CESAR
Aerodynamic Design of Small Commercial Aircraft

The block-structured RANS solver FLOWer

Jochen Raddatz

DLR, Institute of Aerodynamics and Flow Technology

Praha, 2009, 18–19 March

outline

- History
- FLOWer: general overview and numerical features
- Impact of the computational grid – example
- FLOWer
  - installation requirements
  - input + output files
  - spatial discretization
  - time stepping scheme + acceleration
  - turbulence models
- Block structure and grid logic

History

- 1980 +
  begin of development of Euler / RANS solvers at DLR

- 1991 +
  begin of FLOWer development in the framework of national project POPINDA

- 1995 - 2003
  FLOWer development in the framework of German CFD initiative MEGAFLOW

German CFD Initiative MEGAFLOW - Approach

MEGAFLOW

concentrated effort of DLR, aircraft industry and universities

Phase I (1995-1998)

national project supported by the German Government

Phase II (since 1999)

funded by DLR and aircraft industry

Partners:

- DLR (coordinator)
- aircraft industry (Airbus-D, EADS-M, …)
- universities of Aachen, Berlin Braunschweig, Bremen, Darmstadt, München, ….
- GMD

Strategic Cooperation

software developers with specific products and knowledge

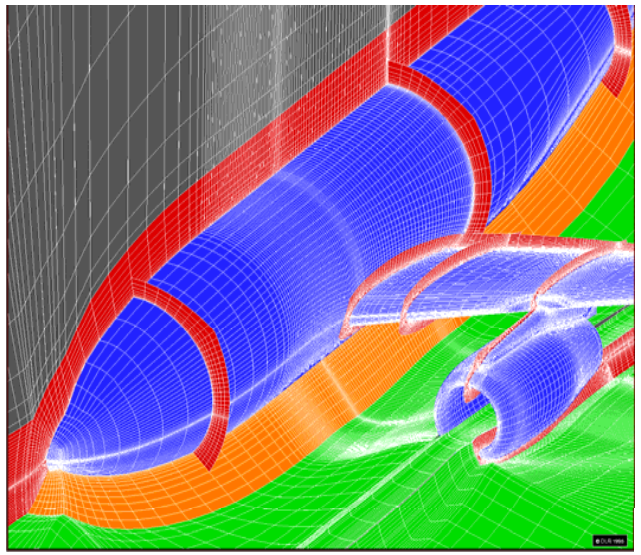
Partners:

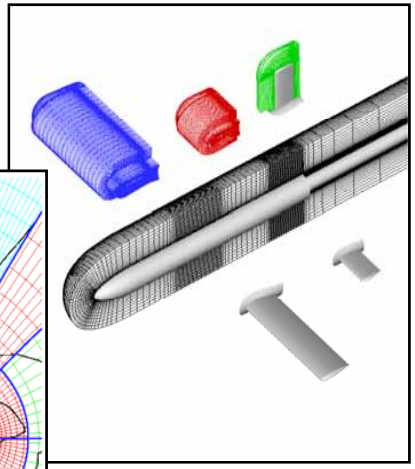
- CentaurSoft *(grid generation)*
- Synaps *(aerodynamic optimization)*
- NEC *(parallel solver)*

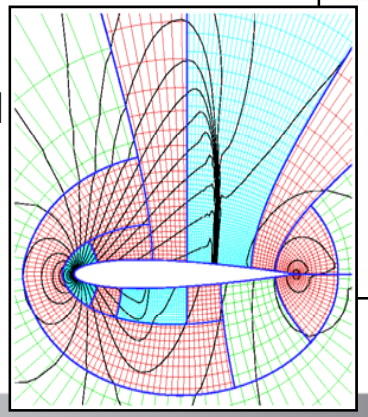
Block-Structured Navier-Stokes Solver
FLOWer

General Overview:

- solution of RANS equations for arbitrary moving bodies
- well adapted for external flows around aircaft-like configurations in subsonic, transonic and supersonic flow region
- efficient solver on vector & parallel computers
- computer platform independent
- high flexibility through discontinous block interfaces and overlapping grid technique (Chimera)
- development performed by a joined team from DLR, universities, Airbus Germany and EADS-M
- comprehensively validated
- routinely used by German Aircraft Industry, DLR and several Universities

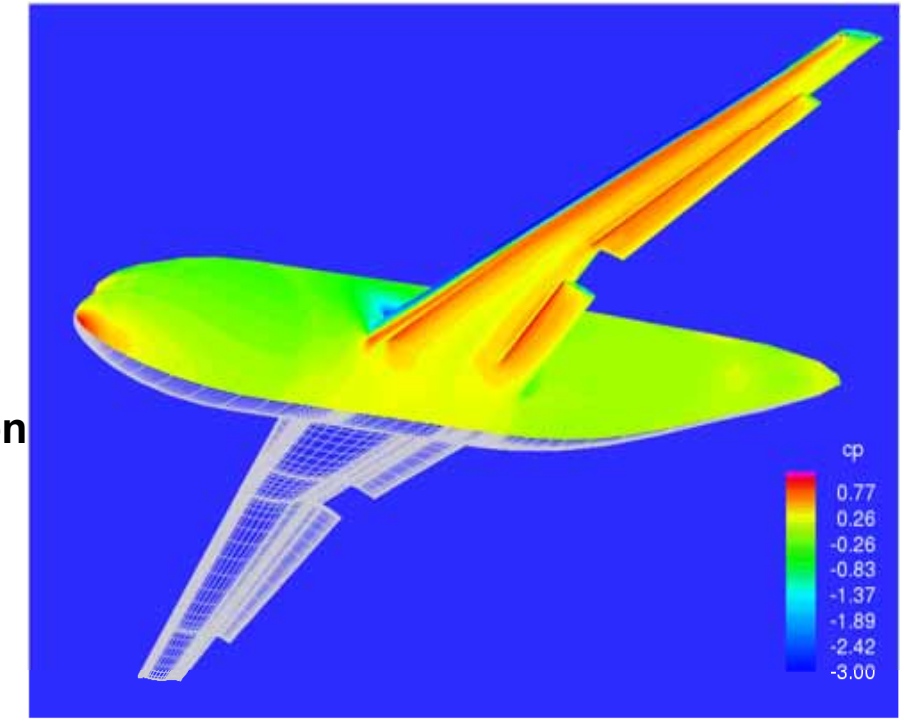






Block-Structured Navier-Stokes Solver
FLOWer

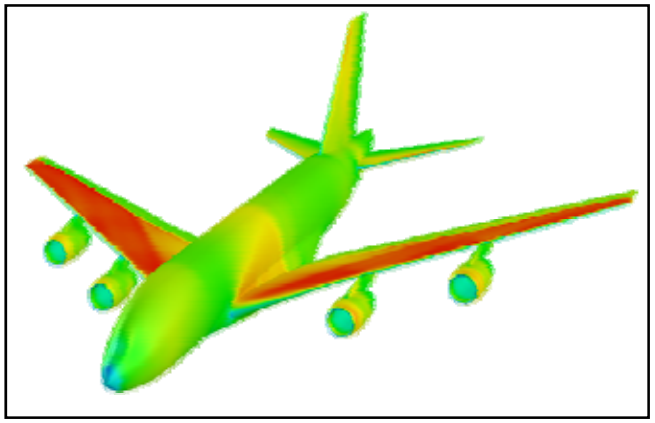
Numerical Features:

