Developing the NEMOVAR ocean DA system to run on GPU based computers

Martin Price[1], David Case[2], Marcin Chrust[3], Wayne Gaudin[4], Matthew Martin[1], Davi Mignac[1], David Norton[4], Andrea Piacentini[5], Andrew Porter[6], Anthony Weaver[5]

1. Introduction

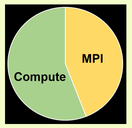
GPUs (Graphical Processing Units) have significant potential benefits in geophysical modelling applications because in the right use cases they offer high power, space and cost efficiency. Their key advantages are much higher memory bandwidth (theoretically of order 5 times higher) - compared with main system memory, and thousands of arithmetic units; so that for an optimal application a GPU can achieve much greater throughput than a CPU socket, with comparable power consumption.

The NEMOVAR ocean data assimilation system is used in the operational forecasting systems at both Met Office and ECMWF, and in other applications including ocean reanalyses. To investigate the potential to run NEMOVAR on GPUs, a series of hackathons have been held since September 2019 including participants from Met Office, CERFACS, ECMWF, NCAS, NVIDIA and STFC (with NCAS and STFC involvement supported by the ExCALIBUR project).

2. NEMOVAR computational characteristics

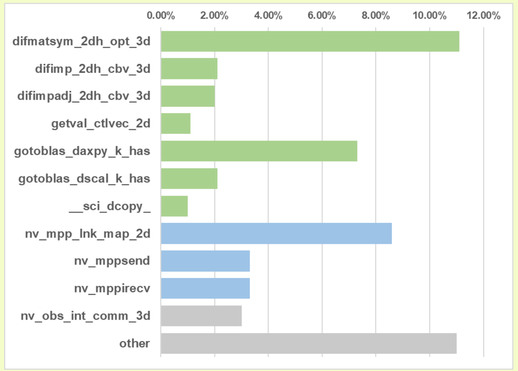
NEMOVAR uses an implicit diffusion operator to model background error covariances, and this is by far the most computationally expensive part of the 3D-Var system.

A profile of NEMOVAR on CPU reveals that roughly 55% of time is spent on compute while 45% is MPI message passing (pie chart, right).



The bar chart (below) shows how the compute time further breaks down into individual routines.

The green bars are all components of the diffusion operator; while the blue bars are MPI support routines also most heavily used by the diffusion operator. Together these account for around 75% of the compute time.



The diffusion operator routines are thus an obvious target for GPU acceleration.

The grey bars have not yet been targeted for GPU acceleration; the 'other' category includes a large number of routines each accounting for less than 1% of the total compute time.

During late 2020 following analysis at the hackathons the diffusion routines were optimized on CPU, reducing total compute time by around 14%

3. Approaches to porting

On a system with suitable GPUs and compiler, code can be ported by adding directives to target routines. We have mainly used NVIDIA V100 GPUs and the NVIDIA Fortran compiler, where either OpenACC or OpenMP directives can be used. We have used OpenACC but conversion to and from OpenMP should be simple if required.

The code segment (right) shows the OpenACC KERNELS directives which tell the compiler to run the code on GPU; and DATA directives (yellow highlight) which specify data to be transferred between CPU and GPU memory.

```
!$ACC DATA &
!$ACC COPYIN (p_dif,p_dif%npi, p_dif%npj, p_dif%difmskii, p_dif%difmskjj) &
!$ACC COPYIN (p_dif%coflapii, ,p_dif%coflapjj, p_dif%k11, p_dif%k22) &
!$ACC COPYIN (p_fld) &
!$ACC COPY(z_derj,z_deri)
!$ACC KERNELS
DO jj = 1, p_dif%npj - 1
   DO ji = 1, p_dif%npi - 1
      z_deri(ji,jj) = p_dif%coflapii(ji,jj,kk)                          &
         &          * p_dif%K11(ji,jj,kk) * p_dif%difmskii(ji,jj,kk) &
         &          * ( p_fld(ji+1,jj  ) - p_fld(ji,jj) )
      z_derj(ji,jj) = p_dif%coflapjj(ji,jj,kk)                          &
         &          * p_dif%K22(ji,jj,kk) * p_dif%difmskjj(ji,jj,kk) &
         &          * ( p_fld(ji  ,jj+1) - p_fld(ji,jj) )
   END DO
END DO
!$ACC END KERNELS
!$ACC END DATA
```

This is an **explicit memory management** approach which is flexible and powerful, but depends on the developer to ensure correct transfers of data between system and GPU memory. Memory management is often the bigger part of the task, and if incomplete will lead to runtime errors or incorrect results.

