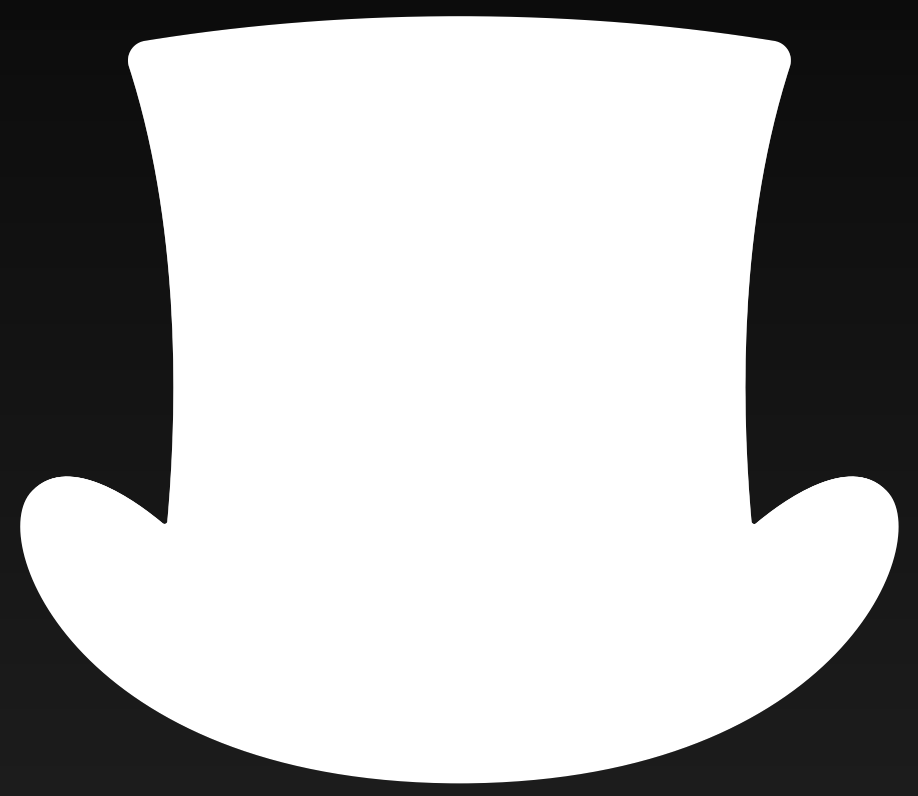


University of Kent

UNIVERSITY OF CAMBRIDGE

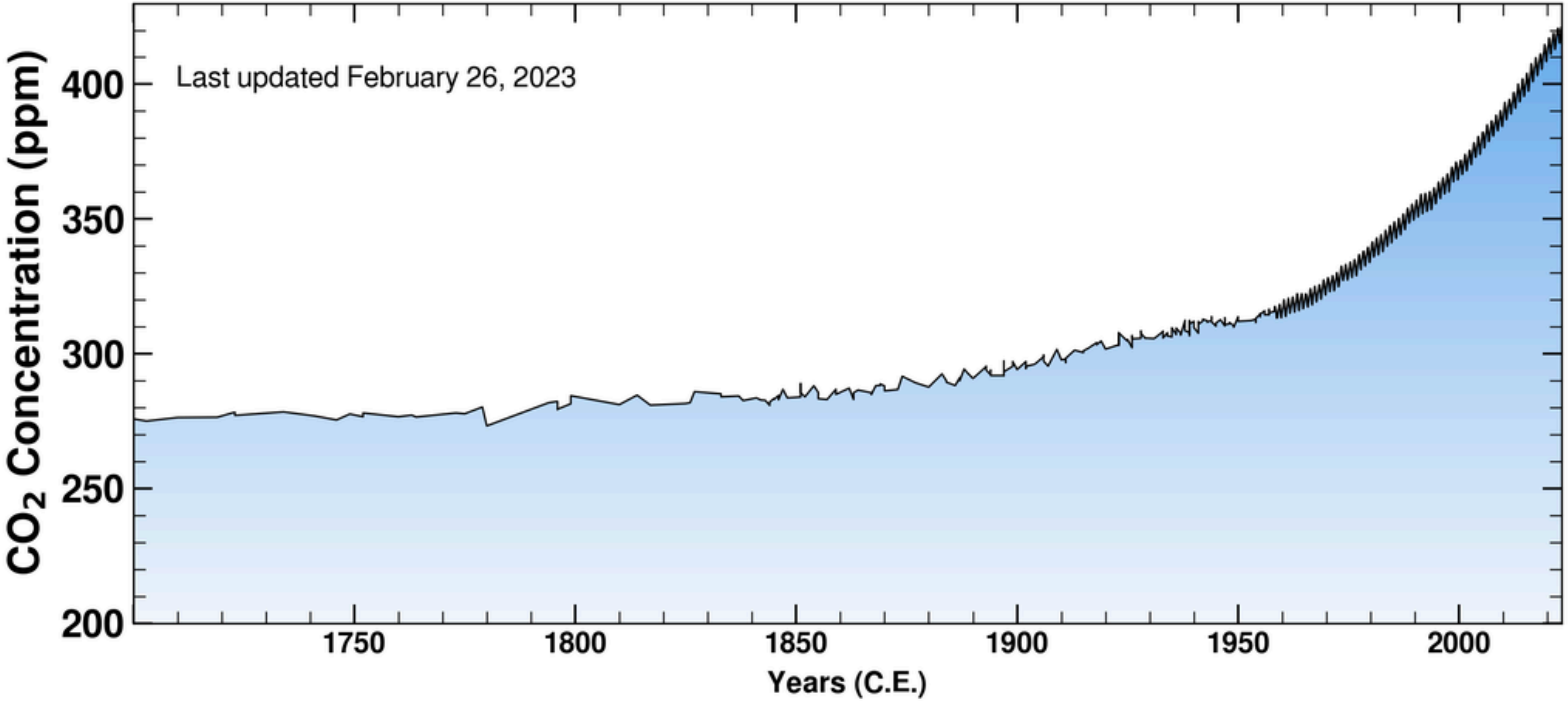
Programming for the Future

Dominic Orchard

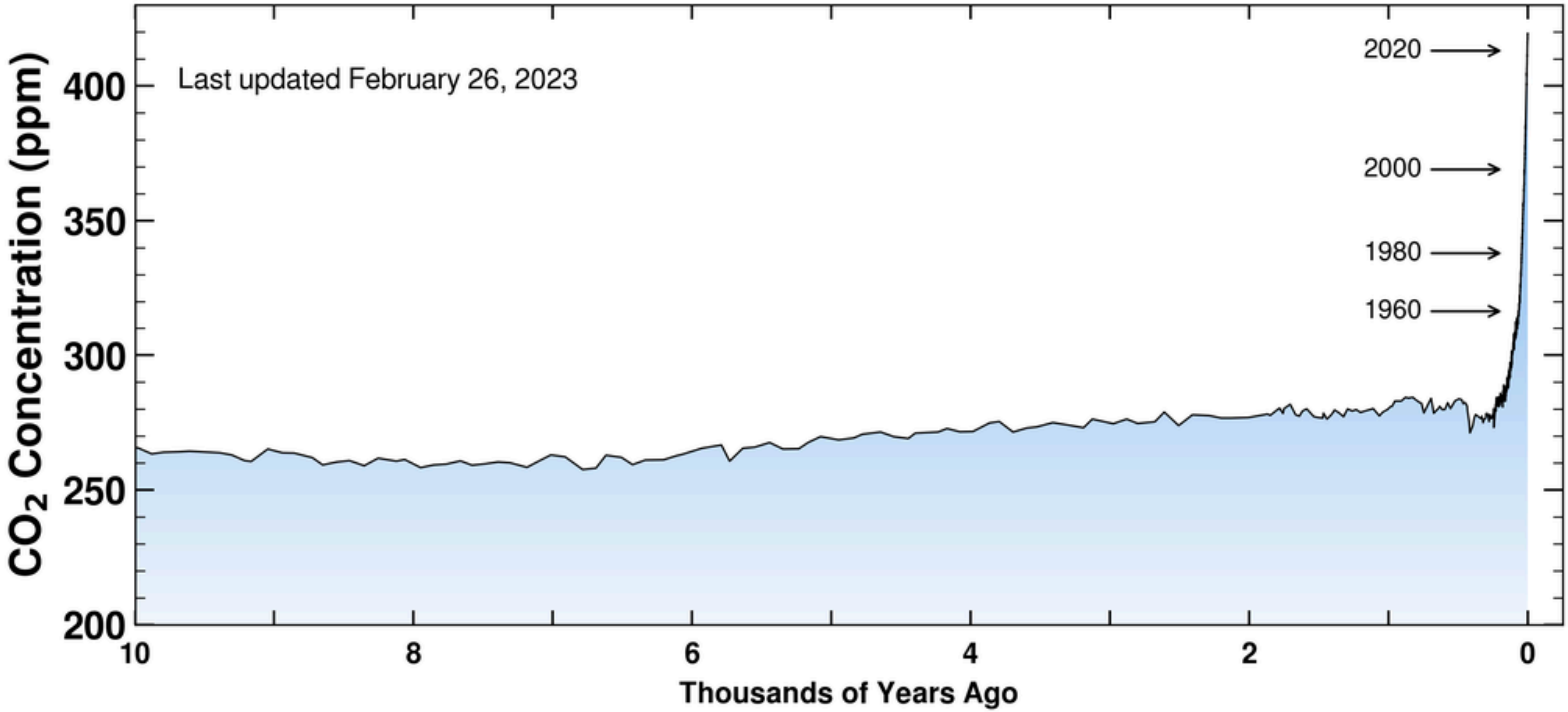
UNIVERSITY OF CAMBRIDGE

Institute of Computing for Climate Science

1850-2022 (Ed Hawkins "Warming stripes")



https://keelingcurve.ucsd.edu/



https://keelingcurve.ucsd.edu/

UN IPCC Projections

IPCC 6 (2022)

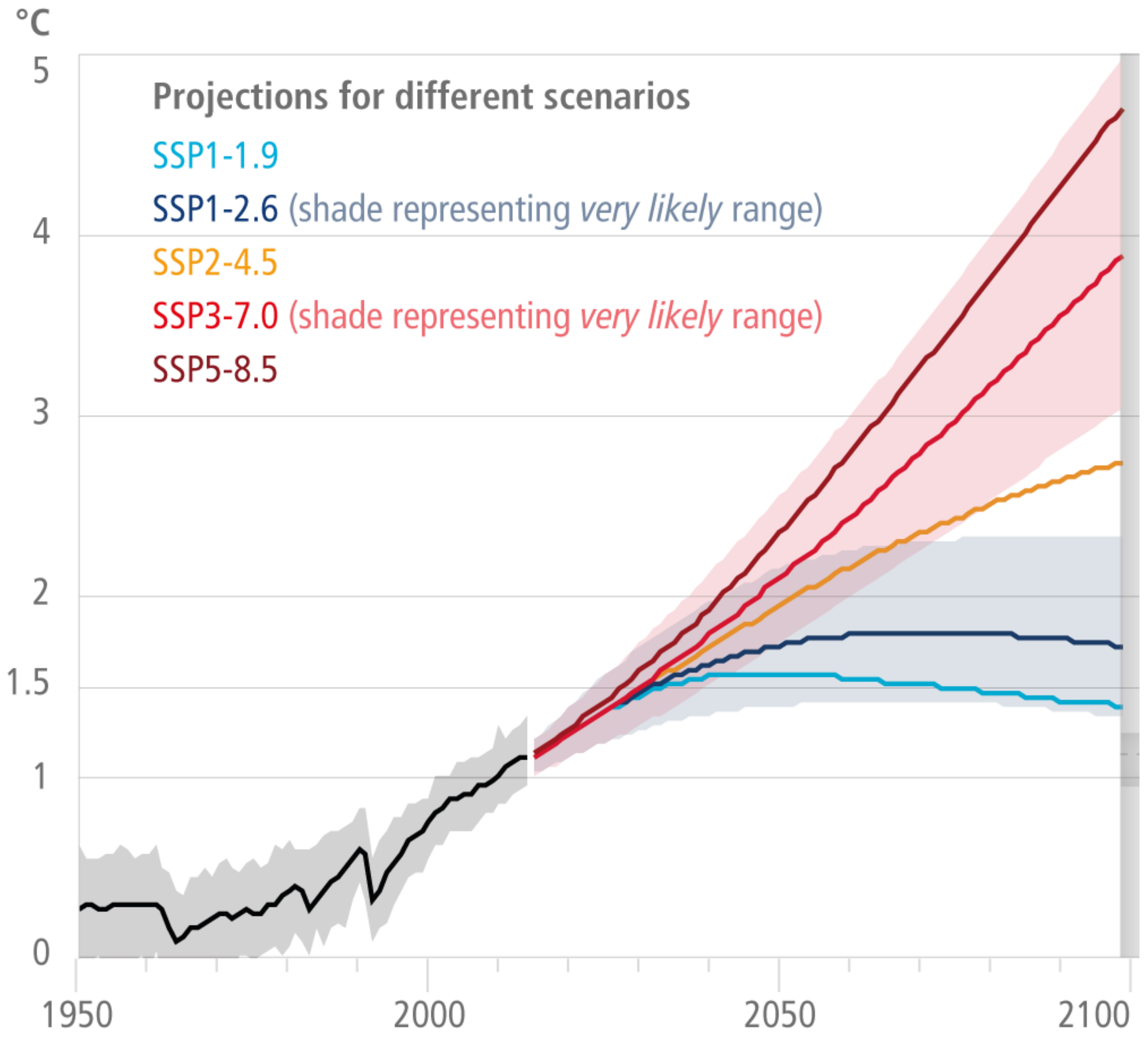
SSP1-1.9 - net zero by 2050

SSP1-2.6 - serious reduction by 2050

SSP2-4.5 - current levels maintained till 2050 then fall to net zero by 2100

SSP3-7.0 - doubling current emissions by 2100

SSP5-8.5 - doubling current emissions by 2050



"A race we are losing, but a race we can win..."

UN Secretary-General António Guterres

UNIVERSITY OF CAMBRIDGE

Institute of Computing for Climate Science

Maximise effectiveness of climate science research via...

