INSTITUT DU
DÉVELOPPEMENT ET DES
RESSOURCES EN
INFORMATIQUE
SCIENTIFIQUE
www.idris.fr

Débuter sur Jean Zay

Recherche et application en Intelligence Artificielle



Équipe Support aux Utilisateurs de l'IDRIS - Octobre 2024

Pourquoi cette documentation ?

- Cette présentation vise à guider les nouveaux utilisateurs IA de Jean Zay.
- L'ensemble des informations délivrées ici est voulu synthétique pour assurer une prise en main rapide du supercalculateur.
- Une documentation complète est mise à jour régulièrement par l'équipe Support aux Utilisateurs sur le site de l'IDRIS. ➜
- Une FAQ est disponible en ligne. ➜

NB : dans cette présentation, les symboles ➜ représentent des liens hypertexte.

Au programme

Principales caractéristiques de Jean Zay

Environnement administratif

Accès aux ressources de calcul

Environnement machine

Environnement de calcul

Soumission de travaux

Consommation des heures de calcul

Pour aller plus loin

Principales caractéristiques de Jean Zay

Principales caractéristiques de Jean Zay

- Jean Zay est un calculateur composé de deux partitions :
  - une partition scalaire ou « CPU » contenant 28800 cœurs de calcul
  - une partition accélérée ou « GPU » contenant 3704 GPU distribués sur des architectures de nœuds variées

Partitions	#nœuds	RAM CPU	#cœurs/nœud	#GPU/nœud	RAM GPU
CPU	720	192 Go	40 cœurs$^{\otimes}$		
GPU	126	192 Go	40 cœurs$^{\otimes}$	4 V100$^{\triangleright}$	16 Go
	270	192 Go	40 cœurs$^{\otimes}$	4 V100$^{\triangleright}$	32 Go
	20	384 Go	24 cœurs$^{\ominus}$	8 V100$^{\triangleright}$	32 Go
	11	768 Go	24 cœurs$^{\ominus}$	8 V100$^{\triangleright}$	32 Go
	52	512 Go	64 cœurs$^{\oslash}$	8 A100$^{\blacktriangleright}$	80 Go
	364	512 Go	96 cœurs$^{\oplus}$	4 H100$^{\triangleright}$	80 Go

($\otimes$) 2 processeurs Intel Cascade Lake 6248 (2×20 cœurs à 2,5 GHz)
($\ominus$) 2 processeurs Intel Cascade Lake 6226 (2×12 cœurs à 2,7 GHz)
($\oslash$) 2 processeurs AMD Milan EPYC 7543 (2×32 cœurs à 2,8 GHz)
($\oplus$) 2 processeurs Intel Sapphire Rapids 8468 (2×48 cœurs à 2,1 GHz)

($\triangleright$) Nvidia V100 SXM2 HBM2
($\blacktriangleright$) Nvidia A100 SXM4 HBM2e
($\triangleright$) Nvidia H100 SXM5 HBM3

Principales caractéristiques de Jean Zay

- Puissance crête cumulée : 126 Pflops/s
- Réseaux d'interconnexion OmniPath (100 Gb/s) et InfiniBand (400 Gb/s)
- Système de fichiers parallèle Lustre
- Trois technologies de stockage :
  - → 7.6 Po sur disques SSD Full Flash (1,5 To/s en lecture et 1,1 To/s en écriture)
  - → 39 Po sur disques rotatifs (350 Go/s en lecture et 300 Go/s en écriture)
  - → 50 Po sur bandes magnétiques

- 5 nœuds frontaux
- 4 nœuds de pré et post-traitement
- 5 nœuds de visualisation

Une description matérielle complète est disponible en ligne. ➜

Environnement administratif

Environnement administratif

GENCI

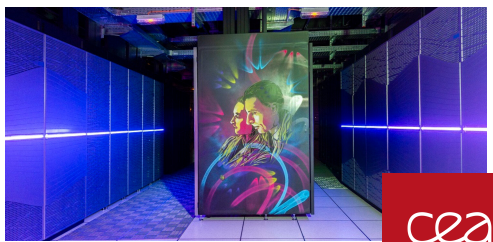
Société civile coordinatrice des trois grands centres de calcul nationaux ➜

