# FAQ

[Where can I find all supported parameters and their default values for the daOption dictionary in runScript.py?](https://dafoam.github.io/mydoc_get_started_faq.html#where-can-i-find-all-supported-parameters-and-their-default-values-for-the-daoption-dictionary-in-runscriptpy)

[How to use a finer mesh?](https://dafoam.github.io/mydoc_get_started_faq.html#how-to-use-a-finer-mesh)

[How to use more FFD points?](https://dafoam.github.io/mydoc_get_started_faq.html#how-to-use-more-ffd-points)

[How to visualize the FFD points?](https://dafoam.github.io/mydoc_get_started_faq.html#how-to-visualize-the-ffd-points)

[How to use more CPU cores?](https://dafoam.github.io/mydoc_get_started_faq.html#how-to-use-more-cpu-cores)

[How to optimize a different airfoil?](https://dafoam.github.io/mydoc_get_started_faq.html#how-to-optimize-a-different-airfoil)

[How to change the flow conditions?](https://dafoam.github.io/mydoc_get_started_faq.html#how-to-change-the-flow-conditions)

[How to use different turbulence models?](https://dafoam.github.io/mydoc_get_started_faq.html#how-to-use-different-turbulence-models)

[How many CPU cores to use and how much memory does it need?](https://dafoam.github.io/mydoc_get_started_faq.html#how-many-cpu-cores-to-use-and-how-much-memory-does-it-need)

[How to extract the optimized geometry?](https://dafoam.github.io/mydoc_get_started_faq.html#how-to-extract-the-optimized-geometry)

[Can I get the optimized geometry in a CAD format?](https://dafoam.github.io/mydoc_get_started_faq.html#can-i-get-the-optimized-geometry-in-a-cad-format)

[How to get sensitivity maps?](https://dafoam.github.io/mydoc_get_started_faq.html#how-to-get-sensitivity-maps)

[How to reduce the size of parallel optimization results?](https://dafoam.github.io/mydoc_get_started_faq.html#how-to-reduce-the-size-of-parallel-optimization-results)

[Does DAFoam support optimization for pure 2D problems?](https://dafoam.github.io/mydoc_get_started_faq.html#does-dafoam-support-optimization-for-pure-2d-problems)

[Can I use my own mesh?](https://dafoam.github.io/mydoc_get_started_faq.html#can-i-use-my-own-mesh)

[Does DAFoam use the exact same primal solvers in OpenFOAM?](https://dafoam.github.io/mydoc_get_started_faq.html#does-dafoam-use-the-exact-same-primal-solvers-in-openfoam)

[Does DAFoam support all the OpenFOAM’s configurations and boundary conditions?](https://dafoam.github.io/mydoc_get_started_faq.html#does-dafoam-support-all-the-openfoams-configurations-and-boundary-conditions)

[How to fix the “Primal solution failed for the baseline design!” error?](https://dafoam.github.io/mydoc_get_started_faq.html#how-to-fix-the-primal-solution-failed-for-the-baseline-design-error)

[How to fix “-> Warning: xx point(s) not projected to tolerance: 1e-12. Max Error: xxx ; RMS Error: xxx. List of Points is: (pt, delta):”?](https://dafoam.github.io/mydoc_get_started_faq.html#how-to-fix---warning-xx-points-not-projected-to-tolerance-1e-12-max-error--xxx--rms-error-xxx-list-of-points-is-pt-delta)

[How to fix the “Conflicting Colors Found!” error?](https://dafoam.github.io/mydoc_get_started_faq.html#how-to-fix-the-conflicting-colors-found-error)

[How to fix “Warning: xxx the ray might not have been long enough to intersect the nearest curve.”?](https://dafoam.github.io/mydoc_get_started_faq.html#how-to-fix-warning-xxx-the-ray-might-not-have-been-long-enough-to-intersect-the-nearest-curve)

[Why the adjoint equation is not converging?](https://dafoam.github.io/mydoc_get_started_faq.html#why-the-adjoint-equation-is-not-converging)

[Why do I keep getting failed mesh quality checks?](https://dafoam.github.io/mydoc_get_started_faq.html#why-do-i-keep-getting-failed-mesh-quality-checks)

[How to know the detailed description for a function’s input parameters defined in runScript.py?](https://dafoam.github.io/mydoc_get_started_faq.html#how-to-know-the-detailed-description-for-a-functions-input-parameters-defined-in-runscriptpy)

