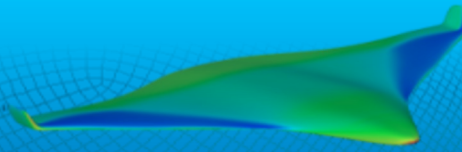




R.Tech
Research &
Technology



$$\begin{aligned} \frac{\partial z}{\partial y} + \text{div } v &= \\ + \text{div}(\rho u) &= -\frac{\partial p}{\partial z} + \text{div}(\mu \text{ grad } w) + S_M \\ \text{div}(\rho w) &= -\rho \text{ div } u + \text{div}(k \text{ grad } T) + \Phi + S_V \end{aligned}$$



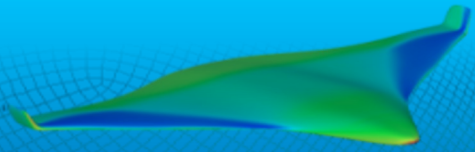
```
for (int i=0; i<grid.nx; i++)
  for (int j=0; j<grid.ny; j++)
    for (int k=0; k<grid.nz; k++)
      // ...
    }
  }
}
```

Extension of ESA's Survival And Risk Analysis tool with hemisphere and lattice shapes

Clean Space Industry Days 2023, ESA-ESTEC, Noordwijk, Netherlands



$$\frac{\partial \rho}{\partial t} + \text{div}(\rho \mathbf{u}) = -\rho \text{div} \mathbf{u} + \text{div}(k \text{ grad } T) + \Phi + S_v$$
$$\frac{\partial p}{\partial t} + \text{div}(\mu \text{ grad } w) + S_M$$
$$\frac{\partial \zeta}{\partial y} + \text{div} v =$$



```
for (int i=0; i<grid.nx; i++)  
{  
  for (int j=0; j<grid.ny; j++)  
{  
    for (int k=0; k<grid.nz; k++)  
{  
      // compute the value of the variable at each point  
      // of the grid  
      double v = ...  
      // store the value in the array  
      v[i][j][k] = v;  
    }  
  }  
}
```

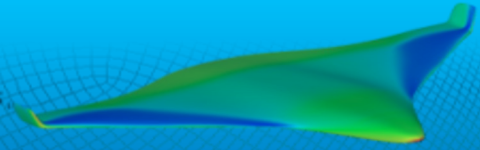
The *Debris Risk Assessment and Mitigation Analysis (DRAMA)* tool is the ESA certification tool for the risk assessment of destructive spacecraft re-entry

The *Survival and Risk Analysis (SARA)* module simulates the re-entry and predicts demise altitude and ground impact information (location, impact energy)

SARA models the spacecraft by a set of user defined primitives



$$\begin{aligned} \frac{\partial z}{\partial x} + \operatorname{div} v &= \\ \frac{\partial p}{\partial z} + \operatorname{div}(\mu \operatorname{grad} w) &+ S_M \\ \operatorname{div}(\rho w) &= -\rho \operatorname{div} \mathbf{u} + \operatorname{div}(k \operatorname{grad} T) + \Phi + S_p \end{aligned}$$



```
for (int i=0; i<nx; i++)
  for (int j=0; j<ny; j++)
    for (int k=0; k<nz; k++)
      S_M[i][j][k] = 0;
    for (int i=0; i<nx; i++)
      for (int j=0; j<ny; j++)
        for (int k=0; k<nz; k++)
          S_p[i][j][k] = 0;
    for (int i=0; i<nx; i++)
      for (int j=0; j<ny; j++)
        for (int k=0; k<nz; k++)
          S_M[i][j][k] = 0;
    for (int i=0; i<nx; i++)
      for (int j=0; j<ny; j++)
        for (int k=0; k<nz; k++)
          S_p[i][j][k] = 0;
    for (int i=0; i<nx; i++)
      for (int j=0; j<ny; j++)
        for (int k=0; k<nz; k++)
          S_M[i][j][k] = 0;
    for (int i=0; i<nx; i++)
      for (int j=0; j<ny; j++)
        for (int k=0; k<nz; k++)
          S_p[i][j][k] = 0;
```

The primitives used in SARA are:

- Box
- Cone
- Sphere
- Cylinder
- Ring



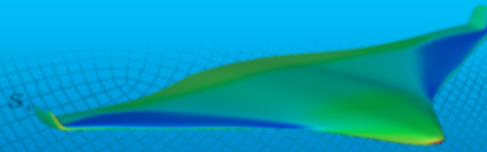
Context



Which of the five primitives to model for example a parabola on a satellite, or a fuel reservoir breaking up?

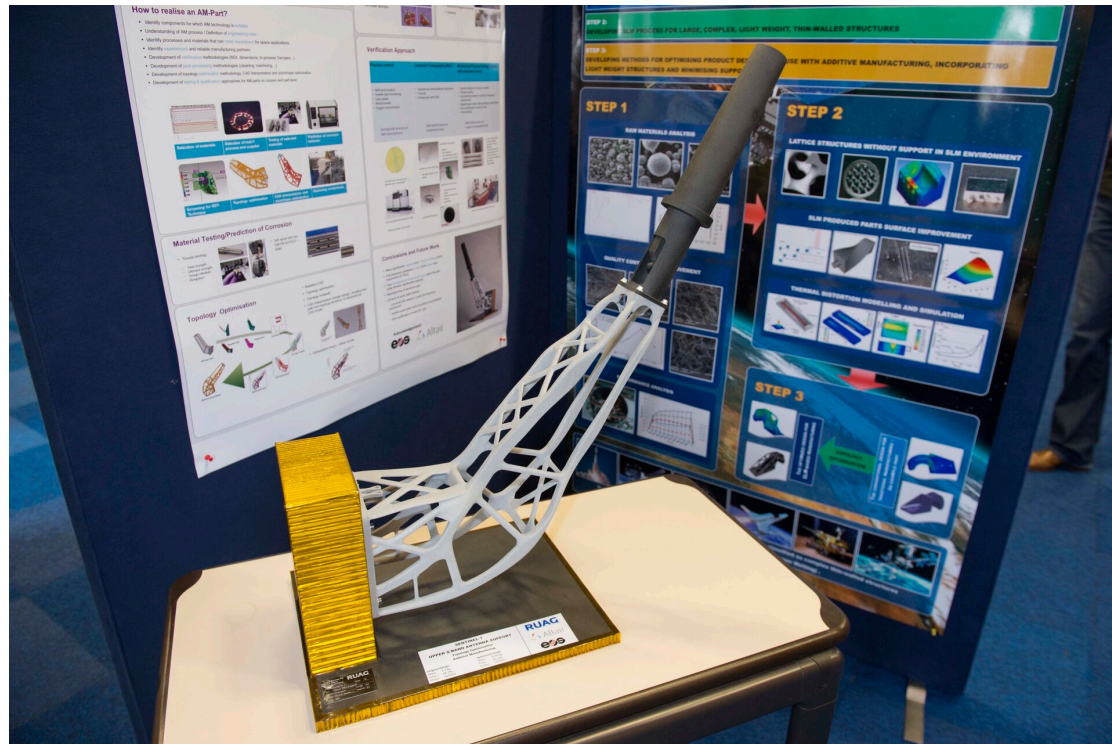


$$\nabla \cdot (\mu \text{ grad } w) + S_M$$
$$\text{div}(\rho w) = -\rho \text{ div } u + \text{div}(k \text{ grad } T) + \Phi + S_T$$

