PARALLEM: massively Parallel Landscape Evolution Modelling

Tuesday 28th November 2017

The University of Sheffield

A. Stephen McGough, Darrel Maddy

J. Wainwright, S. Liang, M. Rapoportas, A. Trueman,

R. Grey, G. Kumar Vinod, and James Bell

Outline

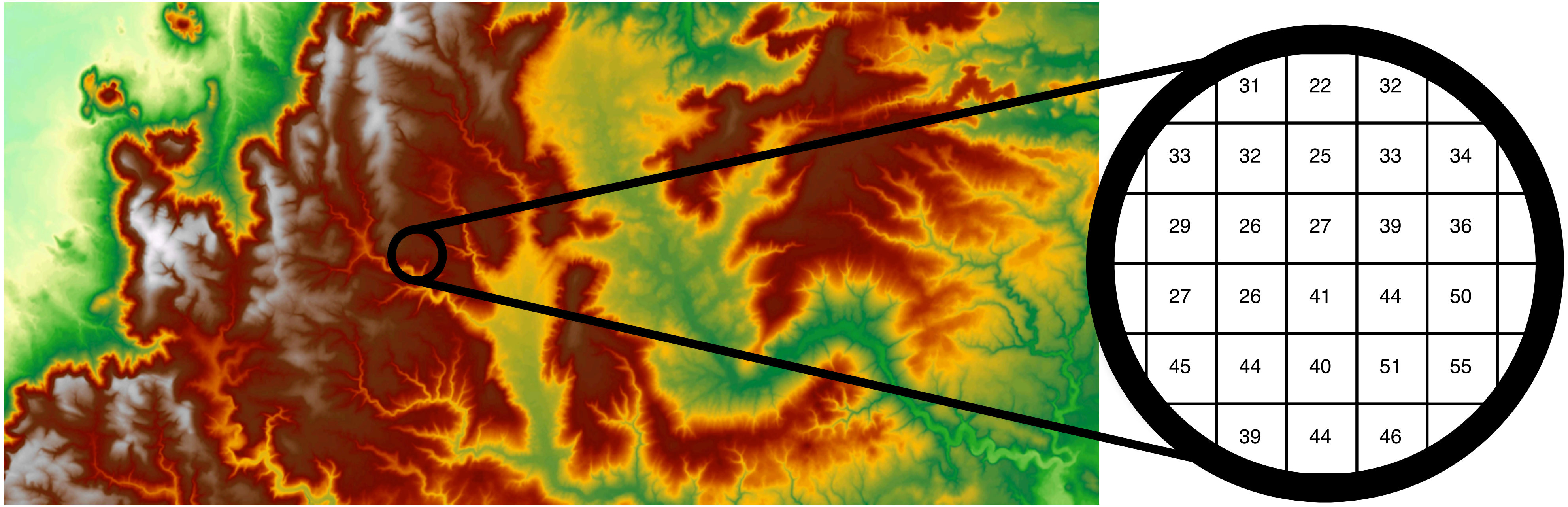
- **What is Landscape Evolution Modelling (LEM)**
- Parallelization of LEM
- Preliminary Results
- The Current Situation
- Future Directions

Landscape Evolution Modeling

- Landscapes change over time due to water/weathering
  - Physical and Chemical Weathering require water to break down material
  - Higher energy flowing water both Erodes and Transports material until decreasing energy conditions result in Deposition of material
- These processes take a long time
  - Many glacial-Interglacial Cycles
    - Cycles are ~100ka for last 800ka, prior to 800ka cycles were ~40ka in length
- We want to use retrodiction to work out how the landscape has changed

Landscape Evolution Modeling

- Use a simulation to model how the landscape changes
  - 3D Landscape is discretized as a regular 2D grid (x, y) with cell values representing surface heights (z) derived from a digital elevation model (DEM)
    - Cells can be 10m x 10m or larger



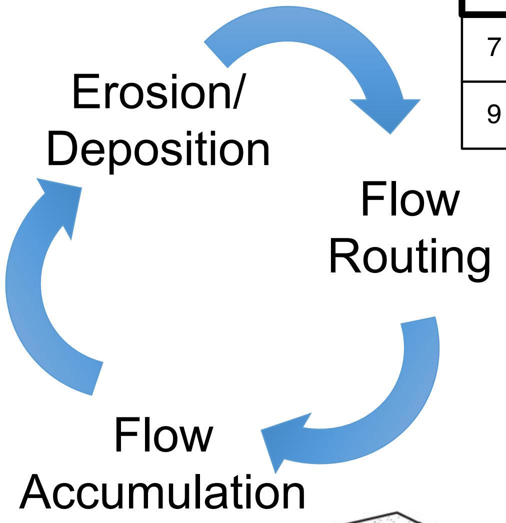
Landscape Evolution Modeling (simplified)

Each iteration of the simulation:

How much material will be removed?
How much material will be deposited?

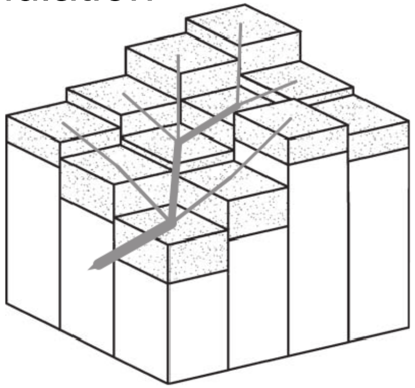
7	8	5	9	6
4	6	7	8	4
5	8	9	10	9
7	10	7	8	7
9	8	4	6	5

- Each step is ‘fairly’ fast…
- But we want to do lots of them 120K to 1M years
- On landscapes of 6-56M cells
- If we could simulate 1 year in 1 minute this would take 83 – 694 days!
  - assuming 1 year = 1 iteration
  - may need more



