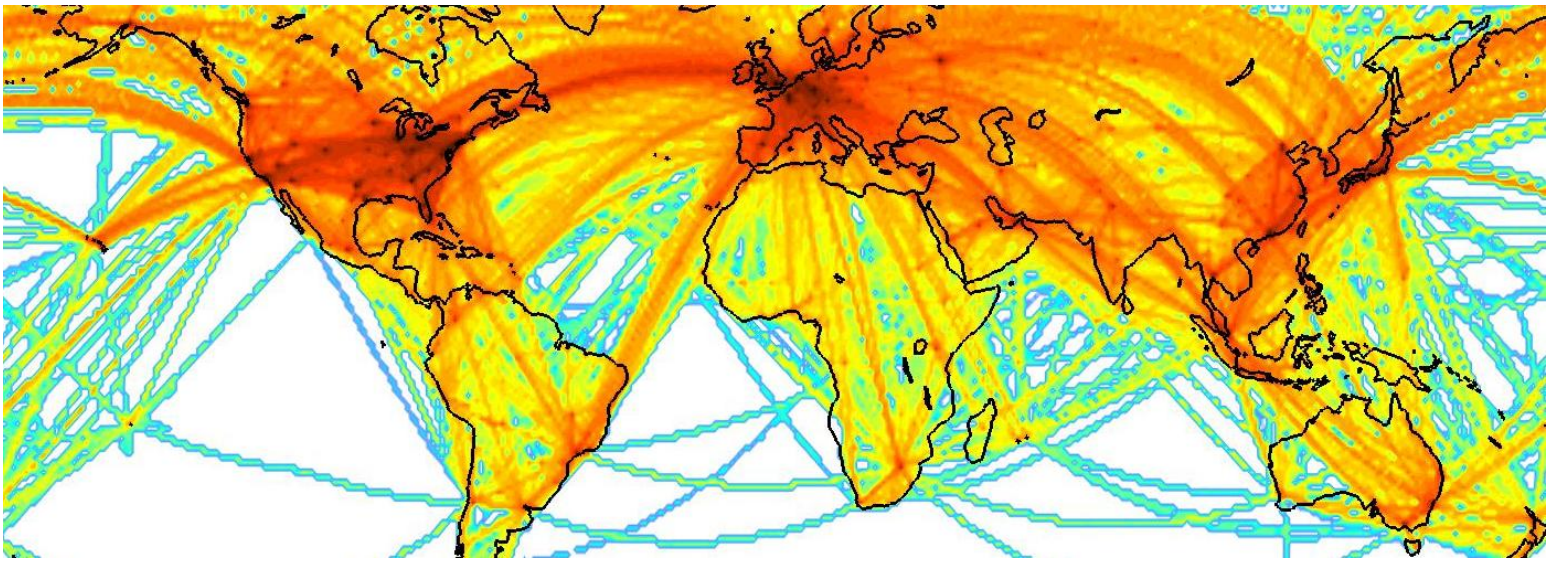
Laboratory for Aviation
and the Environment
Massachusetts Institute of Technology



Aviation Emissions Inventory Code (AEIC) User Manual (R1)

Nicholas Simone, Marc Stettler, Sebastian Eastham and Steven Barrett

A technical note by the MIT Laboratory for Aviation and the Environment in collaboration with the Cambridge University Energy Efficient Cities Initiative

Date: January 2013
Report No: LAE-2013-001-N
Website: LAE.MIT.EDU

Contents

Revision History

Revision	Comments
R1	Initial Release

1 Document Purpose

This document describes version 2.0 of the Aviation Emissions Inventory Code (AEIC). Its purpose is to provide an overview of the code functionality and how to use it. A brief description of the code structure is also included, in case it needs to be modified for certain studies.

The documentation for version 1.0 is also included in Appendix A.

2 Contact

Contact Steven Barrett (Massachusetts Institute of Technology) (sbarrett@mit.edu) with questions.

