Working Group 1

Notes WG 1

- Goal to derive:

  joint community efforts

  ECMWF focus recommendations

I/O, Data Management and Scalability

- Trends and Pressure points
  - 1.5x increase per year in storage; 1/3 from higher resolution, 2/3 from increased variety
  - Moving to ensembles will help computational scaling but will mean more model data volume compared to deterministic runs
  - Volumes of satellite data will level off in ~10 years; model data will continue to increase
  - Improvement in storage cost will continue to improve – Flash still benefiting from Moore's law; tape is not dead yet.
  - Bandwidth to/from storage not scaling as fast as storage and WAN capacity
  - Recommend organized effort to gather real data of data and I/O requirements and how they will scale
- Approaches for reducing data
  - I/O server approaches improve model performance but data still goes to disk and storage
  - Store model history at decreased resolution
  - Store ensemble statistics (PDFs) instead of ens. output
  - Temporal slicing/reduction ; how data is stored (as fields, currently)
  - Compression
  - Automate curation of data and experimental data
- Take processing for pre-/post-processing off critical path of forecast systems and stream between applications. Only write to disk for fault tolerance
- On-the fly post processing, compression, analysis?
  - If it can be done concurrent with (and not impede) critical path
  - If the analysis tasks are known a-priori – for climate they usually are not
  - Store v. recomputed? B.L.'s analysis showed recomputing is 1.67x more expensive than storage (for now)

Numerics and Scalability

- Application tuning
  - Tuning of non-library application code over range of settings laborious and sometimes not done at all. Autotuning frameworks.
  - Unit testing and automated generation of unit tests from applications facilitates detaile profiling for improving flop/s and flops/W (and better software in general
- Low level optimization not sufficient; must look at algorithms
  - Parallel in Time for DA and models
  - Need to revisit, find new algorithms that may have higher operation counts but more locality and less data movement: Spectral Element, Disc. Galerkin, Finite Element Methods
  - Horiz. Explicit/Vert. Implicit (HEVI)
  - Tridiagonal schemes in vertical do not vectorize
- Bit reproducibility may be sacrificed for fault tolerance (e.g. Fault Tolerant Linear Solvers, Mark Hoemmen, Sandia NL)
- Libraries and Frameworks
  - Algorithmic updates to models on a 10 year cycle; can reduce with modular design and supporting infrastructure (e.g. OOPS) but time for testing and acceptance remains fixed
  - PETSc & others – using these packages leverages these efforts and efforts of vendors (NVIDIA, Cray, IBM) to tune for performance and scaling

Hardware/Compilers

- Hardware requirements and co-design dominated by desktop, gaming, and laws of physics
- What can be done regarding power:
  - User control of frequency, power saving modes, with improved vendor supplied tools.
- Incomplete support across for OpenACC, Vectorization, and CAF – affects performance portability
- Dynamic task parallelism. It’s available but need attn. to load imbalance; research topic
- Memory/core and per node: new developments in memory architectures, more information next year.
- OpenACC is here to stay; not clear about combination with OpenMP

Benchmarking

- Metrics for CPU-accelerator comparisons
  - Socket to socket
  - Node to node (2 CPU vs. 1 CPU + 1 GPU)
  - Power envelope
  - Run time is bottom line
- Error resilience
  - Capacity jobs (ensemble) are less of an issue. Loss of ensemble member is recoverable; Mostly a capability issue.
  - Users need to take more active role
  - Fault-tolerant MPI for detection and handling of node failures at application level
  - Checkpointing won’t scale (neither OS nor App level) but Flash memory may help.
  - Detect bad patch at run time and just fill-in from neighbors
  - Numerical algorithms that are fault tolerant.
- Code profiling
  - Important to profile at scale
- Trade-off between productivity and performance
  - People cheaper than power (“pasta cheaper than coal”)
  - Investment in people versus the HPC budget

2. workflows

- Workflows:
  - NWP and climate difference, shelf live longer for climate + time window not as constraint.
  - Lack of focus on workflow so far
  - Assimilation 80% is in the model
  - Processing of observations
  - Grib2 vs netcdf
  - Exascale in time processing streaming data and processing it, existing project at ECMWF
  - Projection is that observational data amount is increasing
  - Not bottleneck at the moment to deal with observations, but remove from critical path
  - Parallel in time for assimilation (helps by one order of magnitude) versus ensembles (increases data and I/O problem) keep options open
  - I/O bottleneck, 4dvar lower resolution models, link to point 7 how to test/benchmark
  - Do we expect models continue to increase in resolution (general point)
  - Issue with inner loop in assimilation
  - Workflows: climate ?
  - Hardwiring in NWP, more flexibility in data formats/definition of parameters, portability important for research,
  - Workflow automation system ? Area of collaboration ? Rose and silk , workflows are unique but tools may be more commonly used, silk from New Zealand met office , ecflow and SMS from ECMWF, do not use file I/O between tasks in workflow but pipeline and stream data unexplored terrain.
  - Error resilience ?

General

- Share components:
  - communicate what we do better, than there is more opportunity to share
  - Dwarf implementations
  - Workflow tools
  - Dynamical core
  - Strategies what works and what not
  - Standards on software development
  - Open source developments, even compilers ?
  - Collection of software requirements from all weather centres to address vendors believed to work
  - Cray: Fortran to stay
  - Fortran community shrinking ? Tools and compiler support problematic ?
  - Library interfaces not necessarily in Fortran, no longer at university

4. Libraries

- Implicit solver, powerful parallel libraries available (PETSC, DUNE) may solve this problem, but not for NEMO if grid choice rigid ? Algebraic multigrid
- Preconditioning can be special but are supported in libraries
-