[What are the commands to start the DAFoam Docker image?](https://dafoam.github.io/mydoc_get_started_faq.html#what-are-the-commands-to-start-the-dafoam-docker-image)

[How to use the Docker image/container for code development?](https://dafoam.github.io/mydoc_get_started_faq.html#how-to-use-the-docker-imagecontainer-for-code-development)

[Why I keep getting a “Permission Denied” error when using the DAFoam Docker image?](https://dafoam.github.io/mydoc_get_started_faq.html#why-i-keep-getting-a-permission-denied-error-when-using-the-dafoam-docker-image)

[How to fix: “Error: There was an error projecting a node at xxx”?](https://dafoam.github.io/mydoc_get_started_faq.html#how-to-fix-error-there-was-an-error-projecting-a-node-at-xxx)

[How to fix: “ImportError: dynamic module does not define module export function”?](https://dafoam.github.io/mydoc_get_started_faq.html#how-to-fix-importerror-dynamic-module-does-not-define-module-export-function)

[How to fix the “No module named dafoam.pyDASolver” error?](https://dafoam.github.io/mydoc_get_started_faq.html#how-to-fix-the-no-module-named-dafoampydasolver-error)

[How to speed up DAFoam repo compilation speed for code development?](https://dafoam.github.io/mydoc_get_started_faq.html#how-to-speed-up-dafoam-repo-compilation-speed-for-code-development)

[How to run a multipoint optimization?](https://dafoam.github.io/mydoc_get_started_faq.html#how-to-run-a-multipoint-optimization)

[How to run an optimization for 3D wings?](https://dafoam.github.io/mydoc_get_started_faq.html#how-to-run-an-optimization-for-3d-wings)

[How to create a new objective function?](https://dafoam.github.io/mydoc_get_started_faq.html#how-to-create-a-new-objective-function)

[How can I contribute to this website?](https://dafoam.github.io/mydoc_get_started_faq.html#how-can-i-contribute-to-this-website)

[How to cite DAFoam?](https://dafoam.github.io/mydoc_get_started_faq.html#how-to-cite-dafoam)

The following are some frequently asked questions (FAQ) for DAFoam. If you have more questions, please post them to [DAFoam Github Discussions](https://github.com/mdolab/dafoam/discussions).

## Where can I find all supported parameters and their default values for the daOption dictionary in runScript.py?

The documentation for all the supported parameters in daOption is available at [here](https://dafoam.github.io/doxygen/html/classdafoam_1_1pyDAFoam_1_1DAOPTION.html). Note that, we typically set essential parameters for daOption, and use the default values for other parameters. Their defaults values can be found in dafoam/pyDAFoam.py. In addition, all the values for daOption will be printed to the screen when the optimization runs.

## How to use a finer mesh?

To increase the mesh density, one needs to modify the parameters in “genAirFoilMesh.py”. For example, changing “dX1PS = 0.005” to “dX1PS = 0.002” will increase the mesh density at the leading edge, changing “dXMaxPS = 0.02” to “dXMaxPS = 0.01” will increase the mesh density for the lower surface of the airfoil, changing “NpTE = 5” to “NpTE = 11” will use 11 mesh points for the blunt trailing edge. To change the mesh density in the marching direction, modify “NpExtrude” (number of layers to extrude), “yWall” (wall distance of the first layer mesh), “marchDist” (marching distance). Refer to [pyHyp](https://github.com/mdolab/pyhyp) for more details of the genAirFoilMesh.py script.

## How to use more FFD points?

To increase the number of FFD points, one needs to increase “nx” (number of FFD points in the x direction) in “FFD/genFFD.py”. Then run “python genFFD.py” in the FFD folder to generate a new “wingFFD.xyz” file. Note that the plot3D file we generate is a 3D mesh, and any internal points can be moved. Therefore, it is not necessary to use more than two points in the vertical (y) direction. Similarly, because it is a 2D case, there is no need to use more than two points in the z direction either. Also note that the “genFFD.py” script supports only uniform FFD points. We recommend using ICEM-CFD to generate more complex FFD points.

