Compare Conda Environments

FAQ

Usage

Errors
**Abstract**

This document is located on the cluster in the ‘/storage’ directory. Another version is located here: gpu.bgu.ac.il

BGU ISE Slurm cluster is a job scheduler and resource manager. It consists of a manger node and several compute nodes.

The manger node is a shared resource used for launching, monitoring and controlling jobs and should **not** be used for computational jobs.

The compute nodes are powerful Linux machines, some of which having GPUs.

The user connects to the manager node and launches jobs that are executed by a compute node.

A job is allocation of compute resources such as RAM memory, cpu cores, gpu, etc. for a limited time. A job may consist of job steps which are tasks within a job.

In the following pages, *italic* writing is saved for Slurm command line commands.

![Diagram of the Slurm cluster setup](image)
Use

Ssh to the Manager Node:   gpu.bgu.ac.il (132.72.44.112)

Use your BGU user name and password to login to the manager node. The default path is to your home directory on the storage.

Python users: create your virtual environment on the manager node.

If you copy files to your home directory, don’t forget about file permissions. E.g. files that need execution permissions, do: chmod +x <path to file>

Remember that the cluster is a shared resource. Currently, users are trusted to act with responsibility in regards to the cluster usage – i.e. release unused allocated resources (with scancel), not allocate more than needed resources, erase unused files and datasets, etc.
Submitting a Job

`sbatch <batch file name>`

- Conda users: Make sure you submit the job while virtual environment deactivated on the manager node (‘conda deactivate’)!
- Jupyter and IDE users: make sure you release the resources, after you were done, by using ‘scancel <job_id>’ (Cancel Jobs).

Jupyter: After submitting the job, open the output file, copy the 2nd token (https://132.72...) and paste it into your web browser’s address bar.

Batch File

```bash
#!/bin/bash

### sbatch config parameters must start with #SBATCH and must precede any other commands. to ignore just add another # - like ##SBATCH

#SBATCH --partition gtx1080    ### specify partition name where to run a job. NVidia 2080: debug: 2 hours; short: 7 days; gtx1080: 7 days
#SBATCH --time 00:01:30:00    ### limit the time of job running. Make sure it is not greater than the partition time limit!! Format: D-H:MM:SS
#SBATCH --job-name my_job     ### name of the job. replace my_job with your desired job name
#SBATCH --output my_job-%j.out ### output log for running job - %j is the job number variable
#SBATCH --mail-user=user@post.bgu.ac.il ### users email for sending job status notifications
#SBATCH --mail-type=BEGIN,END,FAIL ### conditions when to send the email. ALL,BEGIN,END,FAIL, REQUEU, NONE
#SBATCH --gres=gpu:1           ### number of GPUs (can’t exceed 8 gpus for now) ask for more than 1 only if you can parallelize your code for multi GPU

### Print some data to output file ###

echo "SLURM_JOBID"="$SLURM_JOBID"

echo "SLURM_JOB_NODELIST"="$SLURM_JOB_NODELIST"

### Start you code below ####

module load anaconda    ### load anaconda module

source activate my_env    ### activating environment, environment must be configured before running the job

jupyter lab    ### execute jupyter lab command – replace with your own command e.g. ‘python my.py my_arg’
```

The batch file should look like the following:
Example sbatch file location on cluster: /storage/sbatch.example
Information about the compute nodes
sinfo shows cluster information.

sinfo -Nel

NODELIST – name of the node
S:C:T - sockets:cores:threads

List of the User Currently Running Jobs
squeue -l -u$USER

Cancel Jobs
scancel <job id>

scancel --name <job name>

Cancel All Pending jobs for a Specific User
scancel -t PENDING -u <user name>

Running Job Information
Use sstat. Information about consumed memory:

sstat -j <job_id> --format=MaxRSS,MaxVMSize

Complete Job Information
sacct -j <jobid>
sacct -j <jobid> --format=JobID,JobName,MaxRSS,AllocGRES,State,Elapsed,Start

MaxRSS is the maximum memory the job needed.
Advanced Topics

Jupyter Lab
Make the Conda environment available in notebook’s interface:

```
python -m ipykernel install --user --name <conda environment> --display-name "<env name to show in web browser>"
```

High Priority Jobs
Some users have the right to prioritize their jobs, when the required resources for their job are not available. When you give your job high priority, while submitting it, it will preempt another running job or several running jobs.

```
sbatch --qos=<high priority group user name> <batch file name>
```

Job Arrays
Job array feature allows you to run identical version of your script with different environment variables. This is useful for parameter tuning or averaging results over seeds.

To use job array, add the following line to your Slurm batch file:

```
#SBATCH --array=1-10       ### run parallel 10 times
```

Adding this will run your script 10 times in parallel. The environment variable SLURM_ARRAY_TASK_ID for each run will have different values (from 1 to 10 in this case). You can then set different parameter setting for each parallel run based on this environment variable.

Remember to change #SBATCH --output=your_output.out to #SBATCH --output=%a_your_output.out, so the output of each parallel run be directed to a different file. %a will be replaced by the corresponding SLURM_ARRAY_TASK_ID for each run.
Allocate Extra RAM/CPUs

In your sbatch file:

To override default ram:

#SBATCH --mem=72G

To override default cpu:

#SBATCH --cpus-per-task=32

Working with the Compute Node SSD Drive

You may want to use the compute node local drive for fast access to data. /scratch directory on the compute node is intended for that.

