Anmerkungen zum HPC – aka Ein Kessel Buntes

Hans-Joachim Bungartz

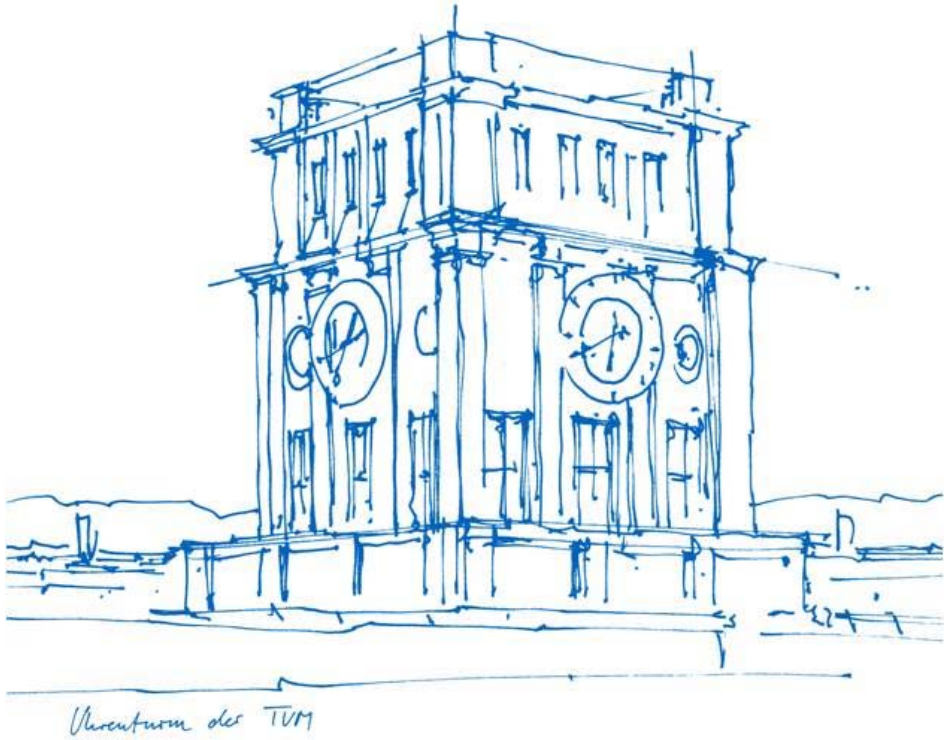
Technische Universität München

Institut für Informatik

10. HPC-Statuskonferenz der Gauß-Allianz

1.10.2020

bungartz@in.tum.de



Inhalt

Software

Effizienz & Performance

Paradigmen

Fehlertoleranz

NHR

Genauigkeit

Co-Design, Algo‘s & App‘s

Die vielfach strapazierten Paradigmen …

„Paradigms of Science“ – eine übliche Sehweise (z.B. G. Strawn, 2012)

➔ **Klassiker:**

#1 Experiment
#2 Theorie

➔ **IT-bezogen:**

#3 Computational Science (& Engineering)
#4 Data-intensive Science (& Engineering)

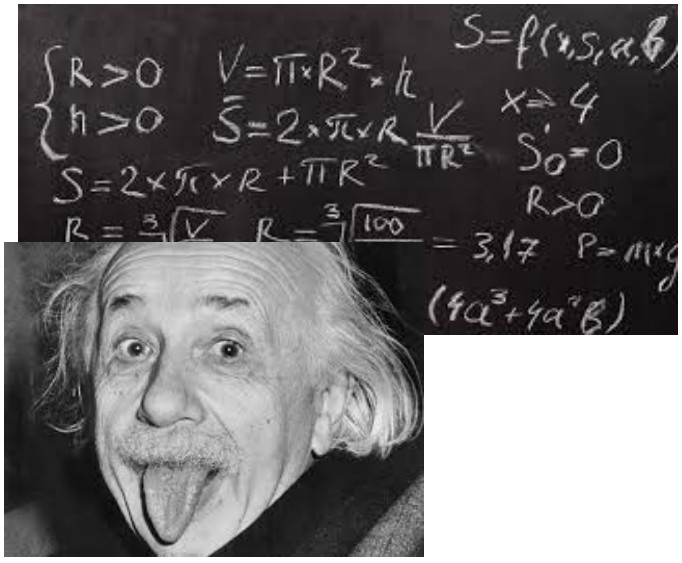
Auch gesehen (D. Robertson)

