



Schweizerische Eidgenossenschaft
Confédération suisse
Confederazione Svizzera
Confederaziun svizra

Swiss Confederation

Federal Department of Home Affairs FDHA
Federal Office of Meteorology and Climatology MeteoSwiss

Dawn - Hands on Exercise

Summer School on Effective HPC for Climate and Weather

24.08.2020



Objective

- Write a simple **dusk** stencil calculating the divergence
- Compile the stencil using **dawn**
- Run the stencil, ensure correctness
 - optional: visualize the results
- We recommend that you read these slides completely first before you start working on the exercise



Preliminary Steps - VM

- If you downloaded your virtual machine **before 11/8/2020** you most likely won't have the folder `ds1s/mch-summer-school/` under your home folder.
- In that case, you need to go through the steps to build a docker image containing everything you need for this exercise (next slide).
- **Otherwise**, you can `cd` to `ds1s/mch-summer-school/` and **skip to slide 5**



Preliminary Steps - Docker build

- Build the docker image (this will take quite some time, possibly 1 hour and more depending on your system)

```
git clone https://github.com/dawn-ico/mch-summer-school
cd mch-summer-school/docker
sudo docker build . --tag summer-school
```

- To run the docker image please run

```
sudo docker run -it summer-school /bin/bash
```



Initialize build system

- The final step in preparation for the exercise is to initialize the build system and to switch into the build directory:

```
source spack-build-env.txt && cd spack-build
```



Note on the Tutorial

From now on, the tutorial works with **relative paths**, so if you want to copy paste commands from these slides, please make sure that you do not change directory but remain in `mch-summer-school/spack-build`. If in doubt, make sure that `pwd` returns one of these paths:

`pwd`

→ `/home/ubuntu/dsls/mch-summer-school/spack-build` (if outside docker, recent vm)

→ `/home/root/mch-summer-school/spack-build` (if inside docker)



Editing the dusk stencil

- To edit the dusk stencil run:

```
vim ../dusk_stencils/divergence.py
```

- If you do not like vim, feel free to install and use another text editor, e.g.

```
apt-get install nano
```

```
nano ../dusk_stencils/divergence.py
```



Computing the divergence using Finite Volumes

Finite volumes is a numerical method to solve Partial Differential Equations (PDEs), as they arise e.g. in climate simulations. For this exercise, we are going to use it to approximate the divergence of a vector field.

- The Finite Volume Method divides the space into small, disjoint control volumes
- The divergence represents the volume density of the outward flux of a vector field from a control volume around a given point.

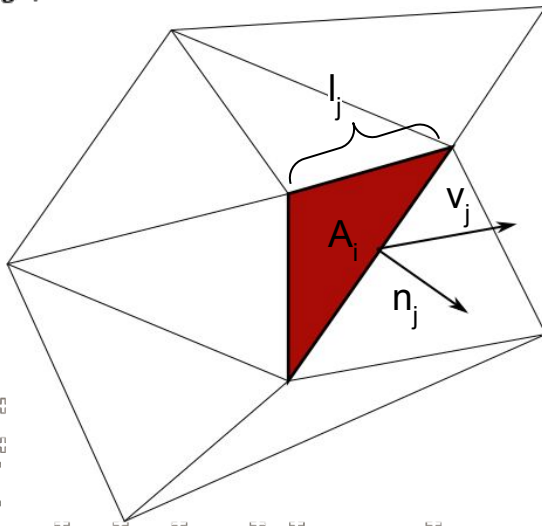




Computing the divergence using Finite Volumes

Thus, the divergence can be approximated using FVM by:

$$\nabla(\underline{v})_i \approx \frac{1}{A_i} \sum_{j=1}^N (\underline{v}_j \cdot \underline{n}_j) \cdot l_j$$



A_i = Cell Area
 \underline{v}_j = Velocity at edge j
 \underline{n}_j = Outward normal at edge j
 l_j = length of edge j



Computing the divergence using Finite Volumes

- The **dusk skeleton** contains the following fields in the API
 - `vec`: input field on edges, premultiplied with normal ($\text{vec} = n_j \cdot v_j$)
 - `edge_length`: length of edges (l_j)
 - `cell_area`: area of cells (A_i)
 - `edge_orientation_cell`: on a mesh, not all normals can point locally outwards for each cell. This sparse field contains correct signs for each cell to flip the normals outwards (3 entries per cell, since each cell has 3 edges)
- The dusk skeleton contains some dummy computations to show you some of the syntax of dusk. They do no harm, you can just append the divergence computation at the end (uncomment the last line and start there)



Compiling and running the dusk stencil

- To build your stencil, save and close your editor and simply run `make`. This calls down in the background and compiles both a cuda and a c++ version of the stencil

`make`

- To run the stencil, call the divergence driver. This will test your stencil, display measured errors against the analytical solution, and show if your computation was correct