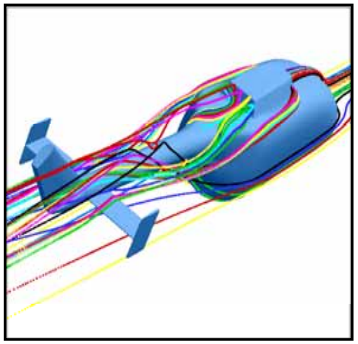


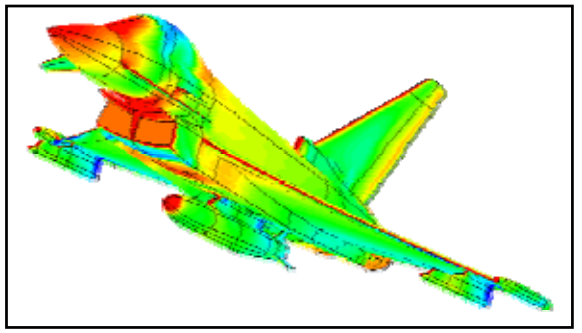
- finite volume discretization on block structured meshes
- cell vertex and cell centered formulation
- central and upwind schemes
- explicit, multi-stage time integration for steady flows
- multigrid
- various turbulence models
- implicit treatment of turbulence equations
- preconditioning for low speed flows
- implicit time integration and time step adaption for unsteady flows

Applications

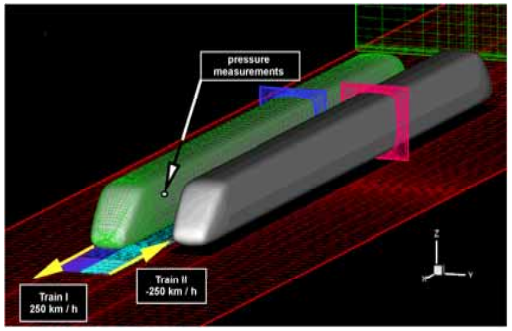
- transport aircraft
- military aircraft
- helicopters
- missiles
- hypersonic vehicles

