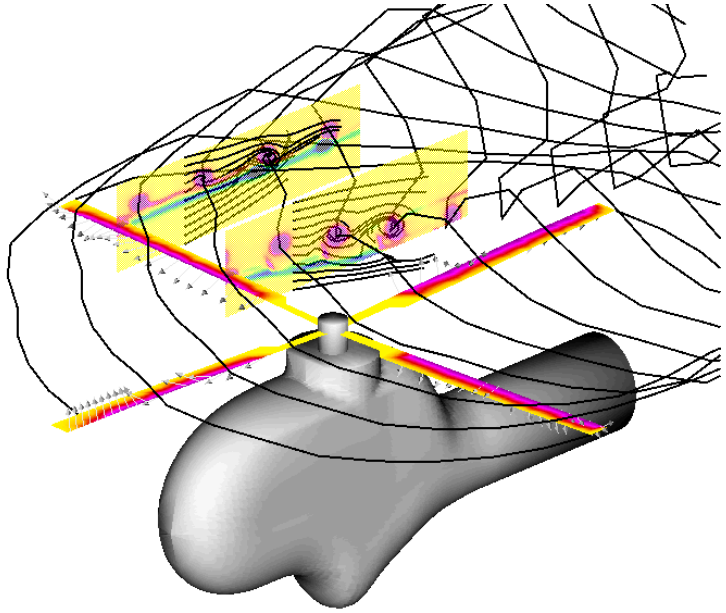
Parallel Rotor Wake Kernel Simulation with Python – Productivity versus Performance

Joachim Illmer, Melven Röhrig-Zöllner, **Achim Baserman**n
German Aerospace Center (DLR), Simulation and Software Technology