Calculateur hébergé à l'IDRIS



Jean Zay ➜

Calculateur hébergé au TGCC



Joliot-Curie ➜

Calculateur hébergé au CINES



Adastra ➜

Liberté • Égalité • Fraternité
RÉPUBLIQUE FRANÇAISE

MINISTÈRE
DE L'ENSEIGNEMENT
SUPÉRIEUR,
DE LA RECHERCHE
ET DE L'INNOVATION

Accès aux ressources de calcul

Accès aux ressources de calcul

- Pour calculer sur Jean Zay, il faut :
  - un **projet**, i.e., une allocation d'heures de calcul (procédure gérée par le **GENCI**)
    - → description du projet scientifique
    - → estimation du nombre d'heures de calcul nécessaires
    - → détails des librairies de calcul dont vous aurez besoin
    - → etc
  - un **compte de calcul** (procédure gérée par l'**IDRIS**)
    - → demande d'ouverture de compte
    - → déclaration des adresses IP de connexion
    - → déclaration du responsable sécurité informatique de votre laboratoire
    - → etc

- Ces deux procédures se font via le portail **DARI** du GENCI.
  Vous devez donc d'abord créer un compte utilisateur sur ce portail. ➜

- Une documentation détaillée de ces procédures est disponible en ligne. ➜

Accès aux ressources de calcul - Projet

- Les demandes d'heures de calcul sont gérées par le GENCI (acces@genci.fr).
- Les modalités d'accès sont détaillées en ligne. ➜
- Il existe deux types d'accès aux ressources selon le nombre d'heures demandées.

Accès Réguliers (AR)	**Accès Dynamiques (AD)**
$\geq$ 500k heures CPU $\geq$ 50k heures GPU normalisées*	$<$ 500k heures CPU $<$ 50k heures GPU normalisées*
Deux sessions d'allocation par an : - en mai (dépôt du dossier avant fév.) - en nov. (dépôt du dossier avant sept.)	Ouverts toute l'année : la validation d'un AD prend quelques jours
Expertises technique et scientifique	Pas d'expertise

(*) 50k heures V100 ou 25k heures A100 ou 12,5k heures H100

Accès aux ressources de calcul - Projet

- Il est possible de demander à la fois des heures CPU, GPU V100, GPU A100 et/ou GPU H100.
- Une allocation est valable **un an** à partir de l'ouverture du projet.
- Un projet peut être **renouvelé** au bout d'un an via le portail DARI.
- Besoin d'**heures complémentaires** ?
  - Pour les Accès Réguliers, des demandes d'**heures complémentaires** peuvent être faites à mi-parcours, lors d'une des deux sessions d'allocation annuelles.
  - Pour des demandes ponctuelles d'heures complémentaires, des "demandes au fil de l'eau" exceptionnelles sont déposables tout au long de l'année sur le portail DARI. Ces heures sont attribuées si la charge de la machine le permet.

Accès aux ressources de calcul - Compte de calcul

- Les ouvertures de comptes de calcul sont gérées par l'IDRIS (gestutil@idris.fr).

- La création d'un compte de calcul prend quelques jours.
  - Attention, l'IDRIS est une Zone à Régime Restrictif et certains utilisateurs peuvent être soumis à une enquête du ministère (compter 8 semaines maximum).

- Un compte de calcul peut être rattaché à plusieurs projets.

- Vous pouvez apporter des modifications à votre compte de calcul (ajout d'une adresse IP de connexion, détachement d'un projet,...) à n'importe quel moment via le Formulaire de Gestion de Compte (FGC). ➜

Environnement machine

Environnement machine - Connexion

- La connexion à Jean Zay se fait en SSH :

```
$ ssh login@jean-zay.idris.fr
```

- La connexion doit être initiée depuis une adresse IP fixe et institutionnelle. Celle-ci doit être déclarée et associée à votre compte de calcul lors de son ouverture ou via le Formulaire de Gestion de Compte. ➜

- Si vous travaillez sous Windows, la connexion peut se faire via un client SSH (PuTTY, MobaXterm, Ubuntu,...).