## How to visualize the FFD points?

You can open a FFD file (\*.xyz; plot3D format) in Paraview and choose “PLOT3D Reader” in the pop-up window. Then, on the left panel, uncheck “Binary File”, check “Multi Grid”, and then hit “Apply”. NOTE: Paraview sometime crashes when loading Plot3D files with a small number of points (it is a bug in Paraview). To avoid this, you can convert a Plot3D file to the Tecplot format. First load the DAFoam environment, and run dafoam\_plot3d2tecplot.py yourFFDFileName.xyz newFFDFileName.dat. Once done, a new file newFFDFileName.dat will be generated in the Tecplot format. You can then use Paraview to load this new file.

## How to use more CPU cores?

To run the optimization using 8 cores, first clean up previous results ./Allclean, then run:

./preProcessing.sh && mpirun -np 8 python runScript.py 2>&1 | tee optLog.txt

## How to optimize a different airfoil?

To run optimization for a different airfoil, one needs to create two new files in the “profiles” folder and put the new airfoil x-y coordinates in these files. The airfoil data should be separated into upper and lower surfaces, they should start from the leading edge and ends at trailing edge. We use blunt trailing edge, so one needs to truncate the lower and upper surface data at about 99.8% of the chord. In other words, the profile data shouldn’t end at x=1.0, delete a few points from the end.

Once the new airfoil data are ready, modify the file names to load for “airfoilProfilePS” and “airfoilProfileSS” in “genAirFoilMesh.py”.

In addition, one may need to change the parameters for “corners” in FFD/genFFD.py to make sure the FFD points fully contains the new airfoil. Once done, in the “FFD” folder, run “python genFFD.py” to generate a new FFD file “wingFFD.xyz”.

## How to change the flow conditions?

To run optimization at different flow conditions, one needs to modify the boundary condition values “U0”, “p0”, and “nuTilda0” from the “runScript.py”.

To run at a different lift coefficient, modify “CL\_target”, then run mpirun -np 4 python runScript.py --task=solveCL. Once the solveCL is done, note down the “alpha0” value that is printed to the screen, and replace the value in “runScript.py”.

This tutorial uses an incompressible flow solver DASimpleFoam, so the Mach number should be less than 0.1. For subsonic flow conditions (e.g., ~0.1 < M < ~0.6), refer to the settings in tutorials-main/NACA0012\_Airfoil/subsonic. For the transonic flow conditions, refer to tutorials-main/NACA0012\_Airfoil/transonic. Note that both runScript.py and the OpenFOAM configuration files (e.g., fvSchemes, fvSolution) are modified for these flow conditions.

## How to use different turbulence models?

To use the kOmegaSST or kEpsilon model, change the RASModel parameter to kOmegaSST or kEpsilon in constant/turbulenceProperties.

## How many CPU cores to use and how much memory does it need?

We recommend using one CPU core and reserve 2 GB memory per 10,000 cells. That being said, for a one million cell case, we recommend using 100 CPU cores and reserve 200 GB memory.

## How to extract the optimized geometry?

If you run the optimization in serial, load the OpenFOAM environment and run this:

surfaceMeshTriangulate -patches '(wing)' -latestTime optShape.stl

If you run the optimization in parallel using 4 cores, run this:

mpirun -np 4 surfaceMeshTriangulate -patches '(wing)' -latestTime -parallel optShape.stl

The above command will extract the patch “wing” to a stl file called “optShape.stl”. If you have multiple patches to extract, modify the “-patches” flag, e.g., -patches ‘(wing body)’. Also, the “-lastTime” flag extracts stl files for the last optimization step. If you don’t add the “-lastTime” flag, it will extract stl files for all optimization steps.

An alternative way to output the optimized geometry is to load it in ParaView, go to the last time step for the optimized shape, select the surfaces you want to output, choose Files-Save Data, and save it as STL files.

If you want to get the optimized geometry from the original STL file instead of the OpenFOAM mesh (e.g., for 3D printing), you can use this [script](https://github.com/DAFoam/tutorials/tree/main/Prowim_Wing_Propeller/deformSTL). It will read the optimized design variables from file and use pyGeo to deform a STL file to its optimal shape.

## Can I get the optimized geometry in a CAD format?

Yes, you can get the optimized geometry in the IGES format through pyGeo. Follow the “deformGeo” section from [here](https://dafoam.github.io/mydoc_tutorials_aero_prowim_wing.html). Refer to the [pyGeo documentation](https://mdolab-pygeo.readthedocs-hosted.com/en/latest/update_pygeo.html) for more general instructions on how to deform the design surface geometry.