Computer Science

Software Engineering

Programming Langauges & systems

Mathematics

Data Science

Machine learning

Emily Shuckburgh

Cambridge Zero
+ CST

Colm Caulfield

Department of Applied Maths and Theoretical Physics

Chris Edsall

University Information Services

Dominic Orchard

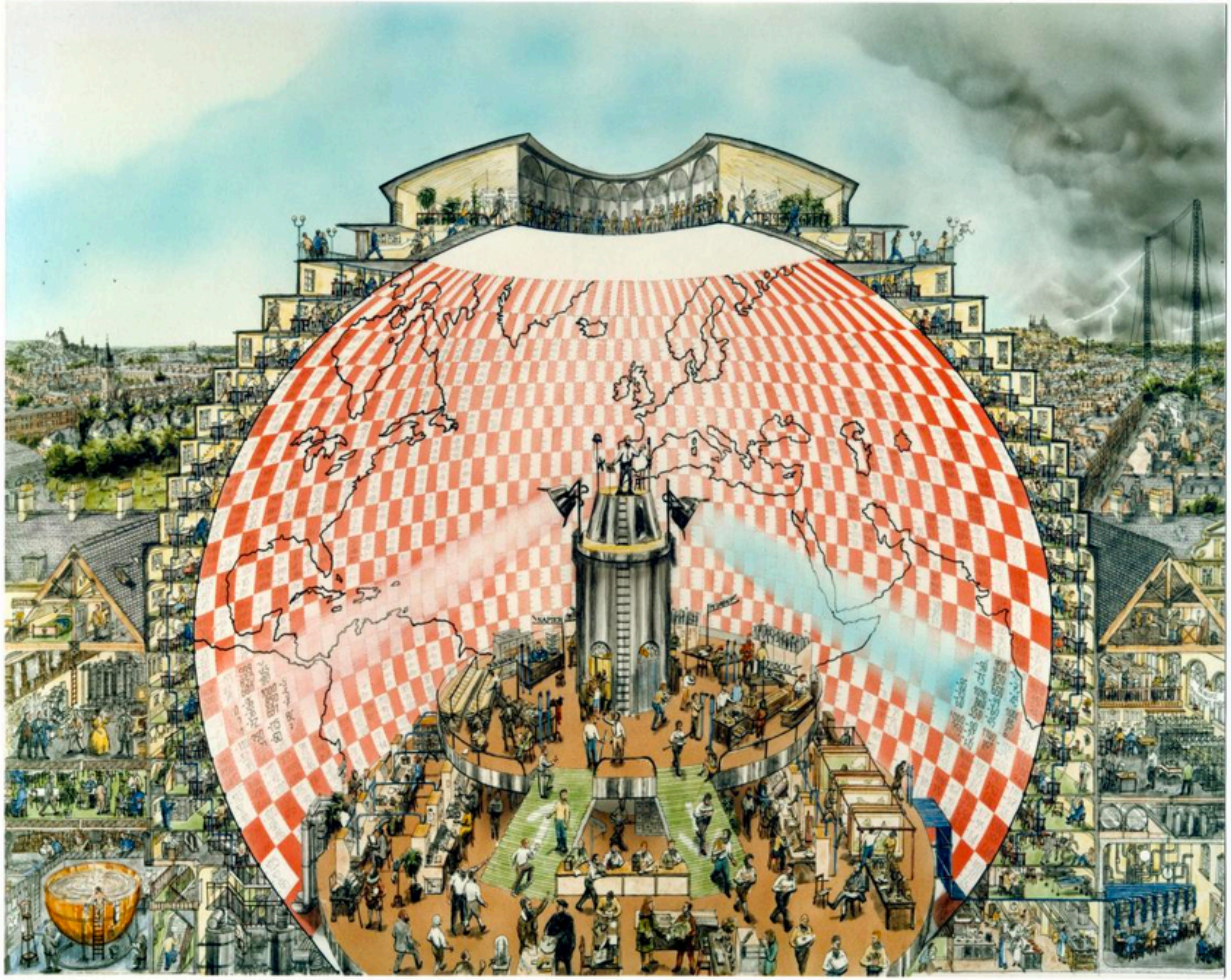
Department of Computer Science & Technology

Marla Fuchs

ICCS

Institute of
Computing for
Climate Science

Climate science and computation



Weather Prediction by Numerical Process

by
L.F.Richardson
1922

Array + stencil!

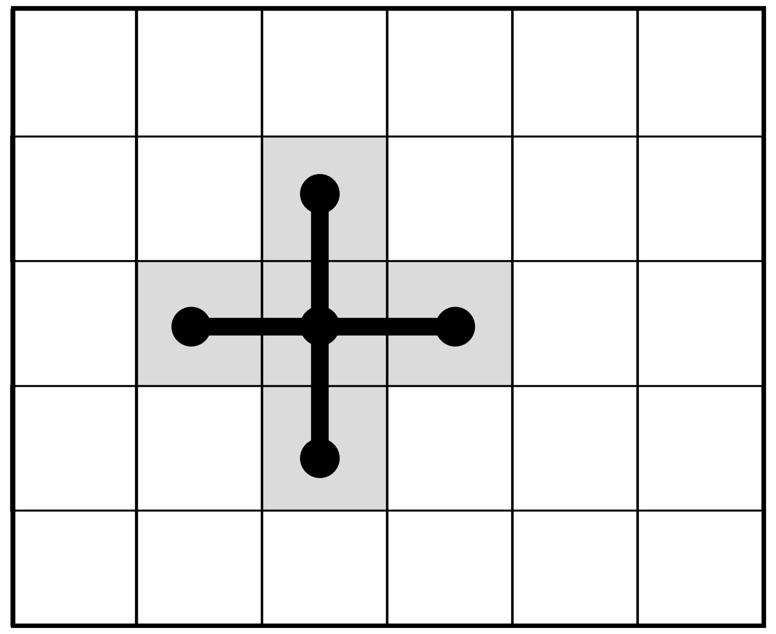
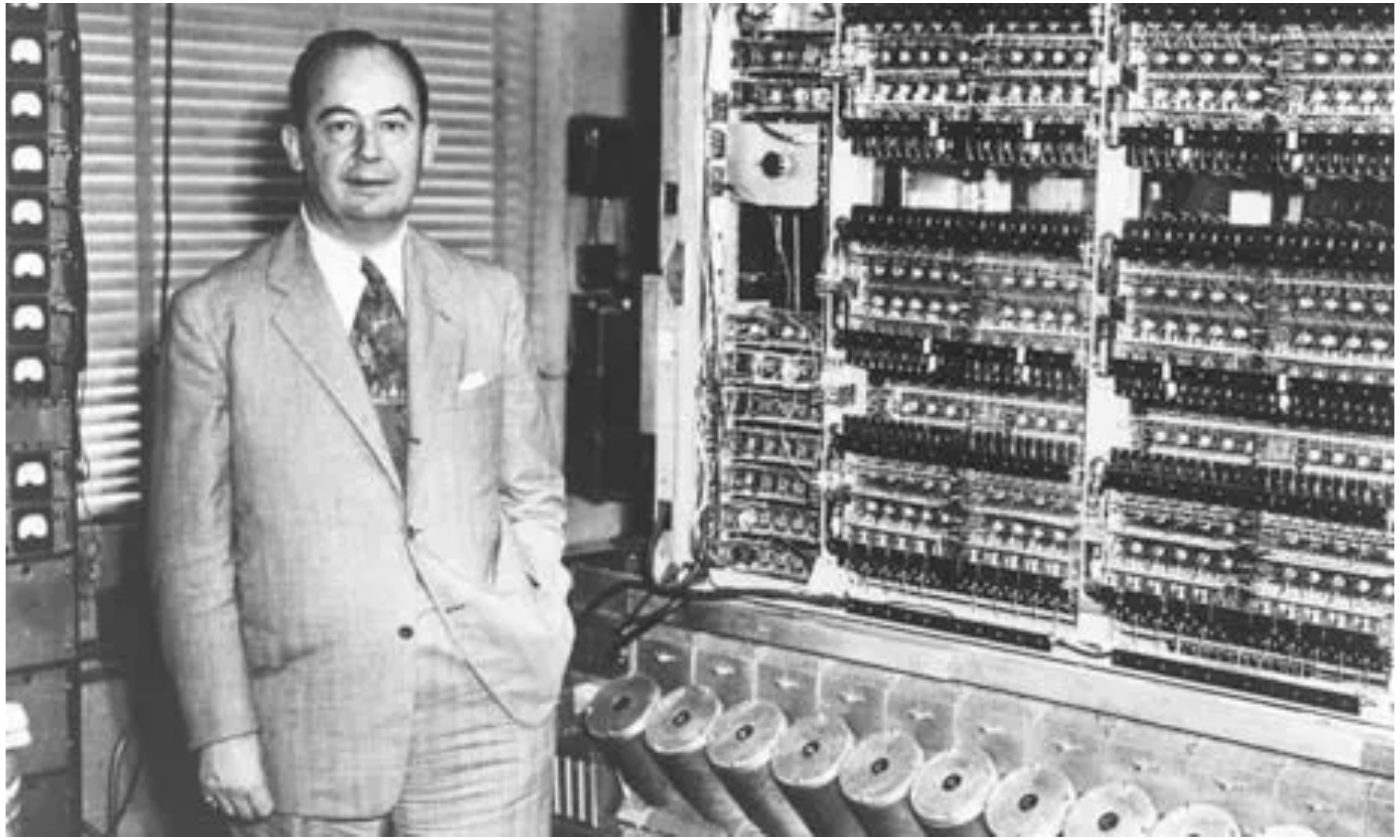


Image: Weather Forecasting Factory by Stephen Conlin, 1986.



John von Neumann
(with the stored-program computer at the Institute of Advanced Study, Princeton 1945)

Jule Gregory Charney

late 1940s first numerical weather forecasts on the ENIAC

Manabe & Wetherald (1967) (1969)

“According to our estimate, a doubling of the CO_2 content in the atmosphere has the effect of raising the temperature of the atmosphere by 2C”

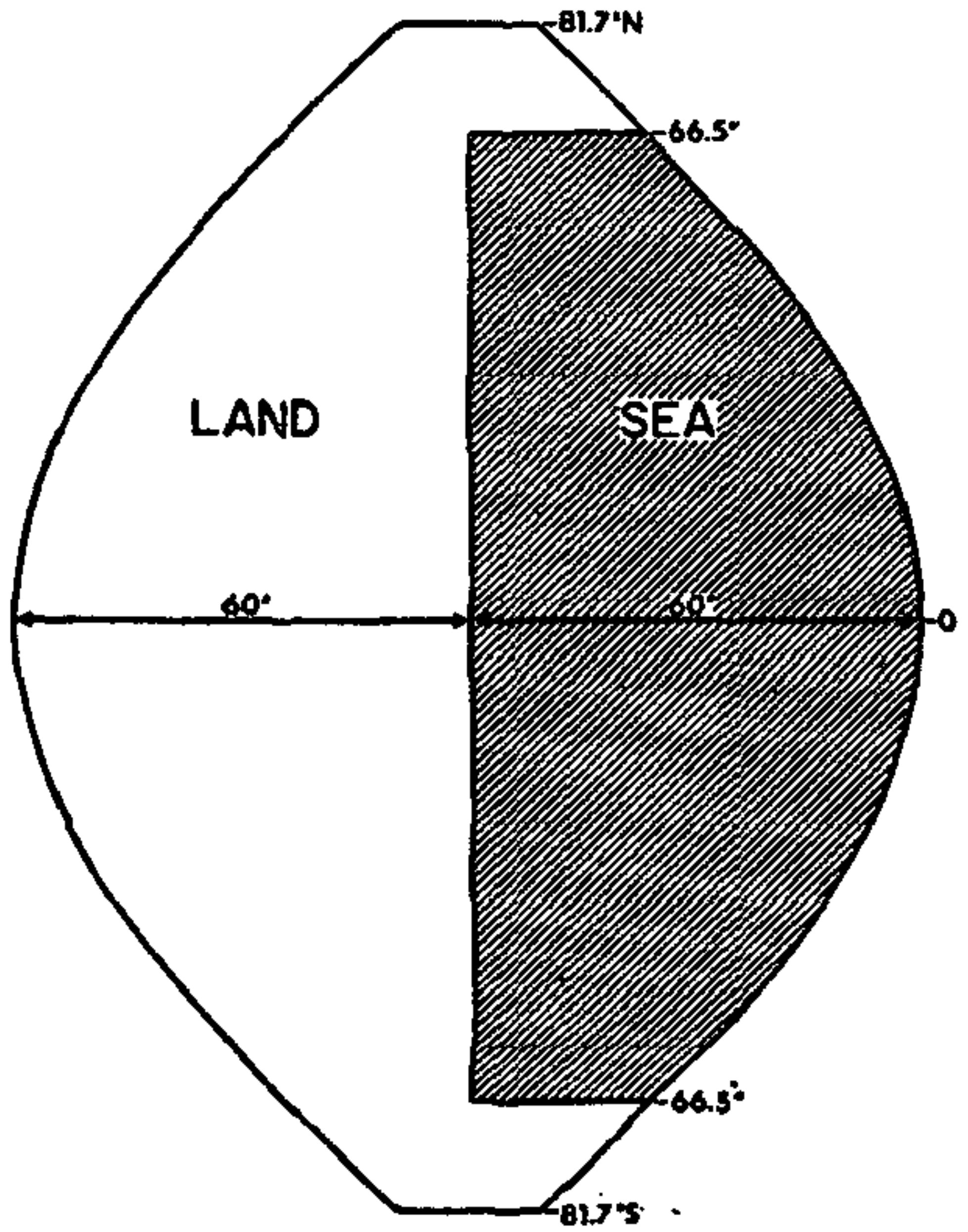
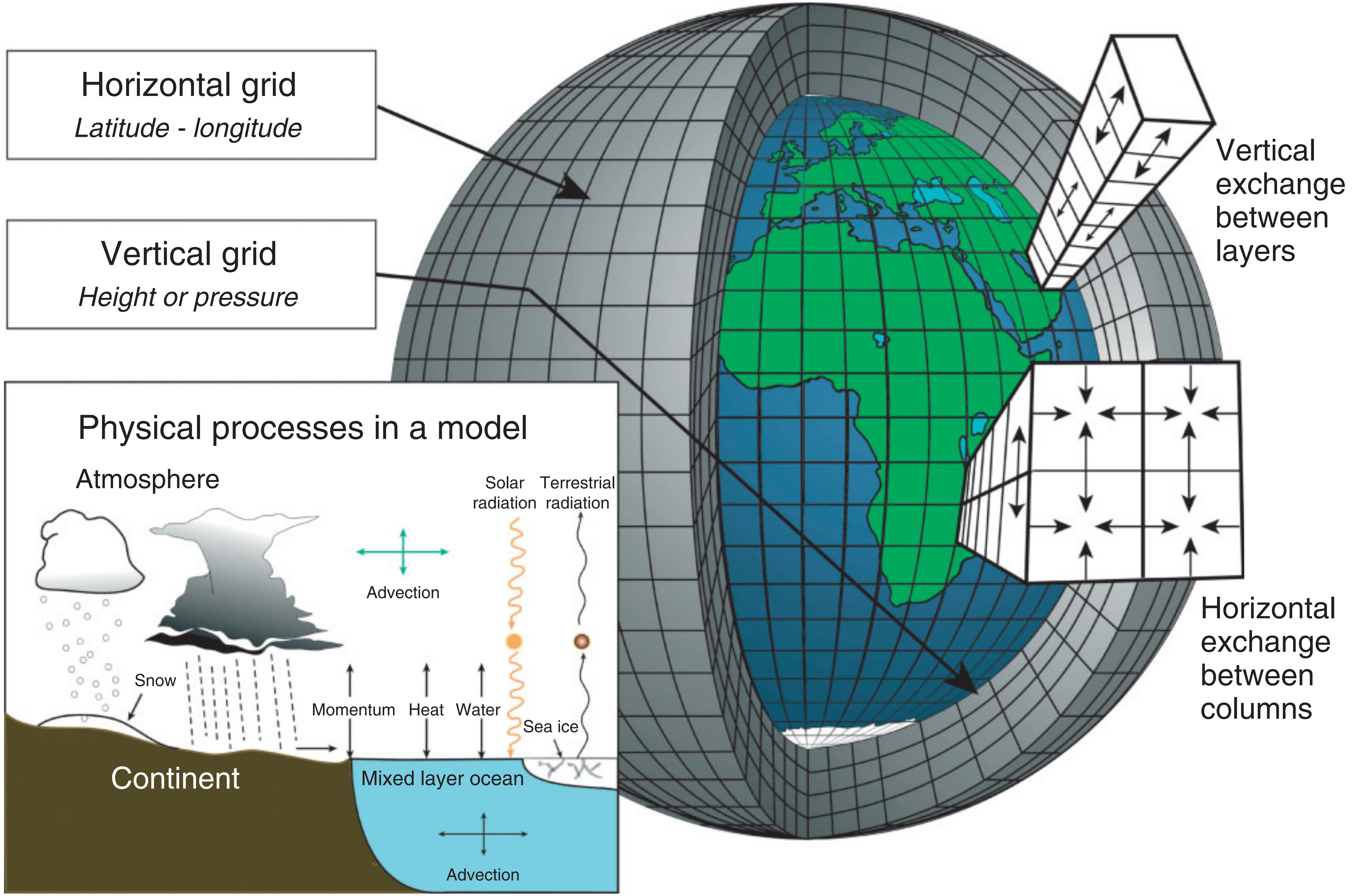


