On the impact of quantum computing technology on future developments in high-performance scientific computing

Matthias Möller, Numerical Analysis,
Delft Institute of Applied Mathematics, TU Delft

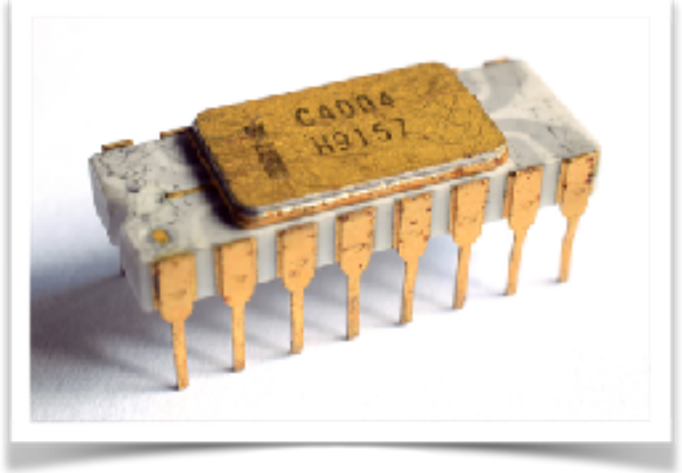
M. Möller and C. Vuik
Ethics and Information Technology
December 2017, Volume 19, Issue 4, pp. 253-269
DOI: https://link.springer.com/article/10.1007/s10676-017-9438-0

Outline

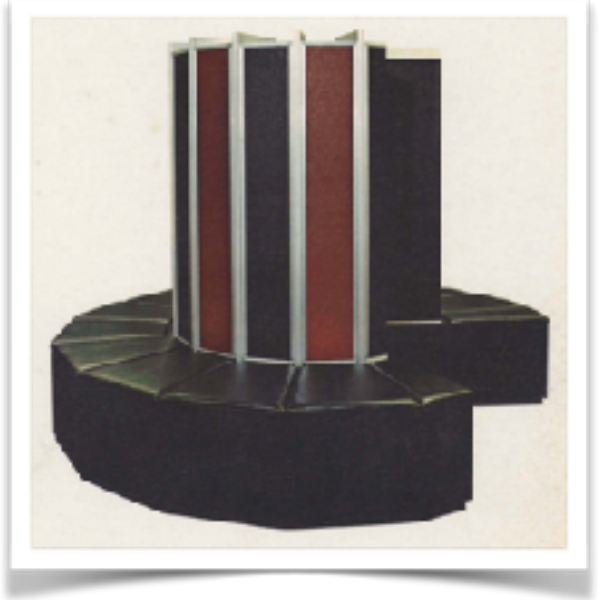
- High-performance scientific computing
  - Past, present and possible future trends
  - FEM making friends with exotic hardware

- Q-accelerated scientific computing
  - Potential use case examples
  - Challenges and open problems

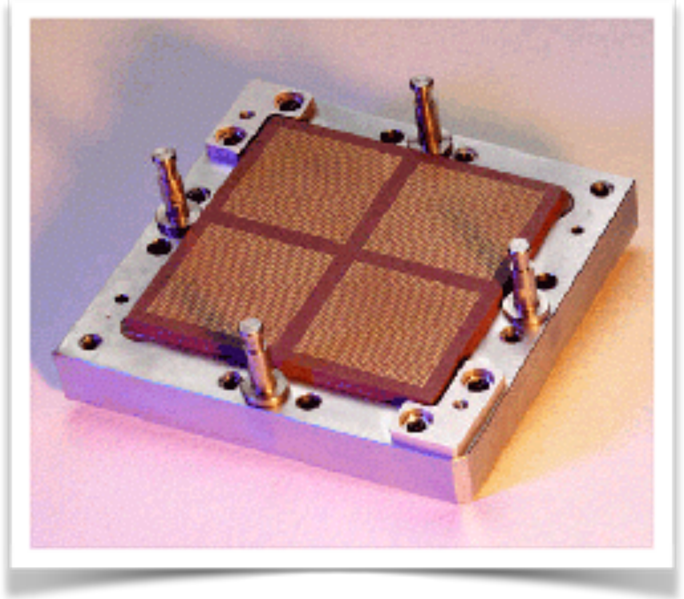
History of computing



1971: Intel 4004 first
commercial microprocessor



1976: Cray-1
first vector processor

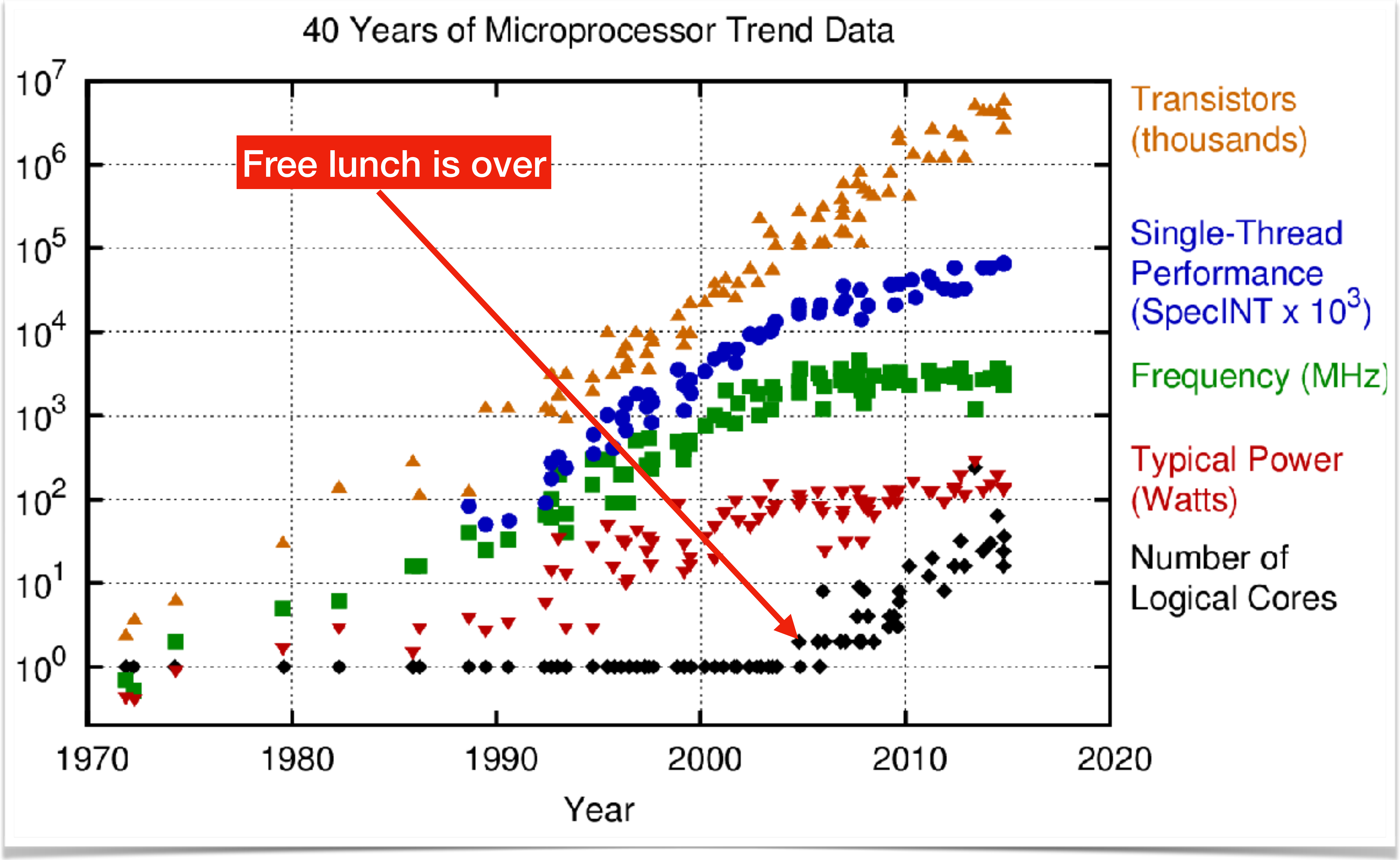


2001: IBM Power4
first multi-core CPU

2017: Intel Xeon Platinum
28 cores and AVX-512

- Redesign of algorithms is mandatory to exploit compute capabilities of modern hardware (multi-threading, vectorisation)
- “Algorithm follows hardware”

History of computing



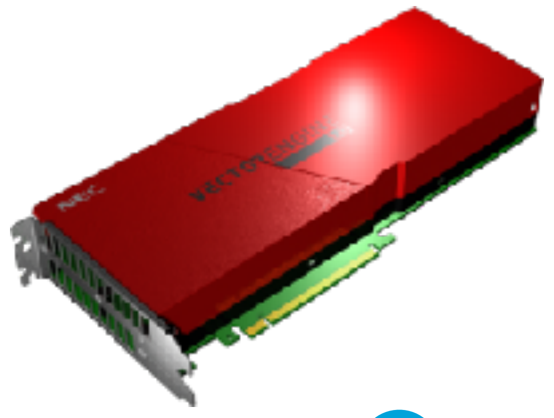
Credit: K. Rupp, 2015

History of accelerated-computing

1999: GeForce 256
“the world’s first GPU”

2006: GeForce G80
first CUDA-capable GPU

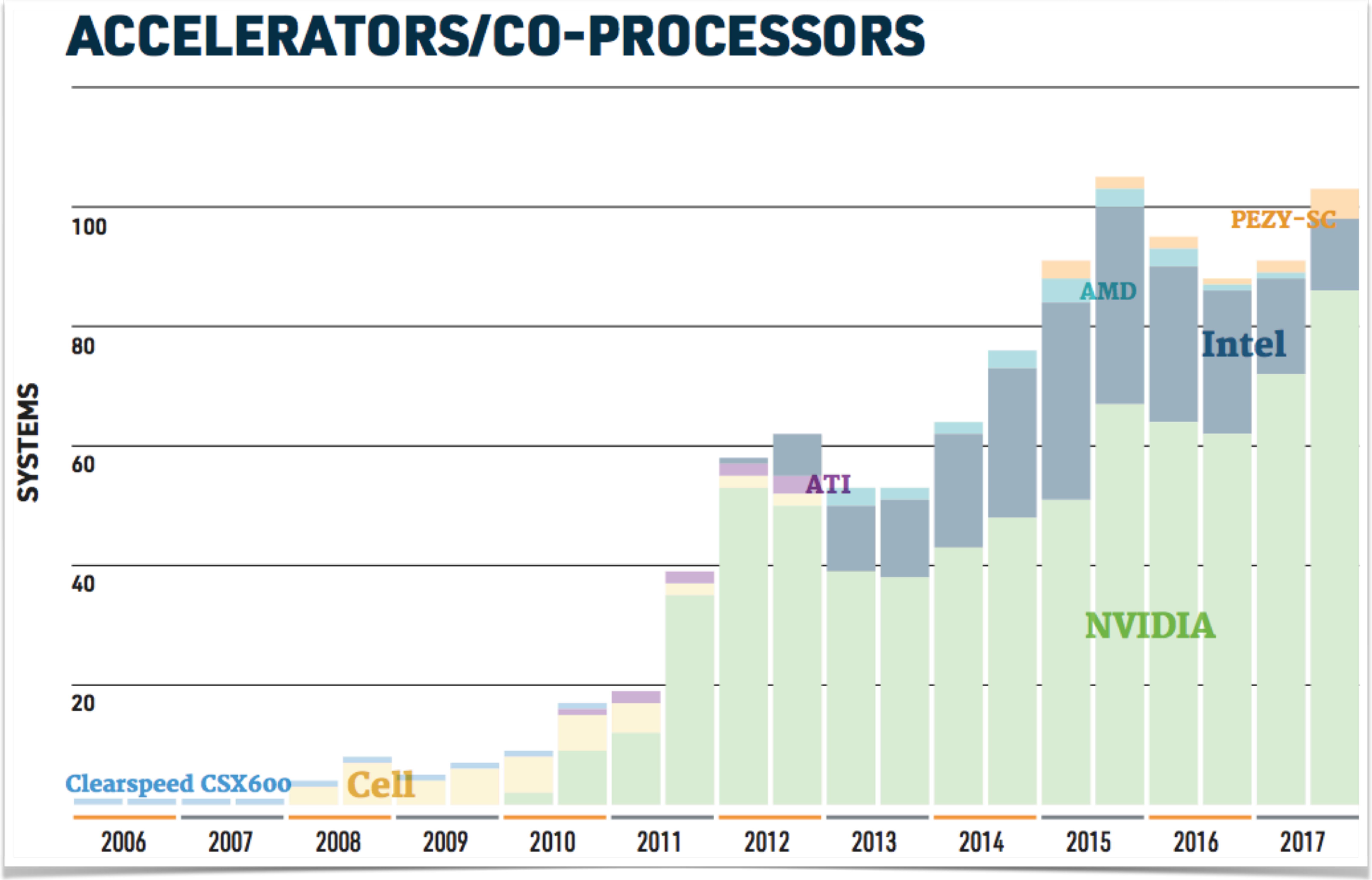
2013: Intel Xeon Phi



2017: NEC
SX-Aurora

2017: Tesla V100

History of accelerated-computing



Credit: TOP500, Nov 2017

History of accelerated-computing

1999: GeForce 256
“the world’s first GPU”

2006: GeForce G80
first CUDA-capable GPU

2012: Maxeler Technologies
FPGA-based DataFlow Engine
(Maximum Performance Comp.)