#A *collecting* information (Jäger & Sammler ➔ #1)
#B *compressing* information (Verarbeiter ➔ #2 & #3)
#C *organizing* information (Strukturierer & Organisierer ➔ #4)

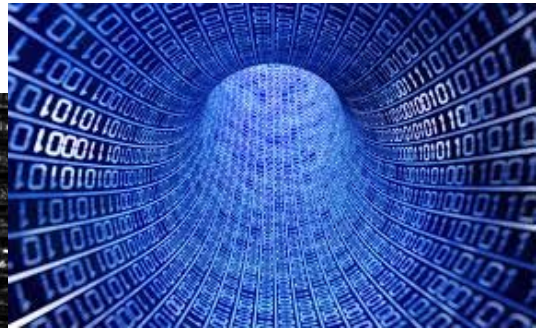
MyParadigms …

#1 – Experiment

#2 – Theorie

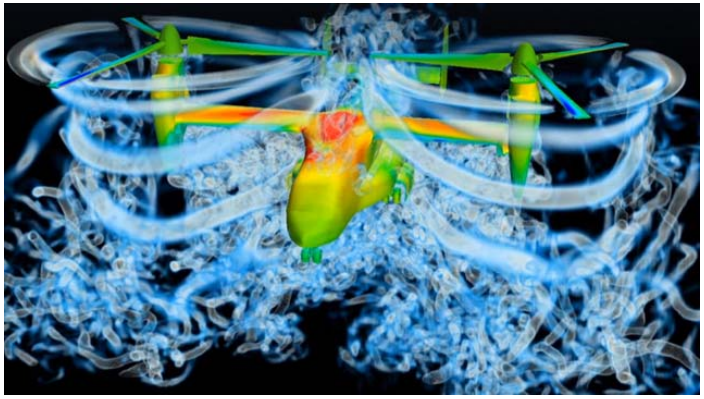


#3 – rechnergestützt



Erscheinungsformen von #3 – „computational“

#3a – modellgetrieben/deduktiv: Simulationswissenschaft



Methodischer Hintergrund u.a.:

mathematische Modellierung
numerische Algorithmik
paralleles Rechnen
Domänenwissenschaften

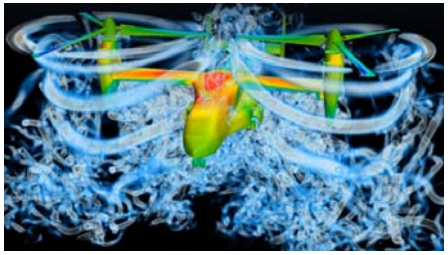
Methodischer Hintergrund u.a.:

theoretische Informatik
diskrete Mathematik
Statistik
Datenanalytik
Domänenwissenschaften

#3b – datengetrieben/induktiv: Datenwissenschaft

Zusammenspiel #3a und #3b, und die Rolle von HPC

#3a – Simulation



#3a: Daten aus Modellen – #3b: Modelle aus Daten
#3: Brücke zwischen Experiment und Theorie,
systematisches Kombinieren von Ergebnissen beider Welten

