailed job information	`scontrol show job 12345`

c. Creating and Editing Script Files on HPC

Before you can submit jobs to SLURM, you need to create script files on the HPC cluster. There are several methods to do this.

Option A: Using nano (Easiest for beginners)

1. **Log in to HPC** via SSH (MobaXterm or PuTTY)
2. **Navigate to your working directory:**

```
cd ~/my_project
```

3. **Create a new script file:**

```
nano job_script.sh
```

4. **Type or paste your script content:**

```
#!/bin/bash
#SBATCH --job-name=test_job
#SBATCH --output=output_%j.txt
```

```
#SBATCH --error=error_%j.txt
#SBATCH --nodes=1
#SBATCH --ntasks=1
#SBATCH --time=01:00:00
#SBATCH --mem=4G
module load python/3.9
python my_program.py
```

5. **Save and exit:**
    - Press Ctrl + O to save (write out)
    - Press Enter to confirm filename
    - Press Ctrl + X to exit
6. **Make the script executable:**

```
chmod +x job_script.sh
```

Nano Editor Quick Reference:

- Ctrl + O : Save file
- Ctrl + X : Exit editor
- Ctrl + K : Cut line
- Ctrl + U : Paste line
- Ctrl + W : Search text
- Ctrl + G : Get help

Option B: Using vi/vim (For advanced users)

1. **Create/edit a file:**

```
vi job_script.sh
```

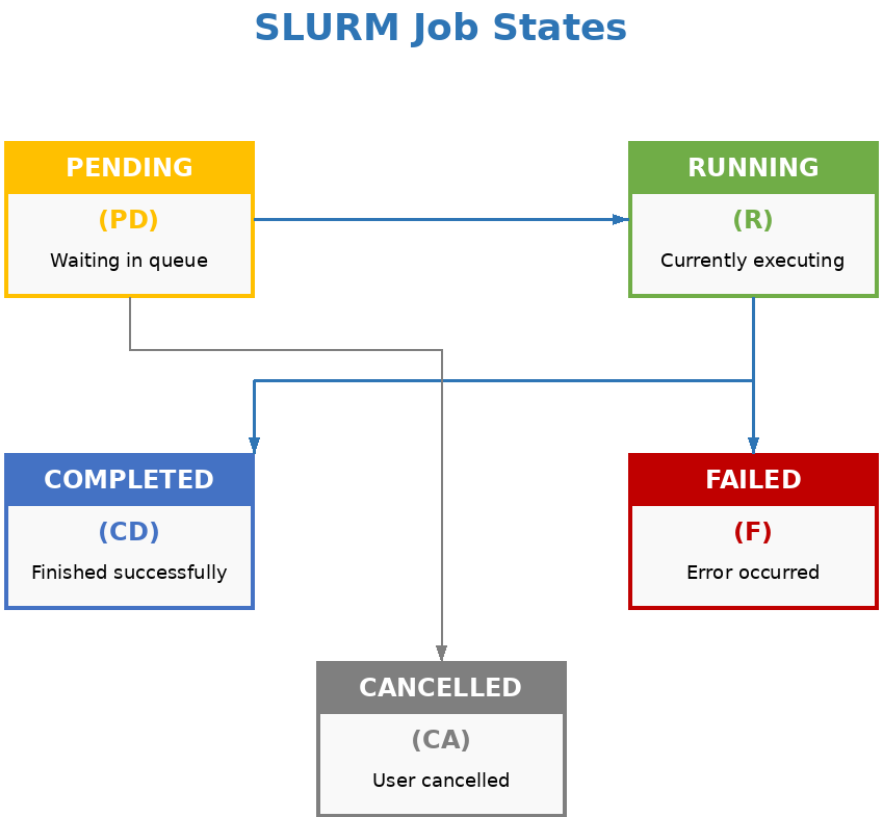
2. **Enter insert mode:**
    - Press i to start typing
3. **Type your script content**
4. **Save and exit:**
    - Press Esc to exit insert mode
    - Type :wq and press Enter to save and quit
    - OR type :q! to quit without saving