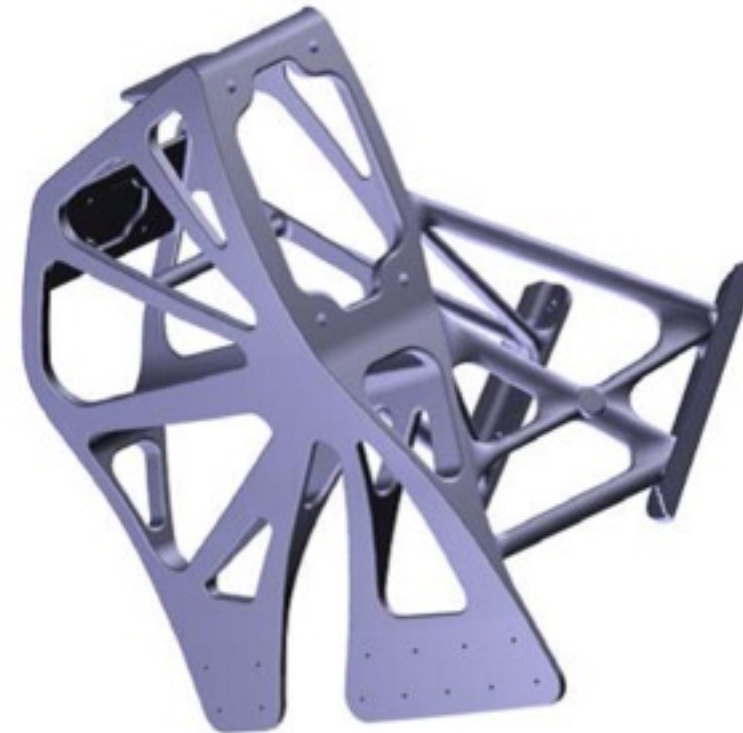


```
for j=1:200  
    [x,y,z] = meshgrid(1:200,1:200,1:200);  
    z = 0.5 * (x.^2 + y.^2);  
end
```

Or more complex: lattice shapes produced using added manufacturing



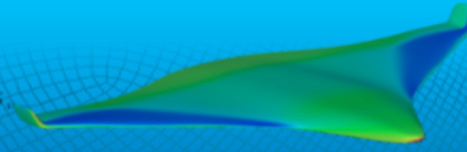
3D printed antenna support (courtesy ESA)



Ariane bracket (courtesy ESA)

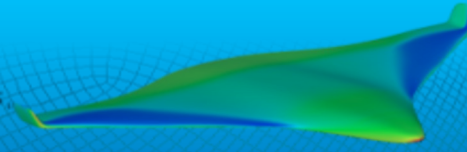


Objectives



Objectives:

- To implement new concave shape (Hollow Hemisphere) -> user need
- To implement ways to model lattice structures -> experimental prototype
- To provide guidelines for modelling lattice structures -> user need



Objectives:

- To implement new concave shape (Hollow Hemisphere)
 - > add sphere cap (hollow hemisphere) primitive
- To implement ways to model lattice structures
 - > add a user shape primitive
- To provide guidelines for modelling lattice structures
 - > think, think, think



For each drama shape implementation we need:

- Aerodynamic coefficients as a function of attitude and flow conditions
- Aerothermal heating rates as a function of attitude and flow conditions
- The effect of shading of the shape on another object to simulate 'blocking' of the external flow on component based object
- A means to assess the outer surface when ablation occurs



For each drama primitive implementation we need:

- 1) Aerodynamic coefficients as a function of attitude and flow conditions
- 2) Aerothermal heating rates as a function of attitude and flow conditions



Use of existing database capabilities

- For the existing primitives the database is produced with simplified tools
- For the new primitives we perform a CFD matrix in a similar way as has been done since 2016 for the DEBRISK software. One objective is to harmonize the databases of the two tools.



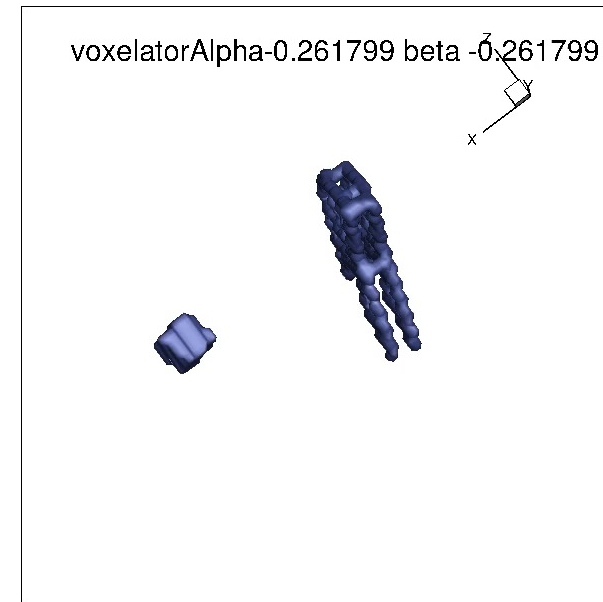
For each drama primitive implementation we need:

3) The effect of shading of the shape on another object to simulate 'blocking' of the external flow on component based objects