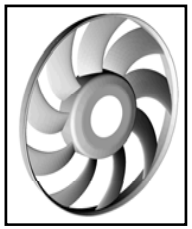




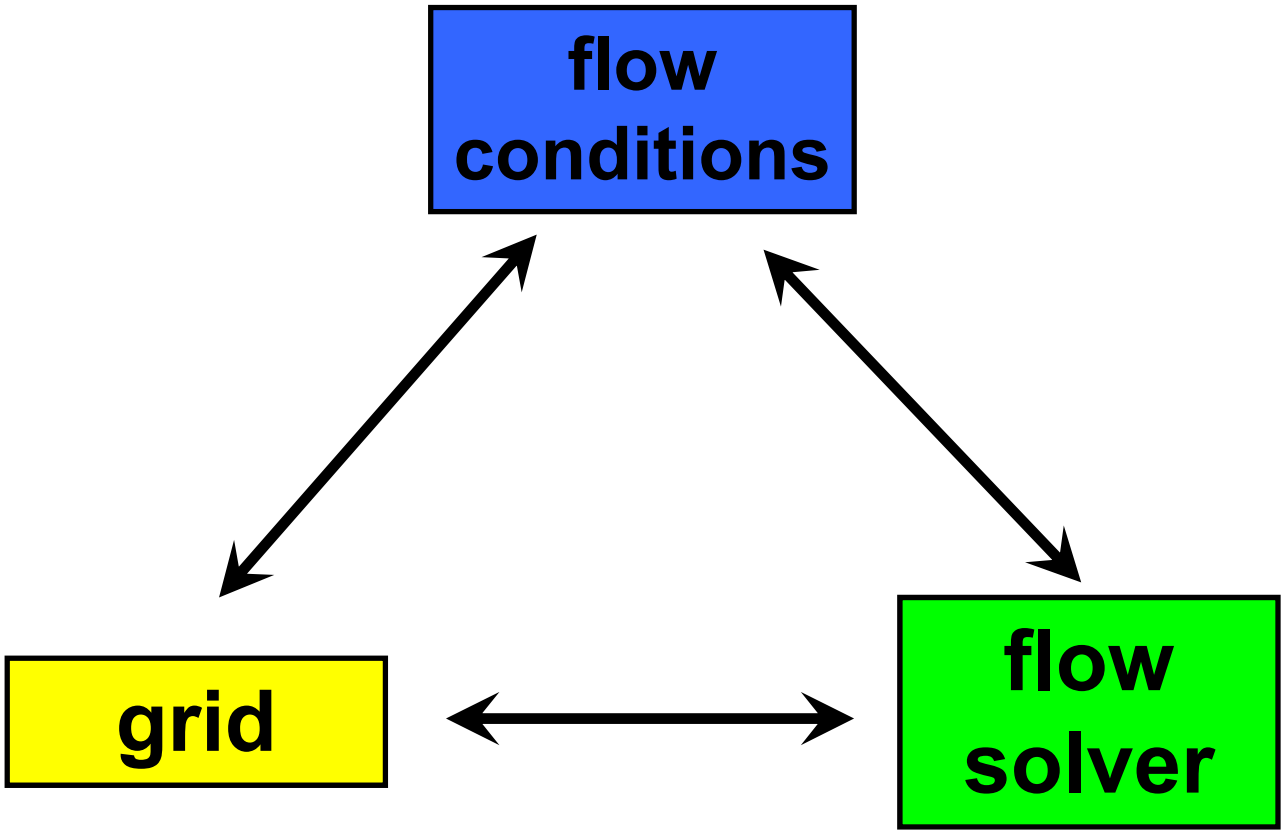


- **trains**
- **cooling fans**
- **wind turbine blades**

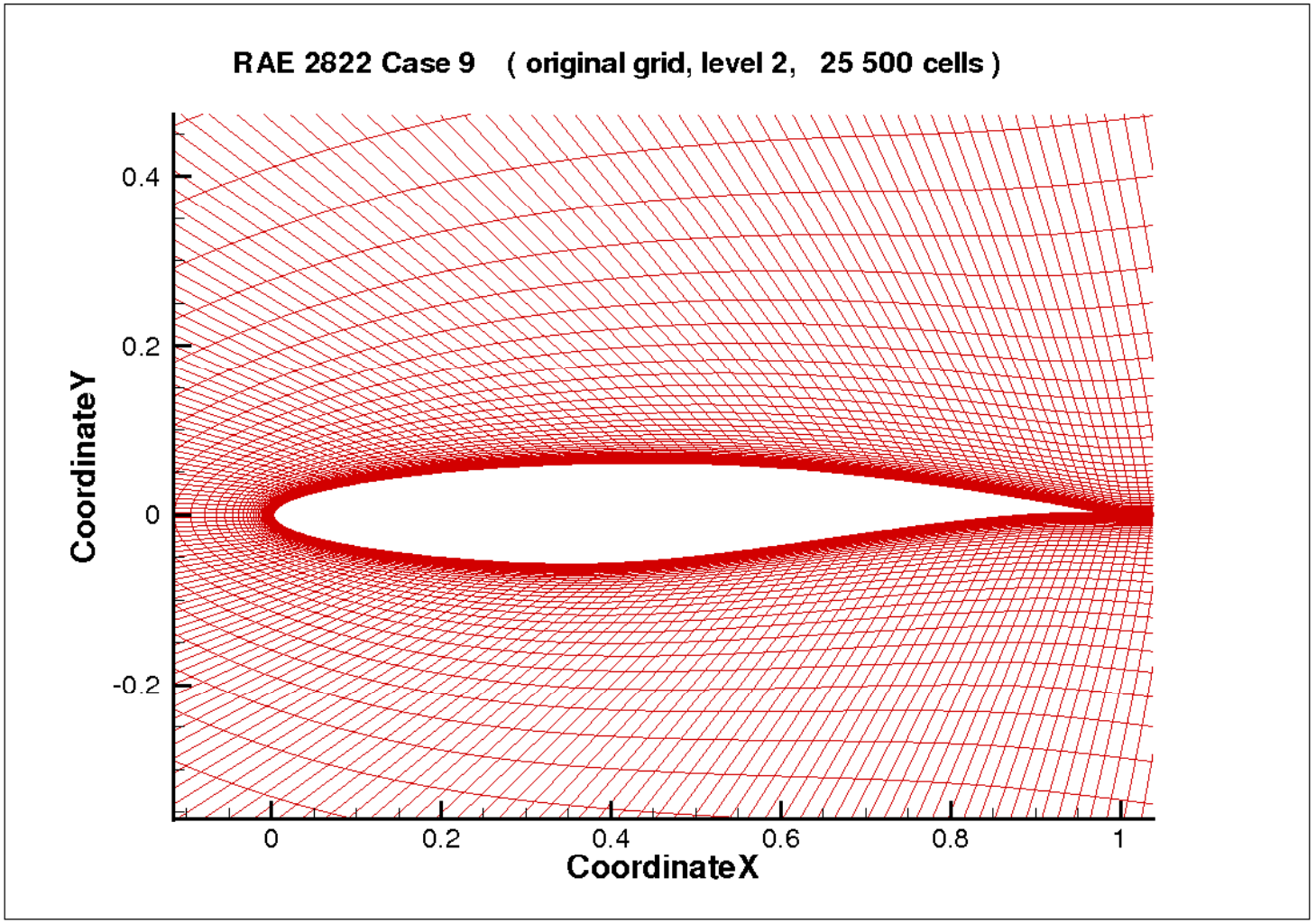




