

# CBAero Analysis Interface Module (AIM) Manual

Ryan Durscher AFRL/RQVC and Marshall Galbraith MIT

January 15, 2026



---

0.1 Introduction . . . . .	1
0.1.1 CBAero AIM Overview . . . . .	1
0.2 Attribution . . . . .	1
0.3 CBAero TPS . . . . .	1
0.3.1 TPS input Dictionary . . . . .	2
0.3.2 TPS Group JSON String Dictionary . . . . .	2
0.4 CBAero Material Group . . . . .	3
0.4.1 Material Group JSON String Dictionary . . . . .	3
0.5 Geometry Representation . . . . .	3
0.6 AIM Inputs . . . . .	4
0.7 AIM Execution . . . . .	5
0.8 AIM Outputs . . . . .	6
0.9 CBAero Data Transfer . . . . .	7
0.9.1 Data transfer from CBAero (FieldOut) . . . . .	7
Bibliography . . . . .	9
Index . . . . .	11



## 0.1 Introduction

### 0.1.1 CBAero AIM Overview

A module in the Computational Aircraft Prototype Syntheses (CAPS) has been developed to interact (primarily through input files) with NASA Ames's CBAero [1]. CBAero (Configuration Based Aerodynamics) software package is an engineering level aero-thermodynamics tool for predicting the aerodynamic and aero-thermodynamic environments of general vehicle configurations. Currently only a subset of CBAero's input options have been exposed in the analysis interface module (AIM), but features can easily be included as future needs arise.

An outline of the AIM's inputs and outputs are provided in [AIM Inputs](#) and [AIM Outputs](#), respectively.

Geometric attributes recognized by the AIM are provided in [Attribution](#).

The accepted and expected geometric representation are detailed in [Geometry Representation](#).

Details of the AIM's automated data transfer capabilities are outlined in [CBAero Data Transfer](#)

## 0.2 Attribution

The following list of attributes drives the CBAero geometric definition.

- **capsLength** This attribute defines the length units that the \*.csm file is generated in. CBAero grids MUST be in units of meter, as such the geometry is scaled accordingly based on this value.
- **capsReferenceArea** [Optional] This attribute may exist on any *Body*. Its value will be used as the reference area in CBAero's input file with its units assumed to be consistent with the attribute "capsLength". No conversion takes place if "capsLength" isn't set. This value may be alternatively set through an input value, "ReferenceArea" (see [AIM Inputs](#))
- **capsReferenceChord** and **capsReferenceSpan** [Optional] These attribute may exist on any *Body*. Their value will be used as the reference moment lengths in CBAero's input file with their units assumed to be consistent with the attribute "capsLength". No conversion takes place if "capsLength" isn't set. These values may be alternatively set through an input value, "Moment\_Length" (see [AIM Inputs](#))
- **capsReferenceX**, **capsReferenceY**, and **capsReferenceZ** [Optional] These attribute may exist on any *Body*. Their value will be used as the center of gravity (CG) location in CBAero's input file with their units assumed to be consistent with the attribute "capsLength". No conversion takes place if "capsLength" isn't set. These values may be alternatively set through an input value, "Moment\_Center" (see [AIM Inputs](#))

## 0.3 CBAero TPS

Structure for the TPSin tuple = (Input, Value).

### 0.3.1 TPS input Dictionary

For the TPSin dictionary the following keywords ( = default values) may be used:

- **SoakOutTime**  
Soak out time
- **MarginOnHeatLoads\_Multiplier**  
Margins on heat load multiplier
- **MarginOnRecession**  
Margins on heat load multiplier
- **InitialTemperature**  
Initial temperature
- **MarginOnInsulation**  
Margin on the insulation
- **BlowingFactor**  
Blowing factor
- **RunWhichRSSedEnvCase0123**  
Blowing factor
- **EnthalpyTableFileName**  
Enthalpy table file name
- **TimeToOption3**  
Time to option 3 in seconds

Structure for the TPS\_Group tuple = ("TPS Name", "Value"). The "Value" must be a JSON String dictionary. If "TPS Name" is a capsGroup, then groupName should not be specified.

