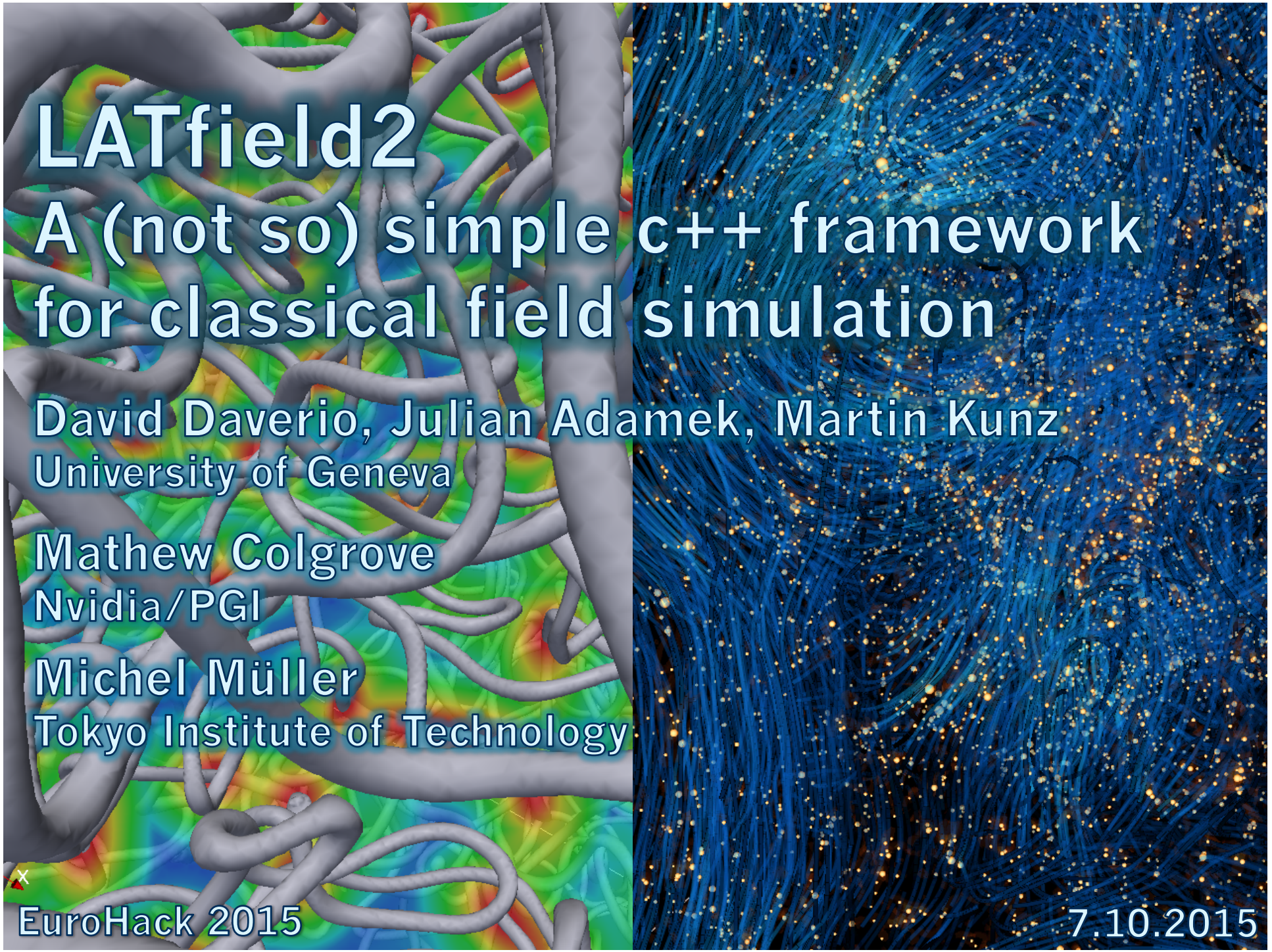


LATfield2

A (not so) simple c++ framework for classical field simulation

David Daverio, Julian Adamek, Martin Kunz
University of Geneva

Mathew Colgrove
Nvidia/PGI

Michel Müller
Tokyo Institute of Technology

EuroHack 2015

7.10.2015

Applications

1) LAH: Lattice Abelian Higgs

Cosmic strings simulation, 3 years of production at CSCS.

typical production run:

- 4096^3 sites. 32k MPI process.
- 150 FFTs of 10 component field. (150 x ~2.5 TB = 375 Tb)

2) gevolution: N-Body code within a general relativity framework:

Still in development, tested with 4096^3 lattice sites and 0.5×10^{11} particles.

