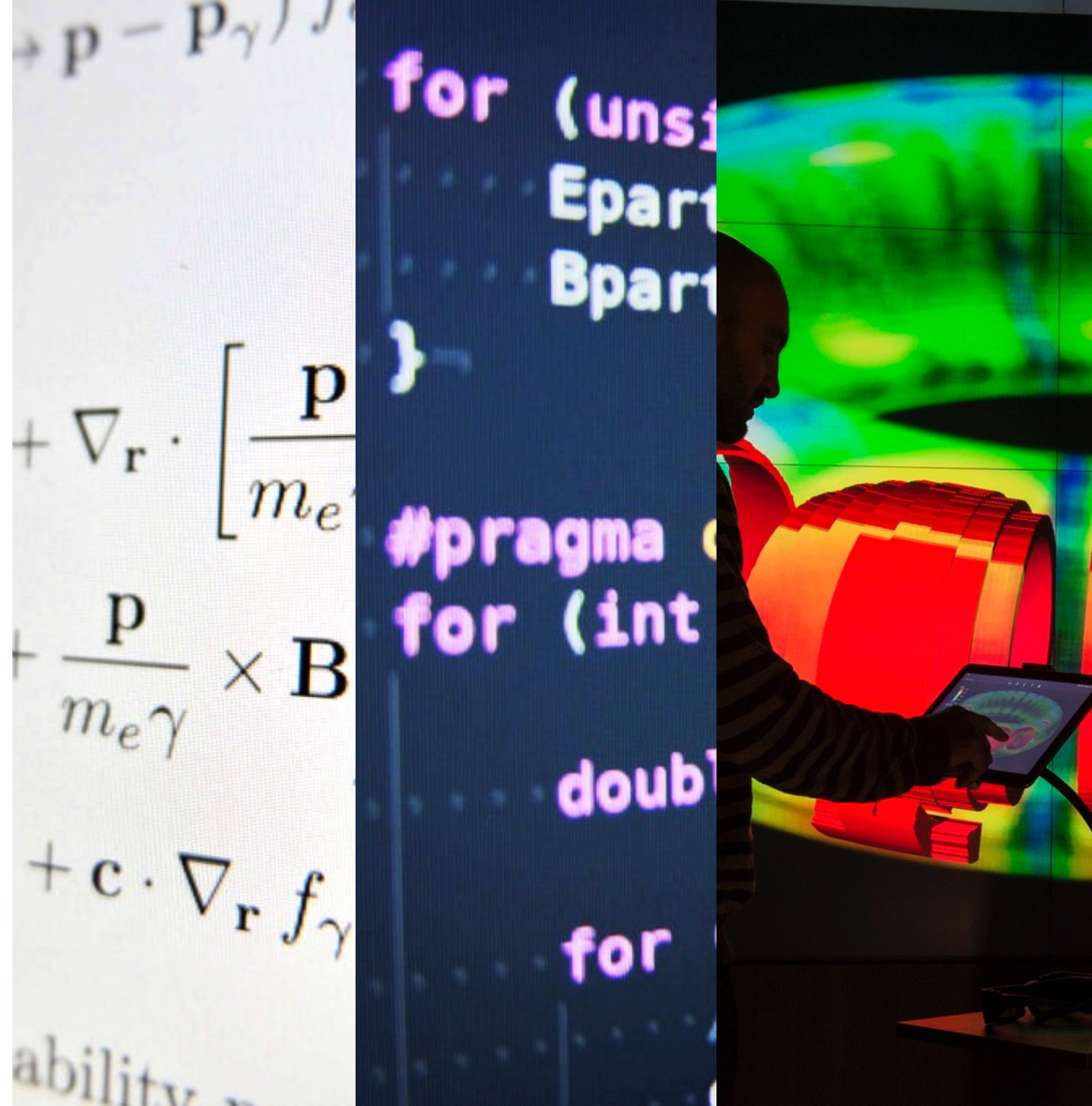


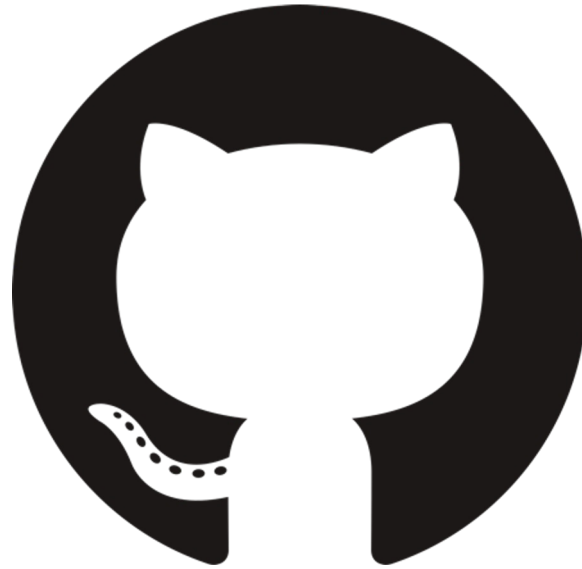
# Prise en main d'un super- calculateur

Master DFE

Mathieu LOBET

Année 2023-2024





<https://github.com/Maison-de-la-Simulation/HPC-DFE-Paris-Saclay>

- Le **mésocentre de l'université Paris-Saclay** regroupe 3 plateformes techniques majeures de calcul scientifique / traitement de données issues des établissements partenaires dans le projet Paris-Saclay
- Ces plateformes sont accessibles aux chercheurs et aux étudiants
- Nous utiliserons le **super-calculateur Ruche** du mésocentre pour ce cours
- Site internet : <https://mesocentre.universite-paris-saclay.fr/>
- Documentation utilisateur de Ruche : [https://mesocentre.pages.centralesupelec.fr/user\\_doc/](https://mesocentre.pages.centralesupelec.fr/user_doc/)
- Le fonctionnement de Ruche est très similaire à la plupart des super-calculateurs au monde, même les plus gros.
- **L'utilisation des ressources du Mesocentre est strictement limitée à un usage en lien avec le Master. Vous pouvez l'utiliser pour les autres matières mais en aucun cas pour des usages personnels ou professionnels.**

