The end of climate modelling as we know it

Bryan Lawrence

NCAS &
University of Reading: Departments of Meteorology and Computer Science

UoR, 01 Mar 19

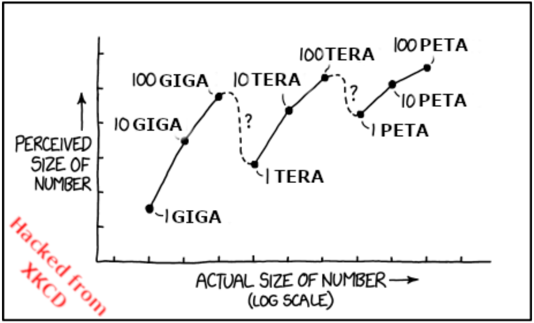
University of Reading

NERC SCIENCE OF THE ENVIRONMENT

Outline

- Expectation: New science can be done based on ever increasing compute resources
- Moores's Law: Delivered ever increasing compute, but it's nearly over
- Post-Moore's Law: We have to be smarter!
- Kryder's Law: is failing us too: We have to be smarter!
- Avoidance: Documentation to avoid duplication of effort

Climate Scientists don't *respect* big numbers!



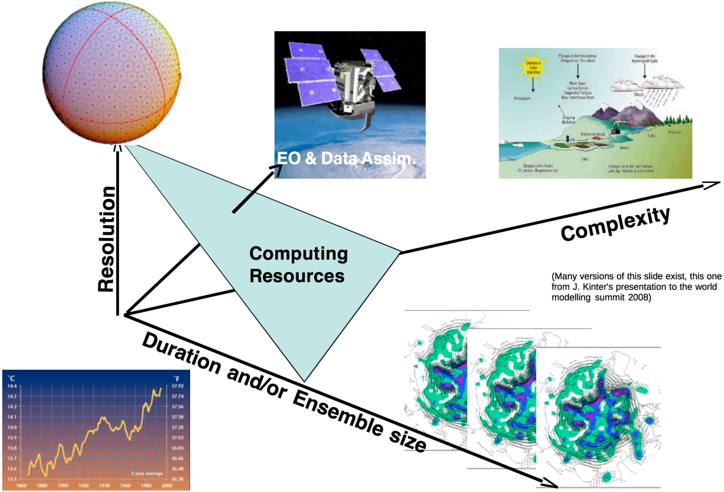
TALKING ABOUT LARGE NUMBERS IS HARD

In experimental design, many underestimate:

- The energy demands and costs of computing associated with their experiments, and
- The difficulty in managing, disseminating, and utilising large volumes of data!

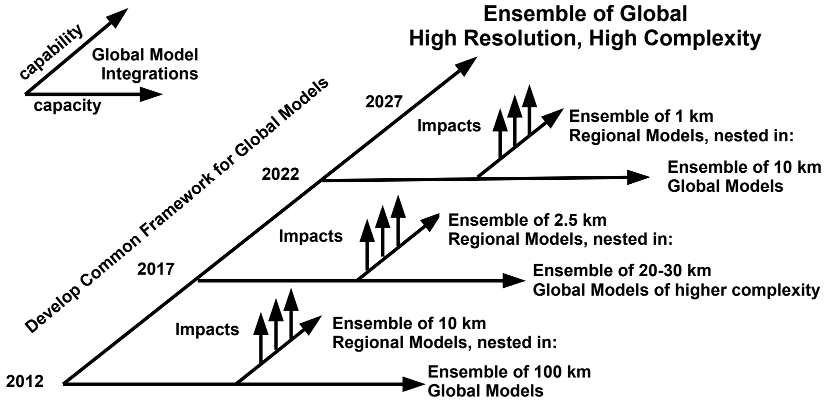
This is only going to become worse unless we do something about it — but this is not a popular message!

Give me more computing?



(Many versions of this slide exist, this one from J. Kinter's presentation to the world modelling summit 2008)

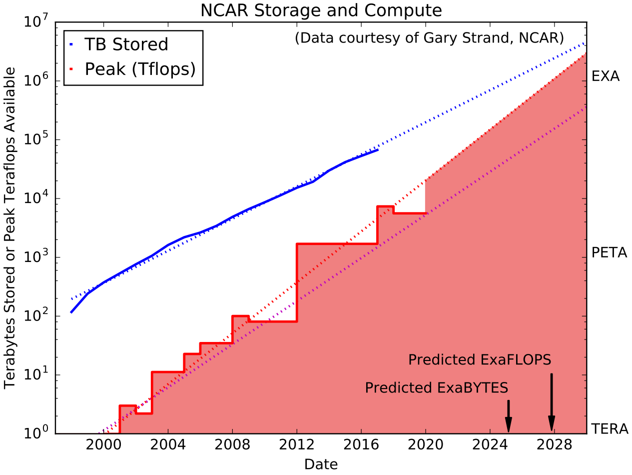
Climate Goals

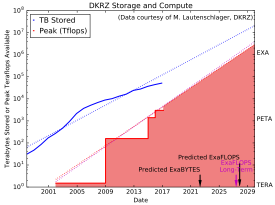


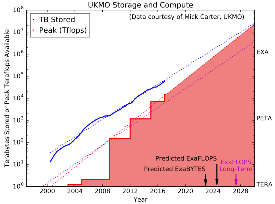
enes EUROPEAN NETWORK FOR EARTH SYSTEM MODELLING

(From "Infrastructure Strategy for the European Earth System Modelling Community" 2012-2022, Mitchell et al, 2012.)

History has given us exponential compute linked to exponential data …

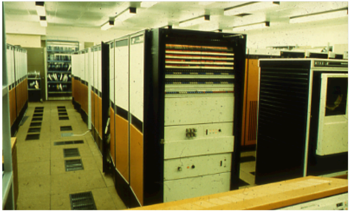






Faster Compute

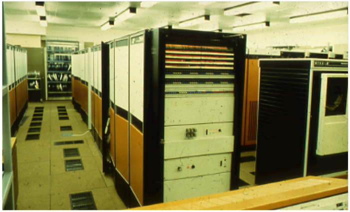
1981: ICL Dist.Array.Proc. (20 MFlops)



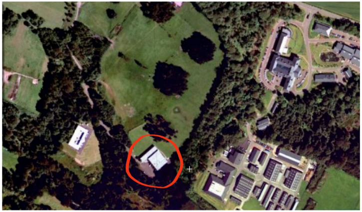
2014: Archer (then 1.4 PFlops)

Faster Compute

1981: ICL Dist.Array.Proc. (20 MFlops)



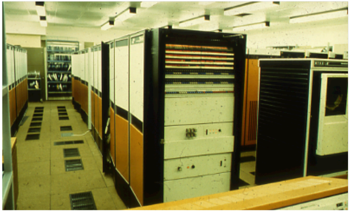
EPCC Advanced Computing Facility, 2014



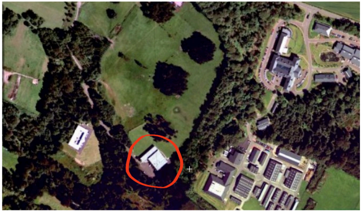
2014: Archer (then 1.4 PFlops)

Faster Compute

1981: ICL Dist.Array.Proc. (20 MFlops)



EPCC Advanced Computing Facility, 2014



2014: Archer (then 1.4 PFlops)

From 1981, without Moore's Law

Slide content courtesy of Arthur Trew: archer | epcc |

Moore's Law and Friends

Moore's Law

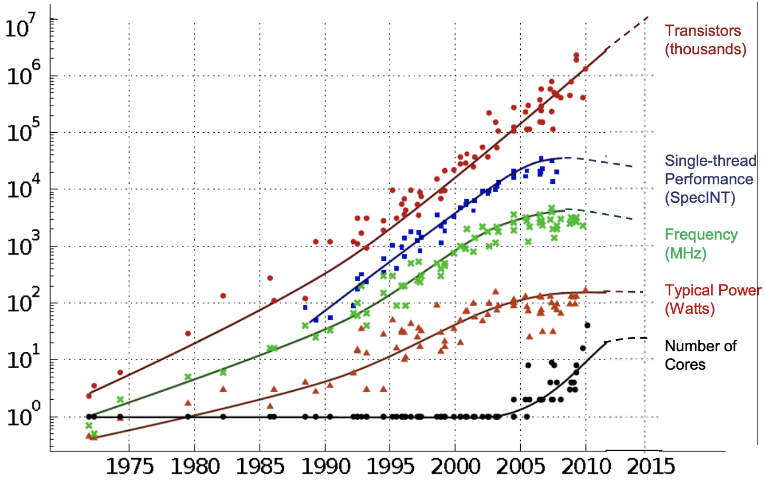
More often misquoted and misunderstood:

- Original, Moore, 1965: The complexity for minimum component costs has increased at a rate of roughly a factor of two per year.
- House (Intel) modified it to note that: The changes would cause computer performance to double every 18 months
- Moore (Modified 1975): The number of transistors in a dense integrated circuit doubles about every two years

Dennard Scaling

- The performance per watt of computing is growing exponentially at roughly the same rate (doubling every two years).
- (Increasing clock frequency as circuits get smaller, but this stopped working around 2006, too much power too small, means meltdown!)

