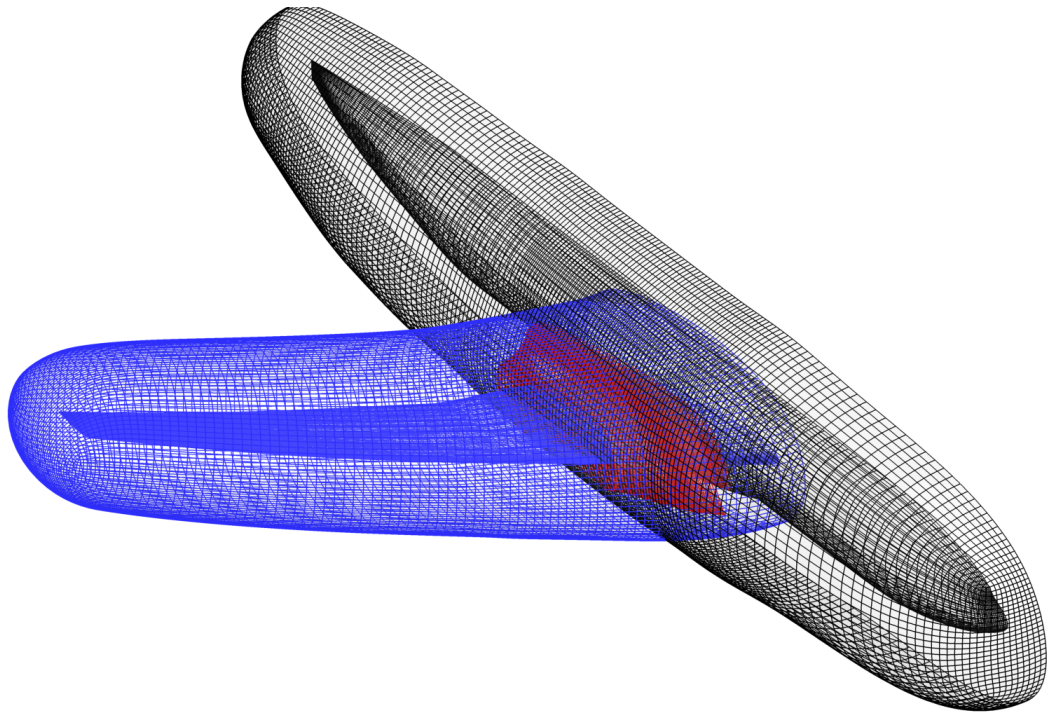
How to Use Overset Meshes in ADflow

Ney Secco

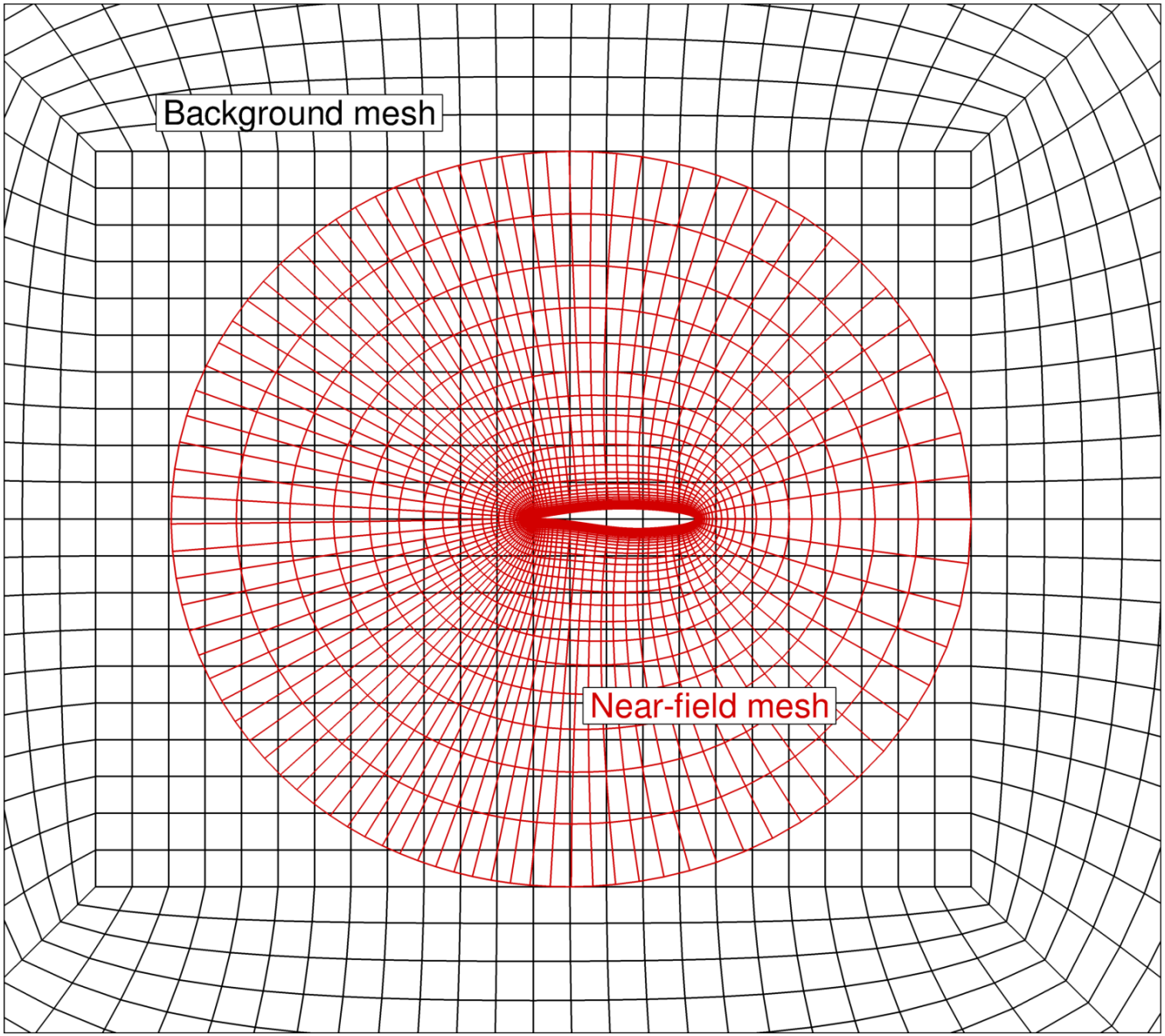
April 27, 2018



mdolab.engin.umich.edu

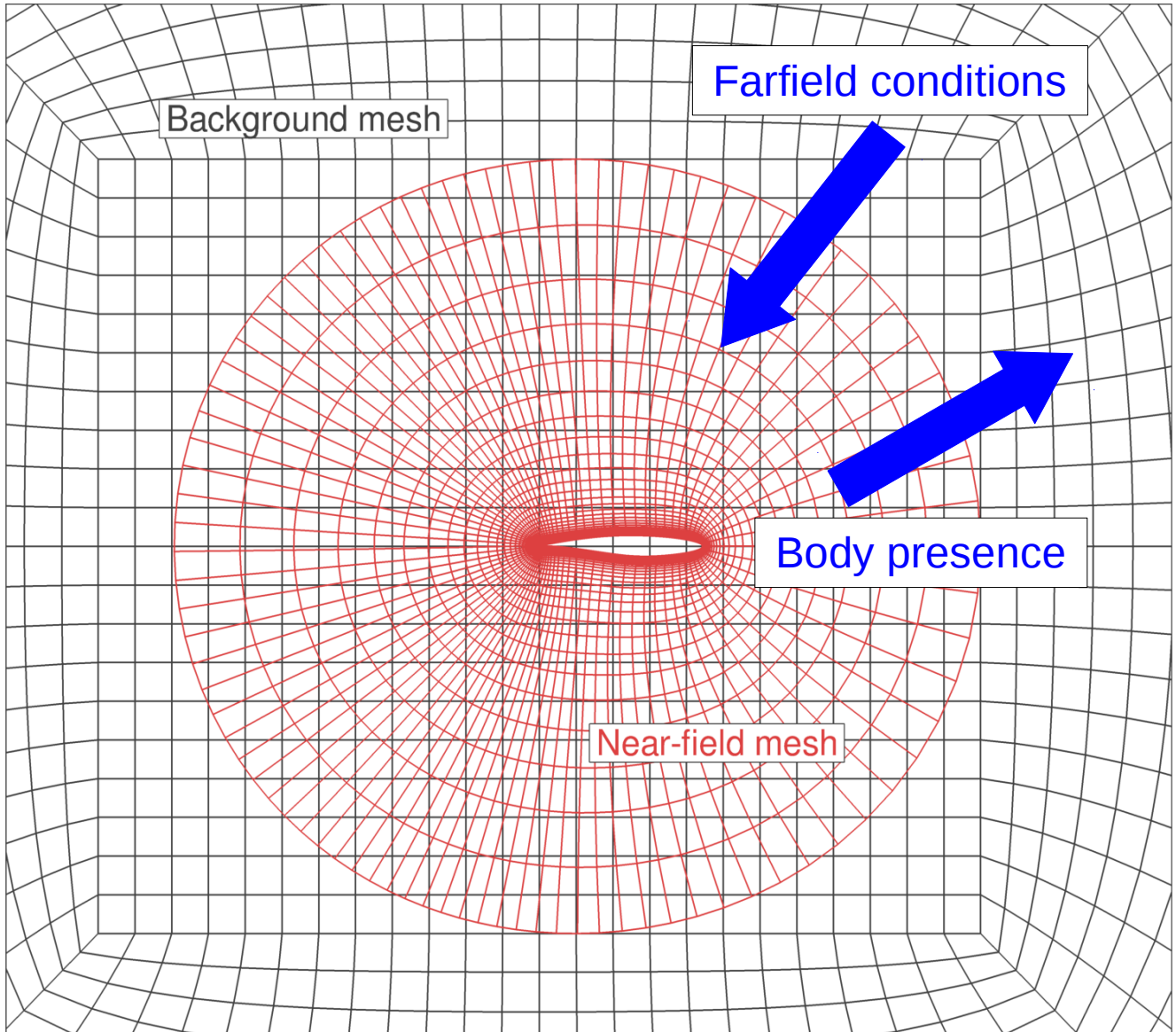
What are Overset Meshes?

