Chair of Scientific Computing in Computer Science
Univ.-Prof. Dr. Hans-Joachim Bungartz

Technische Universität München

**Seminar Final Presentation**

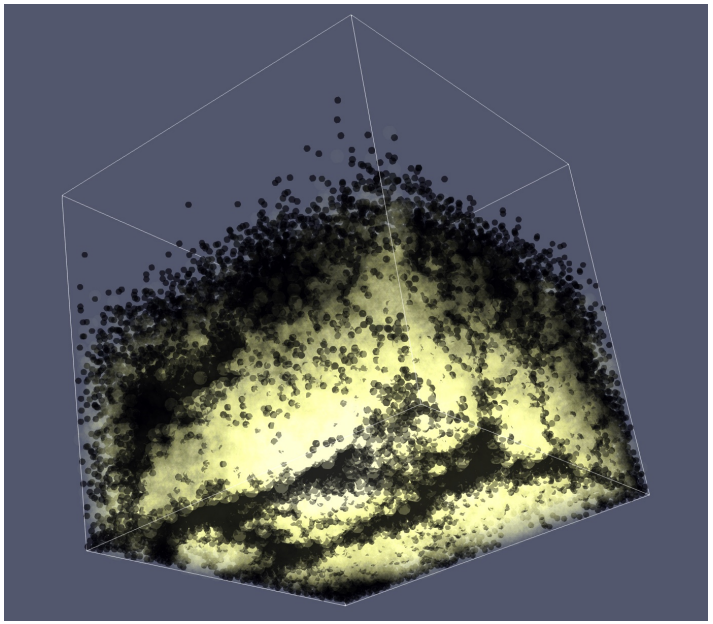
# Algorithm Selection and Auto-Tuning in AutoPas

**Manuel Lerchner**
manuel.lerchner@tum.de

# Table of Contents

# Software Challenges in Molecular Dynamics

- Enormous numbers of particles
  - naively: $\mathcal{O}(N^2)$ interactions
- Solution: highly optimized algorithms
  - Make full use of hardware
  - Smart data structures
- Every millisecond counts



# Table of Contents

# Trade-off: Particle Containers

- How to identify interacting particles?
- Short-range forces allow for cutoff radius $\rightarrow \mathcal{O}(N)$ interactions
- Different implementations possible

