

Fluid simulations accelerated with 16 bits

Approaching 4x speedups on A64FX by squeezing ShallowWaters.jl into Float16

Milan Klöwer¹, Sam Hatfield², Matteo Croci³,
Peter Düben², Tim Palmer¹

¹University of Oxford, UK

²European Centre for Medium-Range Weather Forecasts, UK

³University of Texas at Austin, USA

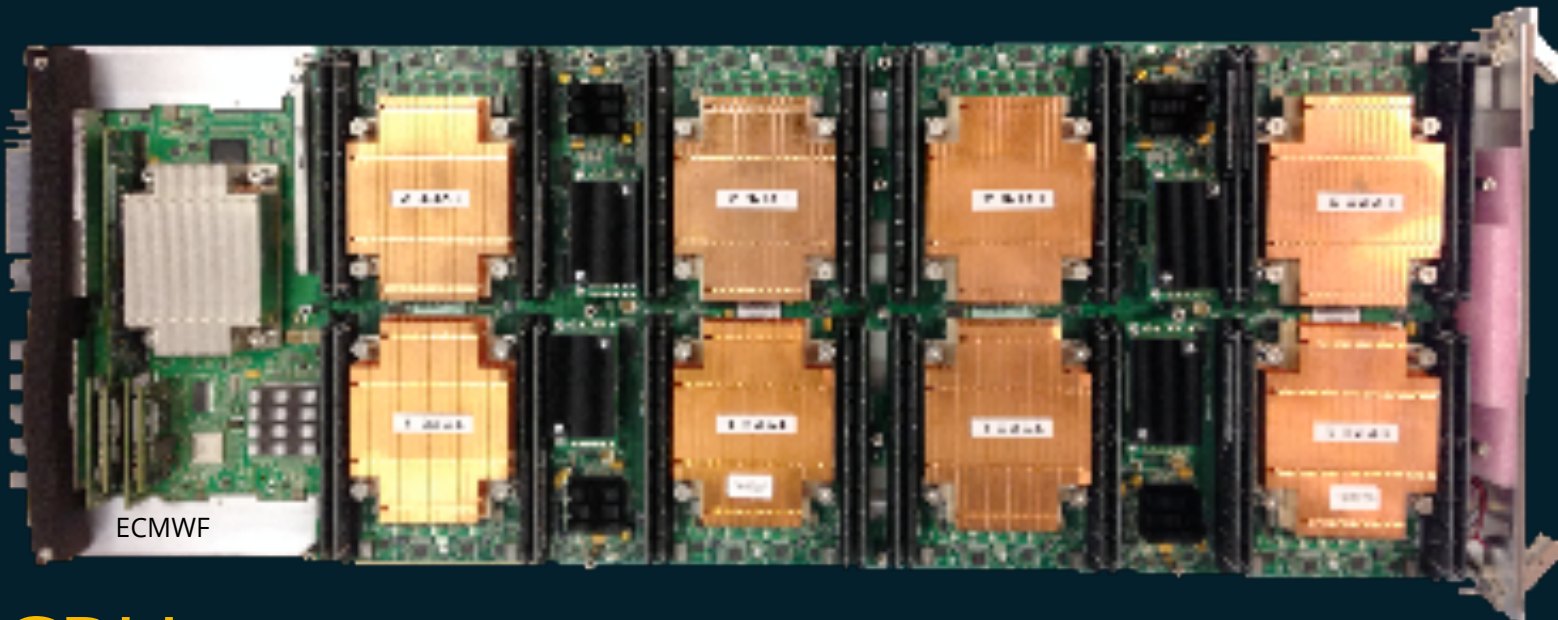


Hardware gets really fast in 16 bits!