2013: Intel Xeon Phi

2016: Google TPUs (ML)

2017: NEC
SX-Aurora

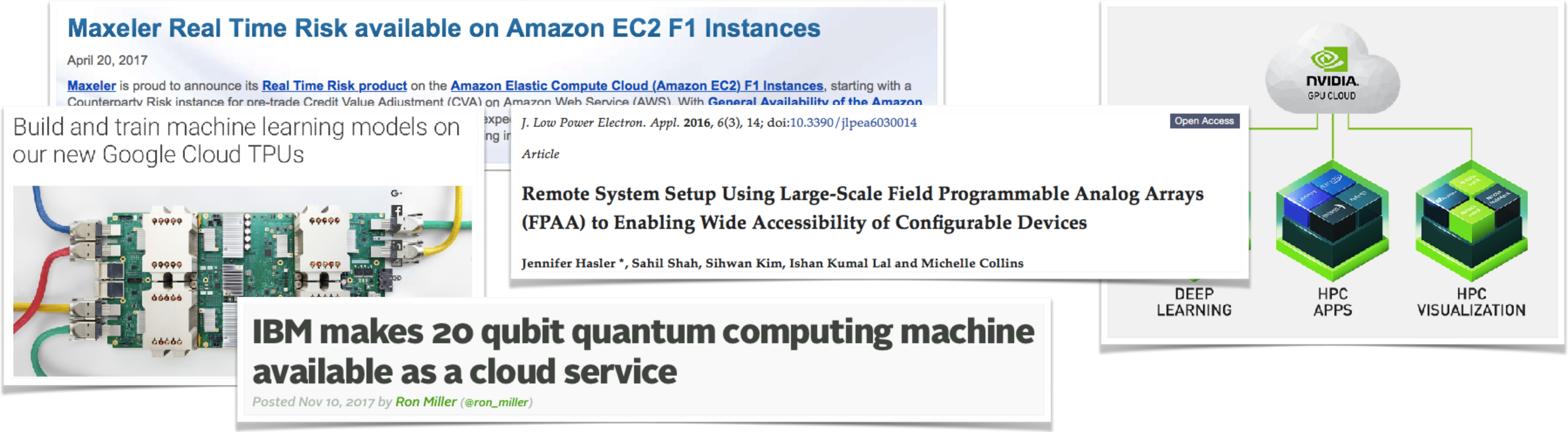
2017: Tesla V100
Tensor Cores (ML)

20??: HIVE

- Accelerated-computing is mainstream but still requires redesign of algorithms
- Application-specific hardware designs are becoming more popular paving the way for special-purpose functional acceleration
- “Hardware follows application”

Future trends in scientific computing

- Heterogeneous (most probably cloud-based) HPC clusters
  - multi-core/socket CPU-based host nodes
  - general-purpose many-core accelerators (GPUs, MICs, VPs)
  - special-purpose functional accelerators (TPUs, FP{G,A}As, QC?)



Future trends in scientific computing

- Expected paradigm shifts (“*since the free lunch is over*”)
  - closer co-design of compute hardware and applications
  - more fine-grained application-specific solution algorithms
  - new concepts like in-memory-computing, computing in space
  - rediscovery of ‘old’ approaches
    - *mixed-precision:* Wilkinson ’63, Strzodka et al. ’08, NVIDIA ’16
    - 

The Finite Element Method

From textbook version to HPC implementation

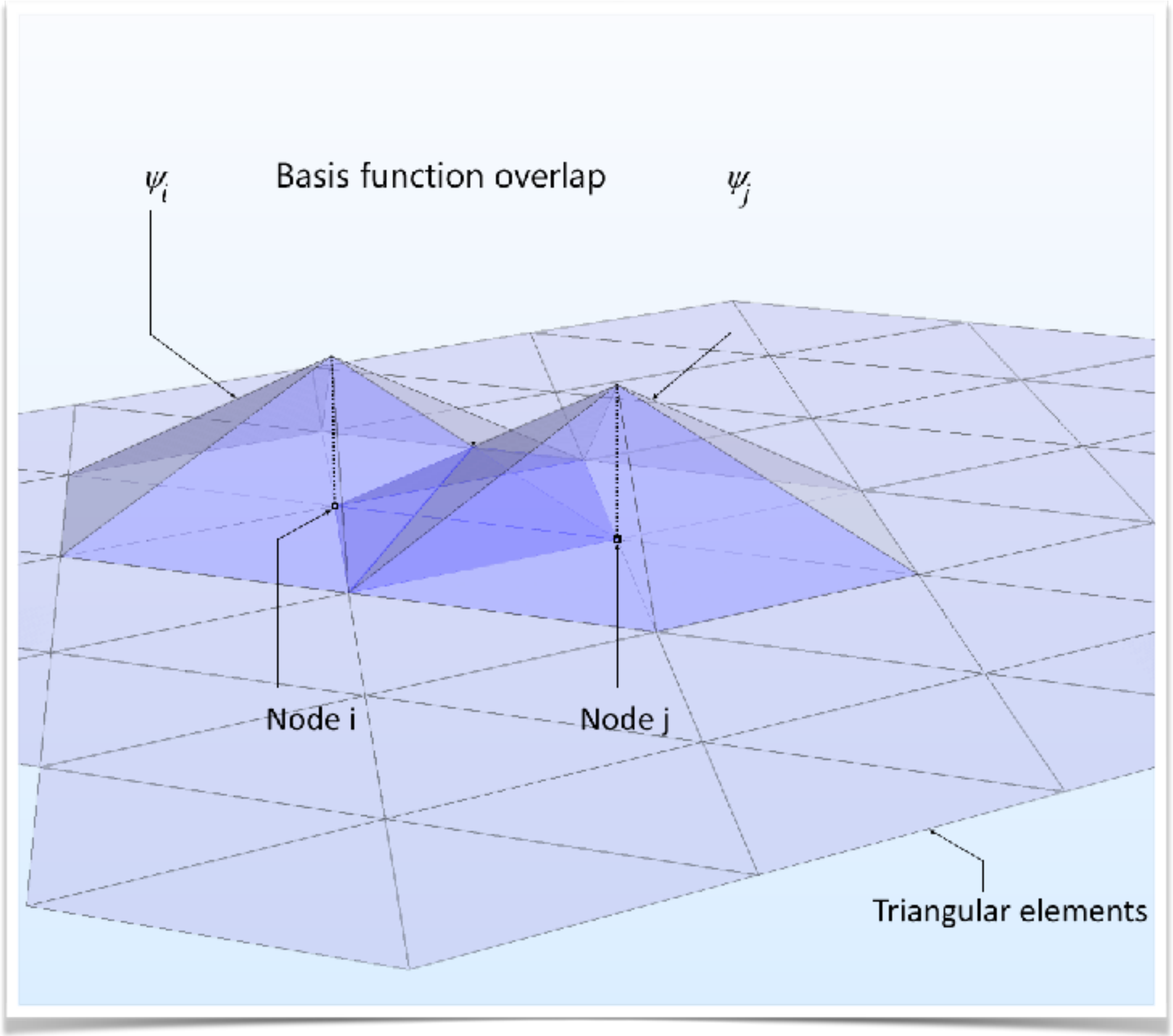
FEM in a nutshell

- Poisson eq: $-\boldsymbol{u}_{xx}-\boldsymbol{u}_{yy}=\boldsymbol{f}$ in Ω s.t. $\boldsymbol{u}=0$ on Γ
- Discretisation by the finite element method:
  - *Basis expansion of the solution*

$$\boldsymbol{u}_h(x,y)=\boldsymbol{u}_1 B_1(x,y)+\ldots+\boldsymbol{u}_n B_n(x,y)$$

Unknown coefficients

User-definable basis functions



Credit: Comsol

FEM in a nutshell

- Poisson eq: $-\boldsymbol{u}_{xx}-\boldsymbol{u}_{yy}=\boldsymbol{f}$ in Ω s.t. $\boldsymbol{u}=0$ on Γ
- Discretisation by the finite element method:
  - *Basis expansion of the solution*

    $$\boldsymbol{u}_h(x,y)=\boldsymbol{u}_1 B_1(x,y)+\ldots+\boldsymbol{u}_n B_n(x,y)$$

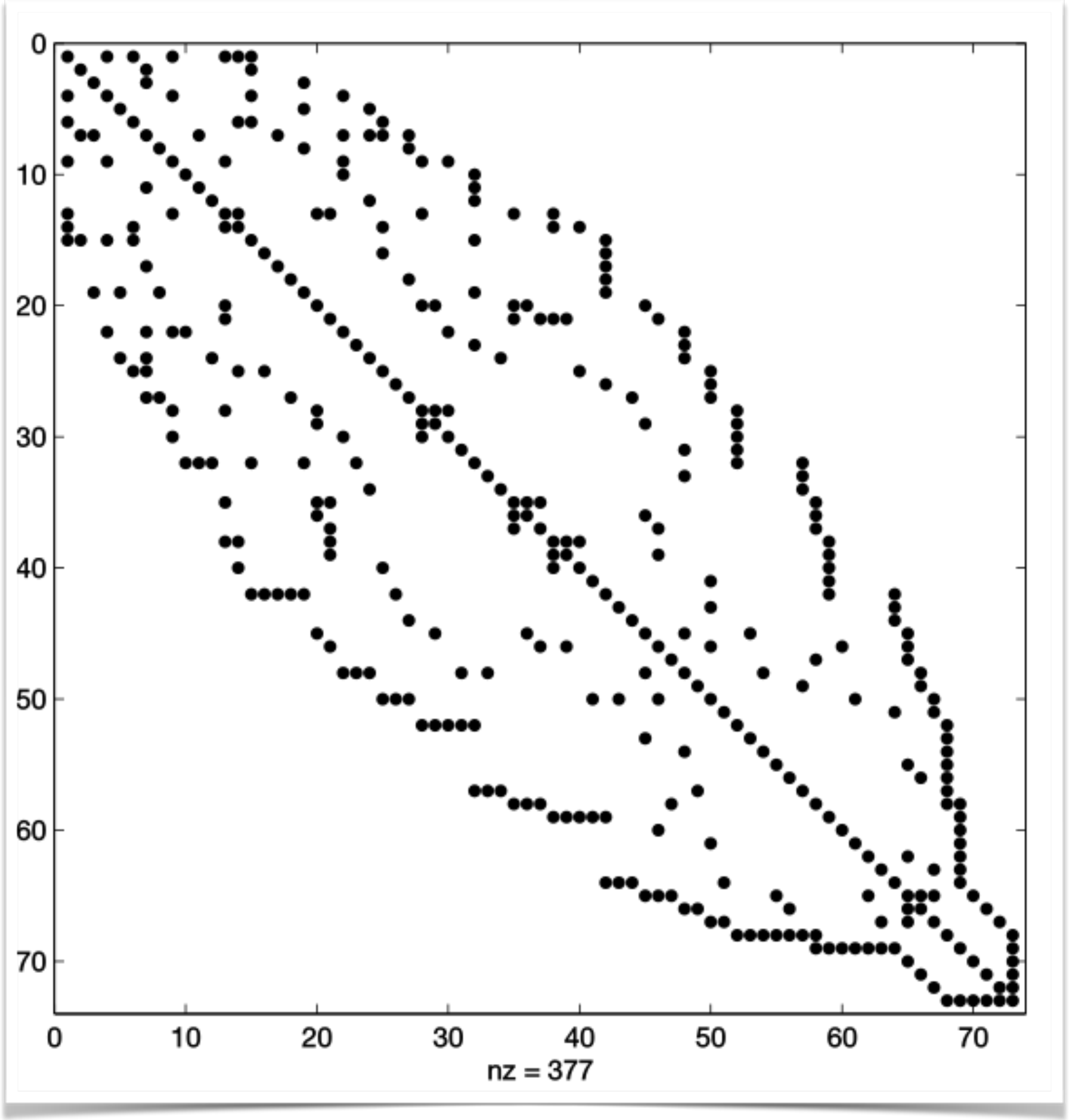
  - *Assembly of stiffness matrix* $\boldsymbol{A}_h$