Environnement machine - Connexion

- Vous arrivez sur un des 5 nœuds de connexion, ou **frontales**, de Jean Zay :

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

  - Ces nœuds sont partagés par l'ensemble des utilisateurs.
  - Ils sont dédiés à la mise en place de l'environnement de calcul (compilation, transferts de données,...).
  - Des limitations en temps d'exécution et en mémoire sont imposées aux scripts tournant sur ces nœuds pour éviter des problèmes de surcharge.
  - Ces nœuds disposent d'un proxy HTTP (contrairement aux nœuds de calcul). Vous pouvez télécharger des données depuis des serveurs distants (avec *git* ou *wget* par exemple).
  - Ils ne sont pas équipés de GPU.

- *bash* est l'interpréteur de commandes maintenu à l'IDRIS. ➜

Environnement machine - Espaces disques

- Il existe 4 espaces disques majeurs sur Jean Zay.

  → Leurs usages sont définis en fonction de leurs capacités de stockage (en Go et nombre d'*inodes* – ou fichiers) et de leurs spécificités techniques (temporalité, accès mémoire,...).

Espace	Capacité par défaut	Spécificité	Usage
`$HOME`	3 Go / 150k *inodes* par utilisateur	· Accueil de connexion	· Stockage de fichiers de configuration et de petits fichiers
`$WORK`	5 To* / 500k *inodes* par projet	· Stockage sur disques rotatifs (350 Go/s en lecture et 300 Go/s en écriture)	· Stockage des sources et données d'entrée/sortie · Exécution en batch ou interactif
`$SCRATCH`	Quotas de sécurité très larges 4,6 Po partagés par tous les utilisateurs	· Stockage SSD (1,5 To/s en lecture et 1,1 To/s en écriture) · Durée de vie des fichiers inutilisés : **30 jours** (inutilisés = non lus ou modifiés)	· Stockage des données d'entrée/sortie volumineuses · Exécution en batch ou interactif · Performances optimales pour les opérations de lecture/écriture
`$STORE`	50 To* / 100k *inodes** par projet	· Espace sauvegardé sur bandes magnétiques, avec un cache sur disques rotatifs	· Stockage d'archives sur le long terme (durée de vie du projet)

(*) Les quotas des espaces « projet » peuvent être augmentés sur demande via l'extranet. ➜

Environnement machine - Espaces disques

- Pour consulter vos quotas disques :
  - Vision utilisateur : ..... `$ idr_quota_user`
  - Vision projet : ..... `$ idr_quota_project`

- Par défaut, vos espaces disques sont cloisonnés : vous seul avez les droits d'accès sur les fichiers qu'ils contiennent.
- Pour partager des fichiers avec les membres de votre projet, il existe trois espaces communs :
  - dans le `$WORK` : `$ALL_CCFRWORK`
  - dans le `$SCRATCH` : `$ALL_CCFRSCRATCH`
  - dans le `$STORE` : `$ALL_CCFRSTORE`

- Si vous travaillez avec des bases de données publiques volumineuses, l'IDRIS peut les installer pour vous dans l'espace disque `$DSDIR`. ➜
  - Cet espace est accessible en lecture à tous les utilisateurs.
  - Cela permet de mutualiser les ressources et ne pas saturer vos espaces disques.

Environnement de calcul

Environnement de calcul - JupyterHub

- Les équipes de l'IDRIS ont mis en place JupyterHub, une solution permettant d'utiliser les Jupyter Notebooks et d'autres applications comme VSCode, MLflow ou Dask via une interface web sans connexion SSH préalable à la machine. ➜



Environnement de calcul - Modules

- L'IDRIS met à disposition un catalogue d'outils (environnements virtuels, librairies compilées,...) accessibles via la commande **module**.

- La liste des modules peut être enrichie sur demande (contacter assist@idris.fr).