Unstructured network of overlapping **structured** meshes.

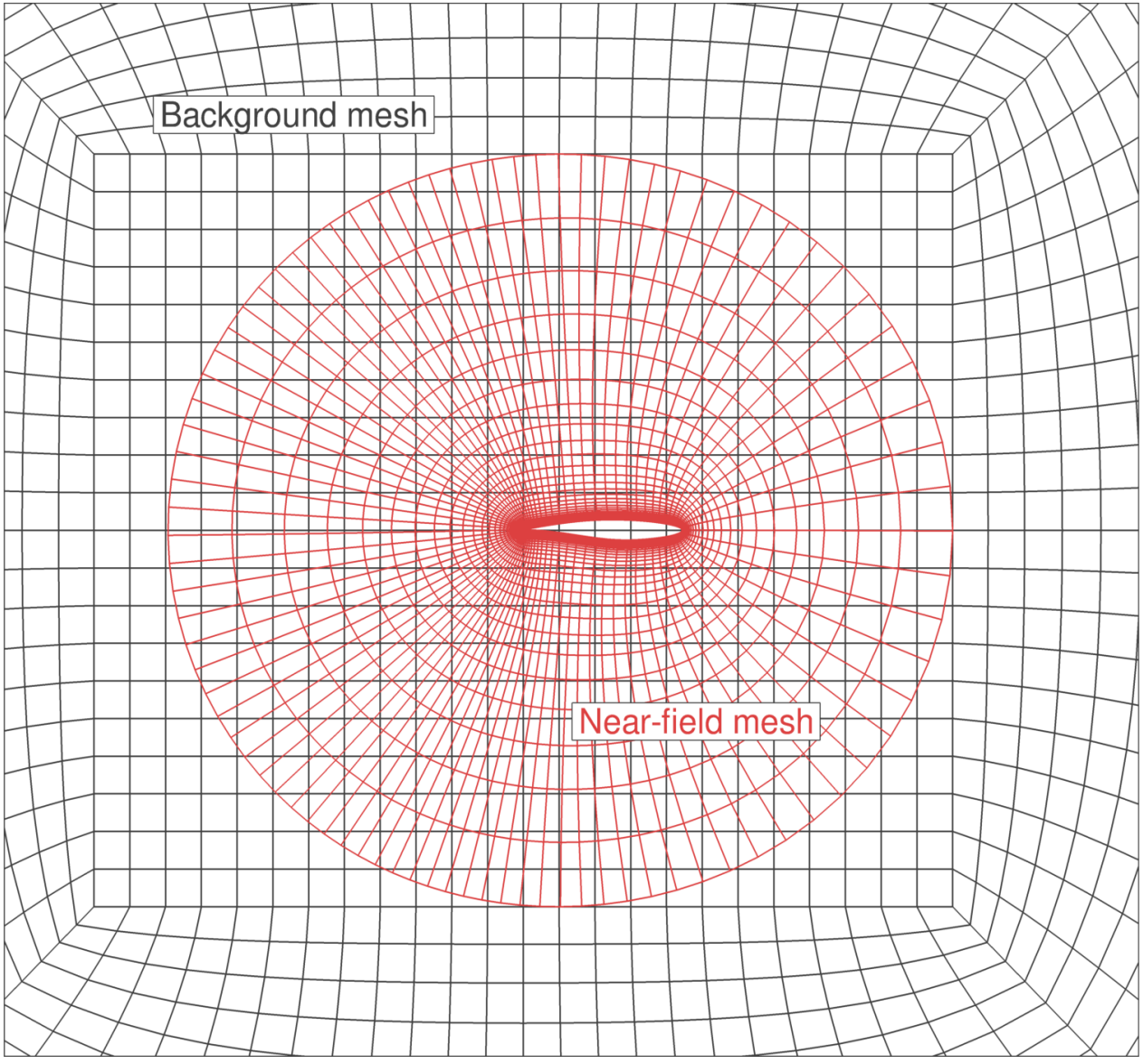


Main principle

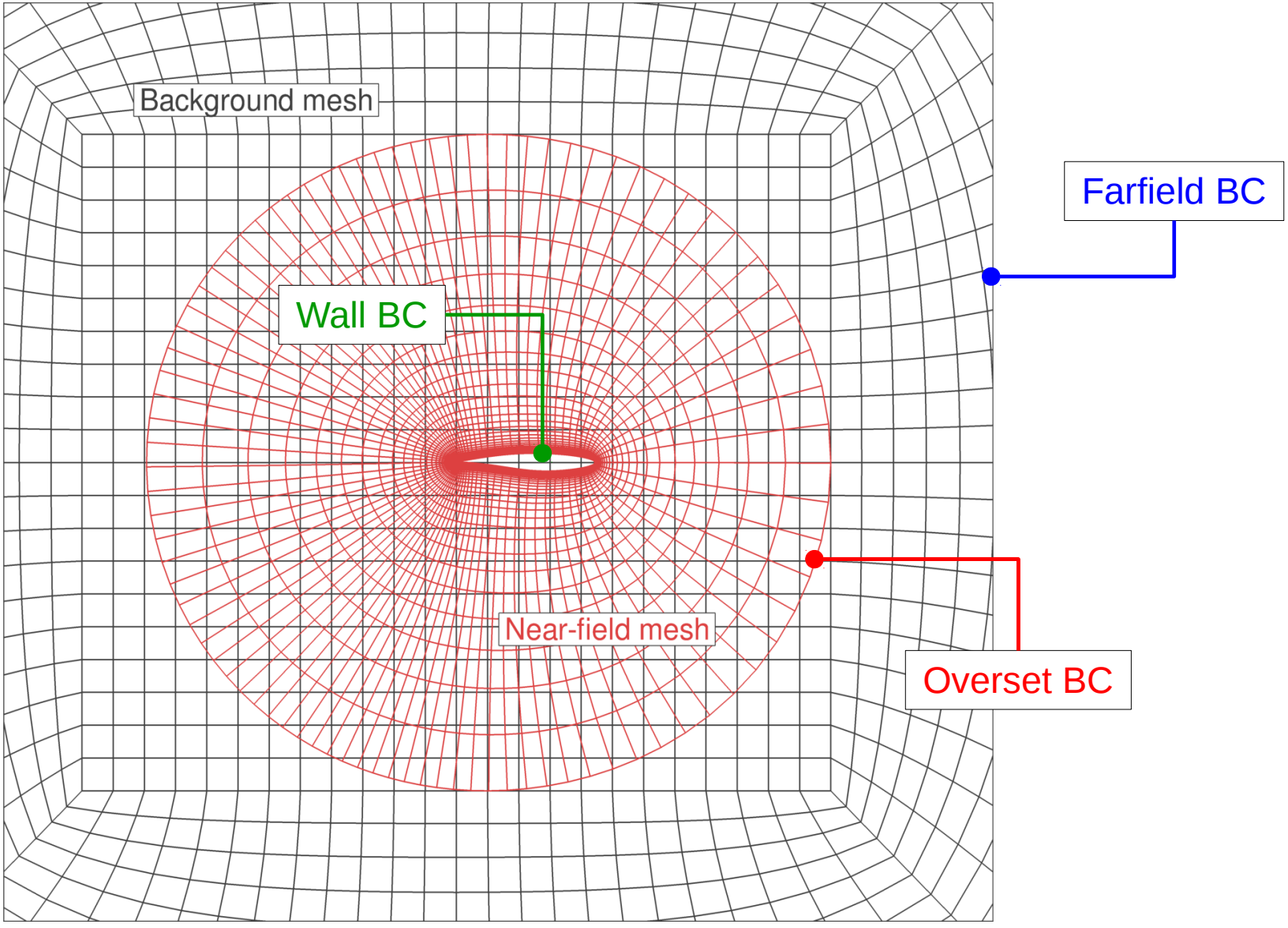
Information is interpolated among overlapping meshes on every solver iteration.