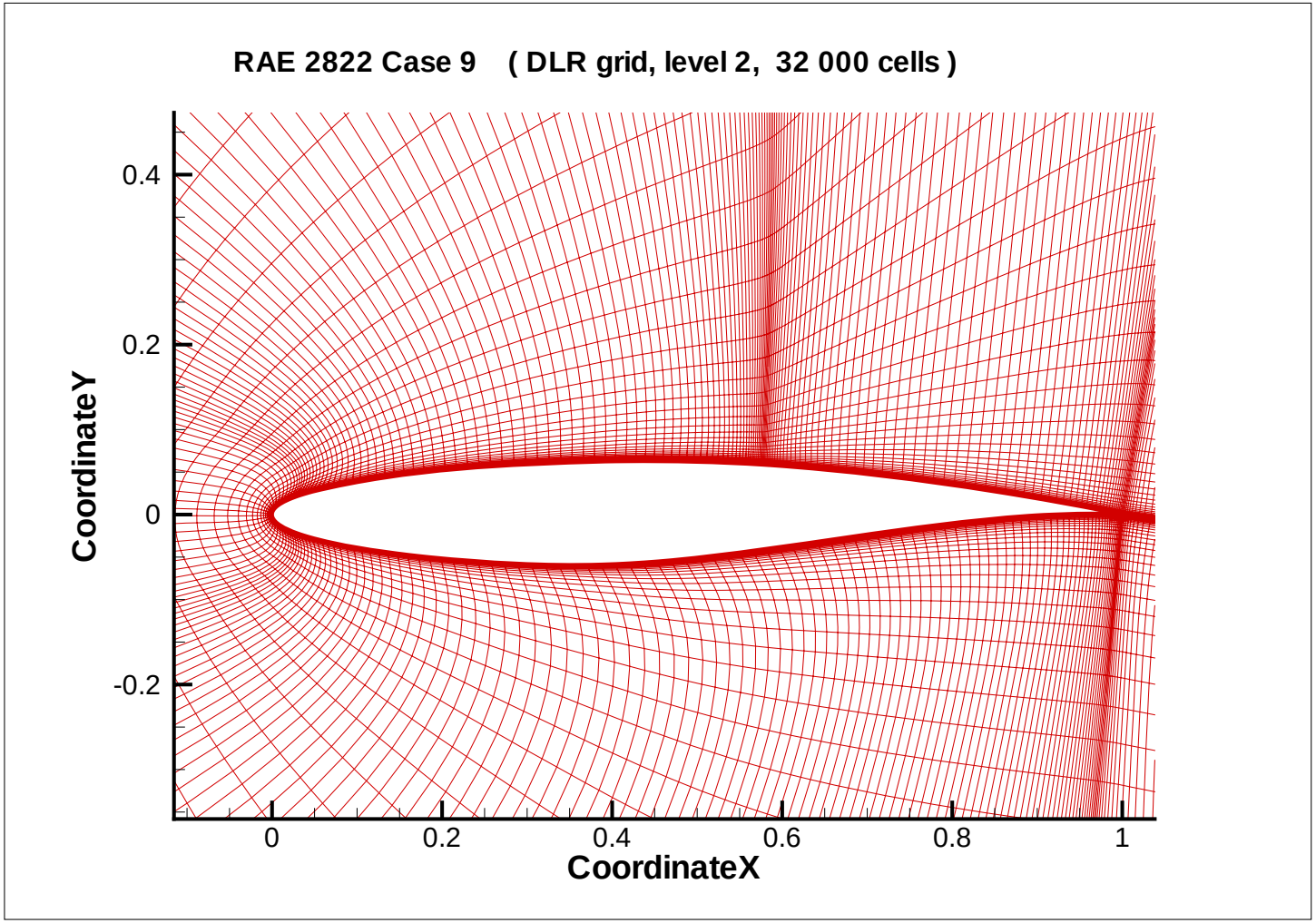
impact of the computational grid



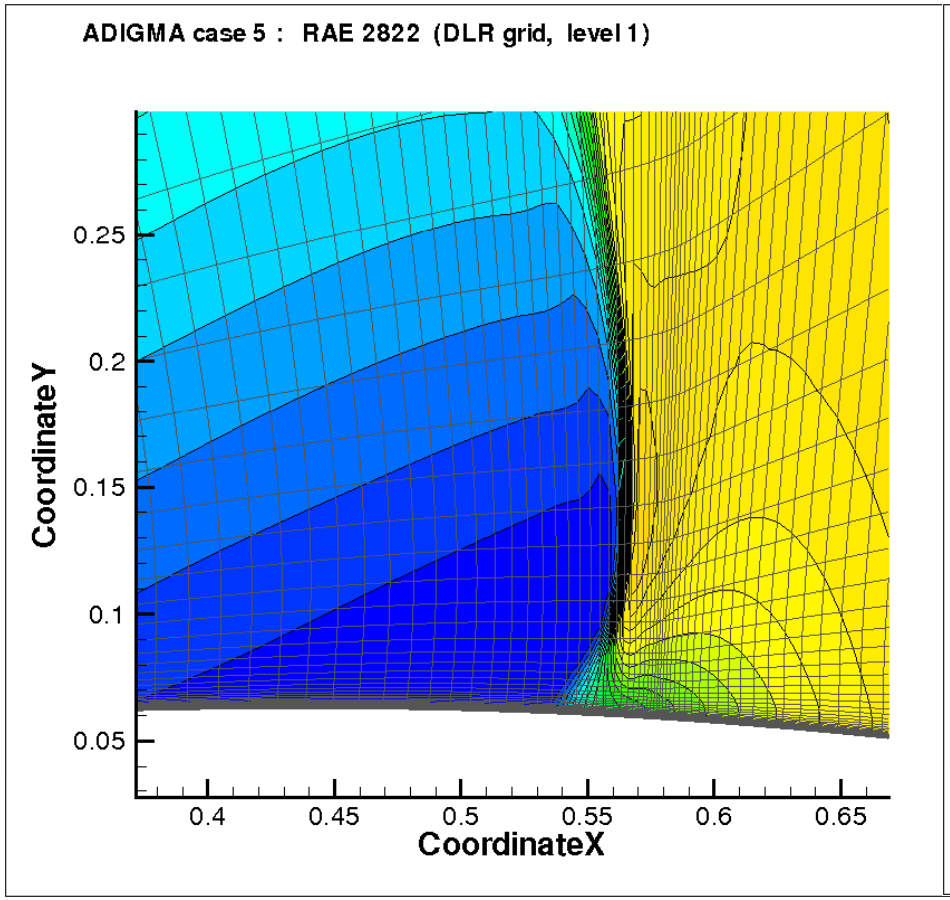
impact of the computational grid - example

