NASA/TM–2016–219342

# ANOPP2 User's Manual

## Version 1.2

*L. V. Lopes*
*Langley Research Center, Hampton, Virginia 23681-2199*

*C. L. Burley*
*Langley Research Center, Hampton, Virginia 23681-2199*

October 2016

NASA/TM–2016–219342

# ANOPP2 User's Manual

## Version 1.2

*L. V. Lopes*
*Langley Research Center, Hampton, Virginia 23681-2199*

*C. L. Burley*
*Langley Research Center, Hampton, Virginia 23681-2199*

National Aeronautics and
Space Administration

Langley Research Center

October 2016

# Acknowledgments

This work is funded by NASA's Aeronautics Research Mission Directorate (ARMD), specifically the Transformative Tools and Technologies (TTT) Project in the Transformative Aeronautics Concepts Program (TACP), the Advanded Air Transport Technology (AATT), Commercial Supersonic Technology (CST), and Revolutionary Vertical Lift Technology (RVLT) Projects in the Advanced Air Vehicles Program (AAVP), and the Environmentally Responsibly Aviation (ERA) Project in the Integrated Aviation Systems Program (IASP).

The use of trademarks or names of manufacturers in this report is for accurate reporting and does not constitute an offical endorsement, either expressed or implied, of such products or manufacturers by the National Aeronautics and Space Administration.

Available from:

NASA STI Program / Mail Stop 148
NASA Langley Research Center
Hampton, VA 23681-2199
Fax: 757-864-6500

# Abstract

This manual documents the Aircraft NOise Prediction Program 2 (ANOPP2). ANOPP2 is a toolkit that includes a framework, noise prediction methods, and peripheral software to aid a user in predicting and understanding aircraft noise. This manual includes an explanation of the overall design and structure of ANOPP2, including a brief introduction to aircraft noise prediction and the ANOPP2 background, philosophy, and architecture. The concept of nested acoustic data surfaces and its application to a mixed-fidelity noise prediction are presented. The structure and usage of ANOPP2, which includes the communication between the user, the ANOPP2 framework, and noise prediction methods, are presented for two scenarios: wind-tunnel and flight. These scenarios serve to provide the user with guidance and documentation references for performing a noise prediction using ANOPP2.

# Contents

# Introduction

The goal of the Aircraft NOise Prediction Program 2 (ANOPP2) is to provide the U.S. Government with the ability to (1) predict aircraft system noise; (2) predict aircraft component noise; and (3) assess aircraft noise reduction technologies and low noise flight procedures [1]. This includes analysis capabilities to process results for the purpose of understanding noise generation mechanisms in support of potential experiments and flight demonstration activities.

The ANOPP2 toolkit:

- Provides a framework in which methods can be developed and used to predict the noise from arbitrary aircraft designs (full-scale and model-scale size), including unmanned aerial vehicles (UAVs) and personal aerial vehicles (PAVs).
- Provides a framework where a combination of acoustic prediction methods of differing fidelity can communicate in a unified system. This ranges from fast computations with reduced order models for system level assessment and design to computationally intensive, high-granularity, physics-based methods for investigating and understanding noise generation and propagation effects at the component and aircraft system level.
- Provides capabilities to allow prediction comparison to flight and model-scale test data. This includes methods to predict the influence of the flight test environment, such as Doppler effects, convective amplification, and atmospheric propagation.
- Includes a framework where physics-based prediction of propulsion airframe aeroacoustics (PAA) [2] and airframe interaction noise, such as scattering of undercarriage noise by the airframe and propulsion interactions like those seen in closely spaced twin jets or open-rotors, can be integrated into a unified system level environment.
- Provides capabilities to aid in assessing noise reduction technologies and approaches. This includes tools for sensitivity analysis and gradient-based optimizations.
- Contains interoperability functionality to enable integration with analysis software such as the FAA's Environmental Design Space (EDS) and NASA's OpenMDAO and GEN2 toolsets [3–5].

This document is organized into the following chapters.

1. Chapter 1 presents (1) background material that explains the challenges of noise prediction of modern and unconventional aircraft, (2) NASA's desire to address those challenges, and (3) a brief history of the Aircraft NOise Prediction Program (ANOPP) created in the 1970s.
2. Chapter 2 explains how ANOPP2's philosophy of mixed-fidelity noise prediction is capable of addressing the challenges of aircraft noise prediction. The concept of nested Ffowcs Williams and Hawkings (FW-H) surfaces is explained, and it is shown how those surfaces can be utilized to combine methods of different fidelity and allow

for the prediction of noise from aircraft in flight or scale models placed in a wind tunnel environment.

3. Chapter 3 describes the implementation of the philosophy by way of ANOPP2's architecture. The components of ANOPP2 are presented along with a brief description of the reference documentation for the ANOPP2 system.
4. Chapter 4 presents the steps a user would take to interface with the architecture of ANOPP2 in order to perform a noise prediction. This includes basic operations such as an introduction to an application programming interface (API), using tags to access information within ANOPP2, error handling, versioning, where to find more information on getting started, and how to obtain ANOPP2.
5. Chapter 5 outlines two examples (wind tunnel and flight) that act as a guide through the steps of creating a prediction using the ANOPP2 paradigm.

# 1 Background

Community noise has been an ongoing problem for aircraft and is projected to be a major concern in the future [6]. The NASA Aeronautics Research Mission Directorate (ARMD) has historically invested in developing technologies that can reduce noise from current and future aircraft configurations [7–9]. Both conventional and unconventional aircraft designs continue to be evaluated [10], and assessments are performed using aircraft noise prediction capabilities as well as measured data. The accuracy of the assessments, particularly for unconventional aircraft, for which measurement data or validated analyses are often unavailable, rely solely on the accurary of the prediction techniques and the expertise of the person performing the analysis. Hence, any comprehensive aircraft noise prediction method must be applicable to new designs and allow for prediction outside of the current experience base.

Predicting aircraft system and component noise, particularly for modern and unconventional aircraft, is challenging because:

- There are many different sources of noise on a modern aircraft. The noise sources for fixed wing aircraft are typically separated into two categories: engine and airframe. Engine sources include those caused by the propulsion system, such as inlet and exit fan, combustor, and jet mixing noise. Airframe sources include those caused by the aircraft body, such as landing gear, flap, slat, and trailing edge. Similary, for rotary wing aircraft, the noise can be separated into engine and rotor noise.

- The noise may be scattered and/or shielded by the aircraft body and may include flight effects that are difficult to recreate in a controlled environment.

- Installation of aircraft components may give rise to additional noise sources. This includes PAA, such as the additional noise source from the jet exhaust impinging on an aircraft's body (referred to as scrubbing noise), or airframe interaction, such as the interaction of the high lift system and the landing gear.