FIG. 1. Ocean-continent configuration of the model.

Syukuro Manabe — Nobel Prize in Physics 2021

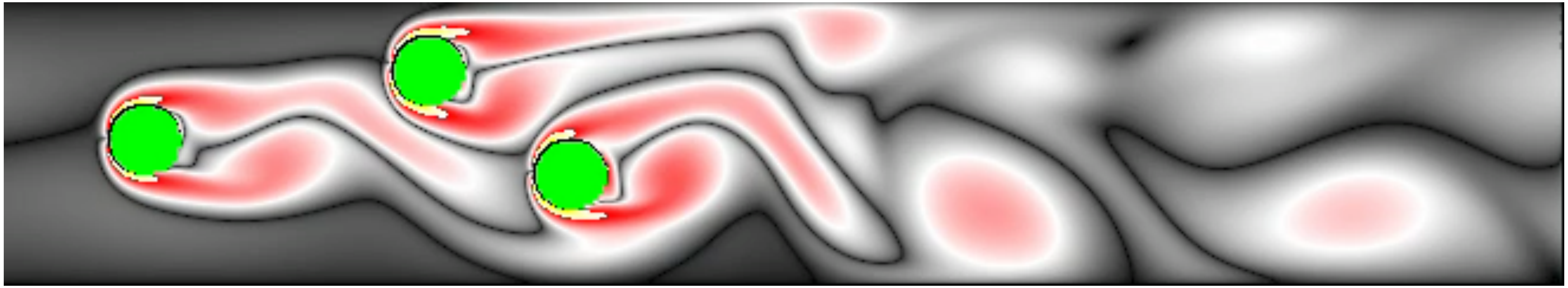
Thermal Equilibrium of the Atmosphere with a Given Distribution of Relative Humidity, Journal of the Atmosphere Sciences

Modern GCMs (Global Circulation Models)



Edwards (2011)

Fundamental dynamics (Navier-Stokes equations)

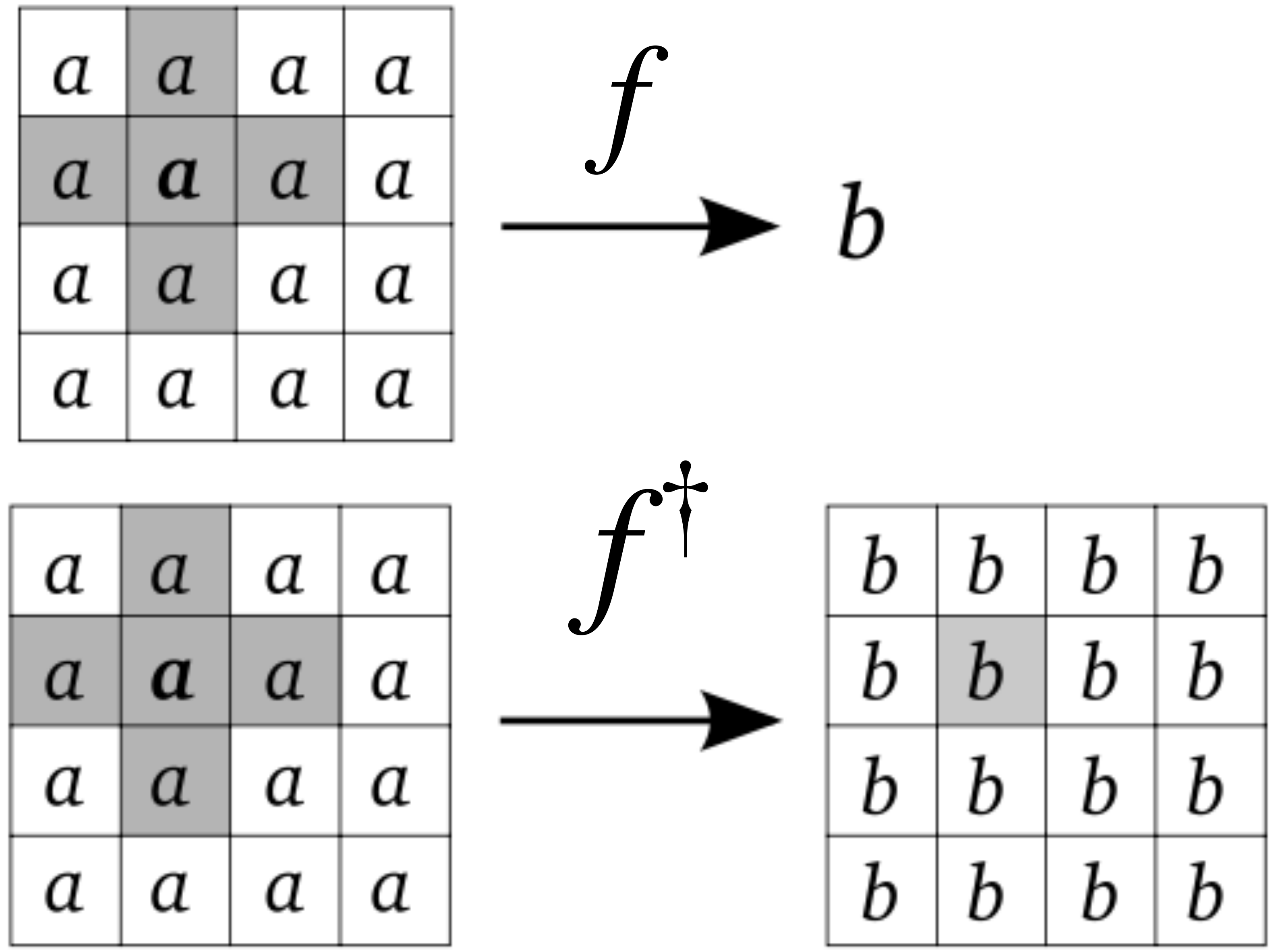


Conservation of momentum + mass for viscous fluid

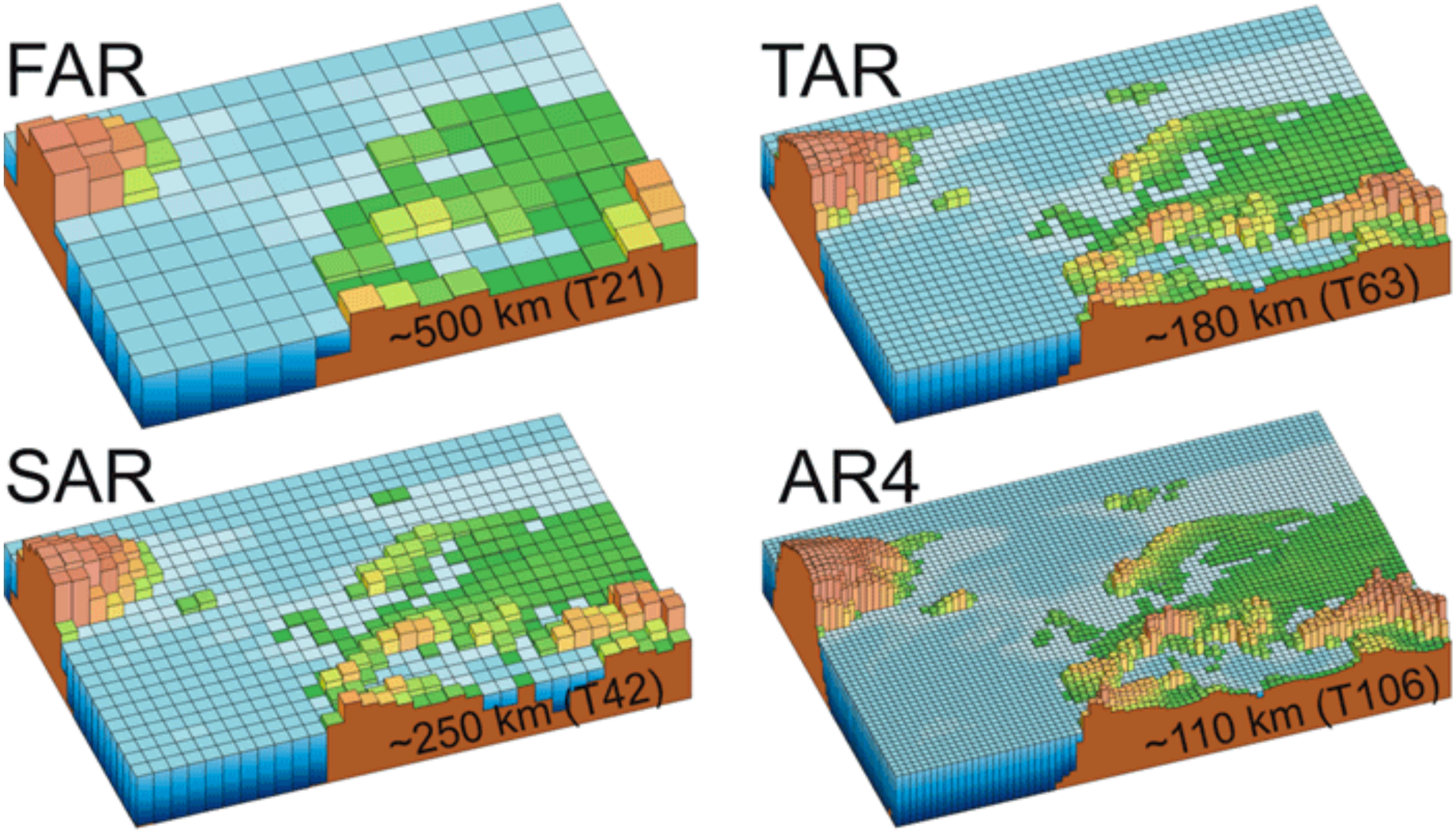
Expensive to compute!

