INSTITUT DU
DÉVELOPPEMENT ET DES
RESSOURCES EN
INFORMATIQUE
SCIENTIFIQUE

www.idris.fr

Get started on Jean Zay

IDRIS User Support Team - May 2022

Au programme

Connect to the supercomputer
Connection to Jean Zay

Project environment
Account and projects
Compute hours billing
Disk spaces

Compute environment
HPC and AI modules
Conda environments

Using resources
Submit jobs with Slurm
Jupyter Notebooks

Connect to the supercomputer

Connection to Jean Zay

To connect to Jean Zay you will need an SSH client on a machine with a fixed IP address declared (IDRIS filters) and attached to your account: 

```
$ ssh login@jean-zay.idris.fr
Last login: Fri Nov 26 16:59:58 2021 from 130.84.15.36
*************************************************************************
* Ceci est un serveur de calcul de l'IDRIS. Tout acces au systeme       *
* doit etre specifiquement autorise par l'IDRIS. Si vous tentez de      *
* de continuer a acceder cette machine alors que vous n'y etes pas      *
* autorise, vous vous exposez a des poursuites judiciaires.             *
*                               ---                                     *
* This is an IDRIS compute node.  Each access to this system must be    *
* properly authorized by IDRIS. If you go on accessing this machine     *
* without authorization, then you are liable to prosecution.            *
*************************************************************************
*                                                                       *
* Orsay      CNRS / IDRIS - Frontale  -  jean-zay.idris.fr   France     *
*                                                                       *
*************************************************************************
login@jean-zay3:~$
```

IDRIS only supports *bash* shell (~/.bash_profile, ~/.bashrc).

For Windows users there are several options such as PuTTY, MobaXTerm, etc.

Connection through a bounce server (ProxyJump)

To make your life easier when you need to connect through a bounce server you can edit your local `$HOME/.ssh/config`:

```
host jean-zay jz
  hostname jean-zay.idris.fr
  user <your login>
  proxyjump <login_machine_de_rebond>@<machine_de_rebond>
```

It allows the connection to Jean Zay with the command `ssh jean-zay`.

Connection to Jean Zay (continued)

You will be connected to one of the 5 **login node**: 

```
$ hostname
jean-zay[1-5]
```

- Login nodes are shared with other users:
- They are dedicated to setting up your environment (compilation, file transfers, . . .)
- They have an HTTP proxy to allow data transfers from distant servers (*git*, *wget*, . . .)
- They DO NOT have GPUs
- We tolerate the execution of small scripts (pre or post processing) on login nodes. Heavier computation are forbidden to avoid overloading the nodes.
- We limit resources available to 1 core, 5 GB of RAM and 30 minutes of CPU time (Elapsed time can be longer).

Project environment

Account and projects

Reminder:

- **1 user = 1 account** ...
- ... attached to a **project** with allocated hours
- A project may have several users.
- A user may belong to several projects.

1 login = 1 user

To know the projects you belong to: 

```
$ idrproj
Available projects:
--------------------
  abc (123456)
  def (167890) [default][active]
```

Compute hours billing

We define the number of hours h used by a job by:

- on a CPU partition : $h =$ CPU cores allocated $\times$ *elapsed* time
- on a GPU partition : $h =$ GPUs allocated $\times$ *elapsed* time

The RAM available for your job depends on the resources allocated. On the CPU and 4-GPUs partitions:

4GB per core (160GB / 40 cores CPU)

We bill all the cores on nodes when Slurm's `--exclusive` option is set or when you allocate more than one node.

Compute hours billing (continued)

To know how many hours you used: 

```
$ idracct
Derniere mise a jour le 01-12-2021 08:00:01

###############################################################################
                          PROJET 12345 SUR JEAN-ZAY
###############################################################################

===============================================================================
 GPU (du 15-11-2021 au 31-12-2022)                      Allocation : 1000 h.gpu
===============================================================================

  Compte   Proprietaire                  Consommation Nb travaux Remboursement
  -------- ----------------------------- ------------ ---------- -------------
  login1   ALPHABET Claude                      100.00         16             -
  login2   FARFADET Camille                      50.00          9             -
  -------- ----------------------------- ------------ ---------- -------------
                                 Totaux        150.00         25          0.00
                             Utilisation        15.00%
```

Disk spaces

Jean Zay have 4 main disk spaces.

→ Their intended usage depends on the storage capacity (Volume and inodes) and their technical specificities (memory access, temporality, . . .).