- **Attention**, pour accéder aux modules adaptés à la partition A100 ou H100, il faut charger au préalable :
  - Pour la partition A100 : `module load arch/a100`
  - Pour la partition H100 : `module load arch/h100`

Environnement de calcul - Commandes module de base

- Pour afficher le catalogue complet : ..... `module avail`
- Pour rechercher un outil précis : ..... `module avail <package>`
- Pour obtenir des infos sur un module : ..... `module show <package>`
- Pour charger un module : ..... `module load <package>/<version>`
- Pour décharger un module : ..... `module unload <package>`

- Pour afficher la liste des modules chargés : ..... `module list`
- Pour repartir d'un environnement vierge : ..... `module purge`

Environnement de calcul - Environnements virtuels

- Les logiciels pour l'Intelligence Artificielle sont installés pour Python 3 dans des **environnements virtuels conda**.
- Les environnement virtuels sont accessibles via la commande **module**.
  → L'environnements est activé (`conda activate`) lors du chargement du module.
  → Il n'est **pas** désactivé (`conda deactivate`) lorsque le module est déchargé.
- Chaque environnement s'articule autour d'une de ces quatre librairies majeures : TensorFlow, PyTorch, MXNet ou Caffe.
- Pour visualiser l'ensemble des environnements disponibles :

```
module avail tensorflow pytorch mxnet caffe
```

- Une fois l'environnement activé, vous pouvez visualiser l'ensemble des paquets Python qu'il contient grâce aux commandes `pip list` et `conda list`.
  → Tout environnement peut être enrichi sur demande (contacter assist@idris.fr).

Environnement de calcul - Installations personnelles

- Il est fortement conseillé d'utiliser les environnements installés par nos soins pour obtenir les meilleures performances, mutualiser les ressources et éviter de saturer vos quotas.

- Pour des besoins spécifiques, vous pouvez faire des installations personnelles :
  - en enrichissant un environnement existant :

```
$ module load <env>
$ pip install --user --no-cache-dir <paquet>
```

  - en créant votre propre environnement conda :

```
$ module load miniforge/24.9.0
$ conda create -n myenv
```

- Les avantages et inconvénients des installations personnelles sont détaillés dans la documentation en ligne. ➜

Soumission de travaux

Soumission de travaux - Système de file d'attente Slurm

- Les travaux s'exécutent sur les nœuds de calcul de Jean Zay.
- Pour avoir accès à un ou plusieurs nœuds de calcul, il faut soumettre une demande d'allocation de ressources.
- La file d'attente pour l'accès aux ressources de calcul est gérée par le gestionnaire **Slurm** pour l'ensemble des utilisateurs.
- Un système de priorité est mis en place pour garantir un partage des ressources le plus équitable possible. ➜

Soumission de travaux - Partitions Slurm

- Lorsque vous réservez des ressources de calcul, vous devez indiquer à Slurm la **partition** visée (i.e., le type de nœuds).
  - Une description matérielle détaillée est disponible en ligne ➜
- La partition **CPU** est sélectionnée par défaut si vous ne réservez pas de GPU.
  - Cette partition est accessible si vous avez fait une demande d'heures CPU.
- La partition accélérée **quadri-GPU V100** est sélectionnée par défaut si vous réservez des GPU.
  - Cette partition contient à la fois des GPU à 16 Go et 32 Go de mémoire.
- **Attention**, les comptabilités d'heures GPU V100, A100 et H100 sont distinctes.
  - Pour réserver des ressources GPU V100, il faut spécifier `--account=xyz@v100`
  - Pour réserver des ressources GPU A100, il faut spécifier `--account=xyz@a100`
  - Pour réserver des ressources GPU H100, il faut spécifier `--account=xyz@h100`

Soumission de travaux - Partitions Slurm

- Pour viser une partition GPU spécifique, il faut rajouter l'option Slurm correspondante lors de la réservation des ressources :