#3b – KI/ML

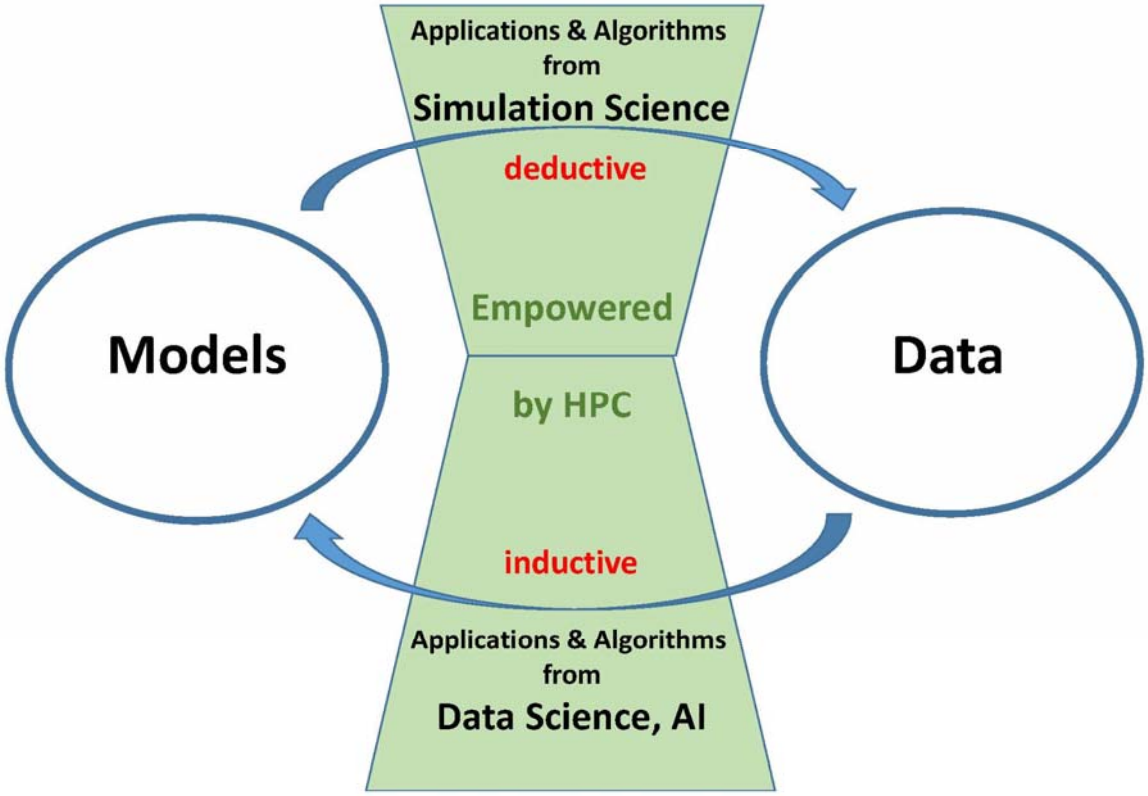
in weiten Teilen notwendige Grundlage & zentraler Enabler *(belegt)*

HPC ⟷ AI/ML

in weiten Teilen notwendige Grundlage & zentraler Enabler *(vielen, aber noch nicht allen klar)*

High-Performance Computing:
(Numerische) Algorithmen, massiv parallel / großskalig, Performance, Effizienz, Workflows

Aus HPC-Sicht: #3a ➔ #3b = mehr Anwendungen



Algorithmische Nähe von #3a und #3b

Mathematisches Fundament, „Mathematisierung“

- Simulationswissenschaft: lange etabliert (Modellierung, Optimierung, Numerik, Stochastik, UQ)
- Datenwissenschaft/KI: relativ jung (black box vs. explainable AI)

Algorithmische Grundlagen von Datenanalytik & KI

- viel Numerik: iterative Verfahren allgemein, numerische lineare Algebra (Eigenwerte und -vektoren, SVD, Quadratur, GEMM 4 CNN, …)
- ähnliche Design Patterns: z.B. Hierarchie (multi-level vs. multi-layer)
- viel Statistik/Stochastik

Effizienz/Performance als Leitprinzip

- algorithmische Effizienz & „Implementierungseffizienz“
- gleiche Prinzipien auch bei unterschiedlicher Hardware
- HPC unabdingbar („HP“ im eigentlichen Sinn, nicht zwangsläufig large-scale)

Alles Philosophie ☺ – oder konkrete Konsequenzen?