Add to the sbatch script file:

    #SBATCH --tmp=100G  ### Asks to allocate enough space on /scratch

Then in users code section:

    export SLURM_SCRATCH_DIR=/scratch/${SLURM_JOB_USER}/${SLURM_JOB_ID}
    cp /storage/*.img $SLURM_SCRATCH_DIR    ### copy data TO node’s local storage
    mkdir $SLURM_SCRATCH_DIR/testtttt
    ...
    some user code
    ...
    cp -r $SLURM_SCRATCH_DIR $SLURM_SUBMIT_DIR    ### copy back final results to user home or other accessible location

When job has finished, canceled or failed, ALL data in $SLURM_SCRATCH_DIR is erased! This temp folder lives with running jobs only!
Job Dependencies

Job dependencies are used to defer the start of a job based on other job’s condition.

```
sbatch --dependency=after:<other_job_id> <sbatch_script>   ### start job after other_job started
```

```
sbatch --dependency=afterok:<other_job_id> <sbatch_script>   ### start job after other_job ends with ok status. E.g. sbatch --dependency=afterok:77:79 my_sbatch_script.sh -> start after both job 77 and 79 have finished
```

```
sbatch --dependency=singleton   ### This job can begin execution after any previously launched jobs, sharing the same job name and user, have terminated
```

pyCharm

Make sure you have pyCharm Professional installed (free for students/academy people).

Create an interactive session:

Ssh to Slurm and copy the script file /storage/pycharm.sh to your working Slurm directory.

You can modify the following lines at the beginning of the file:

```
########################################
# USER MODIFIABLE PARAMETERS:
PART=short     # partition name
TASKS=28       # 28 cores
TIME="2:00:00" # 2 hours
GPU=1          # 1 GPU
QOS=normal     # QOS Name
########################################
```

Run the script by typing: . /pycharm.sh

The output lists the node’s ip address and the job id.
Open pyCharm.

Go to settings->Project->Project Interpreter

On the upper right hand corner next to Project Interpreter: press the settings icon, choose ‘add’

On the left hand side choose SSH Interpreter. Under ‘New server configuration’ fill in the compute node’s ip address and your BGU user name. Click Next. Warning message box click ‘yes’.

Enter your BGU password. Click Next.

In the ‘Interpreter:’ line, enter the path to the desired interpreter. You can find your environments interpreters under /home/<your_user>/.conda/envs/<your_environment>/bin/python.
Click Finish.

Visual Studio Code

Create an interactive session:

Ssh to Slurm and copy the script file /storage/pycharm.sh to your working Slurm directory.

You can modify the following lines at the beginning of the file:

`########################################
# USER MODIFIABLE PARAMETERS:
PART=short  # partition name
TASKS=28    # 28 cores
TIME="2:00:00"  # 2 hours
GPU=1       # 1 GPU
QOS=normal  # QOS Name
########################################`
Run the script by typing:  . /pycharm.sh

The output lists the newly allocated compute node’s ip address and the job id.

Assuming you have VS Code installed and a supported OpenSSH client installed, install the ‘Remote – SSH’ pack.

Install the Python package, if needed.

Press the green button (><) on the most lower left side of the window (under the ‘settings’ button).

On the middle upper side of the window, choose “Remote – SSH: Connect to host...” and enter <your_BGU_user>@<compute_node_ip_address>

A new window opens. Enter your BGU password, when prompted.
Ctrl+Shift+P for the Command Palette then choose ‘Python: Select Interpreter’ (start typing – it will show up) and choose the interpreter from your desired environment (~/.conda/envs/<environment>/bin/python).

Fiji – Image Analysis Tool

You can read about Fiji here: [https://fiji.sc/](https://fiji.sc/)

Run Fiji on the cluster

`srun --x11 --gpus=1 --partition=gtx1080 /storage/apps/Fiji/ImageJ-linux64`
Appendix

Conda

Viewing a list of your environments
conda env list

list of all packages installed in a specific environment
conda list

to see a not activated environment
conda list -n <my_env>

Activating / deactivating environment
source activate <my_env>
or (depends on conda version)
conda activate <my_env>
conda deactivate

Create Environment
conda create -n <my_env>

with specific python version
conda create -n <my_env> python=3.4

with specific package (e.g. scipy)
conda create -n <my_env> scipy
Or
conda create -n <my_env> python
conda install -n <my_env> scipy

with specific package version
conda create -n <my_env> scipy=0.15.0

with multiple packages
conda create -n <my_env> python=3.4 scipy=0.15.0 astroid babel

Update Conda
conda update conda

Compare Conda Environments
The following python (2) script compares 2 conda environments and can be found in ‘/storage’ directory.

python conda_compare.py <environment1> <environment2>
FAQ

Usage
  ❖ Can I ssh the cluster when I am away from university?
    This can be done by using VPN.

Errors
  ❖ RuntimeError: CUDA out of memory. Tried to allocate 448.00 MiB (GPU 0; 10.73 GiB total capacity; 9.64 GiB already allocated; 124.69 MiB free; 195.47 MiB cached)

    The problem arises when trying to allocate more memory than available (10.73GiB). Try to reduce the size of the batch in order to fit the GPU’s available memory.

    Use: pip install wget