The end of Dennard Scaling



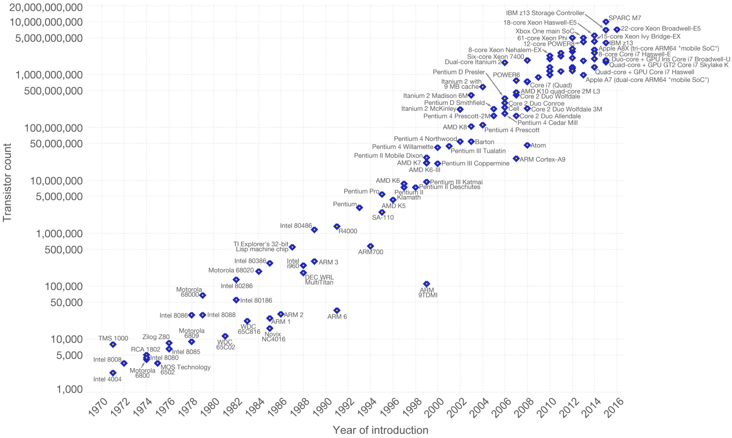
Original data collected and plotted by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond and C. Batten

Moores's Law

Moore's Law – The number of transistors on integrated circuit chips (1971-2016)

Our World in Data

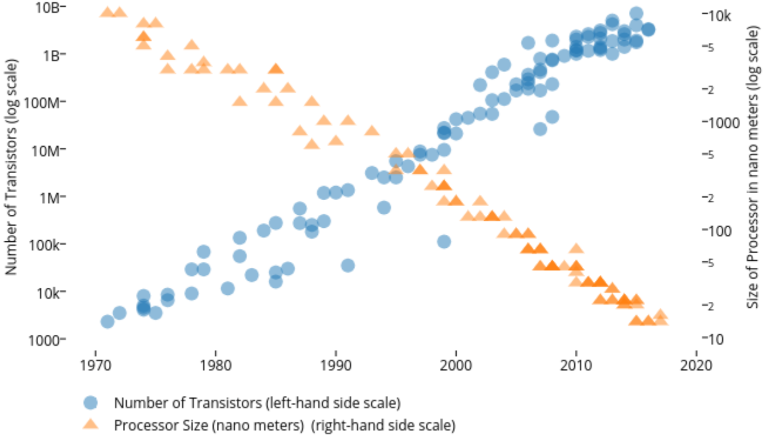
Moore's law describes the empirical regularity that the number of transistors on integrated circuits doubles approximately every two years. This advancement is important as other aspects of technological progress – such as processing speed or the price of electronic products – are strongly linked to Moore's law.



Data source: Wikipedia (https://en.wikipedia.org/wiki/Transistor_count)
The data visualization is available at OurWorldinData.org. There you find more visualizations and research on this topic.

https://en.wikipedia.org/wiki/Transistor_count

Moores's Law



https://www.yaabot.com/31345/quantum-computing-neural-chips-moores-law-future-computing/

Moore's 2nd Law aka Rock's Law

- The cost of a semiconductor chip fabrication plant doubles every four years.
- Noyce, 1977: "…further miniaturization is less likely to be limited by the laws of physics than by the laws of economics."

The Register® Biting the hand that feeds IT

NTRE SOFTWARE SECURITY DEVOPS BUSINESS PERSONAL TECH SCIENCE

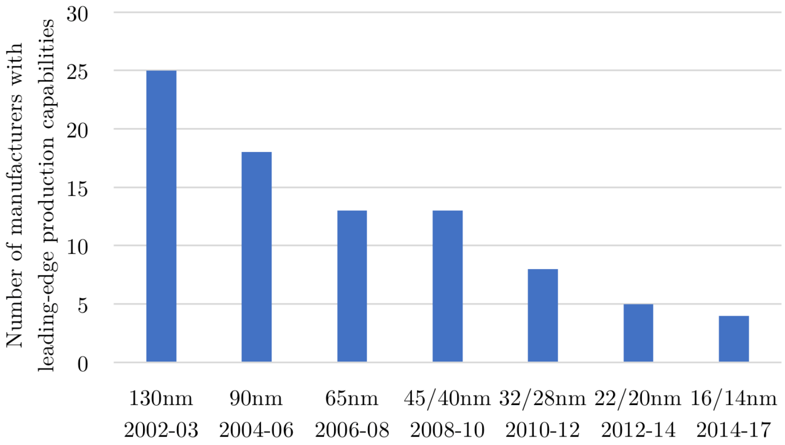
Personal Tech

GlobalFoundries scuttles 7nm chip plans claiming no demand

AMD promptly dumps it and hires TSMC for next-gen chips

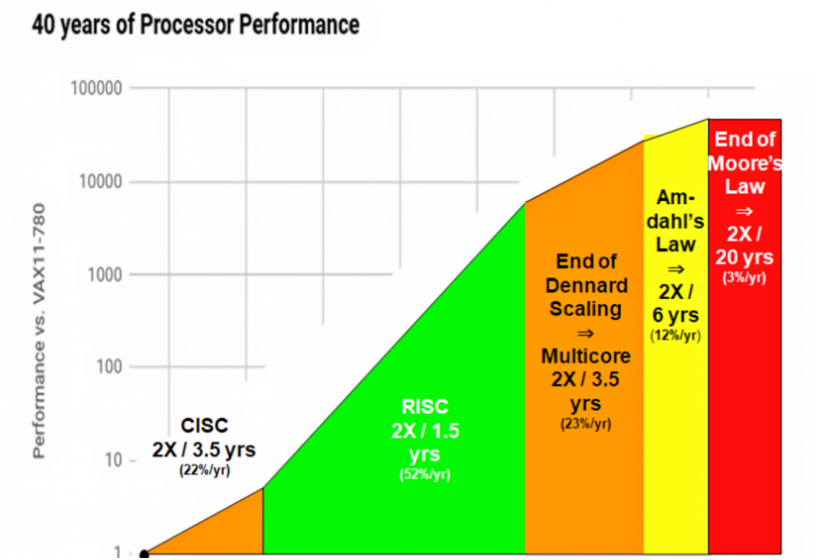
By Shaun Nichols in San Francisco 27 Aug 2018 at 23:55 18 SHARE ▲

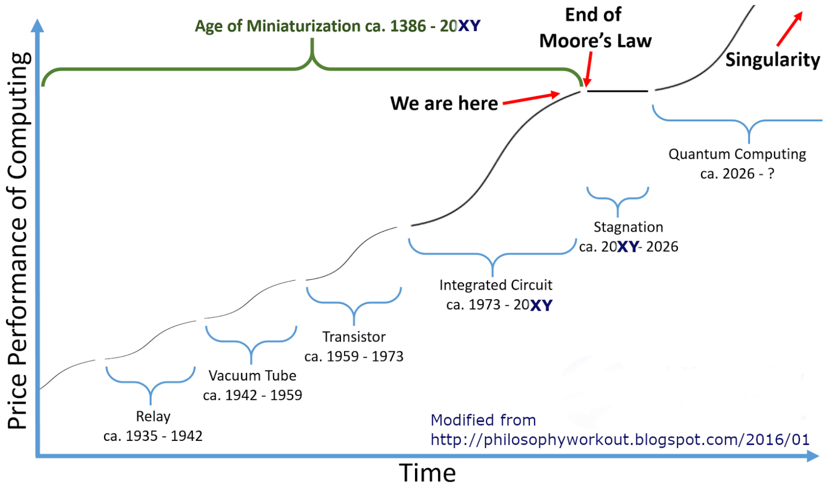
- …to shift resources (including R&D) to the 14 and 12nm efforts where …most of their chip customers …are planning to stay with the current-gen architectures and squeeze performance out by other means.
- 7nm is expensive, it's cheaper and easier to improve the performance and density of 12nm, and hardware accelerators and custom chips …



https://www.nextplatform.com/2019/02/05/the-era-of-general-purpose-computers-is-ending/

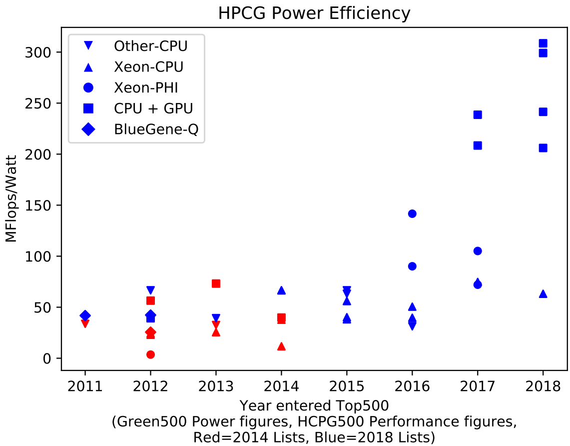
The Evolving Moore's Law





Modified from http://philosophyworkout.blogspot.com/2016/01

Power Consumption and Performance



What now then?

No more advances for free on the back of computer hardware improvements and relatively little pain! Need to "resort" to

Maths

Algorithms

Customised Hardware

Software Solutions for performance, portability, and productivity.

Avoidance and Sharing

No more free lunch, a very different climate modelling world!

Smarter Maths? Techniques!

Parallel Time-Stepping

Not radical (in principle):

$$\mathbf{X}_{t+1}(x, y, z, t) = f(\mathbf{X}_{t-1}, \mathbf{X}_t)$$

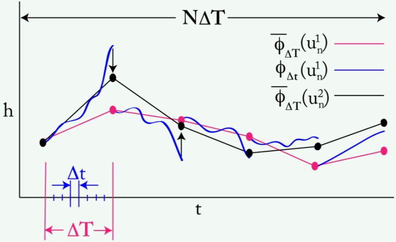
The function f can involve several steps (iterates) or some sort of prediction/correction.

$$\begin{aligned} \text{Predictor:} \quad \mathbf{X}^p_{t+1} &= f_p(\mathbf{X}_{t-1}, \mathbf{X}_t) \\ \text{Corrector:} \quad \mathbf{X}_{t+1} &= f_c(\mathbf{X}^p_{t+1} + \mathbf{X}_t) \end{aligned}$$

There is scope to do some of this in parallel with several methods discussed in the literature.

Parallel in Time

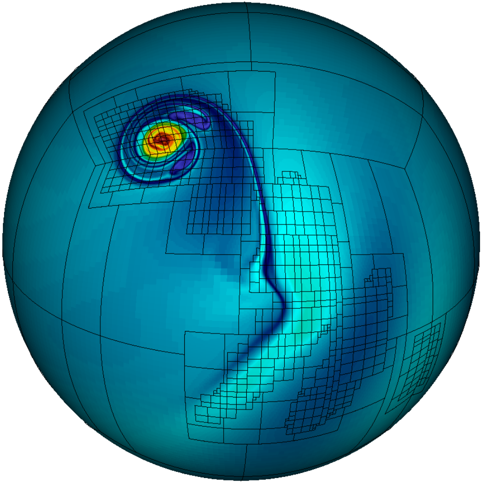
Quite radical:



Predict using a coarse model with long timesteps. Correct in parallel with a finer resolution model.
Some experiments in the literature ...