3 Other Resources

AEIC (v1.0 and v2.0) can be downloaded from MIT's Laboratory for Aviation and the Environment (LAE) website (http://lae.mit.edu). Emissions datasets and other publications relating to the code are also available on the website.

A publication and thesis outlining the methodologies used are forthcoming. Please check the LAE website for details.

4 General

AEIC v2.0 was written in MATLAB R2010a and should work with all newer versions. It can be run through the MATLAB GUI or the command line. The MATLAB mapping toolbox is required for generating non-LTO emissions.

A list of airports that are supported by AEIC can be found in 'ALLairports_99pcPAX.txt' in the 'Model_Files' directory.

5 Directory Structure

There are five main directories of interest in AEIC:

1. '.': The top level directory. It contains the main scripts and functions for running the model.

2. ‘./Model_Files’: This directory contains the scripts and functions that are called from the main scripts.

3. ‘./Schedule_Files’: This directory contains the 2005 schedule information for all of the airports included in AEIC, as well as some other files for decoding aircraft and airport designations.

4. ‘./Logs’: Logs of screen output during AEIC runs are stored here.

5. ‘./Emissions_Files’: Output in .mat format from AEIC runs is stored here.

6 First Time Setup

If AEIC is going to be used to calculate non-LTO (landing and takeoff) emissions, a license for BADA must be obtained from EUROCONTROL (http://www.eurocontrol.int/services/bada). After a license is obtained, follow EUROCONTROL’s instructions to download all associated BADA files. After extracting AEIC, the BADA files must be placed in the directory ‘./Model_Files’ in a folder named ‘BADA’.

If AEIC is going to be used to calculate LTO emissions only, BADA is not required. After the code is downloaded, it only needs to be extracted in order to run it.

7 Running the Model

AEIC can be run using a script or a function call. Using a script is the normal method for running. However, a function wrapper has been created to aid in running Monte Carlo simulations, where only a few model outputs are of interest. The code can be parallelized to use multiple cores or nodes on a server during sequential model runs. Each of these methods is described here.

7.1 Main Script: ‘Sequence_World_Func.m’

The script ‘Sequence_World_Func.m’ contains all of the necessary function calls and code to run the AEIC model. It can be altered to run a different set of airports or generate different output by simply activating certain switches or altering lists. Available run options are covered first, followed by required inputs.

7.1.1 Available Run Options

This section covers available run options. They are listed in the order they come up in ‘Sequence_World_Func.m’. The line number each option appears in is also included. Bold

words represent an option or input described in this manual, while italic words refer to available values for options. Defaults are identified with [DEFAULT].

- **runType (Line 30):** This option specifies the type of run that you would like to perform. The options are *allDep [DEFAULT]*, *allArr*, or *allFlights*. The **runType** affects which flights at each of the airports in the **AIRPORT_list** are analyzed. If *allDep* is selected, flights departing from each airport in **AIRPORT_list** will be run. If *allArr* is selected, flights arriving at each airport in **AIRPORT_list** will be run. *allFlights* runs the LTO portion of all flights for each airport in **AIRPORT_list** and is intended to be used when only LTO emissions are of interest. It mimics the functionality of AEIC v1.0.

- **runNonLTO (Line 33):** This option controls whether the non-LTO portion of each flight is run (operations above 3000 above field level). Set to 1 [DEFAULT] to run all portions of a flight and set to 0 to run LTO only. NOTE: if **runType** is set to *allFlights*, **runNonLTO** will automatically be set to zero.

- **makeGrid (Line 46,52):** This option controls whether AEIC generates gridded output. Set to 1 to generate gridded output and set to 0 [DEFAULT] to not generate gridded output. The value on line 46 is used when 'Sequence_World_Func.m' is run as a script and the value on line 52 is used when 'Sequence_World_Func.m' is run using the function wrapper. For specifics on gridding methodology, please refer to Report #LAE-2012-012-N at lae.mit.edu.