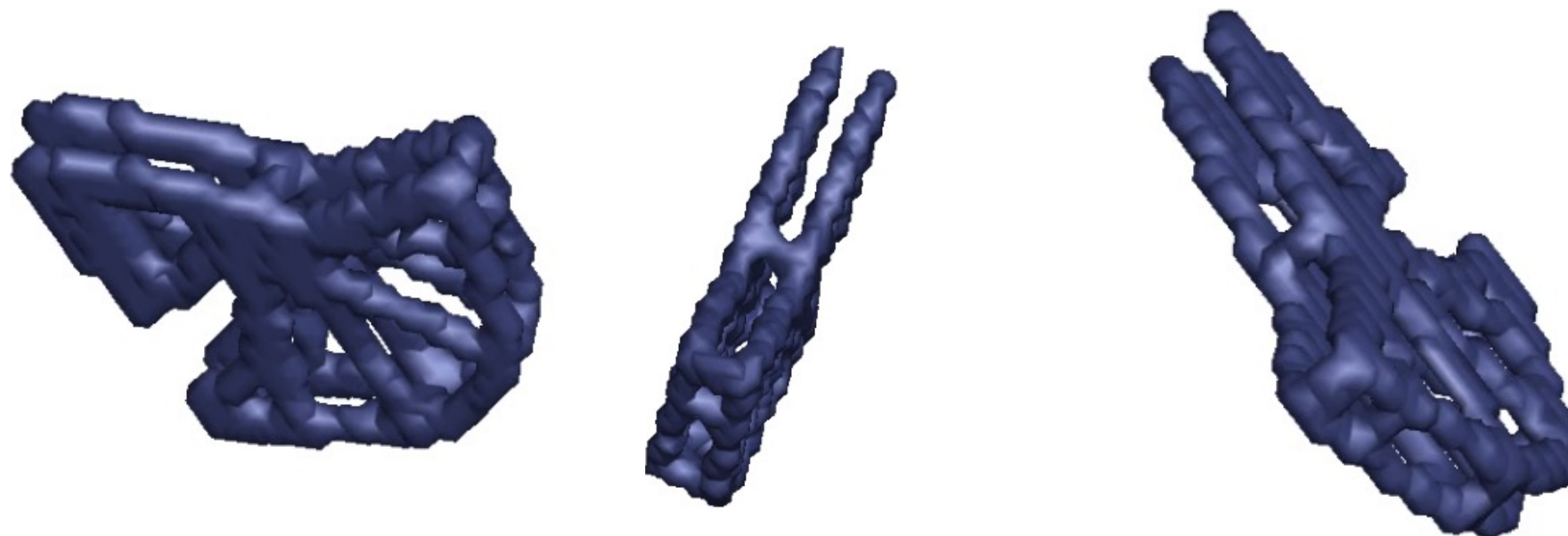
Use of existing voxelator tool

- For the existing primitives the voxelation can be performed with analytical relations
- For the sphere cap the voxelation can be performed with analytical relations
- For the User Shapes the voxelation is performed numerically. The primitive needs to be supplied as a discretized shape





Voxelator for User Shapes



- > Voxelator use for arbitrary geometry
- > Objective : Determinate the shading factor for user shapes and projected area
- > Input : Point cloud. The resolution need to be finer than the voxelator resolution. VTP format.

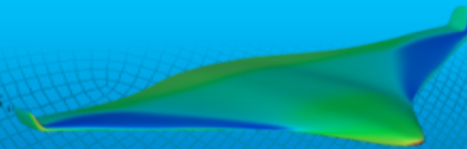


For each drama primitive implementation we need:

4) A means to assess the outer surface when ablation occurs



- For the existing primitives the relation between ablated mass and outer surface can be computed analytically
- For the sphere cap the relation between ablated mass and outer surface can be computed analytically
- For the User Shapes the relation between ablated mass and outer surface is performed numerically. The primitive needs to be supplied as a discretized volume mesh.

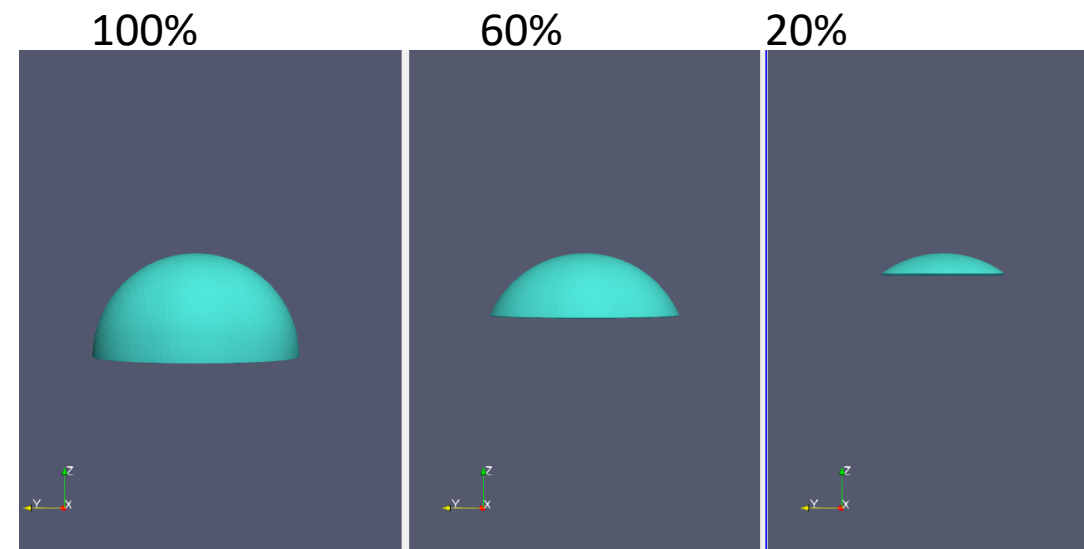


```
for (int i=0; i<nbNodes; i++)  
{  
  double x=nodes[i].x; double y=nodes[i].y; double z=nodes[i].z;  
  double nx=nodes[i].nx; double ny=nodes[i].ny; double nz=nodes[i].nz;  
  double dx=nodes[i].dx; double dy=nodes[i].dy; double dz=nodes[i].dz;  
  double dV=dx*dy*dz;  
  double dS=0.5*(nx*ny+nz*nx+nz*ny);  
  double dW=0.5*(nx*ny+nz*nx+nz*ny);  
  double dM=0.5*(nx*ny+nz*nx+nz*ny);  
  double dR=0.5*(nx*ny+nz*nx+nz*ny);  
  double dI=0.5*(nx*ny+nz*nx+nz*ny);  
  double dJ=0.5*(nx*ny+nz*nx+nz*ny);  
  double dK=0.5*(nx*ny+nz*nx+nz*ny);  
  double dL=0.5*(nx*ny+nz*nx+nz*ny);  
  double dM=0.5*(nx*ny+nz*nx+nz*ny);  
  double dR=0.5*(nx*ny+nz*nx+nz*ny);  
  double dI=0.5*(nx*ny+nz*nx+nz*ny);  
  double dJ=0.5*(nx*ny+nz*nx+nz*ny);  
  double dK=0.5*(nx*ny+nz*nx+nz*ny);  
  double dL=0.5*(nx*ny+nz*nx+nz*ny);  
}
```

Ablation by height since highest fluxes are on the rim (for random tumbling)

Definition percentage of height:

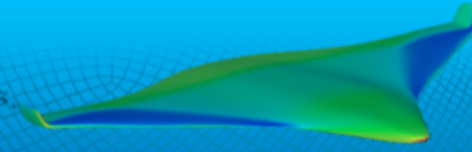
Geometry :



Attitudes:19

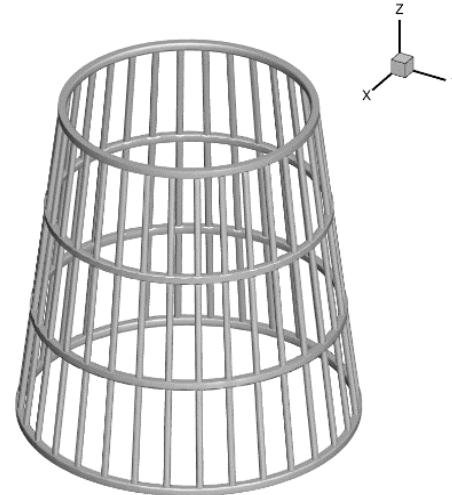
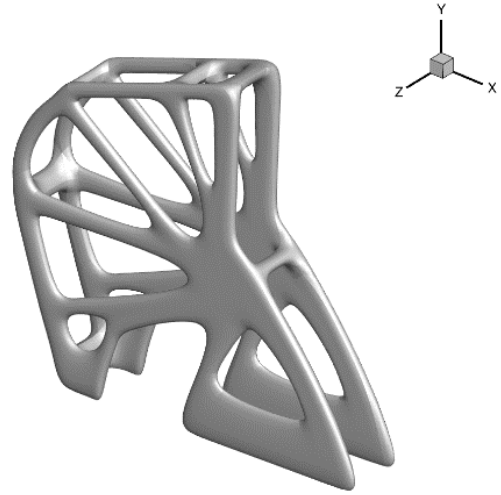


CFD databases



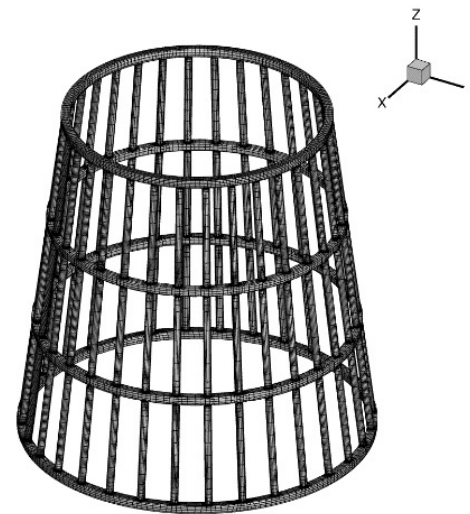
```
for (int i=0; i<mesh.nNodes(); i++)
{
  double x=mesh.getNodeX(i);
  double y=mesh.getNodeY(i);
  double z=mesh.getNodeZ(i);
  double u=mesh.getNodeU(i);
  double v=mesh.getNodeV(i);
  double w=mesh.getNodeW(i);
  double p=mesh.getNodeP(i);
  double rho=mesh.getNodeRho(i);
  double mu=mesh.getNodeMu(i);
  double nu=mesh.getNodeNu(i);
  double k=mesh.getNodeK(i);
  double epsilon=mesh.getNodeEpsilon(i);
  double omega=mesh.getNodeOmega(i);
  double gamma=mesh.getNodeGamma(i);
  double beta=mesh.getNodeBeta(i);
  double alpha=mesh.getNodeAlpha(i);
  double delta=mesh.getNodeDelta(i);
  double theta=mesh.getNodeTheta(i);
  double phi=mesh.getNodePhi(i);
  double chi=mesh.getNodeChi(i);
  double psi=mesh.getNodePsi(i);
  double xi=mesh.getNodeXi(i);
  double eta=mesh.getNodeEta(i);
  double zeta=mesh.getNodeZeta(i);
  double theta=mesh.getNodeTheta(i);
  double phi=mesh.getNodePhi(i);
  double chi=mesh.getNodeChi(i);
  double psi=mesh.getNodePsi(i);
  double xi=mesh.getNodeXi(i);
  double eta=mesh.getNodeEta(i);
  double zeta=mesh.getNodeZeta(i);
}
```

Geometry :



Attitudes: 50

3D volume
Mesh :





CFD/DSMC Simulations

Geometry	Mach	Height/Radius ratio	Simulation/Condition	Total simulations
BRACKET	5, 20, 22.85	X	50	150
LATTICE	5, 20, 22.85	X	50	150
HOLLOW HEMISPHERE	5, 20, 22.85	1(0.5/0.5), 0.6(0.3/0.5), 0.2(0.1/0.5)	19	171

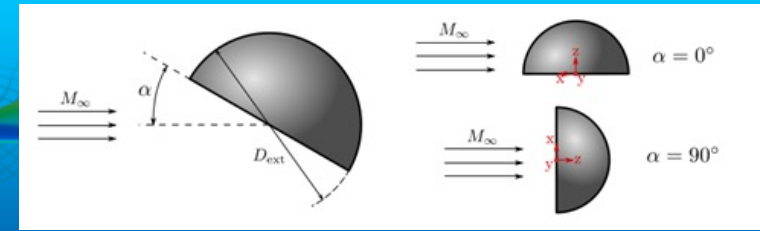
-> Unsteady cases for some Hollow Hemisphere simulation cases.

Conditions have been chosen based on preliminary Pampero trajectories in the continuum and rarefied regime.

