Input/output & debugging

some updates

Geoff Lesur, Sep. 19th 2024

I- Debugging updates

Reminder

Debugging in Idefix

- Always start by enabling Idefix_DEBUG in ccmake, or -DIdefix_DEBUG=ON in cmake and recompile

- Keep in mind the rules of debugging:

1. Never use pointers to the host memory space in an `idefix_for`

2. Always make shallow copies of whatever you need before calling `idefix_for`

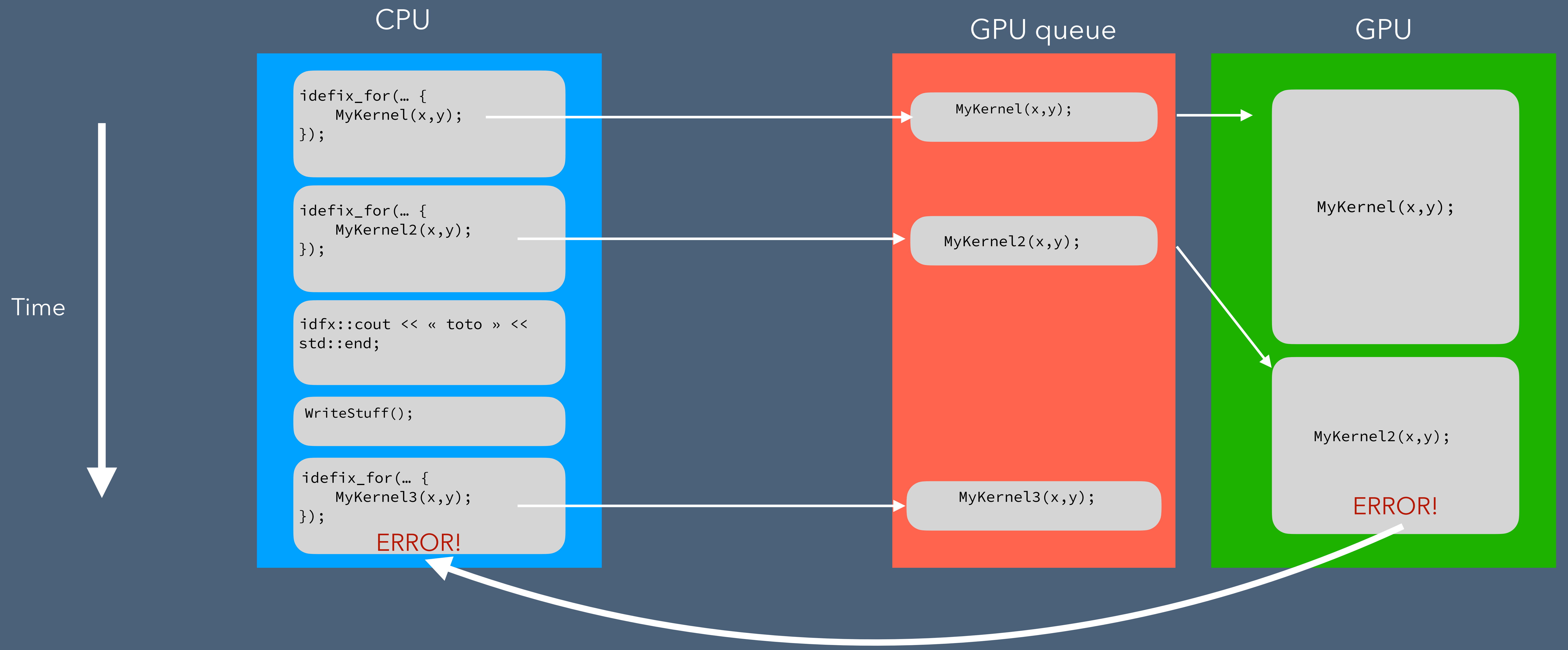
3. A segmentation fault always shows up after the faulty instruction (sometimes 100s of lines after…)

4. Always check that performances are what you expect

no idea what I'm talking about? check out Idefix tutorial: github.com/idefix-code/tutorial

No more need for Kokkos kernel_logger

- Previously, when debugging a segmentation fault on GPUs, the fault could be triggered long after idefix_for was launched



- We used to dynamically link Idefix with Kokkos kernel logger to force sync (see Idefix days 2023)

- **Not needed anymore:** Idefix_DEBUG in cmake automatically force synchronisation at the end of each idefix_for

Embedded profiler

No more space-time-stack (since v2.0.0)

- Profiling on multiple architectures is cumbersome (each manufacturer has its own tool: gprof, intel Vtune, AMD Perfecto, Nvidia System & Compute…)

- We used to rely on Kokkos tools space-time-stack (e.g. idefix days 2023)

- **Now**: profiler is directly embedded in idefix (no need to re-compile!). Just add `-profile` to the command line (since v2.0.0)

- And yes, it works on any architecture, with/without MPI (the MPI version gives you a report per MPI process)

- Live demo!