Managed memory is an alternative approach provided by the NVIDIA GPU driver, which manages memory transfers at runtime. There are some limitations, for example arrays and derived types must be ALLOCATABLE to be handled. Performance is potentially slightly lower with managed memory, but on the other hand the development task is simpler allowing more routines to be ported.

The **PSyclone code-generation and translation system** is being adapted for the NEMOVAR code (and others) as part of the ExCALIBUR project. PSyclone inserts the required GPU directives as a pre-processing step, a very powerful and flexible approach that could reduce development and maintenance effort, and increase portability (for example between different directive languages or types of accelerator hardware). PSyclone's handling of NEMOVAR is at an early stage with work ongoing, for example required support for Fortran derived types has recently been added.

4. Progress and performance

As of May 2021 almost all of the parallel code in the diffusion operator for 3D variables is running on GPU. Large computational sections are 'resident' – meaning data transfers back and forth to the GPU have been minimized. Achieving residency is typically the first optimization step, because data transfers are relatively slow. The diffusion operator for 2D variables is not yet ported, this is the next step to get the whole diffusion operator fully resident.

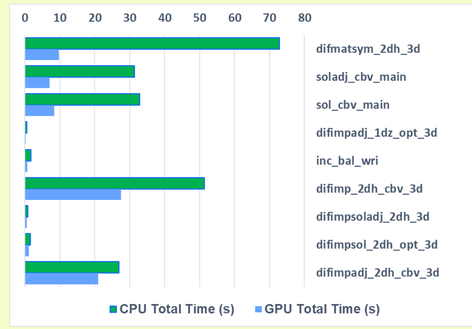
The table (right) shows NEMOVAR runtime for a 1 degree ocean test case, comparing a V100 GPU with a single Intel XEON core. Total speedup for this test case is 1.8x, but most of the initialization has not been accelerated – excluding this phase speedup is 2.3x.

Platform	Total	Initial.	Rest
XEON 6230 CPU core	322	107	215
Volta V100 GPU	177	85	92
Speedup	1.8	1.3	2.3

5. Progress and performance (continued)

So far GPU performance as a whole is modest however optimization is at an early stage. Initial work with the diffusion stencil calculations achieved a speedup of up to 5x an 18 core CPU **socket**. Below is CPU vs GPU time for the routines with greatest speedup.

The figure (right) shows total GPU vs CPU time, for the same 1 degree ocean run, for individual routines, ordered by GPU speedup. The greatest speedup of around 7.5x, is for the core diffusion stencil operations for 3D variables. This, together with the three other routines showing 2x speedup account for almost all the speedup seen so far.



Kernel	Tot Time (s)	Avg Time (us)	Mem read (GB/s)	Mem write (GB/s)	Mem total (GB/s)
difmatsym_L2049	2.64	10.7	324.4	80.1	404.5
difmatsym_L2063	934.1	3.78	2.4	17.4	19.8

The table (left) shows time and **memory bandwidth** for the two KERNELS regions in difmatsym_2dh_3d – the fastest and most time consuming routine.

Memory bandwidth is a key performance measure on GPUs, with a peak available of around 700 GB/s on the V100 Volta GPUs being used. The first KERNEL listed reaches total 404.5 GB/s which is reasonable; however the second only reaches 19.8 GB/s. This shows a new target for optimization, and likely indicates that even our fastest routine should be faster.

6. Conclusions and further work

An initial port of NEMOVAR to GPUs is under way and has made good progress, with much of the diffusion operator now running on GPU. Performance so far is modest, but optimization is at an early stage.

Work is ongoing with the following aims during 2021:

- Complete porting the NEMOVAR implicit diffusion operator to achieve full residency in GPU memory.
- Continue optimization of kernels to minimize their runtime (using runtime and memory bandwidth as a guide).
- Test a larger configuration, most likely a 0.25 degree ocean grid (ORCA025) using multiple GPUs.

Once this work is complete we expect to be able to draw some firmer conclusions about the performance potential of NEMOVAR on GPUs.

[1]**Met Office** FitzRoy Road, Exeter, Devon, EX1 3PB United Kingdom. Tel: +44 370 900 0100
Email: martin.price@metoffice.gov.uk

[2]**National Centre for Atmospheric Science** Reading, United Kingdom

[3]**European Centre for Medium Range Weather Forecasting** Reading, United Kingdom

[4]**NVIDIA** Santa Clara, California, USA

[5]**CERFACS / CECI CNRS UMR 5318** Toulouse, France

[6]**Hartree Centre STFC Daresbury Lab**, Warrington United Kingdom