Testen numerischer Modelle

Universität Hamburg
Seminarvortrag in *Softwareentwicklung in der Wissenschaft*
Sommersemester 2017

Marcel Heing-Becker

7heing@informatik.uni-hamburg.de

Ablauf

1. Grenzen der Testbarkeit von wissenschaftlichen Modellen
   - Validierung, Verifikation
   - Kalibrierung, Bestätigung
2. Modelle
   - Realität $\longrightarrow$ Simulationsergebnisse
   - Tests auf dem Weg
3. Beispiele aus der Klimaforschung

Grenzen der Testbarkeit

> "Verification and validation of numerical models of natural systems is impossible. This is because natural systems are never closed and because model results are always non-unique.
>
> – Oreskes, Shrader-Frechette, Belitz. 1994. [1]

Verifikation

- *Verifikation*: Nachweis eines wahren Sachverhalts
- *Verifiziertes Modell*: Abbildung der Realität

Ein offenes System zu verifizieren: **unmöglich**.

Verifikation

Geschlossenes System:

$$\mathbf{p} \Rightarrow \mathbf{q}$$

In einem geschlossenen System ist sicher, dass aus einem wahren $\mathbf{p}$ ein wahres $\mathbf{q}$ folgt.

Verifikation

Offenes System:

Halte ich diesen Vortrag, so bestehe ich das Seminar.

Nicht modelliert: E-Mail mit Abgabe des Seminarberichts an falsche Adresse geschickt. Oder: Ein γ-Strahl des Vela-Pulsars hat das Bestehen-Bit im Serverpuffer beim Eintragen in STiNE geflippt.

Die Verifikation dieses Modells *Seminarpartizipation* schlägt fehl.

Offenes System

Not logged in Talk Contributions Create account Log in

WIKIPEDIA
The Free Encyclopedia

Article Talk | Read Edit View history | Search Wikipedia

Main page
Contents
Featured content
Current events
Random article
Donate to Wikipedia
Wikipedia store

Interaction
Help
About Wikipedia
Community portal
Recent changes
Contact page

Tools
What links here
Related changes
Upload file
Special pages
Permanent link
Page information
Wikidata item

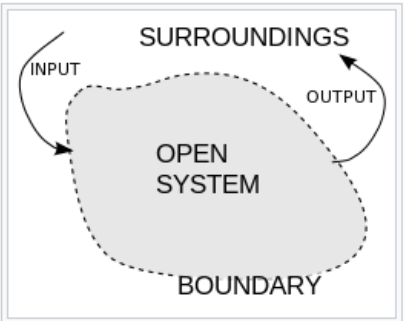
Open system (systems theory)

From Wikipedia, the free encyclopedia

An **open system** is a system that has external interactions. Such interactions can take the form of information, energy, or material transfers into or out of the system boundary, depending on the discipline which defines the concept. An open system is contrasted with the concept of an isolated system which exchanges neither energy, matter, nor information with its environment. An open system is also known as a constant volume system or a flow system.

The concept of an open system was formalized within a framework that enabled one to interrelate the theory of the organism, thermodynamics, and evolutionary theory.[1] This concept was expanded upon with the advent of information theory and subsequently systems theory. Today the concept has its applications in the natural and social sciences.

In the natural sciences an open system is one whose border is permeable to both energy and mass.[2] In Thermodynamics a closed system, by contrast, is permeable to energy but not to matter.



Open system have input and output flows, representing exchanges of matter, energy or information with its surroundings.

Open systems have a number of consequences. A closed system contains limited energies. The definition of an open system assumes that there are supplies of energy that cannot be depleted; in practice, this energy is supplied from some source in the surrounding environment, which can be treated as infinite for the purposes of study. One type of open system is the radiant energy system, which receives its energy from solar radiation – an energy source that can be regarded as inexhaustible for all practical purposes.

[.] https://en.wikipedia.org/wiki/Open_system_(systems_theory) – 08. Mai 2017

Offenes System

Im physikalischen Bereich qualifizierend:

- Eingabeparameter, die inhärent nicht vollst. bekannt sind, *For example, hydrogeological models require distributed parameters such as hydraulic conductivity [...] which are always characterized by incomplete data sets.* [1]
- Fakten, die derzeit nur approximiert genau sind
- Gesetzmäßigkeiten, die skalenabhängig verlustbehaftet oder berechtigt kritisch gesehen sind
- durch Schaffung von Hilfsannahmen zur Modellierung

Validierung

Ein Modell als *valide* betrachtet, wenn:

- frei von logischen Fehlern: Widersprüchen,
- frei von begründbar widerlegbaren Annahmen und
- angemessene Wahl der Eingabeparameter.

Validierung als Qualität der Modellkonsistenz zu sehen.

Kalibrierung, Bestätigung

- *Kalibrierung* bei unbek. unabhängigen Daten:
  1. Training: unabhängige Daten bis zur Übereinstimmung adjustieren
  2. Test: trainiertes Modell auf übrige Daten anwenden
- *Bestätigung*: Testen von Hypothesen im Vergleich mit Beobachtungen

→ Bestätigung verifiziert nicht! Sie macht die Hypothese un-/ wahrscheinlicher valide.

Kritik

- P. J. Roache, *Building PDE codes to be verifiable and validatable*. 2004. [2]
- Begriffe *Verifikation*, *Validierung* und *Bestätigung* seien unzureichend oder irreführend behandelt
- Vernachlässigung der Tatsache, dass beliebige Präzision Korrektheit bedeuten kann
- V&V seien im physikalischen Simulationskontext hinreichend akzeptiert beschrieben