Partition souhaitée	Option Slurm correspondante
Quadri-GPU V100	
Quadri-GPU V100 + RAM GPU 16 Go	`--constraint=v100-16g`
Quadri-GPU V100 + RAM GPU 32 Go	`--constraint=v100-32g`
Octo-GPU V100	`--partition=gpu_p2`
Octo-GPU V100 + RAM CPU 384 Go	`--partition=gpu_p2s`
Octo-GPU V100 + RAM CPU 768 Go	`--partition=gpu_p2l`
Octo-GPU A100 SXM4 80 Go	`--constraint=a100`
Quadri-GPU H100 SXM5 80 Go	`--constraint=h100`

- Documentation détaillée en ligne ➜

Soumission de travaux - Partitions Slurm

Partition souhaitée	Option Slurm correspondante
Partition prepost	`--partition=prepost`
Partition compil	`--partition=compil`
Partition compil_h100	`--partition=compil_h100`
Partition archive	`--partition=archive`

- Documentation détaillée en ligne ➜
  - La partition **`prepost`** est dédiée aux travaux de pré/post-traitement.
  - La partition **`compil`** est dédiée aux compilations coûteuses.
  - La partition **`compil_h100`** est adaptée aux compilations de binaires qui s'exécuteront sur la partition Eviden H100.
  - La partition **`archive`** est dédiée aux transferts de données longs.
- Les heures utilisées sur les partitions `prepost`, `compil` et `archive` ne sont pas décomptées de votre allocation.

Soumission de travaux - *QoS* Slurm

- Lorsque vous réservez des nœuds de calcul, vous devez spécifier une ***QoS*** (*Quality of Service*) qui calibre votre besoin en ressources (nombre de GPU, temps d'exécution,...).

- Chaque partition propose 2 ou 3 *QoS* différentes dont les limites sont détaillées dans la documentation en ligne ➜

- La *QoS* est un facteur important dans le calcul de priorité de vos travaux.
  - La priorité sera plus élevée sur les *QoS* `-dev` et plus basse sur les *QoS* `-t4`.

Soumission de travaux - Script batch et connexion interactive

- Vos travaux peuvent être soumis via script batch ou exécutés en mode interactif.

	Script batch	**Mode interactif**
Quand ?	En période de production	En période de développement ou débogage
Pourquoi ?	Pour lancer des jobs conséquents de manière asynchrone	Pour enchaîner rapidement des tests courts
Avantages	· Soumission de nombreux jobs en parallèle · Gestion de dépendances possible entre les jobs soumis . L'exécution persiste si vous êtes déconnecté	· Une seule allocation de ressources à demander pour une série d'exécutions . Gestion fluide des exécutions (interruption, correction, relance,...)
Inconvénients	. Gestion des exécutions moins fluide (interruption, correction, relance,...)	. L'exécution est interrompue si vous êtes déconnecté

- Des exemples de soumission de travaux sont présentés dans la suite de ce document. Plus de détails sont disponibles sur la documentation en ligne pour :
  - la soumission de script batch ➜ (section "Exécution/contrôle d'un code GPU")
  - l'exécution en mode interactif ➜

Soumission de travaux - Script batch

- Un script batch contient :
  - un en-tête de configuration du job (nom du job, ressources demandées,...) sous forme d'une liste d'options Slurm précédées du mot-clef `#SBATCH`.
  - l'ensemble des lignes de commande à exécuter (chargement des modules, lancement de l'exécutable,...).

- Dans le script de soumission, le lancement de l'exécutable se fait avec la commande **`srun`**, qui prend en compte le paramétrage batch.