Space	Default Capacity	Specifities	Usage
`$HOME`	3 GB / 150k *inodes* per user	· Saved	· Storage for configuration and small files
`$WORK`	5 TB / 500k *inodes* per project (*)	· Saved · Storage on hard drive (100 GB/s write/read)	. Storage sources and input/output files . Batch or interactive execution
`$SCRATCH`	Large quotas (10% of total space and 150M inodes) per project 1,3 PB shared by all users	· **Not saved** · SSD storage (300 GB/s read/write) · Unused files lifetime: **30 days** (unused = not read or modified)	· Storage for large input/output data . Batch or interactive execution . Best performance for IO
`$STORE`	50 TB/100k *inodes* per project (*)	· **Not saved**	· Long term storage for archives (lifetime of the project)

(*) Quotas of "project" spaces can be increased on IDRIS extranet.

Disk spaces (continued)

Information about disk usage is available with: 

```
$ idrquota -h
```

More details with: `idr_quota_user` and `idr_quota_project`.

By default, you are the only one with access rights on files stored on your disk spaces. We provide shared folders for each project on 3 disk spaces:

- on `$WORK` : `$ALL_CCFRWORK`
- on `$SCRATCH` : `$ALL_CCFRSCRATCH`
- on `$STORE` : `$ALL_CCFRSTORE`

We have a special disk space for large public databases. User support can download it for you on `$DSDIR`.

- It is available for all users.

Disk spaces (continued)

You will find a python script and a submission script in the common WORK directory. Copy it in you personal WORK directory for the rest of the presentation:

```
$ cp -r $ALL_CCFRWORK/TP_idris $WORK/.
```

Compute environment

HPC modules

You can access several compilers, libraries and scientific software with the `module` command.

To get the complete list use `module avail`: 

```
$ module avail
---------------------------- /path/to/modules -----------------------------
 arm-forge/19.1.1             intel-all/2019.5(19.0.5)   intel-itac/2020.0         intel-tbb/2018.6(18.0.5)
 arm-forge/20.1.2             intel-all/2020.0           intel-itac/2020.1         intel-tbb/2019.2(19.0.2)
 arm-forge/20.2.1             intel-all/2020.1           intel-itac/2020.2         intel-tbb/2019.6(19.0.4)
 arm-forge/21.1.0             intel-all/2020.2           intel-itac/2020.3         intel-tbb/2019.8(19.0.5)
 ccfr/1.0                     intel-all/2020.4           intel-mkl/11.3.4(16.0.4)  intel-tbb/2020.0
 cuda/9.2                     intel-compilers/16.0.4     intel-mkl/2018.1(18.0.1)  intel-tbb/2020.2
 cuda/10.0                    intel-compilers/18.0.1     intel-mkl/2018.4(18.0.5)  intel-tbb/2020.3
 cuda/10.1.1                  intel-compilers/18.0.5     intel-mkl/2019.2(19.0.2)  intel-vtune/2018.1(18.0.1)
 cuda/10.1.2                  intel-compilers/19.0.2     intel-mkl/2019.4(19.0.4)  intel-vtune/2018.4(18.0.5)
 . . .
```

Looking for a specific package? Use `module avail <package>` : 

```
$ module avail fftw
---------------------------- /path/to/modules -----------------------------
fftw/2.1.5-mpi  fftw/3.3.8  fftw/3.3.8-mpi  fftw/3.3.8-mpi-cuda  fftw/3.3.10-mpi  fftw/3.3.10-mpi-omp
```

AI modules

The main AI frameworks (TensorFlow, PyTorch, ...) are available as preset conda environments: 

```
$ module avail tensorflow pytorch
---------------------------- /path/to/modules ----------------------------
tensorflow-gpu/py2/1.4              tensorflow-gpu/py3/1.14-deepmd     tensorflow-gpu/py3/2.3.1+hvd-0.21.0
tensorflow-gpu/py2/1.8              tensorflow-gpu/py3/1.14-mpi        tensorflow-gpu/py3/2.4.0
tensorflow-gpu/py2/1.13             tensorflow-gpu/py3/1.14-openmpi    tensorflow-gpu/py3/2.4.0-noMKL
tensorflow-gpu/py2/1.14             tensorflow-gpu/py3/1.15.2          tensorflow-gpu/py3/2.4.1
. . .
---------------------------- /path/to/modules ----------------------------
pytorch-cpu/py3/1.4.0  pytorch-gpu/py3/1.3.0+nccl-2.5.6  pytorch-gpu/py3/1.7.0                    pytorch-gpu/py3/1.8.0
pytorch-cpu/py3/1.7.1  pytorch-gpu/py3/1.3.1             pytorch-gpu/py3/1.7.0+hvd-0.21.0         pytorch-gpu/py3/1.8.1
pytorch-gpu/py2/0.4.1  pytorch-gpu/py3/1.3.1+nccl-2.5.6  pytorch-gpu/py3/1.7.0-rc1                pytorch-gpu/py3/1.9.0
pytorch-gpu/py2/1.1    pytorch-gpu/py3/1.4.0             pytorch-gpu/py3/1.7.0-rc2                pytorch-gpu/py3/1.10.0
. . .
```

Once the module is loaded, the environment is activated and the Python packages installed are visible with `pip list` or `conda list`.