- **resolveTemporally (Line 47,53):** This option controls whether AEIC computes a temporal resolution to its output. Set to 1 to temporally resolve output and set to 0 [DEFAULT] to generate annual output only. The value on line 47 is used when 'Sequence_World_Func.m' is run as a script and the value on line 53 is used when 'Sequence_World_Func.m' is run using the function wrapper. See also **tempResolve_HourVec** in the inputs section.

- **mcSwitch, hourlySwitch, pmnvolSwitch (Lines 65, 85, 91):** Please refer to the AEIC v1.0 README file (Appendix A) for details on these options. They are only supported for LTO operations using a **runType** of *allFlights*.

- **useAnalyzeList (Line 186):** Switch to use **analyzeAirportList** (see details in inputs section)**.** Set to 1 [DEFAULT] to use a list and set to 0 if you don't want to use **analyzeAirportList.** This switch only has an effect if **runType** is set to *allDep* or *allArr*. Note that if **useAnalyzeList** is set to 0 and an airport in **AIRPORT_list** has a flight to or from an airport that is not supported by AEIC, the run will not successfully complete.

- **saveCruiseTotals (Line 515):** This option controls whether non-LTO portions of each flight, with lateral inefficiencies incorporated, are saved. Set to 1 [DEFAULT] to save this data and set to 0 to not save the data. This switch is useful to have enabled when looking at the breakdown of fuel burn for different flights. If only grids are being examined, this option can be disabled (set to 0) to conserve memory.

7.1.2 Required Inputs

This section covers required inputs for running AEIC. They are listed in the order they come up in 'Sequence_World_Func.m'. The line number each input appears in is also included. Bold words represent an option or input described in this manual, while italic words refer to available values for options. The type of input (real, string, etc.) is specified as well.

- **OAG_Year (Line 8):** Real Input. Year that flights are being analyzed for. Only the year 2005 is currently supported.

- **SCENARIO (Line 10):** String Input. Name of scenario being analyzed. Used to create directories and in output logs/files.

- **AIRPORT_list (Line 43, 49, 50):** Cell Array Input. List of airports to investigate. This is the list that **runType** looks at**.** For example, if a **runType** of *allDep* is specified, AEIC will run all departures between each airport in **AIRPORT_list** and each airport in **analyzeAirportList**. For a **runType** of *allFlights*, the LTO operations for every flight at each airport in **AIRPORT_list** will be run.

  Lines 49 and 50 are meant as examples of inputting a list directly, while line 43 shows how to import it from a file. In the 'Model_Files' directory, there are precompiled airport lists for 99% of global passenger enplanements (ALLairports_99pcPAX.txt = all supported airports), 99% of US passenger enplanements (USairports_99pcPAX.txt), 95% of US passenger enplanements (USairports_95pcPAX.txt), and the top UK airports by passenger enplanements (Top40UK.txt).

- **analyzeAirportList (Line 187):** Cell Array Input. List of connecting airports to investigate. By default, this is set to be the same as **AIRPORT_list**, but can be specified separately. This list only affects the run if **useAnalyzeList** is enabled and **runType** is not *allFlights.*

  The specific interaction between **AIRPORT_list** and **analyzeAirportList** is best illustrated with a few examples. Assume **runType** is set to *allDep,* **AIRPORT_list** contains BOS and

DEN and **analyzeAirportList** also contains BOS and DEN. All flight from departing from BOS and going to DEN and all flights departing from DEN and going to BOS will fly (both ways are covered because **AIRPORT_list** = **analyzeAirportList)**. Now assum we change **analyzeAirportList** to include LAX only. Now, all flights departing from BOS and going to LAX, and all flights departing from DEN and going to LAX are flown. If **runType** is changed to *allArr*, then all flights arriving at either DEN or BOS from LAX would be flown.

- **tempResolve_HourVec (Line 27, temporalResolution.m):** Real Input. This vector is only used when resolving flights temporally (**resolveTemporally=1**). The input is a set of hours used as "breakpoints" for the temporal resolution. Flights between each set of hours will be summed and output. Note that currently all flights are flown, regardless of if they fit into the hours given here. By default, **tempResolve_HourVec** is set up to provide monthly emissions.