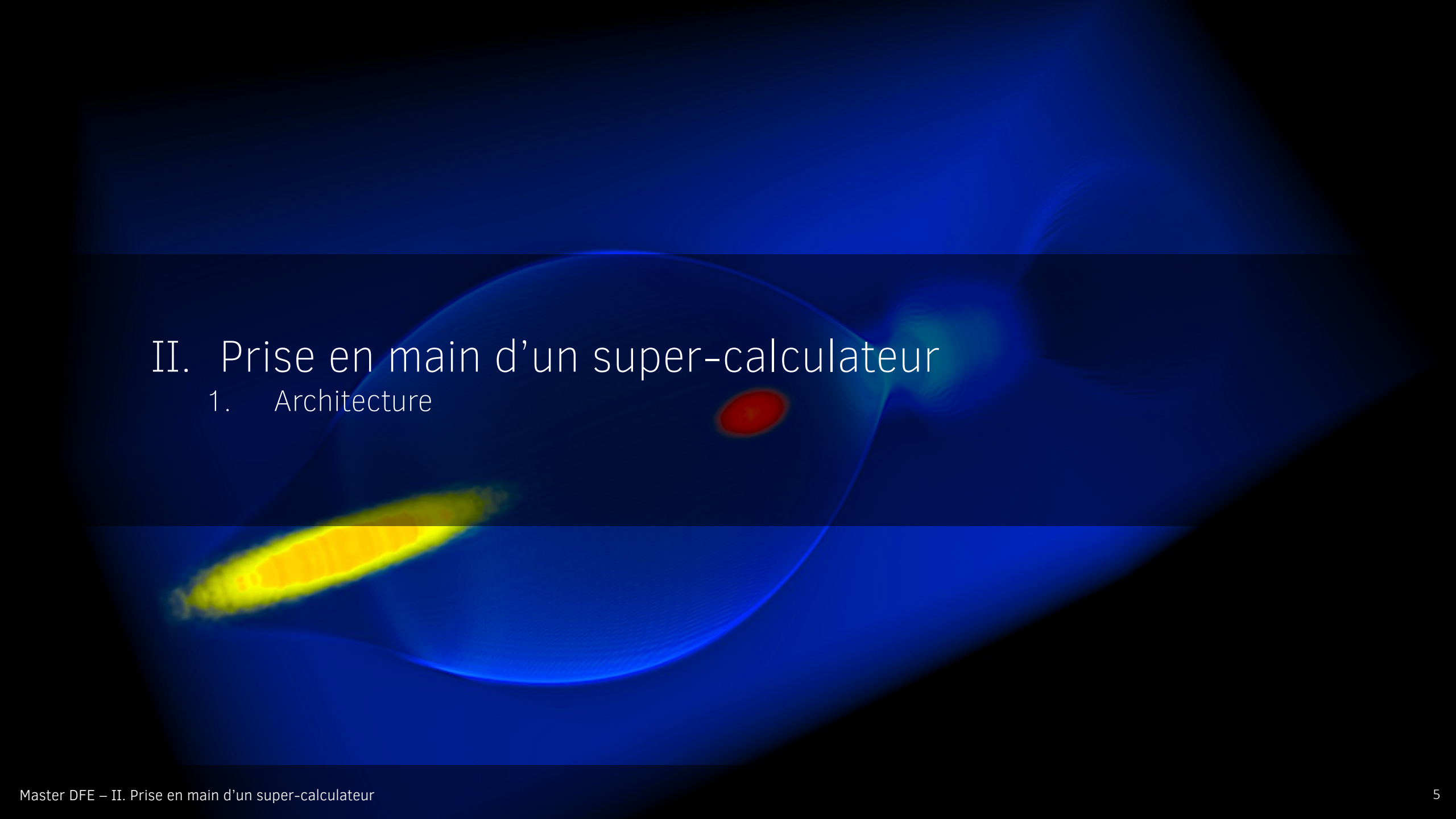
I. Introduction au calcul parallèle

II. Prise en main d'un super-calculateur

III. Introduction au parallélisme par échange de message via MPI

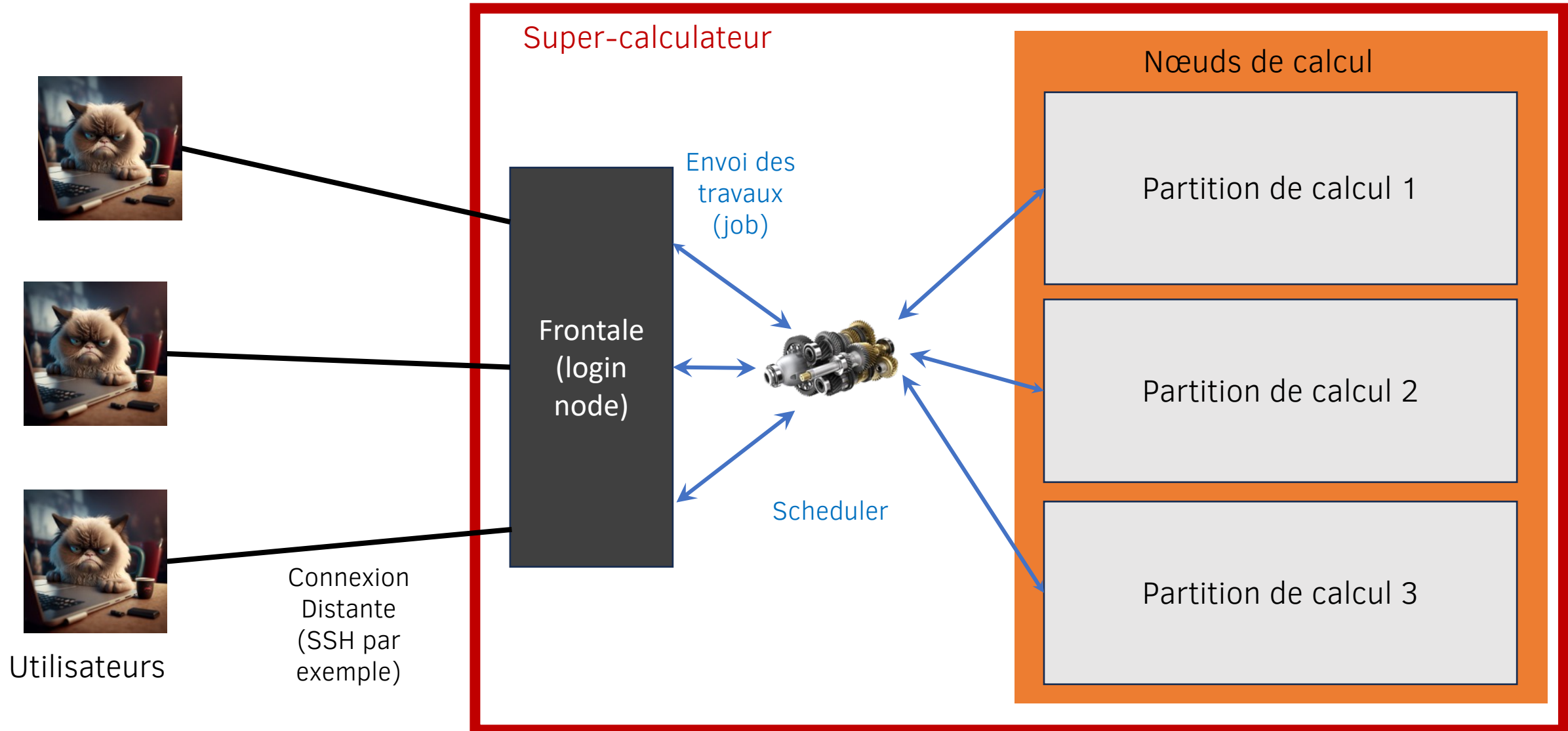
IV. Mesure de la performance

V. Travaux pratiques

The background of the slide is a dark blue gradient with abstract, flowing light patterns. A prominent yellow and orange elongated shape is visible in the lower-left quadrant, and a smaller red oval is located near the center-right. The text is overlaid on this background.

## II. Prise en main d'un super-calculateur

### 1. Architecture



- Vous vous connectez en premier lieu sur la **frontale** : c'est un serveur partagé qui accueille les utilisateurs et qui permet de gérer son environnement et d'exécuter de petits scripts