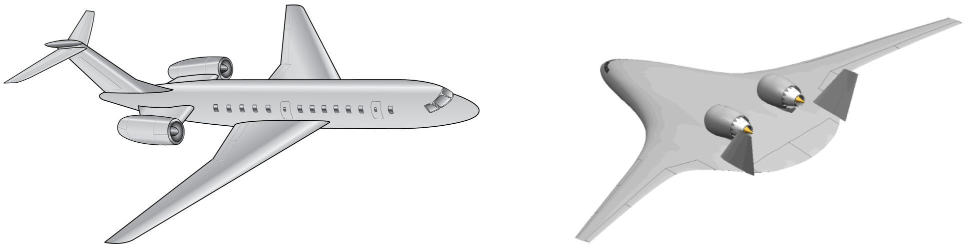
- The atmospheric and ground conditions influence the propagation of noise from the aircraft to the observer on the ground.

- The information required to predict the noise is often incomplete and in many different formats and coordinate systems.

- The perception of noise at the observer is subject to the listener's ear and includes factors that estimate community annoyance.

NASA initiated and has continued development of ANOPP [11, 12] since the late 1970s to provide the U.S. Government with the ability to predict aircraft noise. Many of the prediction methodologies implemented within ANOPP are predominantly physics-based formulations based on the acoustic analogy or semi-empirical models. They have been and continue to be developed and updated using the best available experimental data sets. Many of the prediction methods work well for conventional aircraft configurations but may lack capability and fidelity required for design changes and/or non-conventional configurations without some modification to the model formulation or calibration.

Figure 1(a) shows a conventional tube-and-wing aircraft configuration with rear mounted engines. A tube-and-wing design typically includes noise from the engines and airframe components such as the landing gear and flaps. These noise sources are not the same in isolation as when installed. For example, the noise from landing gear is directly related to the local flow field, which is affected by the free-stream flow velocity and the aircraft's high-lift system settings for an installed main gear [13–15]. This flow-field can be significantly different from the uniform freestream flow velocity. Similarly, the engine noise sources can be reflected by the wing for under-the-wing configurations, or partly shielded for over-the-wing configurations. These effects, as well as other source noise phenomena, are more evident for unconventional configurations as shown in Figure 1(b).



(a) Conventional tube-and-wing aircraft design.

(b) Unconventional Hybrid Wing Body (HWB) concept.

Figure 1. Conventional and unconventional aircraft designs.

The requirement to not only predict system noise for arbitrary aircraft designs but to also advance understanding of noise physics involves higher fidelity capabilities beyond what has been available within NASA (ANOPP) or in the public domain. The NASA Programs continue multidisciplinary efforts that cultivate new technologies and mature existing technologies from conceptual design to the current airspace system. This includes bringing new computational capabilities, such as computer aided design techniques, computational fluid dynamics (CFD), and gradient-based optimization frameworks such as Phoenix Integration's Model Center® [16] or NASA's OpenMDAO [4,17] to the forefront of the aircraft design process. ANOPP2 enables the noise prediction process to fully account for aircraft design details by offering a framework that can accommodate a full range of computational methods, from CFD to those embodied in ANOPP, and measurement data, from model to full-scale tests, in a unified prediction process.

# 2 Philosophy

The challenges of aircraft noise prediction are addressed by ANOPP2 by providing the capability and a framework to integrate acoustic data and prediction methodologies of differing fidelity for source noise component prediction, installation effects, propagation to the far-field, and the influence on the observer. The framework of ANOPP2 allows for prediction methods that include the fidelity and flexibility required to predict outside any experience base that currently exists. A focal point of ANOPP2 is the ability to combine acoustic approaches of differing fidelity and methodology in a single, unified system; that is, to offer several options for prediction of a specific noise mechanism depending on requested fidelity and execution speed. This allows ANOPP2 to include fast methods for design optimization while including methods that contain the fidelity required for understanding and provide insight into controlling noise physics. By allowing for multiple methods for the same noise source, ANOPP2 can provide direct comparison between methods in the same framework, which allows for greater understanding of noise generation and propagation.

## 2.1 Mixed-Fidelity Noise Prediction

Fidelity is the accuracy of the representation when compared to the real world and can be characterized by error, precision, sensitivity, resolution (granularity), tolerance, and validity [18]. The set of values associated with each of these characteristics is referred to as a "fidelity set." Having a mixture of methods with different fidelity, that is methods that may vary significantly in computation time and granularity, allow for a user to 'dial-in' fidelity where necessary and compare methods of differing fidelity in order to understand the acoustics and propose potential noise reduction technologies. For this reason, ANOPP2 is not strictly high-fidelity, nor is ANOPP necessarily low-fidelity. However, due to the constraints at the time of development and the availability of prediction methods, the fidelity set that ANOPP can explore is limited; whereas the design of ANOPP2 can accommodate methods of differing fidelity simultaneously, and can allow for a mixture of fidelity to communicate in a common framework, herein called mixed-fidelity noise prediction.

ANOPP2 offers capabilities to integrate a number of acoustic approaches of varying fidelity into a unified system. Figure 2 shows four stages for an aircraft noise prediction: source noise generation, installation effects, propagation (effects of atmosphere and terrain), and the observer's perception. Each of the four stages of a single prediction may be of differing fidelity and the full aircraft prediction may contain many noise generating mechanisms. The challenge is weaving together a mixture of methods of varying fidelity into a unified system.

## 2.2 Nested Acoustic Data Surfaces of Differing Fidelity

ANOPP2 addresses this challenge through the use of nested acoustic data surfaces. Within ANOPP2, all the acoustic information is stored on an "acoustic data surface (ADS)," which easily and efficiently allows communication and transfer of acoustic data between methods

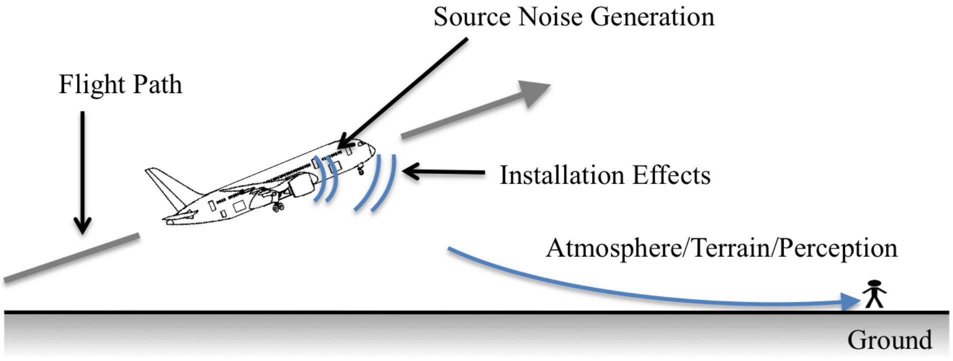


Figure 2. The capabilities of ANOPP2 include: 1) engine and airframe source noise prediction such as inlet noise (shown), 2) installation effects (PAA and airframe interactions) such as scattering and shielding, 3) propagation from near the aircraft towards the observer on the ground including the effects of the atmosphere, terrain, and 4) observer perception.

of different fidelity. The ADS can accommodate acoustic data in almost any form, ranging from acoustic pressure, density, and velocity time histories to spectra and integrated metrics such as effective perceived noise level (EPNL). The highest fidelity data surface available in ANOPP2 is the time-accurate FW-H surface [19] that contains pressure, density, and velocity. The pressure, density, and velocity on a FW-H surface completely defines all noise sources contained inside to any observer outside the surface. The FW-H technique has been successfully used to predict noise by coupling a CFD solution to a FW-H solver to provide the noise at an observer location. [20–23] As long as nonlinear effects, such as shocks that cause high speed impulsive (HSI) noise in helicopters [24], and volume sources, such as turbulence in jet exhausts [25], are inside the surface, the conditions on the surface fully define the contribution from all sources inside the surface to observers outside the surface. By using the conditions on the acoustic data surface as input into a FW-H solver, the computation is reduced to a manageable surface integration without direct knowledge of the complicated sound source inside. Within ANOPP2, pressure data surfaces are used for low-granularity methods, such as some methods within ANOPP, and FW-H surfaces are used with higher-granularity methods, such as those based on CFD. It is also worth noting that pressure data surfaces, such as $1/3$-octave band sound pressure level (SPL) spectra and time-accurate FW-H surfaces are not the only available types of acoustic information that can be cast on an ADS within ANOPP2. However, they do represent the extremes of granularity. ANOPP2 is capable of handling a number of acoustic metrics within this range. It is also worth noting that the ADS technique may also be applied to measurement data.