CPU → GPU

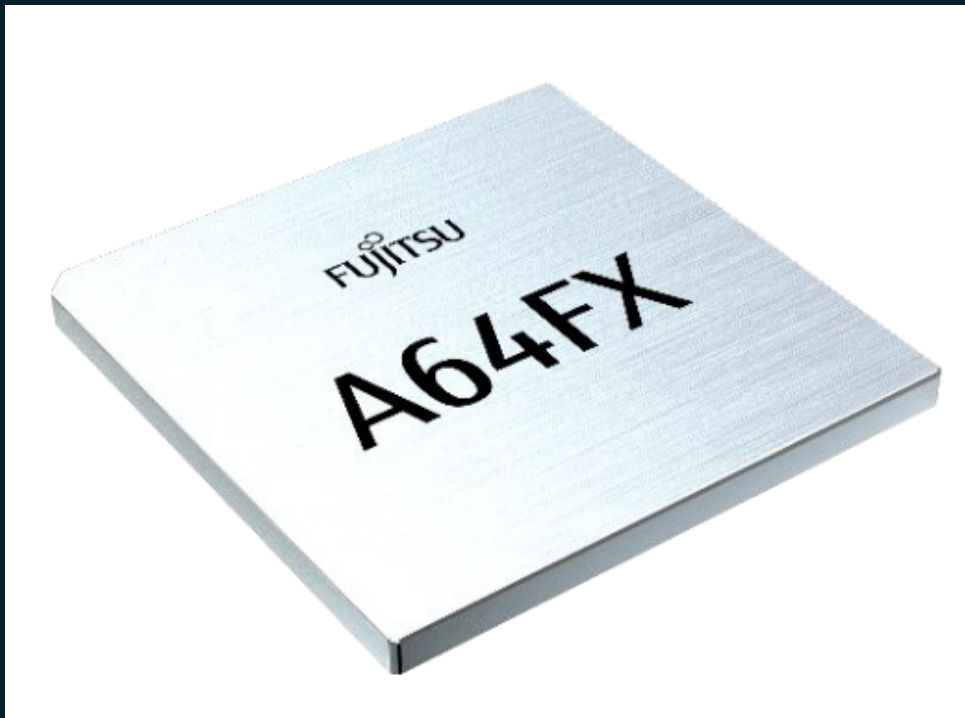
Nvidia A100 GPU

Cray XC40 compute blade, 8xCPU

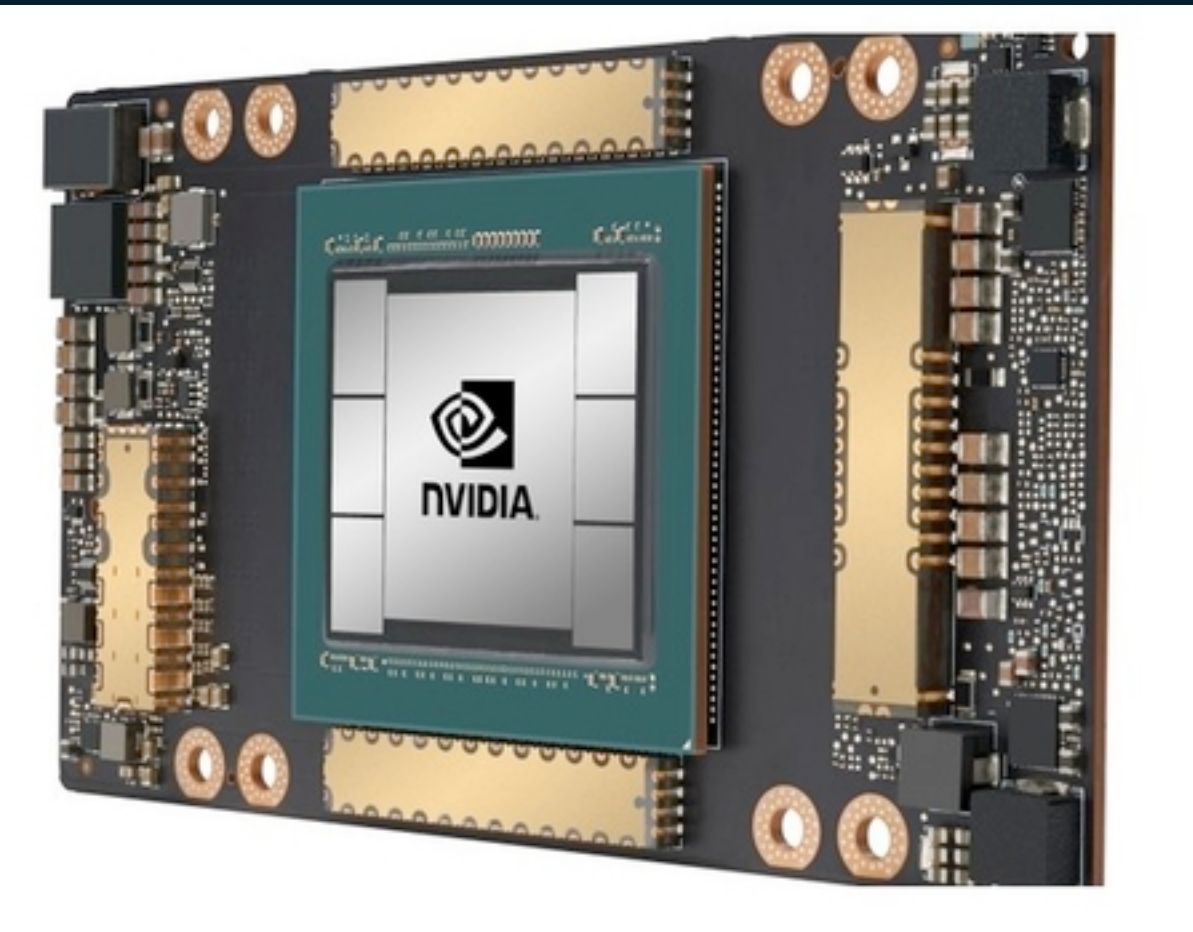


Each CPU	18 cores
Energy	145W
64-bit performance	1 TFLOP/s
Number formats	Float64, Float32
	1x, 2x speedup

Fujitsu A64FX



48 cores
180 W
3 TFLOP/s
Float64, Float32, Float16
1x, 2x, 4x speedup



6912 cores
200-400W
10 TFLOP/s
Float64, Float32, Float16, BFloat16, TFloat32
16-bit TensorCore: 32x speedup