Anzustrebende Nähe bei Ausbildung

- Simulationswissenschaft – Datenwissenschaft: komplementär, aber nicht orthogonal
- Beispiel: Rufe aus KI/ML nach mehr mathematischen Grundlagen

Anzustrebende Nähe bei Betrieb

- Kompetenzzentren für das eine oder das andere sind nicht zwei Welten!
- Aufstellen der NFDI orthogonal zur „computational Infrastruktur“ (keine Förderung – „das ist doch da“; keine Querschnittskonsortien auch im zweiten Call) wirkt etwas unglücklich

Richtiges Positionieren von HPC im Verhältnis zu #3, und insb. AI/ML

Anzustrebende Nähe bei Förderprogrammen

For Performance, Algorithms are the Essence

Broad range of functionalities:

- Discretization schemes, numerical schemes
- Parallelization, communication, load balancing
- Compression, dimension reduction
- Statistic, stochastic
- Analytical (ML, …)

Algorithm life cycle: design – analysis – implementation – tuning

➔ **Algorithm Engineering: design – analysis** – implementation – tuning

➔ **Performance Engineering:** design – analysis – **implementation – tuning**

… and: efficiency comprises both!

Notions of efficiency:

- Long way from the classical Landau order-of-N-to-the-something to today's multi-faceted performance landscape

Notions of Efficiency

#op – the classical complexity measure (still not a bad idea to improve …)

#flops – % of peak performance (good if optimized for the best algorithms)

- at node level
- at system level, weak / strong scalability

#bytes (or #bytes/flop) – the communication-related one

- Communication-avoiding / memory-efficient / compute-bound as good guidelines

#Watt (or #Watt/flops or #flops/Watt) – the energy-related one

- “Cool” solvers – are they really different from classical “fast” ones? May slower be cooler?

Time-to-solution

- Q: what to include into consideration, i.e. concerning “time” – is runtime-only sufficient?

digits per flop – accuracy-related

science (pubs) per flop, science (insight) per flop, …

Another View at Efficiency: Cost-Benefit Ratio

General principle:

- For a continuous problem **P** with solution ***u*** (the quantities of concern), provide discrete approximations $\mathbf{P}_n$ and $\boldsymbol{u}_n$ such that some error ε gets small.

Cost-benefit consideration *(cf. notion of ε-complexity):*

- **benefit**: obtained error/accuracy ε
- **cost** (I): the classical ones – (1) $N(\varepsilon)$ d.o.f. to obtain ε, (2) $g(N)$ ops. to solve $\mathbf{P}_n$
- **cost** (II): runtime $T(g(N))$, energy consumption $E(g(N))$ for building & solving $\mathbf{P}_n$
- **cost** (III): more metrics – memory/cache efficiency, communication avoiding, …

Classical approach (PDE-based models, *d* dimensions, regularity *p)*:

- exponential growth of ***N*** in *d*: $N = O(\varepsilon^{-d/p})$ (curse of dimension)
- polynomial growth of *g* in ***N***: $g = O(N^k)$

Starting points for improvements:

- fast solvers for $\mathbf{P}_n$ (multilevel): ***g* optimal** (linear in ***N***, for example)
- efficient discretizations: **improve *N*** (small, no or only weak *d*-dependence)
  - approaches of higher order, adaptive mesh refinement (*problem-dependent)*
  - **a priori grid structure**, *problem-independent*

Co-Design – the Right Way

Important: “inclusive/holistic” co-design, i.e. including the algorithmic core, i.e.

- **Co-design of applications and *algorithms***

  PLUS

- **Co-design of *algorithms / algorithmic patterns* and systems**

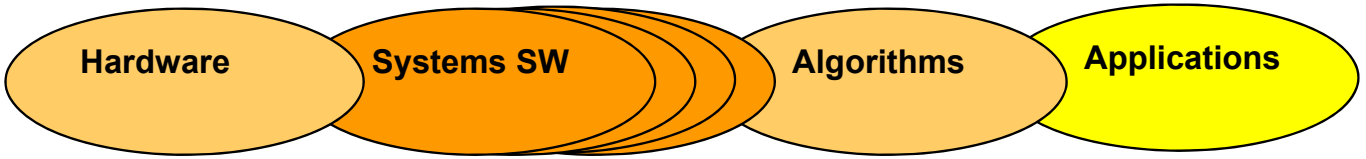
… in a tightly interwoven way (… and maybe even more sub-steps)

The frequent shortcut *“applications & systems”* is not equivalent!