- **fuelBurnLat, fuelBurnLon, fuelBurnAltFt (Line 63-65, allocateResultsWithIneff.m):** Real Input. These vectors define the grid points utilized when gridding the output from AEIC (**makeGrid = 1**). The data is gridded using a nearest neighbor allocation scheme (emissions applied to closest node). **fuelBurnLat** / **fuelBurnLon** should be in degrees and **fuelBurnAltFt** should be in feet. The default grid is a Generic 1°x1° grid with a vertical resolution of 200 feet. Other grid definitions in the file are GMAO 2°x2.5° and GMAO 4°x5° degree grids with a vertical resolution of 200 feet. Any other grid may be manually defined.

7.2 Function Wrapper: 'runAEIC.m'

The function wrapper 'runAEIC.m' allows for AEIC to be called as a function, and output only defined parameters. This is especially helpful for Monte Carlo simulations. An example of its implementation for a Monte Carlo simulation is included in the script 'loopFunc.m'. The output is a vector with the values of the parameters requested. The inputs currently required are:

- **AIRPORT_list:** See description under 'Required Inputs' section.

- **outList:** String of parameters desired to be output. See line 18 of 'loopFunc.m' for an example

- **mcSwitch:** Switch (string) for Monte Carlo – 'off' or 'on'.

- **mcrvVector:** Real vector of random variables for Monte Carlo. See 'loopFunc.m' for more details on definition. Size should be 49x1. Each random variable is for a variable in the Monte Carlo. See 'uncertaintySpecs.m' for details of Monte Carlo variables.

- **mcsVector:** Real vector of switches (0 or 1) for Monte Carlo. See 'loopFunc.m' for more details on definition. Size should be 49x1. Each switch is for a variable in the Monte Carlo. See 'uncertaintySpecs.m' for details of Monte Carlo variables.

- **saveName:** String. Name of .mat file to save output parameters as. If you would not like output saved, simply enter an empty string ('').

Other inputs are as defined in the 'Required Inputs' section. Any input can be added by editing 'runAEIC.m' and 'Sequence_World_Func.m' accordingly.

7.3 Multicore Processing: 'loopFunc_multicore.m' and 'startAEICSlave.m'

The script 'loopFunc_multicore.m'contains an example of a multicore implementation of AEIC for a Monte Carlo simulation. It can be modified as needed. Before running this script, an empty directory called 'multiTest' must be created in the top level directory of AEIC and the following multicore processing package must be downloaded from the MATLAB File Exchange:

http://www.mathworks.com/matlabcentral/fileexchange/13775

The downloaded files should be placed in a directory named 'multicore' in the top level directory of AEIC. To run the script, first start up MATLAB sessions and run 'startAEICSlave.m'. This will initiate the slave workers that will run. Then run the 'loopFunc_multicore.m' script. The multicore capability can be used to utilize multiple cores on single machine or multiple nodes on a server cluster. Currently, it can only be used to run successive simulations of AEIC. A single simulation is not yet parallelized.

8 Model Output

Table 1 lists some of the main variables that are available as output from AEIC v2.0. Currently, all output is saved as a .mat file. It can then be manipulated and output to various formats as needed using the built in MATLAB capability. Other variables than those listed below exist in the workspace but may be variables used for calculations or interim variables. Output available will vary depending on run options.

The units of the emitted species are:

- Fuel Burn: kilograms
- CO, CO_2, SO_2, SO_4 : grams
- HONO, NO, NO_2, NO_X: grams, using NO_2 mass basis
- HC: grams, using CH_4 mass basis