### 0.3.2 TPS Group JSON String Dictionary

For the JSON string "Value" dictionary (e.g. "Value" = {"surfaceType": 1, "emissivity": 0.8}) the following keywords ( = default values) may be used:

- **StackUpListFileName**  
Full path to stack up list file name
- **StructuresStackUpListFileName**  
Full path to structure stack up list file name

- **SplitLineMarginsFileName**  
Full path to split line margins file name
- **EnvironmentMarginsFileName**  
Full path to environment margins file name
- **BumpFactorFileName**  
Full path to environment margins file name
- **UseOnlyOneMaterial**  
Integer flag for material usage
- **UseHeatLoadSubZones**  
Integer flag subzone heat load
- **NumberOfHeatLoadSubZones**  
Integer number of heat load subzones
- **groupName = "(no default)"**  
Single or list of `capsGroup` names on which to apply the material (e.g. "Name1" or ["Name1", "Name2", ...]).

## 0.4 CBAero Material Group

Structure for the Material\_Group tuple = ("Material Name", "Value"). The "Value" must be a JSON String dictionary.

If no Material\_Group is specified, a default material of "Catalytic" with emissivity of "0.8" is applied to all capsGroups.

### 0.4.1 Material Group JSON String Dictionary

For the JSON string "Value" dictionary (e.g. "Value" = {"surfaceType": 1, "emissivity": 0.8}) the following keywords ( = default values) may be used:

- **surfaceType**  
The surface type must be a string integer, e.g. "0" or "10, <br> or a case insensitive partial match to one of:  
 - "Non Catalytic" <br> - "Fully Catalytic" <br> - "RCG" <br> - "TUF1" <br> - "ORCC Coated ACC" <br>  
 - "PCC Coated NEXTEL 440" <br> - "RLV Design Goal" <br> - "SIRCA" <br> - "TAB1" <br> - "Grey C-9  
 Coated NEXTEL 440" <br> - "SiC – SiC" <br> - "C-CAT" <br> - "LVP Coated ACC" <br> - "SiC Coated  
 Carbon – Russian" <br> - "INCONEL 617 PreOxidized" <br> - "Mars Fully Catalytic, No O2 Recombination"  
 <br> - "Venus Fully Catalytic, No CO Oxidation" <br> </ul> <ul> <li> <B>emissivity</B> </li> <br>  
 Emissivity of the material [0 - 1] </ul> <ul> <li> <B>groupName = "(no default)"</B> </li> <br>  
 Single  
 or list of <c>capsGroup</c> names on which to apply the material (e.g. "Name1" or ["Name1", "Name2", ...]).

## 0.5 Geometry Representation

The geometric representation for the CBAero AIM requires that the body be either a solid body (SOLIDBODY) or a manifold sheet body (SHEETBODY).

## 0.6 AIM Inputs

The following list outlines the CBAero inputs along with their default values available through the AIM interface.