Many different ADS can be used in a single aircraft noise prediction. Figure 3 schematically shows the nested ADS approach implemented in ANOPP2. Several acoustic “source-surfaces” are shown around aircraft noise components, which are nested within the acoustic “mid-surface” that surrounds the aircraft. The source- and mid-surfaces are not restricted to spheres as shown in Fig. 3, but can be arbitrarily shaped, allowing for inclusion of distributed sources such as those associated with the jet exhaust. The source-surface encapsulates the component source noise prediction, regardless of the acoustic approach. There can be several source-surfaces, one for each unique source or subset of sources. The source-surface is provided as input to the method that calculates the noise on the mid-surface. This method can include the fidelity to account for PAA and airframe interaction [26–29]. The noise stored on the mid-surfaces, one for each unique source-surface, can be summed or kept separate. If the mid-surface is a penetrable FW-H surface, several mid-surfaces can be added together and still retain the constructive/destructive interference and directivity of the encapsulated acoustic sources. The "far-surface" is in the acoustic far-field, coinciding with the observer location(s) of interest. Once the noise is defined on the far-surface, it is then summed and the result is the total noise. It should be noted that often source noise methods provide only acoustic pressure and do not provide the acoustic velocity, which is required by the FW-H surface. The ANOPP2 framework allows for the storage of acoustic velocity and also for incorporation of methods that can determine the acoustic velocity.

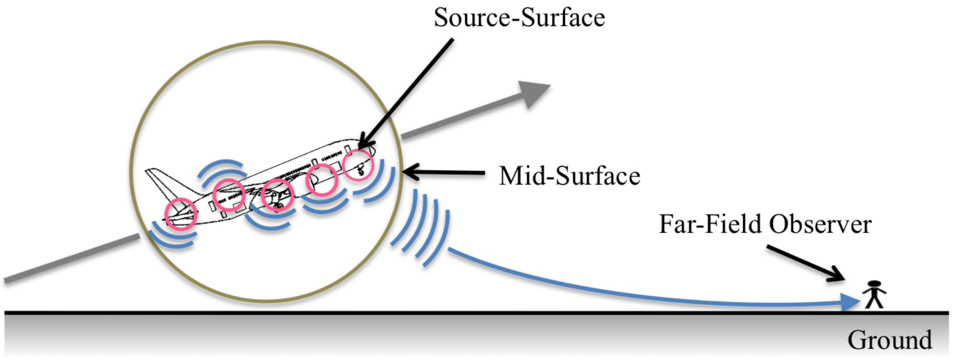


Figure 3. Representation of the acoustic data surfaces in the ANOPP2 prediction system. The source-surface encapsulates the source or subset of sources; mid-surface encapsulates the aircraft; and the far-surface is at the observer on the ground.

Although Fig. 3 shows each noise source wrapped in a source-surface, all source-surfaces wrapped in a mid-surface, and an observer placed at the far-surface, ANOPP2 is capable of handling methods that do not necessarily follow this paradigm. For instance, methods that predict only on the mid-surface can be combined with methods that predict only at the far-surface, shown in Fig. 4. In this configuration, the acoustic sources on the aircraft are predicted directly on the mid-surface, without the intermediary step of the source-surface. This is analogous to how ANOPP is often used to predict 1/3-octave band SPL spectra on a psuedo-observer sphere surrounding the aircraft. Similarly, methods may predict directly on the far-surface; the flexiblity of ANOPP2's framework allows results from different prediction methods to be combined in a consistent process.

The philospohy of nested ADS also lends itself to utilizing the ANOPP2 framework to predict noise in a manner that allows for direct comparison to wind tunnel acoustic measurements. A wind tunnel environment is simulated by excluding the portion of the computation that accounts for the propagation to the far-surface. This results in a computation that includes either a mid-surface prediction method or one that combines a source-surface prediction method and mid-surface propagation method. As a result, the noise is calculated at points that coincide with measurement locations. This type of configuration is shown in Fig. 5.

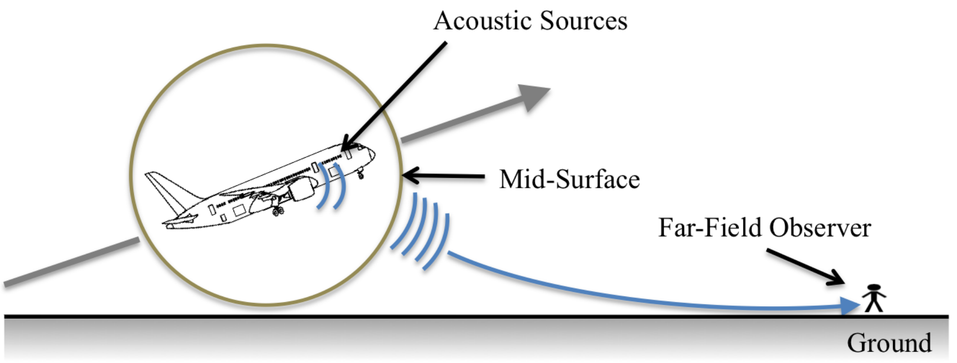


Figure 4. Representation of a prediction using ANOPP2 with methods capable of predicting the noise at the mid-surface, centered at the aircraft center of gravity. These predictions are then coupled with a method capable of propagating the noise to the observer, computing time, frequency, and spatially integrated metrics.