How to set up boundary conditions?



Boundary conditions for overset meshes

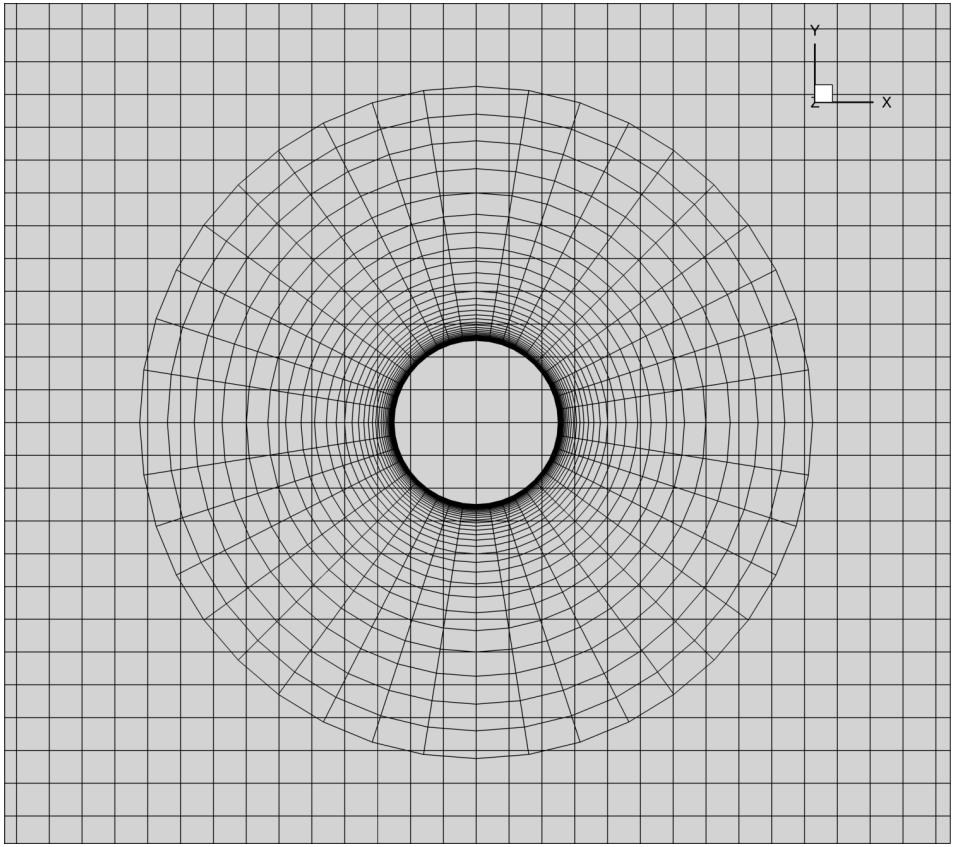


Cells assume different tasks in an overset mesh

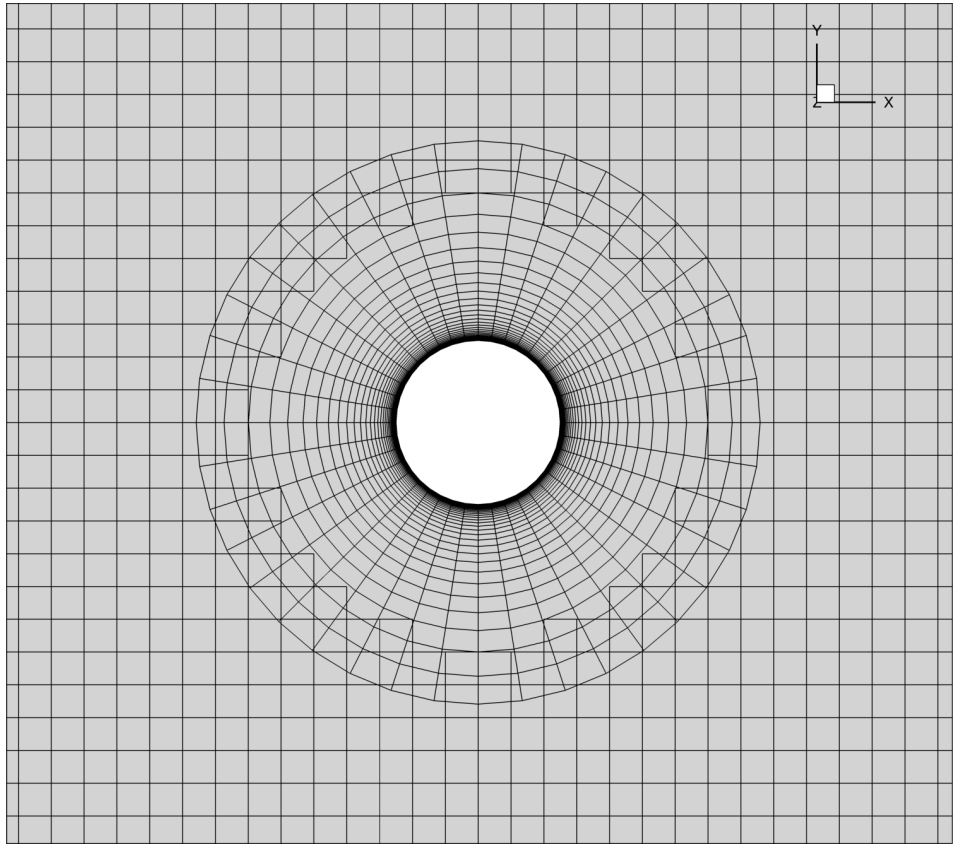
- **Compute Cells:** Active cells that are relevant to the solution as they represent the volume. The PDEs are enforced on these cells.
- **Blanked Cells:** Non-representative cells as they may be inside bodies or overlapped by better quality cells.
- **Interpolated Cells (Receivers):** Cells that will inherit state variables from donor cells that belong to other overlapping meshes on every iteration. They are responsible by the *Overset* boundary condition.

We use Implicit Hole Cutting (IHC) to classify cells

- IHC is an automatic way to assign overset connectivities.
- It is based on the assumption that cells are finer near walls.
- Preserves smaller cells and blanks/interpolate bigger ones.