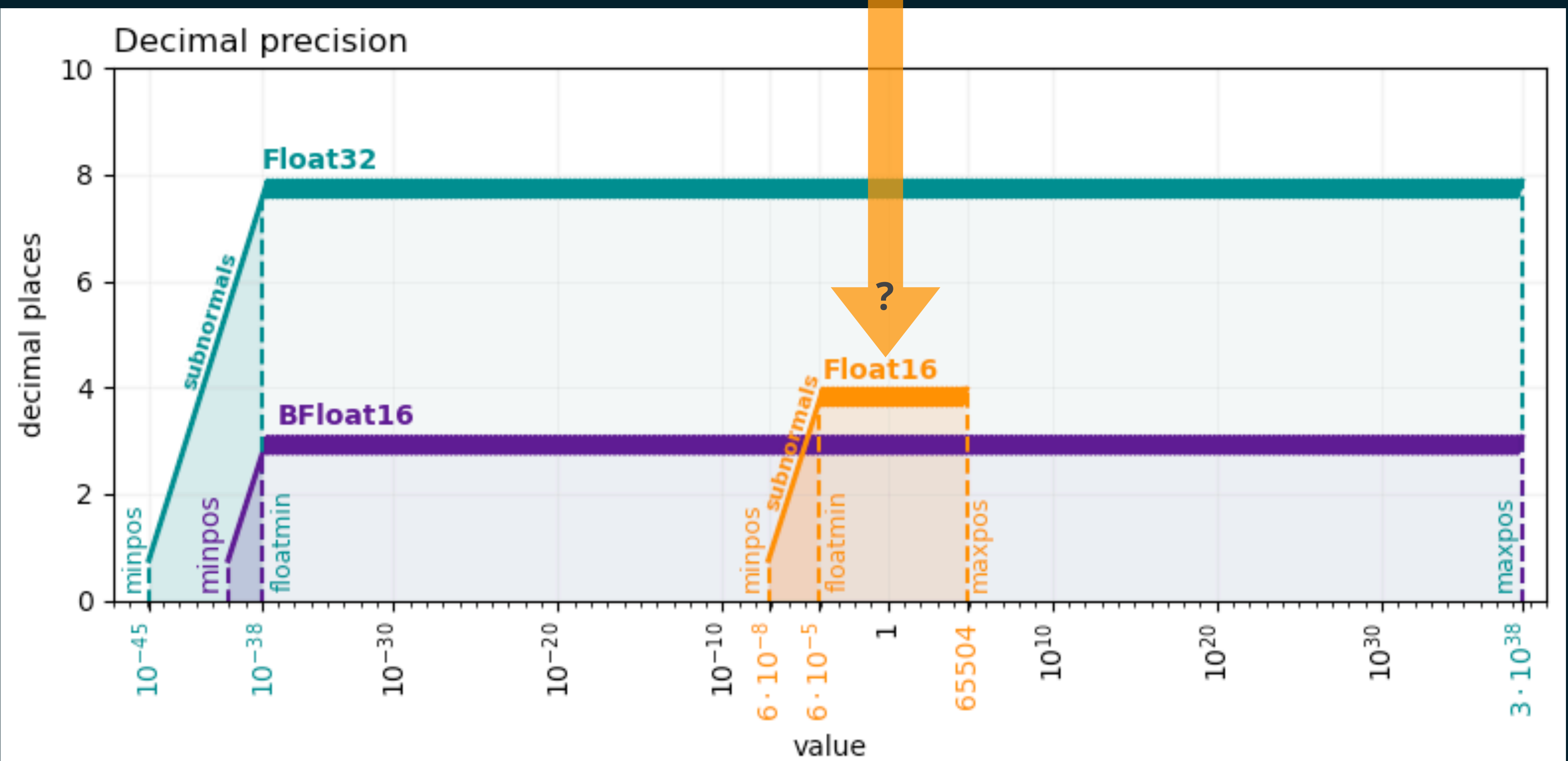
Float16: Is it enough?

Equations to be solved numerically

$$\partial_t \mathbf{u} + \mathbf{u} \cdot \nabla \mathbf{u} + f \mathbf{z} \times \mathbf{u} = -g \nabla \eta + \nu_B \nabla^4 \mathbf{u} - r \mathbf{u} + \mathbf{F}$$