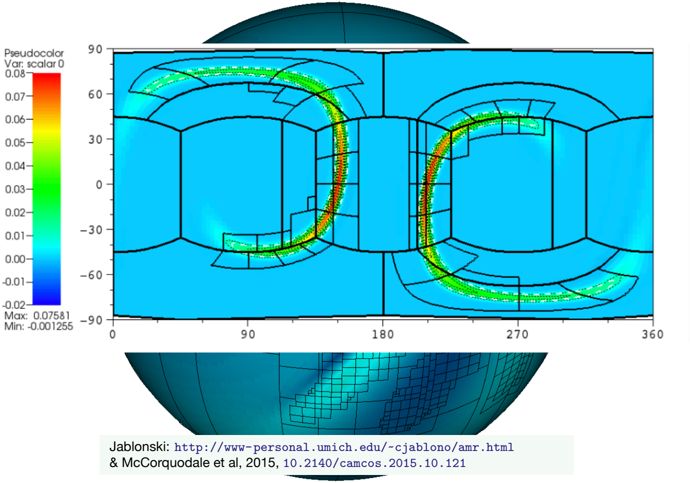
Smarter Maths? - Adaptive Grids

If we can't have ever increasing uniform grids:



Smarter Maths? - Adaptive Grids

If we can't have ever increasing uniform grids:



Jablonski: http://www-personal.umich.edu/~cjablono/amr.html
& McCorquodale et al, 2015, 10.2140/camcos.2015.10.121

Growing impact of Machine Learning and Artificial Intelligence

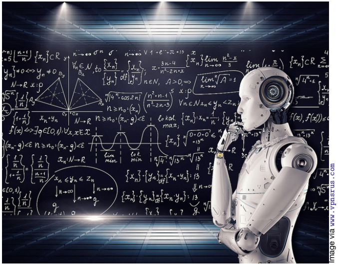


Image via www.vpnsrus.com

Gratuitous "robots are coming" image

Expect ML and AI to have major implications for both

- HPC architectures, and
- Algorithms, in use before, during, and after simulation (analytics)!

Growing impact of Machine Learning and Artificial Intelligence

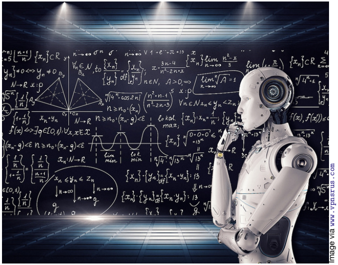


image via www.vpnsrus.com

Gratuitous "robots are coming" image

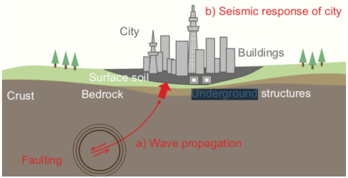
Expect ML and AI to have major implications for both

- HPC architectures, and
- Algorithms, in use before, during, and after simulation (analytics)!

Initial emphasis on climate services, parameter estimation (for parameterisations) and emulation (potentially avoiding avoid long spin-up runs).

Two interesting examples contributed to the Gordon Bell competition this year:

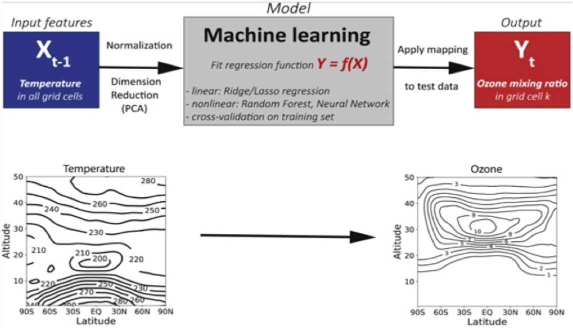
- Preconditioning implicit solvers using artificial intelligence — ground breaking (!) simulations of earthquakes and building response : Ichimura et al 2018.



- Exascale Deep Learning for Climate Analytics - Extracting weather patterns from climate simulations: Kurth et al 2018, co-winner of 2018 Gordon Bell prize.

Using machine learning to bypass calculations

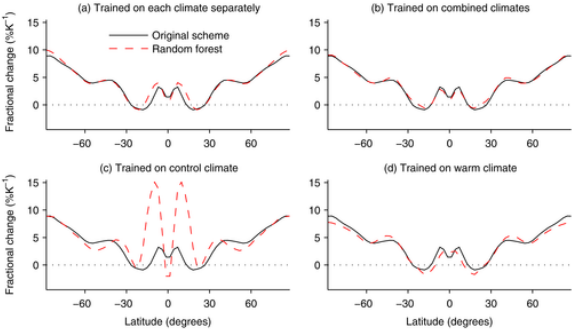
Using machine learning to build temperature-based ozone parameterizations for climate sensitivity simulations. Nowack, Braesicke, Haigh, Abraham, Pyle, and Voulgarakis (2018): doi:10.1088/1748-9326/aae2be



- "…the regression reliably predicts three-dimensional ozone distributions at monthly to daily time intervals."
- "an important stepping stone towards a range of new computationally efficient methods to consider ozone changes in long climate simulations …"

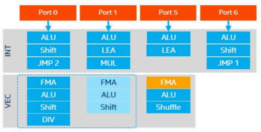
Use of ML in climate needs care (no surprises there!)

Using Machine Learning to Parameterize Moist Convection: Potential for Modeling of Climate, Climate Change, and Extreme Events. O'Gorman and Dwyer (2018): 10.1029/2018MS001351

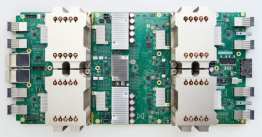


- Random-forest parameterization of convection gives accurate GCM simulations of climate and precipitation extremes in idealized tests
- Climate change captured when trained on control and warm climate, or only on warm climate, but not when trained only on control climate

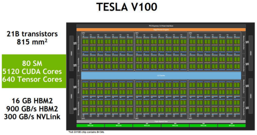
From decades of the same to a Cambrian Explosion



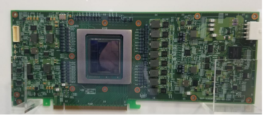
Vector Processors on Intel Zeon



Google's Tensor Programming Unit



GPUs from NVIDIA and AMD



Vector Processing Units from NEC

Server chips based on ARM designs

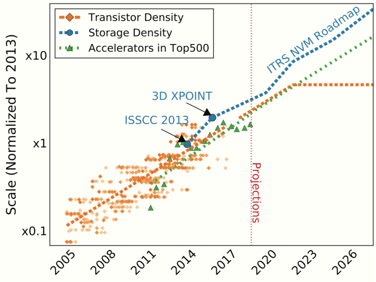
FPGA from many sources

The end of Moore's Law means more specialisation: all with very different programming models!

Even more complicated Hardware/Software Co-Design

Scaling Datacenter Accelerators With Compute-Reuse Architectures

Fuchs and Wentzlaff, 10.1109/ISCA.2018.00038

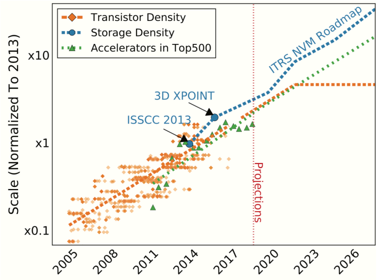


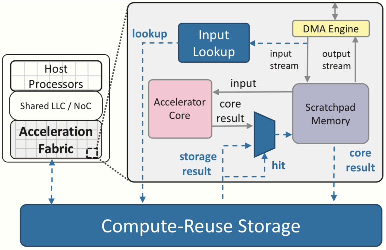
- Observed that while the future for compute and transistors is hitting a physical wall, the same is not yet true of memory (both "normal" and "storage class")

Even more complicated Hardware/Software Co-Design

Scaling Datacenter Accelerators With Compute-Reuse Architectures

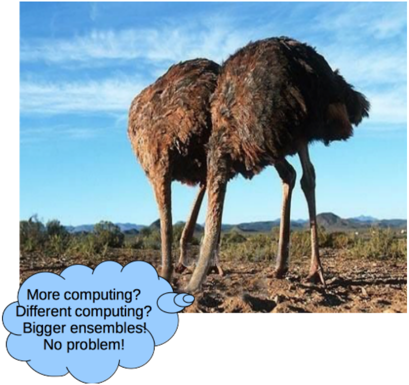
Fuchs and Wentzlaff, 10.1109/ISCA.2018.00038

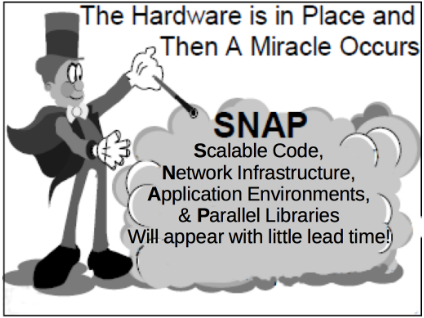




- Observed that while the future for compute and transistors is hitting a physical wall, the same is not yet true of memory (both "normal" and "storage class")
- **Accelerate by re-using (stored) previous computations.**

What about software?





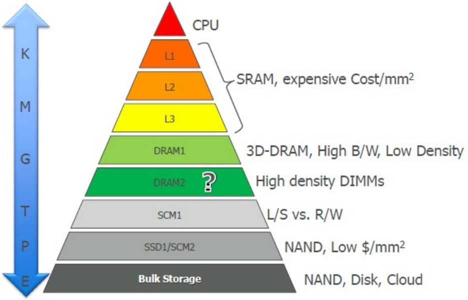
Some people have a very naive idea about the relationship between the hardware and the software!

Too many levels of parallelism

- Vector Units (on chip)
- Parallelism Across Cores
- Shared Memory Concurrency
- Distributed Memory
- Numerical Method Concurrency
- Internal Component Concurrency
- Coupled Component Concurrency
- I/O and Diagnostic Parallelism
- (Storage System Parallelism)

Too many levels of parallelism

- Vector Units (on chip)
- Parallelism Across Cores
- Shared Memory Concurrency
- Distributed Memory
- Numerical Method Concurrency
- Internal Component Concurrency
- Coupled Component Concurrency
- I/O and Diagnostic Parallelism
- (Storage System Parallelism)



Nearly everything is processor/system dependent! (except green layers on left).