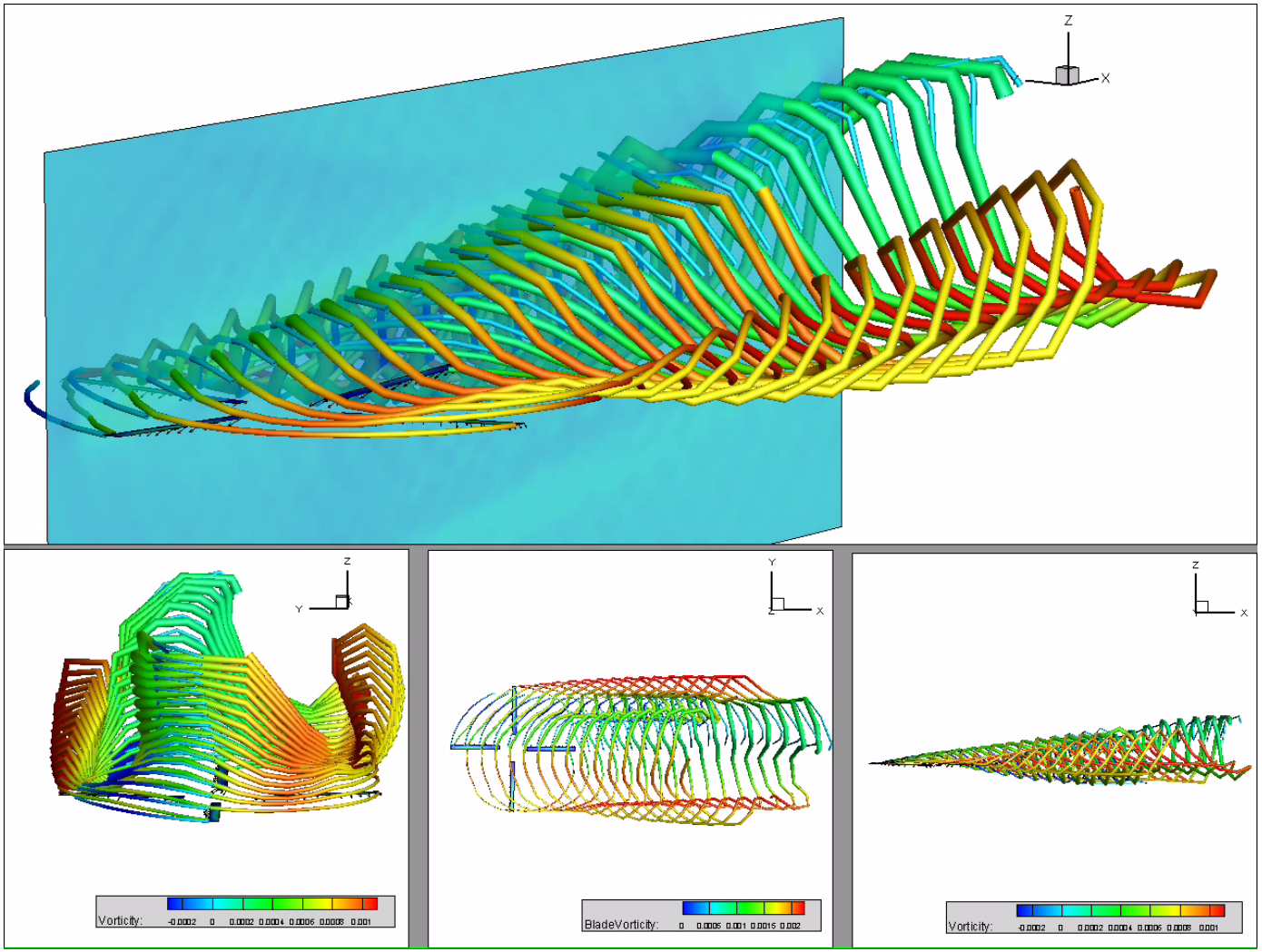
DLR

DLR Project Free-Wake



- Developed 1994-1996
- Simulates the flow around a helicopter's rotor
- Vortex-Lattice method
- Discretizes complex wake structures with a set of vortex elements
  - Account for vortex aging, stretching and compression
  - Vortex element length $\Delta\psi = 5°$, time step $\Delta\psi = 1°$
- Based on experimental data (from the international HART-program 1995)
- MPI-parallel implementation in Fortran
- Loose coupling with the DLR rotor simulation code S4