Typical production runs will be:

- 4096^3 sites. 16k MPI process.
- ~13k FFTs (13000 x ~0.5 Tb = 6.5 PB)
- FFTs: 63% of the execution time

Starting Point

- Compiled with GNU compiler for production
- Compiled small examples (test cases) of LATfield2 which only perform FFTs and verify the result up to machine precision. (PGI compiler)

Porting Steps

Goal: Port the FFT wrapper, based on 1d serial FFT of FFTW to cuFFT

2 Branches:

Using cuFFT interface to FFTW

Rewriting the wrapper using openACC for data movement/transpositions and cuFFT for 1d transforms.

!! BOTH DONE !!
Thanks everyone

Issues
cuFFT Documentation

Poor documentation makes the library unnecessarily hard to use:

A good doc should contain:

- A complete list of functions and datatypes:
  Function doc should only contain what it does, which parameters it takes and what is the output. In a concise text.
- Hyperlinks to navigate from example to specs (and back).
- A search engine.
  We where using the firefox “search in text” to look for function definitions in the doc...

This week at least 10 hours have been lost to search, cry or laugh with the doc.

Issues
cuFFT: fftw interface 1

Interface issue with fftw_plan_many_dft

```
fftw_plan_many_dft(int rank, const int *n, int howmany, fftw_complex *in,
                   const int *inembed, int istride, int idist,
                   fftw_complex *out, const int *onembed, int ostride,
                   int odist, int sign, unsigned flags);
```

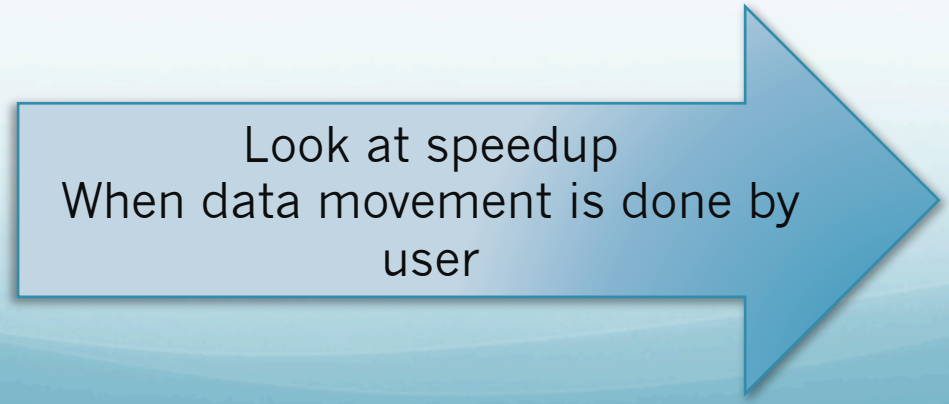
If i-onembed are set to NULL, fftw still uses stride and dist parameters, cuFFT interface will ignore them.

Issues
cuFFT: fftw interface 2

cuFFT interface have internal management of data movement.

A switch to user management of data movement would be extremely useful.

A simple solution would be:
cuFFT interface to FFTW which would accept device pointer.



Issues with and pgc++ and Cray Environment

Compiler

- Host_data give internal compiler error for class members
  Solution: use temporary pointers out of the class.

Cray Wrapper

- Hard to deal with in a CPU/GPU Hybrid environment.
  - Module loading leads to automatic defines / includes / links that are unwanted, especially in a CPU-only version meant for reference purposes
  - Example: _OPENACC macro is defined even without –acc flag passed to CC => Standard way to decide between CPU +GPU version breaks.

Speed down and up