Soumission de travaux - Script batch

- Exemple de script pour une exécution sur la partition **quadri-GPU V100**
  - Exécution sur un nœud entier contenant 4 GPU à 16 Go de RAM

```
#!/bin/bash
#SBATCH --job-name=TravailGPU          # nom du job
#SBATCH --output=TravailGPU%j.out      # fichier de sortie (%j = job ID)
#SBATCH --error=TravailGPU%j.err       # fichier d'erreur (%j = job ID)
#SBATCH --constraint=v100-16g          # demander des GPU a 16 Go de RAM
#SBATCH --nodes=1                      # reserver 1 nœud
#SBATCH --ntasks=4                     # reserver 4 taches (ou processus)
#SBATCH --gres=gpu:4                   # reserver 4 GPU
#SBATCH --cpus-per-task=10             # reserver 10 CPU par tache (et memoire associee)
#SBATCH --time=01:00:00                # temps maximal d'allocation "(HH:MM:SS)"
#SBATCH --qos=qos_gpu-dev              # QoS
#SBATCH --hint=nomultithread           # desactiver l'hyperthreading
#SBATCH --account=xyz@v100             # comptabilite V100

module purge                           # nettoyer les modules herites par defaut
conda deactivate                       # desactiver les environnements herites par defaut

module load pytorch-gpu/py3/2.3.0      # charger les modules

set -x                                 # activer l'echo des commandes
srun python script.py                  # executer son script
```

Soumission de travaux - Script batch

- Exemple de script pour une exécution sur la partition **octo-GPU A100**
  - Exécution sur deux nœuds entiers contenant 8 GPU chacun

```
#!/bin/bash
#SBATCH --job-name=TravailGPU          # nom du job
#SBATCH --output=TravailGPU%j.out      # fichier de sortie (%j = job ID)
#SBATCH --error=TravailGPU%j.err       # fichier d'erreur (%j = job ID)
#SBATCH --constraint=a100              # demander des GPU A100 80 Go
#SBATCH --nodes=2                      # reserver 2 nœuds
#SBATCH --ntasks=16                    # reserver 16 taches (ou processus)
#SBATCH --gres=gpu:8                   # reserver 8 GPU par noeud
#SBATCH --cpus-per-task=8              # reserver 8 CPU par tache (et memoire associee)
#SBATCH --time=20:00:00                # temps maximal d'allocation "(HH:MM:SS)"
#SBATCH --hint=nomultithread           # desactiver l'hyperthreading
#SBATCH --account=xyz@a100             # comptabilite A100

module purge                           # nettoyer les modules herites par defaut
conda deactivate                       # desactiver les environnements herites par defaut

module load arch/a100                  # selectionner les modules compiles pour les A100
module load pytorch-gpu/py3/2.3.0      # charger les modules

set -x                                 # activer l'echo des commandes
srun python script.py                  # executer son script
```

Soumission de travaux - Script batch

- Exemple de script pour une exécution sur la partition **quadri-GPU H100**
  - Exécution sur deux nœuds entiers contenant 4 GPU chacun

```
#!/bin/bash
#SBATCH --job-name=TravailGPU          # nom du job
#SBATCH --output=TravailGPU%j.out      # fichier de sortie (%j = job ID)
#SBATCH --error=TravailGPU%j.err       # fichier d'erreur (%j = job ID)
#SBATCH --constraint=h100              # demander des GPU H100 80 Go
#SBATCH --nodes=2                      # reserver 2 nœuds
#SBATCH --ntasks=8                     # reserver 8 taches (ou processus)
#SBATCH --gres=gpu:4                   # reserver 4 GPU par noeud
#SBATCH --cpus-per-task=24             # reserver 24 CPU par tache (et memoire associee)
#SBATCH --time=20:00:00                # temps maximal d'allocation "(HH:MM:SS)"
#SBATCH --hint=nomultithread           # desactiver l'hyperthreading
#SBATCH --account=xyz@h100             # comptabilite H100

module purge                           # nettoyer les modules herites par defaut
conda deactivate                       # desactiver les environnements herites par defaut

module load arch/h100                  # selectionner les modules compiles pour les H100
module load pytorch-gpu/py3/2.4.0      # charger les modules

set -x                                 # activer l'echo des commandes
srun python script.py                  # executer son script
```

Soumission de travaux - Script batch

- Pour soumettre un script batch Slurm : ..... `sbatch <script>`
- Pour suivre l'état de soumission de vos jobs : ..... `squeue -u $USER`

  → États possibles : R = *running*, PD = *pending*, CG = *completing*
- Pour afficher l'ensemble des paramètres d'un job soumis :