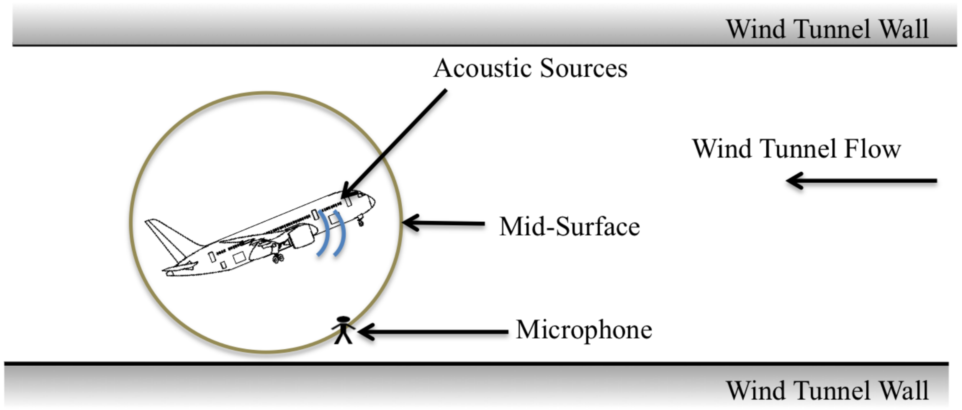
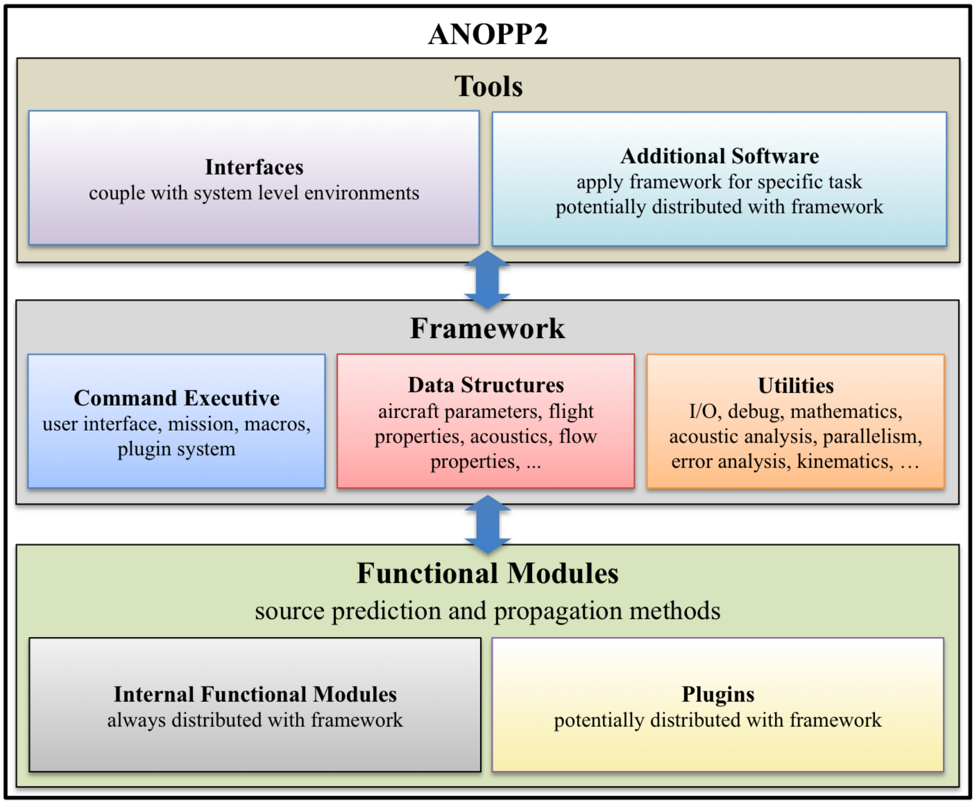


Figure 5. Representation of an acoustic prediction using ANOPP2 that facilitates comparison to wind tunnel measurements. This representation shows a prediction at the mid-surface which coincides with the microphone location. Another option is to utilize a method capable of predicting at the source-surface and a method that can predict on the mid-surface, taking into acount installation effects such as scattering.

# 3 Architecture

Aircraft noise prediction requires information to model the physics of the noise generator and prediction algorithms to predict the noise utilizing that information. The information required includes geometrical and flow quantities, flight trajectories, atmospheric conditions, ground properties, and observer locations. The type of algorithm, and thus the information required, depends on the application. An architecture handles the selection of acoustic algorithms and communication of information to those algorithms.

ANOPP2 is a toolkit, that is a set of tools which a user can employ to perform a noise prediction. The ANOPP2 toolkit separates the handling of information, such as the storage of acoustic information, and the generation of information, such as a noise calculation, into two distinct portions: "framework" and "functional modules," as shown in Fig. 6. The functional modules perform acoustic calculations while the framework provides the functionality that creates, organizes, communicates/handles input/output data, and executes the functional modules. In addition to the framework and functional modules, "tools" include interoperability capability for coupling with system level environments and any additional software that utilizes the ANOPP2 framework for specific tasks. Tools may or may not be distributed with the ANOPP2 framework. Tools that are not distributed with the ANOPP2 framework may still be aquired; however, a separate software usage agreement is required.



ANOPP2

Tools

**Interfaces**
couple with system level environments

**Additional Software**
apply framework for specific task
potentially distributed with framework

Framework

**Command Executive**
user interface, mission, macros, plugin system

**Data Structures**
aircraft parameters, flight properties, acoustics, flow properties, ...

**Utilities**
I/O, debug, mathematics, acoustic analysis, parallelism, error analysis, kinematics, …

Functional Modules
source prediction and propagation methods

**Internal Functional Modules**
always distributed with framework

**Plugins**
potentially distributed with framework

Figure 6. Framework, functional modules, and tools of the ANOPP2 toolkit.

## 3.1 Framework

The framework is composed of three components: the "command executive," "data structures," and "utilities." The framework of ANOPP2 includes capability to organize information (internal and/or external to ANOPP2), receive/interpret intructions from the user, and perform computations such as the calculation of derived noise metrics, kinematics, mathematical operations, and error catching. The command executive provides the communication between the user and the functional modules. A user of ANOPP2 wishing to perform a noise prediction interfaces with the command executive to organize the functional modules to be executed, the order they are executed, and the inputs the functional modules require. The communication of information between functional modules occurs via the data structures, which define the inputs and outputs of the functional modules. The information includes the atmosphere, flight path, propulsion system, etc. The data structure that records noise is called the "observer data structure". The observer data structure stores the noise defined at a set of locations, such as a certification microphone or source-surface, mid-surface, and far-field observer surfaces. The utilities provide common operations and processing tools to assist the command executive and data structures, as well as the user. The capabilities provided by the utilities include creation of unique tags, error handing, common mathematical operations, calculation of acoustic metrics, and kinematics.

## 3.2 Functional Modules

An "acoustic code" (also called a "prediction method") is an implementation of a specific acoustic prediction algorithm for which the user invokes for a noise computation. A functional module is an interface between an acoustic code and the ANOPP2 framework. The acoustic code may be an integrated component of the functional module, such as code that is compiled together into one binary, or a separate piece of software, such as an executable that is executed by the functional module. There are two types of functional modules: "internal functional modules" and "plugins." An internal functional module is embedded in the ANOPP2 framework and is indistiguishable from the framework. Examples include the functional modules that interface with ANOPP, the formulation functional modules (such as Farassat's Formulation 1A) [30], the functional module that allows for interfacing with a Network Common Data Form (NetCDF) database [31], and the functional module that can account for wind tunnel and flight effects. In addition to the suite of internal functional modules, the command executive of the ANOPP2 framework includes a plugin system which allows for user-generated functional modules, called plugins. Plugins are external functional modules (i.e., not embedded in the ANOPP2 framework) that are loaded at run time and can be written by a user of ANOPP2 or provided by the ANOPP2 development team. Certain plugins written by the ANOPP2 development team are distributed with the framework; however, plugins that are not distributed with the ANOPP2 framework may be aquired through separate software usage agreements. The command executive provides inputs (including information contained within data structures), as directed by the user, to the functional module. The functional module processes and provides the information to the acoustic code. The acoustic code performs the acoustic calculation and returns the noise results back to the functional module which, in turn, stores the result in the observer data structure. The user can then access the results by interfacing with the observer data structure or provide them to another functional module for additional calculations. Communications between the user, framework, functional module, and acoustic codes are shown in Figure 7.