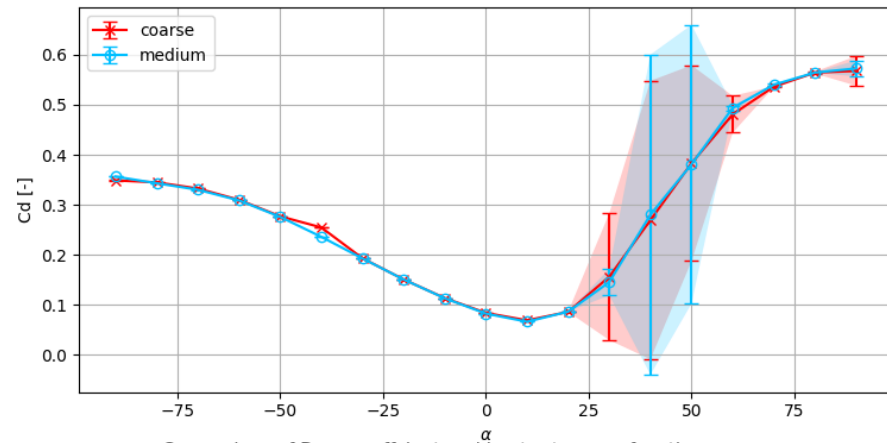
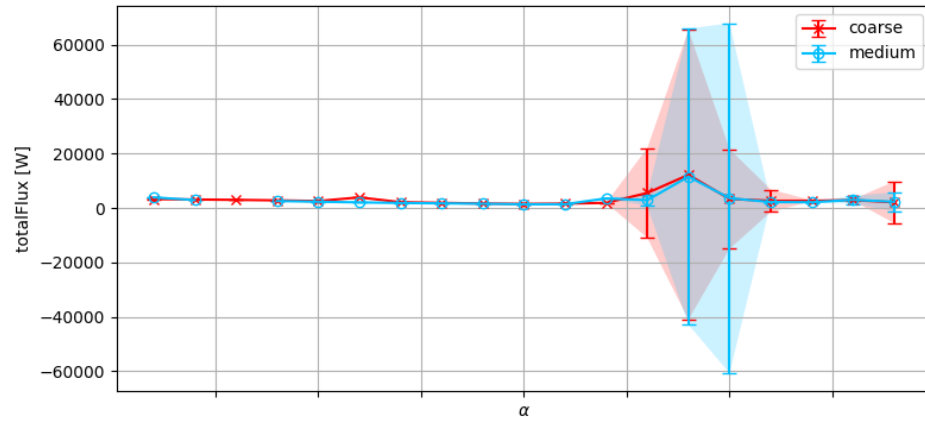
2x CFD matrix, 1x DSMC matrix



HOLLOW HEMISPHERE CFD

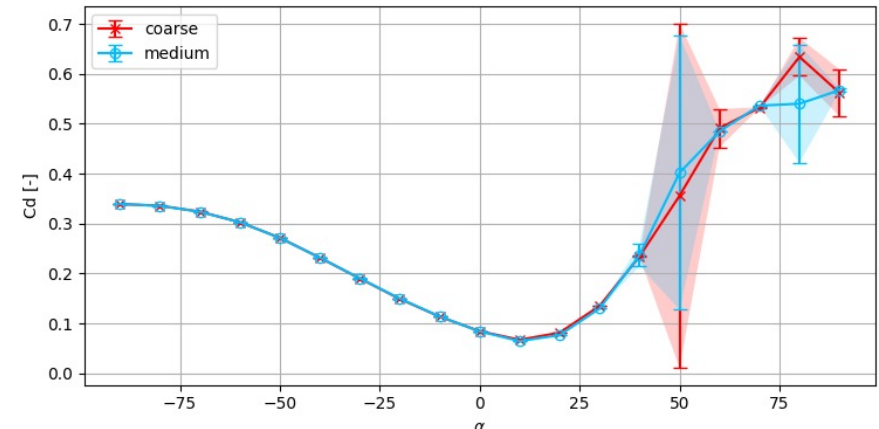
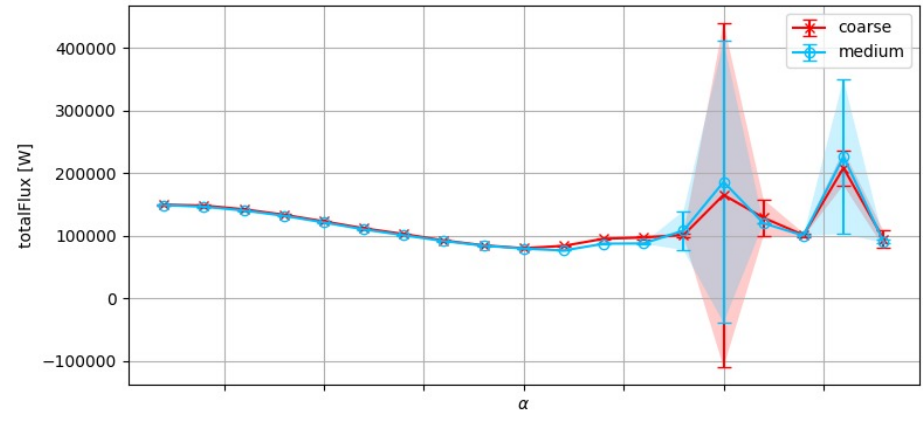


60 % MACH 5



Comparison of Drag coefficient and heat rates as a function of the angle or attack α for different mesh refinements
Hollow hemisphere 60% of height M5

60 % MACH 20

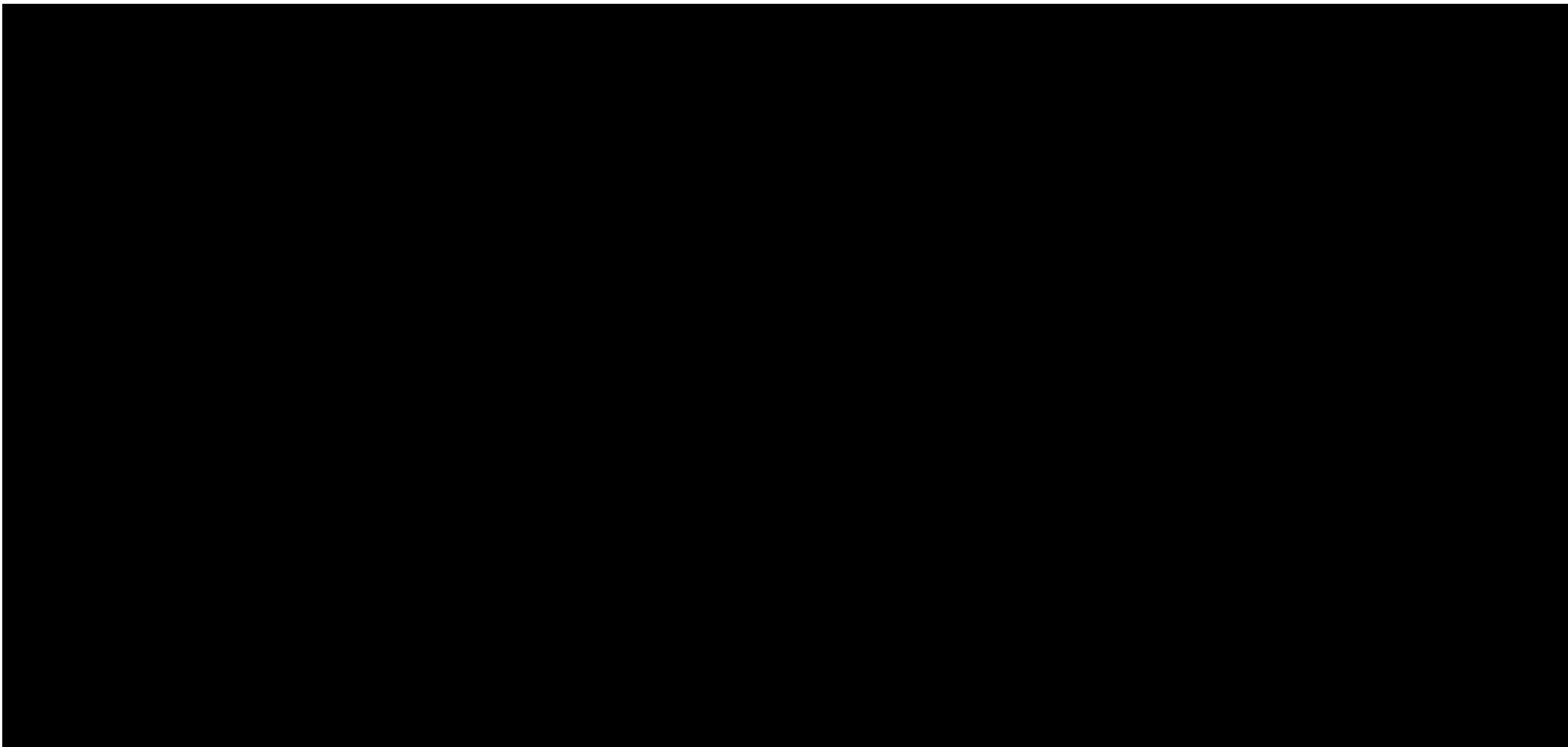
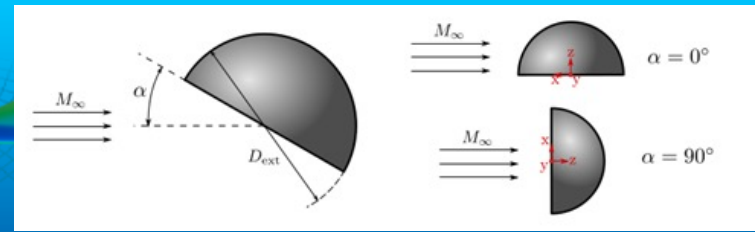


