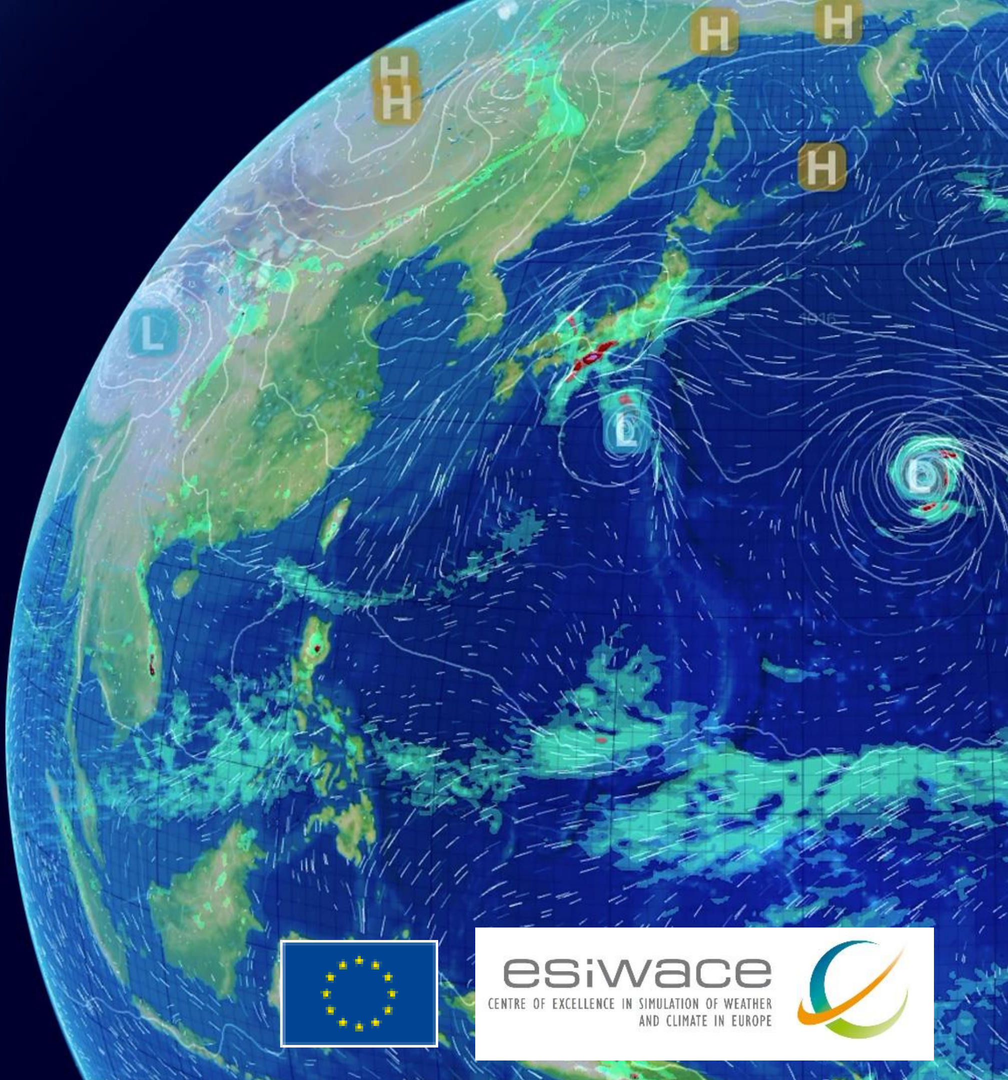
Met Office

LFRic and PSyclone: Utilising DSLs for performance portability

Iva Kavcic, Met Office, UK &

Rupert Ford, Andrew Porter, Sergi Siso (STFC, UK); Joerg Henrichs (BOM, AU); Christopher Maynard and Sam Cusworth (Met Office, UK)