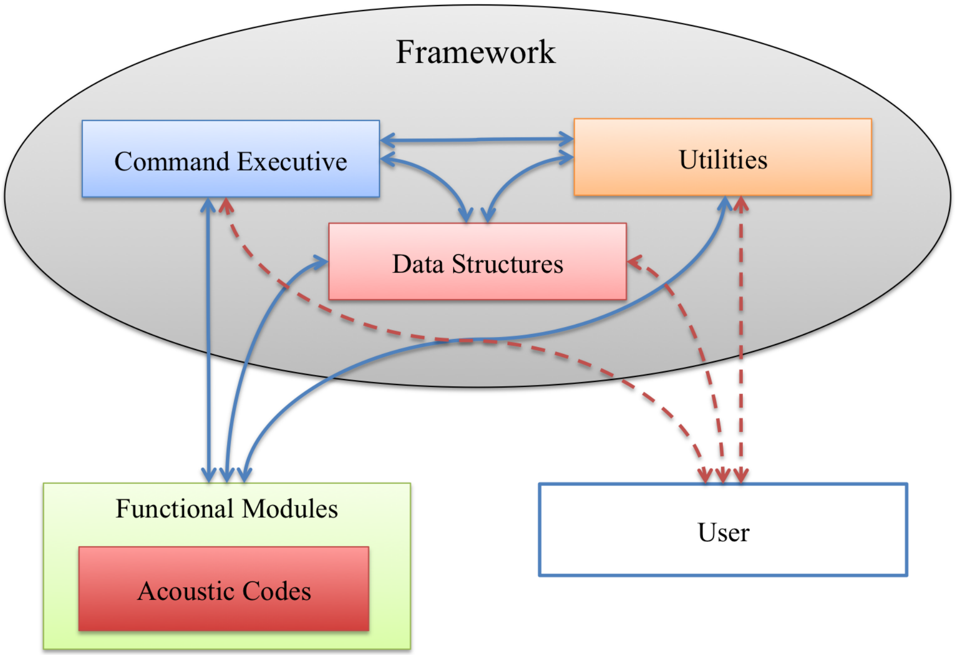


Figure 7. Communications between user, framework, functional modules, and acoustic codes of ANOPP2.

## 3.3 Components

Below is an itemized list of components of the ANOPP2 toolkit. The components are separated into five main categories: command executive, functional modules, data structures, utilities, and tools. Listed with each component is a reference to a manual or manuals, distributed with ANOPP2, that the user can access to get more information about that specific component.

- Command Executive

  The command executive is the highest level of functionality in the ANOPP2 framework. It is responsible for communication between the user and the functional modules, organizing the functional modules into a sequence of computational events, and interfacing with plugins.

  - Interface
    The interface includes functionality that facilitates communication of information between the user and the functional modules. See *CommandExecutiveApiReferenceManual.pdf* for more information [32].

  - Mission
    A “mission” is a collection of functional modules that execute in sequence at designated source times and cooresponding flight conditions. At the end of the sequence, the functional modules housed within a mission will have performed a noise computation and provided that noise data to the data structures within ANOPP2 for access by the user. See *CommandExecutiveApiReferenceManual.pdf* for more information [32].

  - Plugin System
    Functionality that allows for the coupling of ANOPP2 with external functional modules, called plugins. A plugin is a user created functional module that needs to be compiled by the user using very specific functional calls that can be recognized by the plugin system of the command executive. See *PluginSystemManual.pdf* for more information [33].

- Functional Modules

  Functional modules are responsible for the communication between the command executive and the acoustic codes. This includes accessing data structures to aquire information needed by the acoustic code and receiving the acoustic metrics from the acoustic codes and casting them onto the observer data structure for access by the user.

  - Internal Functional Modules
    Internal functional modules are methods that are embedded in the command executive and handle the input/output required for or provided by the acoustic codes. All functional modules require specific input in terms of populated data structures and configuration files. Configuration files are Fortran namelists that contain critical information for the functional modules, including settings and input parameters. See *FunctionalModulesManual.pdf* for more information [34].

  - Plugins
    Plugins are functional modules that are loaded at run time and may be generated and provided by the ANOPP2 development team or written by a user. Plugins are loaded by the command executive’s plugin system and must include very specific functional calls. A unique manual accompanies all plugins provided by ANOPP2.

- Data Structures

  Data structures contain information that defines the flight path, propulsion system, geometrical structure, atmospheric conditions, and acoustic information. All data structure APIs contain access routines, that is, routines that allow for the setting and retrieving of information contained within the data structure. The reference manuals of all data structures include details on the access routines available and what information the user can access.

  - Observer
    The observer data structure provides functionality for organizing, storing, read-

ing/writing, calculating, and modifying noise metrics at a set of aircraft noise prediction and/or measurement locations. This includes absolute levels as well as change in levels (or suppressions) such as Δ 1/3-octave band SPL spectra. Locations can be grouped into shapes, such as a surface (i.e., the source- and mid-surface), or line, such as a line of observer locations on the ground during an aircraft flyover. See *ObserverApiReferenceManual.pdf* for more information [35].

  - Flight Path
    The flight path data structure provides functionality for creating and accessing flight and aircraft properties such as those that would be defined by FLight OPtimization Software (FLOPS) [36]. This includes engine throttle, angle of attack, Mach number, landing gear settings, and aircraft location during flight. See *FlightPathApiReferenceManual.pdf* for more information [37].

  - Atmosphere
    The atmosphere data structure provides the atmospheric properties as a function of space and time. Atmospheric properties include ambient pressure and temperature as well as relative humidity. See *AtmosphereApiReferenceManual.pdf* for more information [38].

  - Propulsion
    The propulsion data structure provides an interface into the propulsion system of the aircraft including performance and geometric properties such as those that would be defined by Numerical Propulsion System Simulation (NPSS) [39,40]. This includes performance properties such as the pressure and temperature at the stages of a turbofan engine as well as geometry properties of the engine, such as the jet exit diameter. See *PropulsionApiReferenceManual.pdf* for more information [41].

  - Geometry
    The geometry data structure defines geometrical structures such as node-based or cell-based surfaces, lines, and volumes that may be deforming in space and time. Also included are point cloud type geometries and the ability to couple with fluid dynamic solutions such as those provided by The NASA OVERFLOW and Fun3D codes [42, 43]. See *GeometryApiReferenceManual.pdf* for more information [44].

- Utilities

  Utilities are groups of common functionality that can be utilized to perform a certain task, such as the calculation of derived acoustic metrics.

  - Acoustic Analysis
    The acoustic analysis utility provides constants, enumerations, and routines for calculating common noise metrics for aircraft noise prediction and measurement. See *AcousticAnalysisApiReferenceManual.pdf* for more information [45].