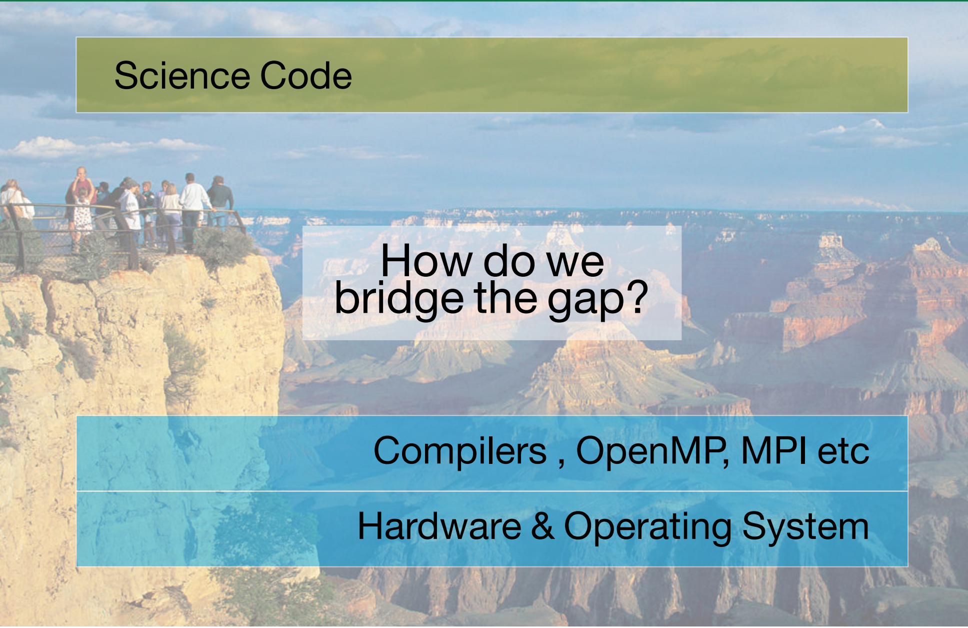
Entirely new programming models are likely to be necessary, with entirely new* constructs such as thread pools and task-based parallelism possible. Memory handling will be crucial!

*New in this context!

Software changing slowly & slowing!

Hardware changing rapidly & accelerating!

How far is it between our scientific aspiration and our ability to develop and/or rapidly adapt our codes to the available hardware?



Science Code

How do we bridge the gap?

Compilers , OpenMP, MPI etc

Hardware & Operating System

Route 1: The Massive Edifice

- No group has enough effort to do all the work needed.
- No group has **all** the relevant expertise.

Route 2: Incremental Advances

- The peril of the local minimum
- Any given span/leap may not be sufficient to cross the next gap!

Route 1: The Massive Edifice

- No group has enough effort to do all the work needed.
- No group has **all** the relevant expertise.

Route 2: Incremental Advances

- The peril of the local minimum
- Any given span/leap may not be sufficient to cross the next gap!

Route 3: Assemble Components

- Share Requirements; Share Development.
- Define Interfaces and Connections.

Science Code

Defined Interfaces and Contracts

High Level Libraries and Tools

Defined Interfaces and Contracts

Libraries and Tools

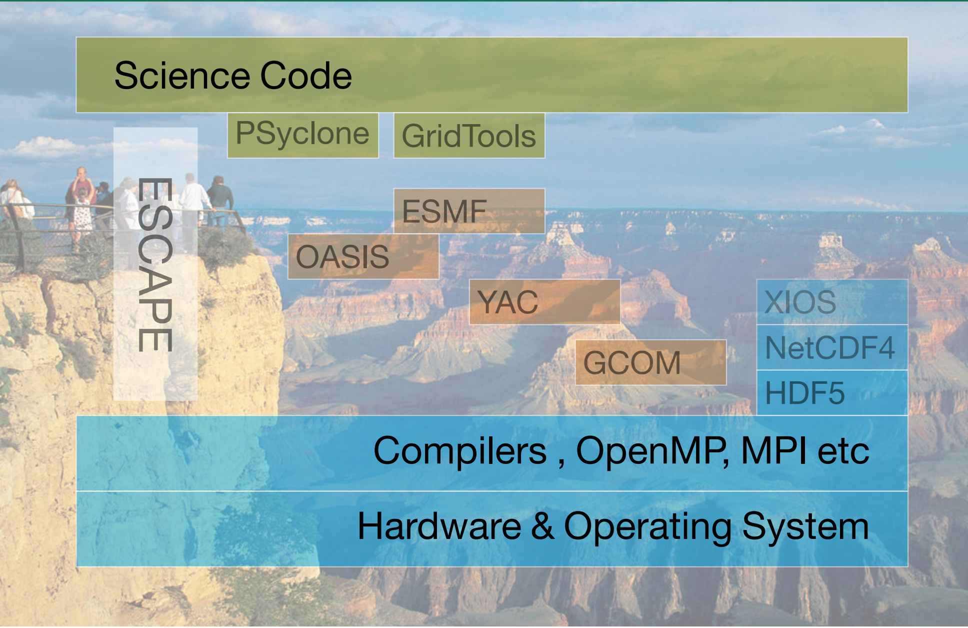
Defined Interfaces and Contracts

Low-Level Libraries and Tools

Defined Interfaces and Contracts

Compilers , OpenMP, MPI etc

Hardware & Operating System


Science Code
PSyclone
GridTools
ESCAPE
ESMF
OASIS
YAC
GCOM
XIOS
NetCDF4
HDF5
Compilers , OpenMP, MPI etc
Hardware & Operating System

Why and What is a Domain Specific Language (DSL)?

Why?

- Humans currently produce the best optimised code!
- Humans can inspect an algorithm, and exploit domain-specific knowledge to reason how to improve performance – but a compiler or generic parallelisation tool doesn't have that knowledge.
- Result: Humans better than generic tools every time, but it's big slow task and mostly not portable!

Why and What is a Domain Specific Language (DSL)?

Why?

- Humans currently produce the best optimised code!
- Humans can inspect an algorithm, and exploit domain-specific knowledge to reason how to improve performance – but a compiler or generic parallelisation tool doesn't have that knowledge.
- Result: Humans better than generic tools every time, but it's big slow task and mostly not portable!

What?

- A domain specific compiler, with a set of rules!
- Exploits a priori knowledge, e.g.
  - Operations are performed over a mesh,
  - The same operations are typically performed independently at each mesh point/volume/element,
  - the meshes themselves typically have consistent properties.
- Leave a much smaller task for the humans!

DSLs in the Wild — two major projects:

- GridTools (formerly Stella)
- PSyclone (from Gung Ho)

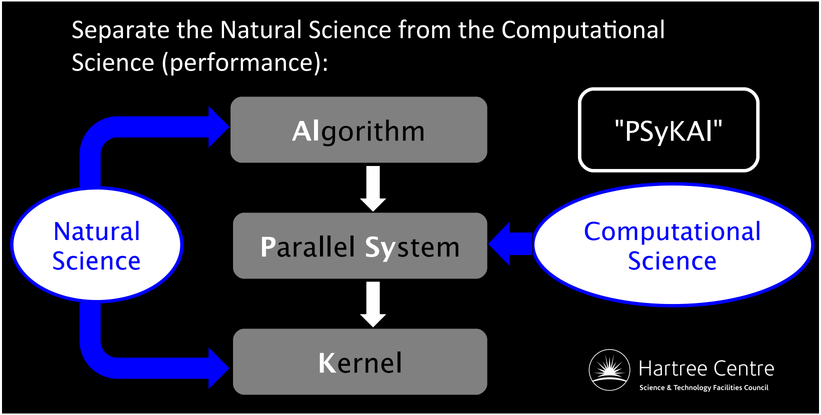
Both are DSELs ... domain specific embedded languages.

- Embedded in C++
- Originally targeted finite difference lat-lon Limited Area Model.
- Backends (via human experts) mapped to the science description via C++ templates.

- Embedded in Fortran
- Originally targeted finite element irregular mesh.
- A recipe of optimisations (via human experts) is used by PSyclone to produce targeted code.

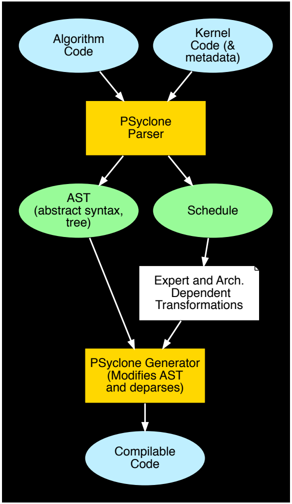
In both cases the DSL approach allows mathematical experts to do their thing, while HPC experts do their thing, and the DSL provides a separation of concerns.