Vi/Vim Quick Reference:

- i : Enter insert mode
- Esc : Exit insert mode

- :w : Save file
- :q : Quit
- :wq : Save and quit
- :q! : Quit without saving
- dd : Delete line
- yy : Copy line
- p : Paste line

A.3 Job Submission Commands



Batch Job Submission

```
sbatch job_script.sh
```

Sample Batch Job Script (job_script.sh):

```
#!/bin/bash
#SBATCH --job-name=my_analysis          # Job name
#SBATCH --output=output_%j.txt          # Standard output (%j = job ID)
#SBATCH --error=error_%j.txt            # Standard error
#SBATCH --nodes=1                       # Number of nodes
```

```
#SBATCH --ntasks=1                    # Number of tasks
#SBATCH --cpus-per-task=4             # CPU cores per task
#SBATCH --time=02:00:00               # Time limit (HH:MM:SS)
#SBATCH --mem=4G                      # Memory per node
# Load required modules
module load python/3.9
module load gcc/11.2
# Run your program
python my_script.py
# Or run a compiled program
# ./my_program input.dat
```

Interactive Job Submission

```
# Request interactive session with default resources
srun --pty bash

# Request interactive session with specific resources
srun --nodes=1 --ntasks=4 --time=01:00:00 --pty bash

# Allocate resources and then run commands
salloc --nodes=1 --ntasks=4 --time=01:00:00
```

Serial Job Example

```
#!/bin/bash
#SBATCH --job-name=serial_job
#SBATCH --output=serial_%j.txt
#SBATCH --ntasks=1
#SBATCH --time=00:30:00
#SBATCH --mem=2G

# Load modules
module load myapp

# Run serial application
./my_serial_program input.txt
```

A.4 Job Control Commands

Command	Description	Example
scancel <job_id>	Cancel specific job	`scancel 12345`
scancel -u $USER	Cancel all your jobs	`scancel -u $USER`
scancel -n <job_name>	Cancel job by name	`scancel -n my_job`
scontrol hold <job_id>	Hold (pause) a job	`scontrol hold 12345`
scontrol release <job_id>	Release a held job	`scontrol release 12345`

A.5 Common SBATCH Directives

Directive	Description	Example
--job-name	Name of the job	`#SBATCH --job-name=test`
--output	Standard output file	`#SBATCH --output=out.txt`
--error	Standard error file	`#SBATCH --error=err.txt`
--nodes	Number of nodes	`#SBATCH --nodes=2`
--ntasks	Number of tasks	`#SBATCH --ntasks=8`
--cpus-per-task	CPUs per task	`#SBATCH --cpus-per-task=4`
--time	Time limit	`#SBATCH --time=01:30:00`
--mem	Memory per node	`#SBATCH --mem=8G`
--mem-per-cpu	Memory per CPU	`#SBATCH --mem-per-cpu=2G`
--partition	Partition to use	`#SBATCH --partition=compute`
--mail-type	Email notification type	`#SBATCH --mail-type=END,FAIL`
--mail-user	Email address	`#SBATCH --mail-user=user@nust.edu.pk`

A.6 Module System Commands

```
# List available modules
module avail
# Search for specific module
module avail python
# Load a module
module load python/3.9
# List loaded modules
module list
# Unload a module
module unload python/3.9
# Unload all modules
module purge
# Show module information
module show python/3.9
```

A.7 Complete Batch Job Examples

Example 1: Python Computation

```
#!/bin/bash

#SBATCH --job-name=python_calc
#SBATCH --output=results_%j.log
#SBATCH --error=errors_%j.log
#SBATCH --nodes=1
#SBATCH --ntasks=1
#SBATCH --cpus-per-task=8
#SBATCH --time=04:00:00
#SBATCH --mem=16G
```

```
# Load Python module
module load python/3.9
# Activate virtual environment (if applicable)
source ~/myenv/bin/activate
# Run Python script
python analysis.py --input data.csv --output results.txt
# Deactivate environment
deactivate
```