Loading modules

When a module is loaded, its dependencies are automatically loaded.

You can check which modules are loaded with `module list`.
Here is an example with fftw: 

```
$ module avail fftw
---------------------------- /path/to/modules -----------------------------
fftw/2.1.5-mpi  fftw/3.3.8  fftw/3.3.8-mpi  fftw/3.3.8-mpi-cuda  fftw/3.3.10-mpi  fftw/3.3.10-mpi-omp
$ module load fftw/3.3.8-mpi
Loading fftw/3.3.8-mpi
  Loading requirement: intel-compilers/19.1.3 intel-mpi/2019.9
$ module list
Currently Loaded Modulefiles:
 1) intel-compilers/19.1.3   2) intel-mpi/2019.9   3) fftw/3.3.8-mpi
```

Loading modules (continued)

A module may cover several compiling environments.

The list is available with `module show <package>/<version>`: 

```
$ module show fftw/3.3.8-mpi
-------------------------------------------------------------------
. . .
Available software environment(s):
- intel-compilers/19.1.3 intel-mpi/2019.9
- intel-compilers/19.1.3 openmpi/4.1.1
- intel-compilers/19.1.3 openmpi/4.1.0
- intel-compilers/19.1.3 openmpi/4.0.5
- intel-compilers/19.1.1 openmpi/4.1.1
. . .
```

You have to load first the modules of the desired environment: 

```
$ module load intel-compilers/19.1.3 openmpi/4.0.5
$ module load fftw/3.3.8-mpi
$ module list
Currently Loaded Modulefiles:
 1) intel-compilers/19.1.3   2) openmpi/4.0.5   3) fftw/3.3.8-mpi
```

Information on modules

The command `module show <package>` have a different behavior whether the module is loaded or not.

Not loaded: 

```
$ module purge # unload all modules
$ module show fftw/3.3.8-mpi
-------------------------------------------------------------------
/path/to/modules/fftw/3.3.8-mpi:

module-whatis   FFTW is a C subroutine library for computing the discrete Fourier transform (DFT) in one or more dimensions, of a
prereq          intel-compilers/19.1.3 intel-compilers/19.1.1 intel-compilers/19.0.4 gcc/9.1.0 gcc/8.3.1
conflict        fftw

Available software environment(s):
- intel-compilers/19.1.3 intel-mpi/2019.9
- intel-compilers/19.1.3 openmpi/4.1.1
. . .

If you want to use this module with another software environment,
please contact the support team.
-------------------------------------------------------------------
```

Information on modules (continued)

The command `module show <package>` have a different behavior whether the module is loaded or not.

Loaded: 

```
$ module load fftw/3.3.8-mpi
$ module show fftw/3.3.8-mpi
-------------------------------------------------------------------
/path/to/modules/fftw/3.3.8-mpi:

module-whatis   {FFTW is a C subroutine library for computing the discrete Fourier transform (DFT) ...}
prereq          intel-compilers/19.1.3 intel-compilers/19.1.1 intel-compilers/19.0.4 gcc/9.1.0 gcc/8.3.1
prereq          intel-mpi/2019.9 openmpi/4.1.1 openmpi/4.1.0 openmpi/4.0.5
conflict        fftw
prepend-path    CPATH /path/to/fftw/3.3.8/include
prepend-path    LD_LIBRARY_PATH /path/to/fftw/3.3.8/lib
prepend-path    LIBRARY_PATH /path/to/fftw/3.3.8/lib
prepend-path    PATH /path/to/fftw/3.3.8/bin
prepend-path    MANPATH /path/to/fftw/3.3.8/share/man
prepend-path    PKG_CONFIG_PATH /path/to/fftw/3.3.8/lib/pkgconfig
prepend-path    CMAKE_PREFIX_PATH /path/to/fftw/3.3.8/

. . .
If you want to use this module with another software environment,
please contact the support team.
-------------------------------------------------------------------
```

Module sub-commands

Reminder of the sub-commands:

- List all available packages: ..... `module avail`
- List the versions of *package*: ..... `module avail <package>`
- Load *package/version* : ..... `module load <package/version>`
- Print information about *package/version* : ..... `module show <package/version>`

Other sub-commands:

- Unload all modules: ..... `module purge`
  Very useful at the begining of batch scripts.
- Unload a package+dependencies automatically: ..... `module unload <package>`
- Switch the version of *package* v1 → v2 : ..... `module switch <package/v1> <package/v2>`
  Dependencies are modified consequently.