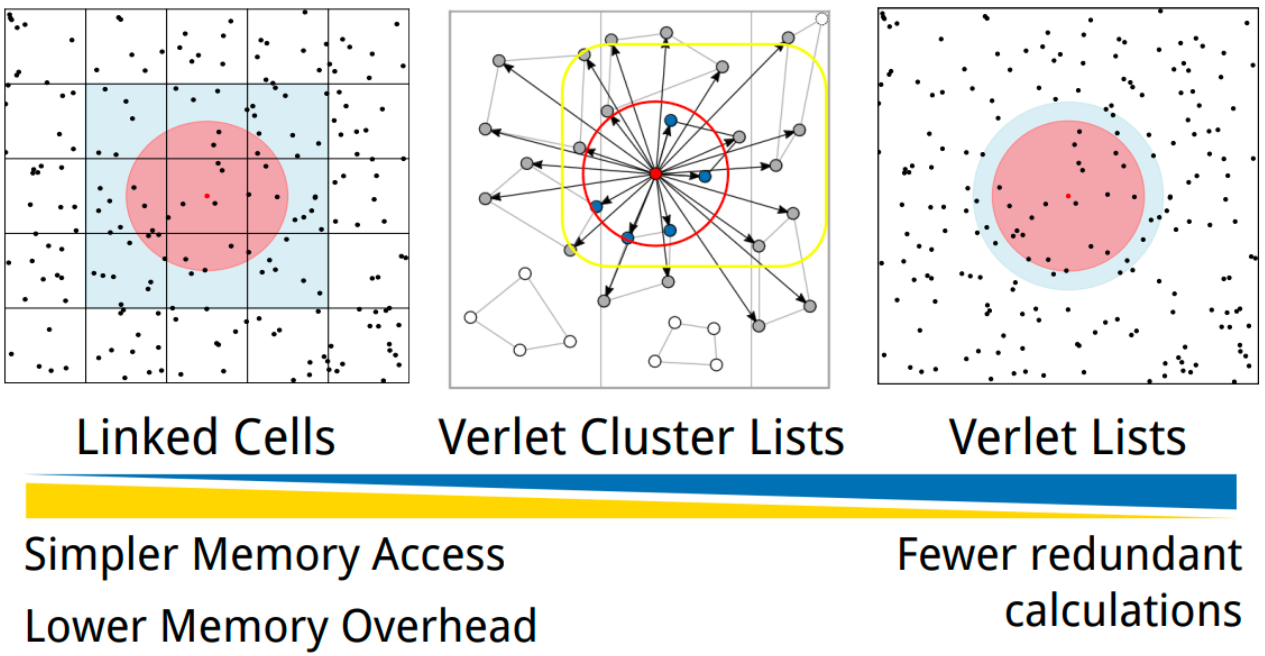