    $$\boldsymbol{a}_{ij}=\int_\Omega B_{i,x}B_{j,x}+B_{i,y}B_{j,y}\,d\Omega$$

    and right-hand side vector $\boldsymbol{f}_h$

    $$\boldsymbol{f}_i=\int_\Omega B_i\,\boldsymbol{f}(x,y)\,d\Omega$$

Q-accelerated integration (Heinrich 2002)?



Credit: Wikipedia

FEM in a nutshell

- Poisson eq: $-\boldsymbol{u}_{xx}-\boldsymbol{u}_{yy}=\boldsymbol{f}$ in Ω s.t. $\boldsymbol{u}=0$ on Γ
- Discretisation by the finite element method:
  - *Basis expansion of the solution*

    $$\boldsymbol{u}_h(x,y)=\boldsymbol{u}_1B_1(x,y)+\ldots+\boldsymbol{u}_nB_n(x,y)$$

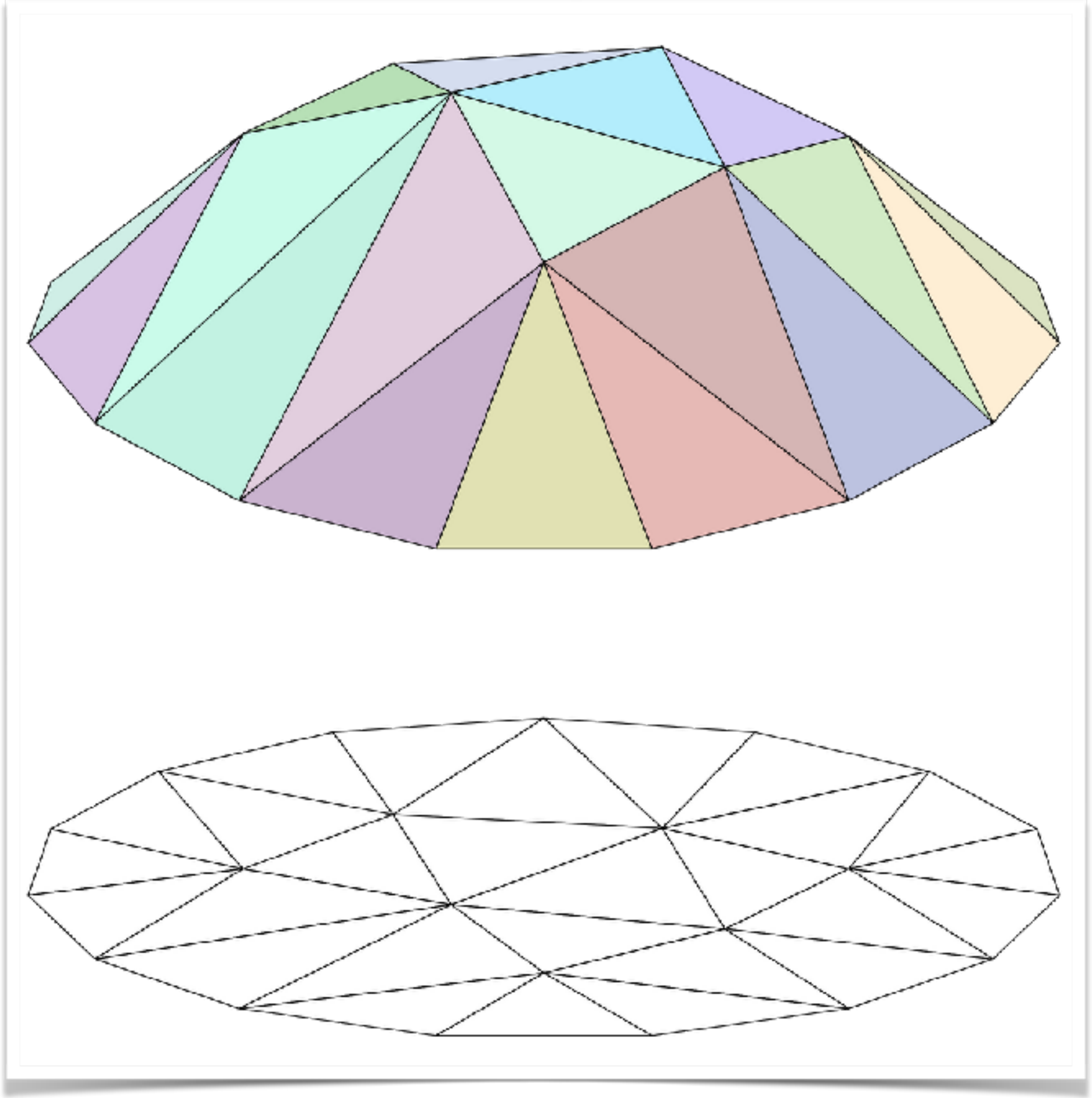
  - *Assembly of stiffness matrix* $\boldsymbol{A}_h$

    $$\boldsymbol{a}_{ij}=\int_\Omega B_{i,x}B_{j,x}+B_{i,y}B_{j,y}\,d\Omega$$

    and right-hand side vector $\boldsymbol{f}_h$

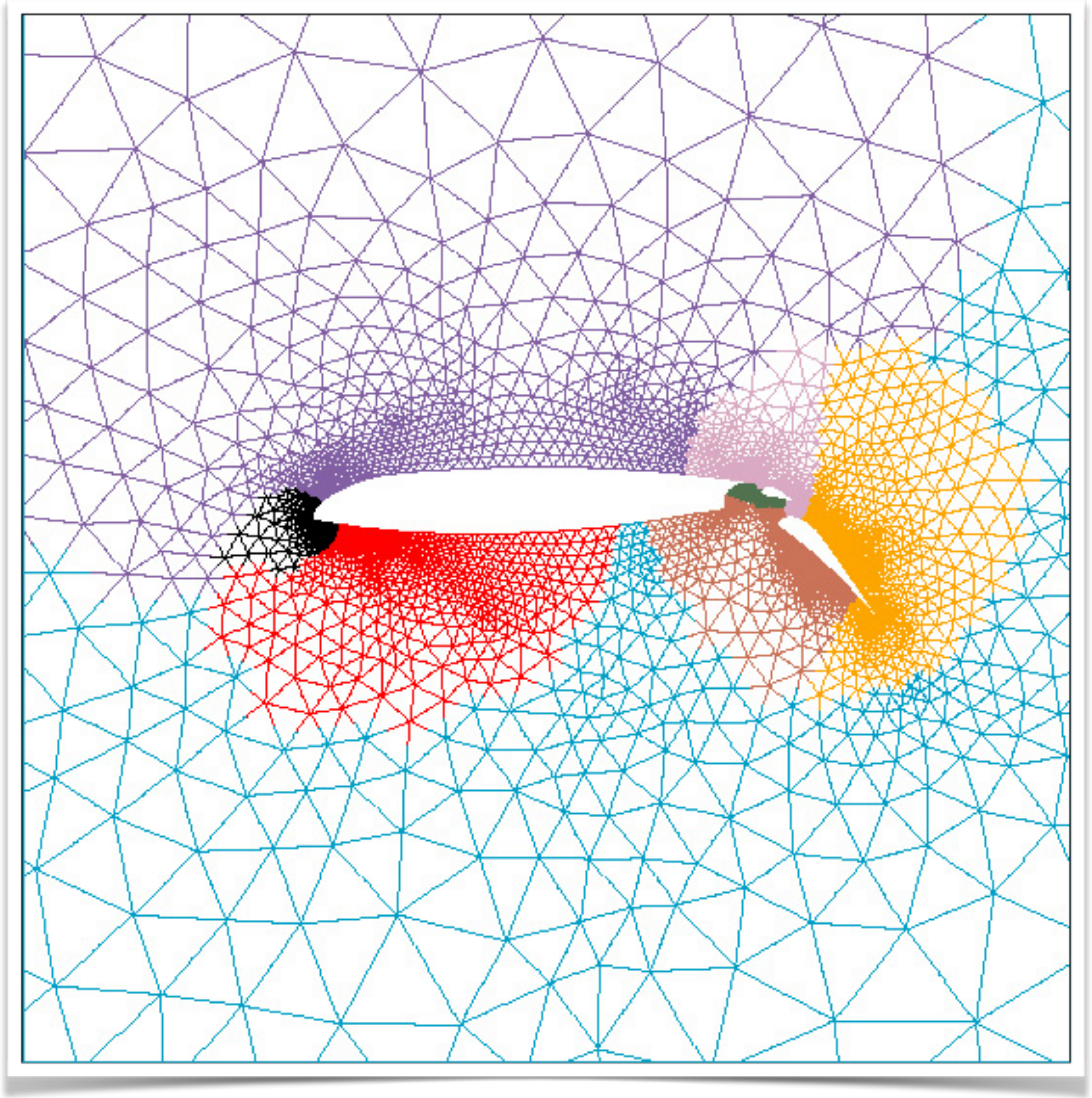
    $$\boldsymbol{f}_i=\int_\Omega B_i\,\boldsymbol{f}(x,y)\,d\Omega$$

  - *Solution of linear system* $\boldsymbol{A}_h\boldsymbol{u}_h=\boldsymbol{f}_h$