- Pour faire tourner un programme, on ne l'exécute pas directement comme on le ferait sur son ordinateur, on le soumet à un **ordonnanceur de tâches**
- Le rôle de l'ordonnanceur est de gérer l'allocation des ressources de calcul pour l'ensemble des utilisateurs en respectant des règles de priorité et en maximisant le taux d'occupation du super-calculateur
- L'ordonnanceur utilisé sur Ruche s'appelle **SLURM**

- Ruche possède :
  - 216 nœuds CPU équipés chacun de 2 processeurs Intel Cascadelake de 20 cœurs chacun
  - 10 nœuds hybrides équipés de GPU NVIDIA V100
  - 9 nœuds hybrides équipés chacun de GPU NVIDIA A100
- Pour des raisons d'économie d'énergie, tous les nœuds ne sont pas allumés en ce moment.
- Dans ce cours nous n'utiliserons que les nœuds CPU



- Les ressources en calcul sont en général **divisées en partitions** qui ont chacune leurs propriétés
- Les partitions CPU sur Ruche :
  - **cpu\_short** : partition pour les jobs courts, c'est celle que nous utiliserons (max 1h de calcul)
  - **cpu\_med** : partition pour les jobs de taille moyenne
  - **cpu\_long** : partition pour les jobs long
  - **cpu\_prod** : partition pour les jobs de production
- Sur d'autres calculateurs, les partitions n'auront pas les mêmes fonctions et nom mais le principe reste globalement le même
- Plus d'information sur les partitions ici :  
[https://mesocentre.pages.centralesupelec.fr/user\\_doc/ruche/07\\_slurm\\_partitions\\_description/](https://mesocentre.pages.centralesupelec.fr/user_doc/ruche/07_slurm_partitions_description/)
- La commande **sinfo** permet de récupérer des infos sur les partitions