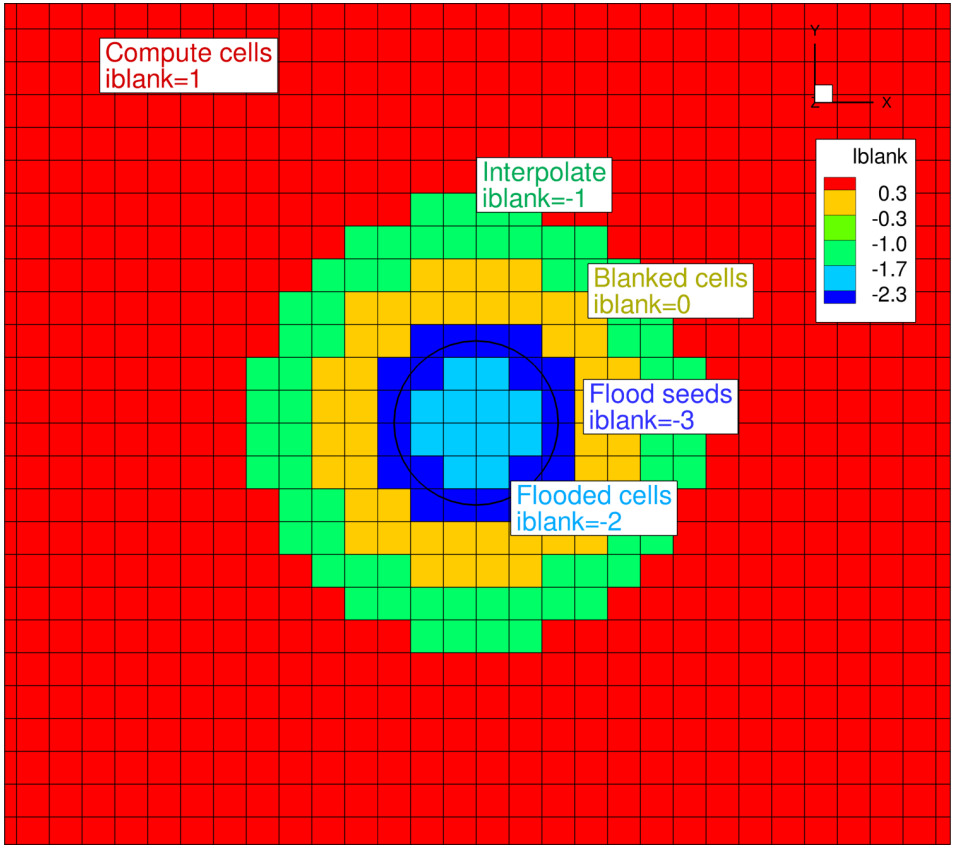
Original mesh



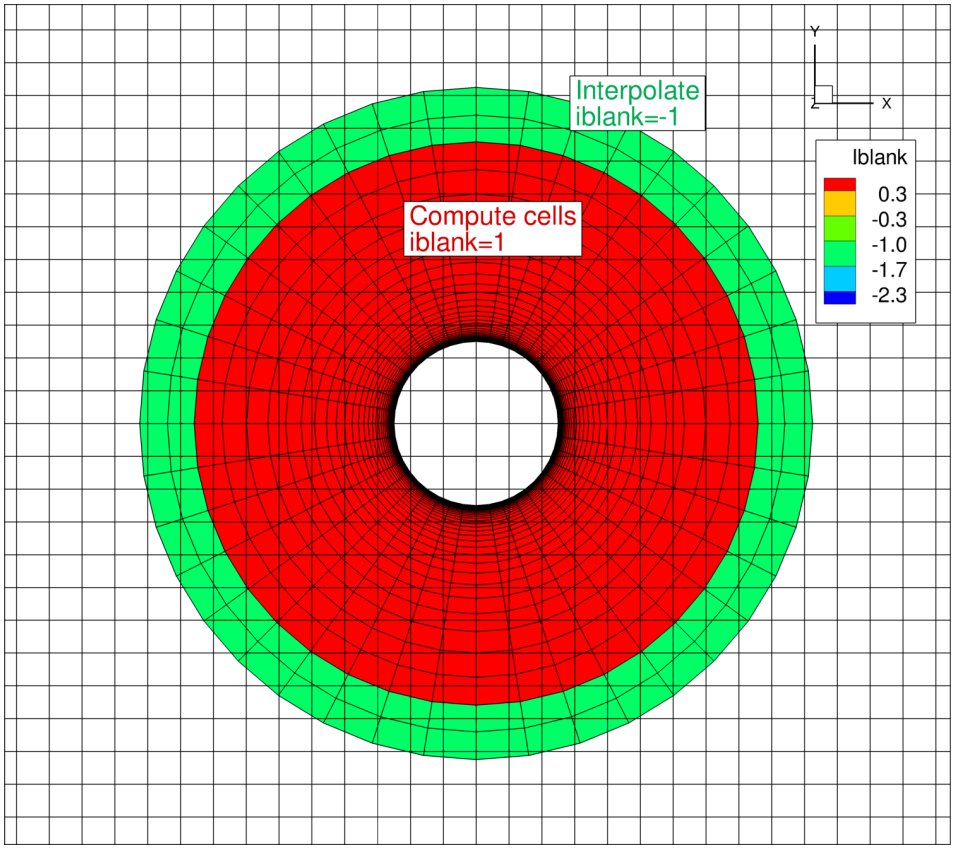
Compute cells

The iblank array indicates the function of each cell

ADflow saves this array in the volume CGNS files.