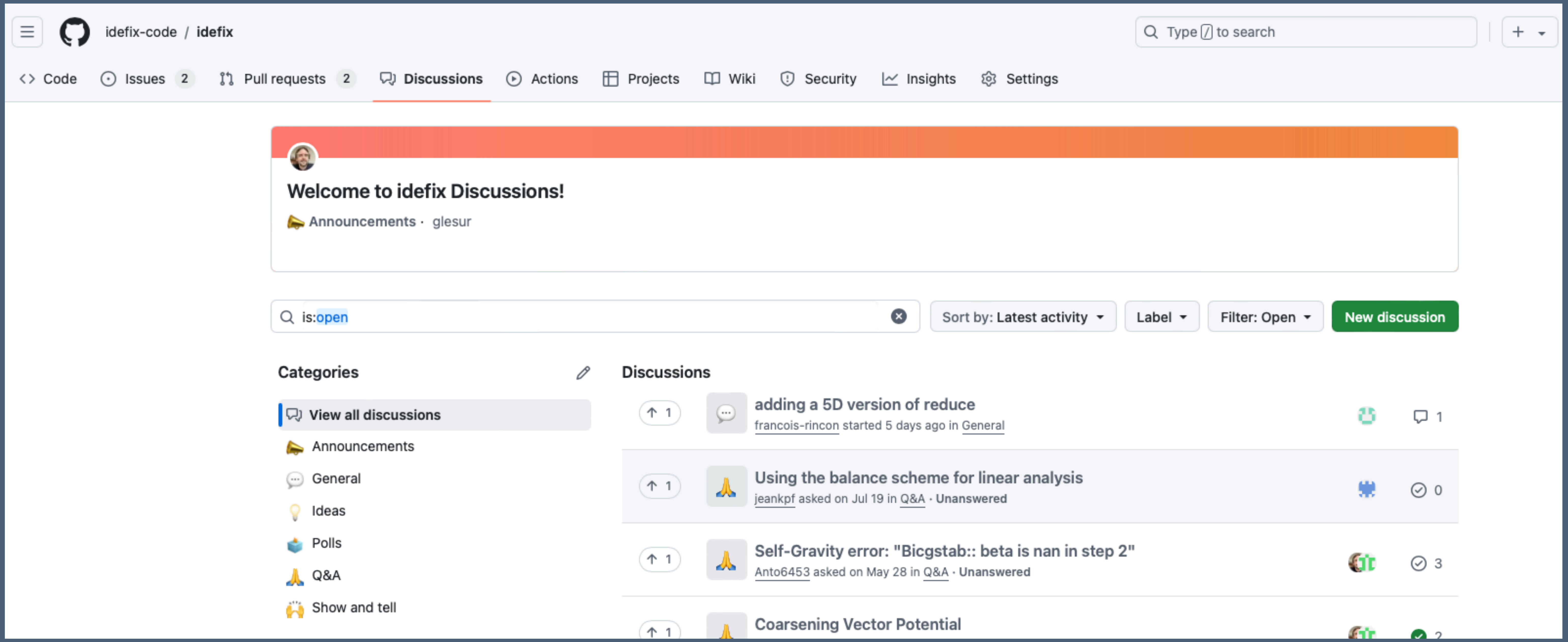
MPI Load balancing

available since v2.0.05

- Initially motivated by over-heated nodes on Lumi (possibly a common problem)
- « MPI imbalance » defined as $\dfrac{\max_{\mathrm{proc}}(t_{\mathrm{evolveStage}}) - \min_{\mathrm{proc}}(t_{\mathrm{evolveStage}})}{\langle t_{\mathrm{evolveStage}} \rangle_{\mathrm{proc}}}$
- Computes the imbalance between cores *excluding MPI communications*
- Throws a warning if above 20%, identifying which MPI process is lagging
- Node name is now explicitly mentioned in the log file of each process (idefix.xx.log)
- Live demo