V2V

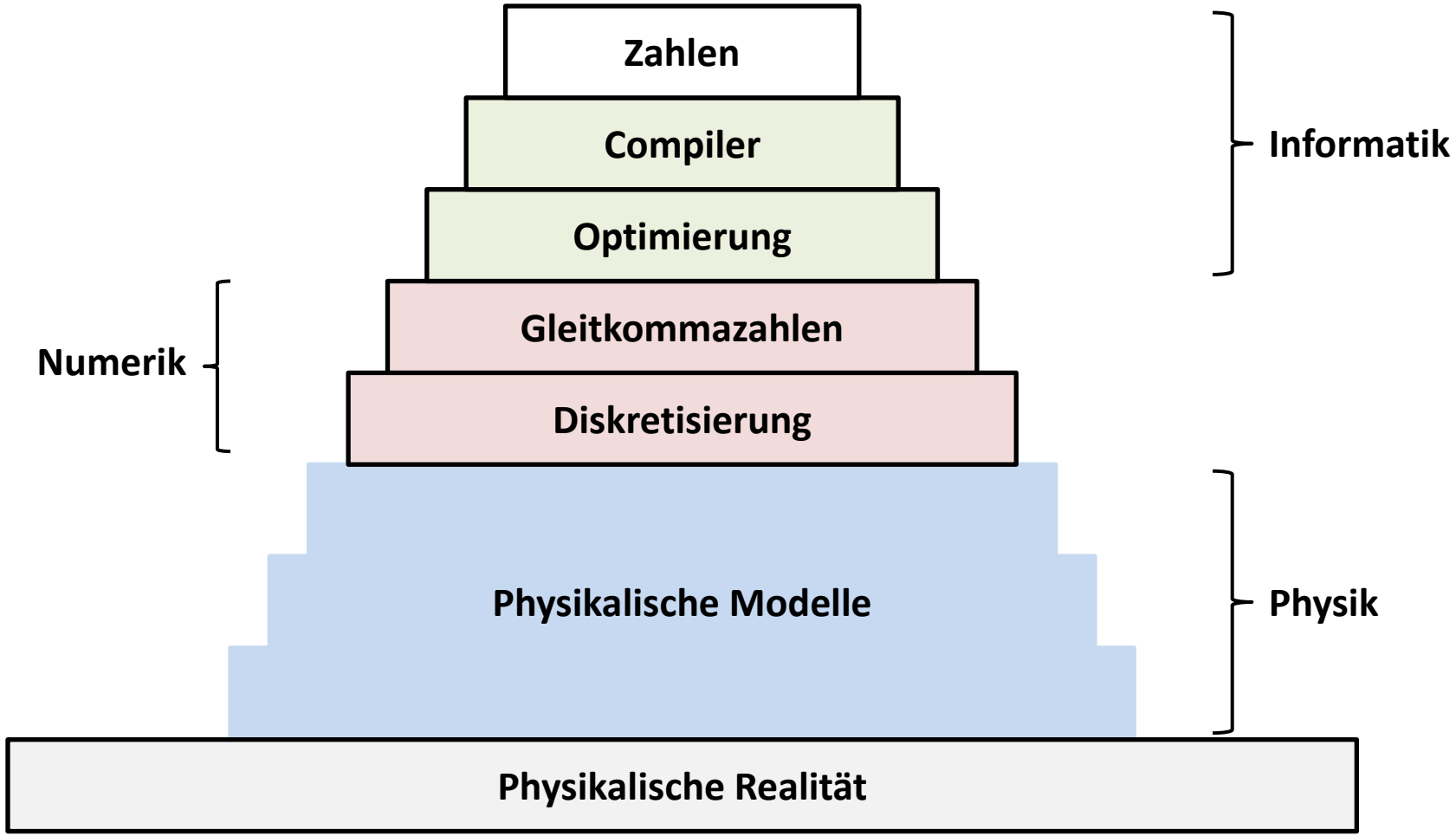
- V2V (V&V):
  - *Code-Verifikation*: Der Programmcode tut, was er tun soll, fehlerfrei, mit der geforderten numerischen Präzision.
  - *Mathematische Verifikation*: Das codierte Modell kann bei einer gewählten Präzision ein gefordertes Ergebnis korrekt liefern.
  - *Validierung*:
    - Vergleich des Modells mit Experimenten
    - Es sei *validiert*/*validierter Code* nie ohne Quantifizierung zu verwenden
    - Auf welchen Skalen und Parameterdefinitionen welcher Fehler?
- Bestätigung: Reproduzierbare Code-Verifikation

Modelle

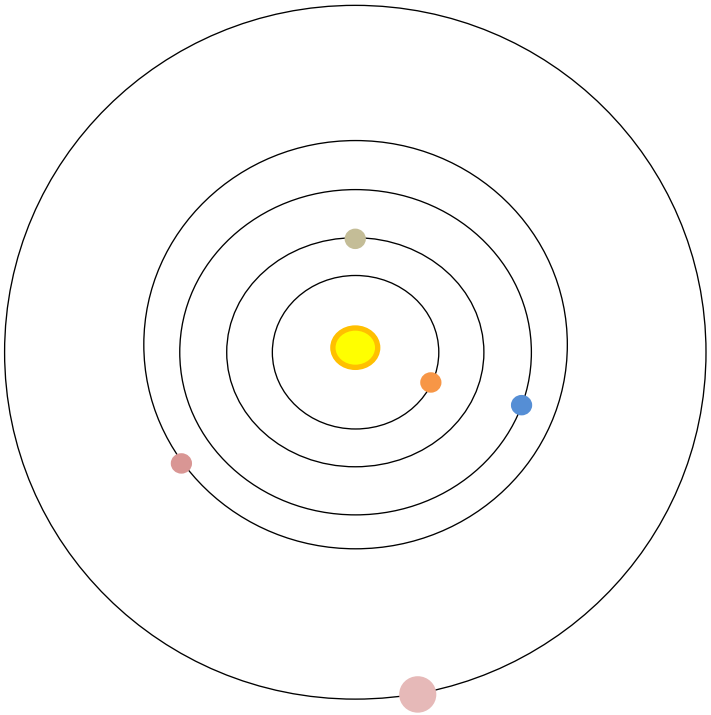
Modelle offener Systeme

- Fiktionale Werke
- Abbildungen gespickt mit
  - *Idealisierungen*: Grenze der Beschreibungsmöglichkeit
  - *Approximationen*: Hinnahme von Informationsverlust
- Umgang mit unzulänglichen Informationen
- Können falsche Hoffnung bestätigen
- In keinem Fall einer Verifikation dienlich

