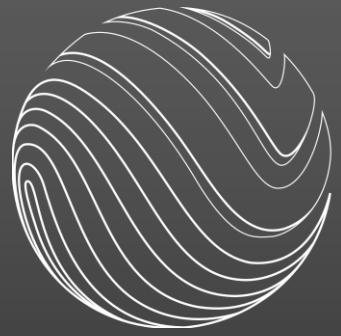


OPTIMAD

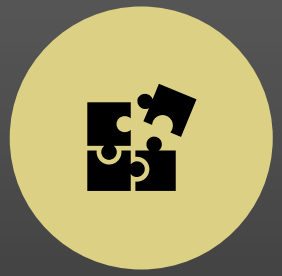
23*rd POP User Webinar - POP*: *The SME Perspective*

Marco Cisternino, PhD – Senior Software Developer

17/09/2021

About us

Spin-Off
Department of Aerospace Engineering
Politecnico di Torino



Provide excellence in fluid-dynamics and scientific computing

10 PhD
3MSc
3 PhD students
1 Marketing & Sales

Leading-edge proprietary tools

Proprietary tools

mimic – Computer aided surface manipulation and mesh morphing

RomBox – Expert system for engineering decision making process

immerFLOW – Hybrid MPI+OpenMP parallel immersed boundary flow solver

Leading-edge proprietary tools

HPC in our business - immerFLOW

Industry standard (especially for SMEs**)**

- cheap models (RANS) -> low reliability + need for experts

Industry needs

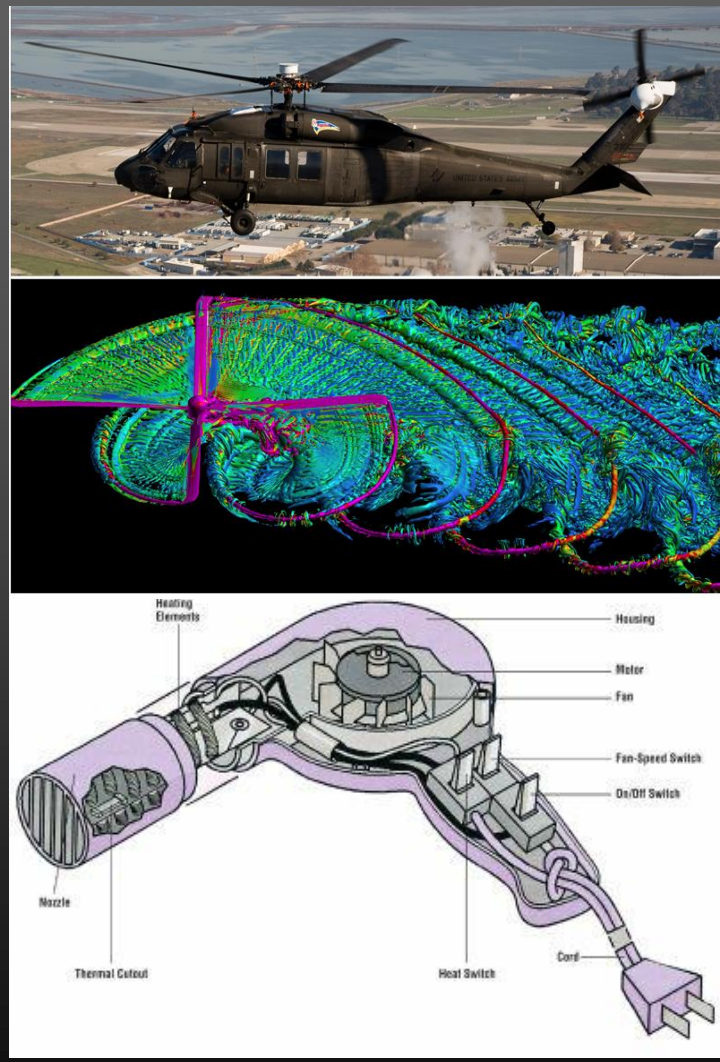
- more reliable models (DNS, LES, WLES) -> higher simulation cost

| more reliable |

Method	Scale of turbulence	Resolution required			
		Surface points	Wake points	Time steps	Total operations
Direct Navier Stokes (DNS)	No modeling	10^{16}	10^{17}	10^{8}	10^{25}
Large Eddy Simulation (LES)	Sub-grid modeling	10^{12}	10^{9}	10^{8}	10^{20}
LES with wall layer	Near wall & sub-grid modeling	10^{10}	10^{9}	10^{7}	10^{17}
Reynolds Navier Stokes (RANS)	All scales are modeled	10^{7}	10^{7}	10^{4}	10^{11}
Euler equation	Scales are absent	10^{7}	10^{7}	10^{3}	10^{10}
Inviscid vortex based methods	Scales are absent	10^{2}	10^{2}	10^{3}	10^{5}

Computational cost of analysis of a wing of AR=10, Re=5 x 10^{6}

HPC services as a leverage for HiFi simulation democratization



POP – our experience with NAG

immerFLOW Performance Assessment:

- Excellent MPI scaling up to 768 cores
- MPI serialization at 1534 cores
- Poor OpenMP performance due to atomic lock contention at high number of threads

immerFLOW Proof-of-Concept:

- Overall efficiency improvements mainly at high number of threads
- Introduction of OpenMP taskloops (GPUs porting)
- Communication/calculations overlapping (10% performance improvement)
- Removal of temporary data copies

Very important insight in our code parallel behaviour, laying the groundwork for future improvements of code efficiency