  - Kinematics

    The kinematics utility provides routines for calculating the kinematics of a point undergoing frame of reference changes, such as the motion of a point on the wing of an aircraft which is undergoing flight. This includes velocity, acceleration, jerk ($3^{rd}$), and snap ($4^{th}$ time derivative) of the point with respect to the ground frame. See *KinematicsApiReferenceManual.pdf* for more information [46].

  - Math

    The math utility provides routines for common mathematical functions for acoustics. See *MathApiReferenceManual.pdf* for more information [47].

  - Tags

    The tags utility provides a system of generating unique numbers within groups. See *TagApiReferenceManual.pdf* for more information [48].

  - Error

    The error utility provides an error handling system. See *ErrorApiReferenceManual.pdf* for more information [49].

- Tools

  Any additional components of ANOPP2 are called "tools." Tools are components, which may or may not be distributed with ANOPP2, and perform a specific functionality using the ANOPP2 framework. Tools are categorized into two components, "interfaces" and "additional software." Interfaces couple the ANOPP2 framework with system level environments such as Model Center (called the ANOPP2 Model Center Interface Code (AMCIC)) [50] and OpenMDAO (called the ANOPP2 OpenMDAO Interface Code (AOIC)). Additional software is any application that uses the ANOPP2 framework to perform a specific task, such as rotorcraft noise prediction, coupling with specific databases, noise manipulations, etc. Each tool distributed with the ANOPP2 framework is accompanied by its own set of manuals.

- Other

  In addition to the above list of manuals, ANOPP2 is also distributed with a Fortran coding standard to which the framework adheres [51]. This standard is offered to the user as guidance, if desired, to the Fortran code embedded in ANOPP2.

  Finally, also provided is the Quick Start Guide to aid a user who is familiar with ANOPP in getting started with ANOPP2 [52].

### 3.4 Documentation

A list and summary description of the ANOPP2 manuals are provided in Table 1.

Table 1. Manuals in the ANOPP2 toolkit. Not shown are the manuals that may come with the tools or plugins of ANOPP2.

| **ANOPP2 Manual** | **Description** |
|---|---|
| User's Manual [this document] | Introduction into ANOPP2 and summary of available components. |
| Command Executive API Reference Manual [32] | Executing functional modules (internal and plugins). |
| Functional Modules Manual [34] | Using internal functional modules including input and output. |
| Plugin System Manual [33] | Creating plugins and using the plugin system. |
| Observer API Reference Manual [35] | Storing ADS and manipulating acoustic metrics at the observer locations. |
| Flight Path API Reference Manual [37] | Aircraft position, orientation, and properties such as throttle setting and angle of attack. |
| Atmosphere API Reference Manual [38] | Storing and accessing ambient atmospheric conditions. |
| Propulsion API Reference Manual [41] | Storing and accessing propulsion system specifications (geometry and state). |
| Geometry API Reference Manual [44] | Interfacing with CFD and creating simple geometrical structures. |
| Acoustic Analysis API Reference Manual [45] | Calculating acoustic metrics such as EPNL. |
| Kinematics API Reference Manual [46] | Relative motion between frames of references including realitive position, velocity, acceleration, jerk, and snap. |
| Math API Reference Manual [47] | Common mathematical operations such as interpolation and integration. |
| Tag API Reference Manual [48] | Unique tag identifiers. |
| Error API Reference Manual [49] | Error reporting and debugging. |
| Quick Start Guide [52] | Several ANOPP2 examples of increasing complexity for users familiar with ANOPP. |
| Fortran Coding Standard [51] | Suggested rules for writing code in Fortran 90 and greater. |

# 4 Usage

A user of ANOPP2 is ultimately interested in a noise prediction utilizing one or more acoustic codes housed in functional modules. An acoustic code (and thus a functional module) requires specific data depending on the noise source and the fidelity of the computation. For example, an empirically based computation of airframe noise may only require some gross aircraft geometric properties; however, a CFD-based computation of the same noise source may require a solution of the flow properties surrounding the entire noise generating region. The functional module provides the information to the acoustic code, this includes information stored in the data structures or provided by the user directly to the functional module. Once all the information has been stored in data structures and the functional modules have been selected, the mission can be executed. When the mission is executed, the functional modules specified by the user are executed, which populate the data structures with noise predictions. Afterwards, the observer data structure stores the noise predictions from the functional modules. The user can access the data stored in the observer data structure or provide the observer data structure to another functional module for additional computations. The process of performing a noise prediction using ANOPP2, including providing and accessing information, is shown schematically in Fig. 8.

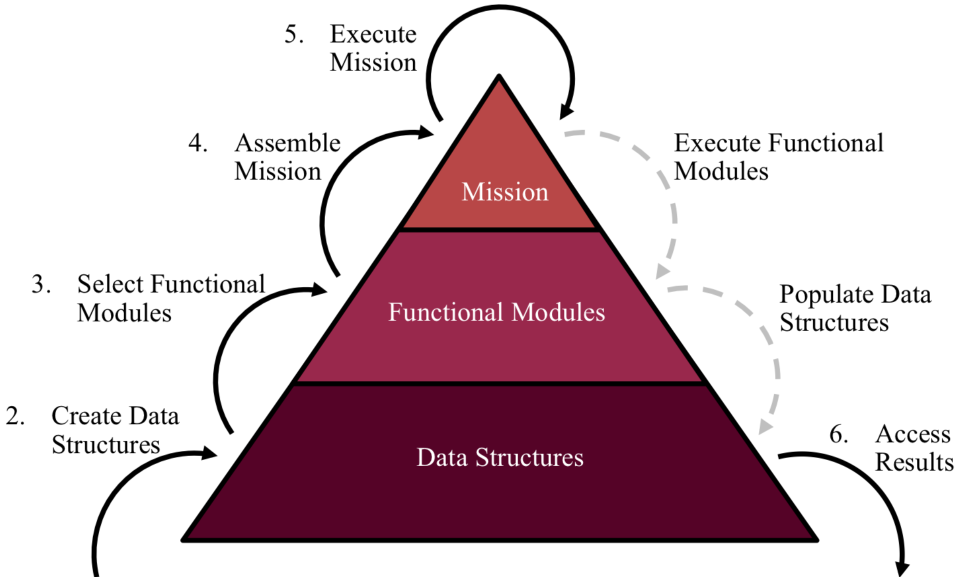


Figure 8. The hierarchy of structures within ANOPP2 and process of performing a noise prediction. Solid lines are explicity executed by the user, dashed lines occur automatically within the framework. Initializing ANOPP2 not shown.

User written code contains the following steps.

1. Initialize ANOPP2 (makes available the ANOPP2 framework to the user code).
2. Provide the required information and create the data structures.
3. Select the functional modules and provide the data structures required for input and

output.

4. Assemble the functional modules into the mission.
5. Execute the mission (and thus execute the functional modules which populate data structures).
6. Access the noise results and/or calculate derived metrics.