Comparison of Drag coefficient and heat rates as a function of the angle or attack α for different mesh refinements
Hollow hemisphere 60% of height M20



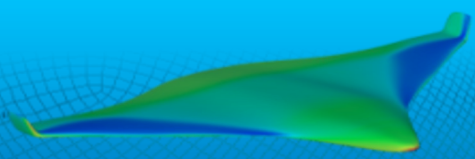
HOLLOW HEMISPHERE CFD

$$\frac{\partial p}{\partial y} + \text{div}(\mu \text{ grad } w) + S_M$$
$$\text{div}(\rho w) = -\rho \text{ div } u + \text{div}(k \text{ grad } T) + \Phi + S_v$$



$$\frac{\partial z}{\partial y} + \text{div}(\rho \mathbf{u}) = -\frac{\partial p}{\partial z} + \text{div}(\mu \text{ grad } w) + S_M$$

$$\text{div}(\rho \mathbf{u}) = -\rho \text{div} \mathbf{u} + \text{div}(k \text{ grad } T) + \Phi + S_\rho$$

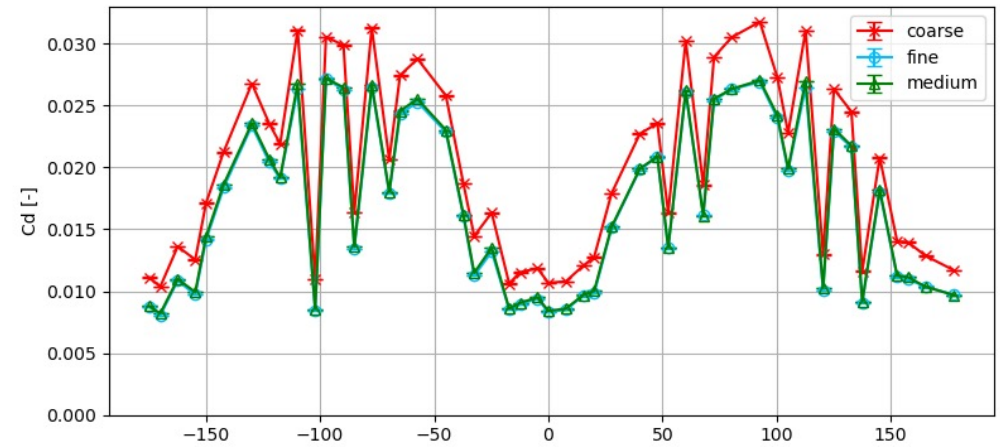
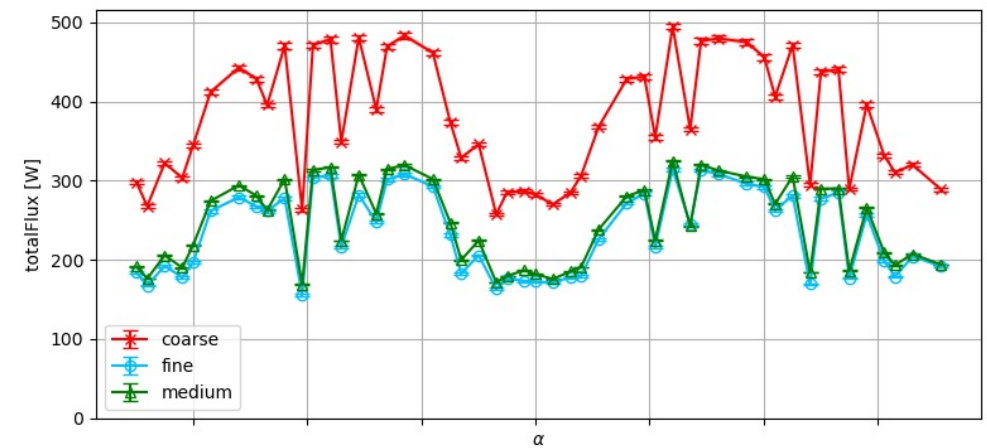


```

for (int i=0; i<nx; i++)
for (int j=0; j<ny; j++)
for (int k=0; k<nz; k++)
{
// compute the source term for species with respect to species
// ...
}
}
}
// ...
}
}
}

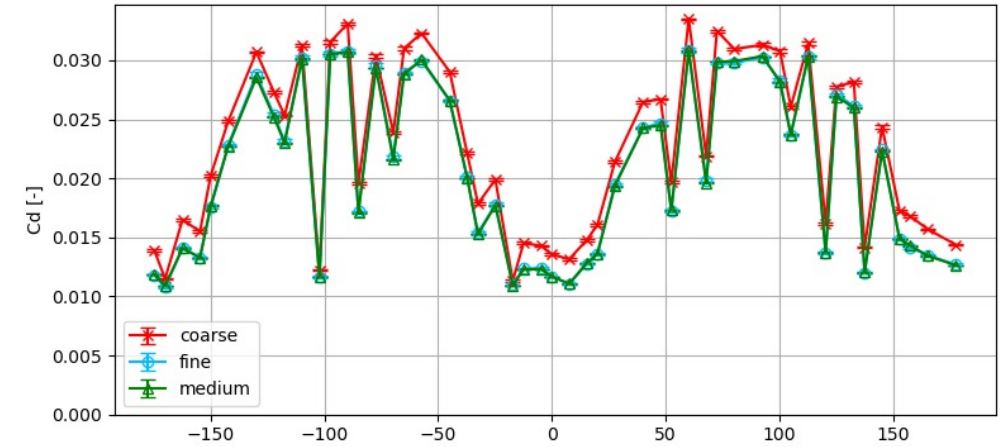
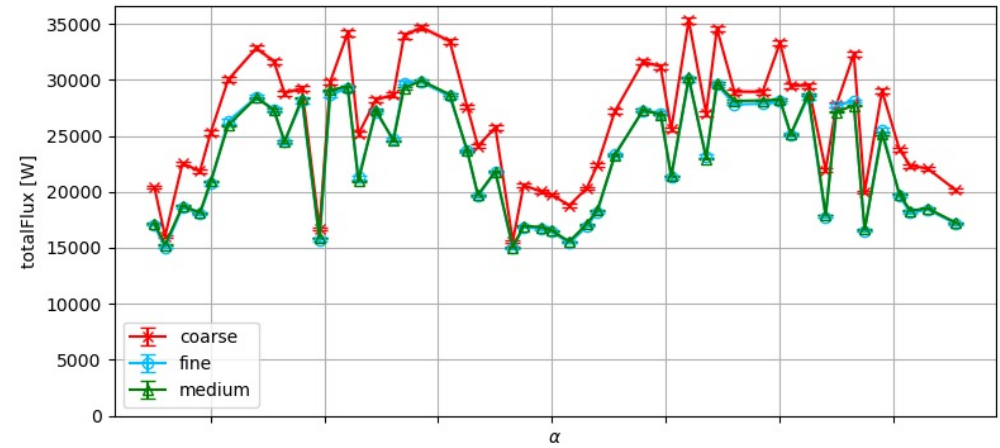
```

