Advanced Fan Test Facility

MATLAB Post-Processer Documentation

Created by Morgan McNulty, SEAP

August 5, 2019

Email: [morganm804@gmail.com](mailto:morganm804@gmail.com)

Phone: 609-342-7242

## Description of App:

The AFTF MATLAB Post-Processer uses AMCA 210-07 standards to process raw data from the Advanced Fan Test Facility and creates a PDF report with necessary dimensions, calculations, and fan performance curves.

## General Instructions for Use:

To run app, click *AFTF\_processing.mlapp*

### Adding a New Test Series:

To add a new test series, click the “add test set up” button and enter all the properties of the test series, including the nozzle dimensions in the Nozzle Set Up tab. Once you click enter, the new test series is stored as a new testSetUp object as defined in the *testSetUp.m* class. All test set up objects are stored from session to session in the *testSetUp.mat* file.

### Editing a Test Series:

Select the test set up you want to edit from the drop down in the left-hand panel. The edit panel will appear with all fields auto-filled with their current values for that test set up. Change the necessary values and press enter to store the changes.

### Deleting a Test Series:

Select the test series to delete from the drop down in the left-hand panel. The app will ask for confirmation of deletion. The test series and all of its properties will be permanently deleted after clicking confirm.

### Generating Fan Performance Curve:

To generate a fan performance curve, select the correct test set up from the drop down menu, enter the number of data points taken and enter/ select the file path of the raw text file output by the DAQ. The program will use dimensions from the selected test set up and read the data from the text file to calculate volumetric flow rate and static pressure.

To explore curve fitting with the generated fan performance curve, click tools in the figure window then select basic fitting. Click the right arrow button twice in the basic fitting window to access additional functionality.

### Generating Report:

To generate a full PDF report with dimensions, calculations, the fan performance curve, estimates of shutoff and free delivery values, and tables of calculated and raw data, fill out all of the fields in the left-hand panel then click generate report. Report will be generated using entered data and properties of selected test set up.

### Importing Files:

Currently, the *importFanData* function imports columns of data into pre-named arrays. The file reads raw data from a file with the following column configuration:

* 1. Run Number
  2. targetDiskPos\_percent
  3. targetConePos\_percent
  4. NominalNfan\_RPM
  5. Nmotor\_RPM
  6. Nfan\_instantaneous\_RPM
  7. Nfan\_RPM
  8. Tq\_lb
  9. Ps2L\_wg
  10. deltaPs2\_5\_2L\_wg
  11. Ps2\_5L\_wg
  12. Ps7L\_wg
  13. Ps5L\_wg
  14. deltaPs5\_6L\_wg
  15. Ps6L\_wg
  16. T2\_5aL\_F
  17. T2\_5bL\_F
  18. T2\_5cL\_F
  19. T7aL\_F
  20. T7bL\_F
  21. T7cL\_F
  22. Trtd7L\_F
  23. T5aL\_F
  24. T5bL\_F
  25. T5cL\_F
  26. Trtd5L\_F
  27. T6aL\_F
  28. T6bL\_F
  29. T6cL\_F
  30. Trtd6L\_F
  31. Pba\_mbar
  32. Pbb\_mbar
  33. Pbc\_mbar
  34. Ps0\_wg
  35. T0a\_F
  36. T0b\_F
  37. T0c\_F
  38. Tdewa\_F
  39. Tdewb\_F
  40. Tdewc\_F
  41. Rha\_percent
  42. Rhb\_percent
  43. Rhc\_percent
  44. DLC\_in
  45. DL0\_in
  46. DL45\_in
  47. DL90\_in
  48. DL135\_in
  49. DL180\_in
  50. DL225\_in
  51. DL270\_in
  52. DL315\_in
  53. EFanPitch\_deg
  54. DiskPos\_percent
  55. ConePos\_percent

## Notes on Calculations:

* Calculations are based on AMCA 210-07 standards, Figure 12. Calculations for losses or for adjustment of calculations for different pressures/temperatures have not yet been incorporated. To do this, refer to instructions on adding additional values.
* Functions for H (torsion element) and compressibility factor (Kp) have been programmed in but are not used in calculations of Q and P
* Measurements taken by multiple sensors, such as the temperature readings taken by the three thermocouples at each station, are reconciled using the *rmoutliersandaverage.m* function. This function defines outliers as those falling more than 1 standard deviation above or below the mean for standard deviations greater than 3 or as values more than 2 standard deviations above or below the mean for standard deviations less than 3. Outliers are defined this way because there are only three data points (A, B, and C sensors), so when one of those points is significantly different than the others, the standard deviation is so large that the obvious outlier is still within 2 standard deviations of the mean. However, if the data points are very close, excluding values only 1 standard deviation away from the mean may exclude values which are actually correct.
* Calculations assume that inlet is rectangular, so hydraulic diameter is calculated in *findFinalValues.m* function based on length and width entered in GUI, if transition duct is not rectangular, just input the circular diameter as Transition Ducting Length (a) and edit *findValues.m* so that D2 = a