➔ *Not every well-established & well-performing code has the best algorithms inside!*

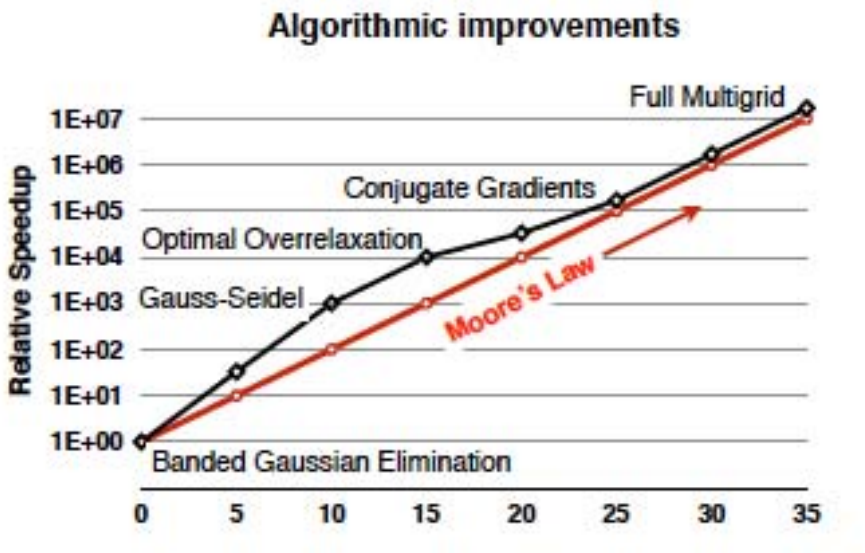
➔ *“100 hours of 80% peak is not better than 10 hours of 8% peak”!*

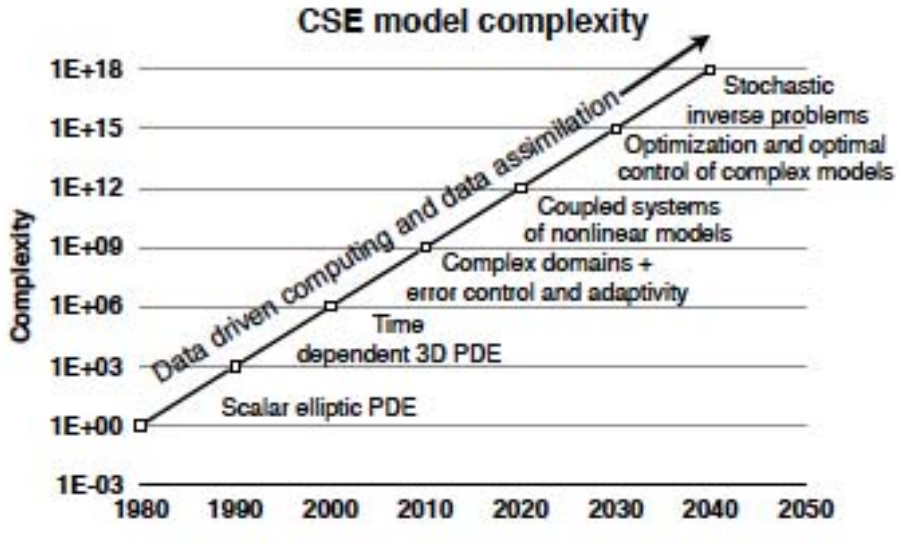
➔ Next question: *“10 hours if it’s not the proper thing?”*



Algorithms – essential for the high performance

Just one success story – linear solvers





Rüde, Willcox, Curfman McInnes, De Sterck: Research and Education in CS&E, SIAM Review 60(3), pp. 707–754, 2018.