## 4.1 Application Programming Interface (API)

The ANOPP2 framework, as described in Section 3.1, is composed of the command executive, data structures, and utilities. Along with the internal functional modules, these are contained in a single library, called *libANOPP2*, shown in Fig. 9. This represents a significant shift in philosophy from ANOPP which relied upon a single input file into a single executable. To use ANOPP2, a user is required to write a Fortran, C++, or Python program, called a "user code," that utilizes the functionality defined in the ANOPP2 library. The functionality comes in the form of constants, enumerators, and routines that are housed in an API, provided along with the ANOPP2 library. Note that the user code is different from an acoustic code; the acoustic code is housed within a functional module and performs a noise computation. A user code executes one or several functional modules, and thus acoustic codes, to perform an aircraft noise computation. The user writes a user code and then compiles and links with the ANOPP2 library. By creating a sequence of actions made available by the ANOPP2 library, the user is able to perform a noise prediction. This is schematically shown in Fig. 10. All reference manuals provided with ANOPP2 contain a chapter on installing ANOPP2, creating a user code, and linking the ANOPP2 library with the compiled user code [32, 35, 37, 38, 41, 44–49].

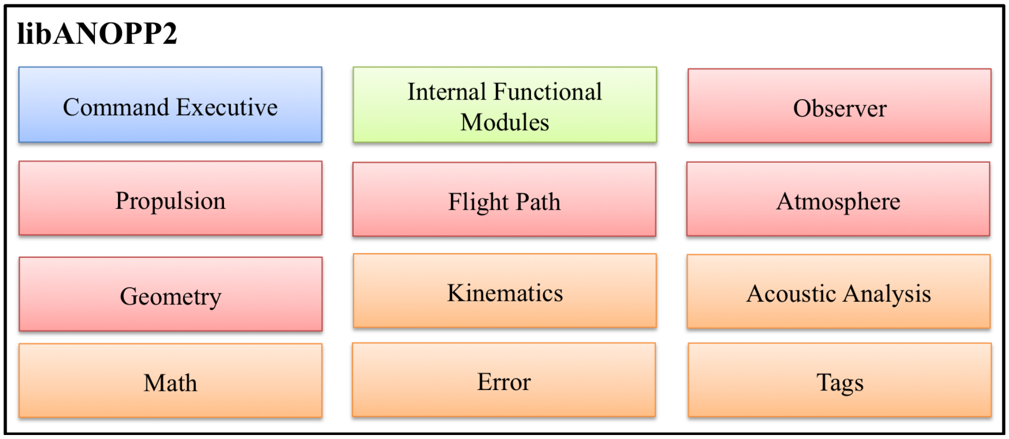


Figure 9. ANOPP2 library includes the command executive, internal functional modules, and all supported data structures and utilities. The command executive is shown in blue, internal functional modules in green, data structures in red, and utilities in orange.

The ANOPP2 library contains all components of the ANOPP2 framework, including all components of the command executive, all data structures, and all utilities. Also provided

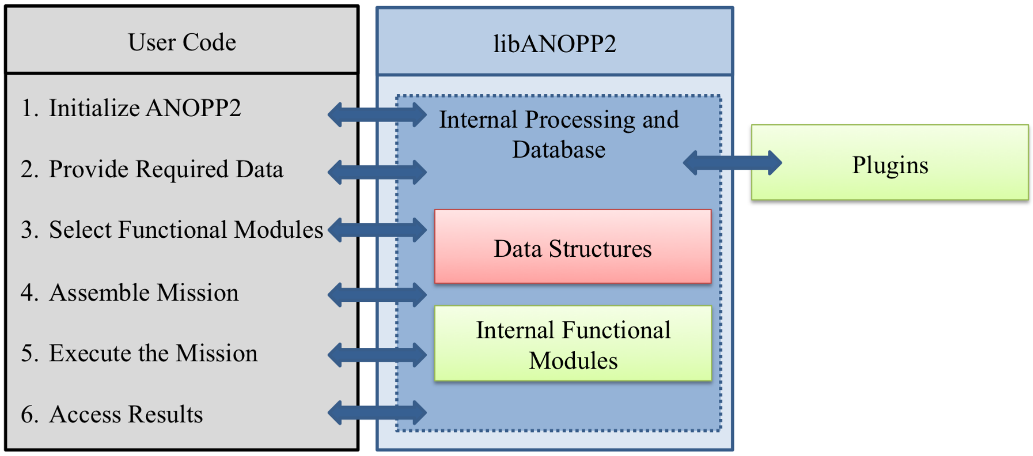


Figure 10. Schematic of interaction between user's user code and the ANOPP2 library.

with ANOPP2 are libraries and interfaces that contain a single data structure or utility. This allows a user who is interested in only a certain piece of ANOPP2 to link only to parts in which the user is interested. For example, the acoustic analysis utility portion of ANOPP2 is housed in *libAcousticAnalysis*; a user may link just that piece into their user code if desired.

## 4.2 Frame of Reference

A noise prediction using ANOPP2 may include many different components including flight paths, observers, geometrical structures, and functionality in the form of acoustic codes that model physical aspects of the aircraft. Every component in the ANOPP2 computation may have a different frame of reference to which they subscribe. It is important to associated all components within the ANOPP2 system to the correct frame of reference. The ANOPP2 toolkit includes a complex and capable system to define frame of reference between objects, such as a component on the aircraft and the observer on the ground. That system is the Kinematics API which defines the orientation of one frame of reference to another by way of a series of frame changes [46]. Using the Kinematics API, components of ANOPP2 can be reoriented such that the inputs and outputs are associated with correct orientations. See the reference manuals of each component for information on its orientation and ways to incorporate the frame of reference changes.

## 4.3 Tags

The ANOPP2 framework does not allow direct access to information stored in data structures by the user. Instead of direct access, unique integer tags are provided by ANOPP2 whenever a data structure is created. ANOPP2 provides routines that, when provided with a tag by the user, return information stored in a data structure associated with the tag. In this way, the interface between the user and the data structures and functional modules is simplified.

## 4.4 Error Handling

To aid the user in developing and debugging user code, ANOPP2 provides error handling capabilities. Nearly all routines in ANOPP2 return an integer which communicates the status of the operation. The integer is returned as zero if no error occurred and non-zero if an error occurred. If an error occurs, ANOPP2 generates an error log (*ANOPP2.log*) that contains more information regarding the error encountered.

## 4.5 Versioning

ANOPP2 utilizes a four integer system for version control: major version, minor version, debug version, and revision number. These are defined as follows:

- Major Version
  This number identifies major version changes in the ANOPP2 coding that will most likely require the user to update their user code due to interface changes, new dependencies, etc. This number is not changed often.
- Minor Version
  This number identifies user interface changes that may require the user to update their user code. It usually indicates additional capability that have been added into the framework or new functional modules that have been created.
- Debug Version
  This number identifies an updated version of ANOPP2 that contains bug fixes and capability updates, including user requested updates. Changes to this number typically do not require user code changes.
- Revision Number
  This number identifies a specific build/version and is primarily used by the ANOPP2 development team to identify a specific set of changes internally.