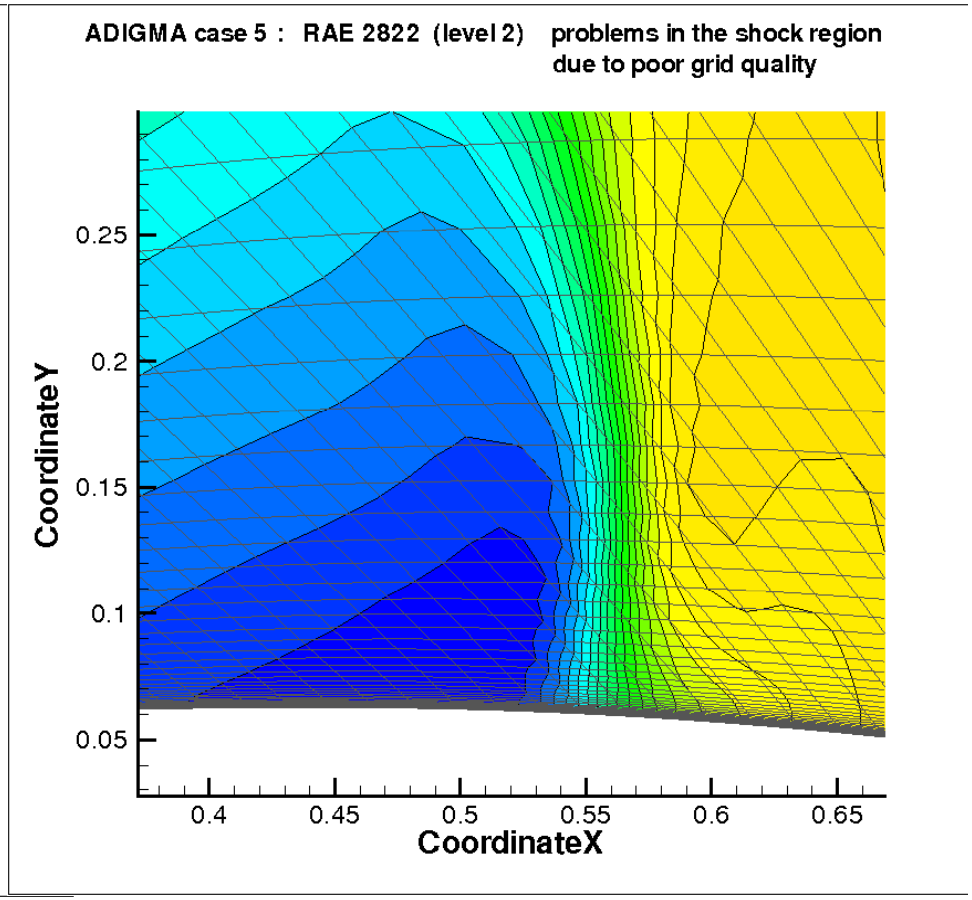


impact of the computational grid - example

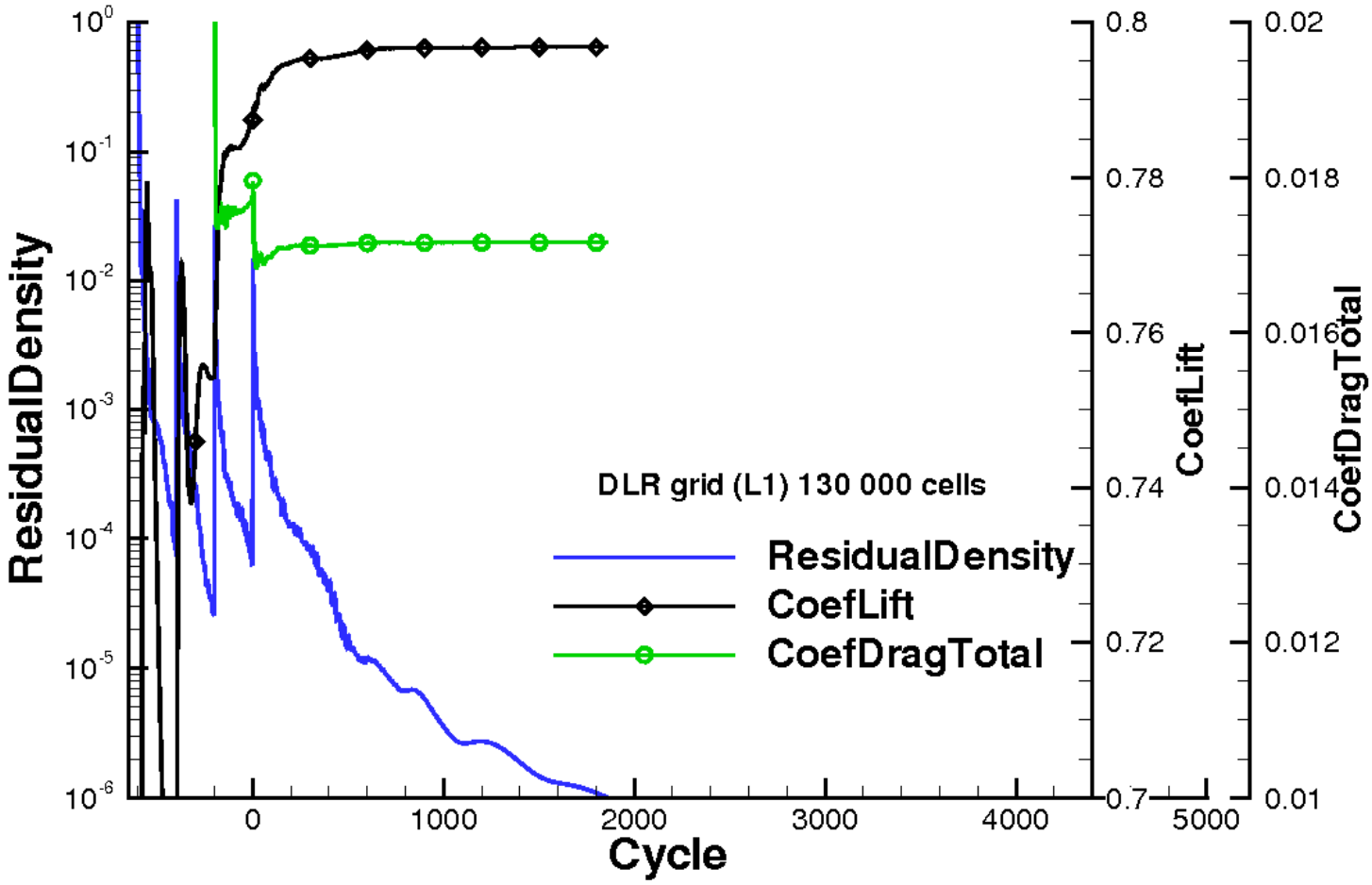


impact of the computational grid - example