- **Proj\_Name = "cbaero\_CAPS"**  
This corresponds to the project "root" name.
- **Mach = 0.0 (default) or [0.0, ... , 0.0]**  
Mach number (can be a single or array of values).
- **Dynamic\_Pressure = 0.0 (default) or [0.0, ... , 0.0]**  
Dynamic pressure [bar] value (can be a single or array of values).
- **Alpha = 0.0 (default) or [0.0, ... , 0.0]**  
Angle of attack [degree] (can be a single or array of values).
- **Beta = 0.0 (default) or [0.0, ... , 0.0]**  
Sideslip angle (can be a single or array of values).
- **ReferenceArea = NULL**  
This sets the reference area for used in force and moment calculations. Alternatively, the geometry (body) attribute (see [Attribution](#)) "capsReferenceArea" maybe used to specify this variable (note: values set through the AIM input will supersede the attribution value).
- **ReferenceChord = NULL**  
This sets the reference chord for used in force and moment calculations. Alternatively, the geometry (body) attribute (see [Attribution](#)) "capsReferenceChord" maybe used to specify this variable (note: values set through the AIM input will supersede the attribution value).
- **ReferenceSpan = NULL**  
This sets the reference span for used in force and moment calculations. Alternatively, the geometry (body) attribute (see [Attribution](#)) "capsReferenceSpan" maybe used to specify this variable (note: values set through the AIM input will supersede the attribution value).
- **Moment\_Center = [0.0, 0.0, 0.0] (NULL)**  
Array values correspond to the x, y, and z center of gravity (CG) locations [meter]. Alternatively, the geometry (body) attributes (see [Attribution](#)) "capsReferenceX", "capsReferenceY", and "capsReferenceZ" may be used to specify the center of gravity, respectively (note: values set through the AIM input will supersede the attribution values).
- **Flow\_Type = "Inviscid"**  
Type of flow to consider. Options (=corresponding integer code): FreeTransition(=0), Laminar(=1), Turbulent(=2), Inviscid(=3).
- **Critical\_Transition = 220.0**  
Critical ratio of Re-theta (Reynolds based on momentum thickness) and Ma (Mach number) for transition.
- **Planet = "EARTH"**  
Planet type. Options include "MERCURY", "VENUS", "EARTH", "MARS", "JUPITER", "SATURN", "URANUS", "NEPTUNE", and "PLUTO".
- **Default\_Body\_Method = "ModifiedNewtonian"**  
Default hypersonic base method. Options (=corresponding integer code): ModifiedNewtonian(=3), TangentCone(=21), TangentConeNormalShock(=22), TangentWedge(=31), TangentWedgeNormalShock(=32), FreeMolecular(=99).
- **Default\_Wing\_Method = "ModifiedNewtonian"**  
Default hypersonic aerodynamic wing method. Options (=corresponding integer code): ModifiedNewtonian(=3), TangentCone(=21), TangentConeNormalShock(=22), TangentWedge(=31), TangentWedgeNormalShock(=32), FreeMolecular(=99).



- **Default\_Low\_Speed\_Method = "FastPanel"**  
Default low speed method. Options (=corresponding integer code): FastPanel(=1), LowAR(=2).
- **Leading\_Edge\_Suction = 1.00**  
Default low speed method integer tag. Range [-1.0, 1.0]
- **Mangler\_Setting = "2D"**  
Default low speed method. Options (=corresponding integer code): "2D"(=1), "Axisymmetric"(=2), "Auto"(=3).
- **Aero\_Surface = NULL**  
Defines the type of aero. surface by associating a "capsGroups" attribute name with a particular panel method - ("capsGroup Name", "Value"), where "Value" can either be "Body", "Base", "Wing", "Inlet", "Cowl", or "↵ Nozzle". If a capsGroup panel method is not defined it will be assumed to be a "Body".
- **Material\_Group = NULL**  
Defines the type of aero. surface by associating a "capsGroups" attributes with a particular material group - ("Material Name", "Value"), where "Value" must be a JSON String dictionary, see [CBAero Material Group](#) for additional details.
- **Trajectory = NULL**  
List of trajectory file names
- **TPSin = NULL**  
Defines the .tpsin input file for TPSSizer, see [CBAero TPS](#) for additional details.
- **TPS\_Group = NULL**  
Defines the type of aero. surface by associating a "capsGroups" attributes with a particular TPS zone - ("TPS Name", "Value"), where "Value" must be a JSON String dictionary, see [CBAero TPS](#) for additional details.
- **TPSSizer = NULL**  
Path to TPSSizer executable, e.g. "tpssizerv36\_linux". If not specified only CBTPS will be executed when TPS information is provided.
- **FIAT = NULL**  
Name of FIAT executable argument used with TPSSizer
- **NumParallelCase = 1**  
Set CBAero -mp to define number of Mach, dynamic pressure, and angle of attack cases to solve simultaneously.  
May be used in conjunction with NumThreadPerCase.
- **NumThreadPerCase = 1**  
Set CBAero -omp to define number of threads to solve a each Mach, dynamic pressure, and angle of attack cases.  
May be used in conjunction with NumParallelCase.
- **Mesh\_Morph = False**  
Project previous surface mesh onto new geometry.
- **Surface\_Mesh = NULL**  
A Surface\_Mesh link.

## 0.7 AIM Execution

If auto execution is enabled when creating an CBAero AIM, the AIM will execute refine just-in-time with the command line:

```
cbaero $(cat cbaeroInput.txt) > cbaeroOutput.txt
```

where preAnalysis generated the file "cbaeroInput.txt" which contains commandline arguments for cbaero.