Sequential version is much slower than this…

1	1	3	1	1
8	2	1	1	5
1	1	1	1	1
2	1	1	1	1
1	1	6	1	2

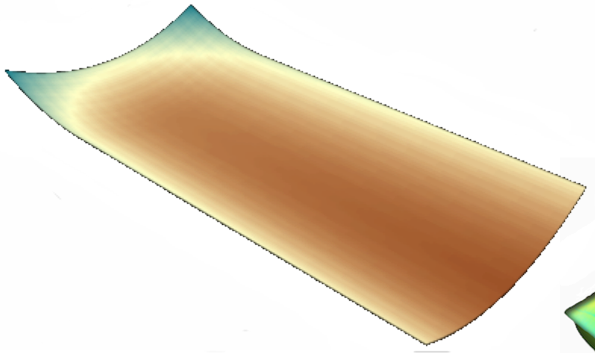


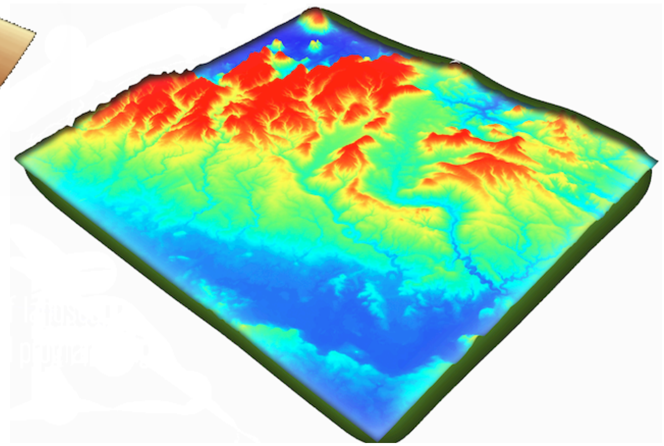
Execution analysis of Sequential LEM

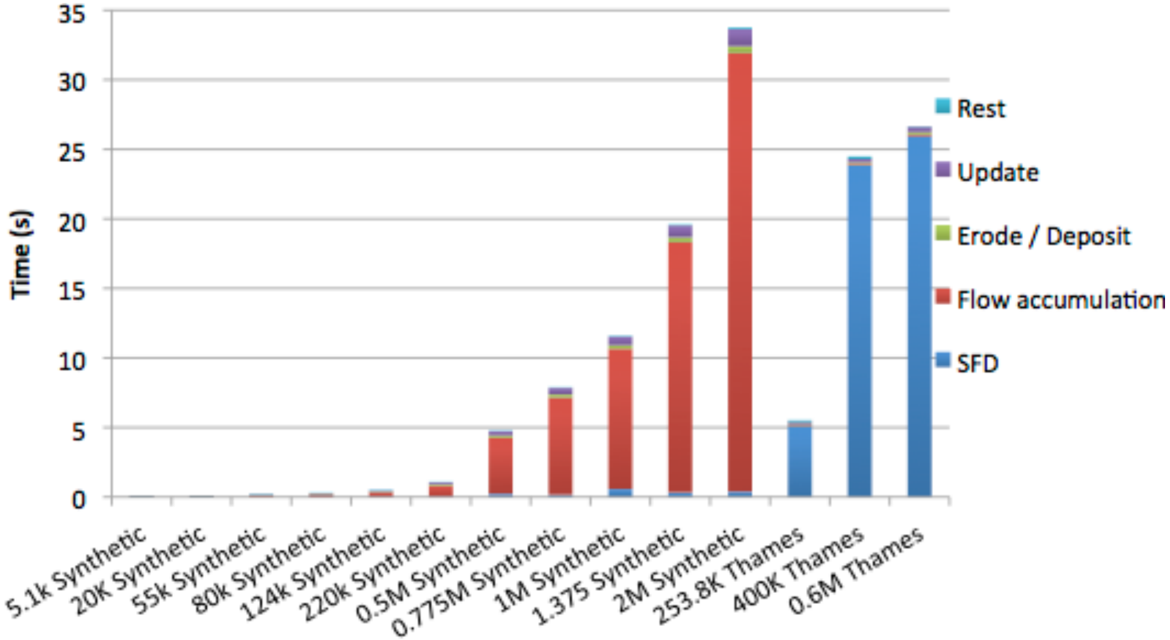
- We started from an existing sequential LEM
  - 51x100 cells for just 120K years took 72 hours
    - estimate for 25M cells 64,000 years
  - This was non-optimal code
    - Reduced execution time from 72 to 4.7 hours
    - 64,000 years down to 300 years
- But this is still not enough for our needs

Execution analysis of Sequential LEM

- Performance Analysis:
- ~74% of time spent routing and accumulating
- Need orders of magnitude speedup
  - So focus was on flow routing / accumulation





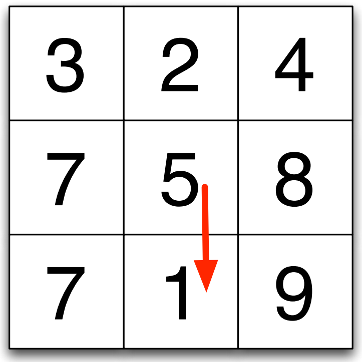


Outline

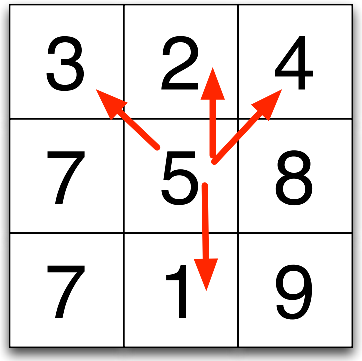
- What is Landscape Evolution Modelling (LEM)
- **Parallelization of LEM**
- Preliminary Results
- The Current Situation
- Future Directions

Parallel Flow Routing

- Each cell can be done independently of all others
  - SFD
    - 100% flow in the direction of steepest decent (normally lowest neighbour)



  - MFD
    - Flow is proportioned between all lower neighbours
    - Proportional to slope to each neighbours



- Almost linear speed-up
  - Problems with code divergence
    - CUDA Warps split when code contains a fork

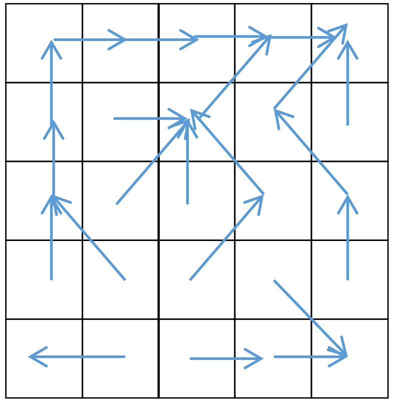
Single flow direction vs multiple flow direction

MFD is 'better' but much more computationally demanding

Parallel Accumulation: Correct Flow

- Iterate:
  - Do not compute a cell until it has no incorrect cells flowing into it
  - Sum all inputs and add self
  - All cells can work independently of each other
    - Some restriction on updates not happening immediately

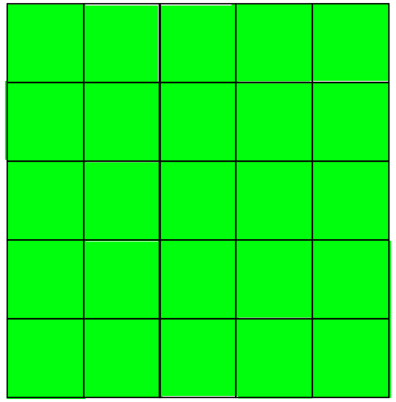
Flow Routing



Accumulation

5	6	7	14	19
4	1	6	3	1
3	1	1	2	2
1	1	1	1	1
2	1	1	2	4

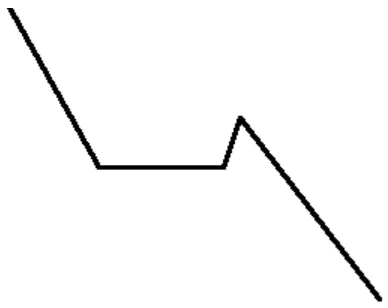
Correct

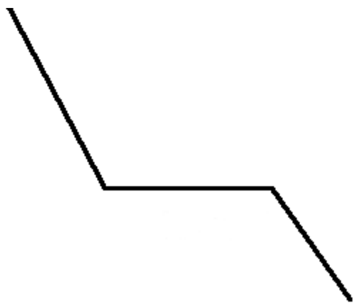


Cell values are not normally 1, but the initial rainfall on the cell

Not the whole story…

- Sinks and Plateaus

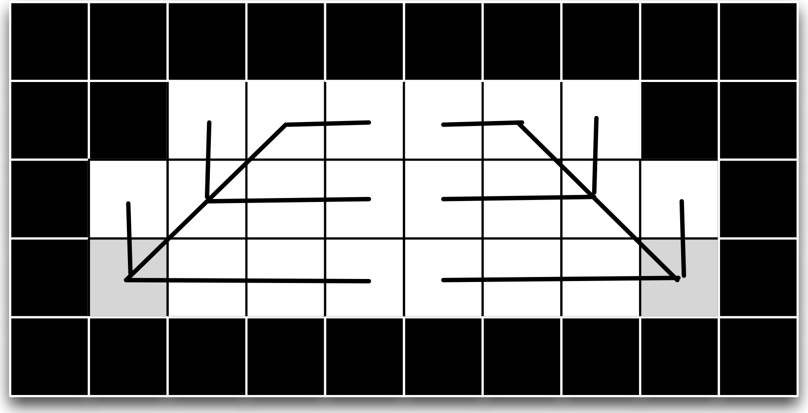


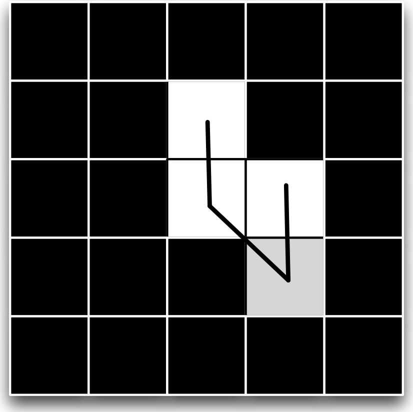


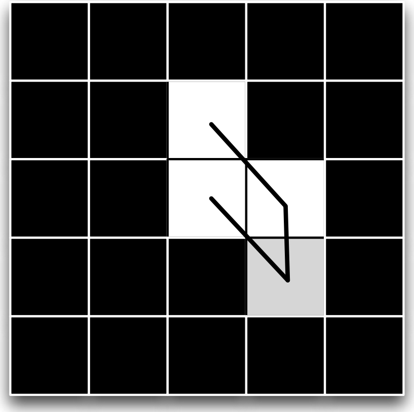
- Can’t work out flow routing on sinks and plateaus
- Need to ‘fake’ a flow routing
  - Fill a sink until it can flow out
    - Turn it into a plateau
  - Fake flow directions on a plateau to the outlet