Separation of Concerns



Courtesy of Andrew Porter and Rupert Ford, STFC

PSyclone Example: NEMOLite2D (shallow-water, finite-difference)

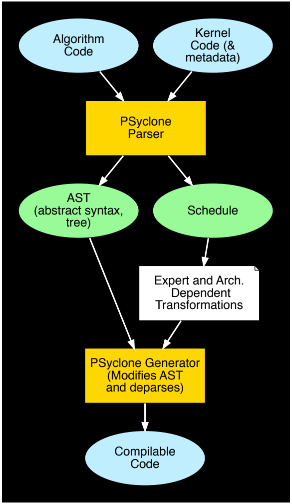


Courtesy of Andrew Porter and Rupert Ford, STFC

Single Timestep Algorithm:

```
call invoke(                                                  &
          continuity(ssha_t, sshn_t, sshn_u, sshn_v,          &
                     hu, hv, un, vn, rdt),                    &

          momentum_u(ua, un, vn, hu, hv, ht,                  &
                     ssha_u, sshn_t, sshn_u, sshn_v),         &

          momentum_v(va, un, vn, hu, hv, ht,                  &
                     ssha_v, sshn_t, sshn_u, sshn_v),         &

          bc_ssh(istp, ssha_t),                               &

          bc_solid_u(ua),                                     &
          bc_solid_v(va),                                     &

          bc_flather_u(ua, hu, sshn_u),                       &
          bc_flather_v(va, hv, sshn_v),                       &

          copy(un, ua),                                       &
          copy(vn, va),                                       &
          copy(sshn_t, ssha_t),                               &

          next_sshu(sshn_u, sshn_t),                          &
          next_sshv(sshn_v, sshn_t)                           &
         )
```

PSyclone Example: NEMOLite2D (shallow-water, finite-difference)



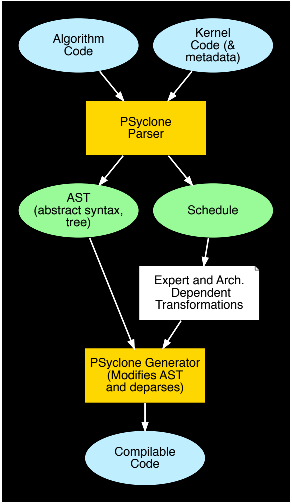
Courtesy of Andrew Porter and Rupert Ford, STFC

One of many Kernels:

```
subroutine continuity_code(ji, jj,                             &
                           ssha, sshn, sshn_u, sshn_v, &
                           hu, hv, un, vn, rdt, e12t)
  implicit none
  integer,                    intent(in)  :: ji, jj
  real(wp),                   intent(in)  :: rdt
  real(wp), dimension(:,:), intent(in)  :: e12t
  real(wp), dimension(:,:), intent(out) :: ssha
  real(wp), dimension(:,:), intent(in)  :: sshn, sshn_u, sshn_v, &
                                             hu, hv, un, vn
  ! Locals
  real(wp) :: rtmp1, rtmp2, rtmp3, rtmp4

  rtmp1 = (sshn_u(ji  ,jj  ) + hu(ji  ,jj  )) * un(ji  ,jj  )
  rtmp2 = (sshn_u(ji-1,jj  ) + hu(ji-1,jj  )) * un(ji-1,jj  )
  rtmp3 = (sshn_v(ji  ,jj  ) + hv(ji  ,jj  )) * vn(ji  ,jj  )
  rtmp4 = (sshn_v(ji  ,jj-1) + hv(ji  ,jj-1)) * vn(ji  ,jj-1)

  ssha(ji,jj) = sshn(ji,jj) + (rtmp2 - rtmp1 + rtmp4 - rtmp3) * &
                rdt / e12t(ji,jj)

end subroutine continuity_code
```

(which conform to a model specific "data model" - i.e. IJK, IKJ, KIJ, order)

PSyclone Example: NEMOLite2D (shallow-water, finite-difference)

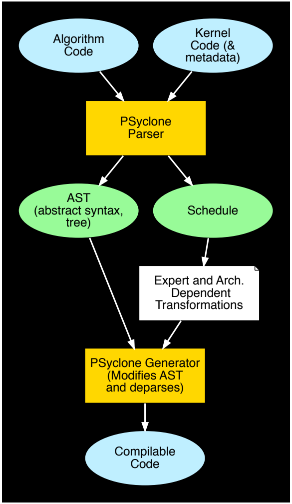


Courtesy of Andrew Porter and Rupert Ford, STFC

After vanilla processing with PSyclone, the algorithm looks a bit like this:

```
do jj = ssha%internal%ystart, ssha%internal%ystop, 1
  do ji = ssha%internal%xstart, ssha%internal%xstop, 1

    call continuity_code(ji, jj,                             &
                         ssha%data, sshn_t%data,             &
                         sshn_u%data, sshn_v%data,           &
                         hu%data, hv%data, un%data, vn%data, &
                         rdt, sshn_t%grid%area_t)
  end do
end do

do jj = ua%internal%ystart, ua%internal%ystop, 1
  do ji = ua%internal%xstart, ua%internal%xstop, 1

    call momentum_u_code(ji, jj, &
                         ua%data, un%data, vn%data, &
                         hu%data, hv%data, ht%data, &
                         ssha_u%data, sshn_t%data,  &
                         sshn_u%data, sshn_v%data,  &
                         un%grid%tmask,  &
                         un%grid%dx_u,   &
                         un%grid%dx_v,   &
                         un%grid%dx_t,   &
                         un%grid%dy_u,   &
                         un%grid%dy_t,   &
                         un%grid%area_u, &
                         un%grid%gphiu)
  end do
end do
```

PSyclone Example: NEMOLite2D (shallow-water, finite-difference)

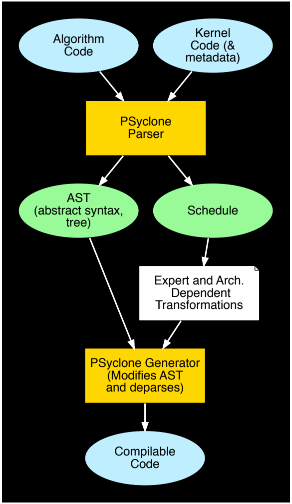


Courtesy of Andrew Porter and Rupert Ford, STFC

An example of the "science agnostic" schedule:

```
GOSchedule[invoke='invoke_0',
           Constant loop bounds=True]
  Loop[type='outer',field_space='ct',
       it_space='internal_pts']
    Loop[type='inner',field_space='ct',
         it_space='internal_pts']
      KernCall continuity(ssha_t,sshn_t,
               sshn_u,sshn_v,hu,hv,un,vn,
               rdt,area_t) [mod_inline=False]
  Loop[type='outer',field_space='cu',
       it_space='internal_pts']
    Loop[type='inner',field_space='cu',
         it_space='internal_pts']
      KernCall momentum_u(ua,un,vn,hu,hv,ht,ssha_u,
                     sshn_t,sshn_u,sshn_v,tmask,
                     dx_u,dx_v,dx_t,dy_u,dy_t,
                     area_u,gphiu) [mod_inline=False]
   ...
```

PSyclone Example: NEMOLite2D (shallow-water, finite-difference)



Courtesy of Andrew Porter and Rupert Ford, STFC

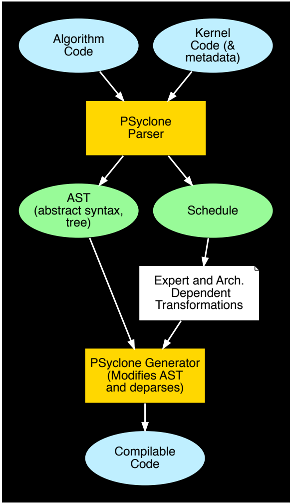
Transformations

Architecture specific transformations can be applied to nodes in the schedule.

As of 2017, PSyclone supported:

- Loop fusion
- Module in-lining of kernels
- OpenMP
- parallel do
- Loop colouring

PSyclone Example: NEMOLite2D (shallow-water, finite-difference)



Courtesy of Andrew Porter and Rupert Ford, STFC

An example of how the schedule changes after transformations (before the Generator step):

```
GOSchedule[invoke='invoke_0',Constant loop bounds=True]
  Directive[OMP parallel]
    Directive[OMP do]
      Loop[type='outer',field_space='ct',
           it_space='internal_pts']
        Loop[type='inner',field_space='ct',
             it_space='internal_pts']
          KernCall continuity_code(ssha_t,sshn_t,sshn_u,
                          ...,area_t)[mod_inline=False]
    Directive[OMP do]
      Loop[type='outer',field_space='cu',
           it_space='internal_pts']
        Loop[type='inner',field_space='cu',
             it_space='internal_pts']
          KernCall momentum_u_code(ua,un,vn,hu,hv,
                       ...,area_u,gphiu)[mod_inline=False]
  ...
```

Whither the DSL?

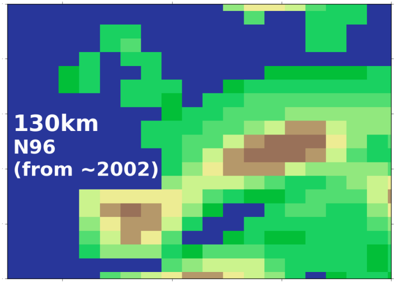
- DSLs are becoming more common across disciplines.
- The Domains are more or less specific …
  - the more specific, the cleaner a domain specific separation of concerns, but the larger the technical debt (maintaining the code and the teams of experts for the backends
  - the more generic, the less the DSL can do for you, and the less the separation of concerns.

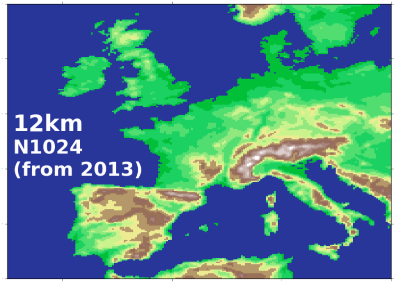
Whither the DSL?

- DSLs are becoming more common across disciplines.
- The Domains are more or less specific …
  - the more specific, the cleaner a domain specific separation of concerns, but the larger the technical debt (maintaining the code and the teams of experts for the backends
  - the more generic, the less the DSL can do for you, and the less the separation of concerns.

- The holy grail is to add further separation of concerns inside the DSL …e.g. can we imagine a GridTools *and a* PSyclone front end to a vendor managed intermediate DSL compiler?
  - compare with MPI: successful because vendors manage their own specific backends with a defined API that we all work with to develop our own libraries (e.g. GCOM, YAXT etc)!

A modest (?) step …





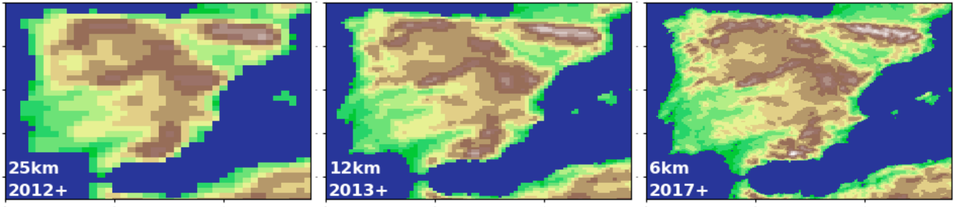
One "field-year" — 26 GB

1 field, 1 year, 6 hourly, 80 levels
1 x 1440 x 80 x 148 x 192

One "field-year" — >6 TB

1 field, 1 year, 6 hourly, 180 levels
1 x 1440 x 180 x 1536 x 2048

Volume — the reality of global 1km grids



What about 1km? That's the current European Network for Earth System Modelling (ENES) goal!