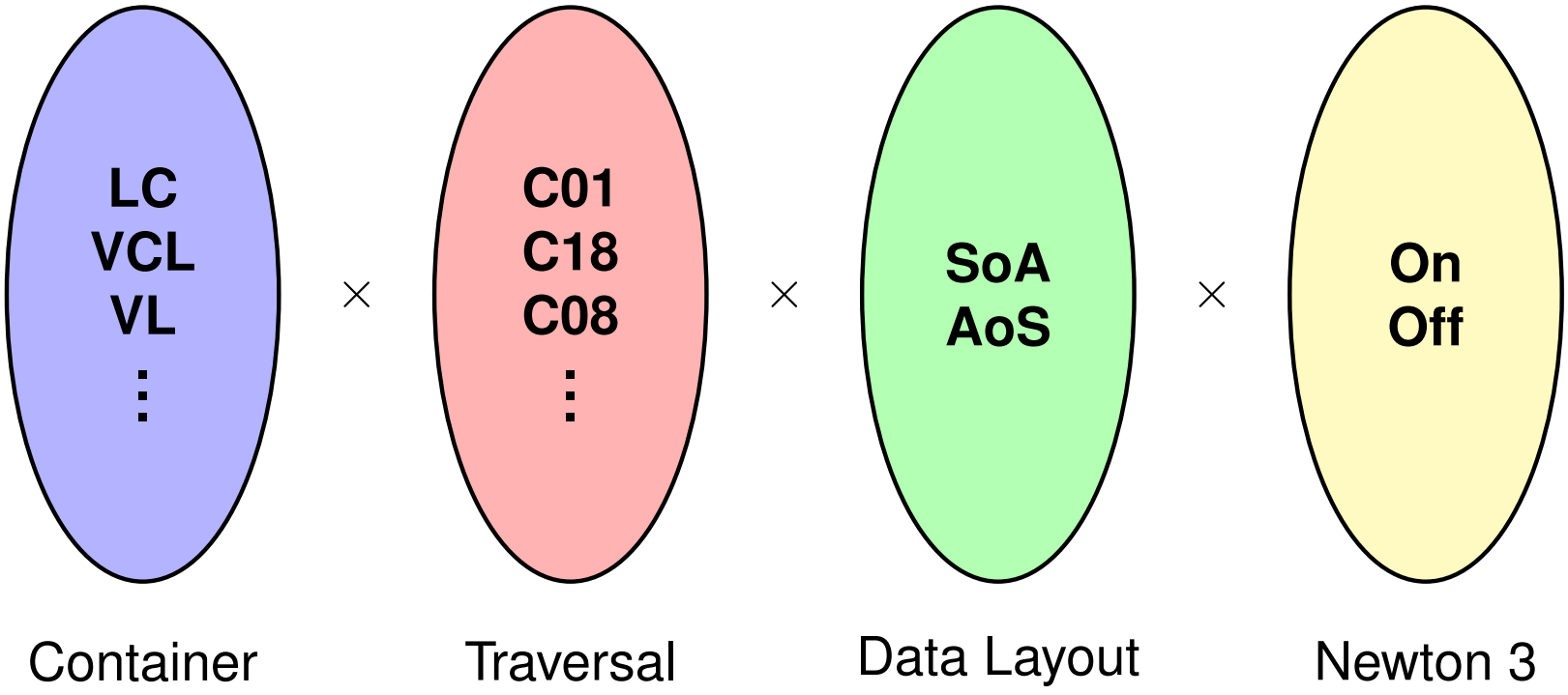