Representable via
array comonads

$$\frac{DA \xrightarrow{f} B}{DA \xrightarrow{f^\dagger} DB}$$

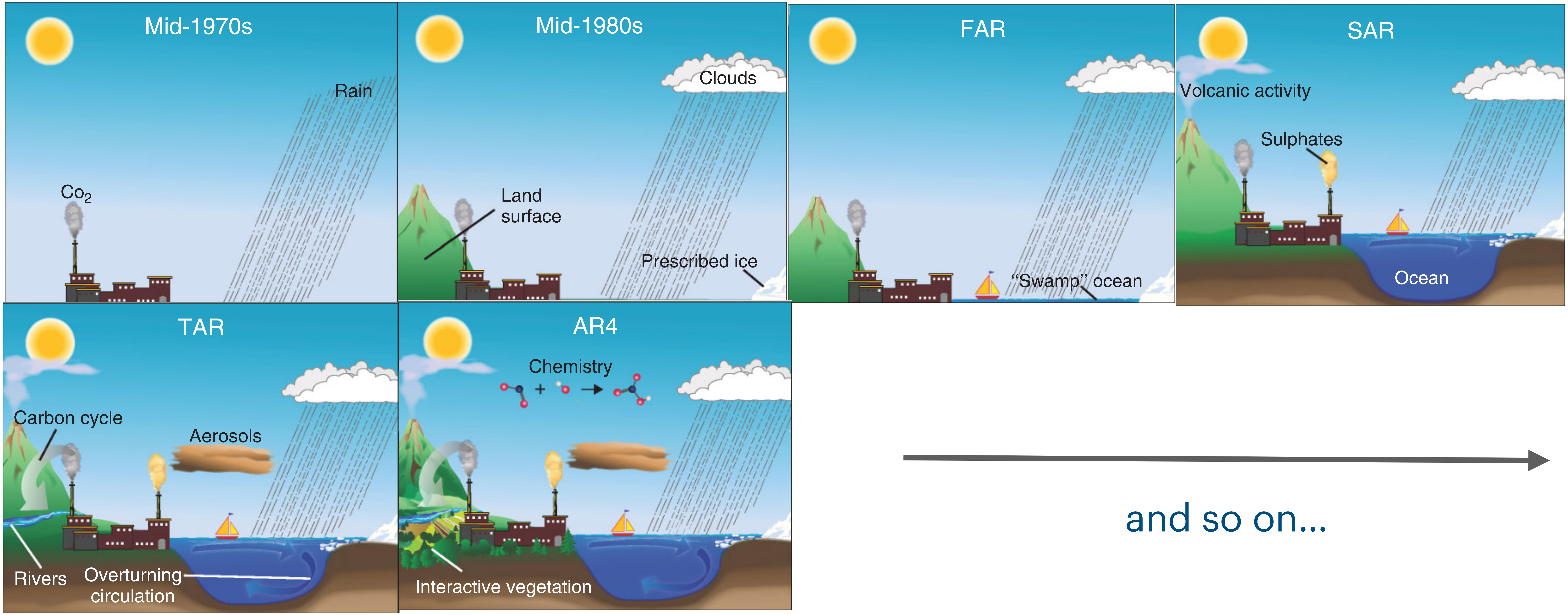


Increasing resolution over IPCC models



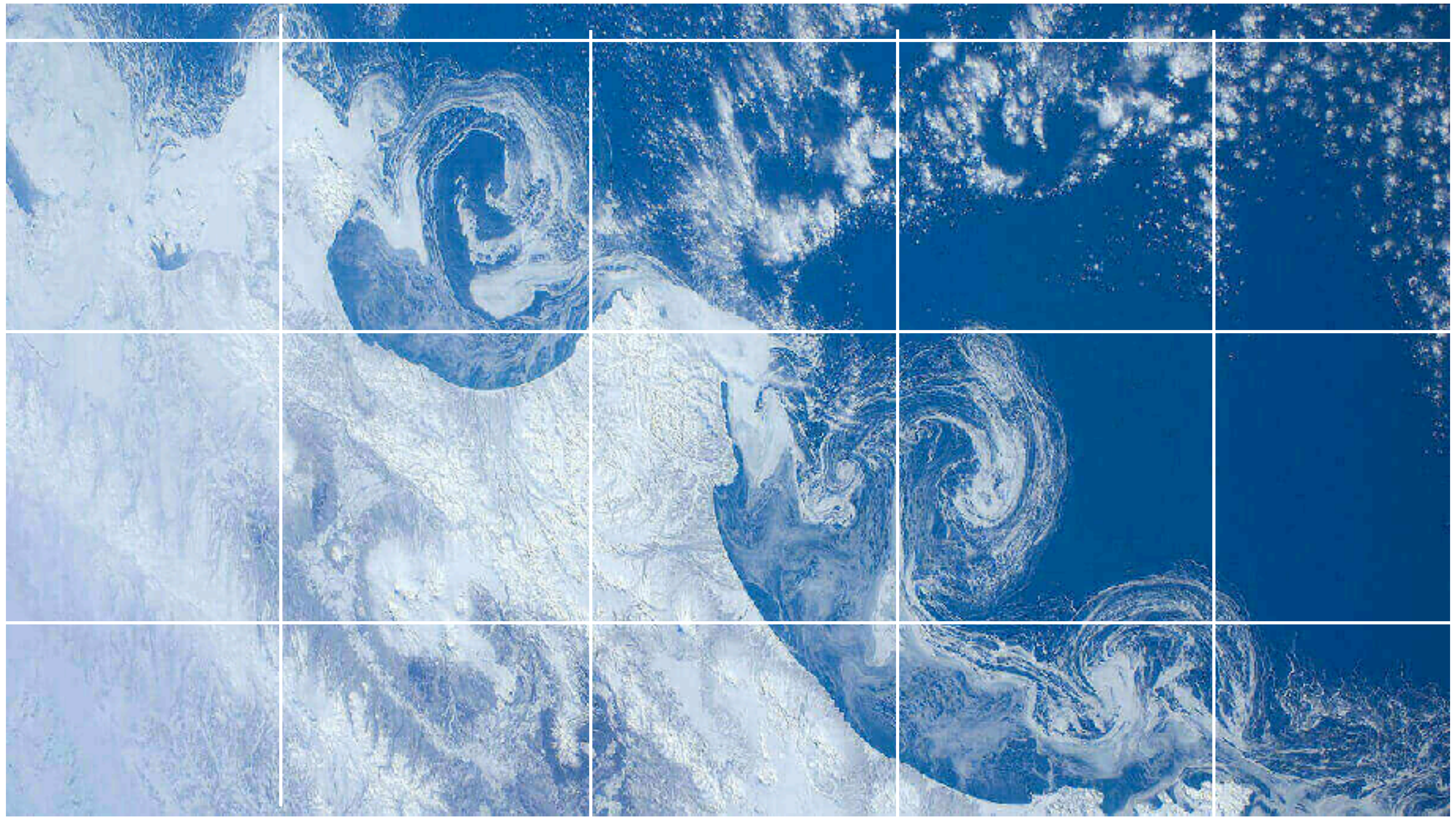
graphics from 4th IPCC report (2007)

Increasing process complexity

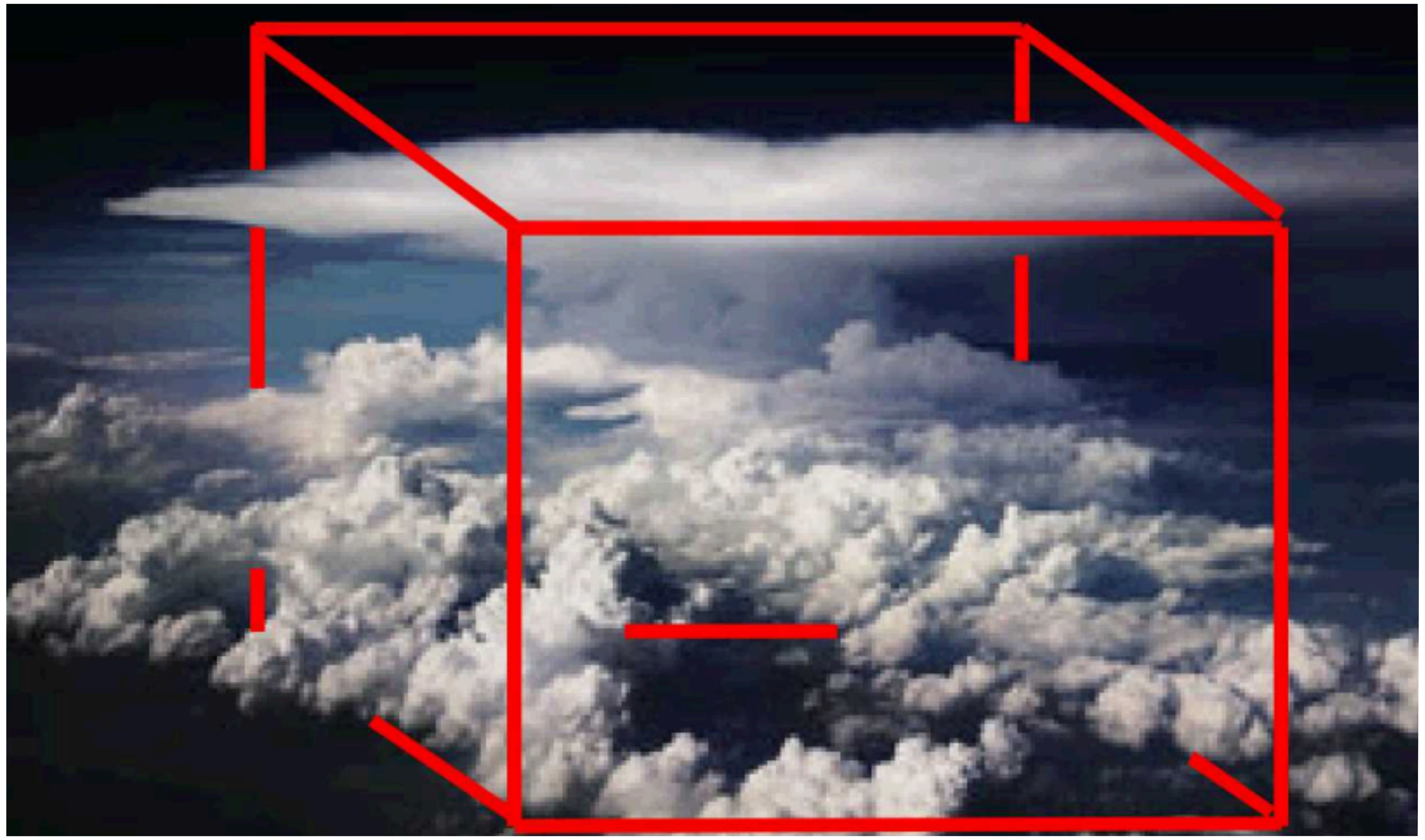


and so on...

Approximations of subgrid processes



NASA / Wikimedia Commons



Hillman et al. 2020

Source of uncertainty in models

UNIVERSITY OF CAMBRIDGE

Institute of Computing for Climate Science

+3 person operations team

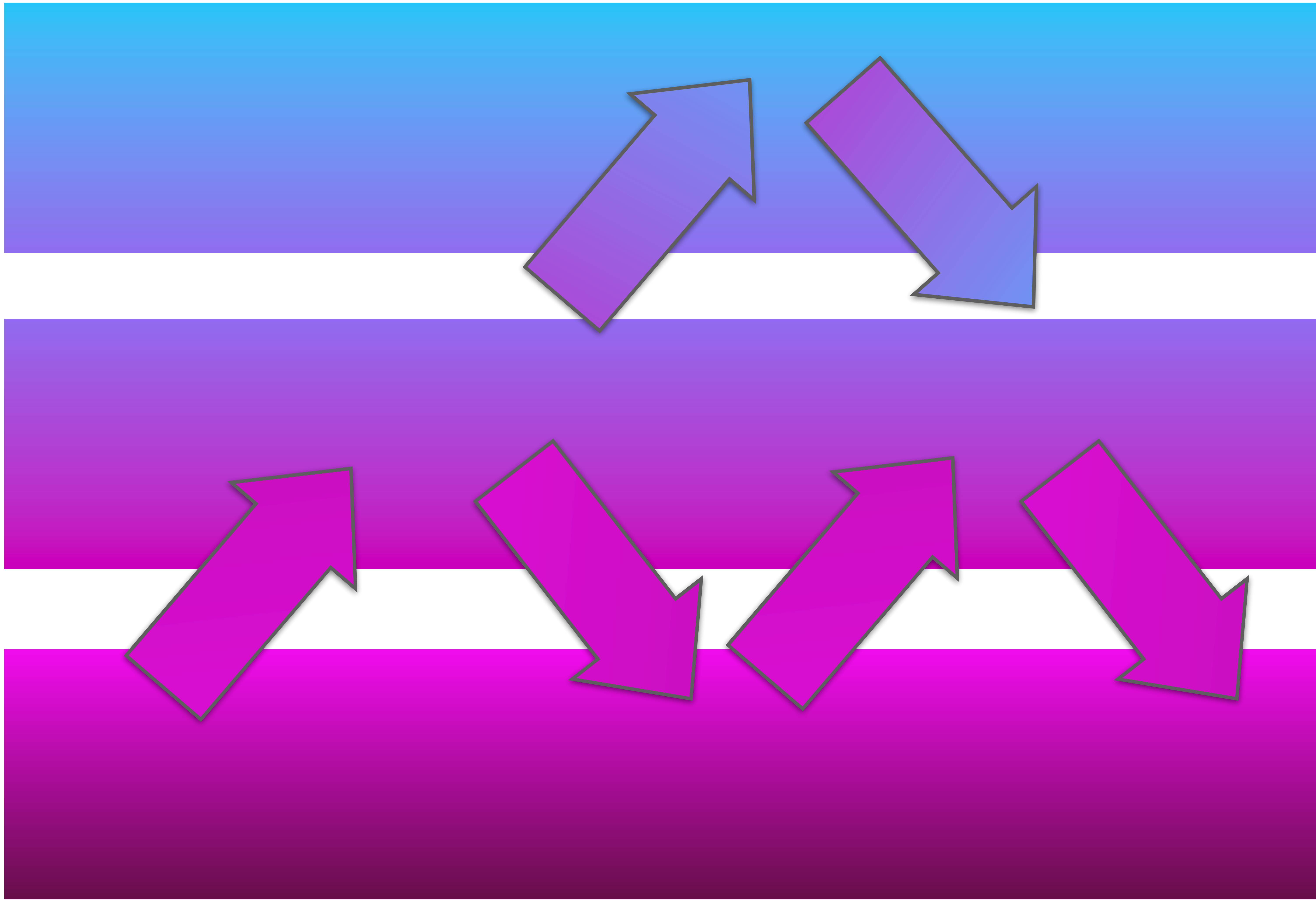
3 senior postdocs (advanced fellows)