ESiWACE2 Virtual Workshop on Emerging Technologies for Weather and Climate Modelling, 30 June 2020



esiwace
CENTRE OF EXCELLENCE IN SIMULATION OF WEATHER AND CLIMATE IN EUROPE

Met Office

Overview

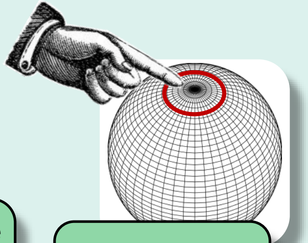
What is LFRic and how it talks to PSyclone

Utilising API information for performance

Summary

In the beginning: GungHo project

Need to make porting the codes from machine to machine easier: Flexible Implementation



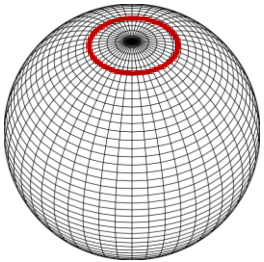
Need a more scalable dynamical core

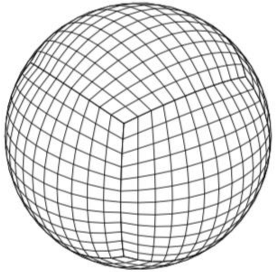
New Infrastructure – LFRic

New Dynamical Core – GungHo 工合

- Met Office, NERC (UK Universities) and STFC collaboration (2010 – 2015)
- Future architectures - MPI? OpenMP? Accelerators? GPUs? ARM? …?
- Increased resolution → exascale computation
- Scientific code can be ~ 10^6 lines (of Fortran)
- Complex parallel code + Complex parallel architectures + Complex compilers = Complex optimisation space => no single solution
- Single source science code & Performance portability(?)

LFRic/GungHo: Preparation for exascale





Unified Model (UM) & ENDGame dynamical core

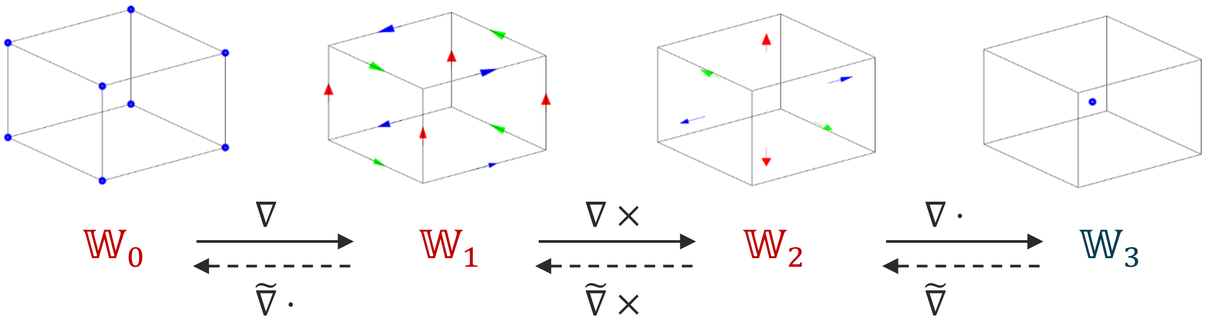
- Staggered Finite Differences
- Fully structured Lat-Lon mesh
- Hard-coded optimisations

LFRic system & GungHo dynamical core

- *Mixed Finite Elements*
- Horizontally unstructured, vertically structured quasi-uniform mesh
- Generated optimisations

Met Office

Mixed FEM (continuous + discontinuous)



Lowest order (k = 0) reference element

Shared dofs (degrees of freedom): W_0 (all), W_1 (tangential components) & W_2 (normal components)