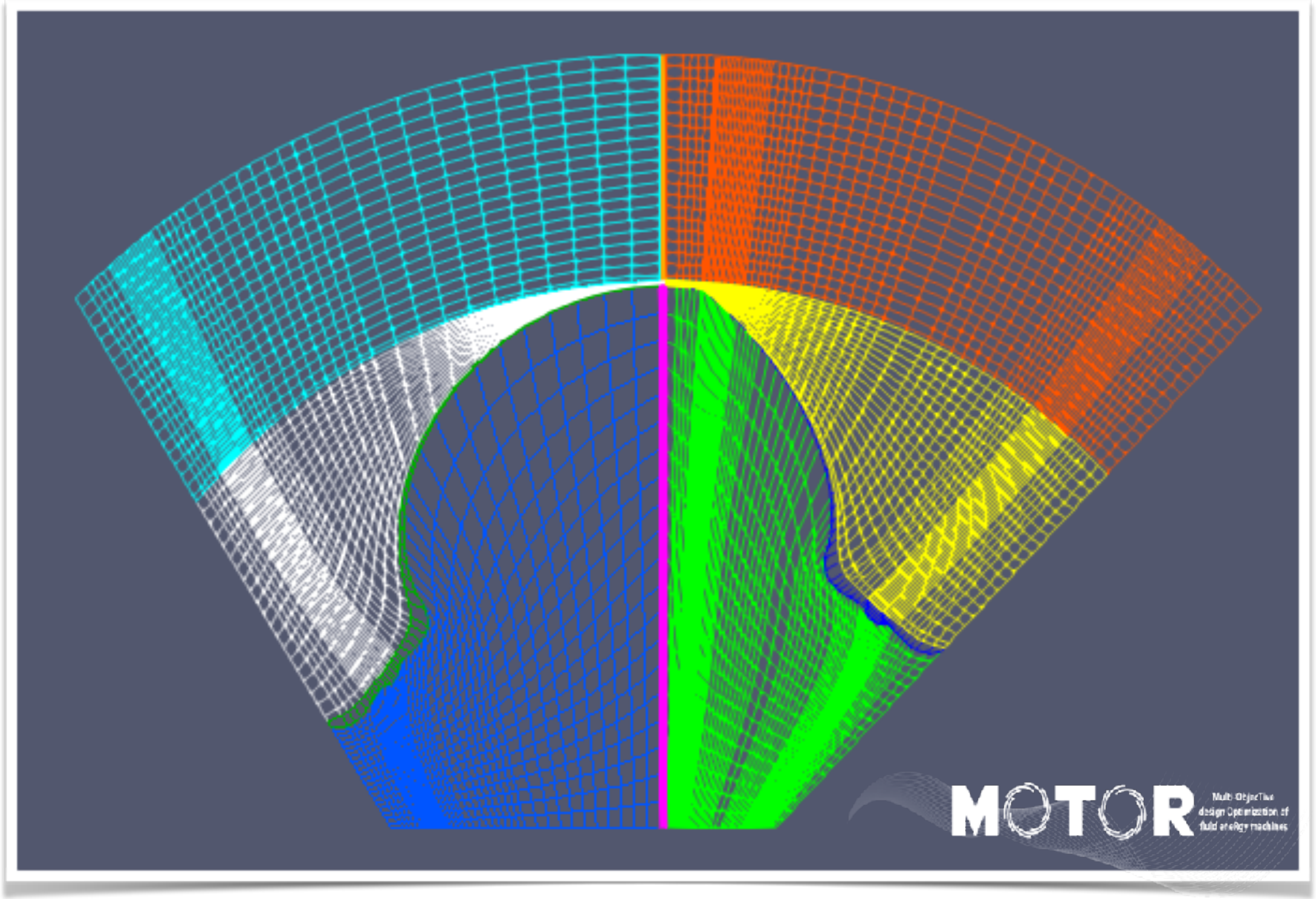


Credit: Wikipedia

Beyond textbook FEM



HPC is a challenge

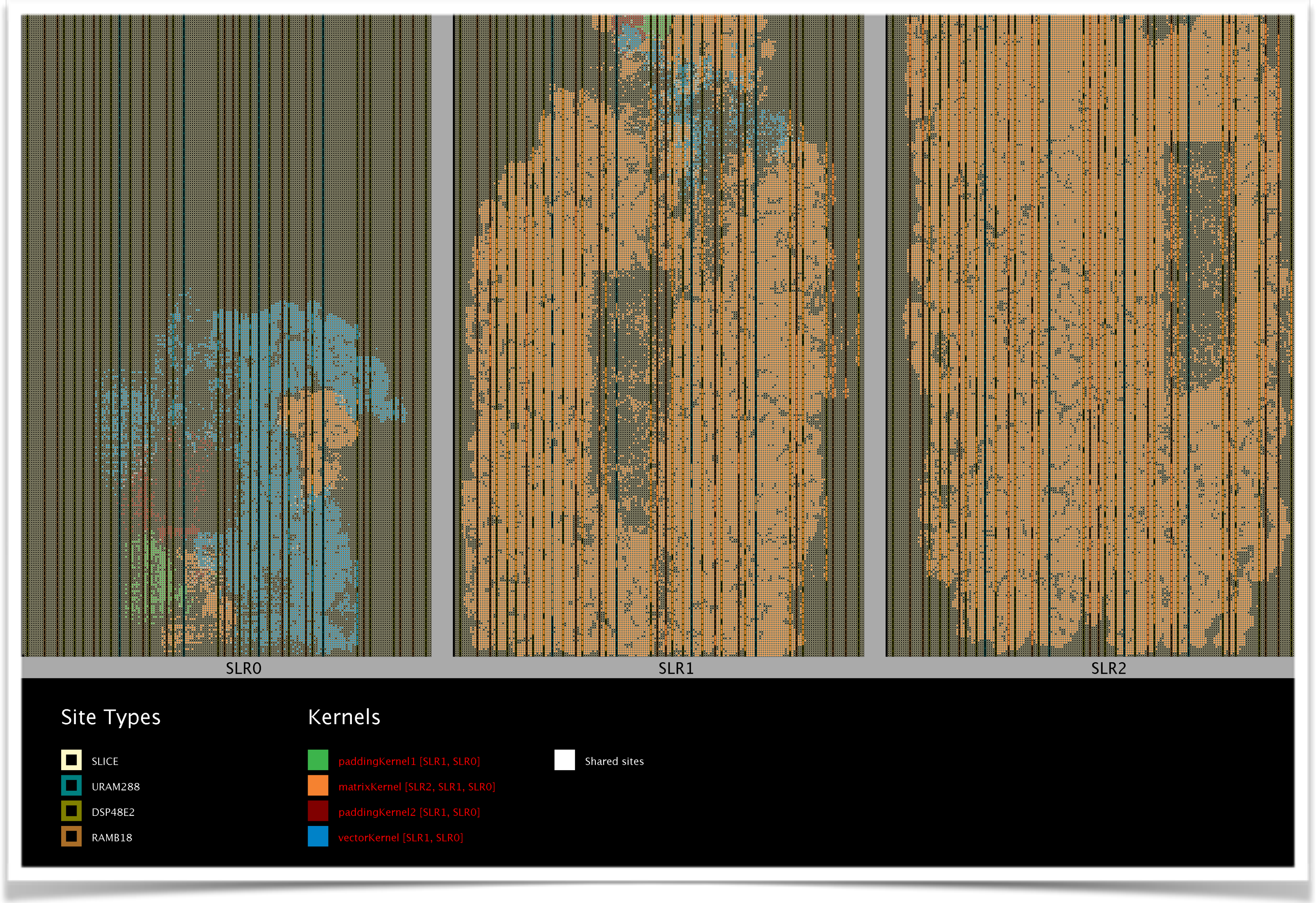


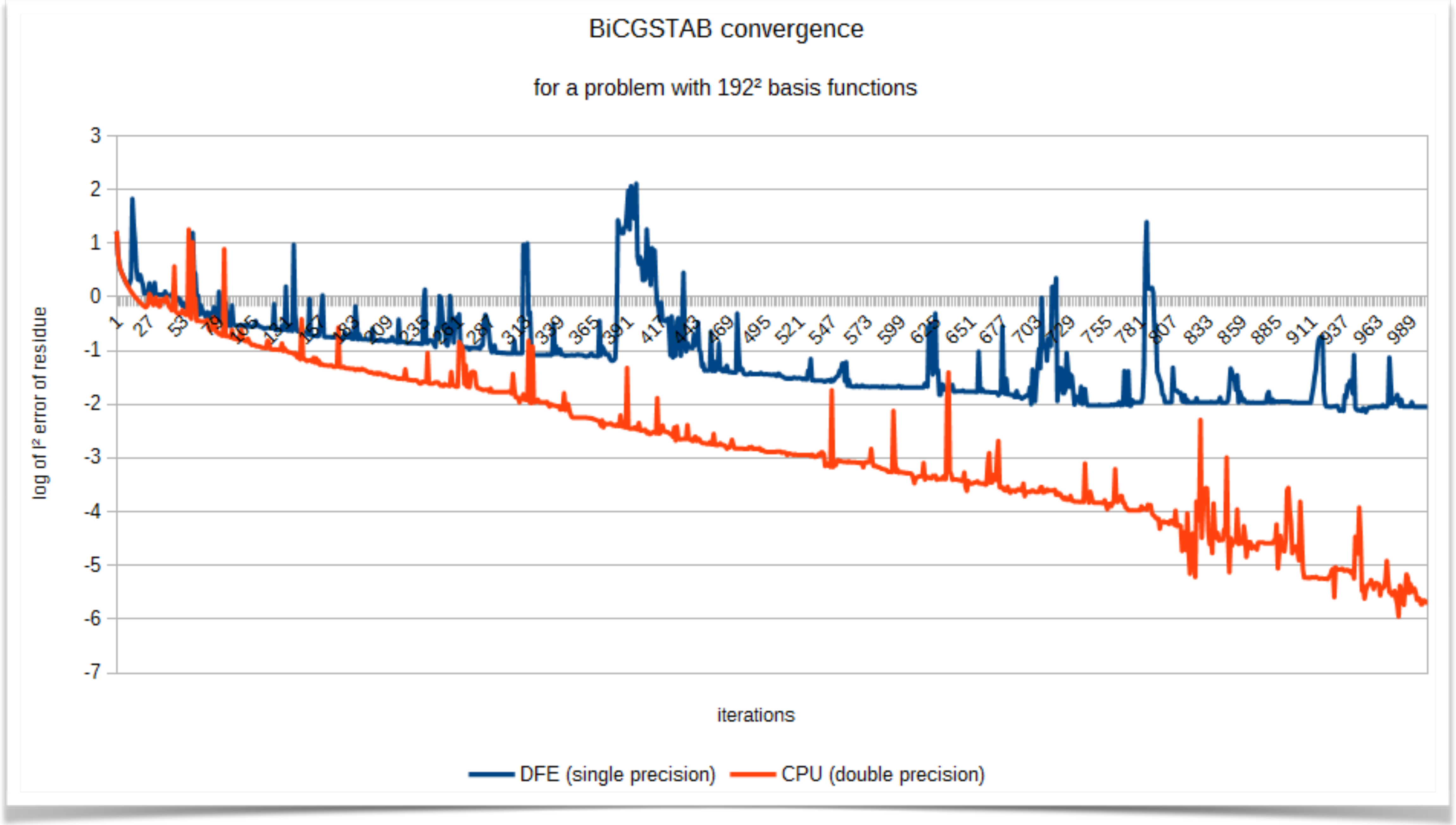
HPC-suitable by design

Credit: J. Hinz, TU Delft

An exotic FEM solver

- Matrix-free solver for quadratic B-Spline basis functions; on-the-fly calculation of matrix coefficients
- MAX4/MAX5 implementations