- Tous les super-calculateurs vous permettent d'accéder à **divers espaces de stockage** (file system)
  - **\$HOME** : c'est l'espace de stockage par défaut à la connexion. C'est un espace relativement petit en général (quelques Go) et sauvegardé
  - **\$WORK** : c'est un espace plus conséquent permettant d'installer les codes, les bibliothèques et de stocker des gros fichiers (souvent quelques To)
  - **\$SCRATCH** : c'est l'espace de travail des simulations. C'est en général un espace très rapide en écriture et lecture pour gérer les besoins des codes mais il n'est pas sauvegardé.
  - **\$STORE** : c'est un espace dédié à la sauvegarde de gros volumes de données à long terme, l'accès est en revanche restreint à de gros fichiers.
- Sur Ruche, nous n'avons qu'un espace \$HOME (50 Go) et \$WORK (500 Go)

The background of the slide features an abstract design with flowing blue and yellow light patterns. A large, bright yellow oval shape is positioned in the lower-left quadrant, while a smaller, glowing blue shape is located in the upper-right. The overall effect is a sense of dynamic energy and technological sophistication.

## II. Prise en main d'un super-calculateur

### 2. Première connexion et gestion de l'environnement

- On se connecte à Ruche en utilisant un terminal et en passant par la commande ssh suivante :

```
> ssh -XY <votre login>@ruche.mesocentre.universite-paris-saclay.fr
```

- On arrive ensuite sur la frontale
- Vous êtes alors sur votre espace de stockage \$HOME

- Vous pouvez connaître votre quota mémoire en utilisant la commande `ruce-quota` :



```
➤ ruce-quota
```

```
Disk usage on home :
```

```
quota: 8,4G / 50G (16%)
```

```
files: 38 727 / 200 000 (19%)
```

```
Disk usage on workdir :
```

```
quota: 160G / 500G (31%)
```

```
files: 316 856 / 400 000 (79%)
```

- Les éditeurs disponibles sont intégrés au terminal : `vim`, `emacs`, `nano`
- Vous pouvez aussi vous connecter à Ruche en passant par `Visula Studio Code` ou un éditeur local qui vous permet une connexion ssh distante
- Les transferts de fichiers peuvent se faire avec la commande `scp`
- Vous pouvez aussi faire du transfert de fichier en utilisant le logiciel `FileZilla`

- A la connexion un super-calculateur vient avec un environnement par défaut qui n'est pas forcément adapté à votre usage
- Les super-calculateurs utilisent la notion de **module pour charger ou décharger des compilateurs, des logiciels ou des bibliothèques**
- La commande `module list` permet d'afficher tous les modules chargés dans actuellement dans votre environnement :



```
➤ Module list
```

- La commande `module avail` permet de connaître tous les modules disponibles

```
➤ Module avail
```

- La commande `module load` permet de charger un module spécifique

```
➤ Module load anaconda3/2022.10/gcc-11.2.0
```

- La commande `module unload` permet de décharger un module spécifique

```
➤ Module unload anaconda3/2022.10/gcc-11.2.0
```

- La commande `module purge` permet de décharger tous les modules chargés

```
➤ Module purge
```



- Pour ce cours, nous utiliserons Python avec les bibliothèques suivantes :
  - Numpy
  - Matplotlib
  - MPI4py
- Par commodité, j'ai fait en sorte que tout soit disponible en utilisant la commande suivante :

```
➤ source /gpfs/workdir/labotm/Installations/miniconda3/m2dfe_env.sh
```