Parallel Plateau routing

- Need to find the outflow of a plateau and flow all water to it
- A common solution is to use a breadth first search algorithm
  - Parallel implementation
  - Though result does look ‘unnatural’
  - Alternative patterns are possible – but acceptable
- We are investigating alternative solutions

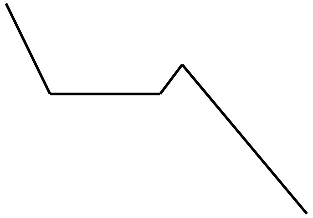


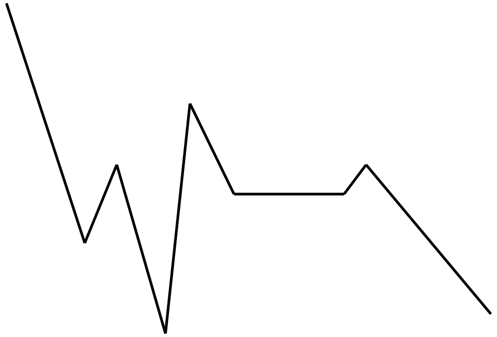




Sink filling

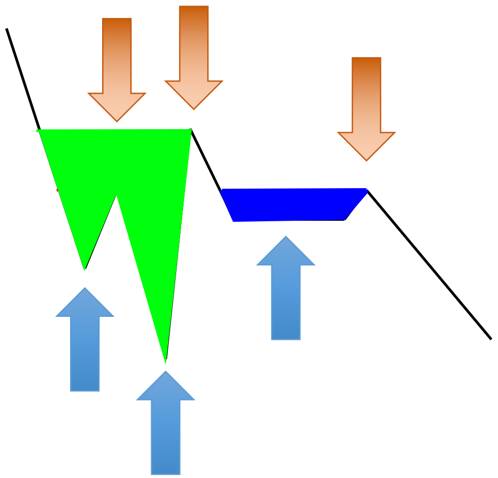
- Dealing with a single sink is (relatively) simple
  - Fill sink until we end up with a plateau (lake)
- But what if we have multiple nested sinks?





Nested Sink filling

- Implemented parallel version of the sink filling algorithm proposed by Arger et al [2003]
  - Identify each sink (parallel)
  - Determine which cells flow into this sink - watershed (parallel)
  - Determine the lowest cell joining each pair of sinks (parallel/sequential)
  - Work out how high cells in each sink need to be raised to to allow all cells to flow out of the DEM (sequential)
  - Fill all sink cells to this height (parallel)

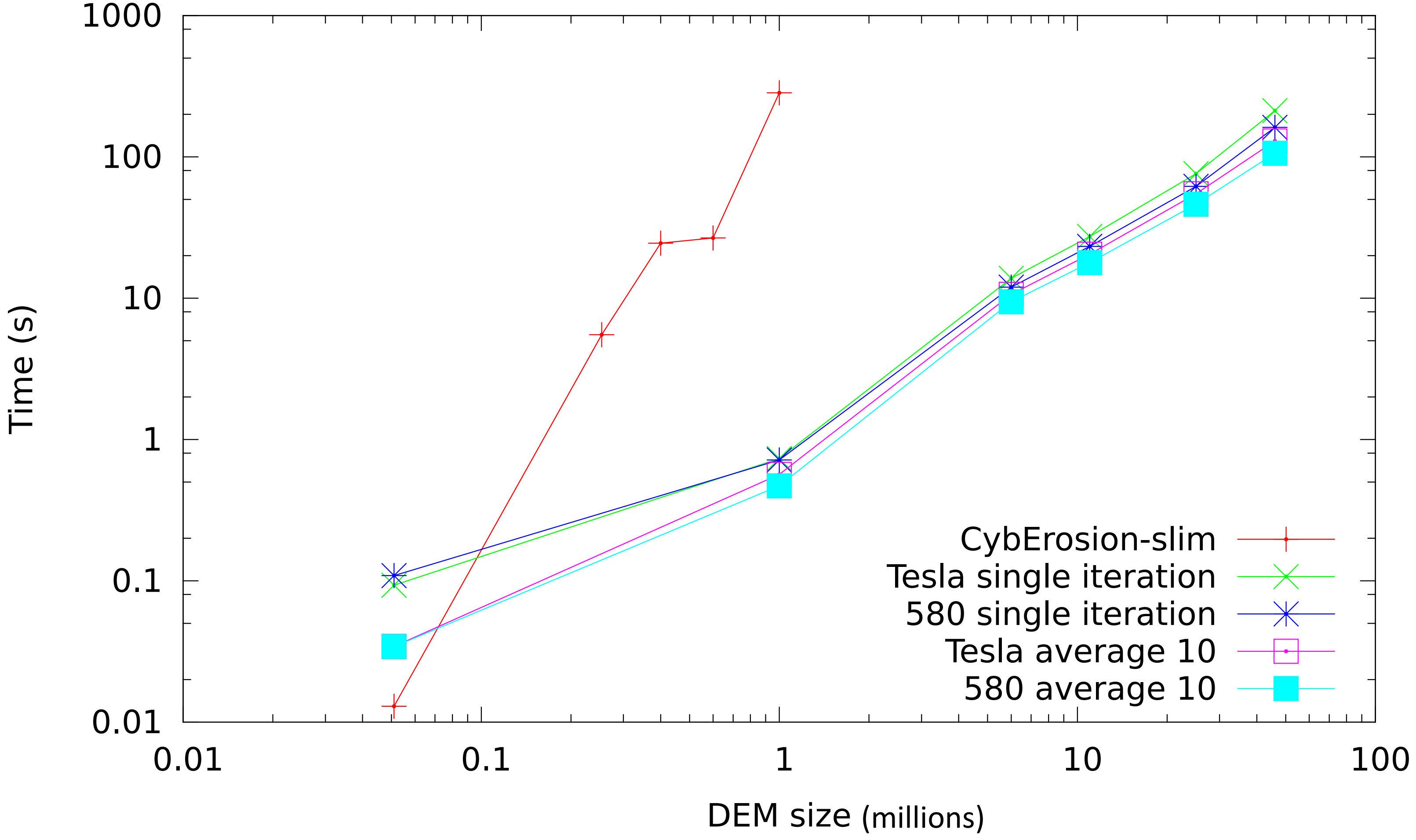


Outline

- What is Landscape Evolution Modelling (LEM)
- Parallelization of LEM
- **Preliminary Results**
- The Current Situation
- Future Directions