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

- Pour annuler l'exécution d'un job : ..... `scancel <jobid>`

- Vous pouvez vous connecter en SSH aux nœuds de calcul affectés à vos travaux afin de surveiller l'exécution de vos calculs (`top`, `htop`, `nvidia-smi`,...) :

```
$ ssh <numéro du nœud>
```

Soumission de travaux - Connexion interactive

- Vous pouvez **ouvrir un terminal directement sur un nœud de calcul** sur lequel des ressources vous sont réservées.
- Exemple avec réservation d'un GPU sur la partition par défaut :

```
login@jean-zay3:~$ srun --ntasks=1 --gres=gpu:1 --<add-options> --pty bash
srun:  job 123456 queued and waiting for resources
srun:  job 123456 has been allocated resources
login@r13i0n8:~$
```

- Pour se déconnecter :

```
login@r13i0n8:~$ exit
exit
login@jean-zay3:~$
```

- **Attention**, MPI n'est pas supporté dans cette configuration.

Soumission de travaux - Connexion interactive

- Vous pouvez également faire une demande d'allocation de ressources et **enchaîner des exécutions** sur ces ressources.
- Exemple avec réservation d'un GPU de la partition par défaut :

```
login@jean-zay1:~$ salloc --ntasks=1 --gres=gpu:1 --<add-options>
salloc:  Pending job allocation 654321
salloc:  job 654321 queued and waiting for resources
salloc:  job 654321 has been allocated resources
salloc:  Granted job allocation 654321
login@jean-zay1:~$ srun python script_0.py
...
login@jean-zay1:~$ srun python script_1.py
...
```

- Pour libérer les ressources :

```
login@jean-zay1:~$ exit
exit
login@jean-zay1:~$ salloc:  Relinquishing job allocation 654321
```

Consommation des heures de calcul

Consommation des heures de calcul

- Le nombre d'heures de calcul h consommées par un job est défini par :

$$h = \text{nb GPU réservés} \times \text{temps } \textit{elapsed}$$

- Un nœud est réservé en exclusivité lorsque plus d'un nœud est demandé, ou si l'option Slurm `--exclusive` est activée. Dans ce cas, le décompte d'heures se fait de la façon suivante :

$$h = \text{nb nœuds réservés} \times \text{nb GPU par nœud} \times \text{temps } \textit{elapsed}$$

- Pour suivre la consommation de vos projets :

```
$ idr_acct
```

- Pour les travaux ayant un besoin conséquent en mémoire RAM CPU, la configuration Slurm du job doit être adaptée et le décompte des heures de calcul est légèrement différente. ➜

Pour aller plus loin

Pour aller plus loin - Formations, workshops et hackathons

- L'IDRIS dispense diverses formations à destination des utilisateurs de calcul scientifique HPC et IA. ➜
  - MPI
  - OpenMP
  - Programmation Hybride MPI/OpenMP
  - Débogage HPC
  - Vectorisation SIMD
  - PETSc
  - Introduction à OpenACC et OpenMP GPU
  - **Introduction Pratique au Deep Learning (IPDL)**
  - **Deep Learning Optimisé sur Jean Zay (DLO-JZ)**
  - **Spécialisation de LLM (SpeLLM)**

- L'IDRIS organise des workshops autour de la prise en main du supercalculateur et de l'optimisation de vos codes de calcul. ➜

- L'IDRIS organise des hackathons en partenariat avec NVIDIA. ➜

Pour aller plus loin - Contacter l'IDRIS

- Pour toute question ou demande sur l'accès à la machine, le déploiement de l'environnement logiciel, le débogage ou l'optimisation de votre code,...
  le Support Utilisateurs de l'IDRIS est joignable :

  du lundi au jeudi de 9h à 18h
  le vendredi de 9h à 17h30
  par mail à assist@idris.fr
  ou par téléphone au +33 (0)1 69 35 85 55

- Pour toute demande relative à l'administration de votre compte de calcul (mot de passe, ouverture de compte, autorisation d'accès, enregistrement des adresses IP,...), contactez gestutil@idris.fr.