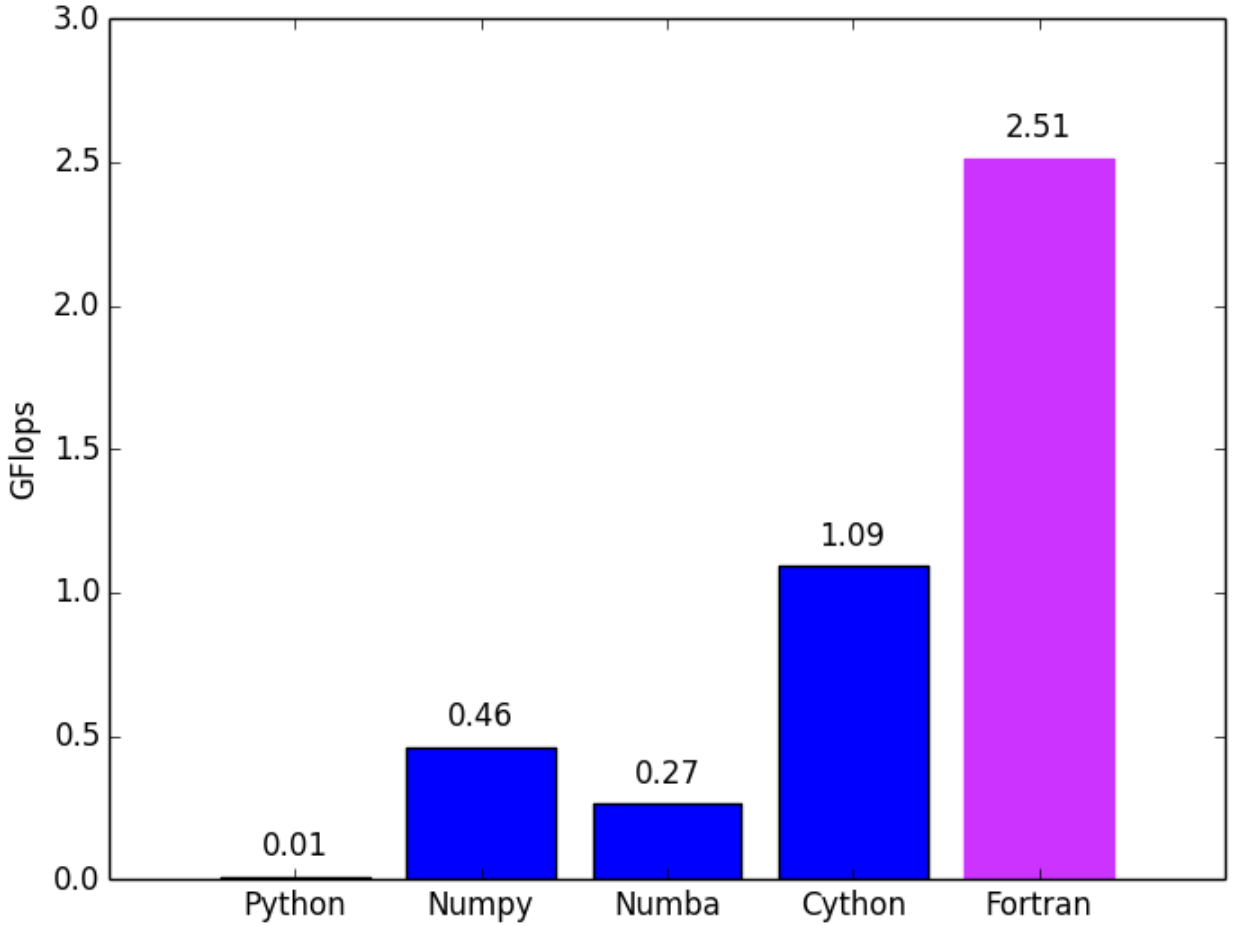
Vortex Visualization



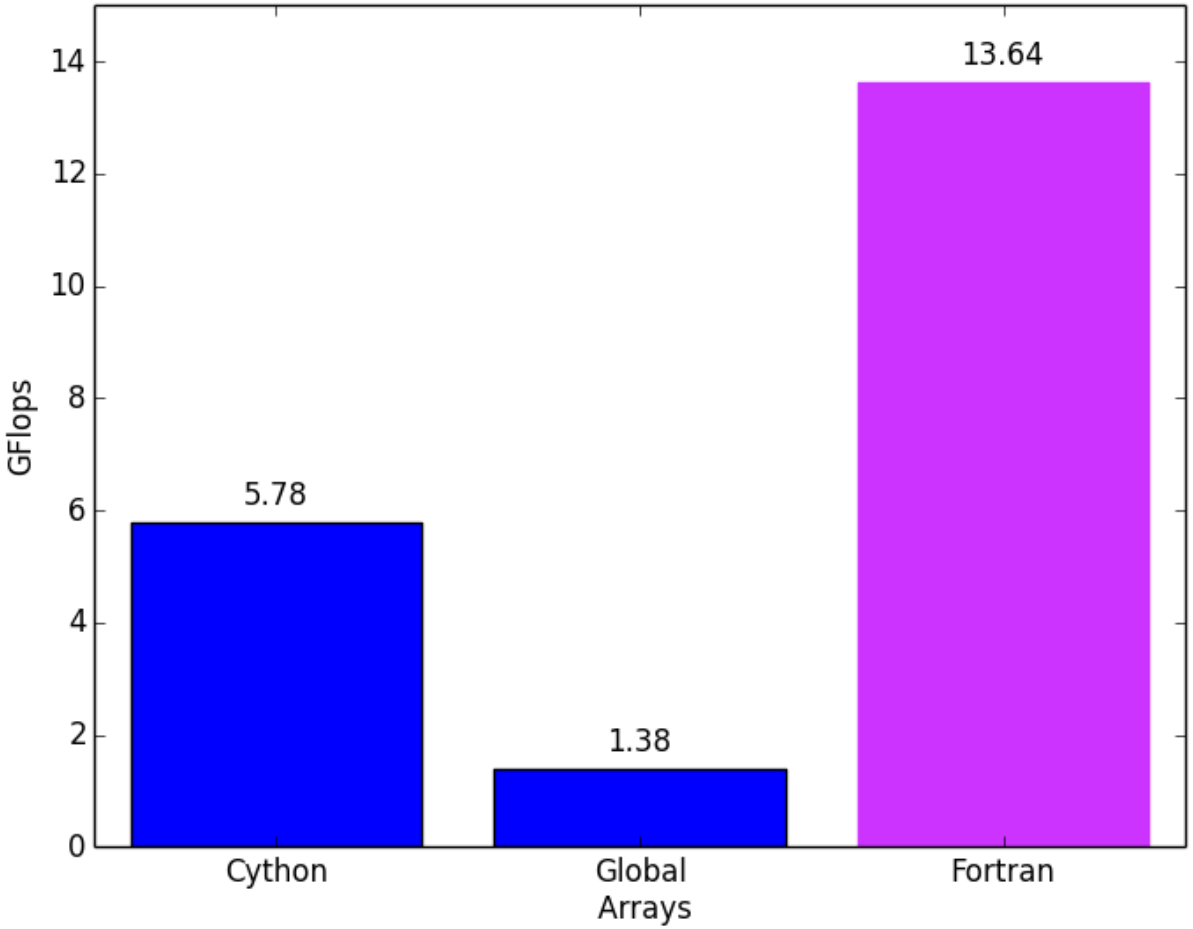
Parallel Python Programming for a Freewake Kernel