Table 1: AEIC Outputs

Variable	Description
[species]_[LTO_Segment]_annual	Emissions output for a particular [species] emitted during specific [LTO_Segment] at airport for the entire year.
[species]_[LTO_Segment]_hour_[#]	Temporally resolved output for a particular [species] emitted during specific [LTO_Segment] at airport. hour_[#] represents the hour the emissions are summed. The emissions included consist of those occurring between that hour and the next hour in tempResolve_HourVec.
[species]_cruiseFlightByLeg	Amount of particular [species] emitted for each flight segment. **NOTE: No lateral inefficiency incorporated.**
[species]_Grid_2D_annual	2D version of annually resolved gridded output (no vertical resolution).
[species]_Grid_2D_hour_[#]	2D version of temporally resolved gridded output (no vertical resolution). hour_[#] represents the hour the emissions in the grid started being summed. The emissions in the grid consist of those occurring between that hour and the next hour in tempResolve_HourVec.
[species]_Grid_annual	Gridded output for a particular [species] during the year.
[species]_Grid_hour_[#]	Temporally resolved gridded output for a particular [species]. hour_[#] represents the hour the emissions in the grid started being summed. The emissions in the grid consist of those occurring between that hour and the next hour in tempResolve_HourVec.
[species]EI_cruise	Emission Index for each segment of flight (non-LTO portion only)
AC_AIRPORT_PAIRS	Structure of unique aircraft airport pairs generated from entire OAG schedule. Column 1 is the AEIC aircraft number and the remaining columns contain how many times each flight flies either annually or for each period defined in tempResolve_HourVec.
acMassFlight	Aircraft Mass during each flight segment (non-LTO portion only) (kg). **NOTE: No lateral inefficiency incorporated.**
all[species]_Ineff_annual	Annually resolved sum of particular [species] (both LTO and non-LTO), with lateral inefficiency incorporated.
all[species]_Ineff_hour_[#]	Temporally resolved sum of particular [species] (both LTO and non-LTO), with lateral inefficiency incorporated. hour_[#] represents the hour the emissions started being summed. The emissions in consist of those occurring between that hour and the next hour in tempResolve_HourVec.

altFlight	Altitude of during each flight segment (non-LTO portion only) (feet)
apt_elev_m	Pressure altitude of airports (meters)
apt_nrunways	Number of runways at airport
apt_runway_m	Length of longest runway (meters)
AptNum	List of Airport Names (index in vector corresponds to airport number for AEIC)
azFlight	Azimuth of flight path during each flight segment (non-LTO portion only) (degrees clockwise from north)
fuelBurnFlight	Cumulative fuel burn for each flight segment (non-LTO portion only) (kg). **NOTE: No lateral inefficiency incorporated.**
fuelFactor	Fuel factor for each flight segment for BFFM2 (non-LTO portion only)
fuelFlowFlight	Fuel flow rate during each flight segment (non-LTO portion only) (kg/min). **NOTE: No lateral inefficiency incorporated.**
horDistFlight	Distance along flight path (non-LTO portion only) (NM). **NOTE: No lateral inefficiency incorporated.**
iata2aeitAC	Conversion table for IATA/AEIC Aircraft Codes (AEIC uses numbers for memory)
iata2icaoAC	Conversion table for IATA/ICAO Aircraft Codes
latFlight	Latitude of flight segments (non-LTO portion only) (degrees). **NOTE: No lateral inefficiency incorporated.**
longFlight	Longitude of flight segments (non-LTO portion only) (degrees). **NOTE: No lateral inefficiency incorporated.**
machFlight	Mach Number of flight segments (non-LTO portion only)
Pamb_cruise	Ambient pressure for each segment of flight based on standard ISA (non-LTO portion only) (Pa)
rv_*	Random variable seed for specific model assumption used during Monte Carlo run
Tamb_cruise	Ambient temperature for each segment of flight based on standard ISA (non-LTO portion only) (K)
TAS_Flight	True Airspeed during each flight segment (non-LTO portion only) (knots)
timeFlight	Cumulative time for each flight segment (non-LTO portion only) (sec). **NOTE: No lateral inefficiency incorporated.**

9 Code Structure

This section gives an overview of how scheduling information, aircraft performance data, and airport data are handled in AEIC, in case edits are required for certain studies or functionality.

The script file 'Sequence_World_Func.m' contains the series of scripts that run to execute the model and bring together all of the following information. The underlying scripts/functions called from 'Sequence_World_Func.m' can be easily modified to add functionality or extracted to run a subset of AEIC (i.e. running 'flyFlight.m' to simulate one non-LTO flight with BADA or running 'allocateResultsWithIneff.m' to re-grid previously generated data).