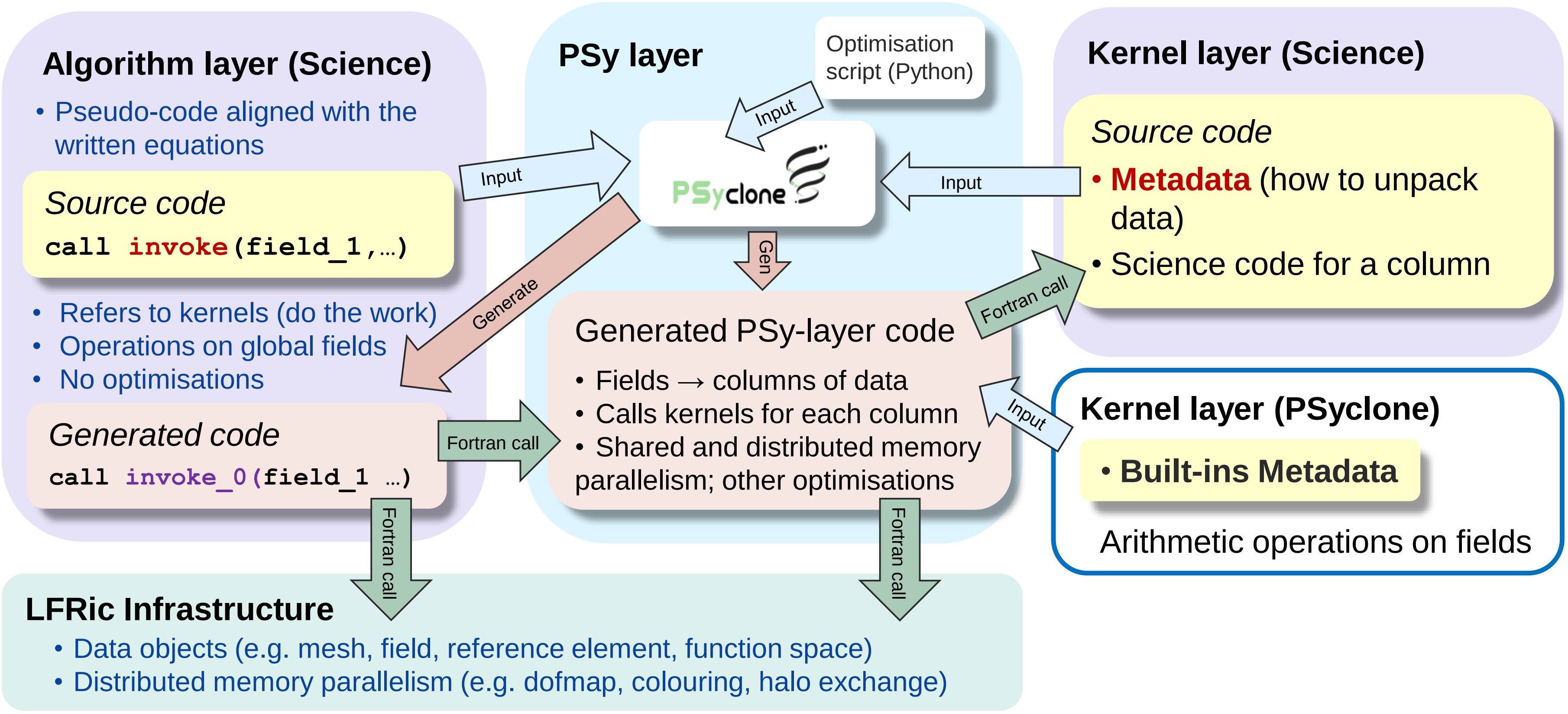
No shared dofs: W_3

LFRic infrastructure: Hierarchy of objects

- **Field**
- **Function Space**
- **Mesh (3D)**
- **Partition**
- **Global Mesh (2D)**

Mesh (3D) → **Reference element** → **FS: Dofmap**

Separation of concerns in LFRic (PSyKAl): Parallel-Systems, Kernels, Algorithms



Use of “PSy + clone” in LFRic

- **Optimisations:** generates optimised PSy-layer code *(and soon transformed kernels – work in progress).*
  - Optimisations encoded as a ‘recipe’ rather than baked into the scientific source code.
  - Different recipes for different architectures.
- **Development**
  - Generates **kernel stubs** (argument declarations and ordering).
  - **fparser2** (F2003-2008) parses LFRic code (also tested on the Unified Model) and base for the **LFRic code style checker**.
- **Tools (profiling, DataAPI)**
  - Insert calls to profiling tools (interface in PSyclone) – tested (and used) in LFRic.
  - Extract data for running smaller code units as stand-alone applications (micorbenchmarks) – work in progress.