Background Mesh

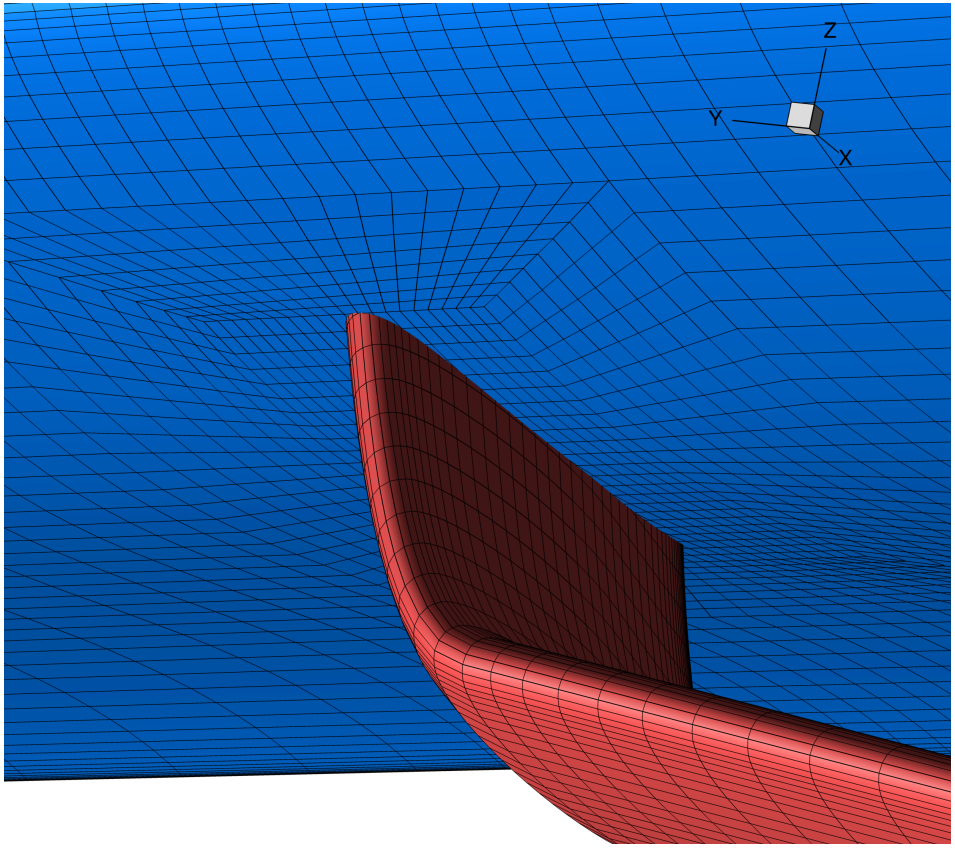


Near-field Mesh

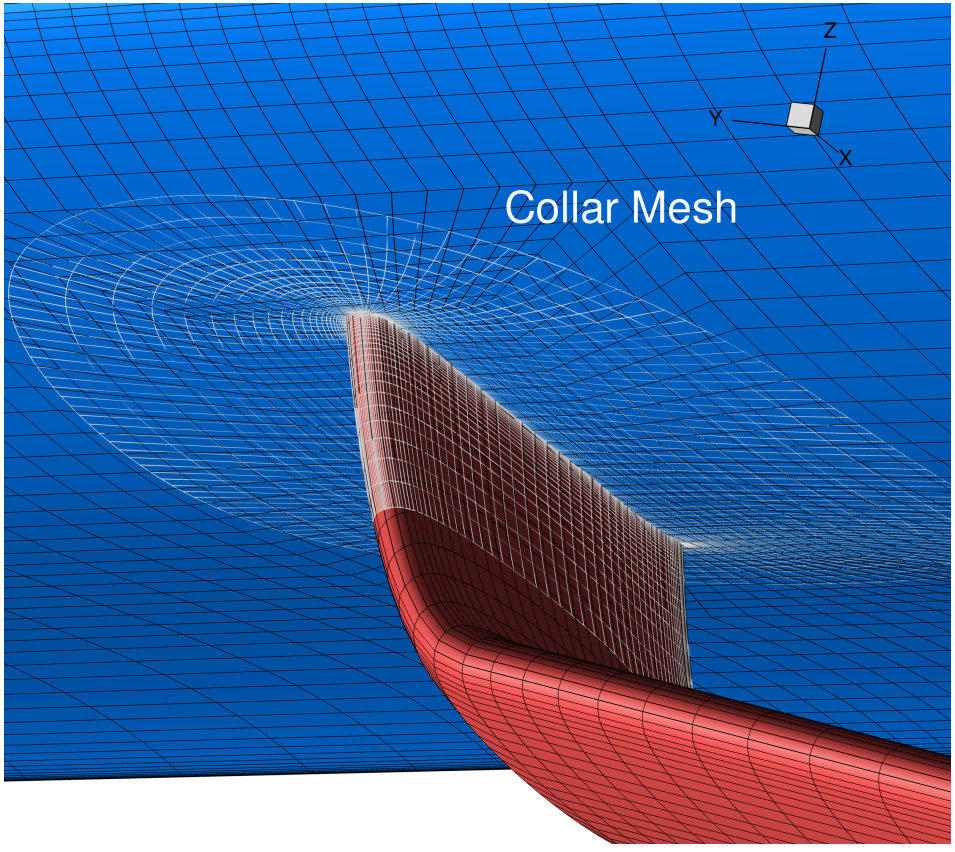
Steps to create an Overset Mesh

- Generate surface meshes for the main components in *ICEM*.
- Extrude surface meshes into volume meshes using *pyHyp*.
- Generate background mesh using *cgns_utils*.
- Merge blocks in a single file using *cgns_utils*.
- Check connectivities using *ADflow*.
- Run your case with *ADflow*

We need collar meshes to represent intersections

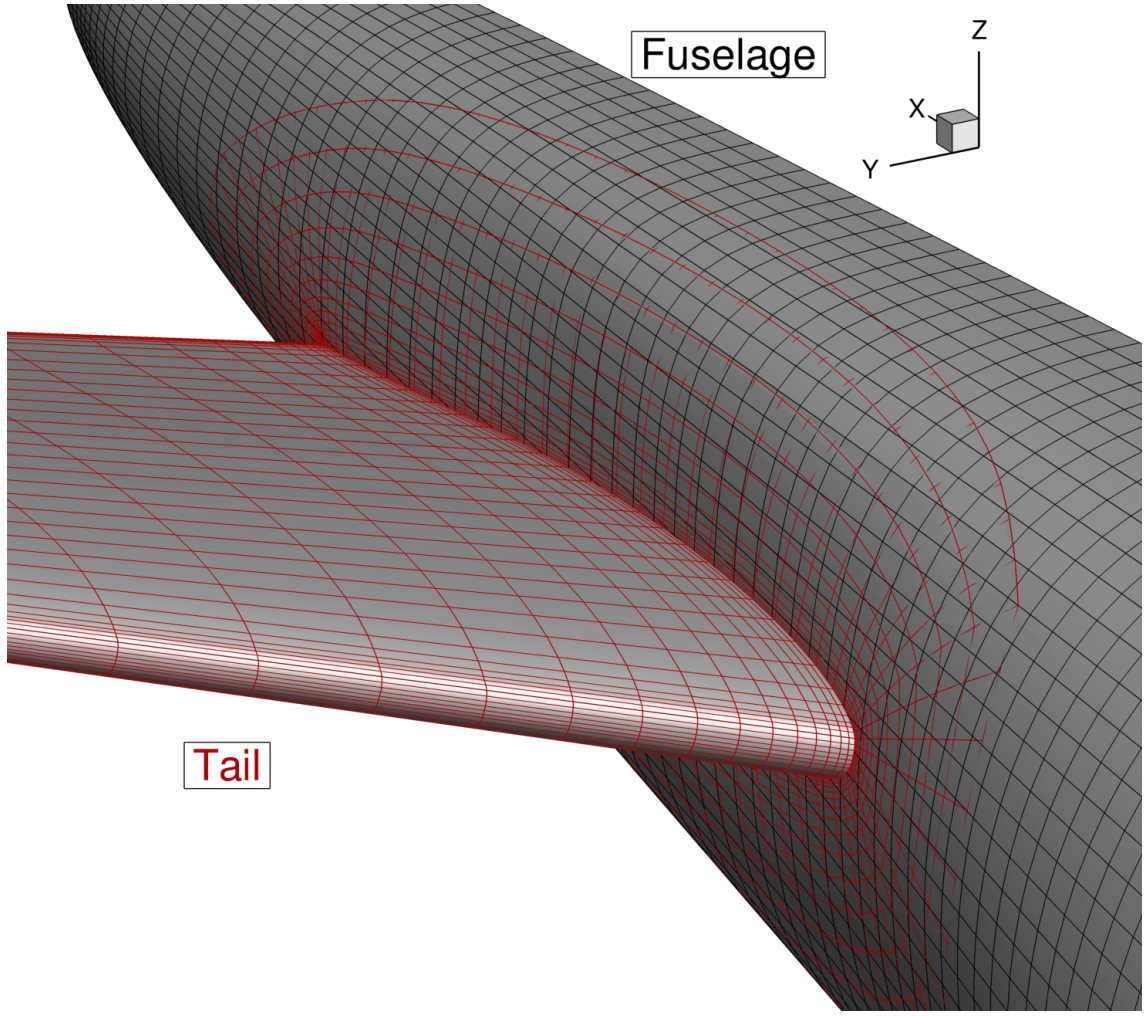


Meshes have no intersection information



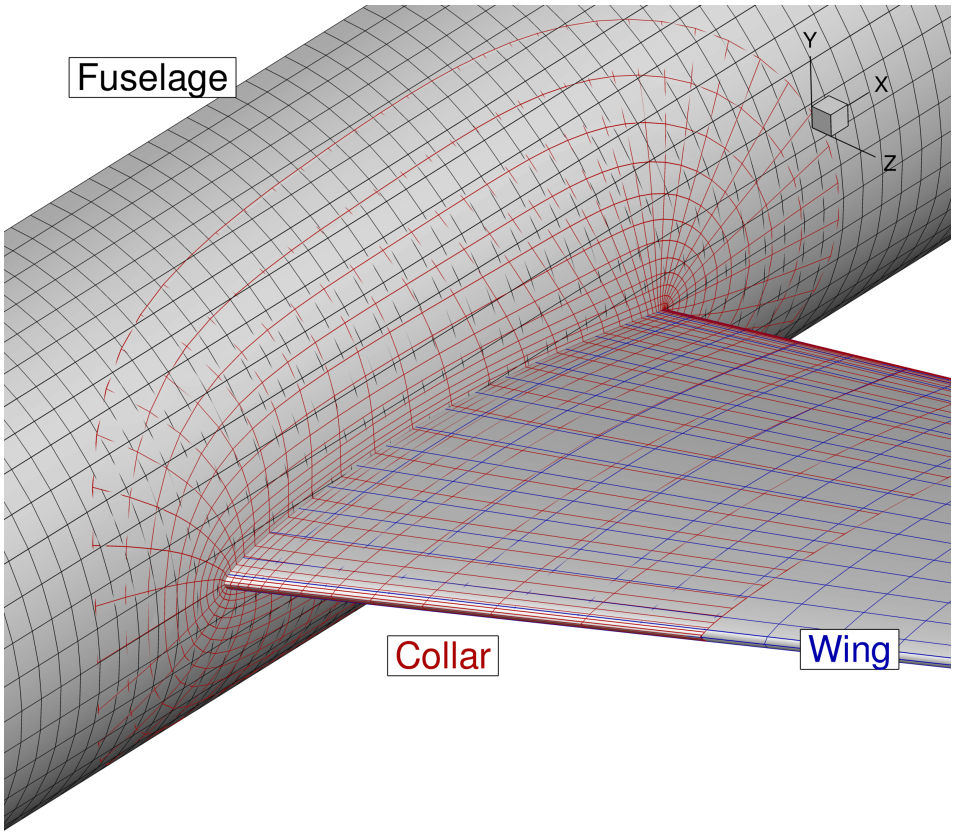
The collar mesh outlines the intersection