Team of 7 amazing Research Software Engineers

Open research questions
5–30 years

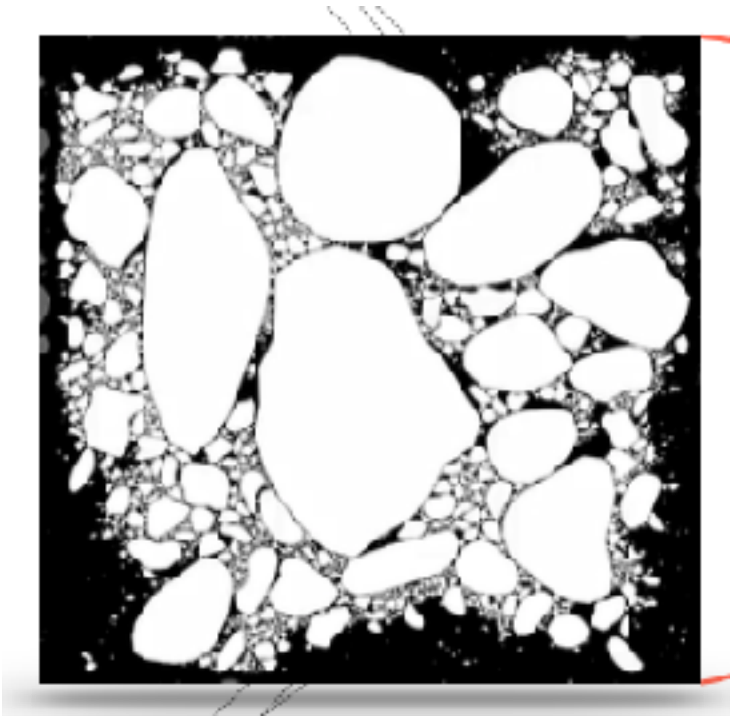
Cross-cutting concerns
2–5 years

Immediate impact
Reactive
6 months – 2 years



Immediate work

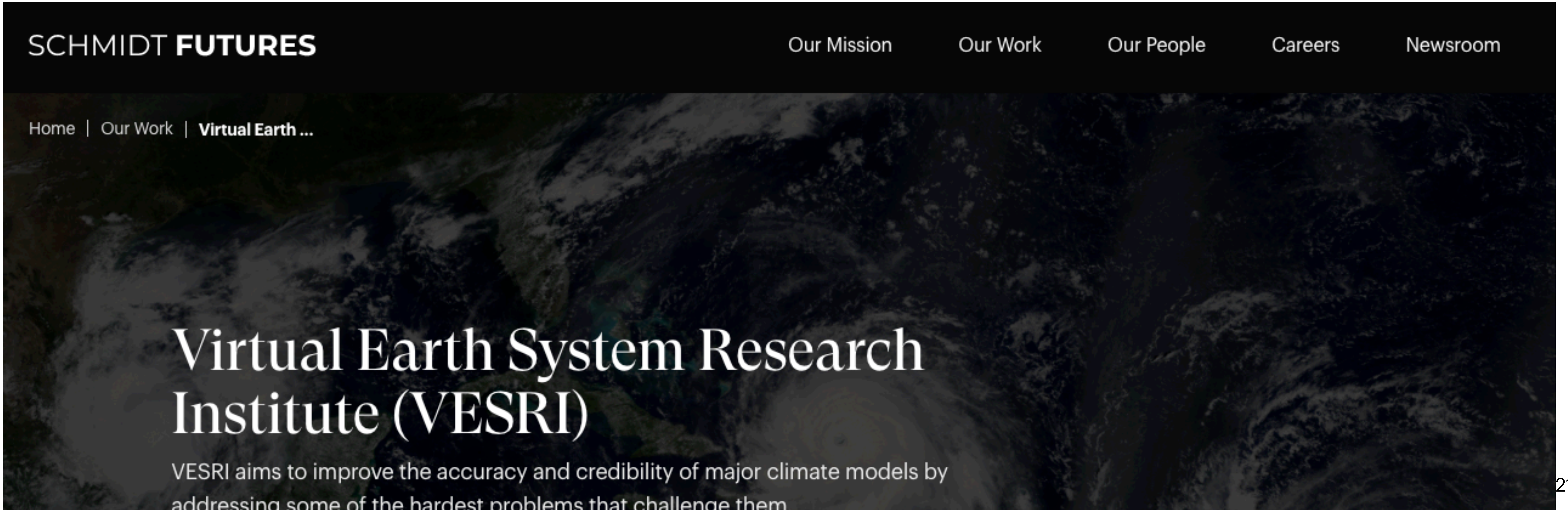
DataWave



SASIP

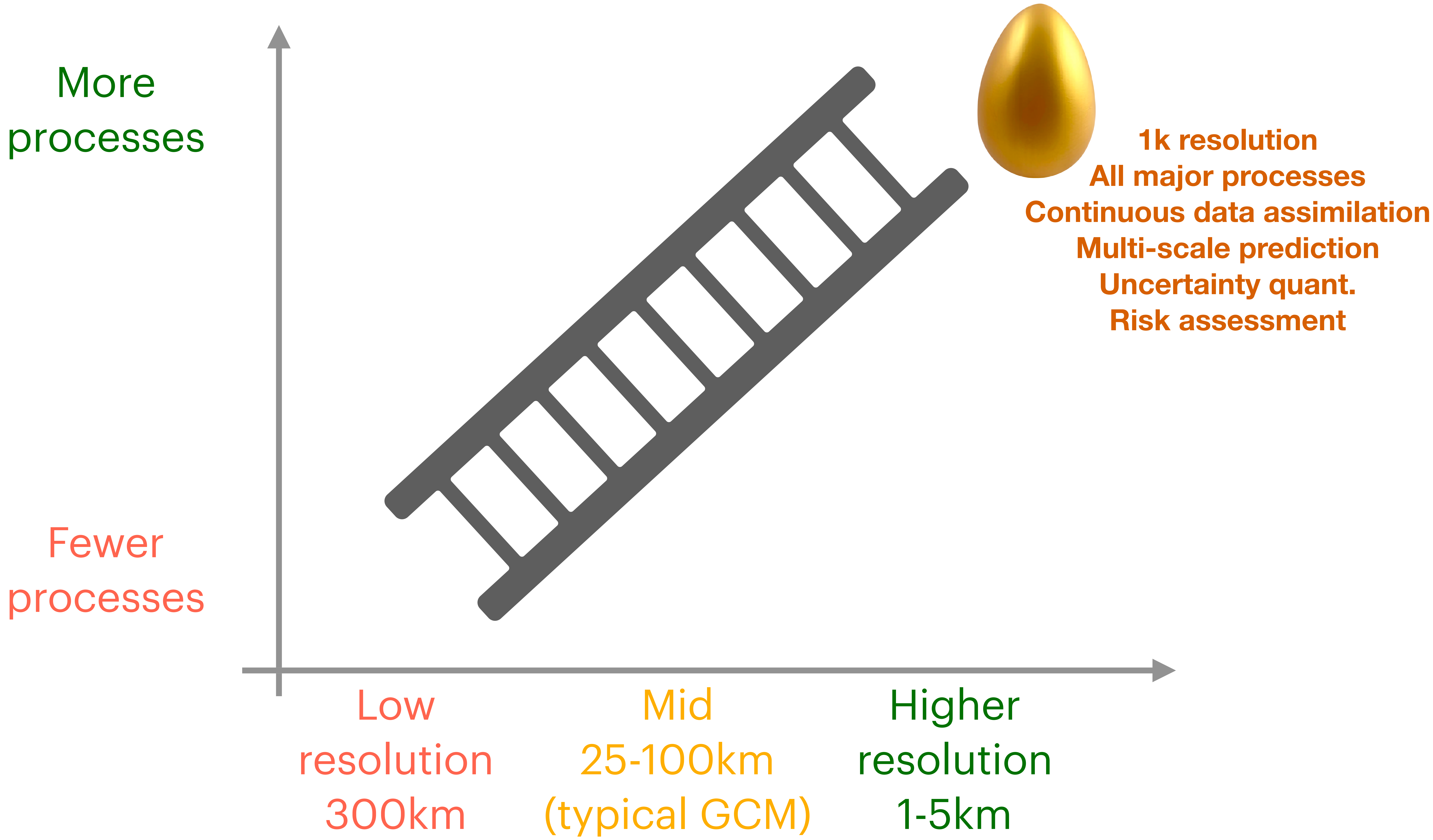
LEMONTREE

M^{2}LInES

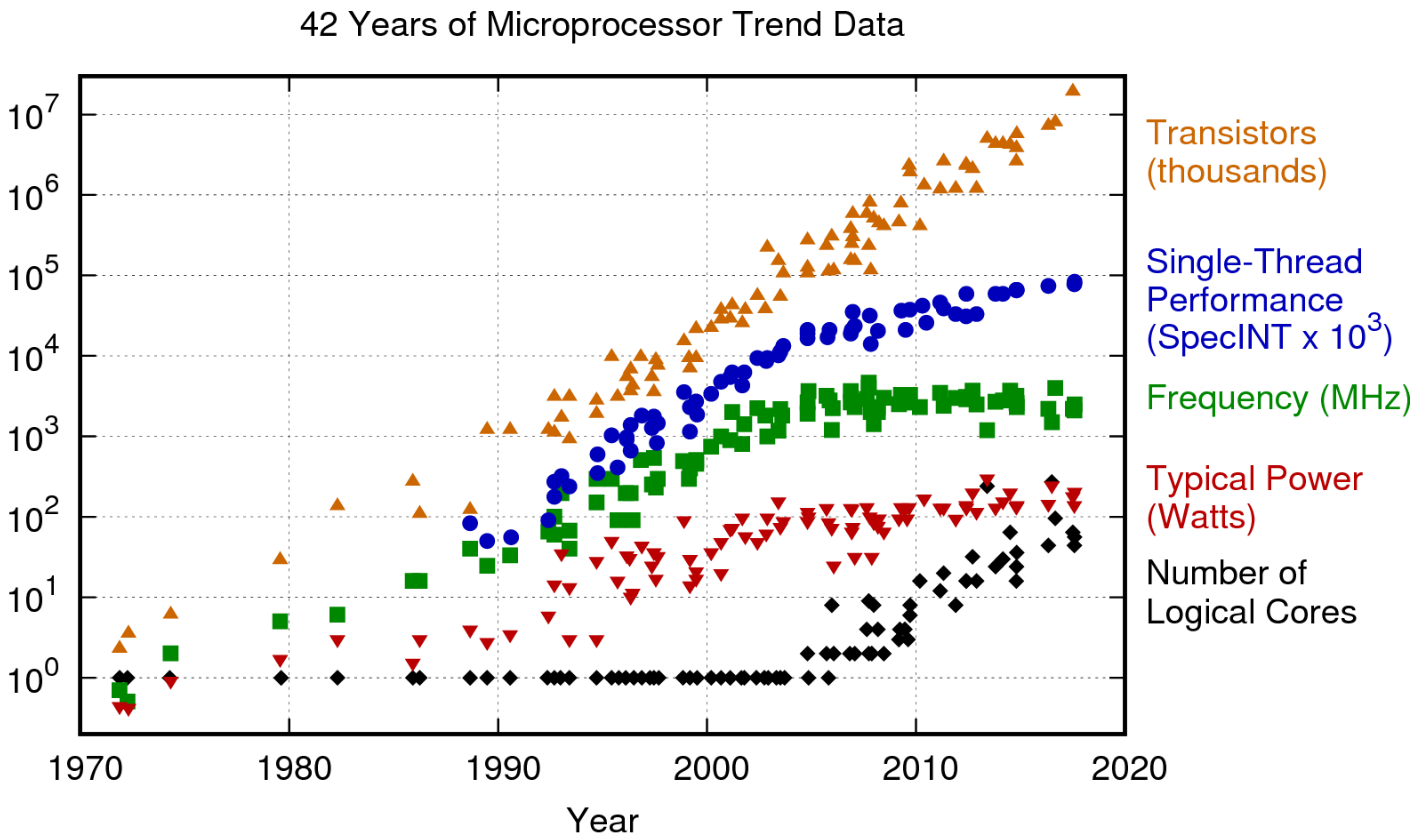


Medium-long term work

Better prediction: “climbing the ladder” (Charney)



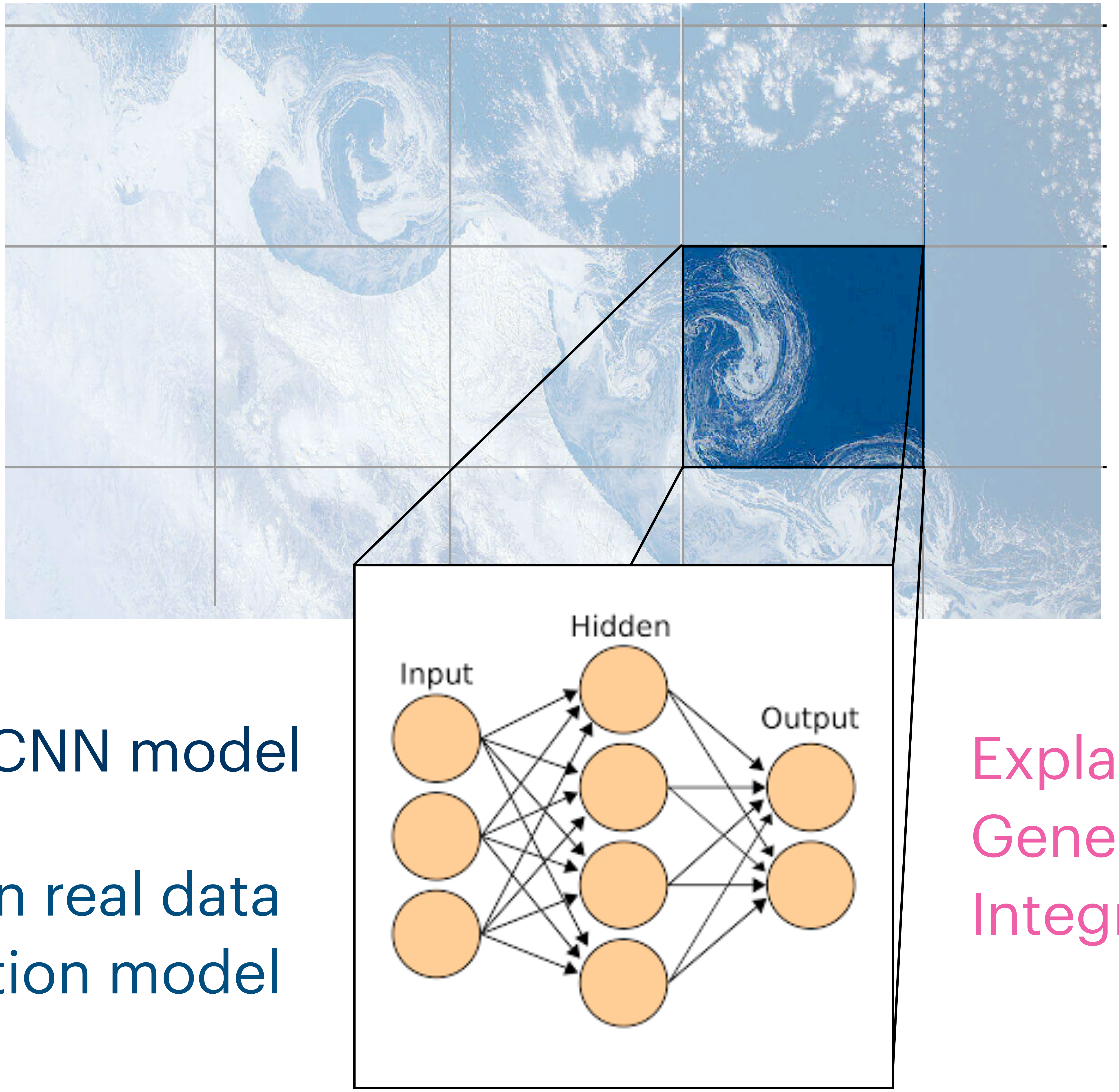
Scaling computation



Original data up to the year 2010 collected and plotted by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, and C. Batten
New plot and data collected for 2010-2017 by K. Rupp