9.1 Scheduling Information

Scheduling information is pre-processed from the Official Airline Guide (OAG). In order to conserve memory in MATLAB, each airport and aircraft supported in AEIC is given a specific number, as opposed to a string. The numbers for the airports are assigned automatically during OAG processing, but the numbers for each aircraft come from a list read in during OAG processing ('Model_Files\aircraftCodes.txt'). The number in this file is not directly utilized for the aircraft; the aircraft number is assigned by order. The IATA codes for the aircraft are required for use with the OAG and the ICAO code for the aircraft is required for BADA.

'convertOAG.m' in 'Model_Files' was developed to convert an OAG schedule and create.mat files to be read in for each airport when the model is run. If the OAG format did not change, 'convertOAG.m' should work to convert a new OAG with no edits and will throw errors for known issues that could come up (non supported aircraft, etc.). During OAG processing, a .mat file for each airport is generated that contains the following information for each individual flight:

- Whether it is an arrival or a departure (ArrDep)
- The destination airport for departures, or originating airport for arrivals (Dest)
- The aircraft being flown (Eqp)
- The hour from midnight, Jan 1st that the flight takes place (hour)

In addition, to decode the numbers assigned to each airport/aircraft, the following keys are created:

- AptNum_[year].mat
- iata2aeicAC_[year].mat

It is important to remember that the above two key files are intrinsically linked with the scheduling information and if they are ever modified without reprocessing the scheduling information, care must be taken to prevent disrupting the aircraft/airport numbers. The OAG conversion is typically only done once per OAG, with the results being saved off. AEIC then uses the resulting .mat files when running.

To save processing time when flying the non-LTO portions of flights, the number of flights for each aircraft between each destination is saved off once it has been calculated. The name of the file that contains this information is 'AC_AIRPORT_PAIRS_[year].mat' in 'Schedule_Files'. AEIC will automatically calculate flight numbers and update 'AC_AIRPORT_PAIRS_[year].mat' if you are analyzing a flight that has not been analyzed before. If changes are made to the scheduling information, this file should be deleted (manually) to ensure it is regenerated with the new schedule.

9.2 Aircraft Performance

Aircraft performance calculations are done differently for LTO and non-LTO portions of the flight. The LTO portion is based on Times-In-Mode (TIMs) at specific power levels using data from the ICAO emission databank, while the non-LTO portion is done using BADA.

9.2.1 LTO

The engine type, number of engines, aircraft class, and other aircraft data is listed in 'EQP_ENG_06May11.txt'. The ICAO emissions databank data for each engine (EI, fuel flow, smoke number, etc.) are included in ENG_EI_UIDs_modeSN.txt. Maximum smoke number and engine pressure ratio information is also in 'UID_PR_SNmax.txt'. Together, these three files contain the information required to analyze the LTO operation of an aircraft. All files are in 'Model_Files'.

9.2.2 Non-LTO

The aircraft performance information from BADA must be added to 'Model_Files'. The only files currently used from BADA are the .PTF files and the .OPF files. Almost all of the .PTF file is used, while only the max payload number from the .OPF file is used.

9.3 Airport Data

AEIC requires the following data for each airport: latitude, longitude, elevation, number of runways, and runway length. All of this information is stored in text format in

‘airports_90el_v3.txt’ under ‘Model_Files’ and in .mat format in ‘AirportData.mat’ under ‘Schedule_Files’. AEIC will use ‘AirportData.mat’ if it exists. If it does not exist, it will read in ‘airports_90el_v3.txt’ and generate ‘AirportData.mat’. If a new airport is added to ‘airports_90el_v3.txt’, ‘AirportData.mat’ must be manually deleted so AEIC will re-process the data.