Conda environments

You can use *conda* enviroments for the main AI frameworks (MXNet, PyTorch, TensorFlow) with the `module` command.

The support team can add new features on demand (email assist@idris.fr).

We recommend their use not to fill your disk spaces.

Conda environments (continued)

In practice you might need to:

- add a package with pip: ..... `pip install --user --no-cache-dir <paquet>`
  (version installed in the module not suitable, dev, ...)
- create your own conda environments

Be careful not to fill your disk spaces. You want at least to move ~/.conda and ~/.local in WORK:

```
$ mv $HOME/.local $HOME/.conda $WORK
$ ln -s $WORK/.local $HOME
$ ln -s $WORK/.conda $HOME
```

Please check our documentation on IDRIS website.

Using resources

Slurm queueing system

Jobs run on the compute nodes of Jean Zay.

They can be submitted with a batch script or interactively.

When a job is submitted, it is placed in a queue.

Slurm manages the queue for the jobs submitted by all users.

To ensure a fair sharing of resources among all users a priority depending on several parameters is assigned to the job. (see IDRIS doc).

Slurm partitions

When you submit a job you need to assign it to a **Slurm partition** to specify the kind of nodes you want.

There are several kinds of partitions on Jean Zay:

- `cpu_p1` $\longrightarrow$ **scalar** nodes (CPU)
- gpu_p13 $\longrightarrow$ **4-GPU accelerated** nodes (default GPU partition)
  - `gpu_p13` (GPU 32 GB or 16 GB)
- `gpu_p2` $\longrightarrow$ **8-GPU accelerated** nodes (dedicated to AI community)
- `gpu_p4` $\longrightarrow$ **8-GPU A100 accelerated** nodes (test nodes)

QoS

You can also specify a **Quality of service** (***QoS***) *for the partition to finely tune the resources you need (number of GPU, time limit, . . .).*

The QoS *modifies the priority of your job.*

Each partition have 3 QoS*:*

Partition	QoS	*Time limit*	*Resources limit*			
			per job	*per user*	*per project*	*per QoS*
CPU	`qos_cpu-t3` *(*)*	*20 h*	*20480 cores*	*48000 cores*	*48000 cores*	
	`qos_cpu-t4`	*100 h*	*160 cores*	*1280 cores*	*1280 cores*	*5120 cores*
	`qos_cpu-dev`	*2 h*	*5120 cores*	*5120 cores*	*5120 cores*	*48000 cores*
GPU	`qos_gpu-t3` *(*)*	*20 h*	*512 GPU*	*1024 GPU*	*1024 GPU*	
	`qos_gpu-t4`	*100 h*	*16 GPU*	*128 GPU*	*128 GPU*	*512 GPU*
	`qos_gpu-dev`	*2 h*	*32 GPU*	*32 GPU*	*32 GPU*	*512 GPU*

(*) default QoS*.*

Batch submission script

A batch script have:

- a job configuration header (job name, resources required, ...) as a list of Slurm options prepended with `#SBATCH`.
- a body with the command lines to run (loading of modules, launching of executables, . . .).

In the submission script, executable are launched with **`srun`**. The command takes into account the parameters given in the header.

Job submission - Batch script (continued)

Example to execute a job on the **4-GPU** accelerated partition:

- Using a complete node of the default partition gpu_p13 $\longrightarrow$ 4 GPUs.

```
#!/bin/bash
#SBATCH --job-name=JobGPU           # Job name
#SBATCH --output=JobGPU%j.out       # Standard output file (%j = job ID)
#SBATCH --error=JobGPU%j.err        # Standard error file (%j = job ID)
#SBATCH --nodes=1                   # 1 node
#SBATCH --ntasks=4                  # 4 tasks (or process)
#SBATCH --gres=gpu:4                # 4 GPU/node
#SBATCH --cpus-per-task=10          # 10 cores per task (with the memory)
#SBATCH --hint=nomultithread        # disable hyperthreading
#SBATCH --time=00:10:00             # time limit "(HH:MM:SS)"
#SBATCH --qos=qos_gpu-dev           # QoS

module purge                        # clean the modules inherited from the submission
environment
conda deactivate                    # deactivate conda environments inherited from the
submission environment
module load pytorch-gpu/py3/1.7.0   # Load modules

set -x                              # activate echo of commands
srun python script.py               # script execution
```