50 s on CPU @2.0GHz (DP)
10 s on MAX4 @20MHz (SP) <- 5x faster

25-30x faster (projected) on MAX5 with 48bit custom data format for reals

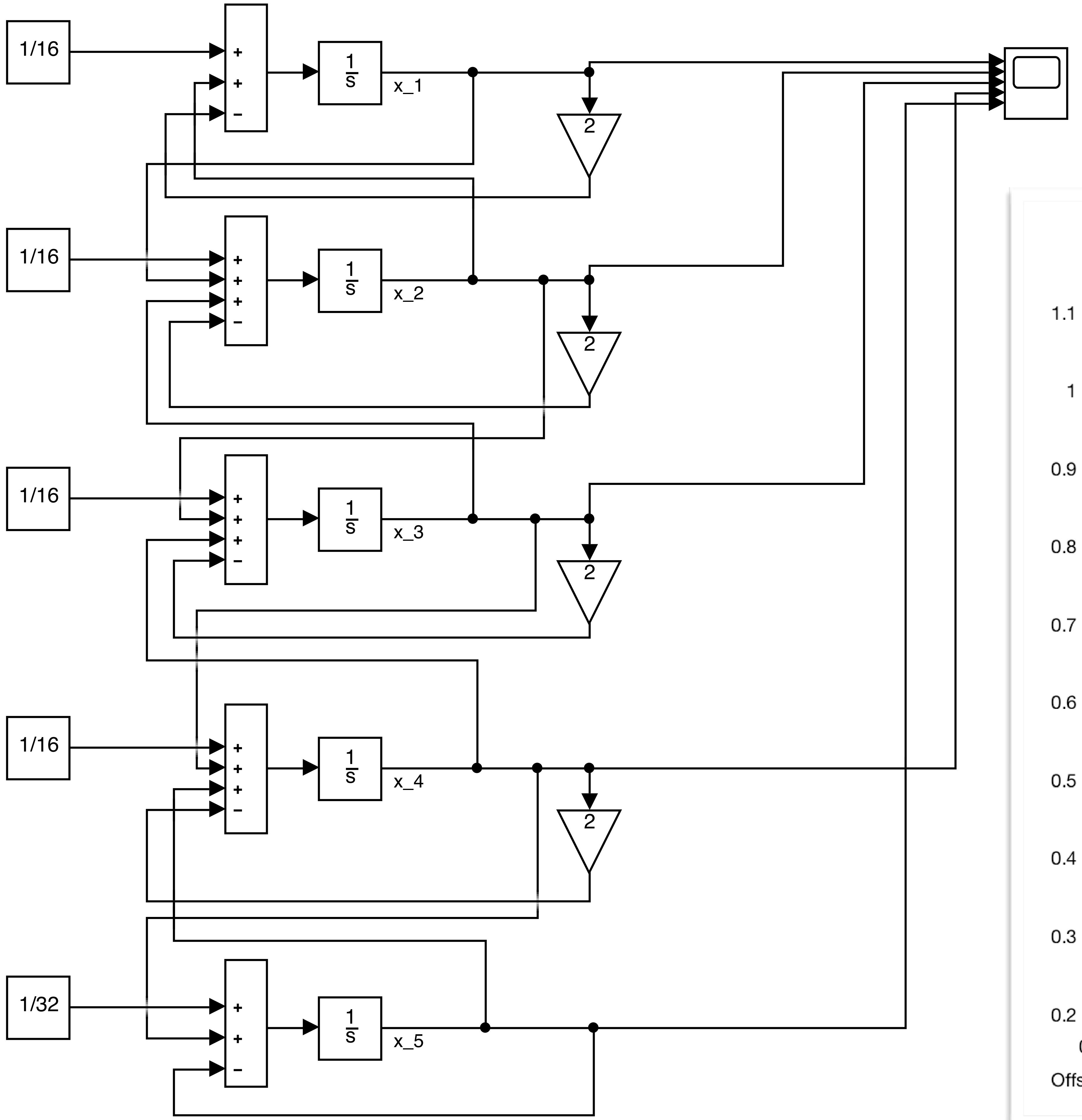
Credit: R. v. Nieuwpoort, TU Delft

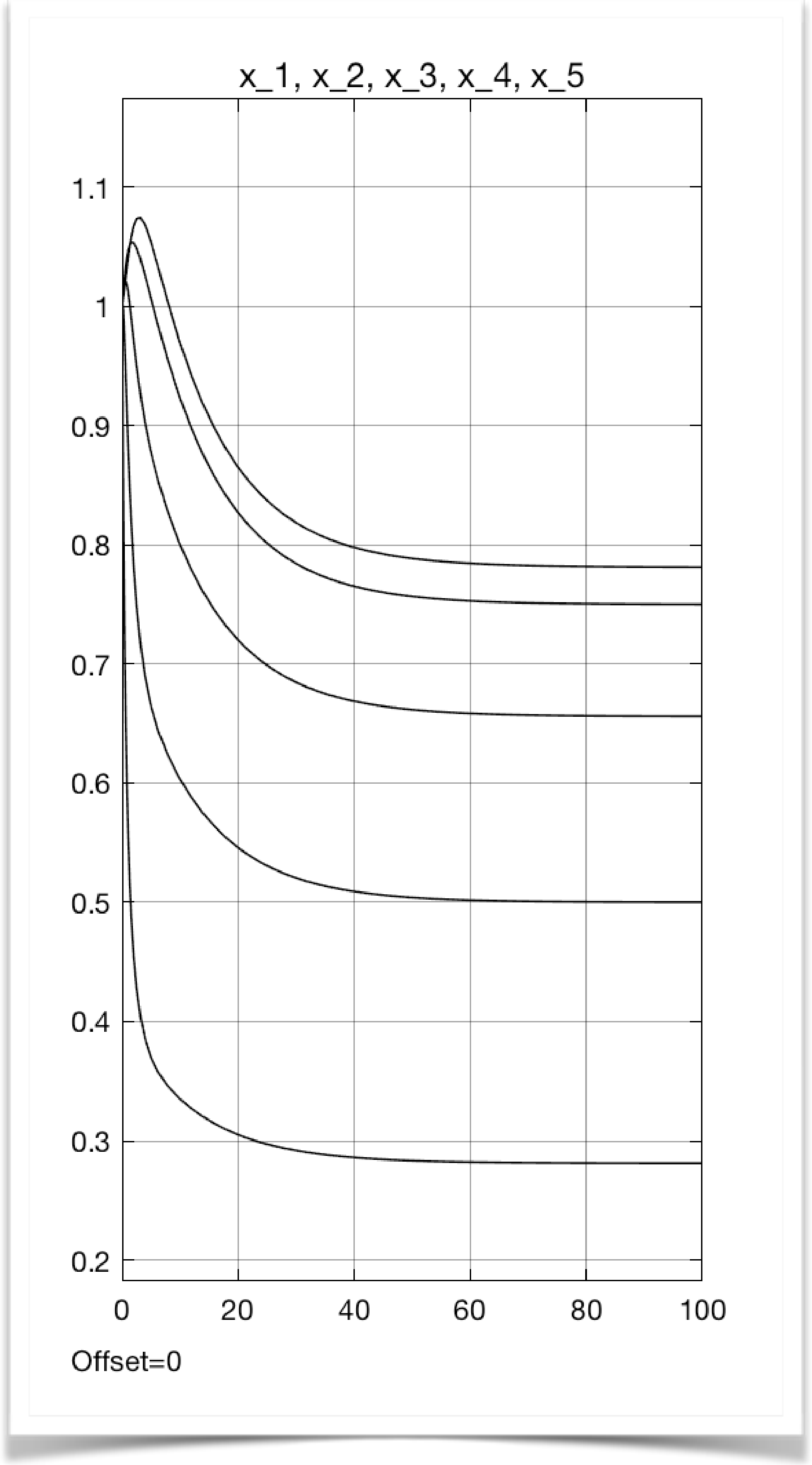
Another exotic FEM solver

- Solution to the linear system $\boldsymbol{A}_h \boldsymbol{u}_h = \boldsymbol{f}_h$ can be interpreted as the steady-state limit of the initial value problem

$$d\boldsymbol{u}_h(t)/dt = \boldsymbol{f}_h - \boldsymbol{A}_h \boldsymbol{u}_h(t),$$
$$\boldsymbol{u}_h(0) = \boldsymbol{u}_0$$

- Acceleration potential using (virtual) analog computing





Another exotic FEM solver

- Solution to the linear system $\boldsymbol{A}_h\boldsymbol{u}_h=\boldsymbol{f}_h$ can be interpreted as the steady-state limit of the initial value problem

$$d\boldsymbol{u}_h(t)/dt = \boldsymbol{f}_h-\boldsymbol{A}_h\boldsymbol{u}_h(t),$$
$$\boldsymbol{u}_h(0) = \boldsymbol{u}_0$$

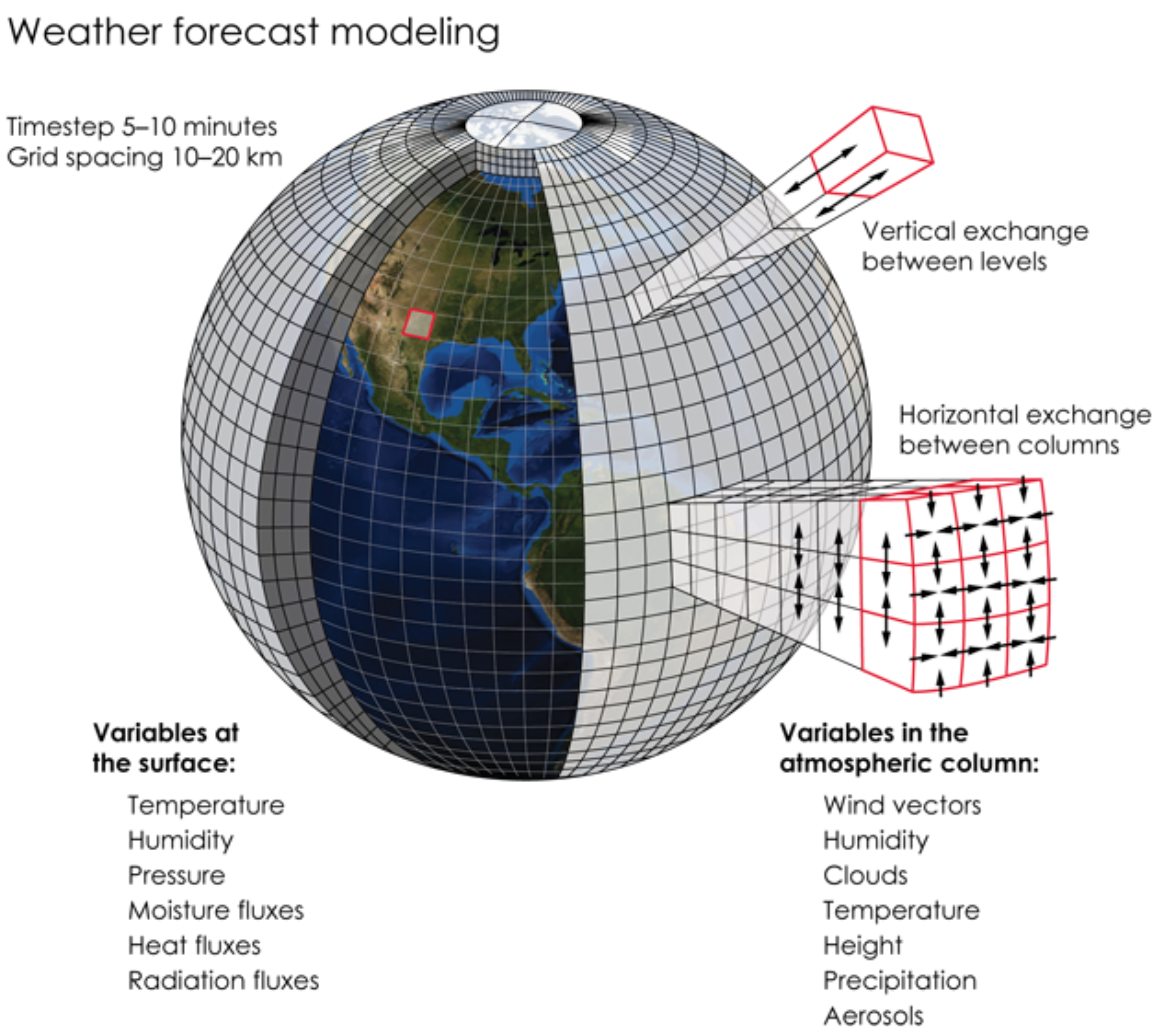
- Acceleration potential using (virtual) analog computing

- Next steps:
  - $(\boldsymbol{A}_h, \boldsymbol{f}_h)/s$—$ts$ value-time scaling to reduce the dynamic range
  - "Assembly" of $\boldsymbol{A}_h, \boldsymbol{f}_h$ via steady state continuous process
  - "Reduction" of $\boldsymbol{u}_h$ into scalar output quantity $J=\boldsymbol{u}_h^T\boldsymbol{M}\boldsymbol{u}_h$
  - Prototype FPGA-implementation
- Inspiration for Q-FEM solver(?)

Quantum Computers

The next step in accelerator technologies?