Appendix A: README File From AEIC v1.0

%%% AVIATION EMISSIONS INVENTORY CODE %%%

v1.0

19th August 2011

README

More information and updated versions available at the MIT Laboratory for Aviation and the Environment website: http://lae.mit.edu

Developed by the MIT Laboratory for Aviation and the Environment and the Cambridge University Energy Efficient Cities Initiative

Note that in early 2013 AEIC v2.0 is scheduled for release. This will enable full-flight emissions calculation, while v1.0 only includes landing and takeoff (LTO) emissions - i.e. emissions below 3000 ft. Freely available full-flight emissions derived are downloadable from http://lae.mit.edu.

---

0. CONTENTS

1. GENERAL DESCRIPTION

The Aviation Emissions Inventory Code is able to estimate emissions arising from the aircraft LTO, auxiliary power units (APU) and ground support equipment (GSE) at a single airport or set of airports. The model is written in Matlab (R2009a) (more recent versions should also be compatible) and run using the Sequence file (distinct files for US or UK). The model can be run in the Matlab GUI (on Linux, Windows) or from the command line on a Linux machine with Matlab using the following command:

```
#> matlab -nosplash -nojvm -r Sequence_US
```

The emissions methodology is outlined in:

Stettler, M.E.J., Eastham, S. and Barrett, S.R.H., 2011. 'Air quality and public health impacts of UK airport. Part I: Emissions'. Atmospheric Environment, 45(31), pp. 5415-5424, doi:10.1016/j.atmosenv.2011.07.012.

The software license is included in the LICENSE file.

2. FILES AND DIRECTORIES

Sequence_UK.m

The main matlab script to run the model for UK airports, with calls to othe sub-scripts and functions. Run this script to run the model for a subset of UK airports.

Sequence_US.m

The main matlab script to run the model for UK airports, with calls to othe sub-scripts and functions. Run this script to run the model for a subset of US airports.

./Model_Files

Contains all the sub-scripts and functions required to run the model.

./Schedule_Files

Contains all 2005 schedules for the top 20 UK airports and top 100 US airports from the OAG in the required format.

./Emissions_Files

This directory is made as a result of running one of the Sequence files. It is where all of the emissions data are stored. Within this directory, different model runs are differentiated by the SCENARIO variable specified in the Sequence file. Within each SCENARIO sub-directory, there are directories for emissions stored in annual and hourly formats.

Note: In the all of the emissions files, fuel burn is reported in kilograms (kg), while all emissions species are reported in grams (g).

./Emissions_Files/SCENARIO/SCENARIO_Total_stats

This file contains summary statistics (mean, median, 5th and 95th percentiles, standard deviation and skew) of the annual emissions estimates summed over all of the airports in the AIRPORT_list, for each emissions species.

./Emissions_Files/SCENARIO/annualEmissions/AIRPORT_SPECIES_annual

These files contain the annual emission estimates for each species and airport resulting from the Monte Carlo model runs in vector format, i.e. each row entry corresponds to one Monte Carlo model run.

./Emissions_Files/SCENARIO/annualEmissions/SCENARIO_AIRPORT_annual_stats

This file contains summary statistics (mean, median, 5th and 95th percentiles, standard deviation and skew) of the annual emissions estimates from a specific AIRPORT.

./Emissions_Files/SCENARIO/hourlyEmissions/AIRPORT_SCENARIO_SPECIES_hourly

These files contain the mean hourly emissions for each airport, each of the 8760 rows corrresponds to an hour of the year 2005, the first row corresponds to 00:00-00:59 on 1st January 2005. The aircraft files (no prefix) contain data in 11 columns. Each column refers to a different mode of the LTO in the following order:

Col 1: Taxi Out

Col 2: Taxi Acceleration (departure)

Col 3: Hold

Col 4: Take Off roll

Col 5: Initial Climb

Col 6: Climb Out

Col 7: Approach

Col 8: Landing roll

Col 9: Reverse thrust

Col 10: Taxi In

Col 11: Taxi Acceleration (arrival)

This separation allows emissions to be spatially distributed as required by the user. To obtain uncertainty ranges (see below) on the hourly emissions, it is important to note that the relationship between the mean and the median, and 5th and 95th percentiles are the same for hourly and annual emissions (as a result of the model design). Thus, hourly uncertainty estimates can be derived from the annual uncertainty estimates.