`./cpp_drivers/divergence_driver`





Compiling and running the dusk stencil

If your program is correct, you should see the following

```
[=====] Running 1 test from 1 test suite.  
[-----] Global test environment set-up.  
[-----] 1 test from nh_diffusion_fvm  
[ RUN      ] nh_diffusion_fvm.manufactured_and_sidebyside  
MEASURED ERRORS: L_inf 0.0343472 L_1 0.0106434 L_2 0.0135502  
[      OK ] nh_diffusion_fvm.manufactured_and_sidebyside (145 ms)  
[-----] 1 test from nh_diffusion_fvm (145 ms total)  
  
[-----] Global test environment tear-down  
[=====] 1 test from 1 test suite ran. (145 ms total)  
[ PASSED ] 1 test.
```

MeteoSwiss



Troubleshooting

- The compilation of the **driver code** fails:

Make sure that you access (or write into) every field of the signature of the dusk stencil. Otherwise, dawn will eliminate the unused field, and the driver code does not fit to the stencil anymore, causing the compilation to fail

- Assertion failed: **'dimsConsistent'** Dimensions consistency check failed at line -1

Make sure that you do not try to add/multiply/... "Edge" and "Cell" fields, and that your reductions conform to the Chain you passed. Unfortunately, our compiler toolchain does not yet support line numbers, so you're on your own to spot the inconsistent dimensions.

- IndentationError: **unindent** does not match any outer indentation level

While dusk is not Python, it adheres to the same rules about whitespace. Make sure that your code block is indented correctly.



Visualize the output

- (docker only) First need to copy the vtk file produced from inside the docker container to the virtual machine: **open up a new terminal** and run

```
sudo docker cp `sudo docker ps -l --format "{{.Names}}"`:/home/root/mch-summer-school/spack-build/out.vtk .
```

- Your virtual machine comes with Paraview, which can be used to look at the in- and output of our stencil

```
paraview out.vtk
```

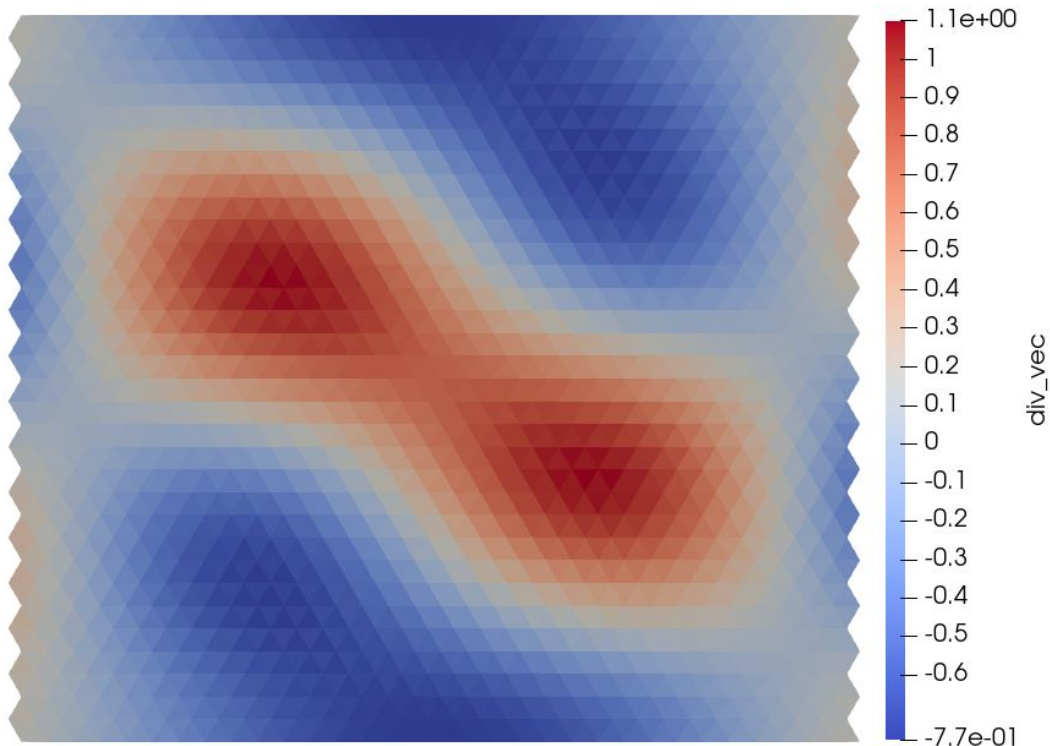
- Click on the “eye” next to **out.vtk** on the left panel
- Below, under Coloring, select *div_vec* from the dropdown menu





Visualize the output

Correctly computed divergence



MeteoSwiss



Other things to do

If you are interested to see what happens behind the scene, you are very welcome to check out the code produced by dawn. Because dawn makes quite a mess, let's first format the code properly:

```
clang-format -i dusk_stencils/generated/divergence_cuda.cpp  
clang-format -i dusk_stencils/generated/divergence_cxx-naive.cpp
```

Then you can inspect the code using e.g.

```
vim dusk_stencils/generated/divergence_cuda.cpp  
vim dusk_stencils/generated/divergence_cxx-naive.cpp
```




Other things to do

You can also toy around with the driver code

```
vim ../cpp_drivers/divergence_driver.cpp
```

On line 99, the grid size is defined

- You can observe the error against the analytical solution by increasing or decreasing the mesh size
- As a (advanced) exercise you can try to plot the error against grid size