Computers becoming bigger not faster

Data-driven subgrid models



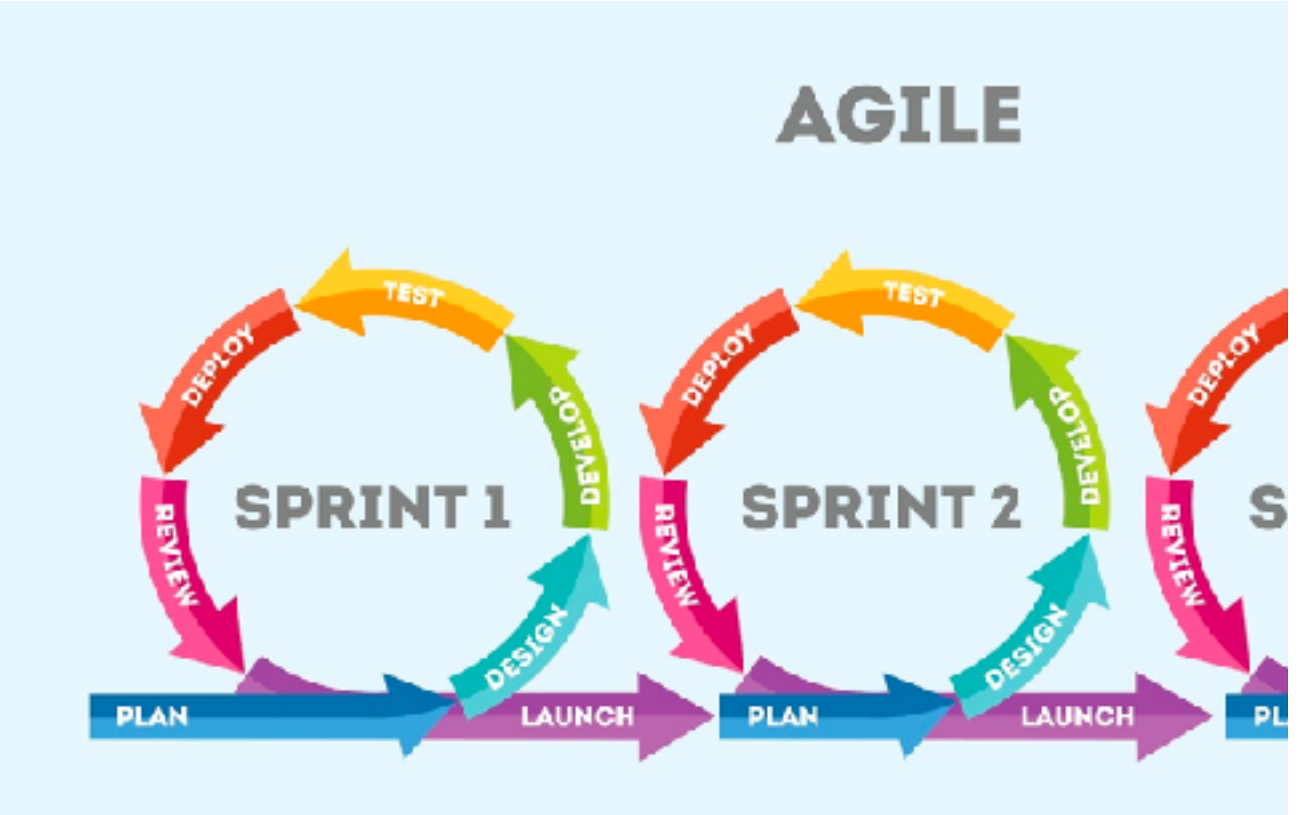
ANN or CNN model

Train on real data
or high-resolution model

Explainability?
Generalisability?
Integration into GCM?

Scaling collaboration: Deploy and train in software engineering tools & techniques

Processes



Version control & public curators

Build systems & containers

Debugging

Profiling

Testing and verification

Structural and cultural/sociological change happening

Software Sustainability Institute

RSE

Society of Research Software Engineers

software carpentry

BETTER SOFTWARE BETTER RESEARCH

The Turing Way

<ReSA>

Research Software Alliance

Scaling communication

i.e. <u>programming</u>

Models in the past...

= maths! (equations in $\mathbb{R}$)

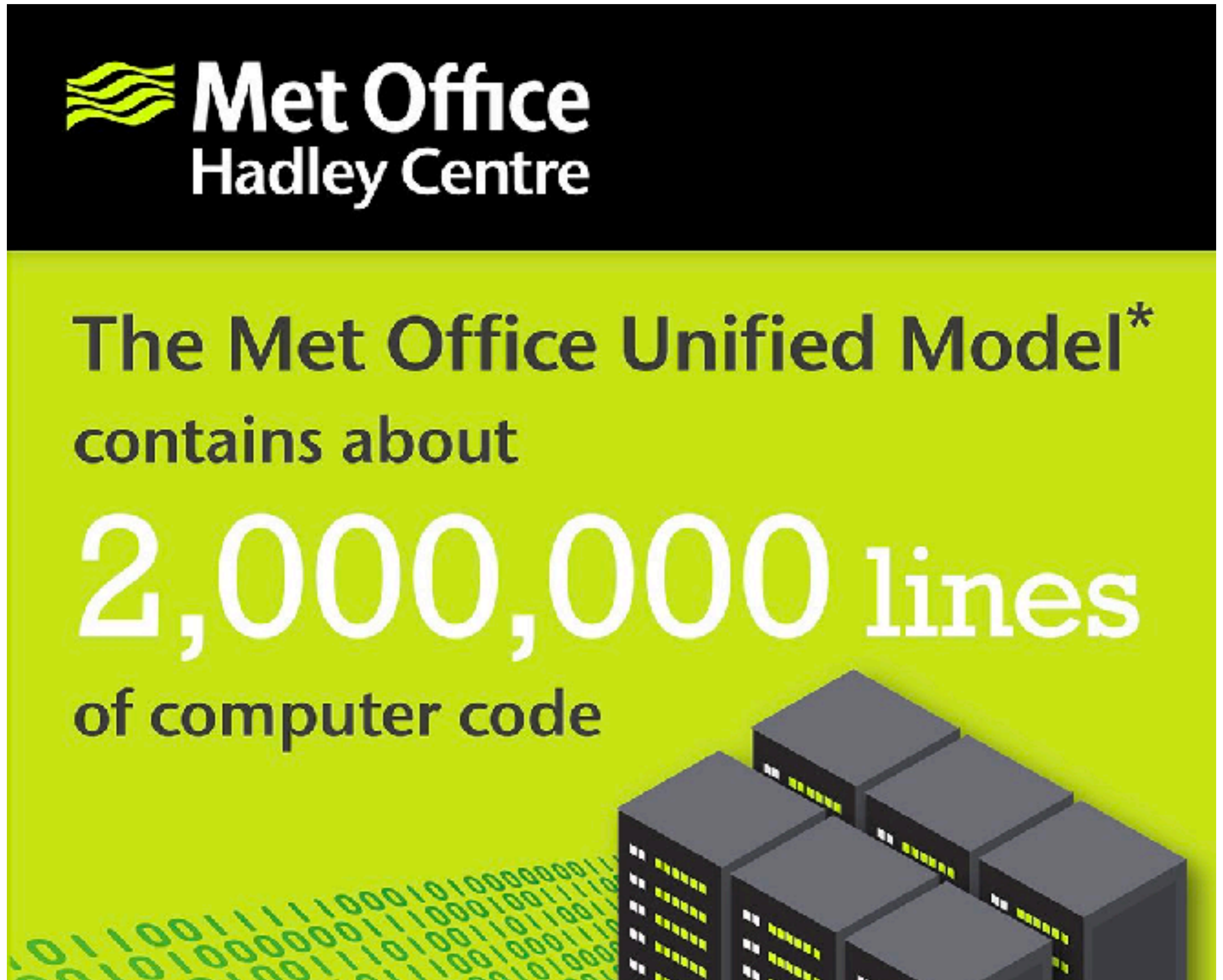
$$F = G\frac{m_1 m_2}{r^2}$$

Isaac Newton

Robert Hooke

Models now...

= code (and lots of it)



Example 1D heat equation

Abstract model

$$\frac{\partial \phi}{\partial t} = \alpha \frac{\partial^2 \phi}{\partial x^2}$$

Solution strategy

$$\phi_x^t = \phi_x^{t-1} + \frac{\alpha \Delta t}{\Delta x^2}(\phi_{x+1}^{t-1} + 2\phi_x^{t-1} + \phi_{x-1}^{t-1})$$

Prediction calculation

```
tend = ...          % end time
xmax = ...          % length of material
dt   = ...          % time resolution
dx   = ...          % space resolution
alpha = ...         % diffusion coefficient
nt = tend/dt        % # of time steps
nx = xmax/dx        % # of space steps
r = alpha*dt/dx^2   % constant in solution

real h(0,nx),        % heat fun. (discretised
     h_old(0, nx);   % in space) at t and t-1

do t = 0, nt
  h_old = h
  do x = 1, nx - 1
       h(i) = h_old(i) + r*(h_old(i-1))
                  - 2*h_old(i) + h_old(i+1)
  end do
end do
```

Conflation of concerns

Abstract model

Solution strategy

Prediction calculation

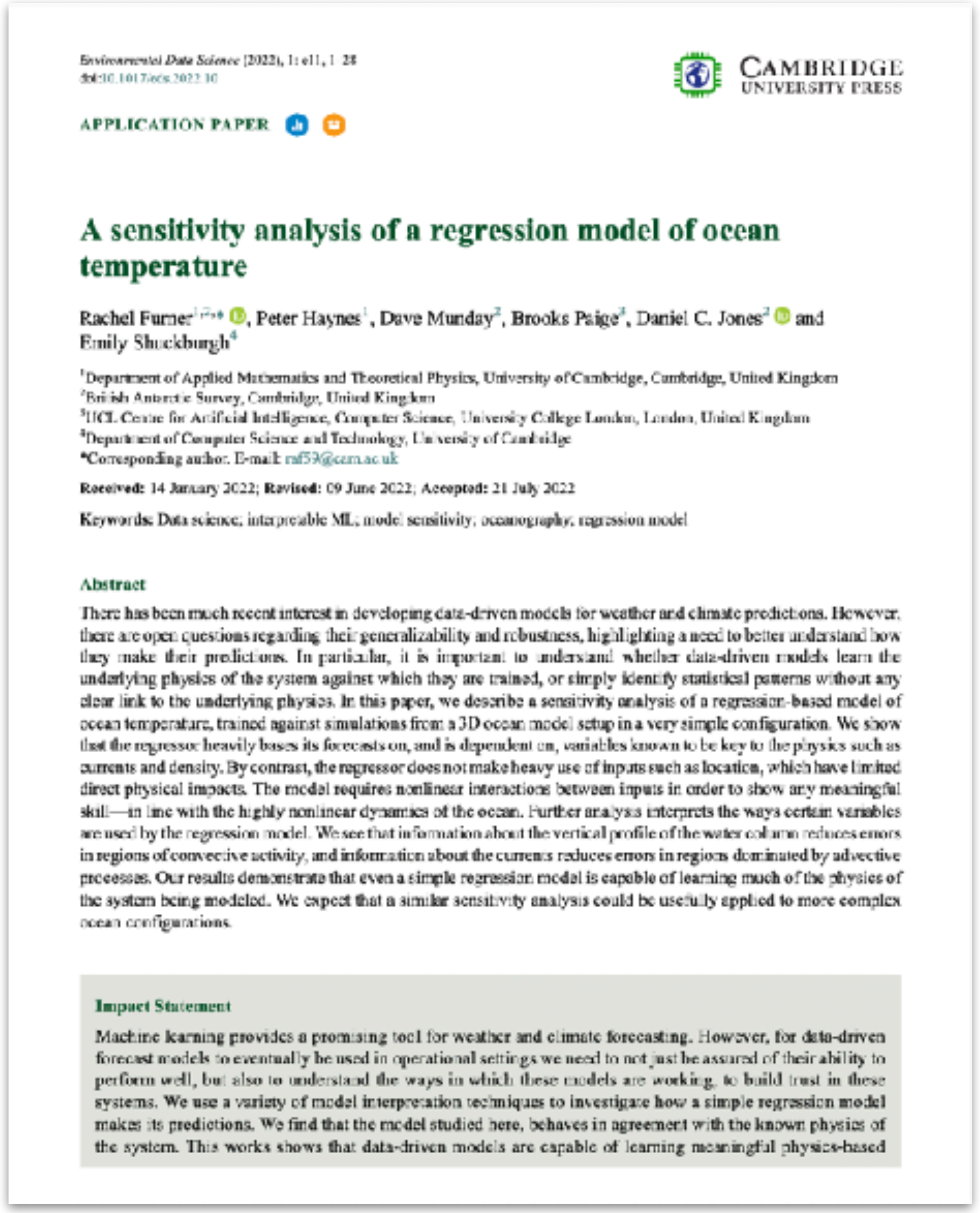
Prediction calculation

Abstract model

Solution strategy

Code conflates & hides many aspects of the model

Gap in explanation....



Environmental Data Science (2022), 1: e11, 1–28
doi:10.1017/eds.2022.10

CAMBRIDGE UNIVERSITY PRESS

APPLICATION PAPER

A sensitivity analysis of a regression model of ocean temperature

Rachel Furner[1,2,*], Peter Haynes[1], Dave Munday[2], Brooks Paige[3], Daniel C. Jones[2] and Emily Shuckburgh[4]

[1]Department of Applied Mathematics and Theoretical Physics, University of Cambridge, Cambridge, United Kingdom
[2]British Antarctic Survey, Cambridge, United Kingdom
[3]UCL Centre for Artificial Intelligence, Computer Science, University College London, London, United Kingdom
[4]Department of Computer Science and Technology, University of Cambridge
*Corresponding author. E-mail: raf59@cam.ac.uk

Received: 14 January 2022; **Revised:** 09 June 2022; **Accepted:** 21 July 2022

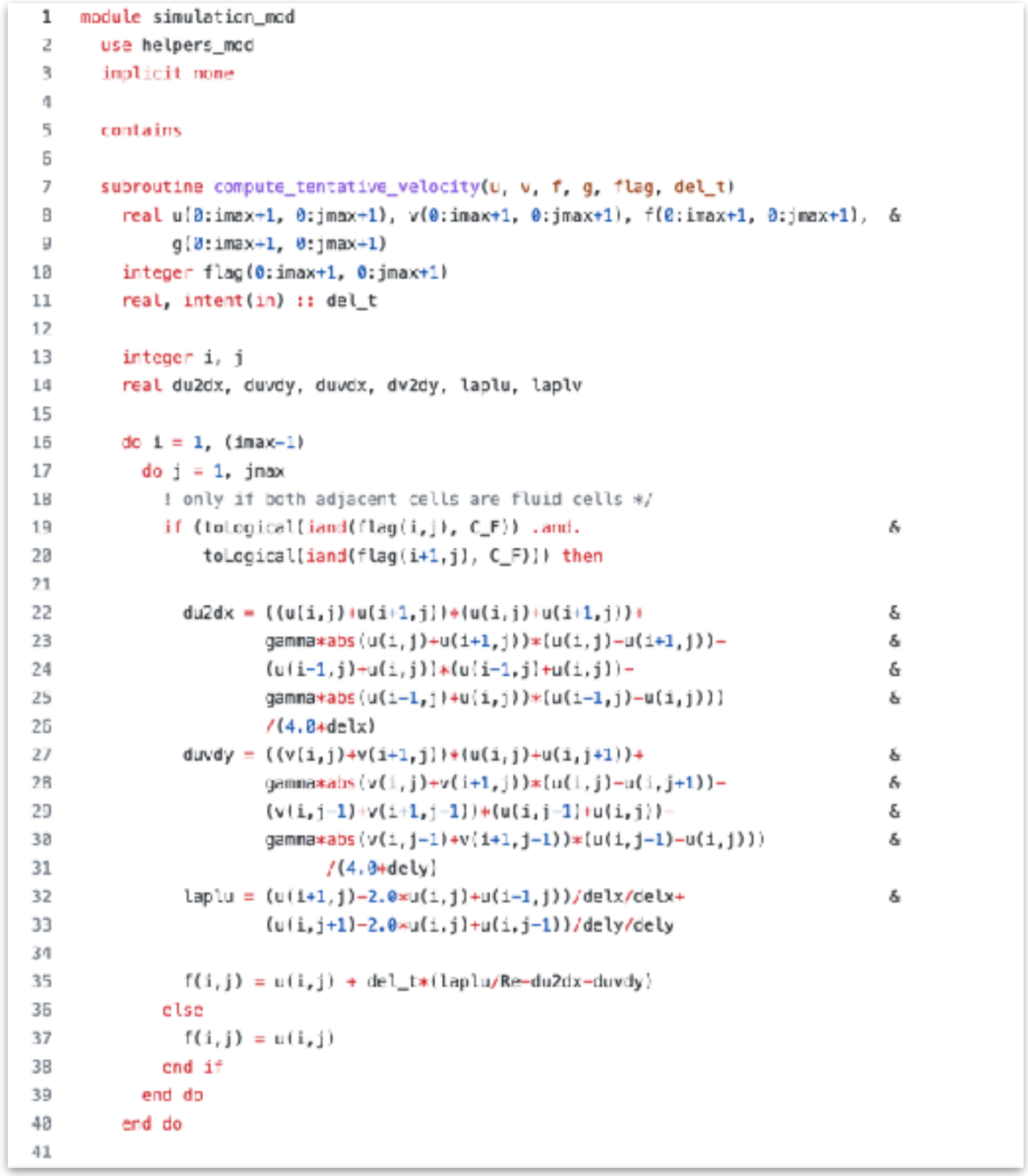
Keywords: Data science; interpretable ML; model sensitivity; oceanography; regression model