BRACKET MACH 5



Comparison of Drag coefficient and heat rates as a function of the angle or attack α for different mesh refinements bracket

BRACKET MACH 20

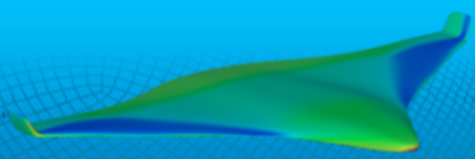


Comparison of Drag coefficient and heat rates as a function of the angle or attack α for different mesh refinements bracket



$$\frac{\partial \rho}{\partial t} + \text{div}(\rho \mathbf{u}) = -\frac{\partial p}{\partial z} + \text{div}(\mu \text{grad } w) + S_M$$

$$\text{div}(\rho \mathbf{u}) = -p \text{div } \mathbf{u} + \text{div}(k \text{grad } T) + \Phi + S_E$$

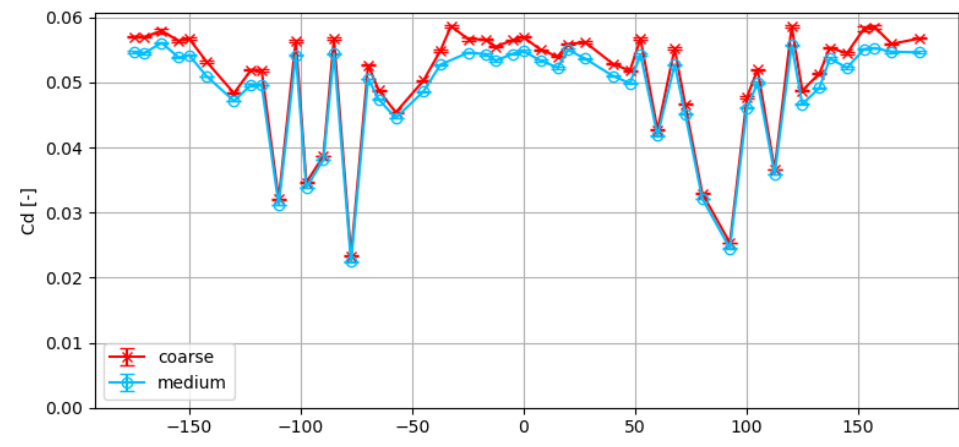
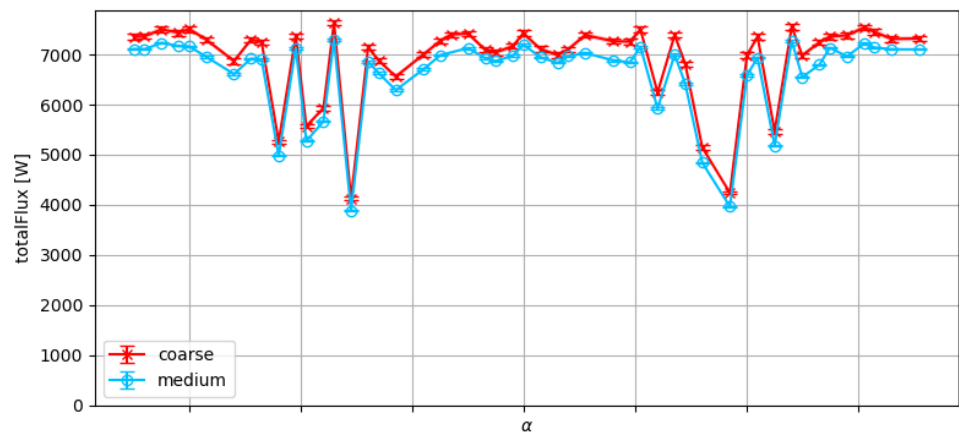


```

for (int i=0; i<nx; i++)
  for (int j=0; j<ny; j++)
    for (int k=0; k<nz; k++)
      // ...
    }
  }
}

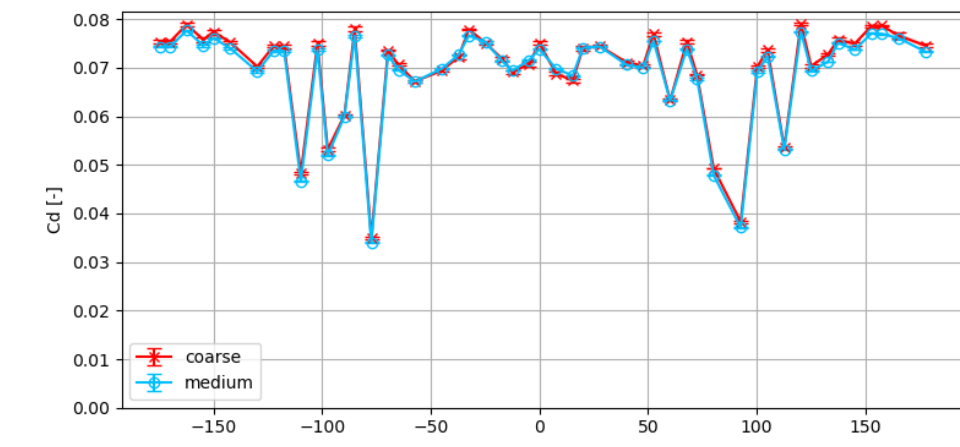
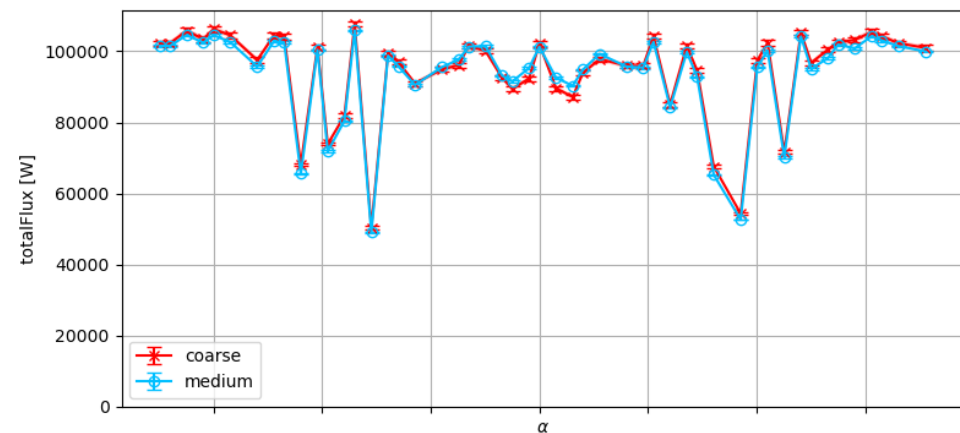
```