Figure: [Newcome et al., 2024]

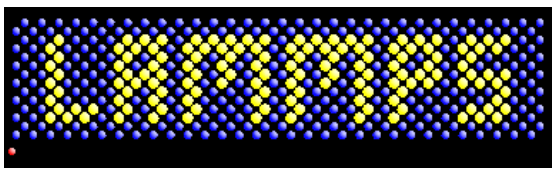
# Many more difficult choices

- No silver bullet!
- Hardware- and Scenario-dependent



# Traditional MD Engines


- Based on a single container
- Performance through:
  - Very optimized code
  - Vectorization
- Drawbacks:
  - Suboptimal for some scenarios
  - Require manual tuning



ls1
Mardyn

# Table of Contents

AutoPas

## What is AutoPas?

- Library for arbitrary N-body simulations
- Implements all algorithms
  - Modular
  - All* configurations possible
- Idea: AutoTuning
  - Finds optimal configuration
  - $\rightarrow$ Thus optimal performance
- Arbitrary user code

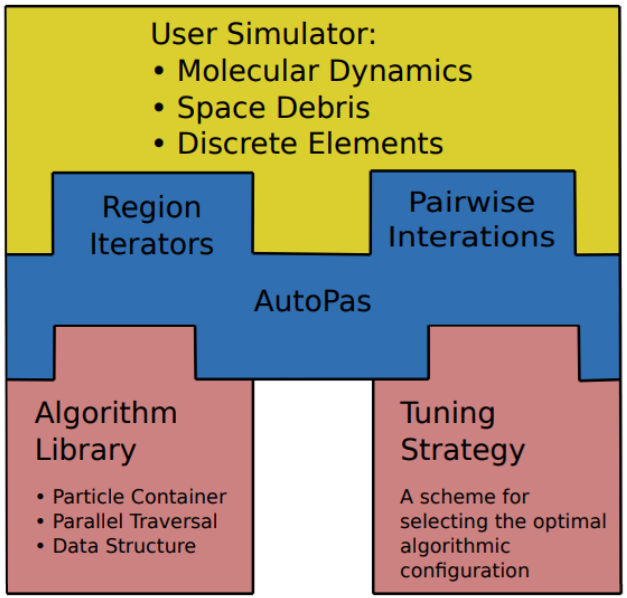
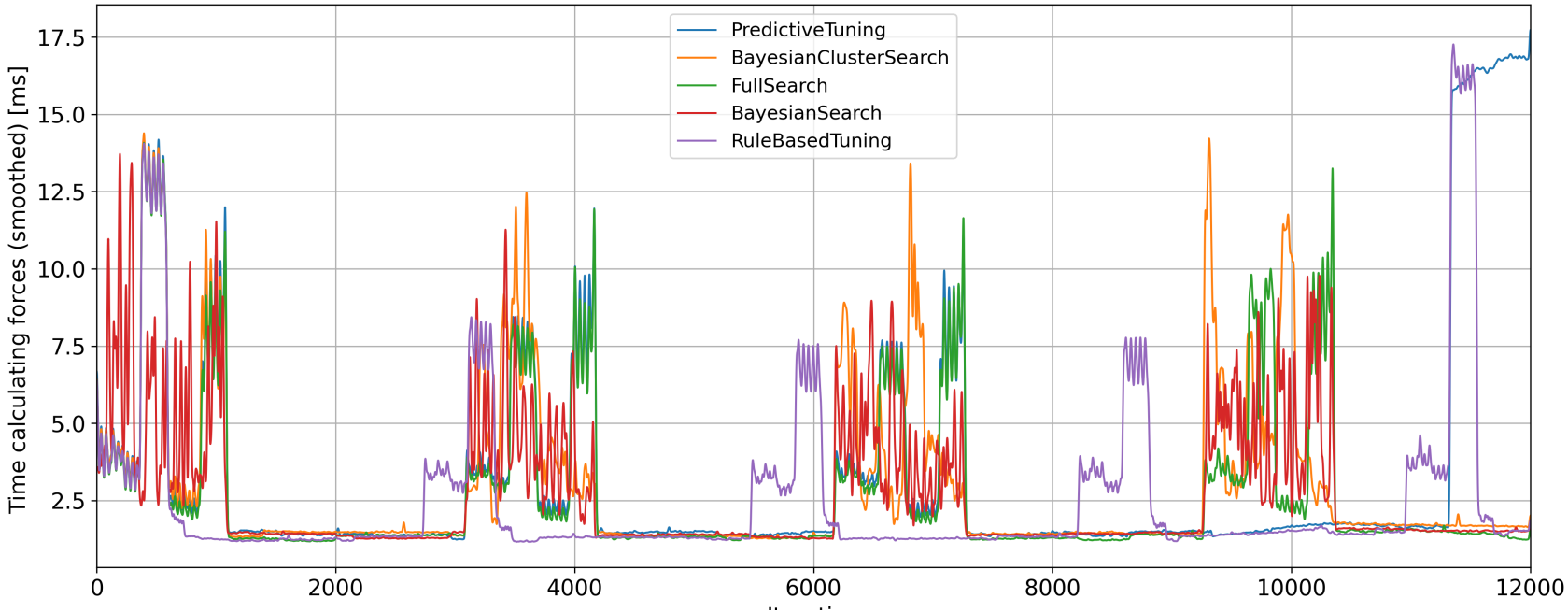