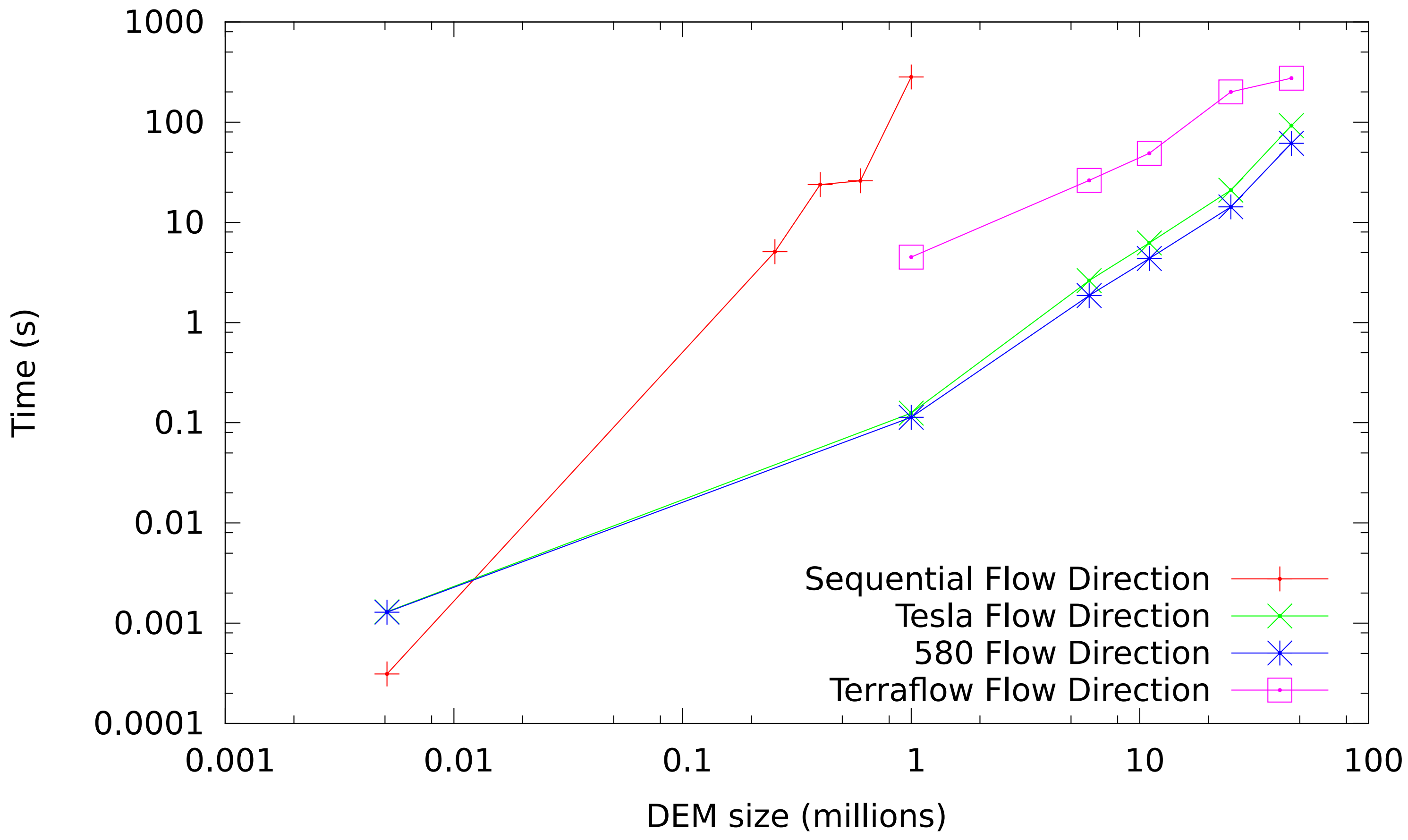
Results : Performance

- Overall performance



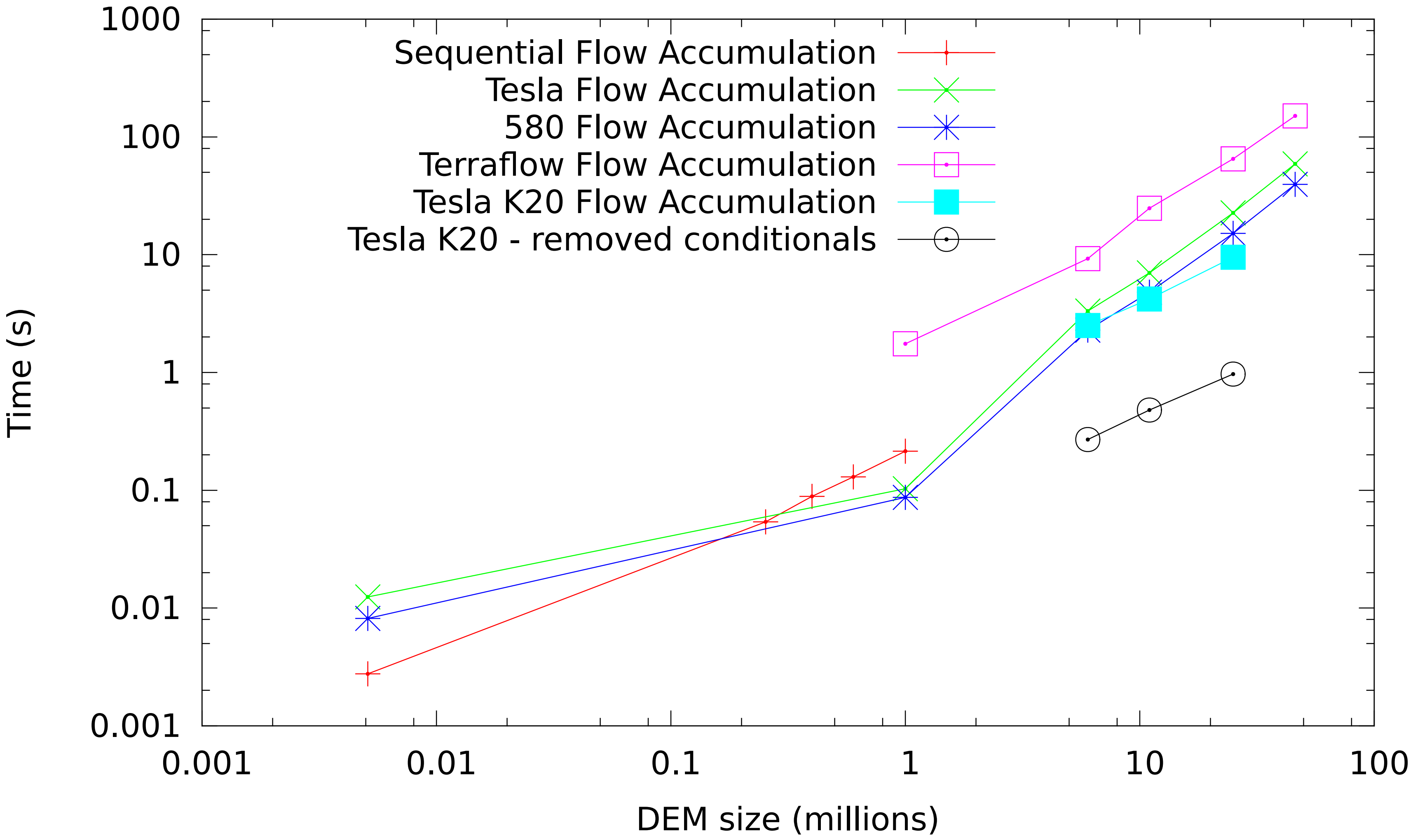
Results : Performance

- Flow Direction
  - Including sink & plateau solution



Results : Performance

- Flow Accumulation

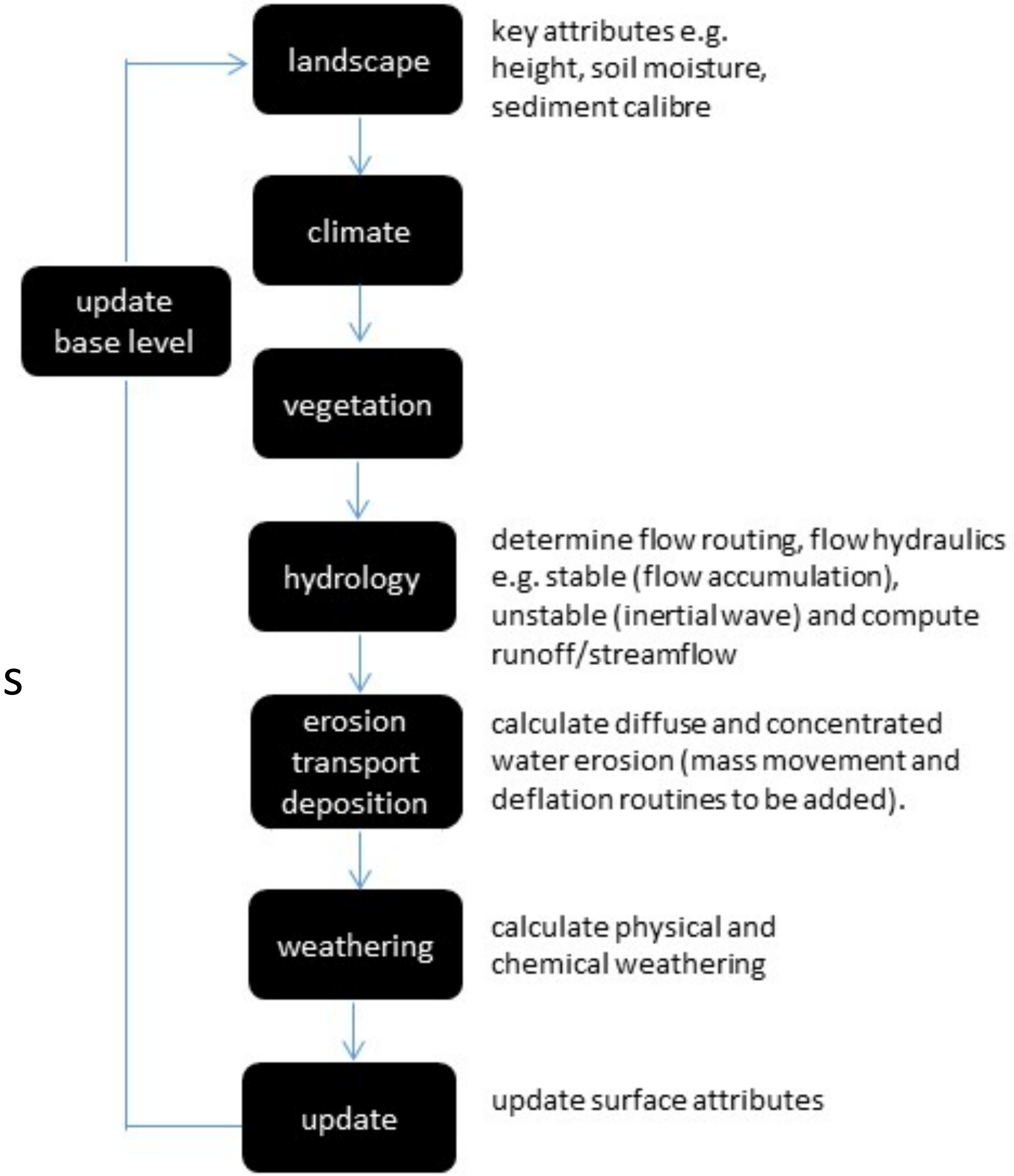


Outline

- What is Landscape Evolution Modelling (LEM)
- Parallelization of LEM
- Preliminary Results
- **The Current Situation**
- Future Directions

The Current Simulation

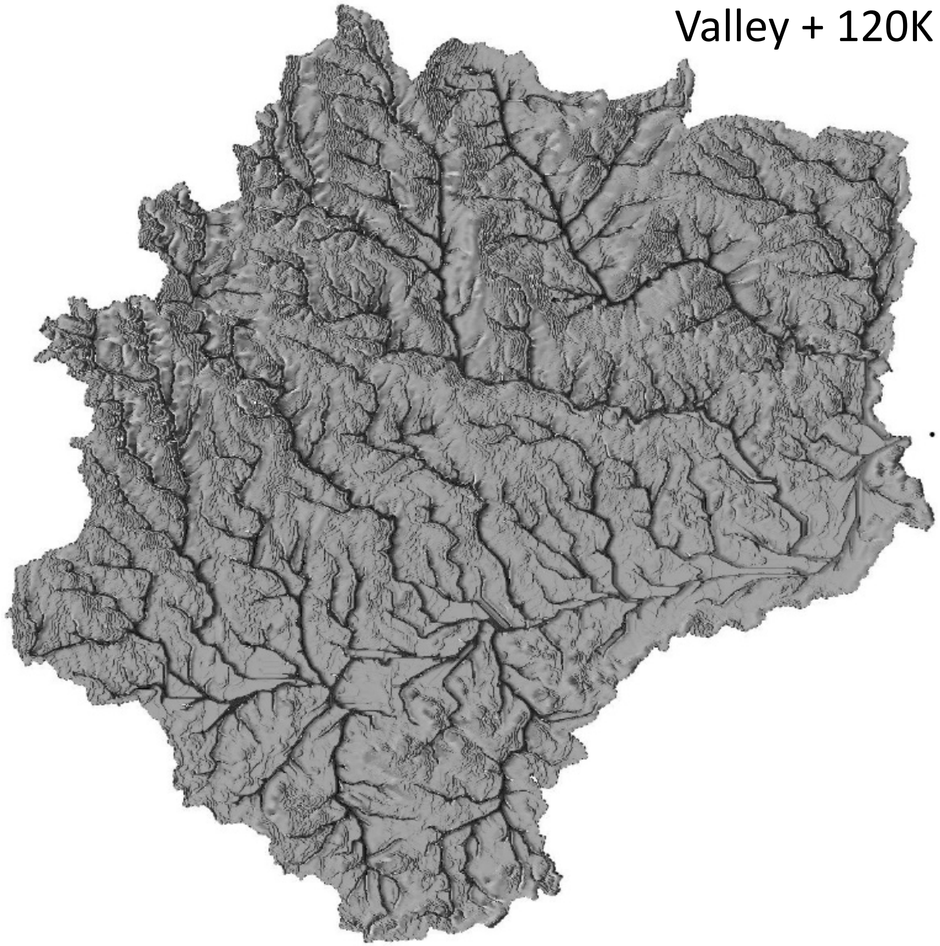
- Core Model now extended with processes
  - Most only affect individual cells (weathering, vegetation)