Simulation-based forecasting



- PDE-based mathematical model: *physical conservation laws*
- Discretisation in space:

  5 million grid points

  x 100 vertical levels

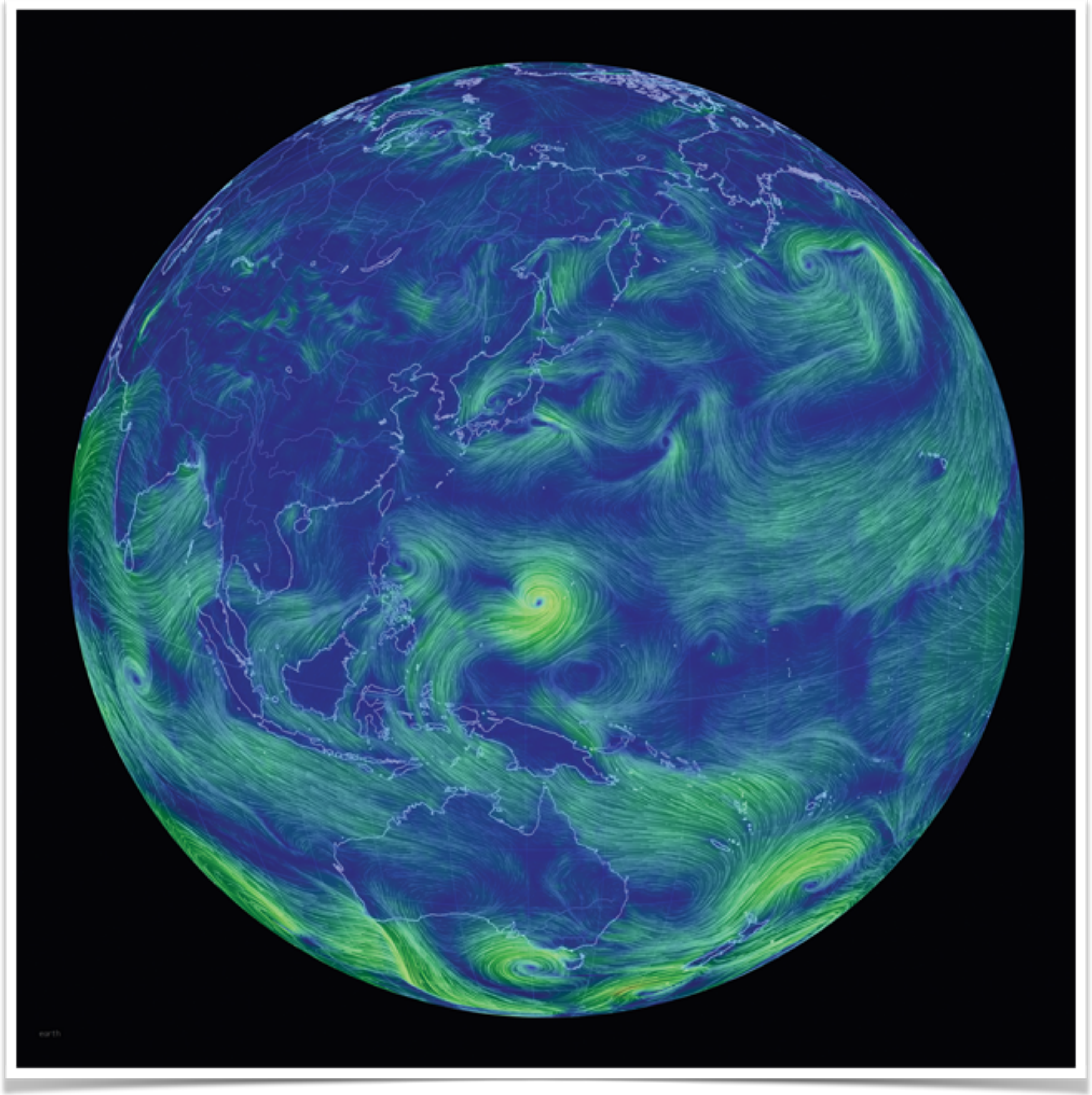
  x 10 prognostic variables

  = 5 billion unknowns in space
- Discretisation in time:

  864 time steps for 72h-forecast

Credit: K. Cantner, AGI

Simulation-based forecasting

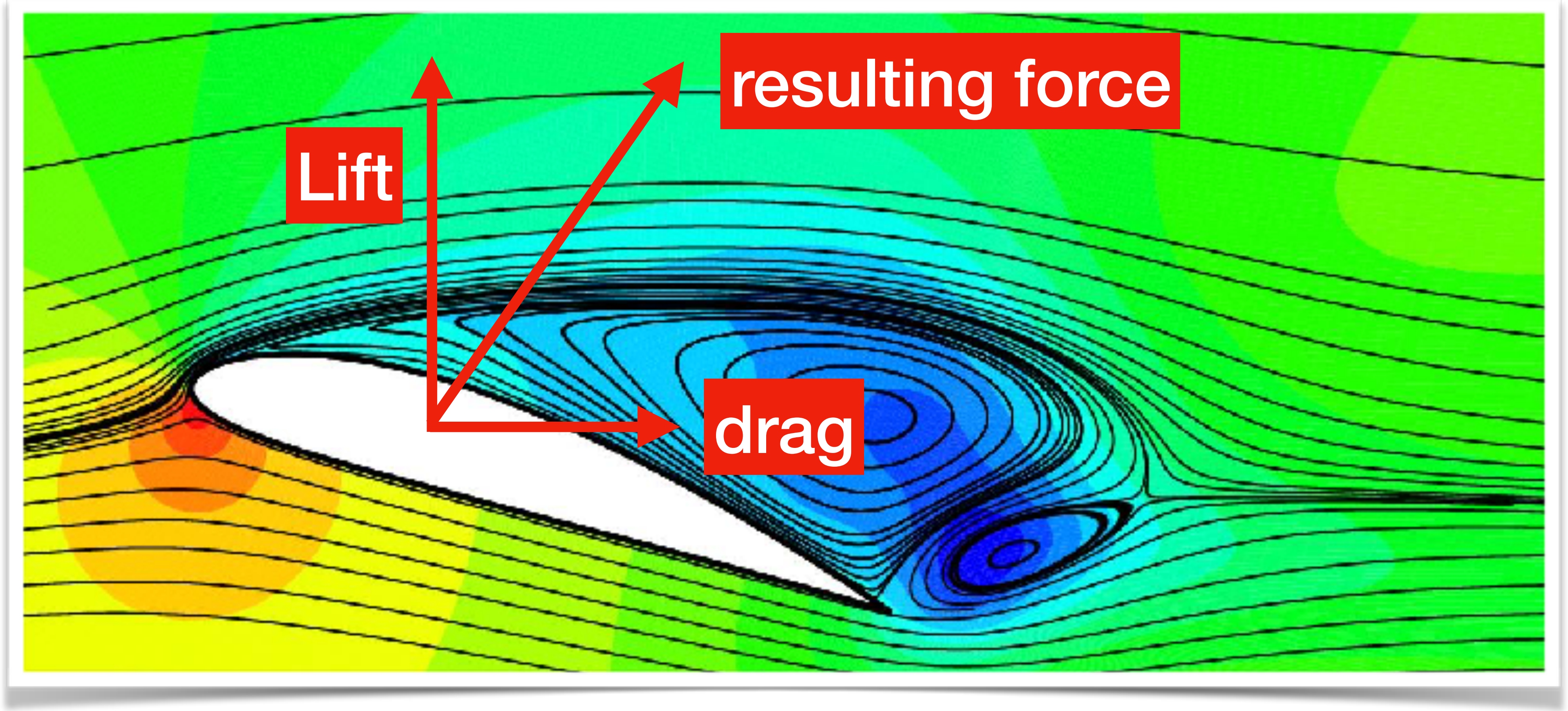


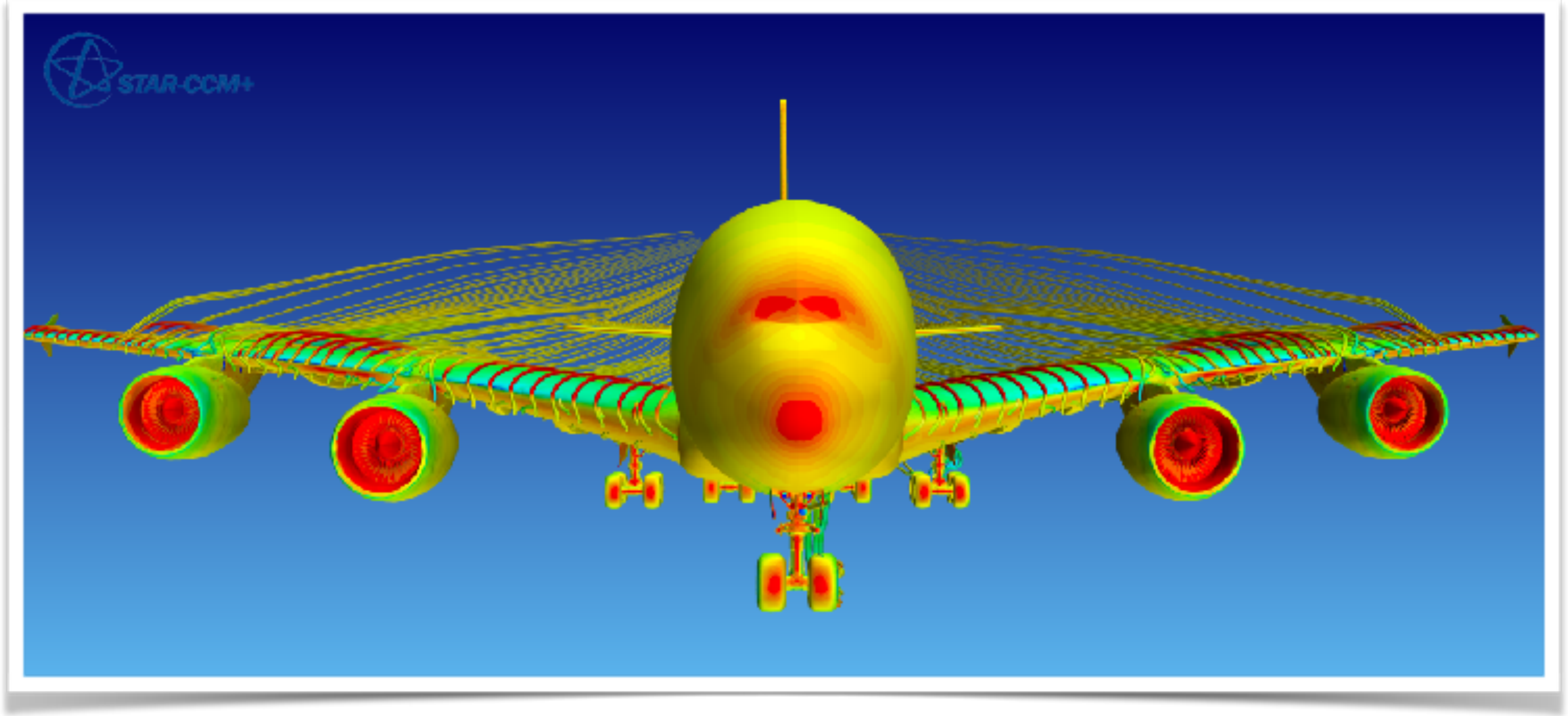
Credit: Cameron Beccario, https://earth.nullschool.net/

- Quantity of interest (QoI):
  Time evolution of field variables
- ECMWF at Reading, UK today:
  simulation of single 10-day forecast in one hour with 10,000 processors
- Future challenges:
  increase model resolution by factor 2,000 (1km, 200 levels, 100 variables) and time-step size to improve forecast accuracy requires 20 million processors!
- Suitable for Q-acceleration:
  no, due to large size of output data

Simulation-based engineering

- QoI: key performance functionals
  "find $J_1(U)$, $J_2(U)$,… such that U solves the discretised problem formulation"
- Example: *cruise L/D (higher is better)*
  B747-200 (1969): 15.3
  B777-200 (1994): 19.3
- Future challenges:
  increase model resolution and complexity; turbulence modelling; CAD integration; virtual twins for lifetime analysis
- Suitable for Q-acceleration:
  maybe, since it fits into QLSA setup but the problem sizes might be too large