$$\partial_t \eta + \nabla \cdot (\mathbf{u} h) = 0$$

$$\partial_t q + \mathbf{u} \cdot \nabla q = -\tau(q - q_0)$$



Allowed manipulations to change the range of numbers

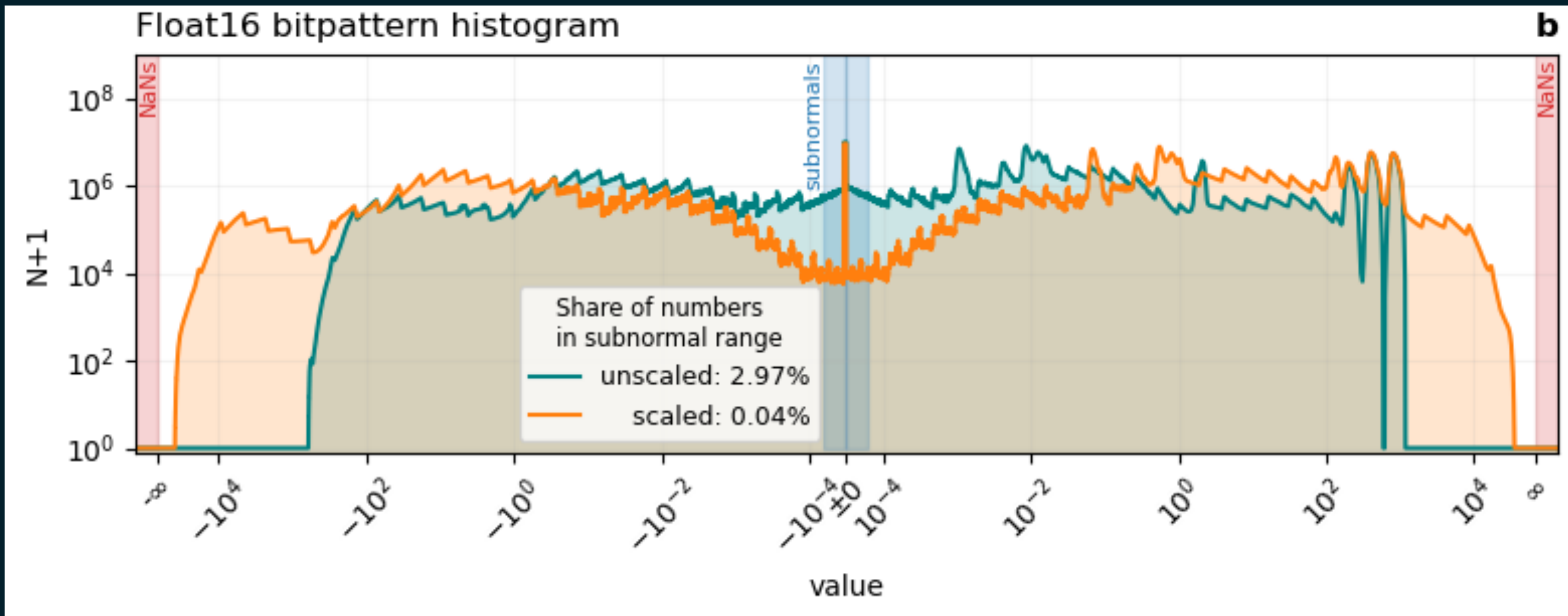
- Re-ordering
- Re-scaling
- Precomputing constants
- ...

