LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

# Environmental computing as a technical concept

Matti Heikkurinen[1] and Dieter Kranzlmüller[1,2]

Presentation at the WCDRR side event

[1]Ludwig-Maximilians-Universität (LMU), Munich, Germany
[2]Leibniz Supercomputing Centre (LRZ), Garching, Germany

## Why environmental computing?

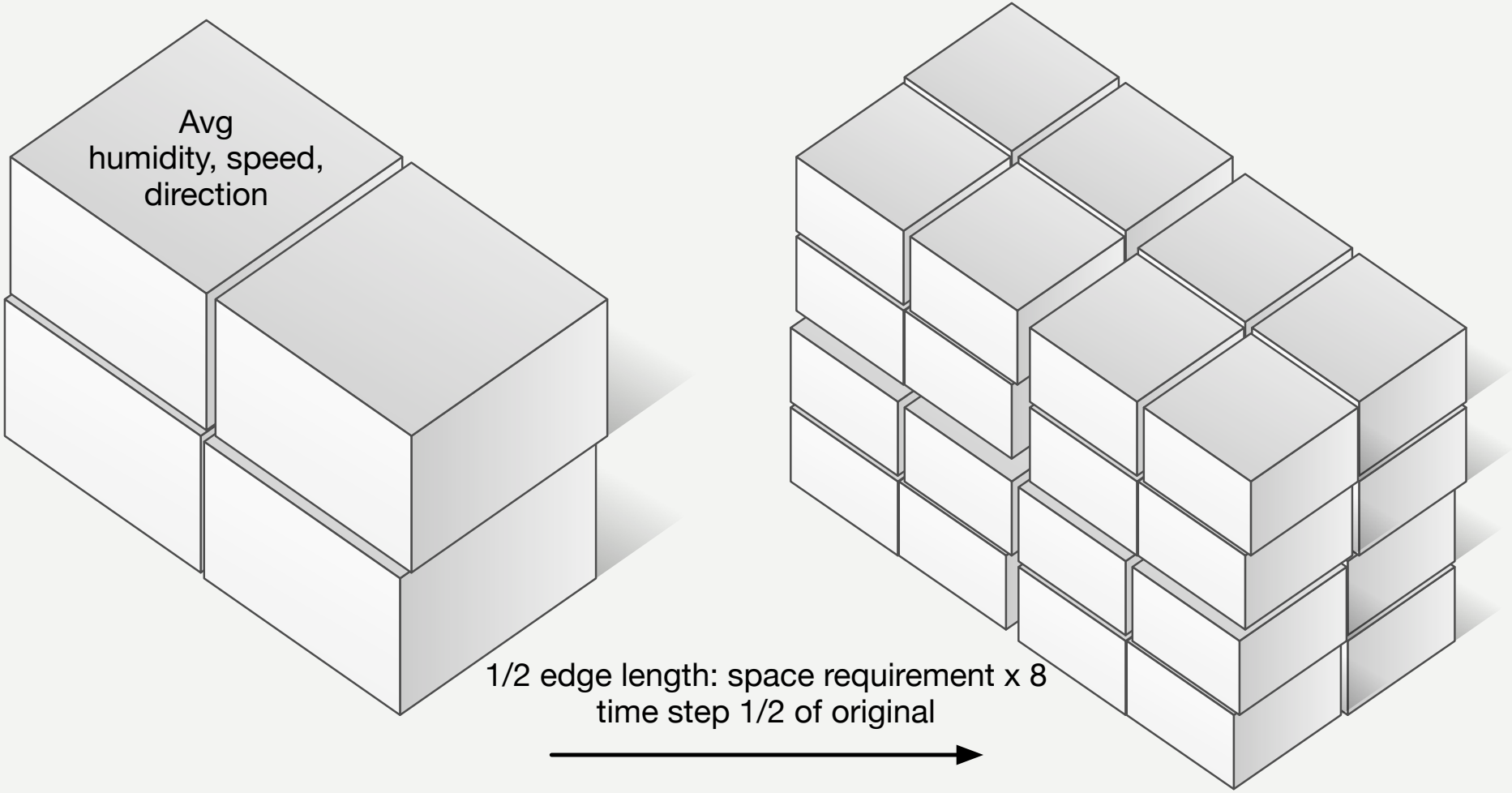
1. More accurate simulations effectively
2. Combine simulations in a more flexible manner
3. To understand what simulations tell (and don‘t tell)

## Accuracy

- Details matter
  - 500m vs. 5km radius of peak rainfall: boat vs. umbrella to work
  - Golf ball: dimples double the drive length
    - Climate change is reshaping and –sizing both „the ball“ and „the dimples“

Source: DoWs for DRIHM and DRIHM2US

## Granularity - interacting „boxes of air“




1/2 edge length: space requirement x 8
time step 1/2 of original

Computational requirements x 16!

## Cost of added detail

- The hypothetical „500m -> 50m: factor of 10 000 increase in requirements
  - „More hardware“: straightforward as long as „air boxes“ fit in memory, and
  - Input data quality is sufficient, and
  - You don‘t run into trouble with rounding errors

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

# Even with enough useful computing capacity, You can still run into a wall

## Why we need mutli-model systems

- Floods: complex interplay between
  - Topography (and changes in topography – landslides, sediments)
  - Soil type and saturation
  - Human activtiy
- Similar situations also in
  - Seismic activity and its impacts
  - Using environmental modelling as input for other models
    - Epidemiology
    - Agriculture
    - ...
- Plugging in new model components
  - Compare model with whole system behaviour

## Pitfalls

- Multi-model system may mask errors
  - One of the components failing might still give (roughly) correct result
- Interface issues
  - Misinterpreting the data between models
  - Accessing HPC (efficiently!)
- Performance
  - Different models may be optimised for different kind of software and hardware infrastructures

## Some things multi-model systems need

- Documented interfaces
  - And common agreements on how they are documented
- Documented behaviour
  - Parameter ranges where preductions are accurate
  - System requirements
- Data (with documented structure)

# Modularity, predictable system-level behaviour?