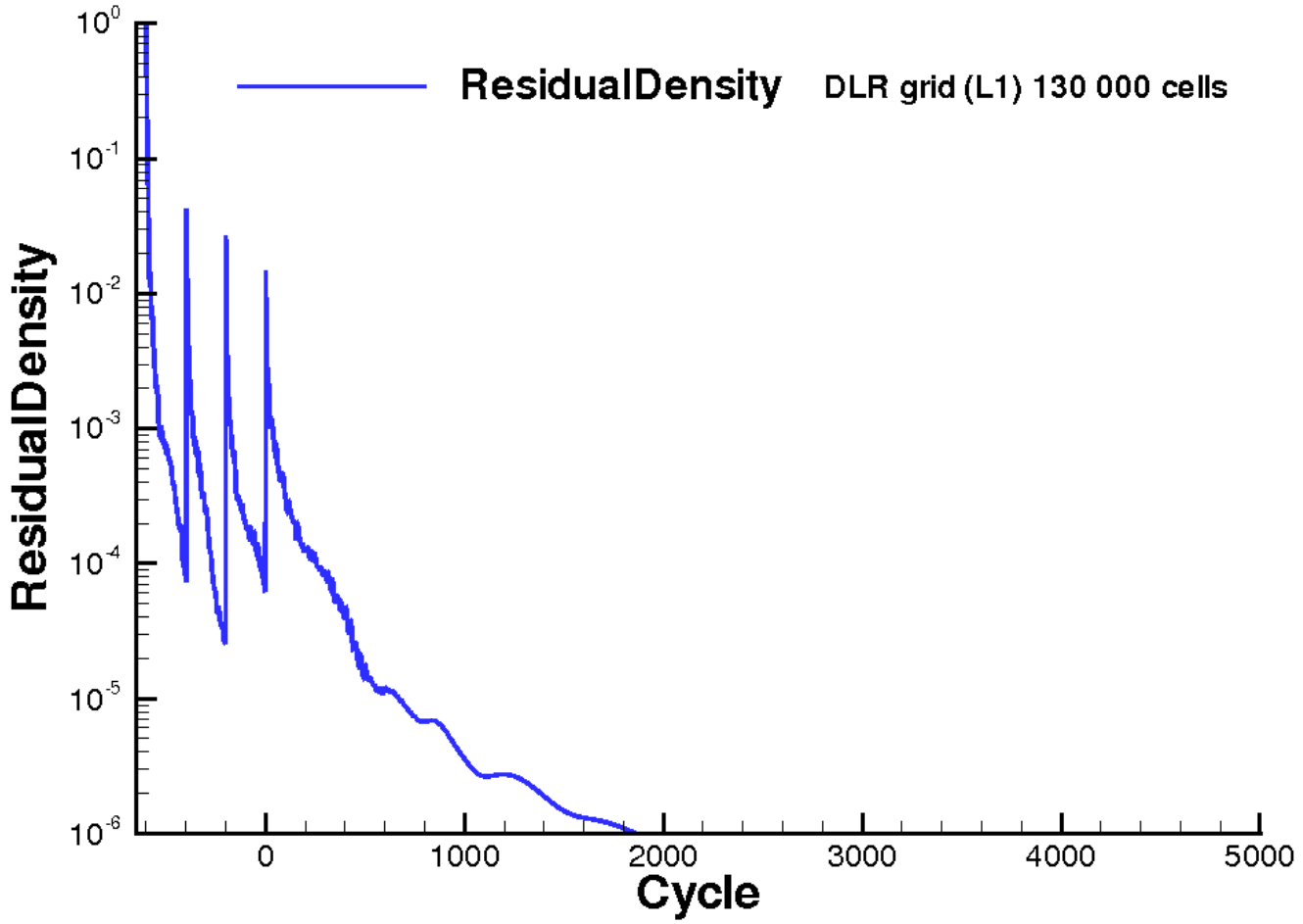




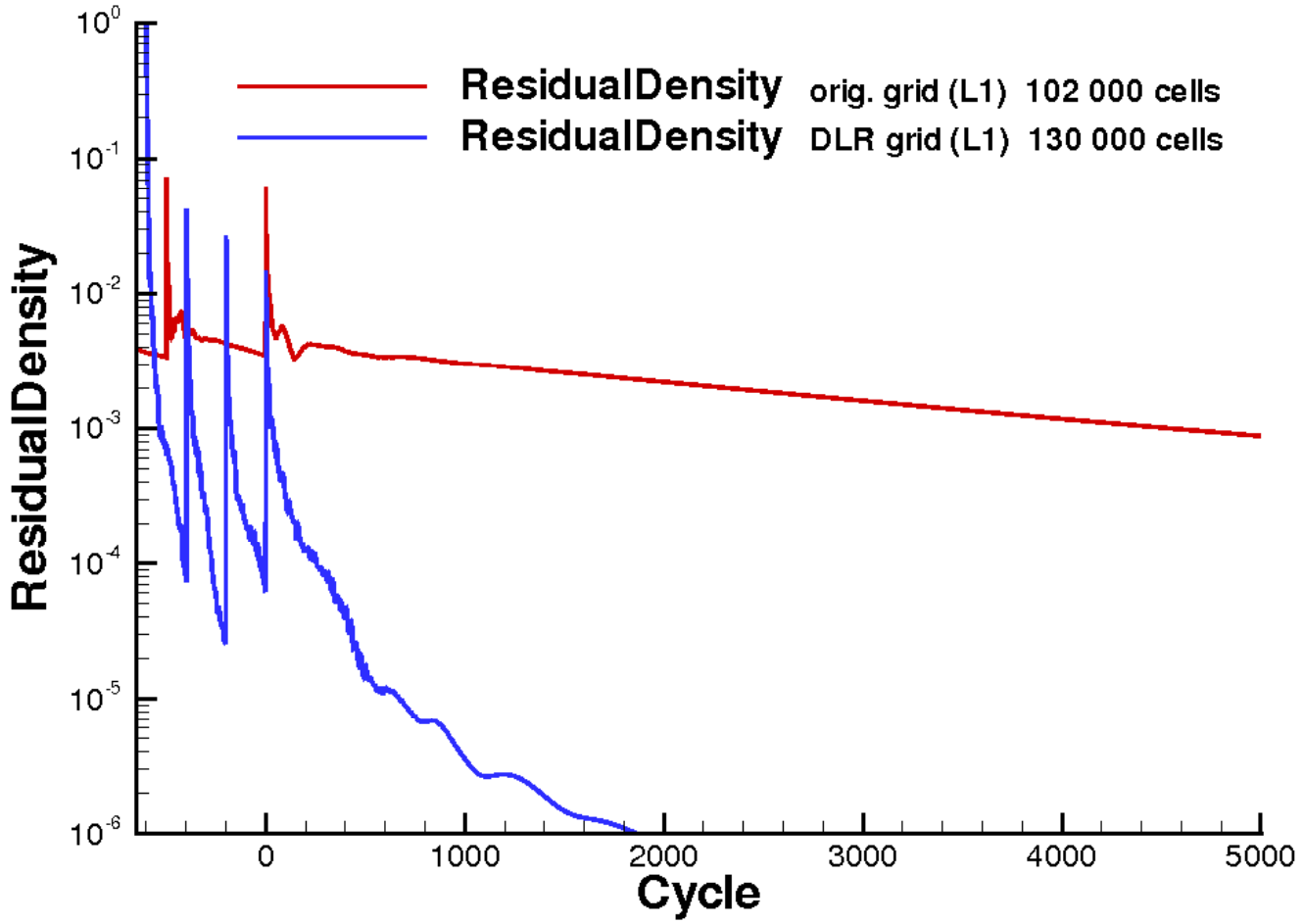
impact of the computational grid - example



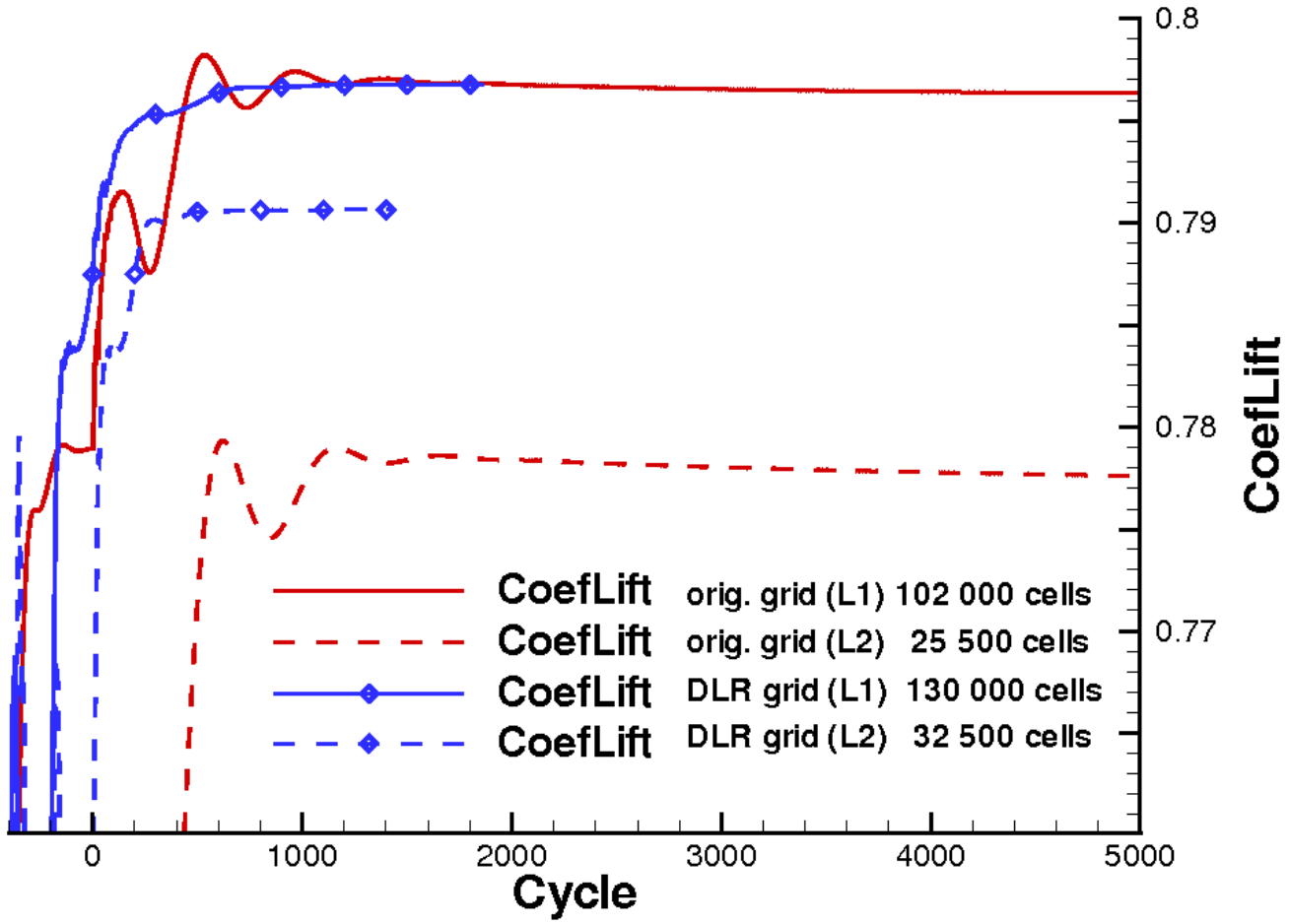
impact of the computational grid - example