No more Idefix Slack

- Information on slack gets lost
- Too expensive to get a paid account for everybody
- Move to GitHub Discussions



idefix-code / idefix

Type / to search

Code | Issues 2 | Pull requests 2 | Discussions | Actions | Projects | Wiki | Security | Insights | Settings

Welcome to idefix Discussions!
Announcements · glesur

is:open

Sort by: Latest activity | Label | Filter: Open | New discussion

Categories

- View all discussions
- Announcements
- General
- Ideas
- Polls
- Q&A
- Show and tell

Discussions

Votes	Discussion	Comments
1	adding a 5D version of reduce — francois-rincon started 5 days ago in General	1
1	Using the balance scheme for linear analysis — jeankpf asked on Jul 19 in Q&A · Unanswered	0
1	Self-Gravity error: "Bicgstab:: beta is nan in step 2" — Anto6453 asked on May 28 in Q&A · Unanswered	3
1	Coarsening Vector Potential	2

II-Input & output

VTK slices

Dealing with large datasets

- For some large simulations, it becomes impossible to load the whole grid for visualisation
- One usually needs cuts through the domain=slices
- Idea: compute the slices on-the-fly while the code is running, and write only the slice data
- Also useful for movie making

VTK slices

How?

- In block [Output] of your input file (idefix.ini):
  - Slice1, every $\Delta T = 2.0$:

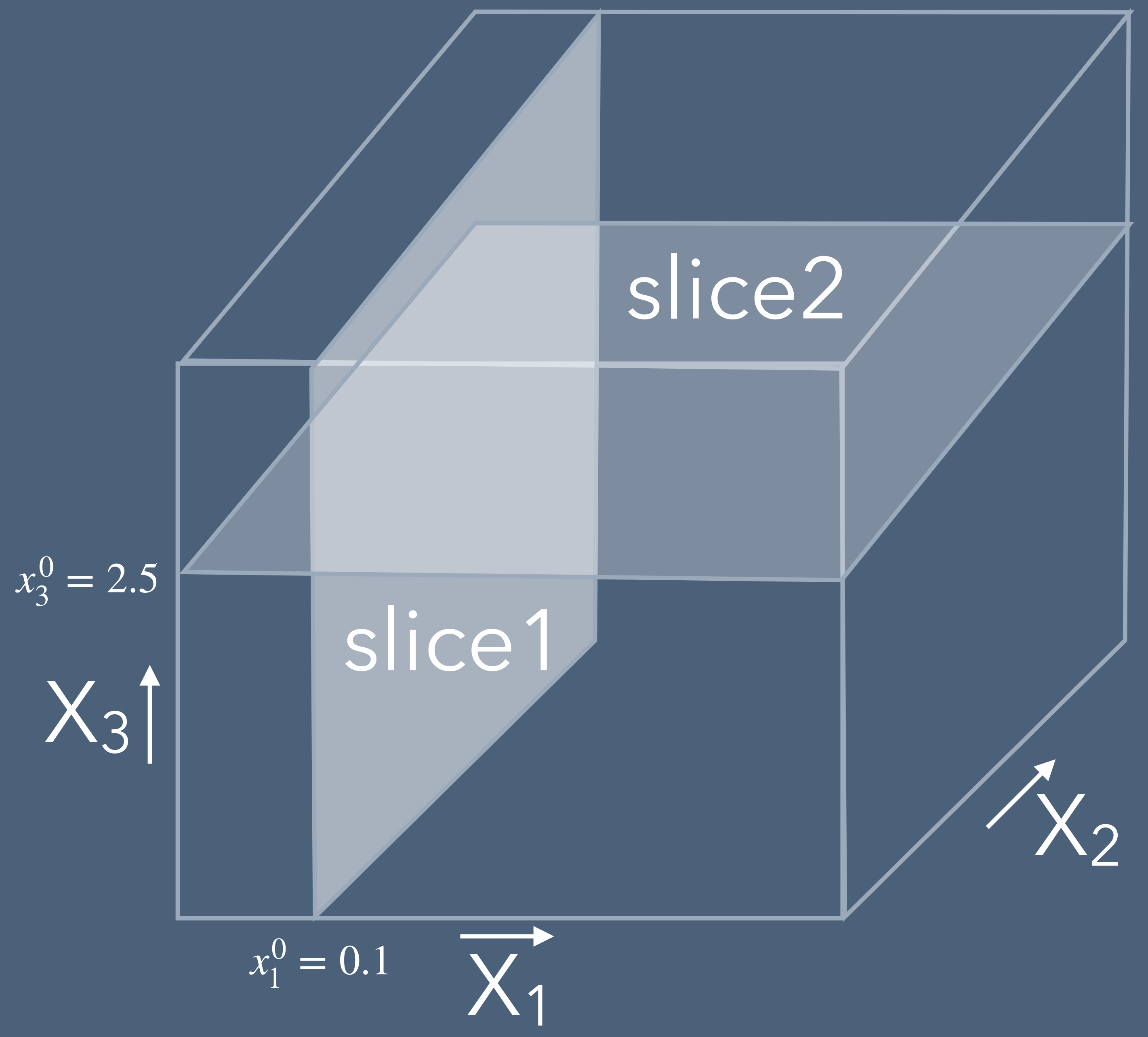
```
vtk_slice1   2.0  0  0.1  cut
```

periodicity (float) — direction (integer) x1=0, x2=1, x3=2 — offset along direction (float) — slice type (cut or average)

  - Slice2, every $\Delta T = 0.1$:

```
vtk_slice2   0.1  2  2.5  cut
```

- One can have as many slices as one wishes
- Average slices are point wise averages (no weight by cell volume)



DumpImage

Load a restart dump file in Idefix

- Idefix can deal automatically with the addition of new physics upon restart (e.g restart from a pure hydro dump file and add dust) using the command line options:
  `–force_init -restart`

- However, Idefix <u>restart procedure won’t work if the grid is changed</u>

- The way to proceed is to generate a new initial condition from an existing dump file.
  For instance:

  - Change dimensionality (restart a 2D run into 3D)

  - Change grid (increase resolution) by designing an adequate interpolation procedure

DumpImage

How to use?

1. Open the dump file

```
DumpImage image("mydump.dmp", &data);
```

2. Use it !

```
for(int k, j, i) {
  int iglob=i-2*d.beg[IDIR]+d.gbeg[IDIR];
  ...// same for j and k
  d.Vc(RHO,k,j,i) = image.arrays["Vc-RHO"](kglob,jglob,iglob)
```

container for the dump data

Name of the array in the dump

NB: restart dump array only contains the full (global) active domain (i.e. it excludes the boundaries, but it is not decomposed accross MPI procs)

Python interface

- Rationale for outputs
  - Some science project requires complex post processing (often done with Python)
  - Instead of post-processing Idefix outputs (e.g. VTK), one could directly call a Python routine fed with Idefix's dataBlock
- Rationale for inputs (initial condition)
  - Some initial conditions might require complex equilibrium computations and/or spectral decomposition
  - In these cases, Python can be used to compute the initial conditions thanks to all of the libraries available

`Pydefix` class, a class that allows Idefix to directly communicate with a Python interpreter

Python interface

How?

- Enable `Idefix_PYTHON` in cmake
- in the `[Python]` block of your input file:
  - script: name of the python script file (without .py !)
  - output_function: name of the function called during output (optional)
  - initflow_function: name of the function called during flow initialisation (optional)
- If using python outputs:
  - in the [Output] block, specify the period of python outputs

Live demo!

```
[Grid]
X1-grid    1  0.0  256  u  1.0
X2-grid    1  0.0  256  u  1.0
X3-grid    1  0.0  1    u  1.0

[TimeIntegrator]
CFL          0.6
tstop        0.5
first_dt     1.e-4
nstages      2

[Hydro]
solver    roe

[Python]
script                 pydefix_example
output_function        output
initflow_function      initflow

[Boundary]
X1-beg    periodic
X1-end    periodic
X2-beg    periodic
X2-end    periodic
X3-beg    outflow
X3-end    outflow

[Output]
log    10
python    0.02
```

Python interface

pros and cons

- Pros:
  - No copy involved of Idefix's dataBlockHost, we just warp IdefixArrays into numpy arrays
  - Works flawlessly when Idefix runs on GPU (python scripts however run on CPU)
  - Possible to run idefix without writing a single line of C++! (no need for a setup.cpp)
- Cons:
  - When using MPI, each process has its own python interpreter → each python script only has access to the local-subdomain
  - Not possible to use python to define boundary conditions or user-defined source terms (would be a performance killer)