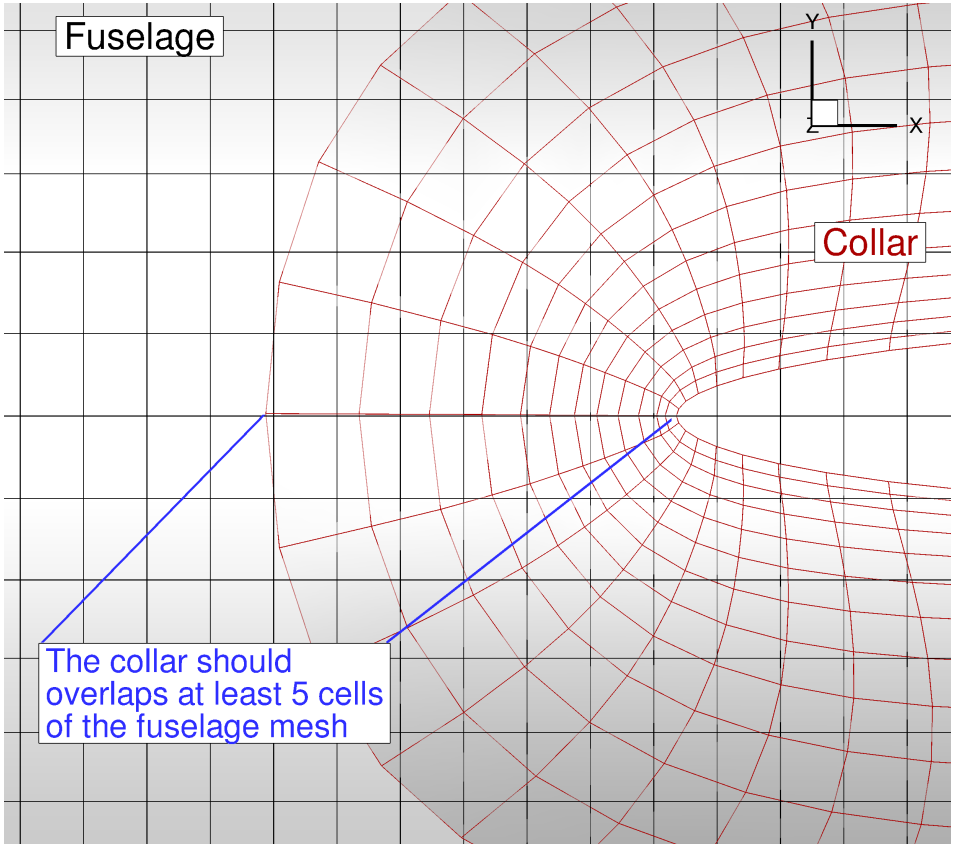
We can also use a half-collar to reduce the number of overset blocks



- The half-collar on the fuselage belongs to the tail mesh
- The half-collar and the rest of the tail mesh share the intersection line