Scaling: Squeezing algorithms into Float16

Scale the equations

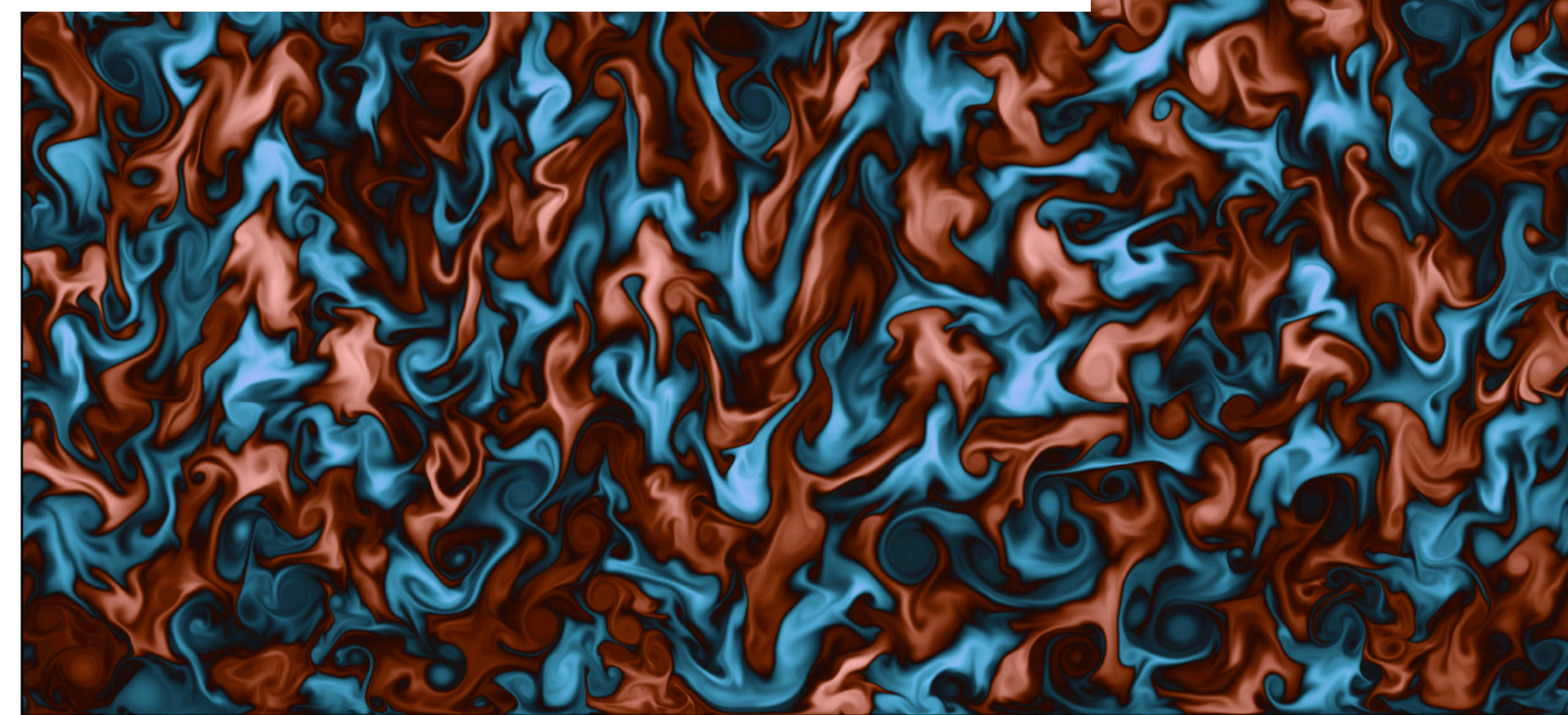
$$\begin{aligned}\hat{\partial}_t \hat{u} &= \frac{[s\Delta x f] + \hat{\xi}}{\hat{h}} \frac{\hat{v}\hat{h}}{s} - \hat{\partial}_x \left(\left[\frac{1}{2s} \right] (\hat{u}^2 + \hat{v}^2) + \left[\frac{sg}{s_\eta} \right] \hat{\eta} \right) + \hat{v}_B \hat{V}^4 \hat{u} - [r\Delta x] \hat{u} + [s\Delta x F_x] \\ \hat{\partial}_t \hat{v} &= -\frac{[s\Delta x f] + \hat{\xi}}{\hat{h}} \frac{\hat{u}\hat{h}}{s} - \hat{\partial}_y \left(\left[\frac{1}{2s} \right] (\hat{u}^2 + \hat{v}^2) + \left[\frac{sg}{s_\eta} \right] \hat{\eta} \right) + \hat{v}_B \hat{V}^4 \hat{v} - [r\Delta x] \hat{v} + [s\Delta x F_y]\end{aligned}$$

- Scaling s mostly combined with precalculated constants [...]