Mixed precision & mixed arithmetics

Idea: Right-sizing precision is one of the best options we have for reducing energy use and increasing speed, now that Moore's law is slowing and the "memory wall" is worse than ever before.

Observation: Reducing some 64-bit representations to 32-bit can *more* than double the speed, because of the memory hierarchy. Variables fit into closer levels of cache. Also more than doubles energy efficiency.

Posits vs. Floats: Posit hardware support is now happening at **11** different companies, mostly in RISC-V and in GPU-like accelerator boards. Posits frequently allow 64-bit floats to be replaced with 32-bit posits.

Good news for mixed arithmetics: In the latest Draft Posit Standard, the number of *es* bits is always 2; this means changing precision is as trivial as it is for integers: Pad with 0s for more precision, or round them off for less precision. It is super-easy to make bitwise adjustments to find the minimum precision that suffices.

Software

Ist und bleibt ein Riesen-Thema

SPPEXA – abgeschlossen

- Erfolgsgeschichte – Nachhaltigkeit eingeschränkt durch Singularität
- Zweiter Berichtsband (Springer LNCSE) erschienen
- Darin auch zusammenfassender Artikel mit Zahlenmaterial und Software-Übersicht

DFG Forschungssoftware Call

- Bearbeitung zweiter Call verzögert

Spannend: Ausmaß an diesbezüglicher Unterstützung durch NHR und NFDI

SPPEXA's 6 Research Directions

Computational algorithms
- Aiming at large-scale machines
- Efficient w.r.t. "modern" complexity measures

System software and runtime libraries
- Process scheduling
- System health monitoring
- Resilience handling

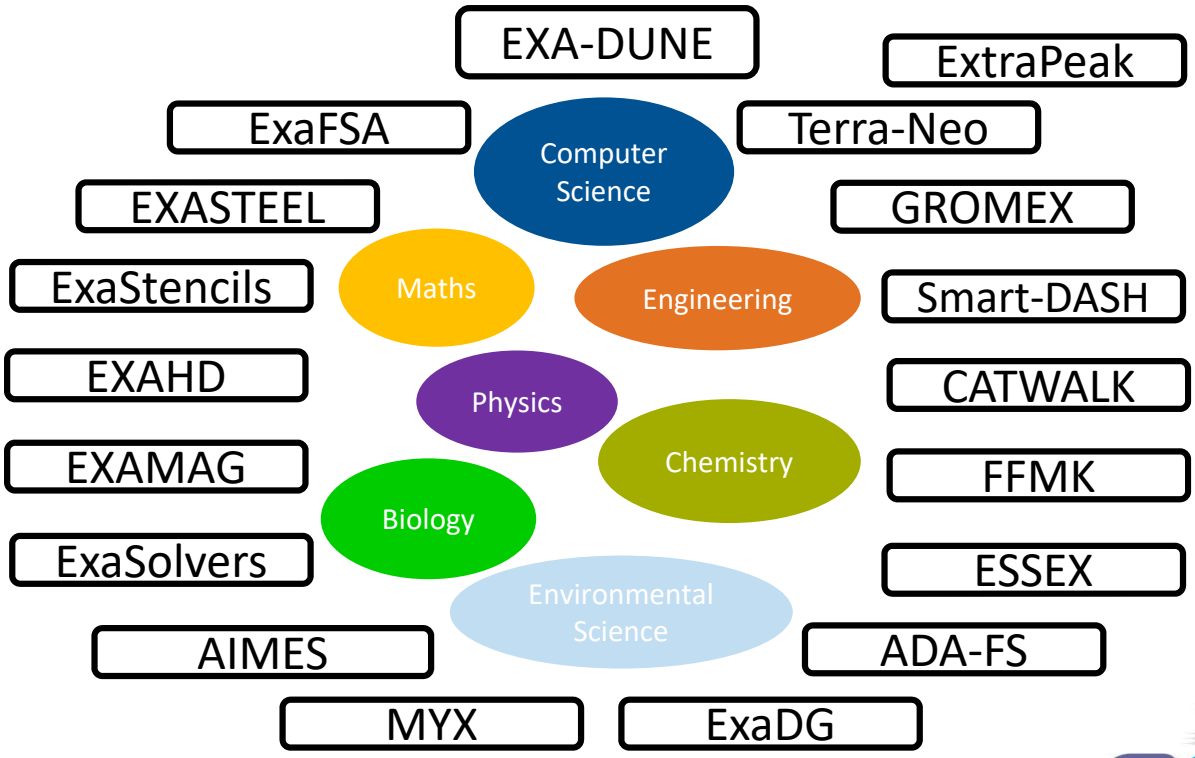
Software tools
- Compiling, running, verifying, testing, optimizing