Abstract

There has been much recent interest in developing data-driven models for weather and climate predictions. However, there are open questions regarding their generalizability and robustness, highlighting a need to better understand how they make their predictions. In particular, it is important to understand whether data-driven models learn the underlying physics of the system against which they are trained, or simply identify statistical patterns without any clear link to the underlying physics. In this paper, we describe a sensitivity analysis of a regression-based model of ocean temperature, trained against simulations from a 3D ocean model setup in a very simple configuration. We show that the regressor heavily bases its forecasts on, and is dependent on, variables known to be key to the physics such as currents and density. By contrast, the regressor does not make heavy use of inputs such as location, which have limited direct physical impacts. The model requires nonlinear interactions between inputs in order to show any meaningful skill—in line with the highly nonlinear dynamics of the ocean. Further analysis interprets the ways certain variables are used by the regression model. We see that information about the vertical profile of the water column reduces errors in regions of convective activity, and information about the currents reduces errors in regions dominated by advective processes. Our results demonstrate that even a simple regression model is capable of learning much of the physics of the system being modeled. We expect that a similar sensitivity analysis could be usefully applied to more complex ocean configurations.

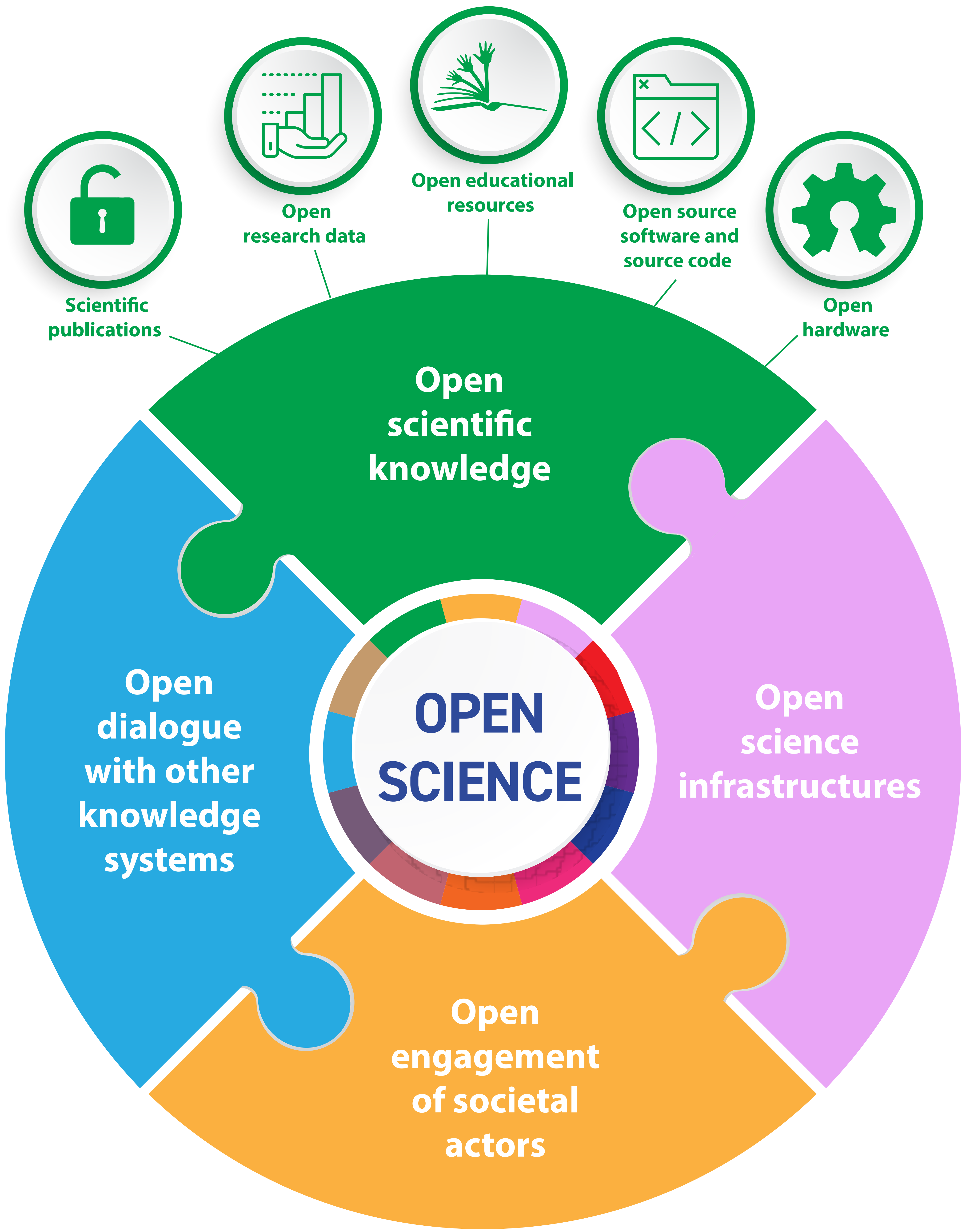
Impact Statement

Machine learning provides a promising tool for weather and climate forecasting. However, for data-driven forecast models to eventually be used in operational settings we need to not just be assured of their ability to perform well, but also to understand the ways in which these models are working, to build trust in these systems. We use a variety of model interpretation techniques to investigate how a simple regression model makes its predictions. We find that the model studied here, behaves in agreement with the known physics of the system. This works shows that data-driven models are capable of learning meaningful physics-based



```
module simulation_mod
  use helpers_mod
  implicit none

  contains

  subroutine compute_tentative_velocity(u, v, f, g, flag, del_t)
    real u(0:imax+1, 0:jmax+1), v(0:imax+1, 0:jmax+1), f(0:imax+1, 0:jmax+1),  &
         g(0:imax+1, 0:jmax+1)
    integer flag(0:imax+1, 0:jmax+1)
    real, intent(in) :: del_t

    integer i, j
    real du2dx, duvdy, duvdx, dv2dy, laplu, laplv

    do i = 1, (imax-1)
      do j = 1, jmax
        ! only if both adjacent cells are fluid cells */
        if (toLogical(iand(flag(i,j), C_F)) .and.                               &
            toLogical(iand(flag(i+1,j), C_F))) then

          du2dx = ((u(i,j)+u(i+1,j))*(u(i,j)+u(i+1,j))+                        &
                  gamma*abs(u(i,j)+u(i+1,j))*(u(i,j)-u(i+1,j))-                 &
                  (u(i-1,j)+u(i,j))*(u(i-1,j)+u(i,j))-                          &
                  gamma*abs(u(i-1,j)+u(i,j))*(u(i-1,j)-u(i,j)))                 &
                  /(4.0*delx)
          duvdy = ((v(i,j)+v(i+1,j))*(u(i,j)+u(i,j+1))+                        &
                  gamma*abs(v(i,j)+v(i+1,j))*(u(i,j)-u(i,j+1))-                 &
                  (v(i,j-1)+v(i+1,j-1))*(u(i,j-1)+u(i,j))-                      &
                  gamma*abs(v(i,j-1)+v(i+1,j-1))*(u(i,j-1)-u(i,j)))             &
                       /(4.0*dely)
          laplu = (u(i+1,j)-2.0*u(i,j)+u(i-1,j))/delx/delx+                    &
                  (u(i,j+1)-2.0*u(i,j)+u(i,j-1))/dely/dely

          f(i,j) = u(i,j) + del_t*(laplu/Re-du2dx-duvdy)
        else
          f(i,j) = u(i,j)
        end if
      end do
    end do
```

Abstract model | Solution strategy | Prediction calculation



UNESCO 2021 Open Science recommendation


Open research data
Open educational resources
Open source software and source code
Open hardware
Open scientific

But.. sharing code includes sharing bugs

+ assumptions
+ incidental decisions
+ approximations

Open problem: separating and relating concerns

Abstract model Solution strategy Prediction calculation

Partial solutions

- Extra technical documentation
- Clear systems design
- High modularity

Could there be better support via a **programming language tailored to science?**

Roadmap

1. Computer science engagement with scientists
2. New systems for abstraction and specification

3. Evolutionary approach for languages

Procedia Computer Science
Volume 29, 2014, Pages 713–727
ICCS 2014. 14th International Conference on Computational Science

A computational science agenda for programming language research

Dominic Orchard[1], Andrew Rice[2]

[1] Computer Laboratory, University of Cambridge
dominic.orchard@cl.cam.ac.uk
[2] Computer Laboratory, University of Cambridge
andrew.rice@cl.cam.ac.uk

Abstract
Scientific models are often expressed as large and complicated programs. These programs embody numerous assumptions made by the developer (*e.g.*, for differential equations, the discretization strategy and resolution). The complexity and pervasiveness of these assumptions means that often the only true description of the model is the software itself. This has led various researchers to call for scientists to publish their source code along with their papers. We argue that this is unlikely to be beneficial since it is almost impossible to separate implementation assumptions from the original scientific intent. Instead we advocate higher-level abstractions in programming languages, coupled with lightweight verification techniques such as specification and type systems. In this position paper, we suggest several novel techniques and outline an evolutionary approach to applying these to existing and future models. One-dimensional heat flow is used as an example throughout.

Keywords: computational science, modelling, programming, verification, reproducibility, abstractions, type systems, language design

1 Introduction

natural & physical sciences

computer science

natural & physical sciences

computer science

natural & physical sciences

computer science

Let's bridge the chasm!

https://camfort.github.io/

CamFort Lightweight verification tools for science

```fortran
program energy
  != unit kg :: mass
  != unit m  :: height
  real :: mass = 3.00, gravity = 9.91, height = 4.20
  != unit kg m**2/s**2 :: potential_energy
  real :: potential_energy

  potential_energy = mass * gravity * height
end program energy
```

```
$ camfort units-check energy1.f90

energy1.f90: Consistent. 4 variables checked.
```

EPSRC Engineering and Physical Sciences Research Council

Bloomberg

Met Office Hadley Centre

CamFort

- Units-of-measure verification
- Stencil computation shape verification
- Basic Hoare logic
- FP linting checks
- Performance checks
- Allocate/deallocate well bracketed

```
!= unit(m) :: d1, d2
!= unit(s) :: t
real :: d1, d2, t, v
v = (d1 + d2)/t
```

```
if a .eq. 0.0 then
   ! . . .
```

possible source
of numerical instability

Met Office Hadley Centre

- Future work?
  - Conservation analysis

```
do i = 2 to n-1
   b(i) = a(i-1) - 2*a(i) + a(i+1)
end do
b(1) = 1.5*b(2)
b(N) = 1.5*b(N-1)
```

→ $\sum \begin{array}{|c|c|c|} \hline 1 & -2 & 1 \\ \hline \end{array} = 0$

$\wedge$

→ Boundary preserves conservation

“I don't know what the language of the year 2000 will look like, but I know it will be called Fortran." — Sir Tony Hoare (1982)

- Fortran’s evolution shows power of expressivity gains
- But success of languages is inscrutable (ride a wave?)
- Recent breakout success: Julia
- Big bet/opportunity for future climate modelling?

julia

CliMA
CLIMATE MODELING ALLIANCE

climate science

Science critical for survival of our species

computer science

Tools for the tool makers for decision making, understanding, forecasting, monitoring

Let's bridge the chasm and together program for our future

Lookout for….

Hiring 3-year postdoc soon…

https://plas4sci.github.io/

PROPL - Workshop on Programming for the Planet

Hopefully at POPL'24!

The Topos Institute Colloquium

talk October 12th

https://topos.site/topos-colloquium/

Programming Languages and Systems for Science laboratory

University of Kent

Complex models in modern science and are now routinely expressed as software. The PLAS4Sci lab (Programming Languages and Systems for Science) at the School of Computing, University of Kent is a sub-group of the PLAS group focussed on improving the state-of-the-art in programming languages, programming systems, and programming tools to support the daily work of scientists.