Tower of Approximations



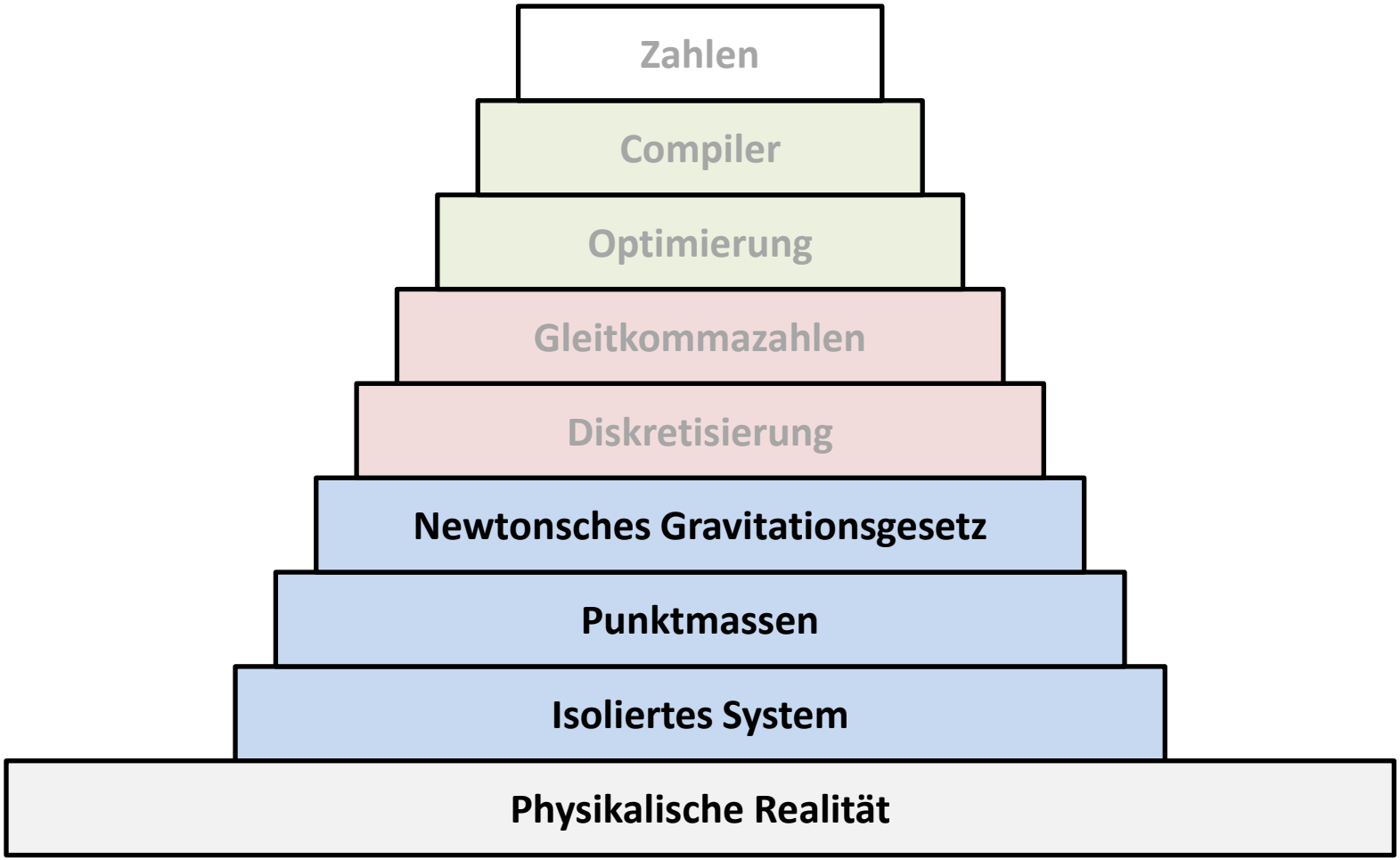
Modell: Sonnensystem



- Keine Nachbarsysteme, Monde
- Annahme: Massenmittelpunkte
- Annahme: stetige Bewegung

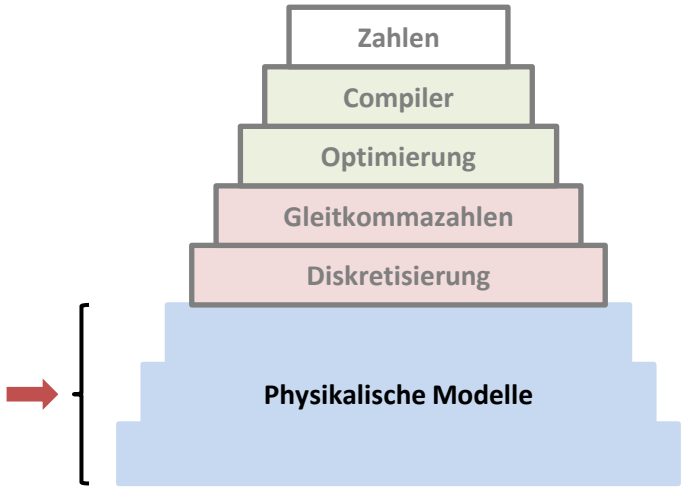
→ Newtonsches Gravitationsgesetz

Tower of Approximations



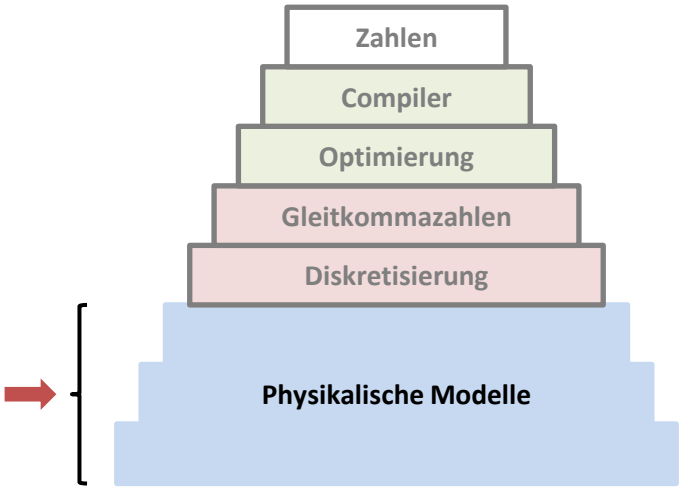
Approx. physikalischer Modelle

- Beobachtungen
- Hypothesen
- Vorhersagen
- (neg.) Bestätigung
- Kalibrierung
- Inferenz



Approx. physikalischer Modelle

- Beobachtungen
- Hypothesen
- Vorhersagen
- (neg.) Bestätigung
- Kalibrierung
- Inferenz

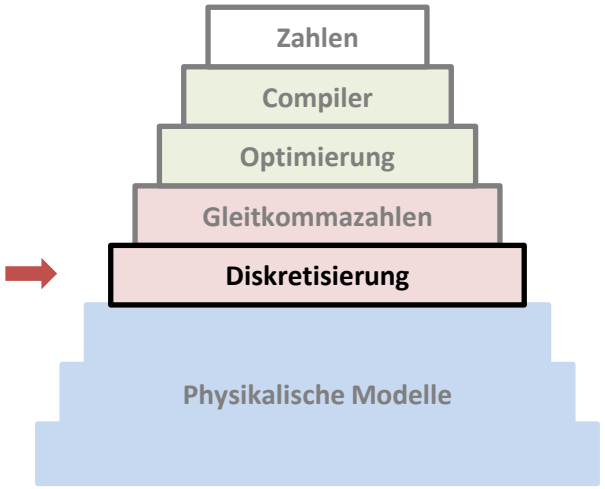


→ Analytisches Modell, z. B.