Figure: [Newcome et al., 2023]

# Table of Contents

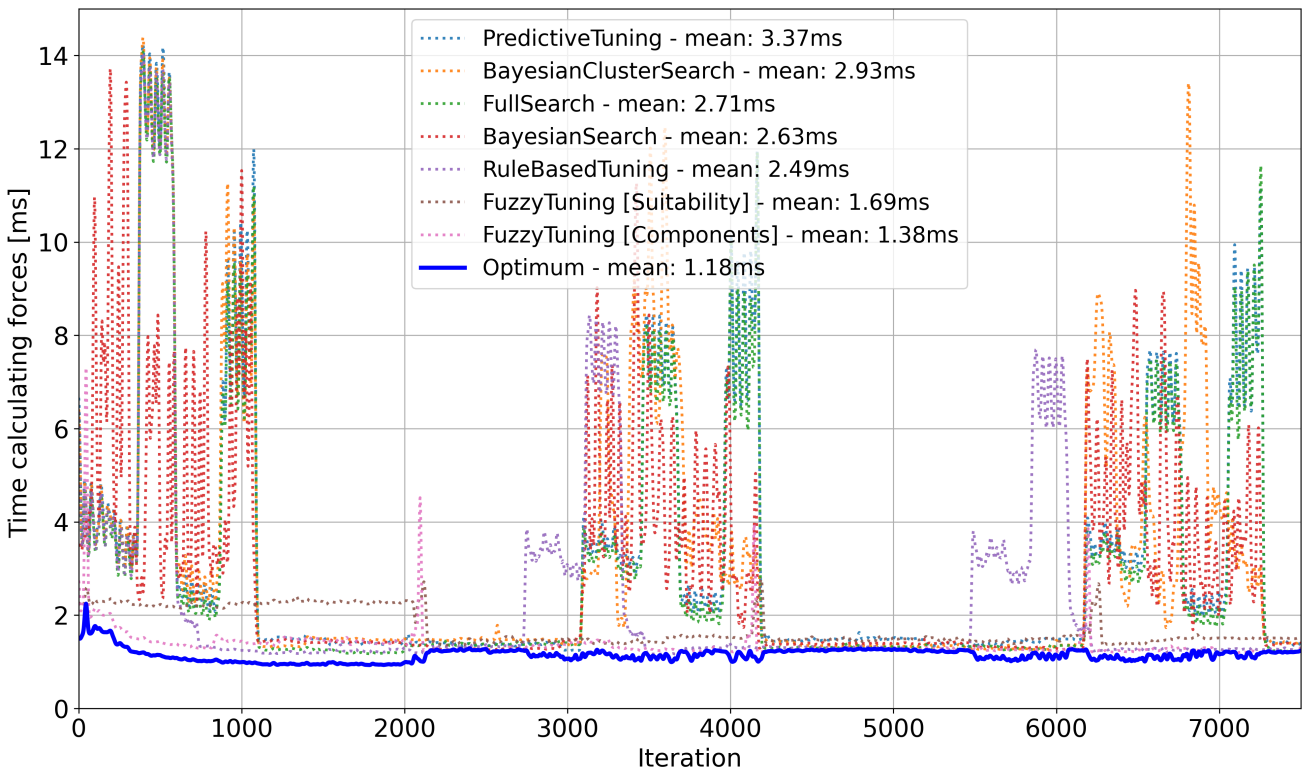
# Auto-Tuning in AutoPas

- Tuning-Phase: Evaluate promising configurations
- Simulation-Phase: Use best configuration
- Capable of reducing simulation time!



# Problems of Auto-Tuning

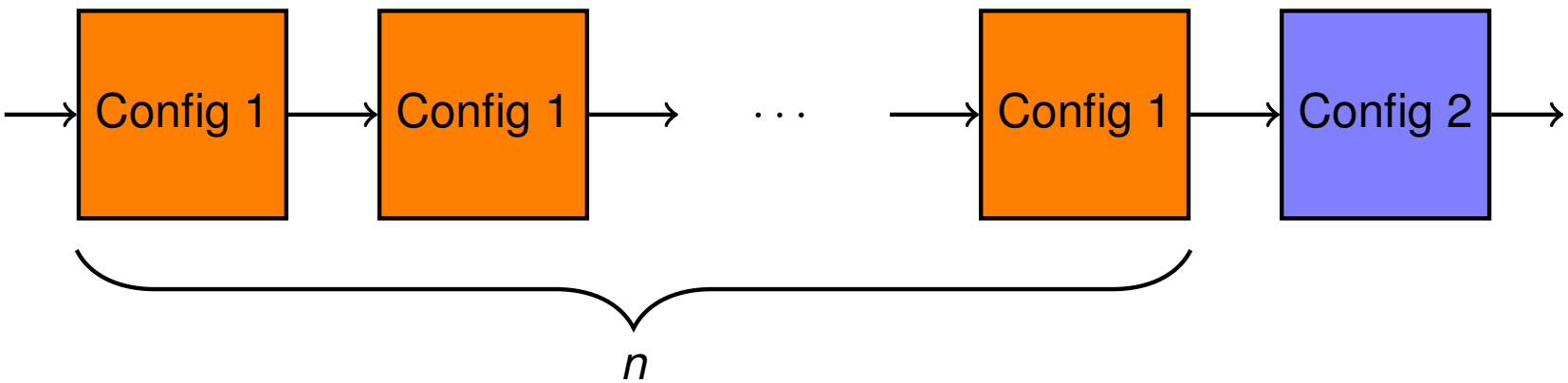
- Overhead from evaluating suboptimal configurations
  - Orders of magnitude slower!
- Unnecessary periodic re-tuning



# Table of Contents

# Early Stopping Optimization

- Idea: Stop long-running evaluations early
- Configurations get sampled multiple times
- Naive approach:
  - Last sample performed poorly?
  - Stop further evaluations