Consider N13256 (1.01km, 26512x19884)):

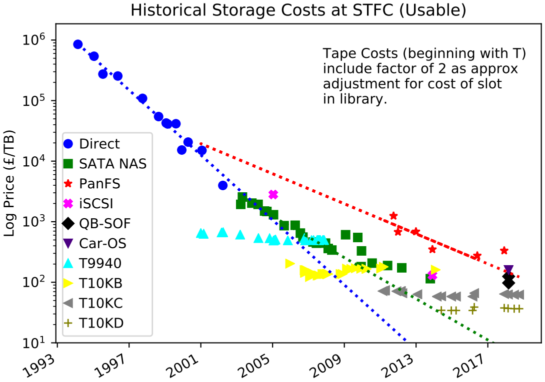
- 1 field, 1 year, 6 hourly, 180 levels
- 1 x 1440 x 180 x 26512 x 19884 = 1.09 PB
- 760 seconds to read one 760 GB (xy) grid at 1 GB/s
- but it's worse that that: 10 variables hourly, $>$ 220 TB/day!

Can no longer consider serial diagnostics, and even parallelised is a challenge for the I/O system!

Real experience with Kryder's Law!

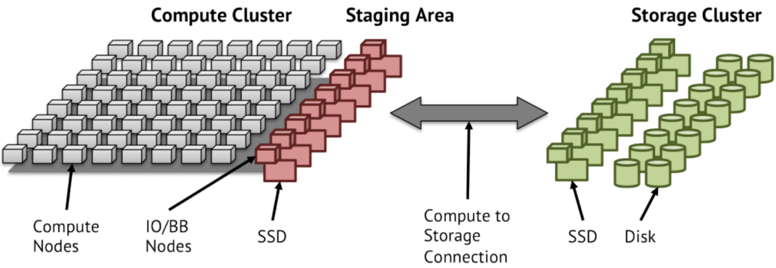
Kryder's Law

- The assumption that disk drive density, also known as areal density, will double every thirteen months. (Hasn't for some time!)
- The implication of Kryder's Law is that as areal density improves, storage will become cheaper:



- Relative cost of **disk** storage going up: each new generation of disk has a "shallower Kryder rate".
- Each new generation of **tape** is cheaper, and price stable over the lifetime.
- Tape has better technical future prospects than disk!

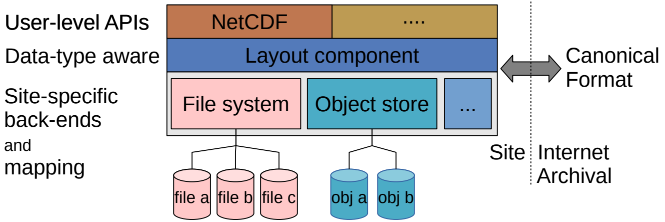
Parallelism in Storage - Getting to and From



Existing filesystems are limiting

- Storage Architecture is complex.
- Difficult to initialise models (takes too long to read and distribute initial data)
- Difficult to get sufficient performance from hundreds of nodes writing to a file system!

Earth System Data Middleware



Key Concepts

- Applications work through existing application interfaces (currently: NetCDF library)
- Middleware utilizes layout component to make placement decisions
- Data is then written/read efficiently avoiding file system limitations (e.g. consistency constraints)
- Potential for deploying with an active storage management system.

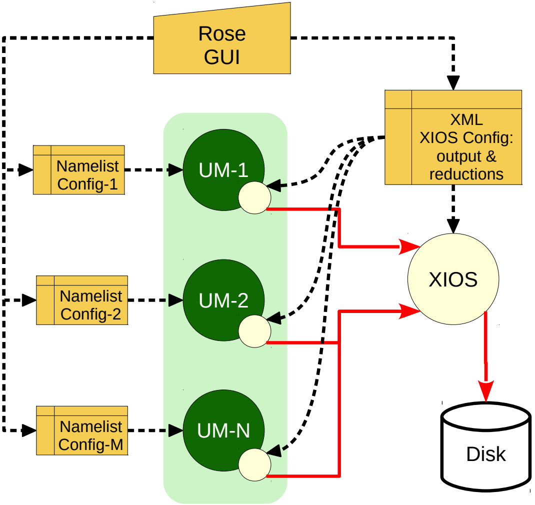
In-Flight Parallel Data Analysis

An ensemble is a set of simulations running different instances of the same numerical experiment. We do this to get information about uncertainty.

Dealing with too much ensemble data

Instead of writing out all ensemble members and doing all the analysis later:

- Calculate ensemble statistics on the fly.
- Only write out some ensemble members.
- (Which ones? A tale for another day, see Daniel Galea's Ph.D work.)



In-Flight Parallel Data Analysis

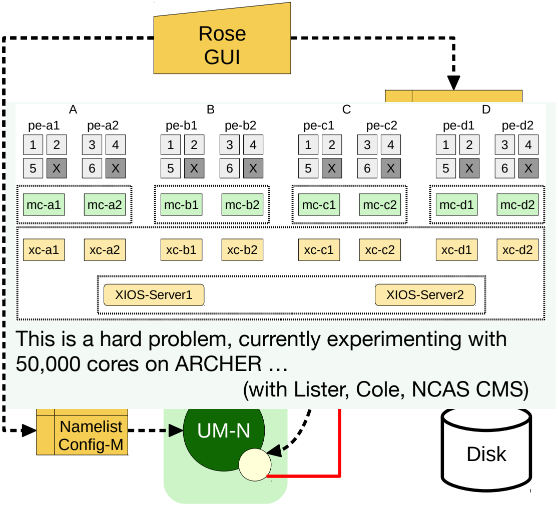
esiwace CENTRE OF EXCELLENCE IN SIMULATION OF WEATHER AND CLIMATE IN EUROPE

An ensemble is a set of simulations running different instances of the same numerical experiment. We do this to get information about uncertainty.

Dealing with too much ensemble data

Instead of writing out all ensemble members and doing all the analysis later:

- Calculate ensemble statistics on the fly.
- Only write out some ensemble members.
- (Which ones? A tale for another day, see Daniel Galea's Ph.D work.)



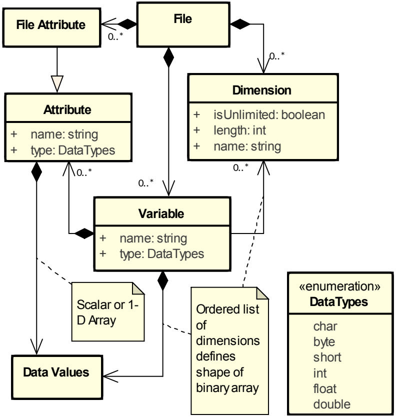
This is a hard problem, currently experimenting with 50,000 cores on ARCHER …

(with Lister, Cole, NCAS CMS)

Climate Forecast Conventions and Data Model

Formats and Semantics

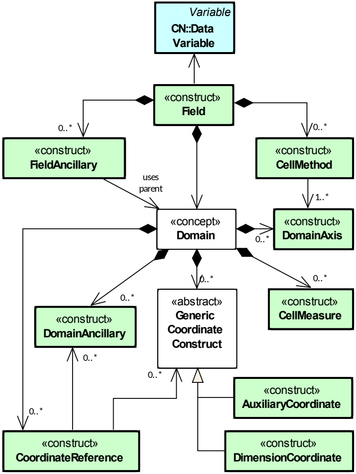
- A file format describes how bits and bytes are organised in some sequence on disk.
- Storage Middleware (e.g. NetCDF) has an implicit or explicit data model for what things are stored in that file.
- The Climate-Forecast conventions describe how coordinates and variable properties are stored in NetCDF.
- We have developed an explicit data model so that these can be used for any storage format.



Climate Forecast Conventions and Data Model

Formats and Semantics

- A file format describes how bits and bytes are organised in some sequence on disk.
- Storage Middleware (e.g. NetCDF) has an implicit or explicit data model for what things are stored in that file.
- The Climate-Forecast conventions describe how coordinates and variable properties are stored in NetCDF.
- We have developed an explicit data model so that these can be used for any storage format.

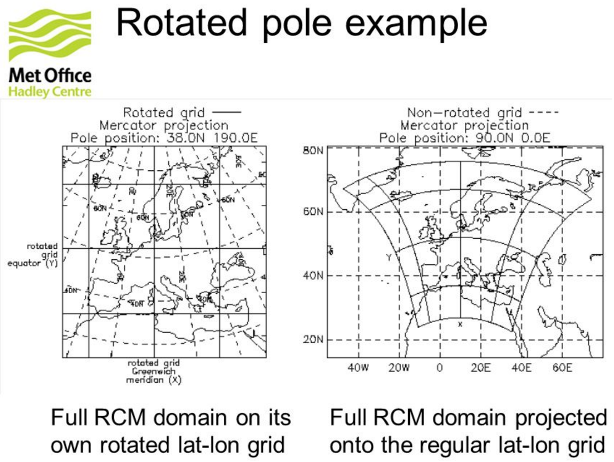


Hassell, D., Gregory, J., Blower, J., Lawrence, B. N., and Taylor, K. E.: A data model of the Climate and Forecast metadata conventions (CF-1.6) with a software implementation (cf-python v2.1), Geosci. Model Dev., 10, 4619-4646, https://doi.org/10.5194/gmd-10-4619-2017, 2017.