$$\mathbf{F}_{ij} = -G\frac{m_i m_j}{\left|\mathbf{r}_i - \mathbf{r}_j\right|^2}\frac{\mathbf{r}_i - \mathbf{r}_j}{\left|\mathbf{r}_i - \mathbf{r}_j\right|}$$

Berechnung phys. Modelle

- $\mathbf{r} \in \mathbb{R}$ mit unendlich Informationen
- Computer-Berechnung mit endlichen Informationen
- Abwägung:
  - Übereinstimmung
  - Präzision
  - verfügbare Computer-Ressourcen

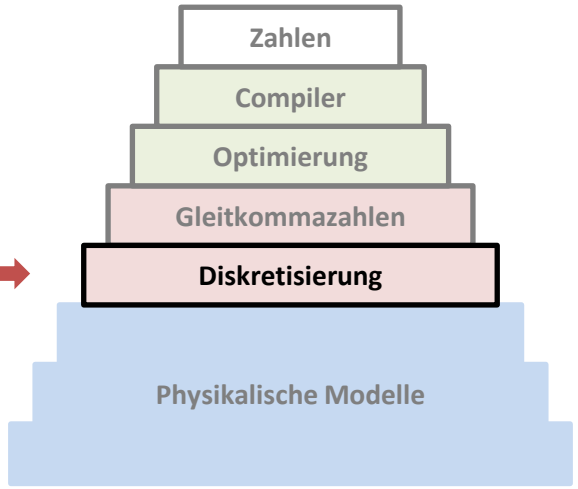


Computable Analysis

- Transformation analytischer Verfahren
- Reelle Zahlen als Funktionen ausgedrückt, z. B.:

$$\pi: \pi(n)$$

Mit einem Approximationsfehler $< 2^{-n}$.



Numerische Verfahren

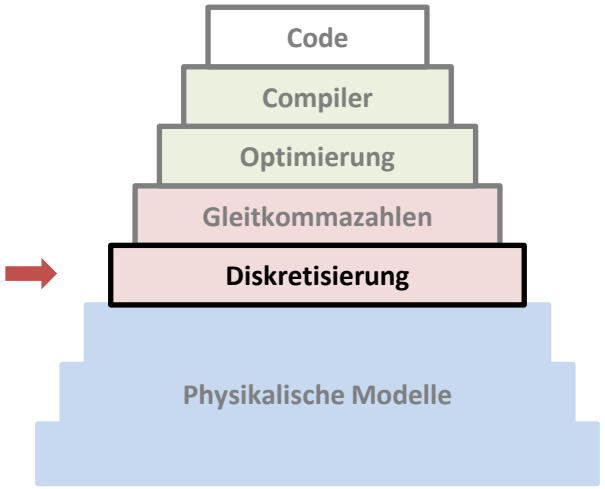
Kondition: $\|f(\tilde{x}) - f(x)\|$

Stabilität: $\|\tilde{f}(\tilde{x}) - \tilde{f}(x)\|$

Konsistenz: $\|\tilde{f}(x) - f(x)\|$

Konvergenz: $\|\tilde{f}(\tilde{x}) - f(x)\|$

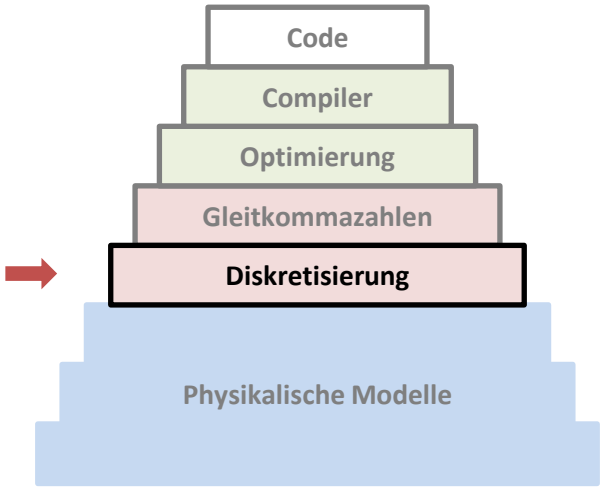
Benchmarking des Bezugsverlusts



MMS

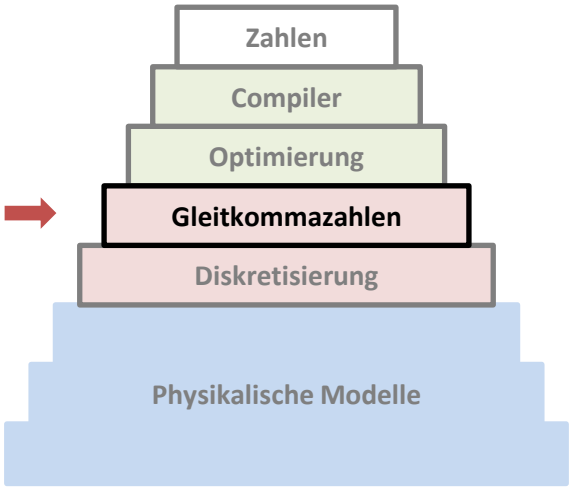
- Code-Verifikation: *Method of Manufactured Solution* (MMS)

- Betrachte Code als nichtlineares System L
- Konstruiere eine analytisch korrekte Lösung M
- M definiert über unabhängige Parameter
- L' als System, in dem M eine Gleichung löst
- L' = L – Q
- Q = L{M}
- Konvergenztests für Definitionsbreiche, Präzision