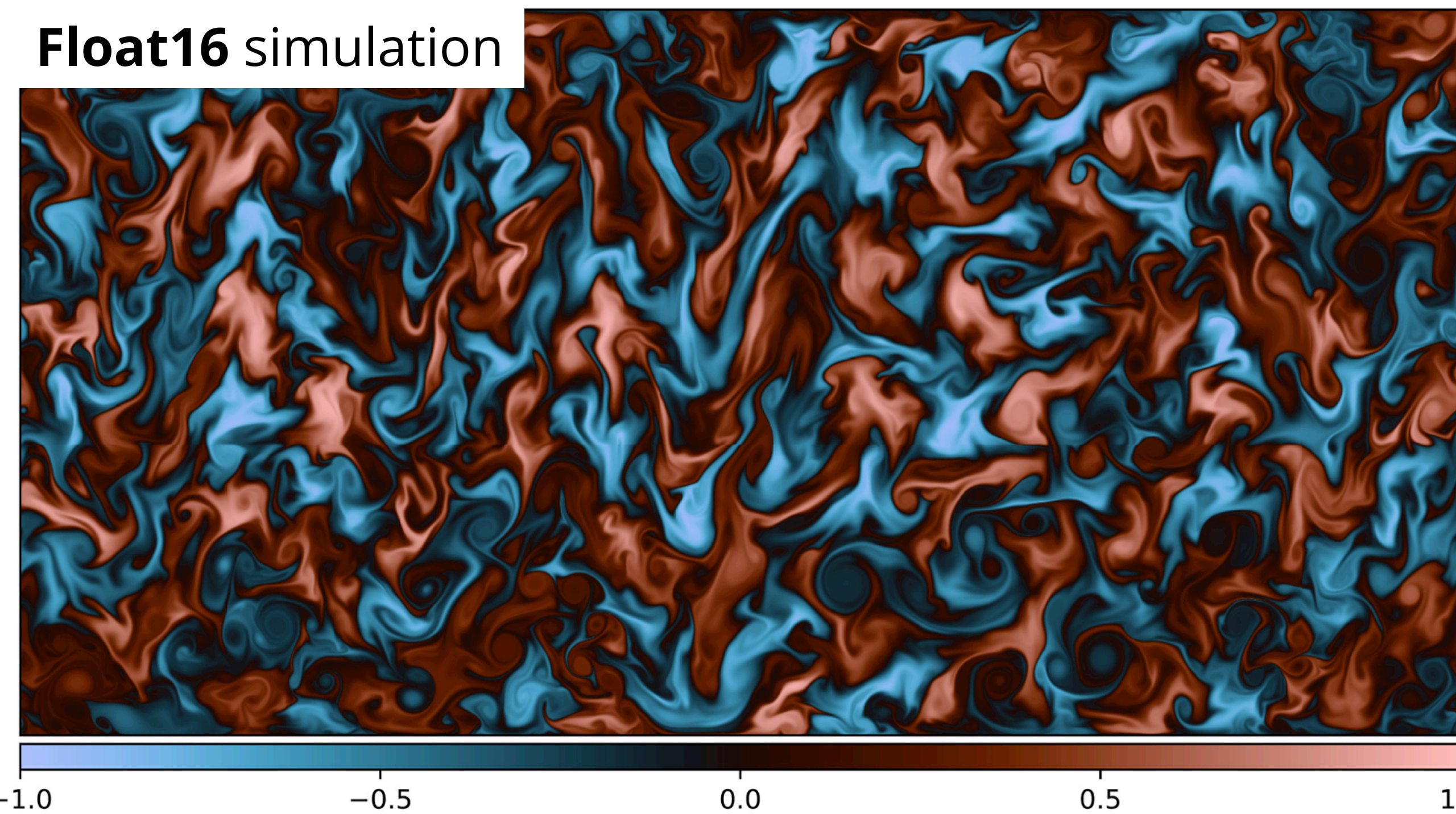


- **Rescaled equations** max out the available range
- Making use of **97%** of available numbers in Float16
- **A64FX** inefficiently supports **subnormals**: Scaling reduces their occurrence to $<0.04\%$, flush the rest to zero

Float64 reference simulation on **A64FX**



Float16 simulation



Fluid Simulations Accelerated With 16 Bits: Approaching 4x Speedup on A64FX by Squeezing ShallowWaters.jl Into Float16

Milan Klöwer [✉](#), Sam Hatfield, Matteo Croci, Peter D. Düben, Tim N. Palmer

First published: 28 January 2022 | <https://doi.org/10.1029/2021MS002684>

Published!