People

- Dominic Orchard - Lab lead
- Benjamin Orchard - Research Assistant and Research Software Engineer
- Laura Bocchi - Reader in Programming Languages
- Vilem-Benjamin Liepelt - PhD student

Partners

UNIVERSITY OF CAMBRIDGE

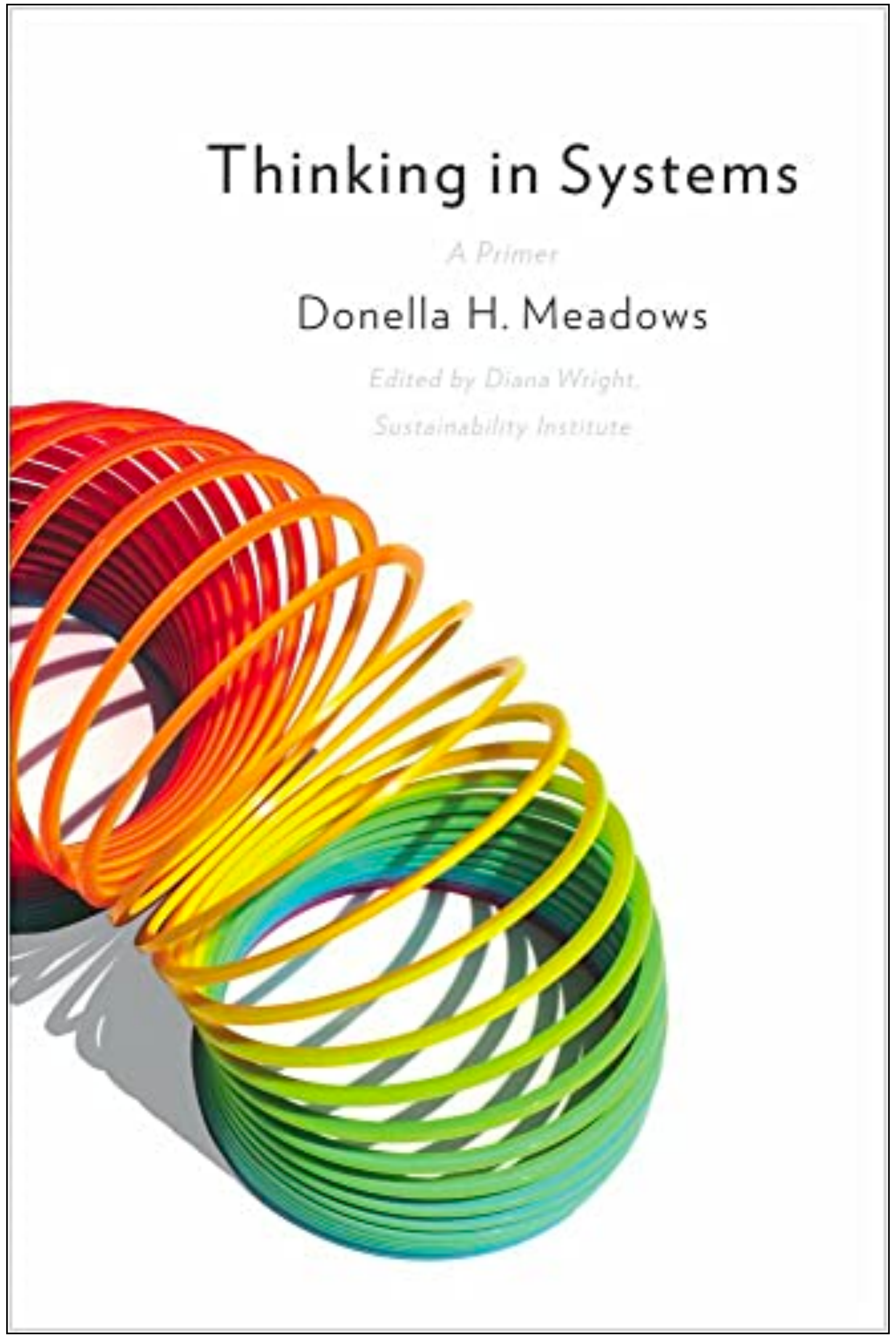
Institute of Computing for Climate Science

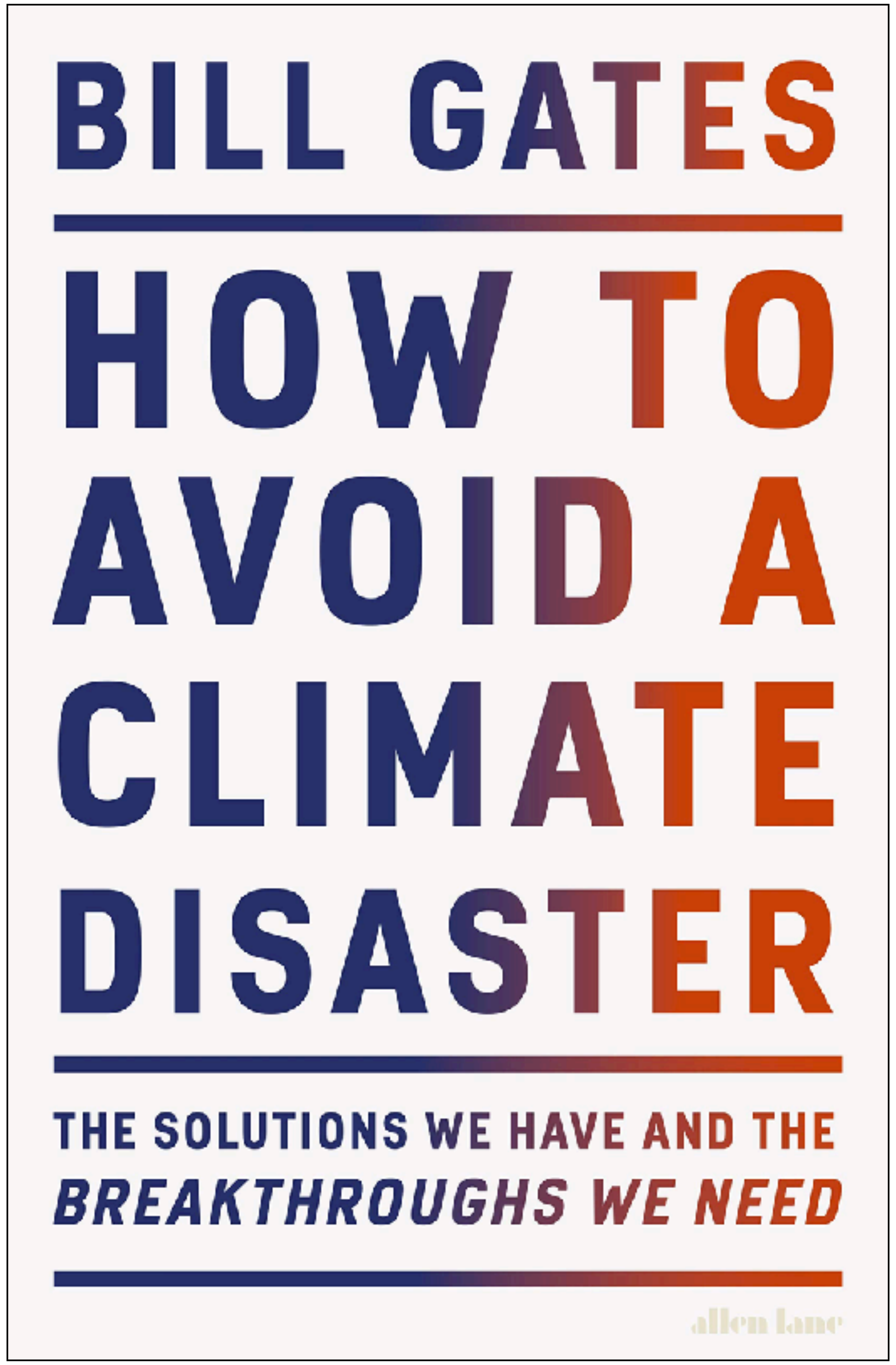
Bloomberg

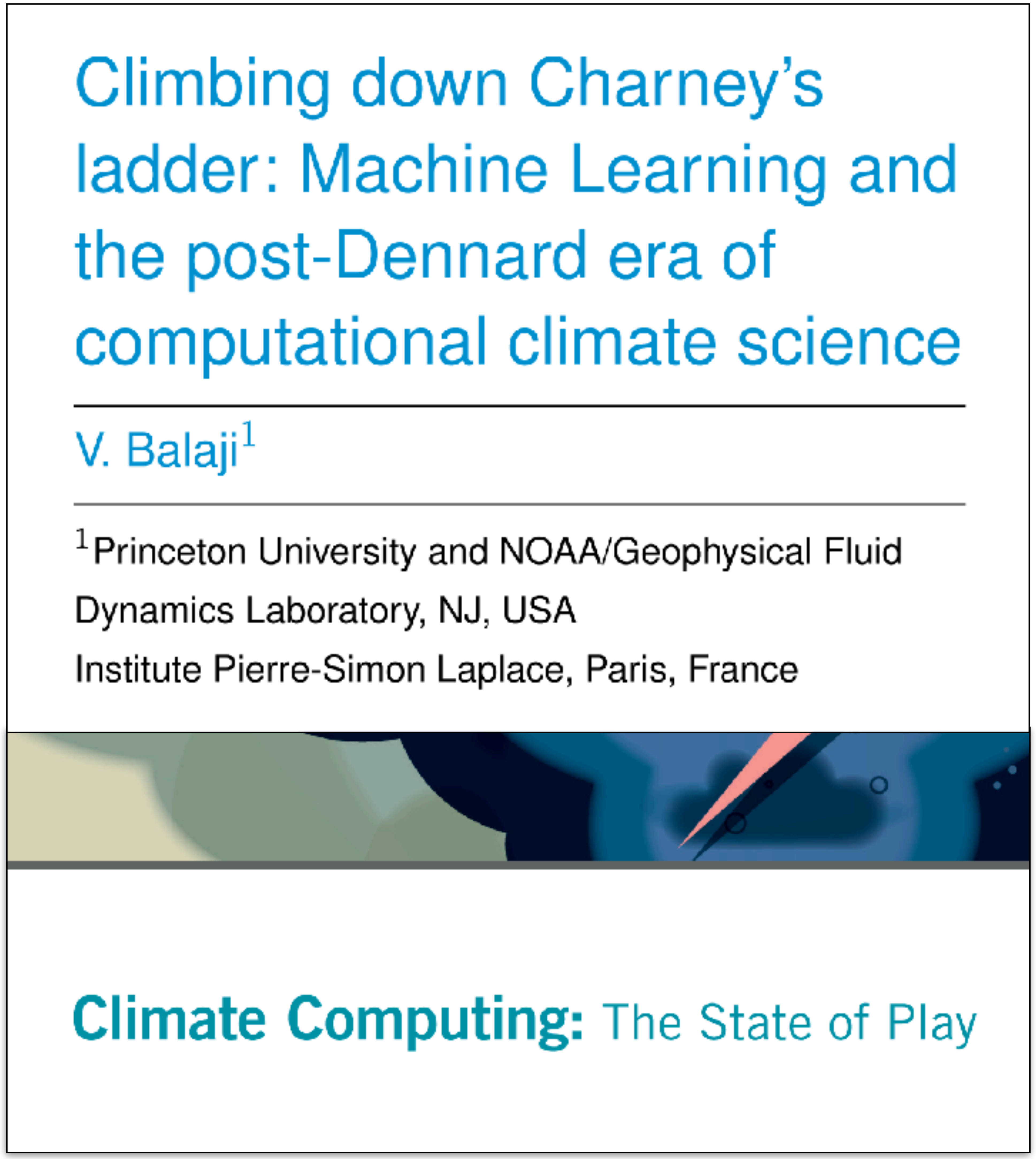
Projects

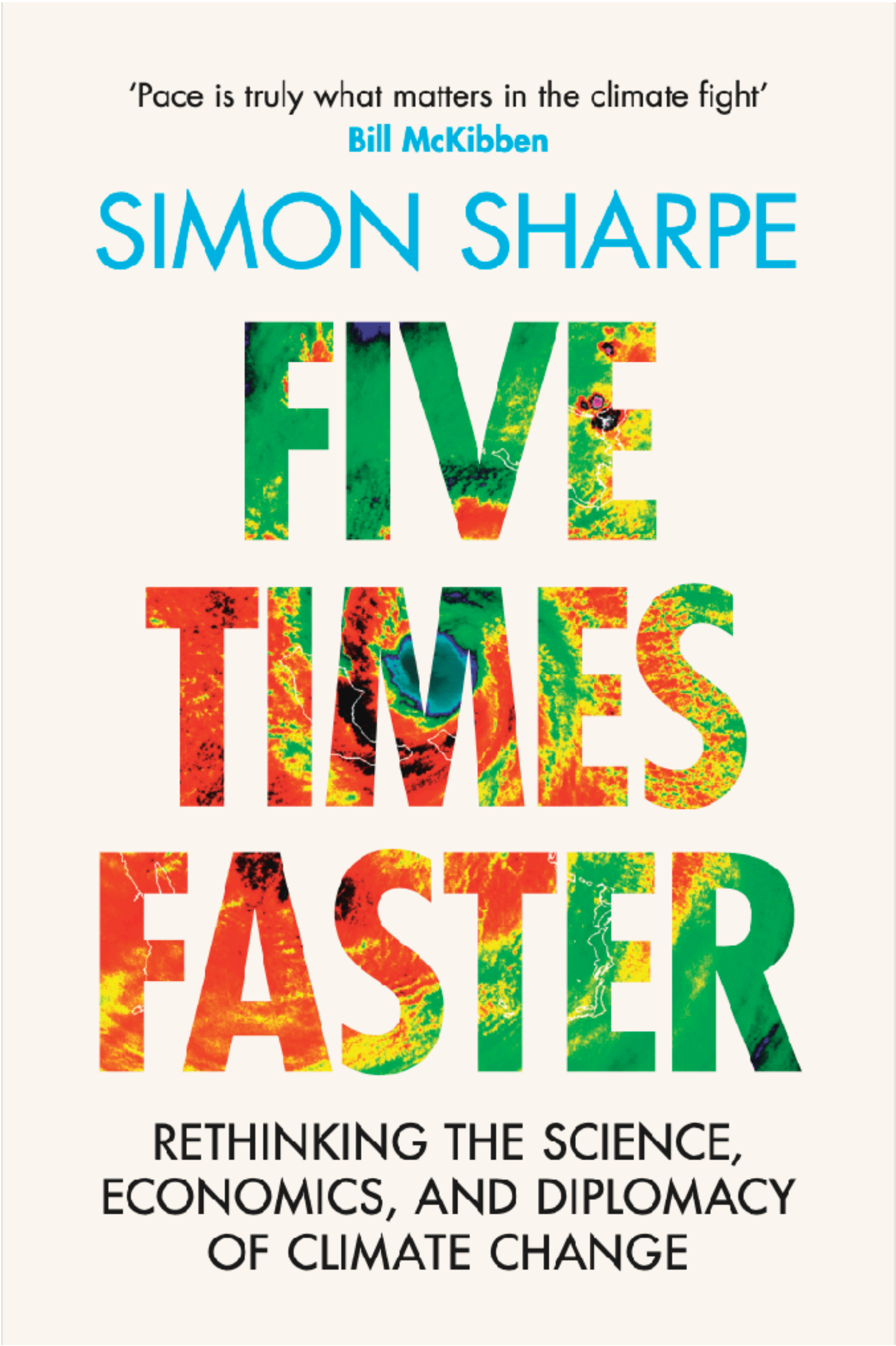
Resources

https://www.carbonbrief.org/









"Still, our appreciation of the risks of climate change is limited by the way our academic institutions encourage each researcher to focus on their own narrow area of expertise."

"Any actor should understand their points of leverage[...] We each have to understand the opportunities presented by our place in the system and do our best to exploit them."

Thanks

@Cambridge_ICCS

https://iccs.cam.ac.uk