./Logs

This directory contains the log files which are automatically created by the program. The name of the log files is determined by the SCENARIO name.

3. USER INPUTS AND OPTIONS

The user has several options when running the model, these are set at the beginning of the Sequence file:

SCENARIO = 'scenario_name'

Set the name of the scenario to anything you like. This input is a string.

- AIRPORT QUEUE

AIRPORT_list = {...}

Set the list of airports you want to estimate emissions for, airports are referenced by their three-letter Internation Air Transport Association (IATA) airport code. This input is a cell array of strings.

- UNCERTAINTY SPECIFICATION

mcSwitch = 'on' - run the model with probabalistically such that uncertainty ranges on emissions may be quantified. String input.

mcSwitch = 'off' - run the model deterministically with nominal values from the input uncertainty ranges. String input.

MCruns = 2000

This refers to the size of the Monte Carlon ensemble. If mcSwitch = 'on', it is recommended to input at least 2000 here, smaller ensembles can be used to test the model runs to the end without errors. If mcSwitch = 0, set MCruns = 1. Integer input.

- OUTPUT SPECIFICATIONS

hourlySwitch = 'off' or 'on'

This variable allows the user to determine whether hourly emissions files are output. To reduce RAM and storage requirements this might be preferable. String input.

- PM METHODS

The user can choose to estimated PM (non-volatile and volatile organic) emissions using FOA3 (Wayson et al., 2009) or the new methods proposed in Stettler et al. (2011). String input.

Non-volatile:

pmnvolSwitch = 'foa3' - use FOA3 (Wayson et al., 2009)

pmnvolSwitch = 'new' - use new method (Stettler et al. (2011)

Volatile organics

pmvoloSwitch = 'foa3' - use FOA3 (Wayson et al., 2009)

pmvoloSwitch = 'new' - use new method (Stettler et al. (2011)

Other than the above, the user can omit sections of code or alter the subscripts and functions as desired. The easiest way to do this is to comment out these sections of code by placing a '%' at the beginning of the line.

4. MEMORY REQUIREMENTS

Memory (RAM) usage is significant and depends on the number of airports in the AIRPORT_list and the number of enplanements at each of those airports, it is not there for not linearly related to the number of airports specified. For example, running the model (without hourly emissions outputs) for the top 100 US airports requires ~16GB of RAM, while running it for Boston Logan (BOS) requires 2GB. If you need run the model for numerous airports but have limited RAM, it is suggested that you split the list of airports into smaller groups as required. The large RAM requirements are a result of the model being optimised to run on a server with 32GB.

5. STORAGE SPACE REQUIREMENTS

Hard disk storage space requirements scale linearly with the number of airports and are affected by whether the user desires hourly emissions (see above).

Annual and Hourly emissions: ~25MB per airport

Annual emissions only: ~1.5MB per airport

6. TIME REQUIREMENTS

The time it takes to run the model depends on the number of input airports (and the number of enplanements), the number of MCruns and the processing power of the machine you are using, such that it it might take anywhere between a few minutes and a few days:

1 run, 1 airport: ~ 2 minutes

2000 runs, 1 airport: ~ 1 hour

2000 runs, 20 airports: ~ 2 days

2000 runs, 100 airports: ~ 3 days

7. SCREEN OUTPUT AND LOGS

Some effort has been made to keep the user informed as to the progress of the program. Key input parameters are output to screen and once all inputs have been read in, the message 'Starting monte carlo loop...' will be output, telling the user that the model has begun to calculate emissions. For every 10% of the number of Monte Carlo runs completed, the ensemble number will be output. Once the Monte Carlo loop has finished, the user is informed that the program is 'Saving output...'.

These screen output is automatically saved to a log file in the 'Logs' directory, with the same name as SCENARIO. Note, if you run the program twice with the same SCENARIO name, the previous log file will be delted and replaced.