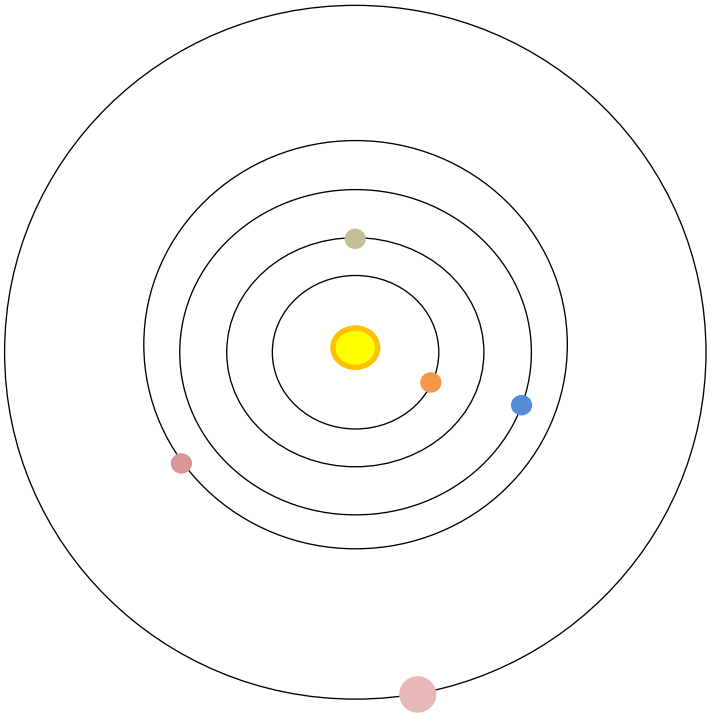


Gleitkommazahlen

- IEEE 754
- einfache, doppelte Präzision



Modell: Sonnensystem



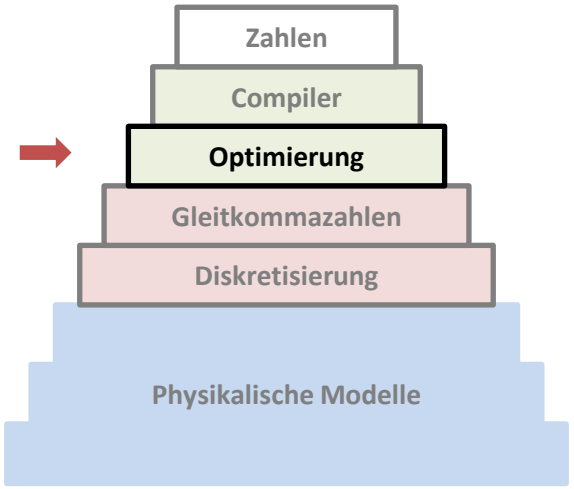
Mathematisch bedingte Einschränkungen:

- Position der Planeten nur zeitdiskret ermittelbar
- Position der Planeten näherungsweise genau

Optimierung

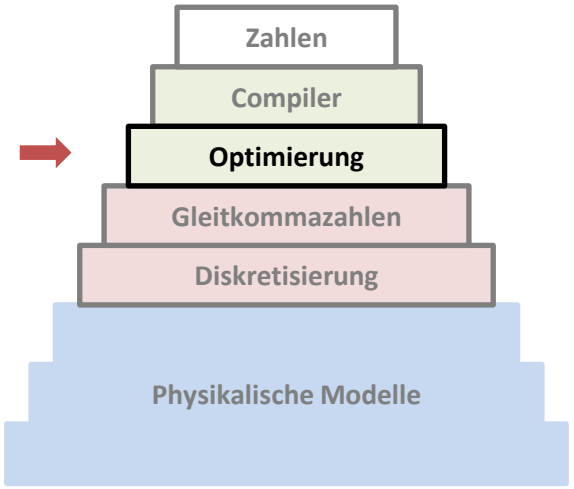
- Anordnung von FLOPs:
  $\mathbf{a * (b + c) \neq a * b + a * c}$
- Parallelisierung
  z. B. nichtdeterministischer Aggregation:

$$s = \sum_{i=1}^{n} x_i y_i$$



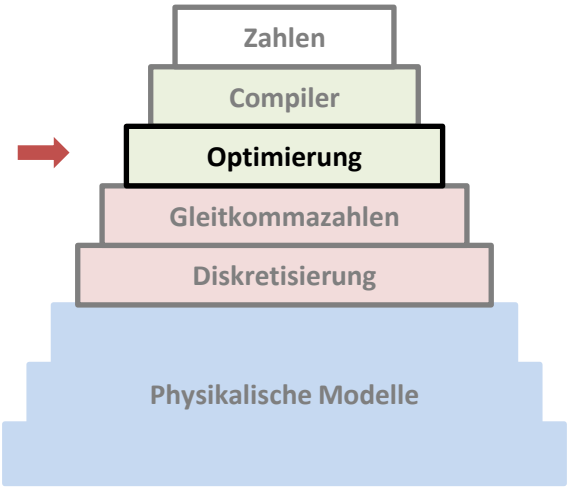
Optimierung: Tests

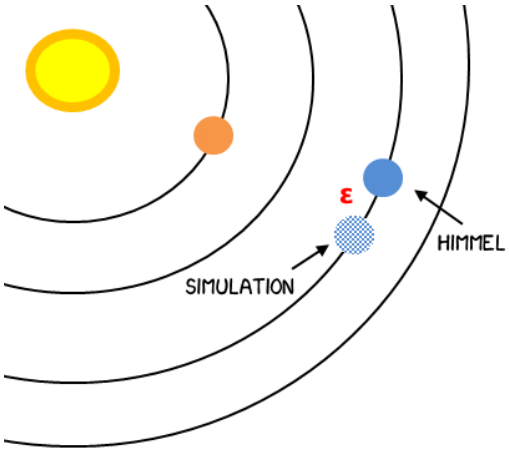
- *Downright cheating*
  - Gut aussehndes Ergebnis finden
  - Test mit Fehlertoleranz schreiben
  - Kann man machen
- Entkoppelter Vergleich
  - Verfahren implementieren
  - Vergleich mit konkurrierendem Modell
- Toleranzbereich?



Optimierung: Tests

- *Downright cheating*
  - Gut aussehndes Ergebnis finden
  - Test mit Fehlertoleranz schreiben
  - Kann man machen
- Entkoppelter Vergleich
  - Verfahren implementieren
  - Vergleich mit konkurrierendem Modell
- Toleranzbereich?