- Page de l'exercice : [https://github.com/Maison-de-la-Simulation/HPC-DFE-Paris-Saclay/blob/master/exercices/ruche/1\\_start.md](https://github.com/Maison-de-la-Simulation/HPC-DFE-Paris-Saclay/blob/master/exercices/ruche/1_start.md)
- Vous pouvez aussi y accéder via votre éditeur en ouvrant le fichier suivant :



```
> vim exercices/ruche/1_start.md
```

- Cet exercice a pour but de faire nos premiers pas sur le calculateur Ruche



## II. Prise en main d'un super-calculateur

### 3. Compréhension de SLURM et lancement d'un premier job

- **SLURM** est un gestionnaire de super-calculateurs et un **ordonnanceur de tâches** (jobs) (<https://slurm.schedmd.com/quickstart.html>)
- Il est l'un des plus utilisé aujourd'hui dans le monde
- SLURM fournit à l'utilisateur un certain nombre de commandes pour soumettre des jobs, les gérer et les analyser.
- **Un job est un ensemble de commande (incluant l'exécution du code mais pas seulement) que l'on souhaite voir exécuter par le super-calculateur**
- Un job est un script dont l'entête est formatée pour fournir à SLURM les conditions d'exécution
- [https://mesocentre.pages.centralesupelec.fr/user\\_doc/ruche/06\\_slurm\\_jobs\\_management/](https://mesocentre.pages.centralesupelec.fr/user_doc/ruche/06_slurm_jobs_management/)

## Launch.slurm



```
#!/bin/bash
#SBATCH --job-name=master_dfe # Nom du job
#SBATCH --output=output # fichier qui réceptionne la sortie standard
#SBATCH --error=error # fichier qui réceptionne la sortie d'erreur
#SBATCH --ntasks=1 # Nombre d'unité de calcul ou de processus MPI
#SBATCH --nodes=1 # Nombre de noeuds à exploiter
#SBATCH --time=00:05:00 # Temps souhaité pour la réservation
#SBATCH --partition=cpu_short # Partition des jobs rapides

# Permet de s'assurer que l'exécution a lieu dans le dossier de soumission du job
cd $SLURM_SUBMIT_DIR

set -x

# On regarde les propriétés des processeurs via la commande lscpu
lscpu

# Attente de 10 secondes pour faire durer un peu le job
sleep 10
```

- Il existe un grand nombre d'options de paramètres SLURM pouvant être passées en entête.
- Voir la page de Ruche ou la documentation SLURM pour plus d'information :
  - [https://mesocentre.pages.centralesupelec.fr/user\\_doc/ruche/06\\_slurm\\_jobs\\_management/](https://mesocentre.pages.centralesupelec.fr/user_doc/ruche/06_slurm_jobs_management/)
  - [https://mesocentre.pages.centralesupelec.fr/user\\_doc/ruche/08\\_slurm\\_examples/](https://mesocentre.pages.centralesupelec.fr/user_doc/ruche/08_slurm_examples/)
  - <https://slurm.schedmd.com/pdfs/summary.pdf>

- Le script SLURM se lance avec la commande `sbatch`



```
> sbatch launch.slurm
```

- Le script se retrouve alors en queue, vous pouvez consulter l'ensemble de la queue en faisant



```
> squeue -a
```

- Vous pouvez lancer plusieurs scripts en même temps, c'est SLURM qui décidera quand les lancer. Vous pouvez consulter leur état en faisant :



```
> squeue -u <login>
```

- A partir du numéro de job récupérer avec `squeue`, vous pouvez annuler un job en attente ou en cours d'exécution avec `scancel`



```
> scancel <job id>
```

- Il existe d'autres commandes SLURM que nous n'utiliserons pas pour ce cours que vous trouverez soit sur la doc de Ruche soit sur celle de SLURM



- Page de l'exercice :
- Vous pouvez aussi y accéder via votre éditeur en ouvrant le fichier suivant :



```
> vim exercices/ruche/2_slurm.md
```

- Cet exercice a pour but de vous apprendre à concevoir vos premiers scripts SLURM et de les exécuter