## How to get sensitivity maps?

The latest version of DAFoam can output sensitivity maps during optimization. You need to set the names of the design variables to “writeSensMap” in runScript.py (check [this example](https://github.com/mdolab/dafoam/blob/c1c3ea12a49ceec7177238f7dc70a25ce260bba9/tests/runTests_DASimpleFoam.py#L46)). A more general description is [here](https://github.com/mdolab/dafoam/blob/c1c3ea12a49ceec7177238f7dc70a25ce260bba9/dafoam/pyDAFoam.py#L552). Now, we only support outputting sensitivity for FFD and Field designVarType.

## How to reduce the size of parallel optimization results?

When running in parallel, OpenFOAM will generate folders for each decomposed domain, e.g., processor0, processor1. This feature takes a lot of space and is slow to transfer and post-process. To fix this issue, wait until the parallel optimization is done, then go to the optimization folder, load the OpenFOAM environment, and run this command reconstructPar to combine all the decomposed flow fields that are stored in processor0, processor1, etc. You will see a bunch of new folders called 0.00000001, 0.00000002, etc. These are the combined flow solutions for each optimization step. So once the reconstructPar command is done, one can delete all the processor0, processor1, etc. folders. When visualizing the flow fields in Paraview, there is no need to choose “Decomposed Case” for “Case Type” because the cases have been reconstructed. This will increase the speed for visualization.

## Does DAFoam support optimization for pure 2D problems?

No, DAFoam does **NOT** support pure 2D optimization. In OpenFOAM, there is an option to do pure 2D simulation, which is setting a patch type to **empty** in constant/polyMesh/boundary. This feature is **NOT** supported in DAFoam. So, one need to change the **empty** patch type to **symmetry** instead, and use one cell in the symmetry direction to mimic a 2D simulation. Refer to the case setup in the [NACA0012 case](https://github.com/DAFoam/tutorials/tree/main/NACA0012_Airfoil/incompressible).

## Can I use my own mesh?

Yes, DAFoam can run optimization with meshes generated by other software. Just put the mesh in constant/polyMesh. Then, you need to make sure the OpenFOAM configuration files are modified properly for the new mesh, e.g., change the patch name/type for the boundary conditions, change the fvSchemes and fvSolution. We suggest you run the built-in OpenFOAM solvers, e.g., simpleFoam, and make sure they run, before running a DAFoam optimization. Also, since you have generated your own meshes, there is NO need to run ./preProcessing.sh.

## Does DAFoam use the exact same primal solvers in OpenFOAM?

Not exactly. DAFoam’s primal solvers are slightly different from the ones in OpenFOAM. 1. OpenFOAM uses volVectorField HbyA(constrainHbyA(rAU\*UEqn.H(), U, p)); in pEqn.H for many primal solvers; however, this constraint may cause inaccurate derivatives in DAFoam. Therefore, DAFoam uses volVectorField HbyA("HbyA", U); HbyA = rAU \* UEqn.H(); to compute HbyA. This was also the implementation used in OpenFOAM-2.4 and before. The difference between these two implementations is less than 0.01% in terms of the objective function values. 2. For the SA model, OpenFOAM uses “correctNut(fv1)”, while DAFoam uses “correctNut()” that recomputes the fv1 value based on the latest nuTilda. For steady state flow, these two implementations should produce the same result at convergence.

## Does DAFoam support all the OpenFOAM’s configurations and boundary conditions?

No. These OpenFOAM features are NOT supported in DAFoam. **Note**: the configurations and boundary conditions used in [DAFoam tutorials](https://github.com/dafoam/tutorials) are tested and working. Use caution if you want to add a new configuration that has not been used the tutorials.

* Unsteady solvers
* AMI boundary condition
* fvOptions and MRF are implemented for only some of primal solvers
* empty boundary condition
* Limited schemes such as Gauss linear limited corrected 0.33 for laplacianSchemes and limited corrected 0.33 for snGradSchemes in system/fvSchemes may cause inaccurate adjoint gradients. Don’t use them!