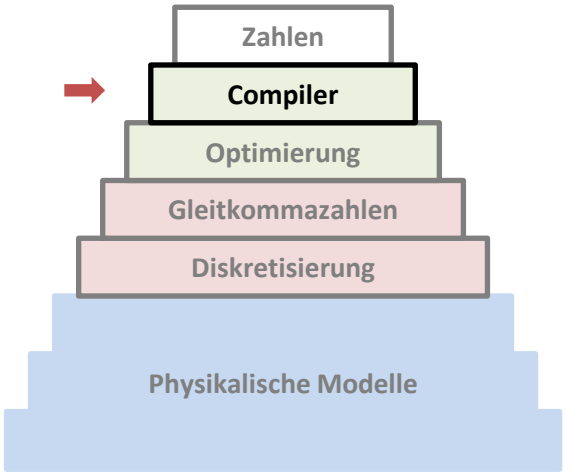


Compiler

Transformations

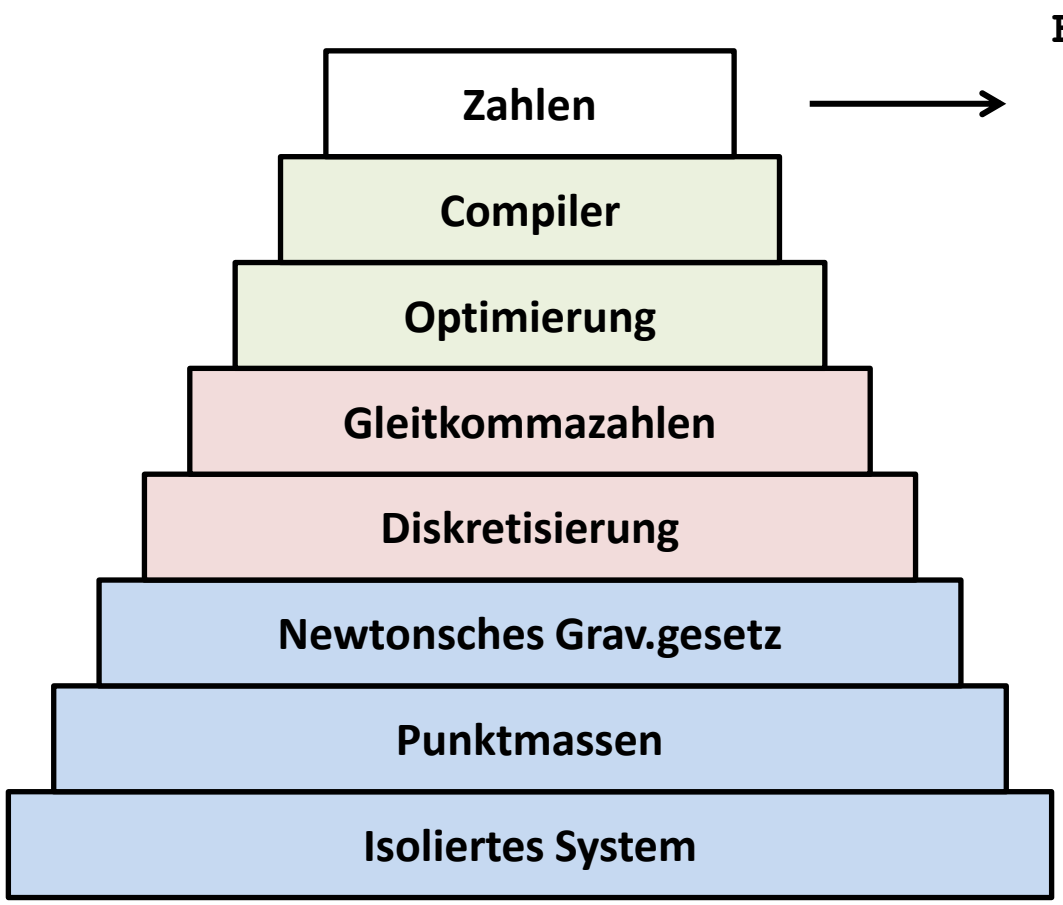
The table below summarizes the most common transformations that affect floating-point semantics. The table uses (MODE) to indicate a typecast to a type with that mode. Variable names that are mode names refer to variables of a type with that mode. Variables starting with a capital C refer to constants known at compile time.

Compliance	Original	Replacement	Violates
Full	`x / C`	`x * (1.0 / C)`	*none*
Default	`0.0 - x`	`-x`	`HONOR_SIGNED_ZEROS`
	`x - 0.0`	`x`	`HONOR_SIGNED_ZEROS`
	`0.0 / x`	`0.0`	`HONOR_SIGNED_ZEROS` and `HONOR_NANS`
	`x / 1.0`	`x`	`HONOR_SNANS`
	`x / -1.0`	`-x`	`HONOR_SNANS`
	`-(a / b)`	`a / -b`	`HONOR_SIGN_DEPENDENT_ROUNDING`
	`-(a / b)`	`-a / b`	`HONOR_SIGN_DEPENDENT_ROUNDING`
Relaxed	`(SF) ((DF) xf)`	`(SF) xf`	*any*
	`-(a - b)`	`b - a`	
	`x - x`	`0.0`	
	`-(a + b)`	`(-a) - b`	
	`-(a + b)`	`(-b) - a`	
	`x / C`	`x * (1.0 / C)`	
	`(a * c) + (b * c)`	`(a + b) * c`	
	`(a + b) * c`	`(a * c) + (b * c)`	
	`a * (b / c)`	`(a * b) / c`	
	`a * (b * c)`	`(a * b) * c`	
	`a + (b + c)`	`(a + b) + c`	
	`x + C1 == C2`	`x == C2 - C1`	
	`x + C1 != C2`	`x != C2 - C1`	



[.] https://gcc.gnu.org/wiki/FloatingPointMath – 01. Juni 2017

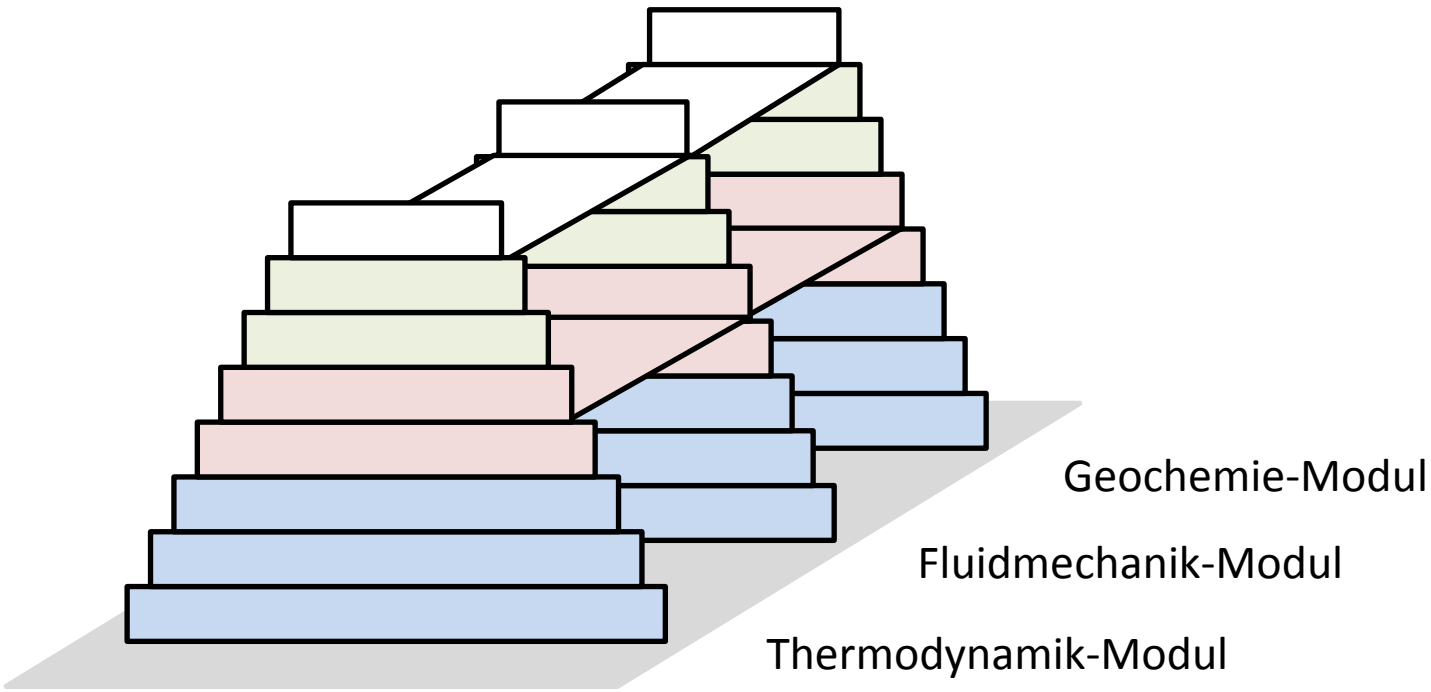
Tower of Approximations




```
Erdposition:
  r:   1.576923
  phi: 2.094395
```