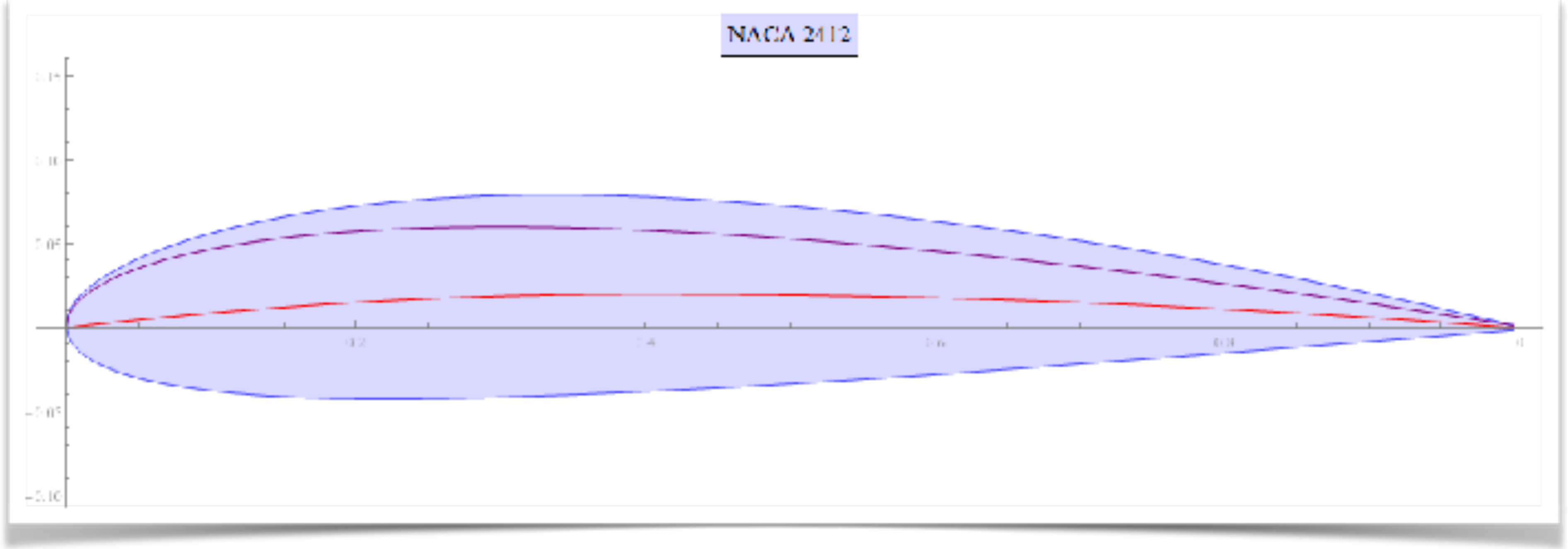


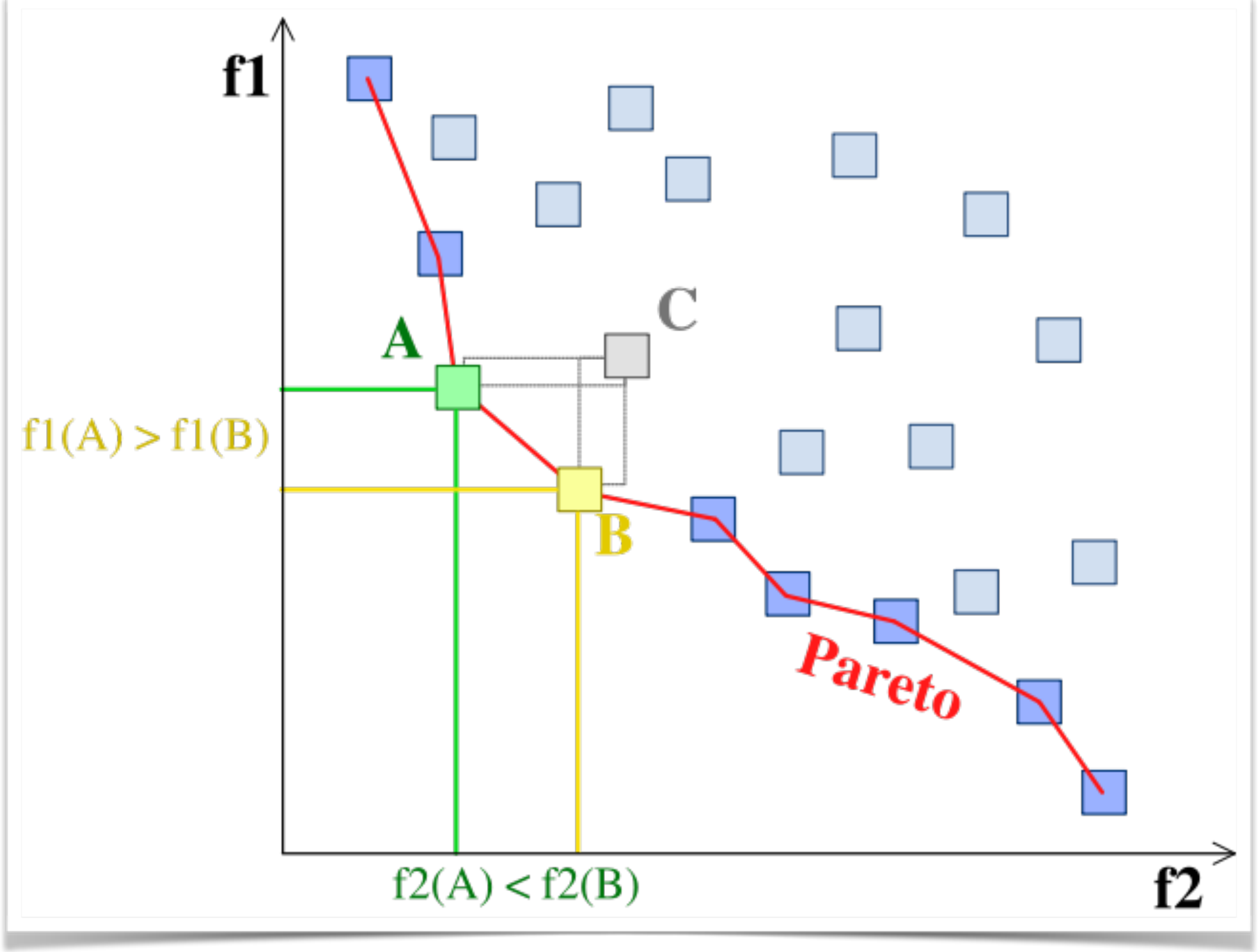


Credit: Siemens

Simulation-based optimisation

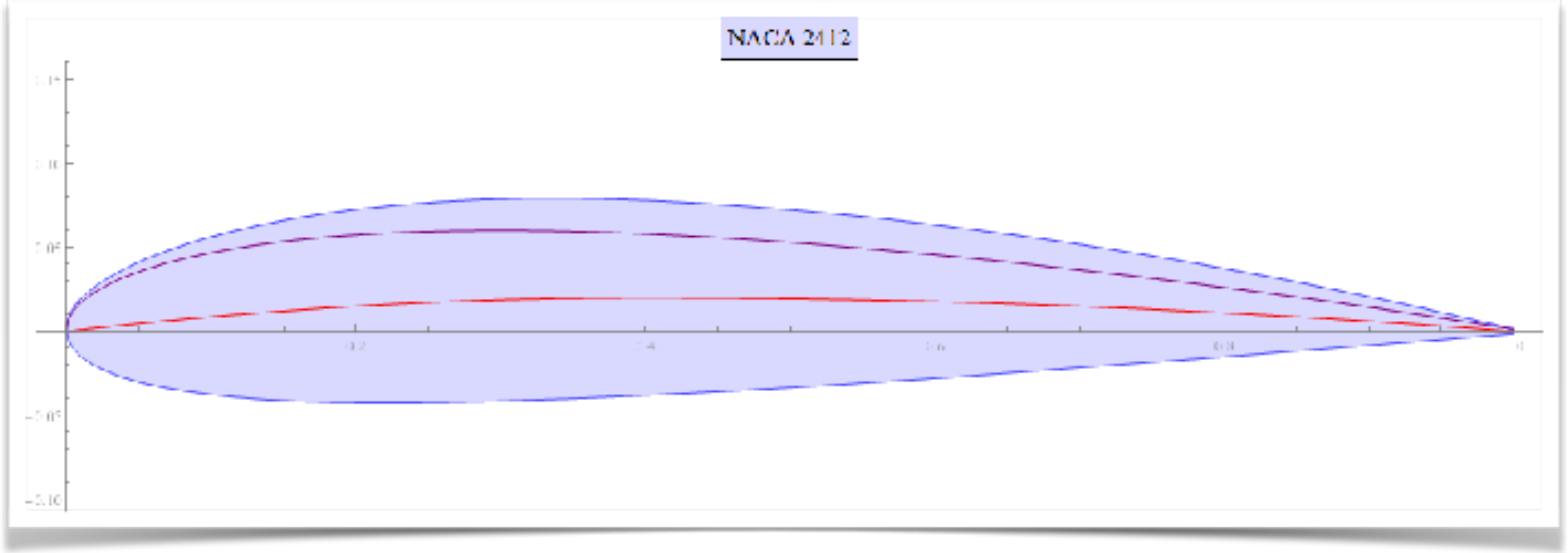
- Input: *design parameters p_1, p_2,… controlling the shape of the wing*
- QoI: *Pareto front of optimal design parameters* $\mathbf{p}^*$ *such that the solutions* $U(\mathbf{p}^*)$ *to the discretised PDE problems maximise ‘all’ key performance functionals f1, f2,…*
- Solution approaches: $\mathbf{p}^k \rightarrow \mathbf{p}^{k+1}$
    - *Gradient-free methods:* evolution-inspired algorithms for generating populations of parameters
    - *Gradient-based methods:* choose parameters ‘in the direction’ that brings $U(\mathbf{p}^{k+1})$ closer to an optimal state