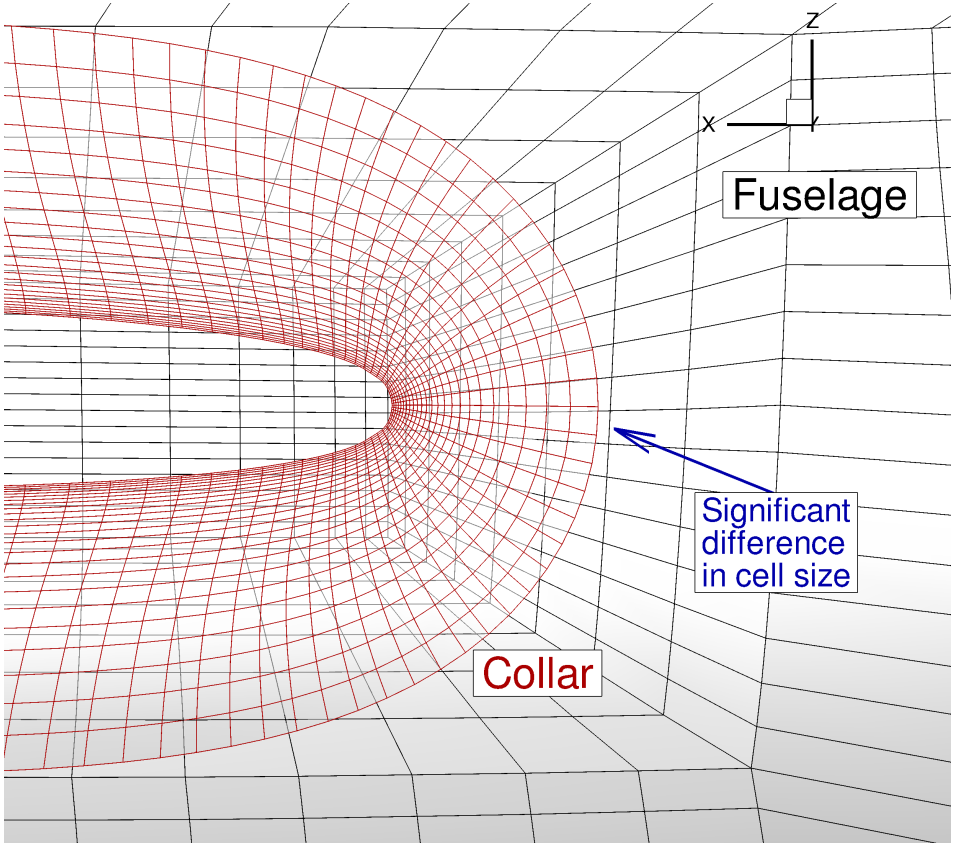
Tip #1: Check the surface overlap



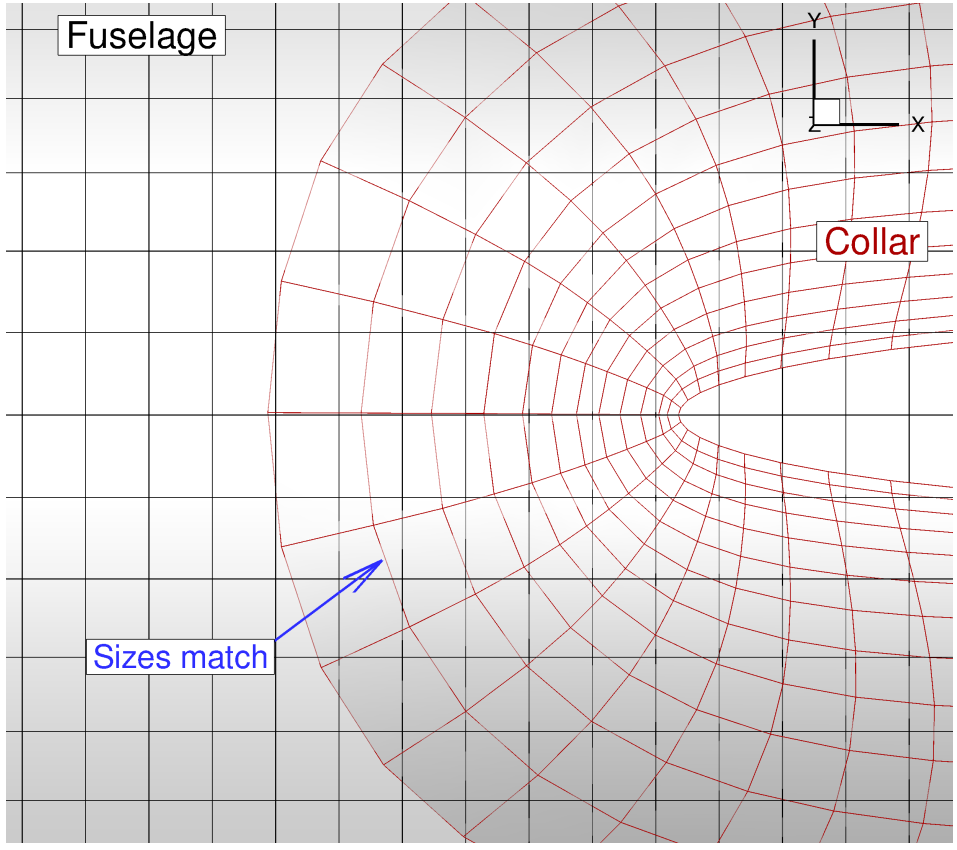


Side view of the intersection leading edge

Tip #2: Match cells sizes of the overlapped meshes, especially near boundaries



Not recommended. May give a valid hole cutting with additional effort.



Better transition.
Easier to find interpolation stencils.

Tip #3: Match the growth ratios of the mesh extrusion

- Use similar values of initial cell height for all meshes (‘s0’ option in pyHyp).
- Make sure that all meshes have similar growth ratios during the pyHyp extrusion. a variation of $\pm$ 0.05 is okay.
- Collar meshes may have smaller values for ‘s0’ and growth ratio to increase their priority during the IHC.

```
#--------------------#
 Total Nodes:    4200
 Unique Nodes:   3774
 Total Faces:    3734
#--------------------#
 Normal orientation check ...
 Normals are consistent!
#--------------------#
Grid Ratio:  1.3711
#--------------------#
```

Grid Lvl	CPU Time	Sub Its	KSP Its	nAvg	Sl	Sensor Max	Sensor Min	Min Quality	Min Volume	deltaS	March Distance	cMax	Ratio kMax
2	0.1	2	21	2	0.031	1.00217	0.95650	0.19145	0.696E-06	0.223E-03	0.282E-03	0.0036	0.0000
3	0.2	2	21	2	0.041	1.00328	0.93413	0.14789	0.959E-06	0.326E-03	0.811E-03	0.0068	1.6905
4	0.2	1	21	2	0.044	1.00413	0.91809	0.12825	0.137E-05	0.446E-03	0.114E-02	0.0073	1.5036
5	0.3	2	21	2	0.052	1.00576	0.88356	0.10747	0.209E-05	0.612E-03	0.220E-02	0.0136	1.5395

```
=======================================
            pyHyp Summary
=======================================
```

ID	grid name	gridRatio	minQualityOverall	minVolumeOverall
00	fuse	1.3938277	0.03954137	5.2459e-04
01	wing_collar_combined	1.3958498	0.12955067	9.9467e-08
02	tail	1.3711248	0.07774588	6.9581e-07

The terminal output indicates if you have a valid hole cutting

```
Flood Iteration:           1 Blanked        28215 Interior Cells.
Flood Iteration:           2 Blanked         6817 Interior Cells.
Flood Iteration:           3 Blanked            0 Interior Cells.
+-------------------------------+
| Compute Cells           :      706816
| Fringe Cells            :       81232
| Blanked Cells           :         457
| Explicitly Blanked Cells:           0
| Flooded   Cells         :       33772
| FloodSeed Cells         :        2939
+-------------------------------+
Total number of orphans:         352
Flood Iteration:           1 Blanked        30503 Interior Cells.
Flood Iteration:           2 Blanked         6866 Interior Cells.
Flood Iteration:           3 Blanked            0 Interior Cells.
+-------------------------------+
| Compute Cells           :      706821
| Fringe Cells            :       81026
| Blanked Cells           :           0
| Explicitly Blanked Cells:           0
| Flooded   Cells         :       34430
| FloodSeed Cells         :        2939
+-------------------------------+
Total number of orphans:           0
Flood Iteration:           1 Blanked        30503 Interior Cells.
Flood Iteration:           2 Blanked         6866 Interior Cells.
Flood Iteration:           3 Blanked            0 Interior Cells.
+-------------------------------+
| Compute Cells           :      706821
| Fringe Cells            :       81026
| Blanked Cells           :           0
| Explicitly Blanked Cells:           0
| Flooded   Cells         :       34430
| FloodSeed Cells         :        2939
+-------------------------------+
Total number of orphans:           0
+-------------------------------+
| Compute Cells           :      706821
| Fringe Cells            :       66555
| Blanked Cells           :       14471
| Explicitly Blanked Cells:           0
| Flooded   Cells         :       34430
| FloodSeed Cells         :        2939
+-------------------------------+
Total number of orphans:           0
```

Valid IHC

```
Flood Iteration:           1 Blanked       170238 Interior Cells.
Flood Iteration:           2 Blanked        68444 Interior Cells.
Flood Iteration:           3 Blanked        33537 Interior Cells.
Flood Iteration:           4 Blanked       143799 Interior Cells.
Flood Iteration:           5 Blanked       353244 Interior Cells.
Flood Iteration:           6 Blanked       116567 Interior Cells.
Flood Iteration:           7 Blanked        35386 Interior Cells.
Flood Iteration:           8 Blanked            0 Interior Cells.
+-------------------------------+
| Compute Cells           :     4578616
| Fringe Cells            :      921244
| Blanked Cells           :        6506
| Explicitly Blanked Cells:           0
| Flooded   Cells         :      913416
| FloodSeed Cells         :       23066
+-------------------------------+
Total number of orphans:          9106
Flood Iteration:           1 Blanked       811589 Interior Cells.
Flood Iteration:           2 Blanked      1046286 Interior Cells.
Flood Iteration:           3 Blanked      1009991 Interior Cells.
Flood Iteration:           4 Blanked       790672 Interior Cells.
Flood Iteration:           5 Blanked       686297 Interior Cells.
Flood Iteration:           6 Blanked       330735 Interior Cells.
Flood Iteration:           7 Blanked        35390 Interior Cells.
Flood Iteration:           8 Blanked            0 Interior Cells.
+-------------------------------+
| Compute Cells           :      875656
| Fringe Cells            :      827472
| Blanked Cells           :       28760
| Explicitly Blanked Cells:           0
| Flooded   Cells         :     4687894
| FloodSeed Cells         :       23066
+-------------------------------+
Total number of orphans:         80369
```

Bad IHC

- Several flooding iterations
- Small number of compute cells
- Orphan cells are present

We need to check leaks in the flooding process

- Set ADflow option: `'nrefine':1`
  This avoids subsequent flood iterations. You will get error warnings but this is fine.

- Set ADflow option: `'usezippermesh':False`
  This skips the zipper mesh generation, which may crash if the hole cutting does not work.

- Run the overset check file: `ihc_check.py`

- Open the `fc_-001_vol.cgns` file in Tecplot

- Use the blanking option to hide cells with *iblank* $\geq$ -1. This will show just the flood seeds (*iblank*=-3) and flooded cells (*iblank*=-2).

- Check which CGNS blocks are fully flooded and then find the cells that are connecting the side of the mesh that is inside the body to the exterior side of the mesh.

Flood troubleshooting

- Flooding is usually caused by cells that grows too fast off a wall.
  The mesh with a high growth ratio may cause the flooding of the other overlapped meshes, since the other meshes will not create a layer of interpolate cells to contain the flood. Check if meshes have similar growth ratios for the pyHyp extrusion.