Rules of engagement: LFRic ⟷ PSyclone LFRic API

LFRic

- **Rules** (e.g. writing algorithms & kernels, data properties)
- Kernel **metadata**
- **Infrastructure support** (e.g. dofmap, colouring, MPI)

LFRic API

- Flexible **optimisations**
- Support for data structures & properties following **API rules**
- Development mandated by requirements

PSyKAl-lite

Required functionality (hand-written)

Distributed + shared memory parallelism

PSyclone transformations

- Dynamo0p3ColourTrans
- Dynamo0p3OMPLoopTrans
- OMPParallelTrans

⇩

LFRic support

- Halo exchange
- Colouring (cell & halo) for fields on continuous FS

…

LFRic: Kernel code

Written by Scientists (Fortran)

Metadata tells PSyclone how to unpack data

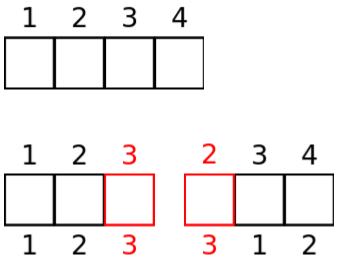
```
module rtheta_kernel_mod
…
  type, public, extends(kernel_type) :: rtheta_kernel_type
    private
    type(arg_type) :: meta_args(4) = (/                               &
         arg_type(GH_FIELD, GH_READWRITE, Wtheta),                    &
         arg_type(GH_FIELD, GH_READ,      ANY_DISCONTINUOUS_SPACE_1), &
         arg_type(GH_FIELD, GH_INC,       W2),                        &
         arg_type(GH_FIELD, GH_READ,      ANY_SPACE_1),               &
         /)
    type(func_type) :: meta_funcs(2) = (/                             &
         func_type(Wtheta, GH_BASIS, GH_DIFF_BASIS),                  &
         func_type(W2,     GH_BASIS, GH_DIFF_BASIS)                   &
         /)
    integer :: iterates_over = CELLS
    integer :: gh_shape = GH_QUADRATURE_XYoZ
  contains
    procedure, nopass ::rtheta_code
  end type
…
```

Science code for a column of *nlayers* levels

```
…
  subroutine rtheta_code( … r_theta, heat_flux, u, &
…sizes, maps, basis functions,
 quadrature points for all function spaces )
…
    real(kind=r_def), dimension(undf_wtheta), &
                      intent(inout) :: r_theta
    real(kind=r_def), dimension(undf_2d),     &
                      intent(in)    :: heat_flux
    real(kind=r_def), dimension(undf_w2),     &
                      intent(in)    :: u
…
    do k = 0, nlayers-1
      do df = 1, ndf_wtheta
        r_theta( map_wtheta(df) + k ) =  &
        r_theta( map_wtheta(df) + k ) - rtheta_e(df)
      end do
    end do
…
  end subroutine rtheta_code
end module rtheta_kernel_mod
```

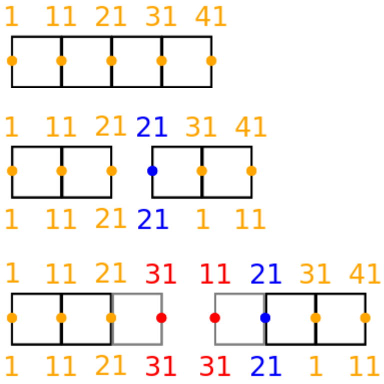
Met Office

Updating fields on continuous vs discontinuous spaces



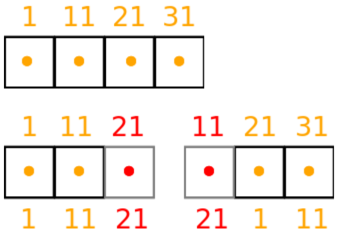
Continuous fields (cell loop)