- Kernel available in Fortran with MPI and OpenACC
- Goal: Comparison with parallel Python implementations for
  - Multi-core CPUs
    - Cython with OpenMP
    - Python Bindings for Global Array Toolkit
  - GPGPUs
    - Numbapro

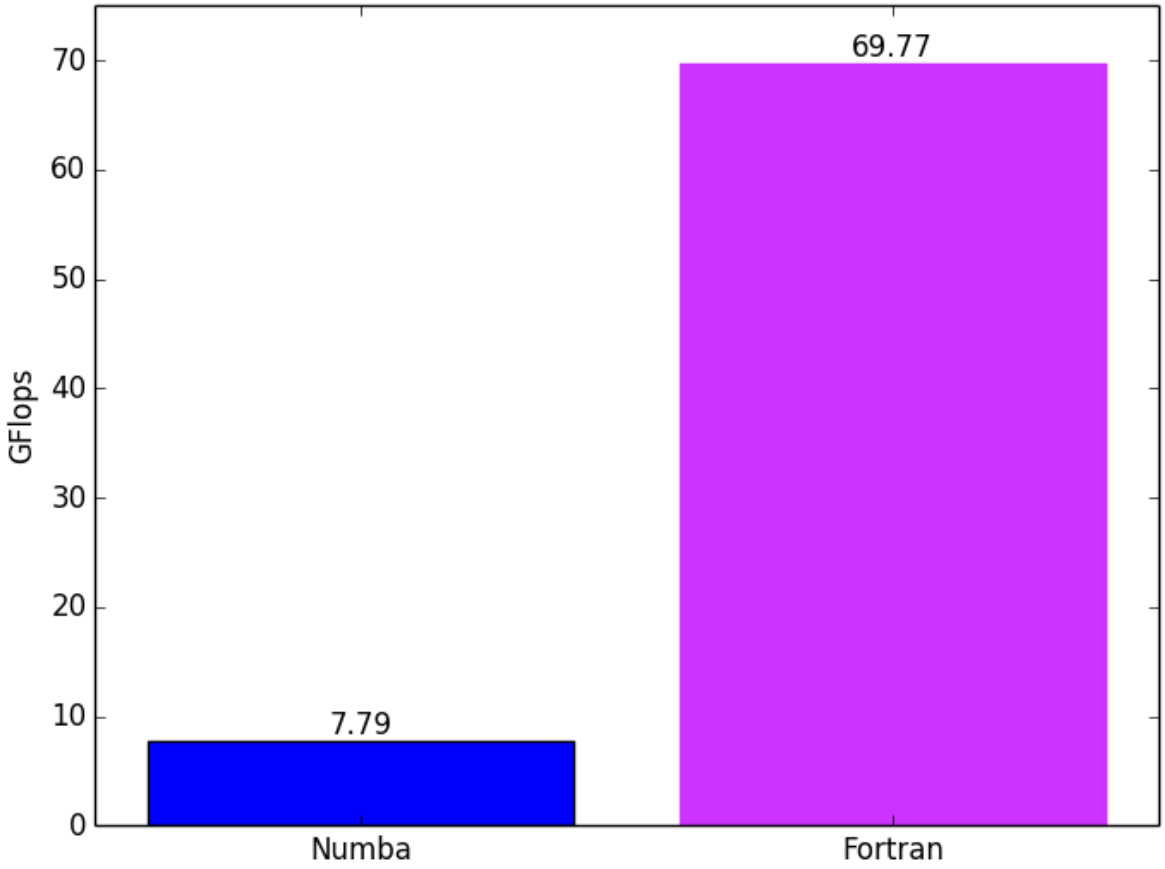
Single-Core Performance Tests (Xeon E5645, 6 Cores)



Multi-Core PerformanceTests (Xeon E5645, 6 Cores)



GPGPU Performance Tests (Nvidia Tesla C2075, 448 CUDA-Cores)



Conclusions

- Python: productivity of parallel programming high
- Really high programming level for
  - Python with Global Arrays for CPU and
  - Numba for GPU
- Cython with OpenMP fastest but nearly C programming level
- Performance for Cython tolerable but still distinctly below Fortran with MPI and OpenACC