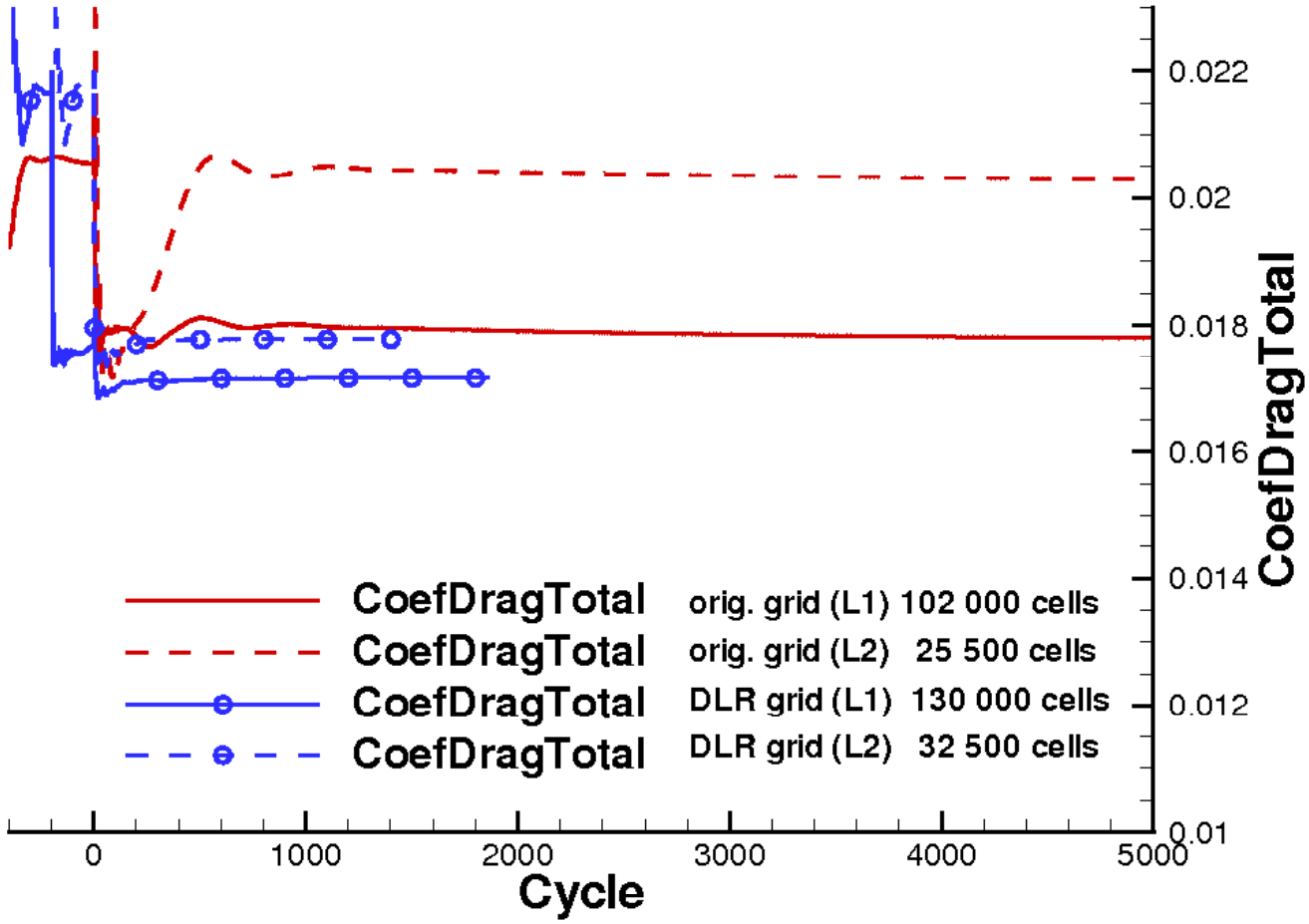
impact of the computational grid - example



impact of the computational grid - example



impact of the computational grid - example



impact of the computational grid - conclusion

- appropriate grid required
  - grid resolution
  - topology
  - smooth

- for comparison of different “cases”
  - ➔ use similar grids

- RANS solvers can’t be used like a “black box”

FLOWer – installation requirements

- **Software requirements**
  - LINUX / UNIX
  - MPI for parallel applictions
  - Tecplot for postprocessing (not mandatory)
  - F90 compiler
  - CPP preprocessor

- **Hardware requirements**

  depends on grids size:

  3D: $1 * 10^6$ cells → about 1 GB main memory

  2D: $1 * 10^6$ cells → about 2 -3 GB main memory

FLOWer – input files

- **input files for standard applications**

mandatory:

- **grid data**	default names:	**grdf.dat / grdu.dat**
- **grid topology data**	default name:	**log.dat**
- **control data**	default name:	**inp.dat**

optional:

- **plot control data**	default name:	**plt.dat**
- **file specifications**	mandatory name:	**files.dat**
- controlling the running code	mandatory name:	control

FLOWer – output files

- **output files for standard applications**

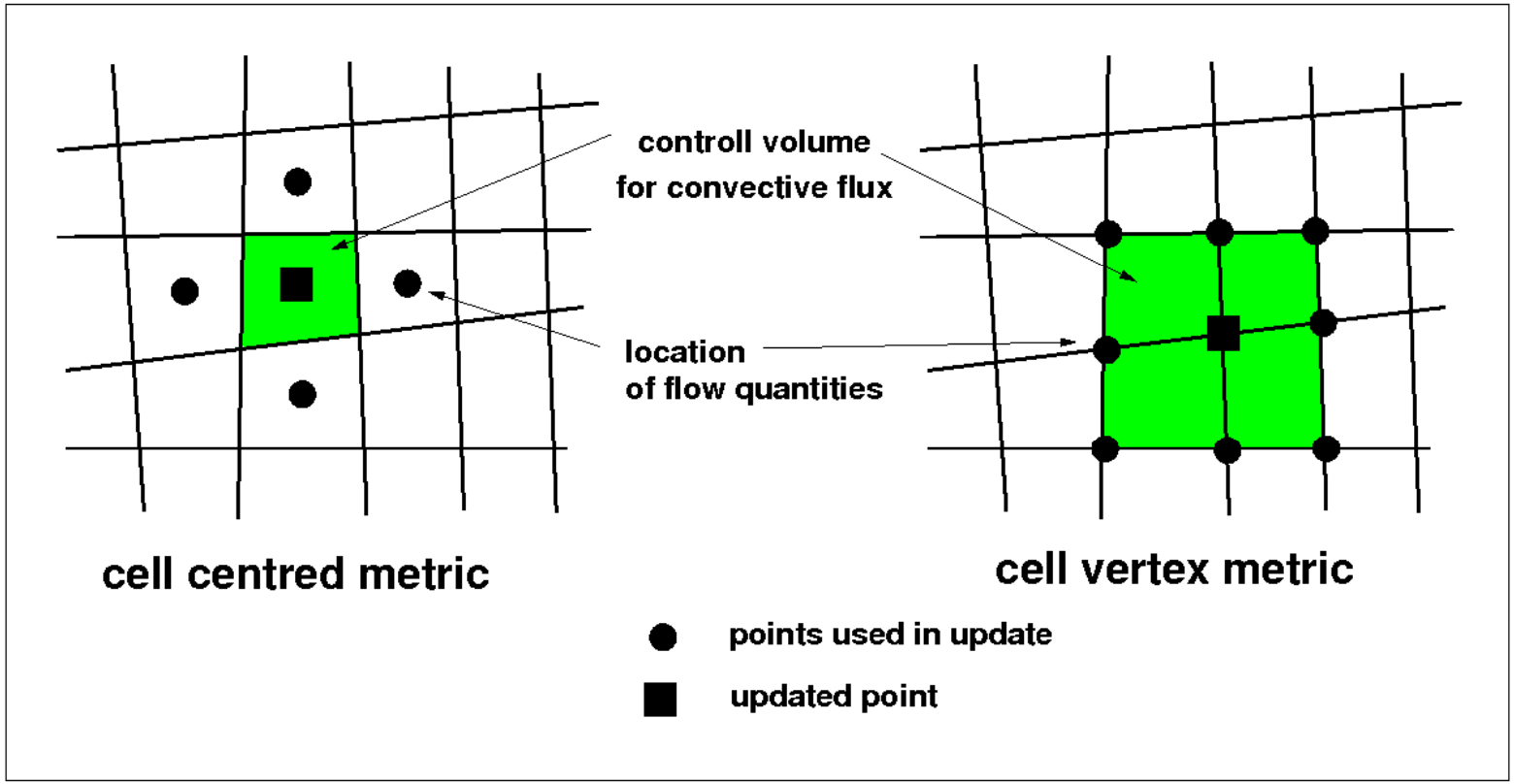
mandatory:

- **standard output**
- **conv.tec** convergence data
- **restart** restart data
- **inpinfo** information on control data used

optional (controlled by plt.dat)

- **surf.tec** surface data
- **plan.tec** output on index planes specified in plt.dat
- **field3d.tec** 3D data

FLOWer – spatial discretization



- recommended: cell centered metric

  input: IDISTYPE = 2 (that's the default)

FLOWer – discretization stencil

convective terms

- central scheme with artificial dissipation
  recommended for subsonic and transonic flows
- upwind schemes (only for cell centred metric)
  - flux vector splitting (AUSM scheme)
  - flux difference splitting (currently no scheme implemented)

viscous terms

- central scheme with artificial dissipation
- recommended: “full thin layer” approximation (input: ITLNS = 0)

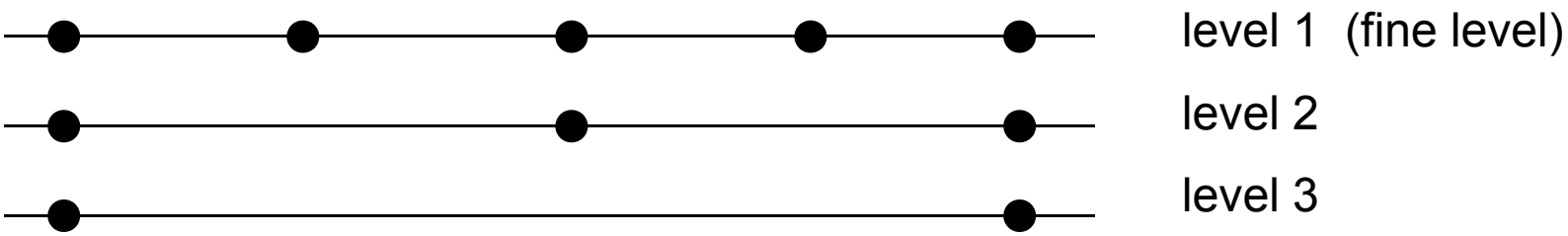
FLOWer – time stepping scheme for steady flows

- **explicit, 5 stage Runge-Kutta time stepping scheme**
  - for main equations
  - for turbulence equations (available but not recommended)
- only for turbulence equations:
  implicit DD-ADI scheme

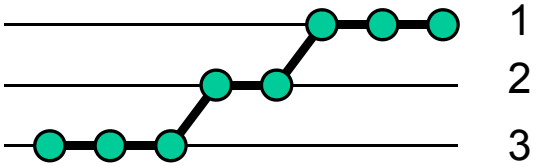
- **acceleration techniques**
  - local time steps
  - enthalpy damping (only usable for inviscid calculations
    input: HM = 0.01 ... 0.2)
  - implicit residual smoothing
  - multigrid

FLOWer – multigrid

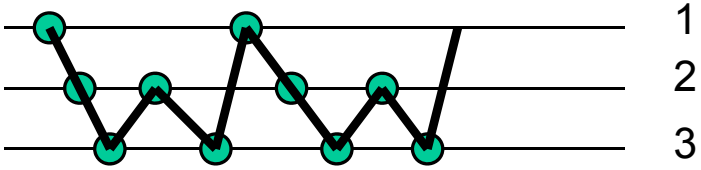
use of coarser grid levels to speed up iteration process



- **grid refinement**



- **multigrid**



- **full multigrid** : combination of grid refinement and multigrid

FLOWer – turbulence modeling – general remarks

- **no wall functions available**
- boundary layer must be resolved by 20 – 80 cells normal to the wall
- Y+ = 1 ... 2 (Y+ depends on first wall distance)

- **use implicit time integration (DD-ADI) for turbulence equations**
  input: TUSCHEME = 4 (that's the default)

- general remark concerning use of turbulence models in FLOWer
  >>> don't modify the input parameters of the turbulence models
  >>> use the default values for the turbulence models

FLOWer – turbulence modeling

- **algebraic models**

Baldwin Lomax	ITURB = 1
Baldwin Lomax with Degani Schiff modification	ITURB = 2

- robust
- fast convergence
- usable for flows without separation
- **grid dependant - only usable for simple grid topologies**

FLOWer – turbulence modeling

- **transport equation models (1 eqn)**
  - Spalart Allmaras — ITURB = 11
  - Spalart Allmaras with Edwards modification — ITURB = 12
  - SALSA — ITURB = 13

  - require general wall distance
  - SA is the standard model in many codes, but NOT in FLOWer
  - very few experience with SALSA

FLOWer – turbulence modeling

- **transport equation models (2 eqns.)**

- TNT k-ω	**ITURB = 22**	slightly improved Wilcox k-ω
- Wilcox k-ω	**ITURB = 23**	**standard model in FLOWer**
- LLR k-ω	ITURB = 24	don't use
- SST k-ω (Menter)	**ITURB = 25**	**well suited for many cases**
- LEA	**ITURB = 26**	similar to Wilcox k-ω (better results for transonic flow)
- Wallin EARSM	ITURB = 27	bad convergence

- **2-eqn models (Wilcox k-ω and k-ω SST are the standard turbulence models in FLOWer**
- SST k-ω requires general wall distance
- Wilcox k-w (ITURB = 23): set BCTURBKW = 2 (B.C. according to Menter)

FLOWer – turbulence modeling

- **Reynold stress models (7 eqns.)**
  - SSG according to Speziale, Sarkar, Gatski — ITURB = 71
  - Wilcox RSM — ITURB = 72
  - SSG – ω – model (combination of ITURB = 71/72) — ITURB = 73

- use of ITURB = 71 is not recommended

- **SSG – ω – model (ITURB = 73) is the best model in FLOWer but very expensive**
  - high computational effort
  - high memory requirements
  - slow convergence

generation of grid topology file “log.dat”

- **log.dat contains:**
  - **boundary conditions**
  - **block to block connectivity**

- **log.dat is a human readable ASCII file**

- **3 ways to generate log.dat :**
  1. **use POPINDA output of grid generator ICEM – HEXA**
  2. **use FLOWer tool ‘logic’**
     - input: grid data file + control file
     - detects block to block connectivity
     - sets boundary conditions depending on grid data
     → **output must be controlled by the user**
  3. **generate log.dat by the user → not recommended**

further informations concerning FLOWer

- DLR ftp-server

  ftp.dlr.de

  user: benchmark

  password: <ask Stefan Melber>