  - Some have cross DEM effects (mass movement) but can use same process as before



The Current Simulation

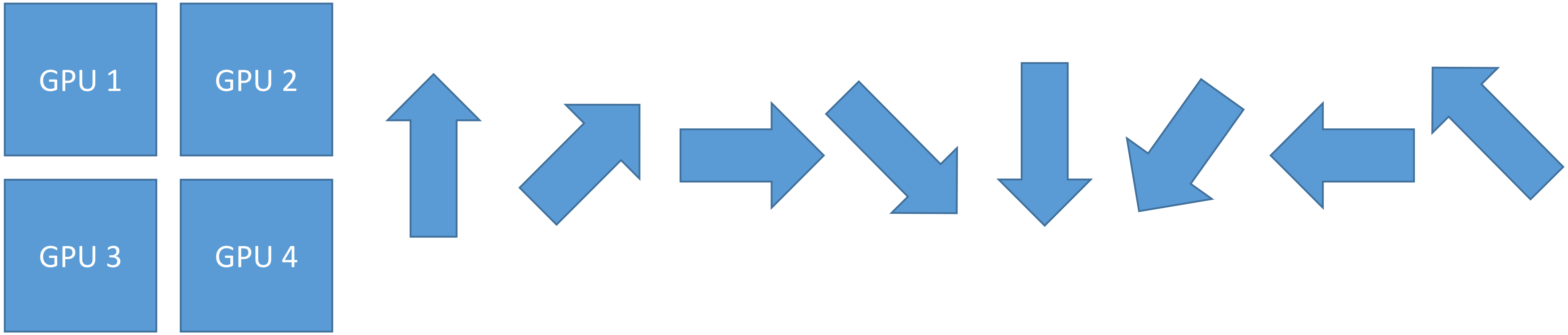
- Actively running landscape models on K40/K80 GPGPUs
- Taking ~7 weeks to run our model (MFD)
  - Leading to interesting results
  - Not seen as models have traditionally been much smaller
  - Taking ~4 weeks for SFD
- Currently running on just 1 GPGPU
  - Running multiple models simultaneously
  - Now have a multi-GPGPU code for running flow accumulation
    - Designed to ‘sweep’ over the landscape