## How to fix the “Primal solution failed for the baseline design!” error?

This error basically says the first primal solution does not converge to the prescribed tolerance (primalMinResTol, default 1e-8), so the optimization aborts. There are two ways to fix it. 1. Increase the primal tolerance primalMinResTol, or 2. Increase how much difference (primalMinResTolDiff, default 1e2) between the prescribed and actual tolerances is considered as a failed primal solution. By default, if the actual tolerance does not drop to at least two order of magnitude higher than the prescribed one, the primal solution is considered to be a failed one.

## How to fix “-> Warning: xx point(s) not projected to tolerance: 1e-12. Max Error: xxx ; RMS Error: xxx. List of Points is: (pt, delta):”?

This error basically says your FFD box does not fully contain the design surface geometry. You need to adjust the FFD points.

## How to fix the “Conflicting Colors Found!” error?

This usually happens when you regenerate the mesh but forget to delete the old coloring files. In this case, DAFoam will read the old coloring files for the new mesh and report a coloring conflict. To fix this error, simply delete the existing coloring files that end with “.bin”, such as “dFdWColoring*.bin” and “dRdWColoring*.bin”, and then rerun DAFoam. If DAFoam can not find the coloring files in the optimization folder, it will regenerate them for the new mesh.

## How to fix “Warning: xxx the ray might not have been long enough to intersect the nearest curve.”?

This usually happens if you have a highly skewed FFD box. To fix this warning, add raySize=5 to the “nom\_addRefAxis” call. Refer to the interface from [here](https://github.com/mdolab/pygeo/blob/c417b0fea7d534458871aac8721a0c452a47eaae/pygeo/parameterization/DVGeo.py#L240)

## Why the adjoint equation is not converging?

If your adjoint equation is not converging well. Try to use the following setup for daOptions in runScript.py

"adjStateOrdering": "cell",

"adjEqnOption": {"gmresRelTol": 1.0e-6, "pcFillLevel": 1, "jacMatReOrdering": "natural", "gmresMaxIters": 2000, "gmresRestart": 2000},

If the above options do not help, change "pcFillLevel": 1 to "pcFillLevel": 2. NOTE: this will significantly increase the memory usage! If still not working, change div(phi,U) bounded Gauss linearUpwindV grad(U); to div(phi,U) bounded Gauss upwind; in system/fvSchemes and also check if the boundary conditions are properly assigned.

## Why do I keep getting failed mesh quality checks?

If you keep getting failed mesh checks throughout the optimization, first check if your baseline mesh quality passes. If yes, you may want to relax the mesh quality criteria, i.e., set checkMeshThreshold in daOption. Its default values are:

"checkMeshThreshold": {

"maxAspectRatio": 1000.0,

"maxNonOrth": 70.0,

"maxSkewness": 4.0,

"maxIncorrectlyOrientedFaces": 0,

},

You need to increase these default values, e.g., set "maxSkewness": 6.0,. If you want to ignore just a few incorrectly oriented faces, set maxIncorrectlyOrientedFaces to be greater than 0.

Another way to fix the issue is setting mesh quality constraints in optimization. Check the runScript\_meshQualityConstraint.py script from the [UBend](https://github.com/DAFoam/tutorials/blob/main/UBend_Channel/runScript_meshQualityConstraint.py) tutorial.

If you need to visualize which part of the mesh has poor quality, you can set "writeMinorIterations": True. DAFoam will write the mesh to the disk every time it tries to run the flow (even the checkMesh fails). You can then load the failed mesh in ParaView to visualize it.

## How to know the detailed description for a function’s input parameters defined in runScript.py?

In runScript.py, we call multiple functions from other modules, e.g., pyGeo, IDWarp, pyOptSparse. To learn the detailed description for a function’s input parameters, we take the function: DVGeo.addGeoDVLocal as an example. One can go to the dafoam/packages/miniconda3/lib/python3.6/site-packages directory and run this command grep -r addGeoDVLocal \*. Then, one can find this function is defined in pygeo/DVGeometry.py with detailed description of its input parameters.

## What are the commands to start the DAFoam Docker image?

For your convenient, here are the commands for [Linux](https://dafoam.github.io/mydoc_get_started_start_docker_linux.html), [MacOS](https://dafoam.github.io/mydoc_get_started_start_docker_mac.html), and [Windows](https://dafoam.github.io/mydoc_get_started_start_docker_windows.html).