CF Conventions in Action

```
print(t)
Field: air_temperature (ncvar\%ta)
---------------------------------
Data              : air_temperature(atmosphere_hybrid_height_coordinate(1),
                    grid_latitude(10), grid_longitude(9)) K
Cell methods      : grid_latitude(10): grid_longitude(9):
                    mean where land (interval: 0.1 degrees) time(1): maximum
Field ancils      : air_temperature standard_error(grid_latitude(10),
                    grid_longitude(9)) = [[0.81, ..., 0.78]] K
Dimension coords: time(1) = [2019-01-01 00:00:00]
                : atmosphere_hybrid_height_coordinate(1) = [1.5]
                : grid_latitude(10) = [2.2, ..., -1.76] degrees
                : grid_longitude(9) = [-4.7, ..., -1.18] degrees
Auxiliary coords: latitude(grid_latitude(10),
                  grid_longitude(9)) = [[53.941, ..., 50.225]] degrees_N
                : longitude(grid_longitude(9),
                  grid_latitude(10)) = [[2.004, ..., 8.156]] degrees_E
                : long_name=
                    Grid latitude name(grid_latitude(10)) = [--, ..., kappa]
Cell measures   : measure:area(grid_longitude(9),
                  grid_latitude(10)) = [[2391.9657, ..., 2392.6009]] km2
...
```

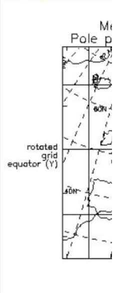
CF Conventions in Action

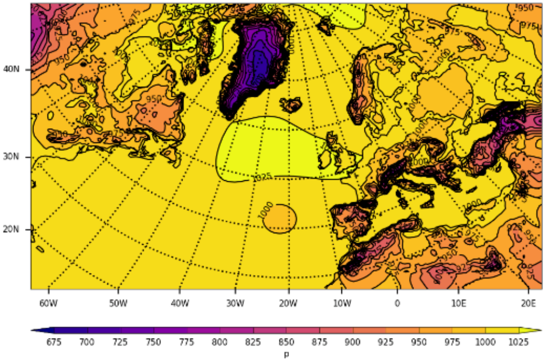



```
oordinate(1),
ne(1): maximum
de(10),
1.5]
ees
rees
]] degrees_N
 degrees_E
[--, ..., kappa]
grid_latitude(10)) = [[2391.9657, ..., 2392.6009]] km2
...
```

CF Conventions in Action

Rotated pole example





ordinate(1),

e(1): maximum
e(10),

.5]
es
ees

| degrees_N

degrees_E

Full R
own r

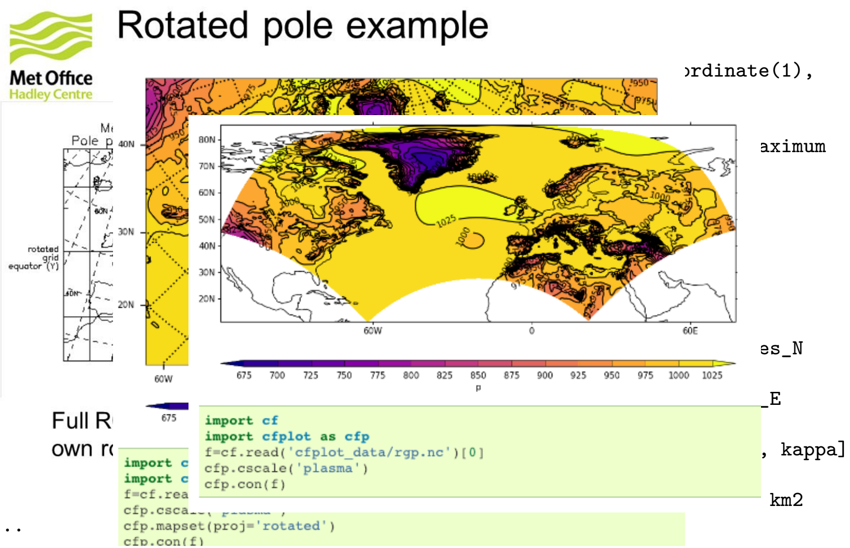
--, ..., kappa]

.6009]] km2

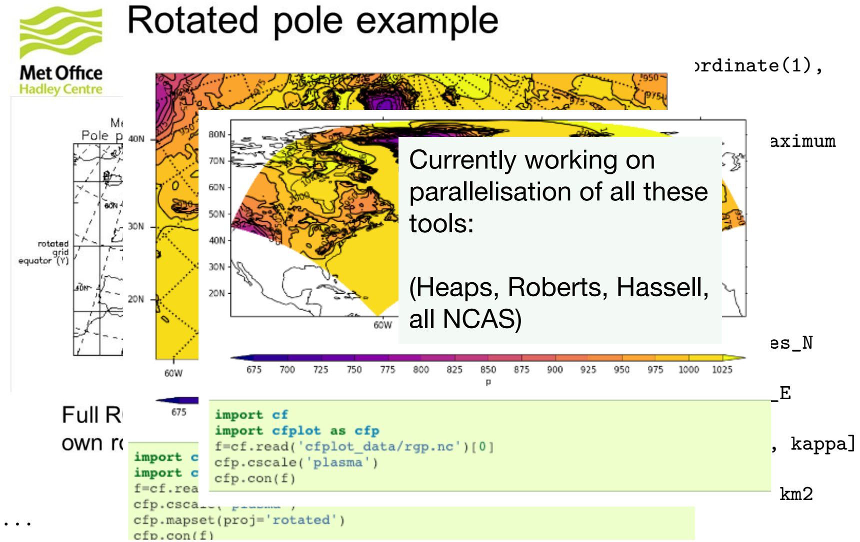
...

```
import cf
import cfplot as cfp
f=cf.read('cfplot_data/rgp.nc')[0]
cfp.cscale('plasma')
cfp.mapset(proj='rotated')
cfp.con(f)
```

CF Conventions in Action



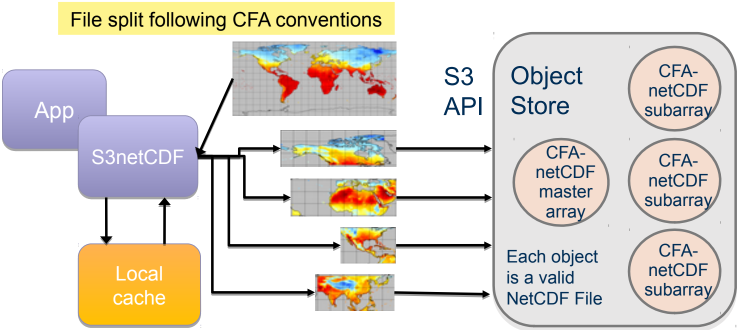
CF Conventions in Action



Currently working on parallelisation of all these tools:

(Heaps, Roberts, Hassell, all NCAS)

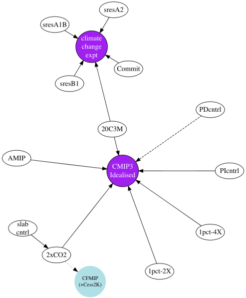
Semantic Storage Layer



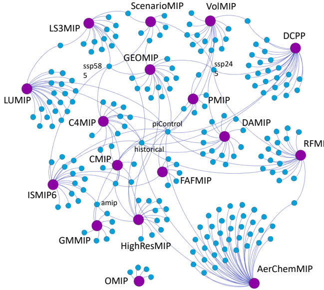
Architecture (with Massey & Jones, STFC)

- Master Array File is a NetCDF file containing dimensions and metadata for the variables including URLs to fragment file locations
- Master Array file optionally in persistent memory or online, nearline, etc. NetCDF tools can query file CF metadata content without fetching them

The CMIP Evolution: from CMIP3 to CMIP6



to



The Logistics of Collaboration

- In HPC we know that the larger the number of cores, the more the communications cost …
- these communications costs need to paid for large scale scientific collaboration too!

From experimental design, to the data request, the (ESGF) dissemination infrastructure, and to the analysis systems; we need to invest more in the supporting infrastructure, and respect the constraints — but this is not a popular message!

Climate Scientists don't *respect* big numbers!

CMIP6 : Timescales, Volumes, Costs

- Designed without a budget.
- Designed without respect for the (energy/financial) cost — Does anyone know of any other scientific endeavour of this scale which has no clear cost/benefit analysis/justification?
- Designed without respect for infrastructure requirements.
  - Advent of the WIP reflects acceptance that there *are* infrastructural *issues*.
- As of today: we still don't know the timing and/or volumes of data delivery into the ESGF.
  - This is obviously problematic for data movement, data management, and the overall performance of the system.
  - "Download at home" did not work for CMIP5, yet there appears to be no real understanding by the designer/user community as to the consequences of the factor of ten in volumes expected for CMIP6.

Climate Scientists don't *respect* big numbers!

CMIP6 : Timescales, Volumes, Costs

- Designed without a budget.
- Designed without respect for the (energy/financial) cost — Does an one know of any other sc entific endeavour of this scale which ... clear cost/bene ... /justification?
- ... out r ... ure re ... the W ... hat th ...
- ... e still ... g and ... nto th ...
- ... usly ... veme ... nagement, and ... ance of th ...
  - "Download at home" did not work for CMIP5, yet there appears to be no real understanding by the designer/user community as to the consequences of the factor of ten in volumes expected for CMIP6.

Hardware Issues

- Centres may (if at all) have resourced CMIP6 in terms of simulation, not analysis.
- Most MIPS wouldn't have considered a need to have someone resource common storage and analysis.

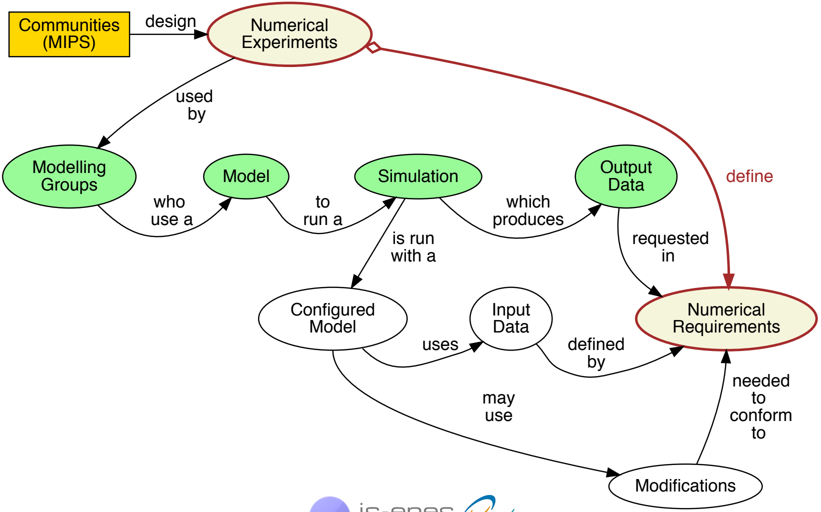
Unrealistic Budget

- No one has provided any requirements or budget for common storage and analysis systems.
- No discussion of the energy costs of simulation and cost/benefit analysis.