- Dofs on **owned** cells + redundant computation in the level-1 **halo**
- GH_INC (R + W) access – requires colouring for OpenMP
- *Kernel code*



Continuous fields (dof loop)

- **Owned** dofs or redundant computation into **annexed** dofs (configuration option)
- *Built-in code*



Discontinuous fields

- *Cell loop:* dofs on **owned** cells (redundant computation optional)
- *Dof loop:* **owned** dofs
- GH_READWRITE (R + W) or GH_WRITE (W) access – no colouring required for OpenMP
- *Kernel code*

Met Office | Science & Technology Facilities Council | esiwace CENTRE OF EXCELLENCE IN SIMULATION OF WEATHER AND CLIMATE IN EUROPE

Example of optimisation based on API knowledge: Dynamics

- **Dynamical core run, baroclinic test case** with four multigrid resolutions and 30 vertical levels.
- Cray XC40/XCS, Aries interconnect, dual socket 18-core Broadwell Intel Xeon node, i.e. 36 cores per node.
- PSyclone configurations: **A**NY_SPA**C**E (general function space, assumes **c**ontinuity); **A**NY_**D**ISCONTINUOUS_SPACE (no shared dofs); **AD** and **R**edundant **C**omputation.
- Columns per task: 1536 for the base MPI run and 768 for 2 x MPI tasks.
- Reduction in number of halo exchanges: **≈ 10% for AD** (no significant change in runtimes) and **≈ 57% for AD+RC**.

MG Resolution	MPI tasks in base run	AD+RC (MPI)	AD+RC (1/2 x MPI, 2 OMP)	AD+RC (2 x MPI)	AD+RC (MPI, 2 OMP)
C48 (≈ 200 km)	18	2.31	NotMul6	2.91	3.71
C96 (≈ 100 km)	72	1.50	2.89	4.75	3.26
C192 (≈ 50 km)	288	2.34	3.58	5.24	6.96
C384 (≈ 25 km)	1152	7.60	3.11	SubFail	5.78

Speed-up (%) compared to AC runtimes for each configuration.

Example of optimisation based on API knowledge: Physics

- LFRic utilises existing parameterisation schemes for UM *(until they need to be refactored).*
- **Interfacing** ("Physics" kernels) **UM Physics**, **SOCRATES** ("Suite Of Community RAdiative Transfer codes") and **JULES** (land surface) models.
- Consequences: "stuck" with data layout and hard-coded OpenMP implementation for UM, **limited scope for optimisation** (e.g. for now not safe to call UM Physics with more than 1 thread).
- **PSyclone: ANY_DISCONTINUOUS_SPACE** metadata in LFRic "Physics" kernels and **redundant computation** switch to **reduce communication costs** (number of halo exchanges).

Halo exchanges	C48: 6*48*48 columns, ≈ 200 km in horizontal			C96: 6*96*96 columns, ≈ 100 km in horizontal		
Application	**AC**	**AD**	**AD+RC**	**AC**	**AD**	**AD+RC**
lfric_atm	324002	285919 (88.24 %)	150649 (46.49 %)	354851	312403 (88.03 %)	164024 (46.22 %)
slow_physics	5997	576 (9.6 %)	0	5997	576 (9.6 %)	0
fast_physics	15961	14046 (88 %)	10430 (65.35 %)	16009	14094 (88.03 %)	10454 (65.3 %)