If the TPS input is specified, then TPSSizer is also executed with

```
cbtps $(cat cbtpsInput.txt) > cbtpsOutput.txt
<TPSSizer> $(cat tpssizerInput.txt) > tpssizerOutput.txt
```

Note that TPSSizer is only executed if a TPSSizer executable is specified in the inputs (see [AIM Inputs](#)).

The analysis can be also be explicitly executed with `caps_execute` in the C-API or via `Analysis.runAnalysis` in the pyCAPS API.

Calling `preAnalysis` and `postAnalysis` is NOT allowed when auto execution is enabled.

Auto execution can also be disabled when creating an refine AIM object. In this mode, `caps_execute` and `Analysis.runAnalysis` can be used to run the analysis, or refine can be executed by calling `preAnalysis`, `system call`, and `posAnalysis` as demonstrated below with a pyCAPS example:

```
print ("\n\preAnalysis.....")
cbaero.preAnalysis()

print ("\n\nRunning.....")
cbaero.system("cbaero $(cat cbaeroInput.txt) > cbaeroOutput.txt"); # Run via system call

if TPS:
    cbaero.system("cbtps $(cat cbtpsInput.txt) > cbtpsOutput.txt") # Run via system call
    cbaero.system("<TPSSizer> $(cat tpssizerInput.txt) > tpssizerOutput.txt"); # Run via system call

print ("\n\npostAnalysis.....")
cbaero.postAnalysis()
```

## 0.8 AIM Outputs

The following list outlines the CBAero outputs available through the AIM interface. All variables currently correspond to values found in the \*.plt file

Reiterate inputs (based on cases):

- **Beta** = Sideslip [degree].
- **Alpha** = Angle of attack [degree].
- **Dynamic\_Pressure** = Dynamic pressure [bar].
- **Mach** = Mach number.

Per-Trb:

- **PerTrb** = PerTrb.

Net Forces - Pressure + Viscous:

- **CLtot** = The lift coefficient.
- **CDtot** = The drag coefficient.
- **CMytot** = The moment coefficient about the y-axis.
- **LoDtot** = Lift to drag ratio.

Pressure Forces:

- **CL\_p** = The lift coefficient - pressure contribution only.

- **CD\_p** = The drag coefficient - pressure contribution only.

Viscous Forces:

- **CL\_v** = The lift coefficient - viscous contribution only.
- **CD\_v** = The drag coefficient - viscous contribution only.

Aero-thermal:

- **Stagnation\_Temperature** = Stagnation temperature [K].
- **Stagnation\_Radius** = Stagnation radius [m].
- **Convective\_Flux** = Convective heat flux [W/cm<sup>2</sup>].
- **Radiative\_Flux** = Radiation heat flux [W/cm<sup>2</sup>].

Trefftz:

- **CL\_Trefftz** = Trefftz lift coefficient.
- **CD\_Trefftz** = Trefftz drag coefficient.

## 0.9 CBAero Data Transfer

The CBAero AIM has the ability to transfer field data from the AIM using the conservative and interpolative data transfer schemes in CAPS.

### 0.9.1 Data transfer from CBAero (FieldOut)

- **"HeatFlux\_IMach\_#\_IDynp\_#\_IAlpha\_#"**  
Retrieves heat flux (QDOT TOATL) from the adb database file, where "#" should be replaced by the corresponding 0-based input index for Mach, Dynamic\_Pressure, and Alpha input arrays. If only one analysis is performed, then the suffix "\_IMach\_#\_IDynp\_#\_IAlpha\_#" may be omitted.



# Bibliography

- [1] David Kinney. Aero-thermodynamics for conceptual design. Number AIAA-2004-31, Reno, NV, Jan. 2004. 42nd AIAA Aerospace Sciences Meeting and Exhibit. [1](#)



# Index

AIM Execution, [5](#)

AIM Inputs, [4](#)

AIM Outputs, [6](#)

Attribution, [1](#)

CBAero Data Transfer, [7](#)

CBAero Material Group, [3](#)

CBAero TPS, [1](#)

Geometry Representation, [3](#)

Introduction, [1](#)