Batch submission script (continued)

Slurm commands:

- Submit a batch script: ..... `sbatch <script>`
- Status of your submitted jobs: ..... `squeue -u $USER`

  → Possible states: R = *running*, PD = *pending*, CG = *completing*
- Display information about a running job:

```
$ scontrol show job <jobid>
```

- Cancel a job: ..... `scancel <jobid>`

Once a node is allocated for you, it is possible to connect to it to monitor your executions (top, htop, nvidia-smi, ...):

```
$ ssh <node ID>
```

Batch submission script (continued)

Hands-on: 
Submit the Slurm script "batch.slurm"

Interactive session

- You can open a terminal directly on a node allocated to you.
- You can perform some tests interactively (with access to GPUs) to configure your batch scripts.
- Example: interactive submission on the partition gpu_p2 :

```
$ srun --pty --partition=gpu_p2 --ntasks=1 --gres=gpu:1 --<other-options> bash
```

- To exit interactive mode: ..... `exit`
- **Attention**, it is not possible to use MPI with this configuration, therefore interactive multinode is impossible.
- Other execution modes are described on IDRIS doc.

Jupyter Notebooks - Connection through a bounce server

If you connect via a bounce server you can use port forwarding to get better performance.

```
$ ssh -N -L 10443:idrvprox.idris.fr:443 <login bounce server>@<bounce server>
```

A security warning might happen, you can accept it.
Then open the page `http://localhost:10443/` on a browser.

The identification phases are described further in the document.

Jupyter Notebooks - Launch the server

You can visualize your notebooks on Jean Zay.
IDRIS provides two commands to launch Jupyter servers:

- `$ idrjup`: for `jupyter notebook`
- `$ idrlab`: for `jupyter lab`

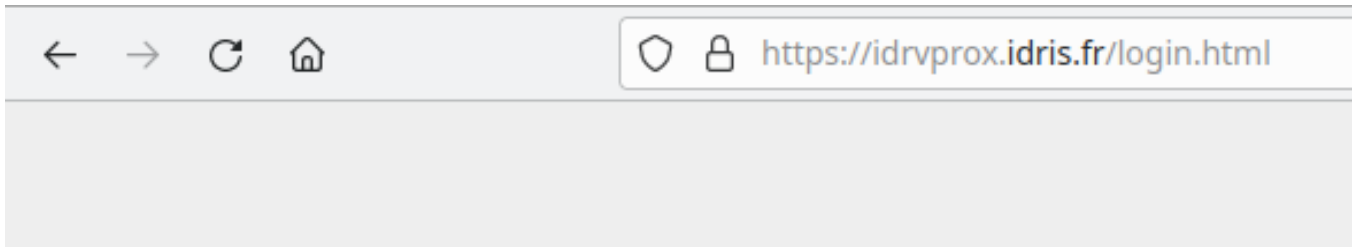
The options of the original commands are accepted.
A password is provided by the command. It will be useful during the second identification phase.

```
$ idrjup --notebook-dir=$PWD
INFO 2019-11-22 12:10:08,916 Starting Jupyter server. Please wait...:
INFO 2019-11-22 12:10:08,933 --Launching Jupyter server. Please wait before attempting to connect...
INFO 2019-11-22 12:10:14,070 --Jupyter server launched. Please connect.
URL de connexion :     https://idrvprox.idris.fr
Mot de passe URL :     <mot de passe utilisateur>
Mot de passe jupyter : abc1defg2hijk3lmno4pqrs5tuvw6xyz
```

The default path for notebooks is HOME. You should provide the option `--notebook-dir=$PWD` to change it.

Jupyter Notebooks - Connection from a registered server

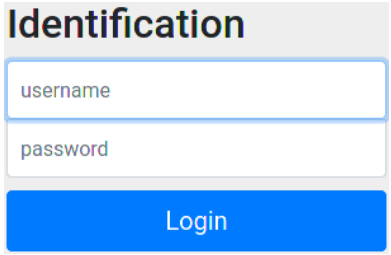
If you connect with a registered server or VPN you can directly open the URL given by the command:



The identification phases are described further in the document.

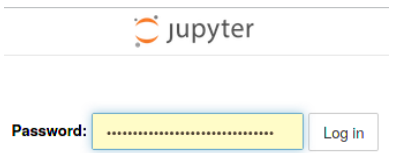
Jupyter Notebooks - Identifications

A first identification is required. You must provides your credentials on Jean Zay:



Choose your session:

The second identification phase requires the password generated by the command `idrlab` or `idrjup`.



IDRIS Documentation

IDRIS new user documentation : http://www.idris.fr/eng/su/debutant-eng.html