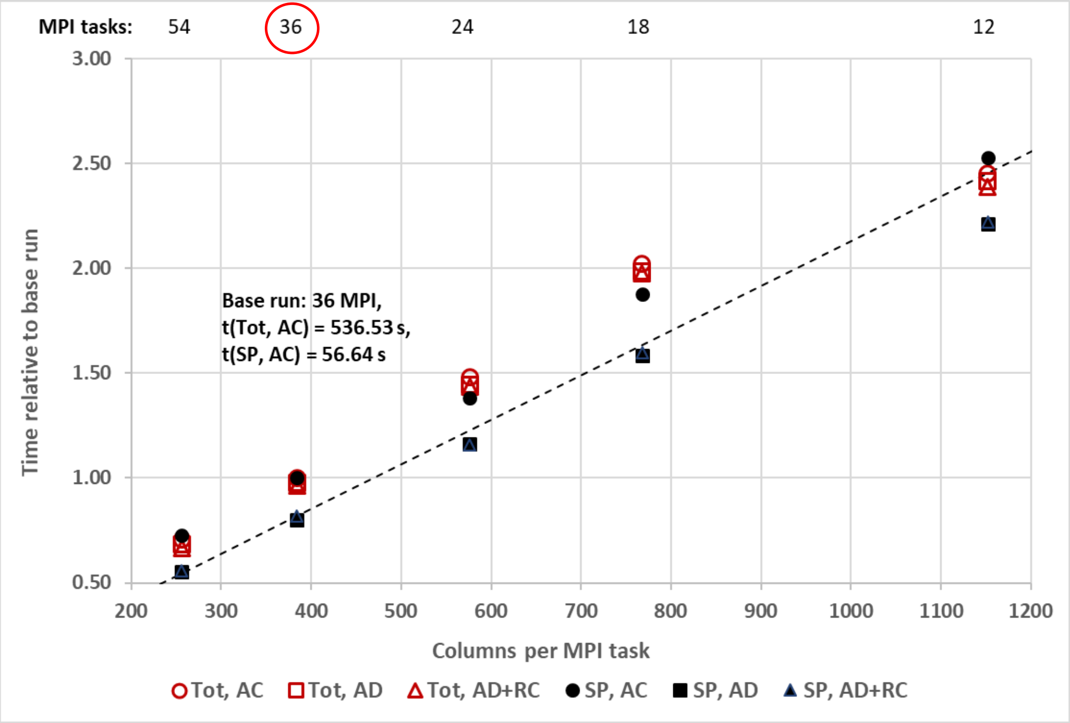
Aquaplanet runs: 38 vertical levels, dt = 300 s, multigrid solver (4 levels)

ANY_SPA**C**E (general function space, assumes **c**ontinuity); **A**NY_**D**ISCONTINUOUS_SPACE ("knows" that most Physics schemes are cell-volume based, no shared dofs); **AD** and **R**edundant **C**omputation into annexed dofs.

Met Office

Aquaplanet run, C48 MG ($6*48^2$ columns ≈ 200 km hor; L38; dt = 300 s)

(Cray XC40/XCS, Aries interconnect, dual socket 18-core Broadwell Intel Xeon node, i.e. 36 CPU cores per node)



Speed-up (%) compared to runtime of AC Physics metadata runs (**Tot**al, **S**low **P**hysics and **F**ast **P**hysics runtimes)

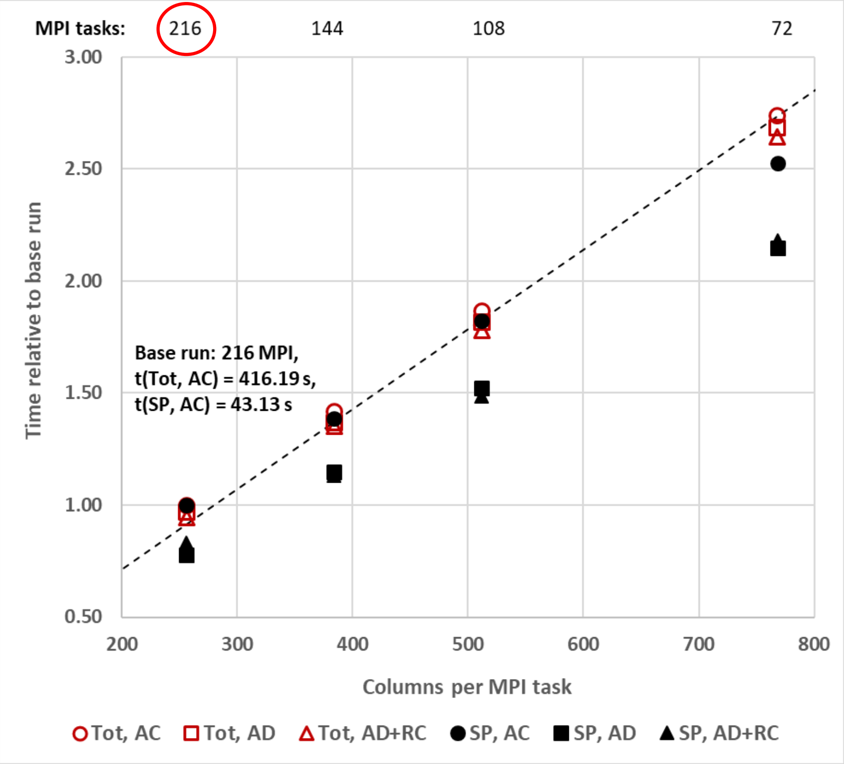
Col per task	Tot, AD	SP, AD	FP, AD
256	2.83	23.84	2.56
384	2.46	20.09	3.35
576	2.44	16.03	1.16
768	1.82	15.58	1.89
1152	1.53	12.61	1.79