## How to use the Docker image/container for code development?

The Docker image is mostly for regression tests and for first-time users. However, it is also useful for code development, especially for non-Linux system users. Here is what we can do: 1. Download the latest docker image docker pull dafoam/opt-packages:latest. 2. Go to the DAFoam folder that has your own development. 3. Create a Docker container using the command from [here](https://dafoam.github.io/mydoc_get_started_faq.html#what-are-the-commands-to-start-the-dafoam-docker-image). 4. Compile DAFoam in the Docker container and test your code.

NOTE: Because we use the --rm flag to start the container, everything you create in the container will be deleted after exiting. To avoid this, you can remove the --rm flag. So after exiting, the container will be still running in the background. You can re-login to the container by using the command: docker exec -it your\_container\_name bash. You can run docker ps -a to get the container name. Make sure you stop and remove the container (docker stop your\_container\_name && docker rm your\_container\_name) after everything is done to save memory.

## Why I keep getting a “Permission Denied” error when using the DAFoam Docker image?

If your OS system has more than one user, and you are not the first user and do not have admin access, you may get “Permission Denied” errors when using the DAFoam Docker image. The workaround is to run optimization in the Docker container. To this end, first, go to the optimization folder (e.g., NACA0012\_Airfoil/incompressible), and then start the DAFoam docker container (use the commands for [Linux](https://dafoam.github.io/mydoc_get_started_start_docker_linux.html), [MacOS](https://dafoam.github.io/mydoc_get_started_start_docker_mac.html), and [Windows](https://dafoam.github.io/mydoc_get_started_start_docker_windows.html)). Once in the docker container, switch to the root account by running sudo su and use dafoamuser as the password. Next, create a tmpRun folder in /home/dafoamuser and go there by running mkdir /home/dafoamuser/tmpRun && cd /home/dafoamuser/tmpRun. Now, copy all the optimization files from the mounted location to tmpRun and set the proper permission by running cp -r /home/dafoamuser/mount/\* /home/dafoamuser/tmpRun/ && chmod -R 777 /home/dafoamuser/tmpRun. Next, exit the root account exit. Finally, we can go to tmpRun and run the optimization there cd /home/dafoamuser/tmpRun && echo y | ./Allclean.sh && ./preProcessing.sh && mpirun -np 4 python runScript.py. Once the job is done, switch to the root account sudo su and copy the tmpRun folder to the mounted location cp -r /home/dafoamuser/tmpRun /home/dafoamuser/mount/. NOTE: not doing this step will lose all the optimization results because the tmpRun folder will be deleted when exiting the docker container. After this, you can exit the docker container, and you should be able to see the tmpRun folder in your optimization folder (NACA0012\_Airfoil/incompressible).

## How to fix: “Error: There was an error projecting a node at xxx”?

This error is likely caused by the incorrect leList and teList for the thickness and volume constraints. So double check the leList and teList in runScript.py and make sure they are completely within the wing geometry. Refer to [here](https://dafoam.github.io/mydoc_get_started_runscript.html#runscriptpy) for more details on how to setup leList and teList.

## How to fix: “ImportError: dynamic module does not define module export function”?

This error is likely caused by not running Allclean before running Allmake for DAFoam, especially between compiling the original, revere-AD, and forward-AD versions. So the solution is to recompile DAFoam and make sure you run Allclean before running Allmake.

## How to fix the “No module named dafoam.pyDASolver” error?

This error is likely caused by using different Python versions to compile DAFoam and run DAFoam cases. **Make sure you DO NOT close the terminal before all the installation steps are done!** For example, one may forget to load the DAFoam environment and use the system built-in Python 2.7 to run a case, while DAFoam was compiled with Python 3.8. To fix this, first check if you can find some compiled DAFoam libraries in the dafoam/dafoam folder, e.g., pyDASolverIncompressible.cpython-38-x86\_64-linux-gnu.so. The Python version is in the library name (cpython-38). So make sure your current Python environment (type python -V to check) matches the version in the library name. They do not match, we suggest you recompile DAFoam following the **exact** steps from [here](https://dafoam.github.io/mydoc_installation_source.html#dafoam)

## How to speed up DAFoam repo compilation speed for code development?

Compiling the DAFoam repo may make take up 30 minutes, depending on your PC performance. If you are modifying the DAFoam source code and need to quickly test your changes, you don’t need to compile all the DAFoam components. To speed up the DAFoam repo compilation, you can use the command Allmake incompressible to compile only the incompressible libraries and solvers. This will significantly speed up the process because all the compressible and solid libraries and solvers will not be compiled. To further speed up the compilation process, you can open dafoam/src/adjoint/Make/files\_Incompressible and delete the solvers, turbulence models, and other stuff you don’t need to compile. For example, if you want to test a new change for DASimpleFoam with the SA turbulence model and force objective, you can use this simplified [files\_Incompressible](https://dafoam.github.io/mydoc_get_started_files_incompressible_simplefoam.html).