Upper Thames Valley + 120K



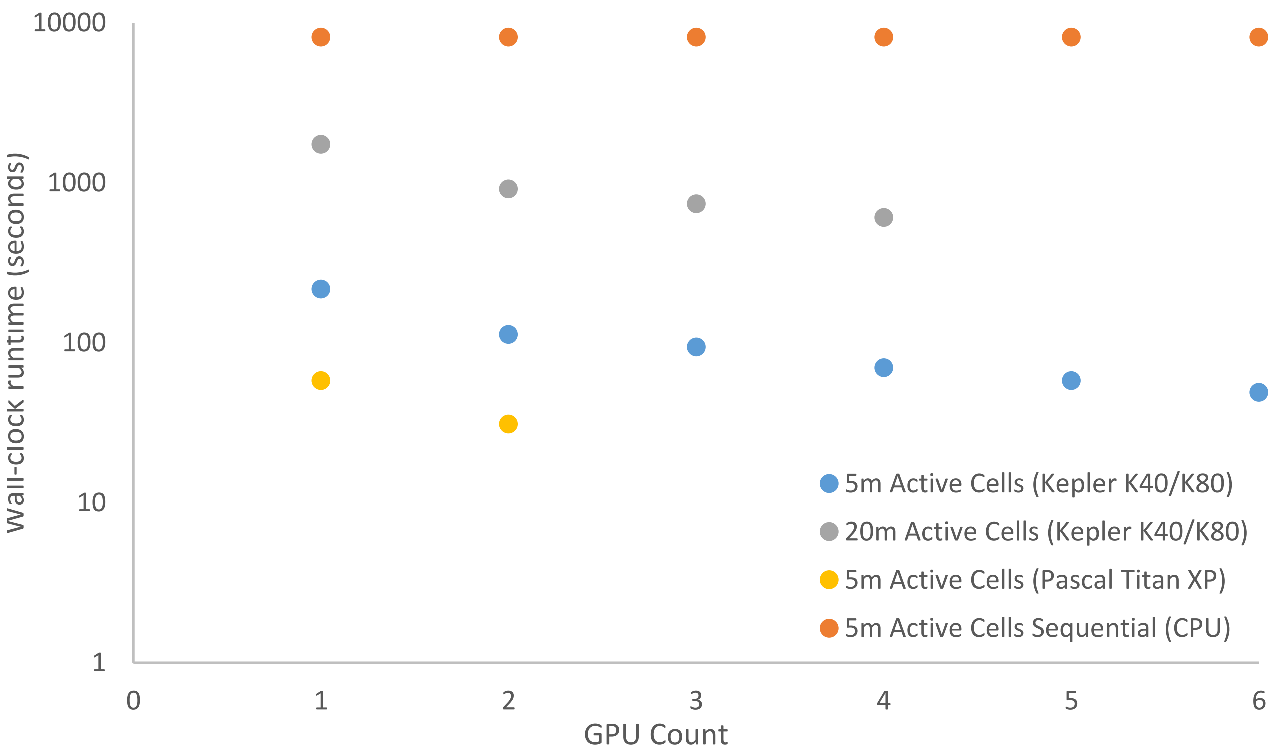
Multi-GPU: Attempt 1

- Flow direction can be done without problems
- Flow accumulation requires communication
- Perform each flow direction as one kernel call
  - No branching
  - Communication easier between cards