Col per task	Tot, AD+RC	SP, AD+RC	FP, AD+RC
256	5.63	23.40	6.48
384	3.62	18.22	4.46
576	3.28	16.09	2.42
768	2.38	14.97	3.19
1152	2.72	12.11	2.26

Met Office

Aquaplanet run, C96 MG ($6*96^2$ columns ≈ 100 km hor; L38; dt = 300 s)

(Cray XC40/XCS, Aries interconnect, dual socket 18-core Broadwell Intel Xeon node, i.e. 36 CPU cores per node)



Speed-up (%) compared to runtime of AC Physics metadata runs (**Tot**al, **S**low **P**hysics and **F**ast **P**hysics runtimes)

Col per task	Tot, AD	SP, AD	FP, AD
256	3.25	22.14	4.37
384	3.42	17.15	3.80
512	2.60	16.62	3.04
768	1.98	15.00	2.75

Col per task	Tot, AD+RC	SP, AD+RC	FP, AD+RC
256	5.41	16.81	4.94
384	4.70	18.44	3.73
512	4.53	18.59	4.40
768	3.40	13.80	4.06

Met Office

Summary and (some) work in progress

- Separation of concerns → flexible optimisations
- Flexible optimisations + API knowledge → performance improvements

Work in progress (and future)

- Implement kernel transformation and optimisation
- Kernel extraction (creating microbenchmarks)
- Portability
  - Run LFRic with GPU (start with OpenACC from PSyclone NEMO API)
  - LFRic container tests on PizDaint (Broadwell - baseline) and Hawk (AMD)

Met Office

Questions?

Acknowledgements

LFRic team, GungHo Atmospheric Science team and other LFRic developers

esiwace
CENTRE OF EXCELLENCE IN SIMULATION OF WEATHER AND CLIMATE IN EUROPE

ESiWACE2 has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 823988

Links and references

- LFRic: https://code.metoffice.gov.uk/trac/lfric/wiki
- LFRic container recipes (to be hosted on https://github.com/MetOffice):
  - https://github.com/eth-cscs/ContainerHackathon/tree/master/LFRIC
  - https://github.com/tinyendian/lfric_reader
- PSyclone and fparser
  - https://github.com/stfc/PSyclone
  - https://psyclone.readthedocs.io
  - https://github.com/stfc/fparser
  - https://fparser.readthedocs.io
- PSyclone in LFRic: https://code.metoffice.gov.uk/trac/lfric/wiki/PSycloneTool
- GHASP (GungHo Atmospheric Science): https://code.metoffice.gov.uk/trac/lfric/wiki/GungHoScience
- stylist: https://github.com/MetOffice/stylist

- Adams et al. (2019), *LFRic: Meeting the challenges of scalability and performance portability in Weather and Climate models*, Journal of Parallel and Distributed Computing,132, 383-396