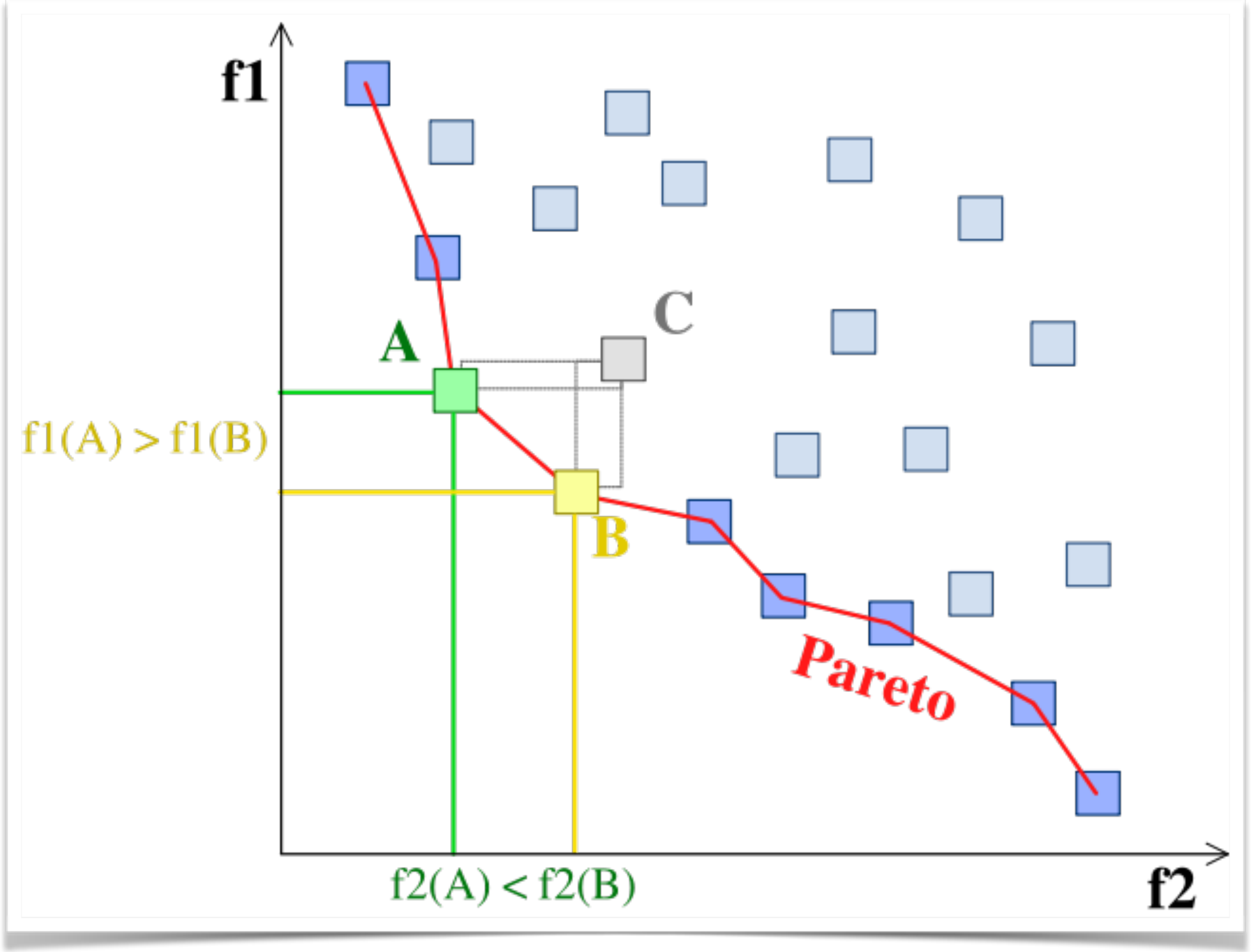




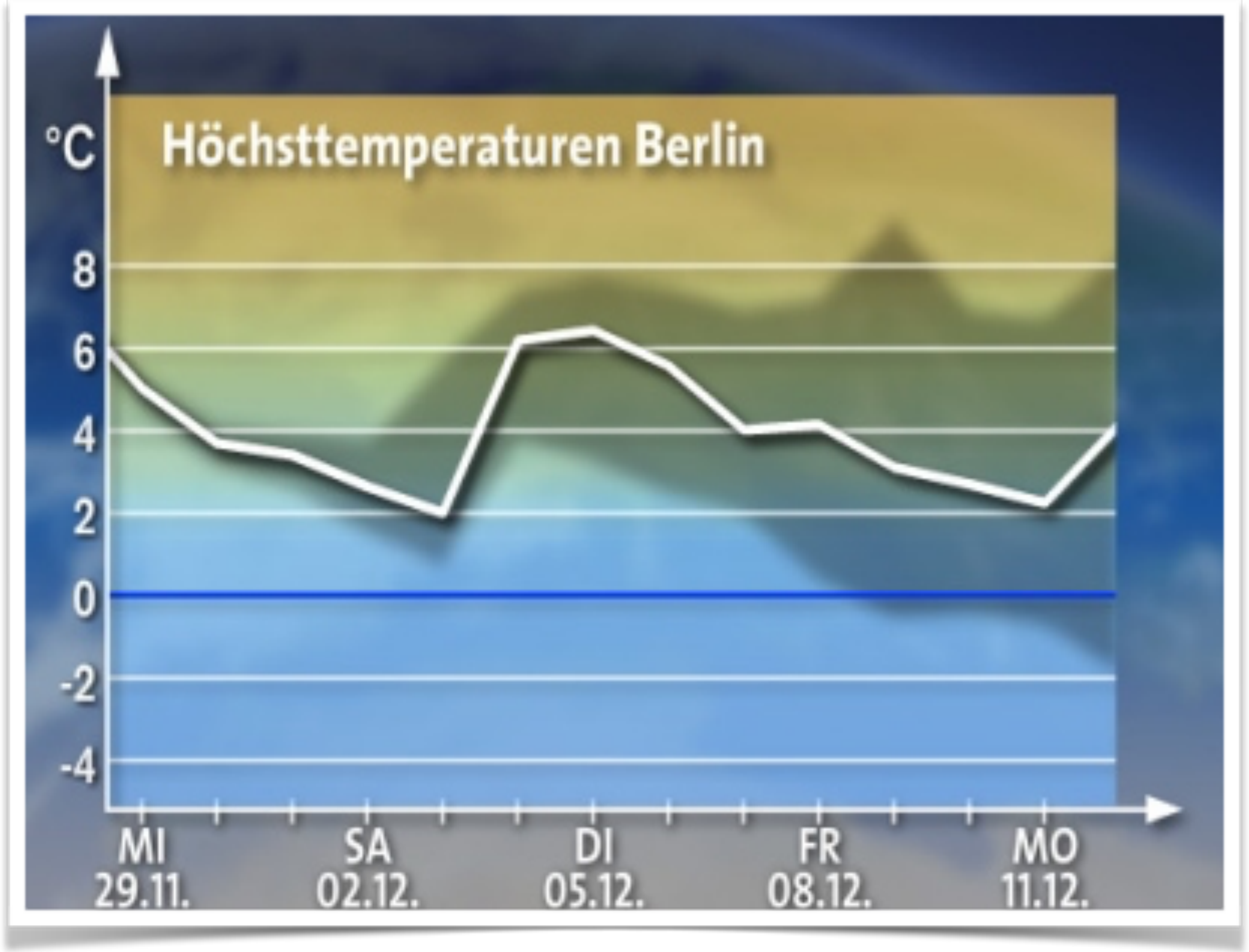
Simulation-based optimisation

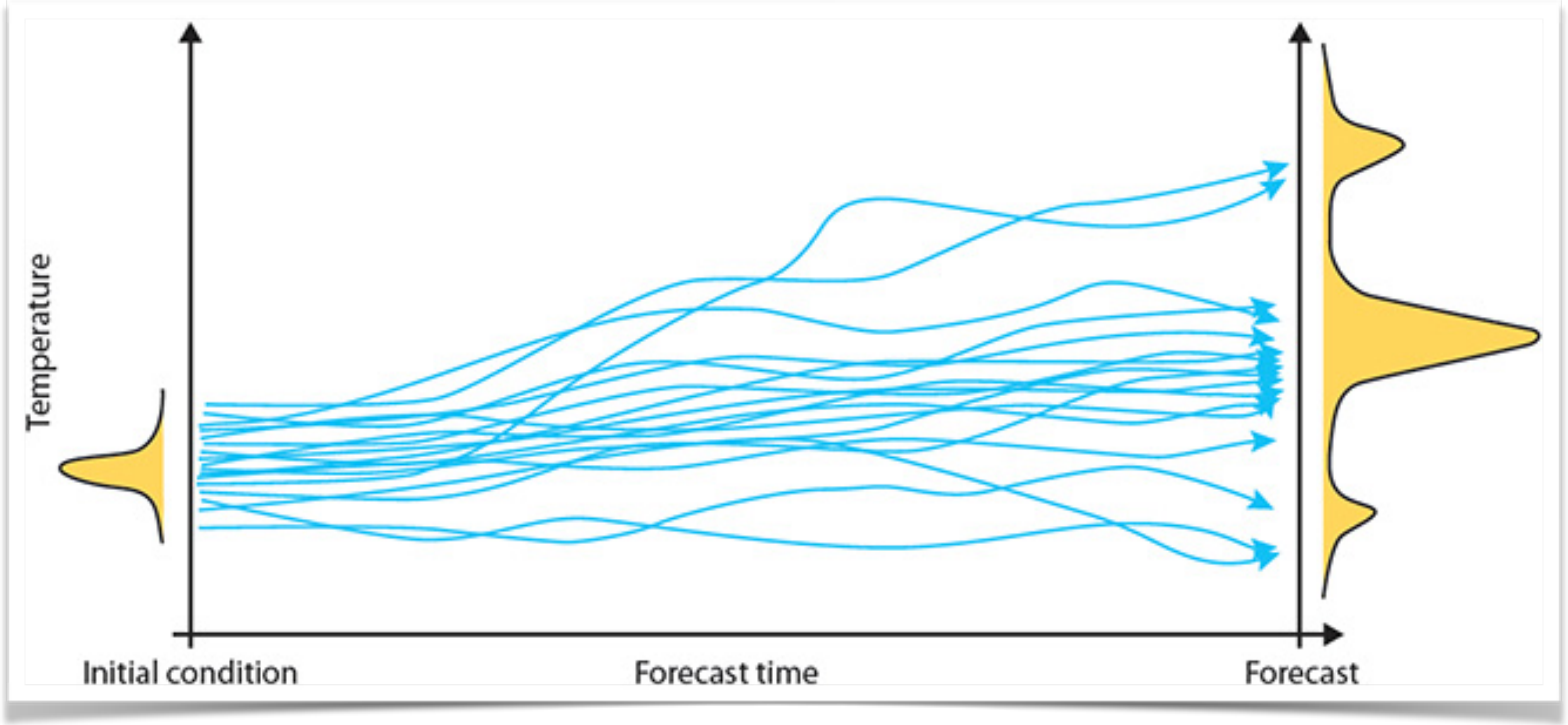
- Future challenges:
  “curse of dimensionality”; efficient exploration of entire design space
- Suitable for Q-acceleration:
  maybe, since efficient quantum algorithms for gradient estimation and quantum-based optimisation exist, however, mainly for discrete optimisation problems





Uncertainty quantification





Credit: ECMWF

- Ensemble forecasting: *Monte-Carlo analysis accounting for uncertainties in initial conditions, mathematical model, data etcetera*
- QoI: *range of possible scenarios (in terms of target functionals) and their likelihood of occurrence*
- Suitable for Q-acceleration: *maybe, since QC requires multiple simulation runs anyway but the problem sizes might be too large*

Quantum linear solvers

- QLSA: HHL ’08, Ambainis ’10, Clader et al. ’13, Childs et al. ’15, Kerenidis ‘17
- Input:
  - *NxN s*-sparse Hermitian matrix ***A*** with condition number $\varkappa = \|A\| \, \|A^{-1}\|$ and eigenvalues $1/\varkappa \leqq |\lambda_i| \leqq 1$, and a right-hand side unit vector ***f***
- Output:
  - Scalar QoI $J(\mathbf{u}) = \mathbf{u}^T \mathbf{M} \mathbf{u}$ (***M*** is a matrix) such that ***u*** solves $\mathbf{A}\mathbf{u} = \mathbf{f}$
- Complexity:
  - Best classical algorithm $O(\varkappa^{1/2} N)$
  - Quantum algorithms $O(\varkappa \log^3 \varkappa \log N)$—$O(\varkappa^2 \log N)$ ⬅ exponential speed-up

Q-FEM solver for Poisson’s equation

- $\boldsymbol{A}_h$ is s.p.d., *s*-sparse, well-conditioned ✓
- $\boldsymbol{f}_h$ is unit vector (after dynamic range scaling) ✓
- Matrix entries in row are accessible in time *O(s)* ✓
- Efficient ‘preparation’ of right-hand side quantum state (✓)

- Can we compute $||\boldsymbol{u}_h||^2 = \boldsymbol{u}_h^T \boldsymbol{u}_h$ where $\boldsymbol{u}_h$ solves Poisson’s equation?

Q-FEM solver for Poisson’s equation

PHYSICAL REVIEW A **93**, 032324 (2016)

Quantum algorithms and the finite element method

Ashley Montanaro and Sam Pallister*
School of Mathematics, University of Bristol, Bristol BS8 1TW, United Kingdom
(Received 5 January 2016; published 17 March 2016)

- *“when one compares quantum and classical algorithms for the FEM fairly by considering every aspect of the problem – including the complexity of producing an accurate approximation of the desired classical output – an apparent exponential quantum advantage can sometimes disappear”*
- *“there are still two types of problem where quantum algorithms for the FEM could achieve a significant advantage over classical algorithms: those where the solution has large higher-order derivatives, and those where the spatial dimension is large”*

Q-accelerated scientific computing

- **Challenges**
  - Sufficient #qubits for realistic problem sizes
  - Q-system hw/sw infrastructure, error correction
- **Open problems**
  - Real numbers (IEEE-754, continuous encoding, custom formats)
  - Divide-and-conquer/domain decomposition (no-cloning principle!)
  - Validation of results (using classical supercomputers?)
  - Reproducibility of simulations (but that’s a problem in HPC as well)

Let’s get ready for Q-accelerated scientific computing and exchange ideas both ways!

Thank you for your attention!