## How to run a multipoint optimization?

Refer to the tutorial tutorials-main/NACA0012\_Airfoil/multipoint.

## How to run an optimization for 3D wings?

Refer to the tutorial tutorials-main/Onera\_M6\_Wing.

|  |
| --- |
|  |

**NOTE:** Once the above modifications are done, go to the tutorial folder and clean up the previous optimization results.

./Allclean

Finally, generate the mesh and run the new optimization using 4 CPU cores:

./preProcessing.sh && mpirun -np 4 python runScript.py 2>&1 | tee optLog.txt

## How to create a new objective function?

The objective functions for DAFoam are stored in the *src/adjoint/DAObjFunc* directory of [this repository](https://github.com/mdolab/dafoam). To create a new objective function, it is recommended to base it off of a similar existing function. You will need to first create yourObjFunc.H and yourObjFunc.C with your constructors, input/output definitions and implementation. Make sure you assign a new type name for your objective function by setting TypeName("nameOfYourNewObjFunc"); in the yourObjFunc.H file. Then in runScript.py, you will need to use the same name for the “type” key in daOptions-objFunc.

Then, you will need to add the paths to the files you created in *src/adjoint/Make/files\_Compressible* and *files\_Incompressible*. At this point you can attempt to re-compile DAFoam to test your code, follow [these instructions](https://dafoam.github.io/mydoc_installation_source.html) to build from source.

Finally, you will need to add the objective function to one of the runTests\_*.py located in the \*tests* directory. Again, use the other objective functions as an example.

When making a pull request, the code coverage test will not pass unless the output from testing your objective function (the value itself and sensitivity values) is copied into the corresponding tests/refs/DAFoam\_Test\_\*.txt file.

## How can I contribute to this website?

The DAFoam website is built based on JekyII. Most of the webpages are written in Markdown (.md) files. To make changes to the DAFoam website, please submit a pull request following these steps:

1. Sign up for an account on Github.com
2. Create a fork of the DAFoam website repo: https://github.com/DAFoam/DAFoam.github.io
3. Change the website .md files on your fork
4. When ready, submit a pull request

We assume you are familiar with pull request submission on Github.com. If not, please refer to [here](https://docs.github.com/en/pull-requests/collaborating-with-pull-requests/proposing-changes-to-your-work-with-pull-requests/creating-a-pull-request).

## How to cite DAFoam?

Please cite the following papers in any publication for which you find DAFoam useful.

Ping He, Charles A. Mader, Joaquim R.R.A. Martins, Kevin J. Maki. DAFoam: An open-source adjoint framework for multidisciplinary design optimization with OpenFOAM. AIAA Journal, 58:1304-1319, 2020. https://doi.org/10.2514/1.J058853

Ping He, Charles A. Mader, Joaquim R.R.A. Martins, Kevin J. Maki. An aerodynamic design optimization framework using a discrete adjoint approach with OpenFOAM. Computer & Fluids, 168:285-303, 2018. https://doi.org/10.1016/j.compfluid.2018.04.012

Latex bib keys:

@article{DAFoamPaper1,

title="{DAFoam}: An open-source adjoint framework for multidisciplinary design optimization with {OpenFOAM}",

author="He, Ping and Mader, Charles A and Martins, Joaquim RRA and Maki, Kevin J",

journal="AIAA Journal",

volume="58",

number="3",

pages="1304--1319",

year="2020",

publisher="American Institute of Aeronautics and Astronautics"

}

@article{DAFoamPaper2,

title="An aerodynamic design optimization framework using a discrete adjoint approach with {OpenFOAM}",

author="He, Ping and Mader, Charles A and Martins, Joaquim RRA and Maki, Kevin J",

journal="Computers \& Fluids",

volume="168",

pages="285--303",

year="2018",

publisher="Elsevier"

}