Example 2: Parallel MPI Job

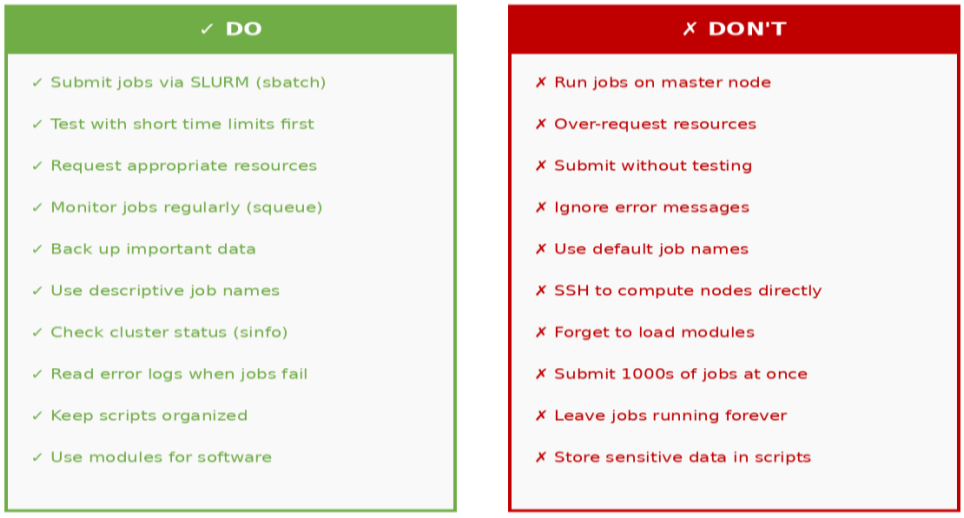
```
#!/bin/bash
#SBATCH --job-name=mpi_simulation
#SBATCH --output=mpi_out_%j.txt
#SBATCH --nodes=2
#SBATCH --ntasks-per-node=4
#SBATCH --time=08:00:00
#SBATCH --mem-per-cpu=4G
# Load MPI module
module load openmpi/4.1

# Run MPI program
mpirun ./my_mpi_program
```

Example 3: Array Job (Multiple Similar Jobs)

```
#!/bin/bash
#SBATCH --job-name=array_job
#SBATCH --output=array_%A_%a.out
#SBATCH --error=array_%A_%a.err
#SBATCH --array=1-10
#SBATCH --ntasks=1
#SBATCH --time=01:00:00
#SBATCH --mem=2G
# Load required modules
module load myapp
# Process file based on array task ID
./process_file input_${SLURM_ARRAY_TASK_ID}.dat
```

5. Best Practices and Important Notes



5.3 Troubleshooting

Job doesn't start:

- Check cluster status: `sinfo`
- Verify your job requirements don't exceed available resources
- Check job queue position: `squeue -u $USER`

Job fails immediately:

- Check error file specified in `--error`
- Verify all required modules are loaded in your script
- Ensure file paths are correct

Out of memory errors:

- Increase `--mem` or `--mem-per-cpu` in your job script
- Check actual memory usage with `sacct -j <job_id> --format=JobID,MaxRSS`

6. Getting Help

Technical Support:

- Email: asa@sines.nust.edu.pk or am.sc@sines.nust.edu.pk
- Our address is:

Supper Computing Lab, Ground Floor, SINES, NUST, Sector H 12, Islamabad Pakistan

SLURM Documentation:

- Official SLURM documentation: https://slurm.schedmd.com/documentation.html
- Quick start guide: https://slurm.schedmd.com/quickstart.html

Useful Resources:

- `man sbatch` - Manual page for sbatch command
- `man srun` - Manual page for srun command
- `man squeue` - Manual page for squeue command