# Instructions for Editing:

## Instructions for Adding New Properties to Test Run (left-hand panel in app):

1. In App Designer
   1. In Design View
      1. Drag correct component into graphpanel
   2. In Code View
      1. In generateReport, add a line setting the new variable name equal to the value of the field entered in the app (*varName =* app.*editFieldName*.Value)
      2. In generateReport, add the *varName* as a parameter at the end of the save function, this will allow you to access the variable value in the report generation script in MATLAB

## Instructions for Adding New Property to Test Set Up:

1. In MATLAB Workspace
   1. In testSetUp.m
      1. Add the property name to the properties list
      2. Add the property as a parameter to the testSetUp constructor
      3. Add newTestSetUp.*propertyName* = *propertyName*; to constructor to assign prorperty to new testSetUp object
   2. In addNewSetUp.m
      1. Add property name to parameters list in function definition (remember that order matters!)
      2. Add property name to parameters list in call to constructor (line 6) (order matters!)
   3. In editTestSetUp.m
      1. Add property name to parameters list in function definition (remember that order matters!)
      2. Add property name to parameters list in call to constructor (line 3) (order matters!)
2. In App Designer
   1. In Design View
      1. Drag correct component into addTestSetUpPanel and into editTestSetUpPanel and format (must make these panels visible under the interactivity tab in the panel properties menu on the right)
   2. In Code View
      1. In addNew(), add the value of the new component into the parameters list of call to addNewTestSetUp(), remember that order matters
      2. In editSetUp(), (), edit the value of the current component by inserting the value of the editPropertyComponent to the parameters list of call to editTestSetUp(), remember that order matters
      3. Run app and enter data for new field, press enter then close app, then in showEditPanel(), autofill the current property by adding app*.editComponentName.*Value= list(1, app.index).*propertyname;*

## To Add Additional Values for Reporting and to Calculate Corrected Q and P:

### To add additional values for reporting (such as fan power output (H) or fan total efficiency):

Create new MATLAB functions that calculate these values based on the raw data. Be sure to only use raw collected data as parameters and then calculate other needed values within function using other MATLAB functions. This will ensure simplicity.

* In *calcFinalValues.m,* add new calculated value as a new returned value in function definition, also add any additional raw data points that are used in new calculations to parameters list.
* Refer to comments in *findFinalValues.m* to add calculations of the new value.
* In App Designer, in *generateCurve\_app()* and in *generateReport* add new parameters to call to *findFinalValues.*
* In App Designer, in *generateReport,* add new variable to be reported to brackets as instructed by comments. Additionally, add the new variable name to the save function so that it can be accessed in the *createReport* function. Refer to additional instructions in modifying report section for a guide to adding graphs.

### To use corrected Q and P in calculations:

* Create new MATLAB functions with necessary calculations and create new functions that calculate corrected Q and P. To use these values in reporting, replace the findQ and findP functions with functions for adjusted values.

## Instructions for Modifying Report:

The report is created in the *createReport.m* MATLAB script. Refer to [MATLAB Report Generator Documentation](https://www.mathworks.com/help/rptgen/index.html?s_tid=CRUX_lftnav) for more information about formatting report and adding different types of content.

### How to Access Values from App:

* **Test Run Values** (those that are entered for each new run in the left hand panel on the app) are stored in the *testData.mat* file and can be accessed using just their variable name. When adding new properties to the test run panel, ensure that their variable name is added to the save function in the *generateReport* function in App Designer so that their value can be accessed in this script.
* **Test Set Up Values** (those that are properties of a test set up, such as the FCA) are stored as properties of test set up objects defined in the *testSetUps.m* class. They can be accessed in the createReport.m file by writing ‘testSetUpList(currentIndex).varName’
* All **raw data from DAQ**  is stored in *reportInfo.mat* and can be accessed using direct variable names

**Important Notes:**

* For date fields, number fields, or other non-string/ character input, those variables must be converted to string using appropriate ‘*var’2string* method before being printed in report
* For multi-line text input, value must be displayed as a table, for example, test personnel are entered as multi line input and is displayed in the report using the following code:

testPersonnelTable = Table(testSetUpList(currentIndex).testPersonnel);

testPersonnelTable.TableEntriesInnerMargin='3px';

add(testInfo, "Test Personnel: ");

add(testInfo, testPersonnelTable);

### Adding New Graphs and Reported Values:

* To add a new graph, ensure that all instructions for adding additional values for reporting have been followed.
* In section where you want to add the new graph, copy and paste the following code and edit italicized/ red values

[xData, yData] = prepareCurveData( *xValues*, yValues );

% Set up fittype and options.

ft = fittype( *fit type: smoothingspline, linearinterp, etc.* );

% Fit model to data.

[fitresult, gof] = fit( xData, yData, ft );

% Plot fit with data.

figure( 'Name', *figure name* );

h = plot( fitresult, xData, yData );

legend('*xdata* vs *ydata’*, 'Fit with *fittype’*);

grid on

title(*graph title*);

xlabel(*x axis title*);

ylabel(*y axis title*);

add(*section name*, Figure());

# Troubleshooting:

#### If test set up dropdown is not working/ error when editing/deleting test set ups:

In the MATLAB workspace, load the testSetUpList and click on namesList to view the values in names. If any values are null, type *deleteTestSetUp(index)* into the command line with the index for each null value. This should fix the issue. If some test ups were deleted, re- enter that data using add new test set up.