Float16 is **3.6x faster**
with scaled equations &
compensated time
integration

Reach out

- milank.de
- github.com/milankl
- [twitter @milankloewer](https://twitter.com/milankloewer)

Next project

milankl / SpeedyWeather.jl Public

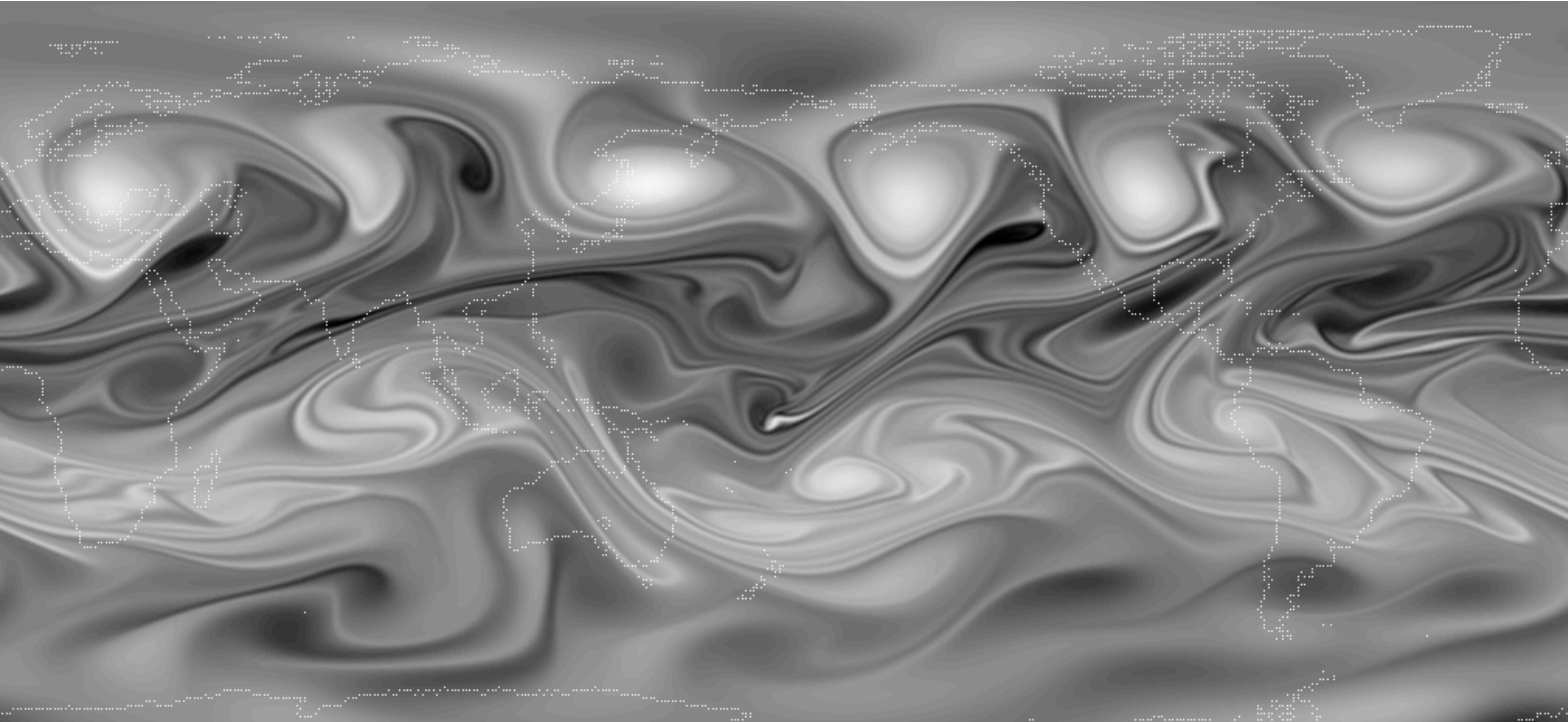
Languages

- Julia 97.9%
- Jupyter Notebook 2.1%

SpeedyWeather.jl

CI passing docs dev

The focus of SpeedyWeather.jl is to develop a global atmospheric model with simple physics, that can run at



A global 16-bit
atmospheric
model on GPUs

get involved!