LATTICE MACH 5

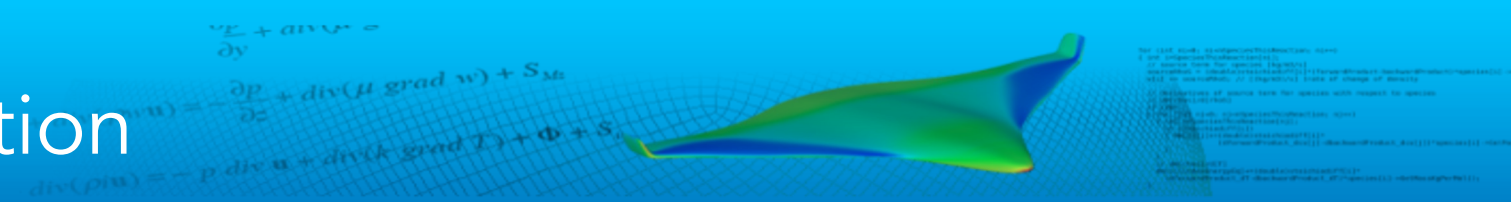


Comparison of Drag coefficient and heat rates as a function of the angle or attack α for different mesh refinements lattice

LATTICE MACH 20



Comparison of Drag coefficient and heat rates as a function of the angle or attack α for different mesh refinements lattice



Ablation management method for « UserShape » primitives



$$\frac{dS}{dV} \text{ netCDFfile} = Cst \quad S_{current} = Cst(V_{current} - V_{init}) + S_{init}$$



TEST VALIDATION

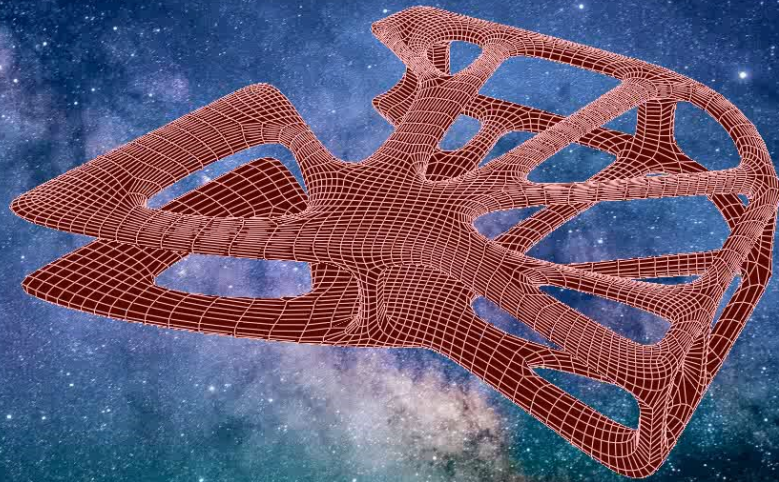
- Initial surface : 0.11156654 m²
- Initial Volume : 0.00017112769 m³
- Volume clipped : 7.563024E-5 m³
- Surface clipped : 0.0542456 m²

The surface calculated by DRAMA is 0.0552916 m², with an error of 1.8% with the surface calculated with Python.



Fragmentation of lattice shapes (Pampero)

Mass = 0.56 kg
Altitude = 89998.0 m

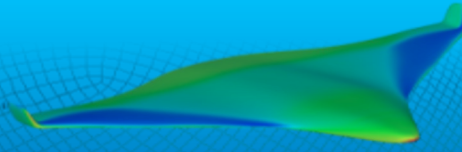


Mass = 0.367 kg
Altitude = 93998.0 m





$$\frac{\partial \rho}{\partial t} + \text{div}(\rho \mathbf{u}) = -\frac{\partial p}{\partial z} + \text{div}(\mu \text{ grad } w) + S_M$$
$$\text{div}(\rho \mathbf{u}) = -\rho \text{ div } \mathbf{u} + \text{div}(k \text{ grad } T) + \Phi + S_\rho$$



```
for (int i=0; i<mesh.nFaces(); i++)
{
  int fID = mesh.faces()[i];
  int fType = mesh.faceType(fID);
  if (fType == 0) continue; // skip internal faces
  if (fType == 1) continue; // skip boundary faces
  // ...
}
```

Conclusions

- Successfully implemented a sphere-cap primitive (HollowHemiSphere) with a CFD database
- Successfully implemented a new user shape primitive demonstrated with two lattice shapes
- Care should be taken with lattice shapes when ablation is occurring: the likelihood of fragmentation is high. If fragmentation occurs the use of higher fidelity tools is recommended.