Application software
- Key driver for exascale
- Hardware-software
- Co-design necessary

Programming
- Make traditional approaches exascale-ready
- New programming models

Data management
- Process large data sets
- Archive, make data available

A Really Interdisciplinary Endeavor



A Really International Endeavor

Japan:

- RIKEN
- Tokyo Tech
- University of Tsukuba
- University of Tokyo
- Tohoku University
- Tokyo University of Science
- Toyo University

France:

- Université de Versailles
- Université de Strasbourg
- Maison de la Simulation, Saclay

other countries:

- TU Delft, Netherlands
- USI Lugano, Switzerland
- Royal Institute of Technology, Sweden
- UCLA, USA
- ANU, Australia
- Hebrew University Jerusalem, Israel

Projects with F/F&J connection:

- **ExaFSA**: Exascale Simulation of Fluid-Structure-Acoustics Interactions
- **ExaStencils**: Advanced Stencil-Code Engineering
- **EXAMAG**: Exascale Simulations of the Magnetic Universe
- **ESSEX**: Equipping Sparse Solvers for Exascale
- **EXASOLVERS**: Extreme-Scale Solvers for Coupled Problems
- **AIMES**: Advanced Computation and I/O Methods for Earth-System Simulations
- **MYX**: MUST Correctness Checking for YML and XMP Programs

Fehlertoleranz

Stand der Technik: Dagstuhl-Seminar
„Resiliency in Numerical Algorithm Design for Extreme-Scale Simulations“

März 2020 – eines der letzten vor dem Lock-down

Abschlussbericht kurz vor Fertigstellung, erscheint als Dagstuhl-Report

NHR

Blick auf die Historie aus persönlicher Sicht – Vorbereitung:

- Nationaler Koordinierungsausschuss zur Beschaffung und Nutzung von Höchstleistungsrechnern – WR-Bericht 2007
- Papier *Empfehlungen zur Einrichtung einer programmatisch-strukturellen Linie „Hochleistungsrechner“ im Rahmen der Förderung von Forschungsbauten an Hochschulen einschließlich Großgeräten* – WR 2008
- Anhörung zur Zukunft des NatKo – WR 2011
- WR-AG 2011 ➔ Papier *Strategische Weiterentwicklung des Hoch- und Höchstleistungsrechnens in D* – WR 2012
- WR-AG 2013/14 ➔ Papier *Empfehlungen zur Finanzierung des Hoch- und Höchstleistungsrechnens in D* – WR 2015

NHR

Blick auf die Historie aus persönlicher Sicht – Umsetzung:

- GWK-Arbeitsgruppe 2016/2017
- Anfrage an den DFN-Verein (Land Berlin für die GWK) wg. Bereitschaft zur Übernahme des Betriebs einer Geschäftsstelle des NHR-Strategieausschusses für die Implementierungsphase des NHR – Sommer 2018
- Umsetzung dessen als 2-jähriges Projekt – Winter 2018/2019
- Besetzung des Strategieausschusses, Festlegung des Prozesses (Anträge, Begutachtung, …) – 2019
- Anträge, Begutachtung, Empfehlungen des Strategieausschusses und des GWK-Ausschusses – Frühjahr/Sommer 2020
- GWK-Entscheidung – November 2020

NHR

Spannend wird …

- … wie‘s dann losgeht,
- … mit welcher Governance,
- … und was am Ende von den ursprünglichen Ideen tatsächlich umgesetzt wird

Vielen Dank!