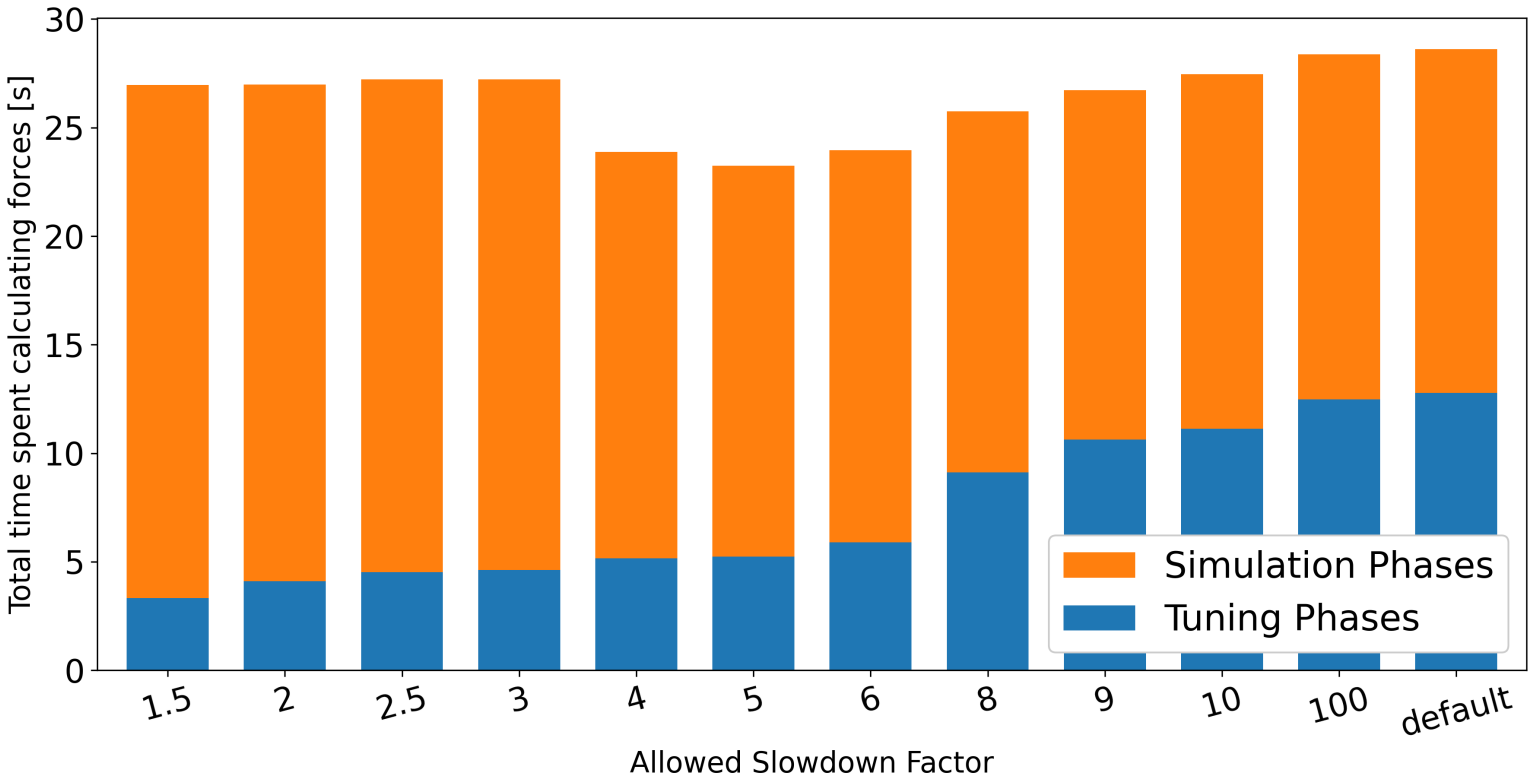


# Naive Early Stopping

- Changes to AutoPas are minimal
  - Keep track of best performance seen so far
  - `slowdownFactor` $= \frac{t_{\text{sample}}}{t_{\text{best}}}$
  - Call `retune()` if slowdownFactor exceeds threshold
- Hyperparameter: *allowedSlowdownFactor*
- Which *allowedSlowdownFactor* should be used?
  - *allowedSlowdownFactor* $\to 1$: Many aborts
  - *allowedSlowdownFactor* $\to \infty$: No early stopping
- Empirical evaluation needed

# Benchmark: Exploding Liquid + PredictiveTuning

- Optimal at *allowedSlowdownFactor* $= 5$
- From 28.6s to 23.2s
  - $\rightarrow$ 18.9% reduction



# Insights from Benchmarking

- Early stopping is beneficial
  - Never increased runtime!
- *allowedSlowdownFactor* probably not universal
  - More benchmarks needed!
  - Investigate other stopping criteria
- Potential for further improvements

**Thank you for your attention!**

# Questions?

# References I


Newcome, S. J., Gratl, F. A., Muehlhaeusser, M., Neumann, P., and Bungartz, H.-J. (2024).
Autopas: Dynamic algorithm selection in molecular dynamics for optimal time and energy.
In *SIAM Conference on Parallel Processing (PP24)*. SIAM.


Newcome, S. J., Gratl, F. A., Neumann, P., and Bungartz, H.-J. (2023).
Towards the smarter tuning of molecular dynamics simulations.
Amsterdam, The Netherlands.