Klimasimulation

Wall of approximations



Testansätze in Klimasimulationen

Vereinfachte Modelle

- Teilsystem isolieren
- Teilsystem mathematisch – wenn möglich – vereinfachen $\longrightarrow$ Analytische Modelle
- Numerisches Modell gegen dieses testen
- Nützlich, um Definitionsmenge numerischer Modelle zu evaluieren

Standardparameter

- Einheitliche Wahl von Testparametern unabhängig von einer Simulation
- z. B. ICRCCM: *Intercomparison of Radiation Codes in Climate Models program*
- Erlaubt Modellvergleiche
- Einschränkungen:
  - Einwirkung unspezifizierten Systemkomponenten vernachlässigt
  - örtlich begrenzt
  - zeitlich begrenzt

Dynamical Core Test

- *Schmetterlingseffekt*: Auswirkung von sich verändernden Anfangsbedingungen in dynamischen Systemen
- Standardisiertes Planetenmodell:
  - keine bzw. einheitliche Topographie
  - keine Jahreszeiten und Gezeiten
  - vereinfachte thermodyn. Faktoren
- Ergebnisse nach endlicher Anzahl simulierter Tage

Der Wasserplanet

- Dient der Untersuchung von Atmosphärenmodellen.
- Vereinfachung der Oberfläche:
  - keine Landmassen, lediglich Wasser
  - einheitliche Wassertemperatur

Evaluation der Simulation

- Simulierter Zeitraum: Jahre
- Gegebenes Verhältnis von Wasser, Land, Eis
- Gegebene Komposition der Atmosphäre: CO_2, O_3
- Anfangsbedingungen: Oberflächentemperatur, Sonneneinstrahlung
- AMIP – *Atmospheric Model Intercomparison Project:*
  - topographische Daten
  - Aerosolkonzentrationen
- Evaluation: Vergleich mit Klimaaufzeichnungen und anderen Modellergebnissen

Evaluation der Simulation

- Herausforderungen der Modell-Kalibrierung:
  - Parameteränderungen nicht linear zueinander
  - Änderungen der topographsichen Auflösung und damit numerischer Stabilität
  - Wahl geeigneter Simulationszeiträume (etwa 5-10 Jahre)
  - Bit-Flips
- Ansatz: Vergleiche nur zwischen inkrementellen Änderungen

Reanalysis climatologies

- Laufende Kalibrierung aus Echtzeitwetterdaten und kurzfristigen Prognosen (etwa 6h)
- Bezug der Daten von Wetterdiensten:
  - ECMWF: *European Centre for Medium Range Weather Forecasts*
  - NCEP*: National Centers for Environmental Prediction*
- Ermittlung nicht homogen möglich: Von dichten Bodenstationen hin zur beschränkten orbitalen Beobachtung
- Nach wie vor unberücksichtigte Faktoren bekannt (2002)

Double-call tests

> "This method runs a new or changed scheme along-side the old scheme. It uses fields directly from the rest of the model but does not feed back to the model. With this method, we can assess the change's direct impact. Normally, if we add a new scheme, indirect effects through feedbacks complicate the signal.
>
> – Pope, Davies. 2002. [4]

Spin-up tendencies

- Für Vergleiche zwischen Modellen oder Parametern
- Lasse hinreichend viele (60) kurze Simulationsläufe (1-5 sim. Tage) verschiedener Anfangszustände durchführen
- Vergleich der gemittelten *Hochschaukel*-Effekte der zu evaluierenden Elemente
- Nicht für träge Elemente geeignet