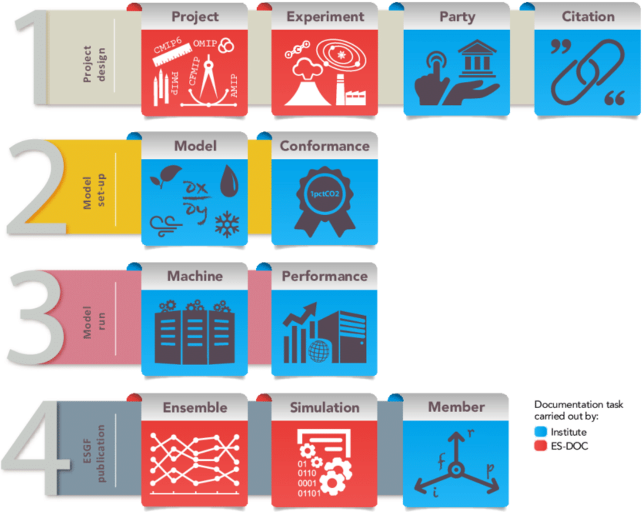
More proactivity needed!

- We will have to be much more proactive in demanding cost/benefit analysis for future activities!
- This will not be popular.

The modelling process

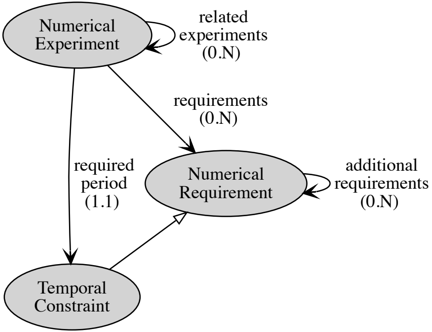


Documentation types within ES-DOC

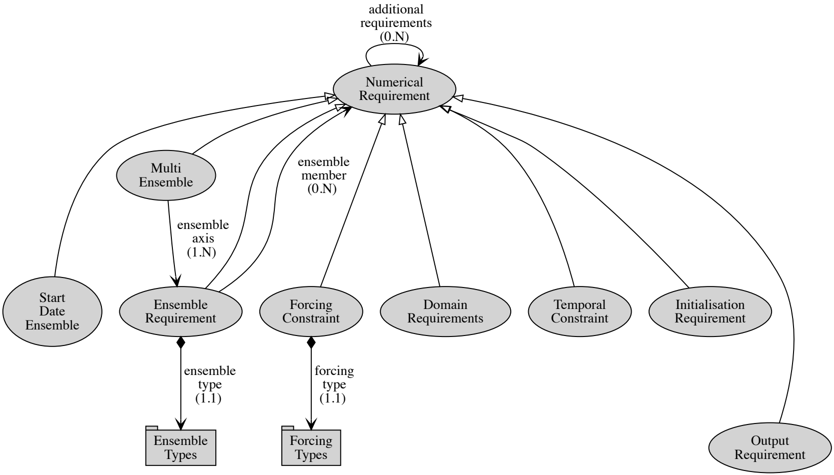


Guillaume Levavasseur, IPSL.

Numerical Experiments



Requirements and Constraints

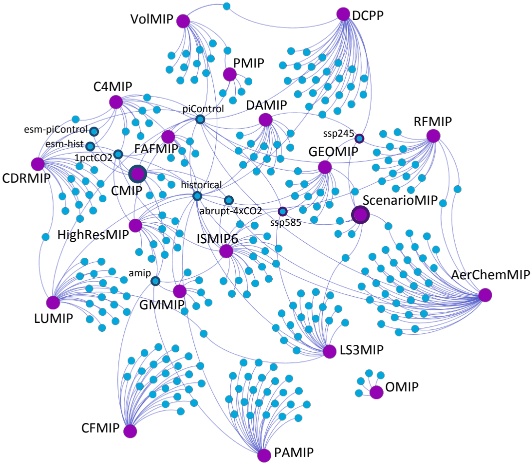


Example: generated view of LUMIP-noFIRE experiment

land-noFire (LUMIP)	
historical land-only with no human fire land management	
Description: Land surface model simulation. Same as land-hist except with fire management maintained at 1850 levels. Start year either 1850 or 1700 depending on standard practice for particular model.	
Rationale: To assess the relative impact of land cover and incremental land management change on fluxes of water, energy, and carbon in combination with other LUMIP land experiments.	
Requirements	
Historical Land Use: Apply the global gridded land-use forcing datasets to link historical land-use data and future projections. This new generation of "land use harmonization" (LUH2) builds upon past work from CMIP5, and includes updated inputs, higher spatial resolution, more detailed land-use transitions, and the addition of important agricultural management layers.	**Historical GSWP3 Meteorological Forcing**: Apply Global Soil Wetness Project phase three (GSWP3) forcing data for offline land surface models running the LS3MIP historical simulation land-hist is provided by the LS3MIP.
1850 Fire Management: Maintain 1850 levels of fire management (anthropogenic ignition and suppression of fire). If ignitions are based on population density, maintain constant population density.	**Historical land surface forcings except fire management**: Apply all transient historical forcings that are relevant for the land surface model except for fire management.
All Land Management Active: All applicable land management active in the land surface model configuration.	**1850-2014 165yrs**: Historical, pre-Industrial to present
1700-2014 315yrs: Historical, from 1700 to 2014.	**LSM Configuration**: Offline land surface model
SingleMember: One ensemble member	

Not all properties shown, view generated using Python interrogating online esdoc repository.

CMIP Experiments as seen by ES-DOC



MIPS (purple dots) and their (shared) experiments (blue dots).

The real core of CMIP is exposed!

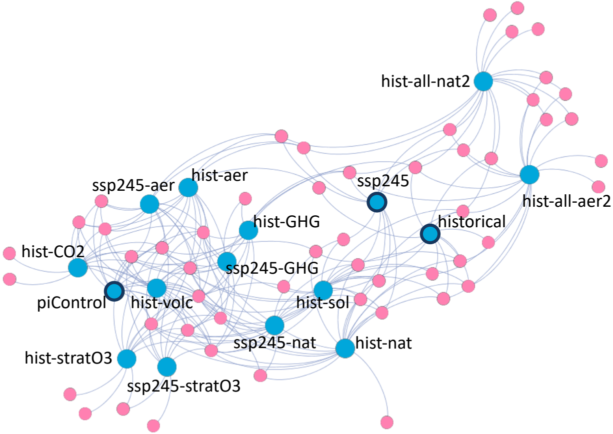
- DECK fundamental experiments (piControl, AMIP) as expected.
- DECK CO2 experiments 1pctCO2 and abrupt-4xCO2, again, as expected
- CMIP6 requirement: historical, obviously
- Perhaps surprisingly: SSP245 and SSP585 from ScenarioMIP

Note also isolation of OMIP and important cross-MIP roles of land-hist, past1000, piCLIm-control and dcppC-forecast-addPinatubo.

Charlotte Pascoe

DAMIP as seen by ES-DOC

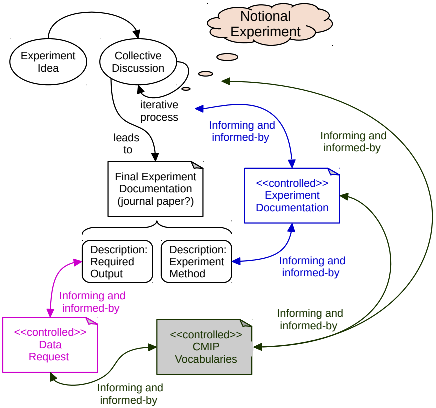
Detection and Attribution MIP



Experiments (blue dots), grouped by common forcing constraints (pink dots)
The three experiments with solid borders are used by DAMIP, but not defined by it.

Charlotte Pascoe

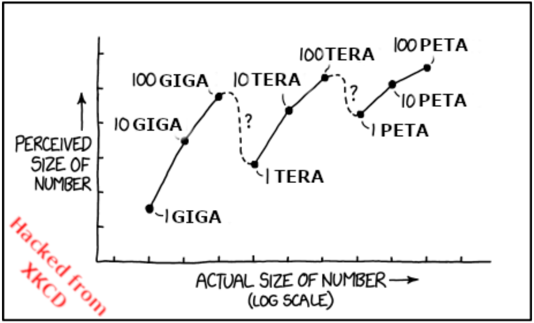
Avoiding Repeated Simulations



Describing and Sharing

- Ideas proceed via discussion to plans, which can be more or less formally documented.
- The more formally they are documented, the more likely that larger groups can buy into a shared vision.
- A key part of shared experiment design is agreeing on model configuration and expected outputs (variables and temporal frequency) …from Stash to a formal data request, and everything in between.

Climate Scientists don't *respect* big numbers!



TALKING ABOUT LARGE NUMBERS IS HARD

In experimental design, many underestimate:

- The energy demands and costs of computing associated with their experiments, and
- The difficulty in managing, disseminating, and utilising large volumes of data!

This is only going to become worse unless we do something about it — but this is not a popular message!

Summary

- Expectation: We need to recalibrate our expectations of future compute.
- Moores's Law: The end will deliver a Cambrian explosion of hardware.
- Post-Moore's Law: *Being smarter*: Crossing the chasm with better maths and community software such as DSLs.
- Kryder's Law: *Being smarter*: Much going on to help us deal with both avoiding writing data, but if we have to have it, handling it efficiently.
- Avoidance: *Being smarter*: ES-DOC project delivering on methodology to share big experiments from research group scale to CMIP scale — but we have to *design and share*!

We need to be investing in being smarter **NOW** …

Scientific Smarts - What will we do then?

This will not yield the end of climate science, but will need to become better at using the right tool(s) for the job, rather than treating everything as a nail because we have a really cool hammer:

- More use of hierarchy of models;
- Precision use of resolution;
- Selective use of complexity;
- Less use of "ensembles of opportunity", much more use of "designed ensembles";
- Duration and Ensemble Size : thinking harder *a priori* about what is needed.
- Much more use of emulation.