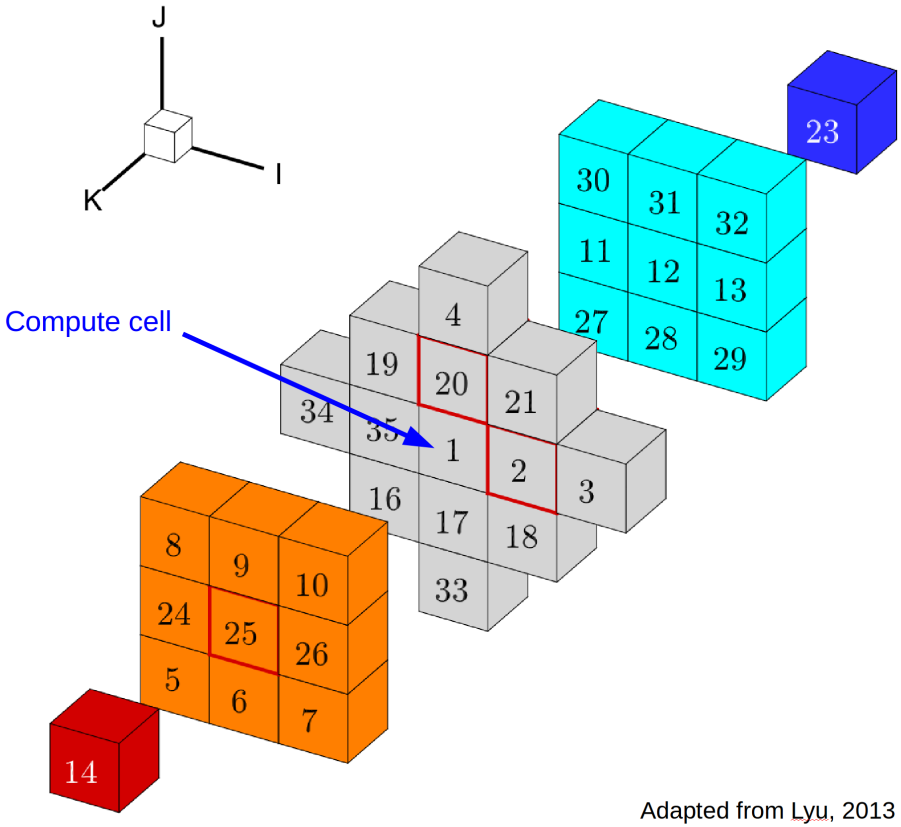
- Change the `'nearwalldist'` option in ADflow.
  This option controls how compute cells are preserved near walls. Changing this value may prevent flooding. We usually use 0.01 for a full-scale aircraft mesh defined in metric units. If a collar mesh is flooding, try increasing `'nearwalldist'` to reduce the number of flood seeds.

- Check for sufficient overlap on the surface and in the volume.
  The overlap should have at least 5 cells from each mesh.

- The background mesh may be too coarse.
  Either extend the near-field meshes or refine the background mesh until you have a 5 cell overlap along the off-wall direction.

Requirement for a valid hole cutting

The stencil of all compute cells should include only other compute or interpolated cells.



Adapted from Lyu, 2013

If this is not satisfied, we have an orphan cell

Orphans troubleshooting

- ADflow outputs the CGNS block id, and the i,j,k position of the orphan cells. The k values (4^{th} column) may point to the issue.

```
Error in connectivity at :        50          3          11          3
Error in connectivity at :        50          4          11          3
Error in connectivity at :        50          5          11          3
Error in connectivity at :        50          6          11          3
Error in connectivity at :        50          7          11          3
```

- Orphans with high k: Lack of volume overlap.

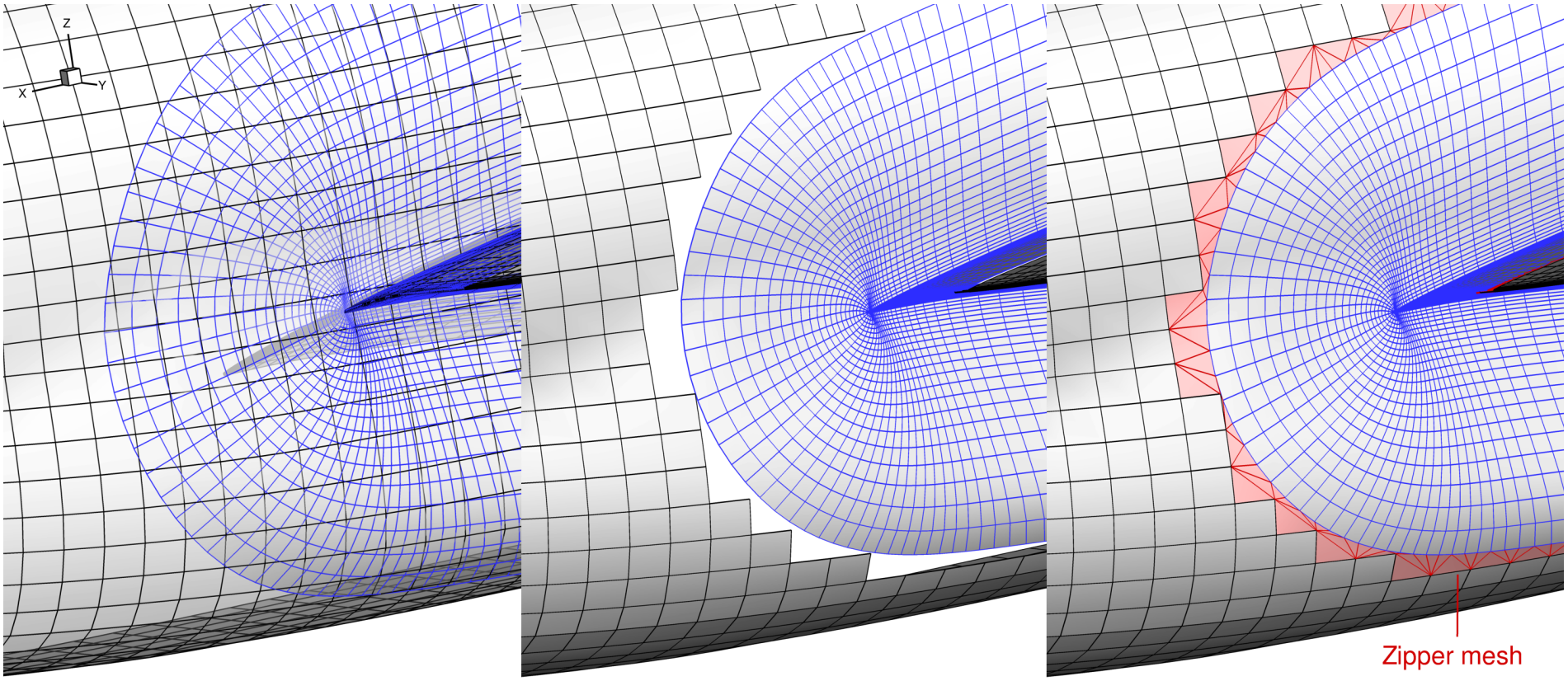
  Some interpolate cells cannot find donors. So they become blanked cells within the stencil of a compute cell. Possible solutions are increasing the mesh extrusion distance (‘`marchDist`’ option in pyHyp) or adding more layers to the mesh extrusion process (‘`N`’ option in pyHyp). You may also refine the background mesh.

- Orphans with small k: Reduce ‘`nearwalldist`’ option in ADflow.

  You have compute cells beneath the surface defined by overlapping meshes.

  The smaller ‘`nearwalldist`’ may flood these unnecessary cells.

Zipper meshes provide watertight surface for force integration



Overlapped meshes

Removed overlaps

Triangulated gaps

References

- Implicit hole cutting:
  Lee, Y. and Baeder, J. D., *Implicit Hole Cutting A New Approach to Overset Grid Connectivity*, 16th AIAA Computational Fluid Dynamics Conference, Washington, DC, June 2003.

- Zipper meshes:
  Chan, W. M., *Enhancements to the Hybrid Mesh Approach to Surface Loads Integration on Overset Structured Grids*, 19th AIAA Computational Fluid Dynamics Conference, San Antonio, Texas, June 2009, AIAA Paper 2009-3990.

- Overset implementation in ADflow:
  Kenway, G. K. W., Secco, N., Martins, J. R. R. A., Mishra, A., and Duraisamy, K., *An Efficient Parallel Overset Method for Aerodynamic Shape Optimization*, Proceedings of the 58th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, AIAA SciTech Forum, Grapevine, TX, 2017. doi:10.2514/6.2017-0357.