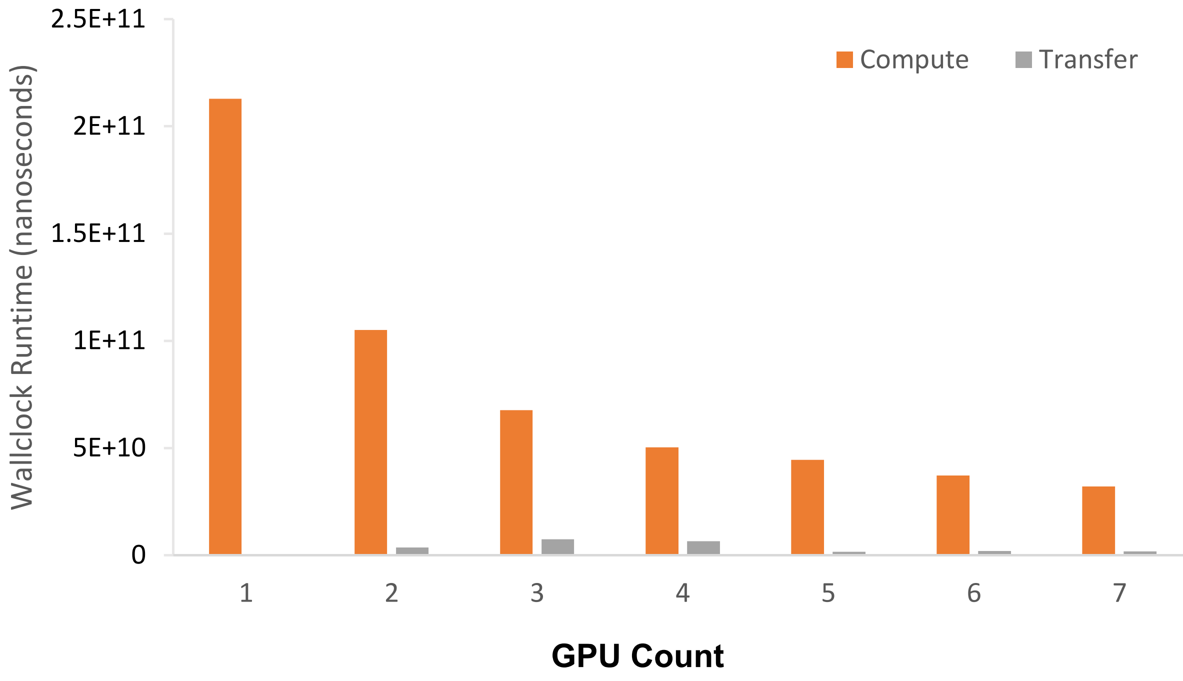


Multi-GPU: Attempt 1

Whole Simulation

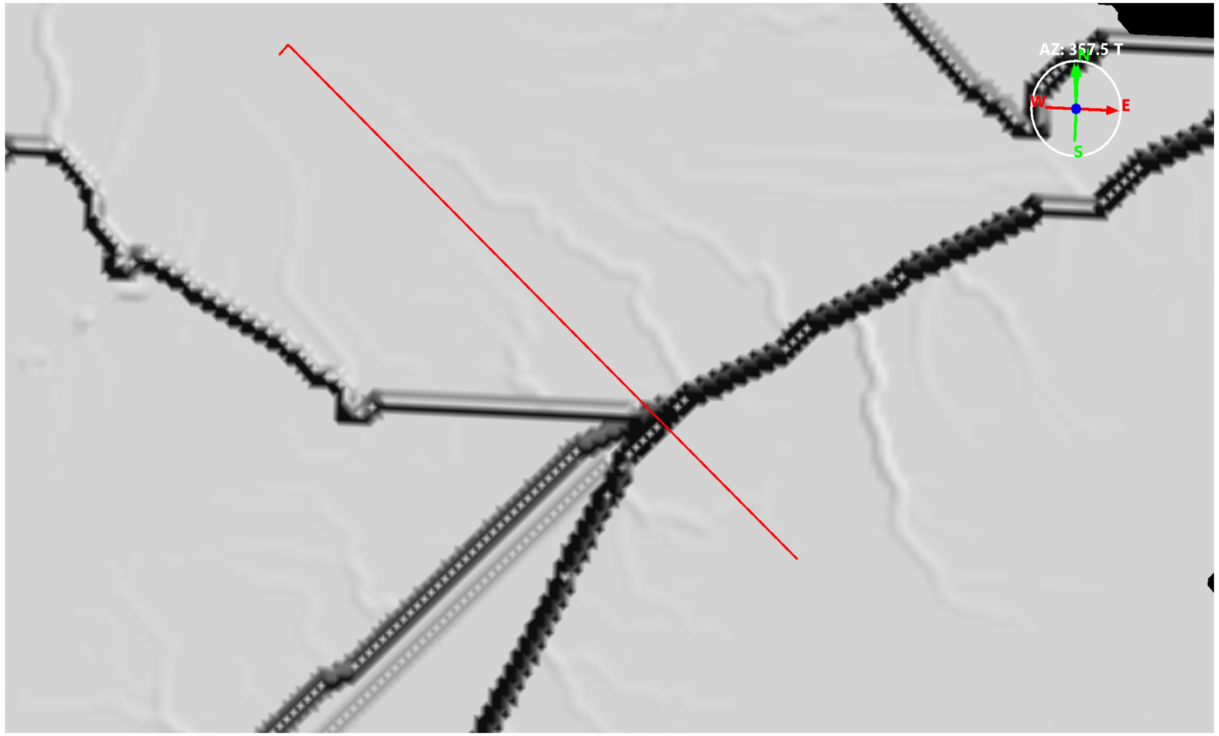


Flow Accumulation

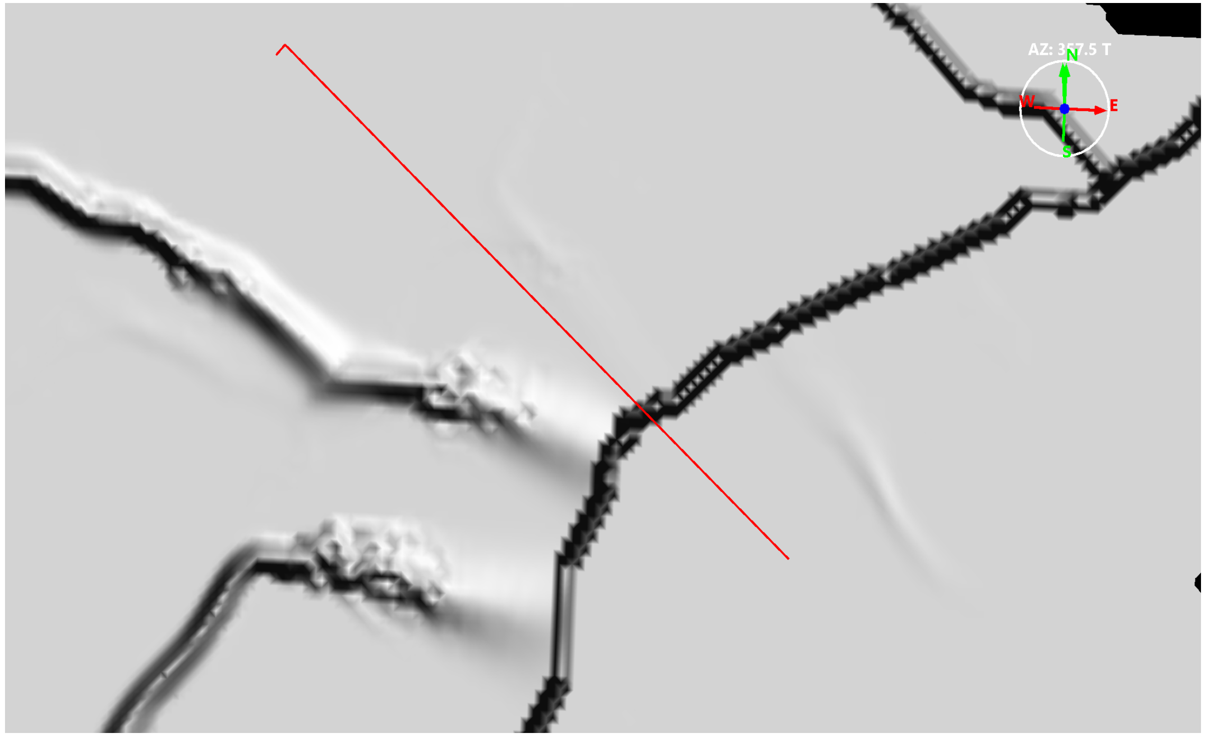


Problem: Landscape Cutting with SFD

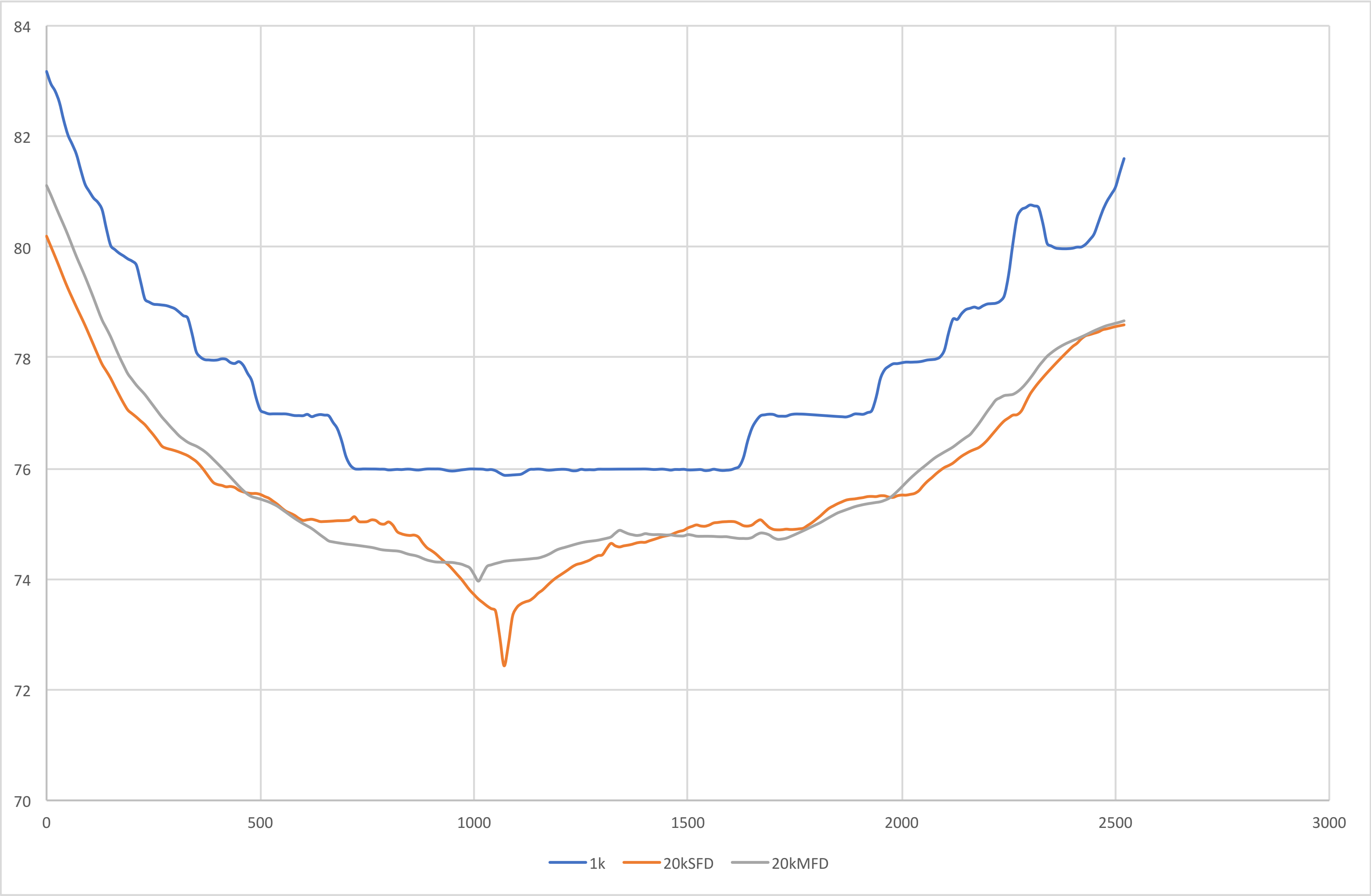
SFD



MFD

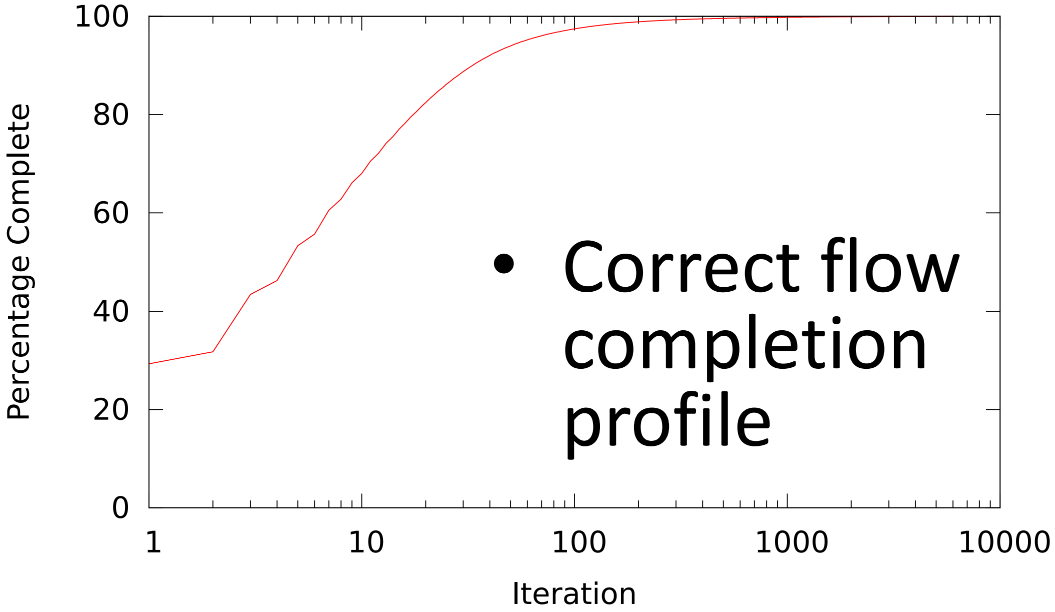


Comparing ‘cut in’ between SFD and MFD

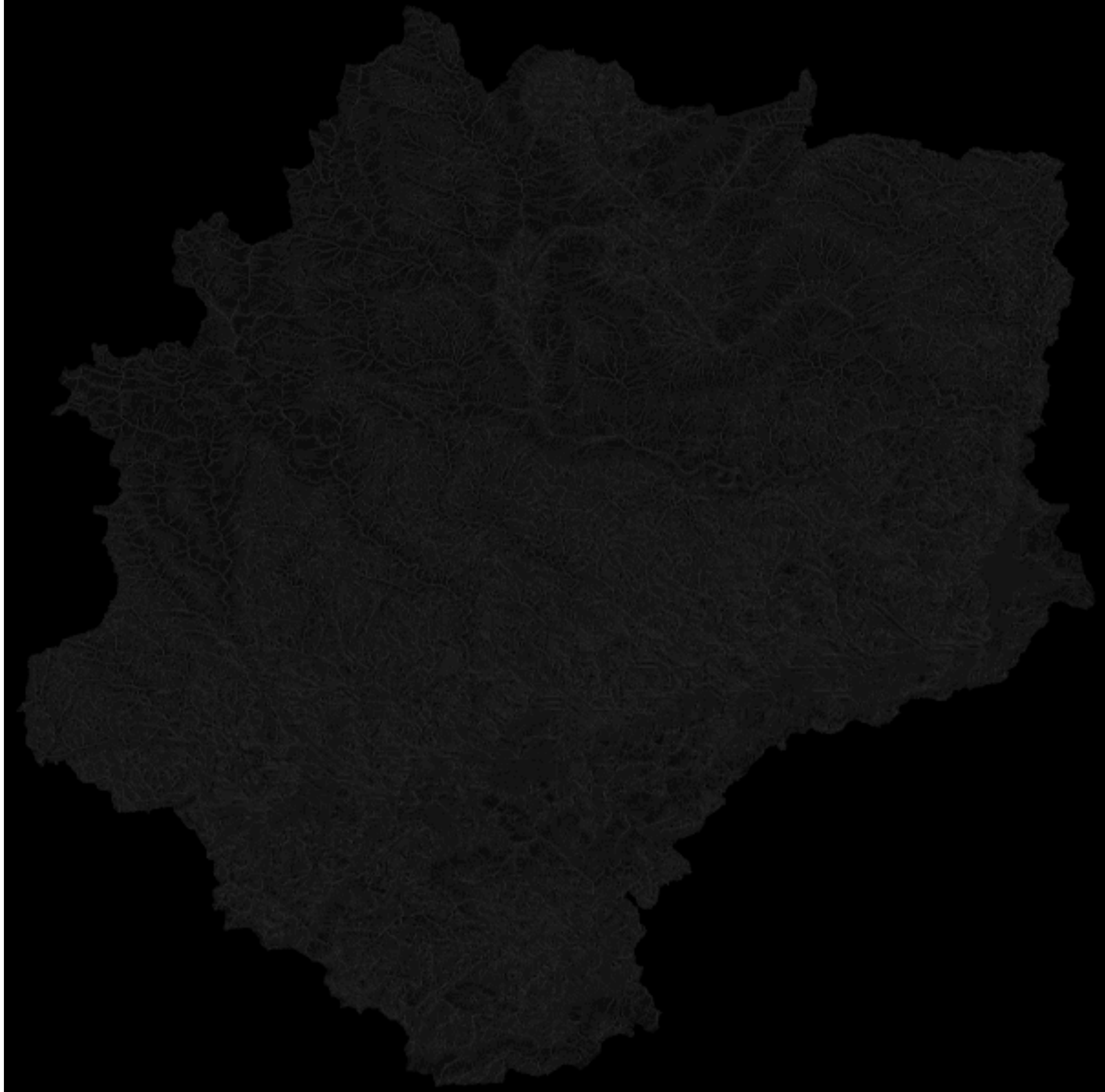


Problem: Algorithm Slow-down

- Correct flow algorithm requires all input cells to be correct before progressing
- Becomes a problem for rivers



- Correct flow completion profile



Outline

- What is Landscape Evolution Modelling (LEM)
- Parallelization of LEM
- Preliminary Results
- The Current Situation
- **Future Directions**

Process Improvements

- Smaller cells lead to greater depth of erosion
  - Rivers are currently only one cell wide
  - Make rivers wider (multi-cell)
- Modification of process algorithms to allow for lateral erosion

IAPETUS
Doctoral Training Partnership
One potential PhD position to work on this

Summary

- Able to show 2+ orders of magnitude speedup in PARALLEM
- Significant potential for further speedup
  - Optimization of the processes
  - Remove sequentialization of correct flow
- The use of GPGPUs has allowed us to redress the execution restriction which has prevented us doing MFD – leading to ‘better’ landscapes

IAPETUS
Doctoral Training Partnership
One potential PhD position to work on this

We Are recruiting:
- 2 PostDoc (Machine Learning)
- Always looking for good PhD Candidates

stephen.mcgough@newcastle.ac.uk
darrel.maddy@newcastle.ac.uk
J. Wainwright, S. Liang, M. Rapoportas,
A. Trueman, R. Grey, G. Kumar Vinod,
and James Bell

Newcastle University