Wettervorhersagen

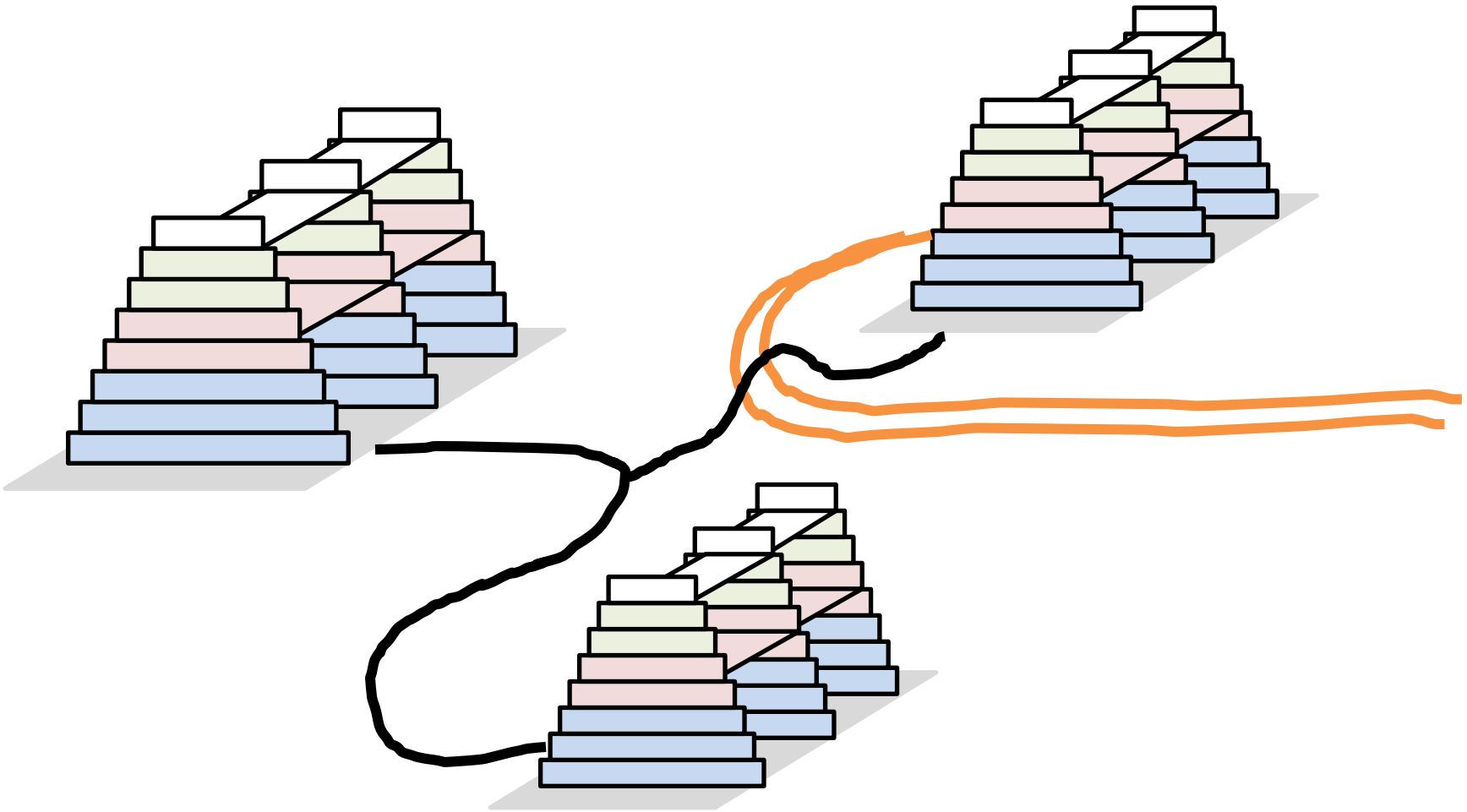
- Im Wesentlichen dasselbe Modell wie das zur langfristigen Klimasimulation
- Andere topographische Auflösungen, Unterordnung langfristiger Effekte
- Evaluation:
  - Parallelbetrieb der Modelle
  - Vergleich mit Beobachtungsdaten
  - Bestimmung der Fehlergröße

Klassische Programmcode-Tests?

ICON-Beispiel

```
364   !! energy_conversion_terms
365   !!
366   !! Purpose:
367   !!
368   !! @par Revision History
369   !! Separated from subroutine dyn and rewritten by Hui Wan (MPI-M, 2009-11-18)
370   !!
371   SUBROUTINE energy_conversion_terms( pt_diag, pt_patch, pt_int_state, &
372                                       p_mdiv, p_mdiv_int,              &
373                                       p_ddt_temp, p_ddt_vn,            &
374                                       opt_lseparate, opt_ddt_temp_fast )
375
376   !! Arguments
377
378   TYPE(t_hydro_atm_diag),INTENT(INOUT) :: pt_diag
379   TYPE(t_patch),TARGET,    INTENT(IN)    :: pt_patch
380   TYPE(t_int_state),       INTENT(IN)    :: pt_int_state
381
382   REAL(wp),INTENT(IN)    :: p_mdiv     (:,:,:)
383   REAL(wp),INTENT(IN)    :: p_mdiv_int (:,:,:)
384   REAL(wp),INTENT(INOUT) :: p_ddt_temp (:,:,:)
385   REAL(wp),INTENT(INOUT) :: p_ddt_vn   (:,:,:)
386
387   LOGICAL, INTENT(IN),   OPTIONAL :: opt_lseparate
388   REAL(wp),INTENT(INOUT),OPTIONAL :: opt_ddt_temp_fast(:,:,:)
389
390   !! Local variables
391
392   LOGICAL  :: lseparate
393   INTEGER  :: jk,jkp
394   INTEGER  :: jb,jbs,is,ie
395   INTEGER  :: nblks_e,nblks_c
396   REAL(wp) :: z2d (nproma,nlev)
397
398   REAL(wp) :: z_geo_mc( nproma,nlev,pt_patch%nblks_c )
399   REAL(wp) :: z_tv_c  ( nproma,nlev,pt_patch%nblks_c )
400   REAL(wp) :: z_tv_e  ( nproma,nlev,pt_patch%nblks_e )
401   REAL(wp) :: z_tvp_c ( nproma,nlev,pt_patch%nblks_c )
402   REAL(wp) :: z_tvp_e ( nproma,nlev,pt_patch%nblks_e )
403   REAL(wp) :: z_tmp_e ( nproma,nlev,pt_patch%nblks_e )
404   REAL(wp) :: z_tmp_c ( nproma,nlev,pt_patch%nblks_c )
405   REAL(wp) :: z_rlp_c ( nproma,nlev,pt_patch%nblks_c )
406   REAL(wp) :: z_fast  ( nproma,nlev,pt_patch%nblks_c )
```

Modellvergleich



CMIP5

- *Coupled Model Intercomparison Project Phase 5*
- Sammlung verschiedener Klimaexperimente:
  - historische Zeiträume
  - mittel- und langfristige Vorhersagen
  - zur Untersuchung von Zusammenhängen in Bezug auf die Atmosphäre
  - Spezifikationen von Grundmodellen, etwa des Wasserplanetens

Zusammenfassung

- Offene Systeme sind nicht verifizierbar
- Ständige Annäherung als best-approach
- Auf dem Weg zur Simulation verlieren wir Informationen
- Testverfahren für Validierung, Kohärenz
- Vergleichsgrundlagen zwischen Modellen

Literatur

- [1] N. Oreskes, K. Shrader-Frechette, K. Belitz. *Verification, Validation, and Confirmation of Numerical Models in the Earth Sciences*. In: *Science*, Vol. 263. 1994. S. 614ff
- [2] P. J. Roache. *Building PDE codes to be verifiable and validatable*. In: *Computing in Science Engineering*, Vol. 6. 2004. S. 30ff.
- [3] K. Hinsen. *The Approximation Tower in Computational Science: Why Testing Scientific Software Is Difficult*. In: *Computing in Science Engineering*, Vol. 17. 2015. S. 71ff
- [4] V. Pope, T. Davies. *Testing and evaluating atmospheric climate models.* In: *Computing in Science Engineering*, Vol. 4. 2002. S. 66ff