## 4.6 Getting Started

ANOPP2 contains a significant number of demonstration cases and examples to aid a new user in getting started with ANOPP2. Specifically:

- For a user who is familiar with noise prediction and ANOPP, the *QuickStartGuide.pdf* contains a set of user codes starting from a simple ANOPP execution, to a complex aircraft simulation using multiple functional modules.
- In addition to the *QuickStartGuide.pdf*, every reference manual contains a set of demonstration user codes specific to the functionality in the manual. For example, the Acoustic Analysis API reference manual contains a demonstration of many of the noise metric functionalities the Acoustic Analysis API provides.

- The *FunctionalModulesManual.pdf* contains example user codes for every one of the internal functional modules and provides the required information for the data structures needed by the functional module.
- Every routine available in the ANOPP2 framework is accompanied by a demonstration user code in multiple programming languages (Fortran, C++, and Python).

## 4.7 Obtaining ANOPP2

ANOPP2 is distributed in two forms: U.S. General Release and General Public Release. The U.S. General Release contains all capabilities of the framework, including all internal functional modules. The General Public Release has been approved for international usage but does not include all capabilities included in the U.S. General Release. In addition to the ANOPP2 toolkit, additional software, such as plugins and peripherals supported by NASA, may require additional authorizations. To request ANOPP2 please contact Dr. Leonard V. Lopes at NASA Langley Research Center at *leonard.v.lopes@nasa.gov*.

# 5 Example Predictions

This section presents two examples of use of ANOPP2 routines in order to perform a noise prediction. The first example shows how to use ANOPP2 to perform a prediction of noise from an aircraft in flight. The second example shows how to utilize ANOPP2 to use CFD results and perform a prediction that can be compared to a measurement of noise from a model-scale source placed in a wind tunnel environment.

## 5.1 Aircraft in Flight

Figure 10, shown previously, contains a representative user code that performs the six steps necessary to perform a noise prediction using ANOPP2. For the prediction of noise from an aircraft in flight using ANOPP2, several data structures may be required, namely the atmosphere, flight path, propulsion system, and two observers (the mid-surface and far-surface observers). Not all of the data structures may be required; the data structures required are dependent on the application. After the data structures are defined, the user is required to select functional modules to perform noise predictions. These are separated into two categories: source noise and propagation. All functional modules are assembled into a mission which can then be executed. The final step is to access the results or calculate derived metrics. These can be summarized in the following code using representative function calls.[1]

1. 
   ```
   INITIALIZE ()
   ```

2. 
   ```
   intSuccess = CREATE_ATMOSPHERE (intAtmosphereTag)
   intSuccess = CREATE_FLIGHT_PATH (intFlightPathTag)
   intSuccess = CREATE_PROPULSION_SYSTEM (intPropulsionSystemTag)
   intSuccess = CREATE_OBSERVER (intMidSurfaceTag)
   intSuccess = CREATE_OBSERVER (intCertificationMicrophoneTag)
   ```

3. 
   ```
   intSuccess =
     CREATE_FUNCTIONAL_MODULE
       (intAirframeSourceModuleTag, intFlightPathTag,
        intAtmosphereTag, intMidSurfaceTag)

   intSuccess =
     CREATE_FUNCTIONAL_MODULE
       (intEngineSourceModuleTag, intFlightPathTag,
        intPropulsionSystemTag, intMidSurfaceTag)

   intSuccess =
     CREATE_FUNCTIONAL_MODULE
       (intPropagationModuleTag, intFlightPathTag, intAtmosphereTag,
        intMidSurfaceTag, intCertificationMicrophoneTag)
   ```

[1] Note that the example shown here does not include the names of the actual routines. The actual routine names are language dependent and are described in the appropriate reference manual.

4.
   ```
   intSuccess =
     CREATE_MISSION
       (intMission, intAirframeSourceModuleTag,
        intEngineSourceModuleTag, intPropagationModuleTag)
   ```

5. `intSuccess = EXECUTE_MISSION (intMission)`

6. `intSuccess = CALCULATE_EPNL (intCertificationMicrophoneTag)`

## 5.2 Wind Tunnel Model

Using ANOPP2 for wind tunnel applications follows the same six steps outlined in Fig. 10. In this example, the Geometry API is used to read data from a CFD solution. The solution, stored in a geometry data structure, is used to predict noise at an observer mid surface data structure. The observer mid surface data structure is then accessed to get the acoustic pressure time history.[2]

1. `INITIALIZE ()`

2.
   ```
   intSuccess = CREATE_ATMOSPHERE (intAtmosphereTag)
   intSuccess = CREATE_FLIGHT_PATH (intFlightPathTag)
   intSuccess = CREATE_GEOMETRY (intGeometryTag)
   intSuccess = CREATE_OBSERVER (intMidSurfaceTag)
   ```

3.
   ```
   intSuccess =
     CREATE_FUNCTIONAL_MODULE
       (intSourceModuleTag, intFlightPathTag, intAtmosphereTag,
        intGeometryTag, intMidSurfaceTag)
   ```

4. `intSuccess = CREATE_MISSION (intMissionTag, intSourceModuleTag)`

5. `intSuccess = EXECUTE_MISSION (intMissionTag)`

6. `intSuccess = GET_ACOUSTIC_PRESSURE (intMidSurfaceTag)`

[2] Note that the example shown here does not include the names of the actual routines. The actual routine names are language dependent and are described in the appropriate reference manual.

# References

1. Lopes, L. V. and Burley, C. L., “Design of the Next Generation Aircraft Noise Prediction Program: ANOPP2,” June 5-8, 2011, AIAA Paper No. 2011-2854, presented at the $17^{th}$ AIAA/CEAS Aeroacoustics Conference ($32^{nd}$ AIAA Aeroacoustics Conference).

2. Thomas, R. H., “Subsonic Fixed Wing Project N+2 Noise Goal Summary,” December 4–5, 2007, presented at the NASA Acoustics Technical Working Group. Available on the Langley Technical Report Server (LTRS).

3. Kirby, M. R. and Mavris, D. N., “The Environmental Design Space,” September 14–19, 2008, AIAA Paper No. 2008-8875, presented at the $26^{th}$ Congress of International Council of Aeronautical Sciences (ICAS).

4. Moore, K. T., Naylor, B. A., and Gray, J. S., “The Development of an Open Source Framework for Multidisciplinary Analysis & Optimization,” September 10–12, 2008, AIAA Paper No. 2008-6069, presented at the $12^{th}$ AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference.

5. Wells, D. P. and Olson, E. D., “Hybrid Wing Body Planform Design with Vehicle Sketch Pad,” January 4–7, 2011, AIAA Paper No. 2011-360, $49^{th}$ AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition.

6. Martin, B. and Dillingham, G. L., “Aviation and the Environment: Airport Operations and Future Growth Present Environmental Challenges,” Tech. Rep. GAO/RCED-00-153, United States General Accounting Office, Washington, D.C. 20548, August 2000.

7. Springer, T., “Aeronautics Research Mission Directorate Programs,” September 2015, retrieved from *http://www.aeronautics.nasa.gov/programs.htm*.

8. Hill, G. A. and Thomas, R. H., “Challenges and Opportunities for Noise Reduction Through Advanced Aircraft Propulsion Airframe Integration and Configurations,” November 11–12, 2004, presented at the $8^{th}$ CEAS Workshop on Aeroacoustics of New Aircraft and Engine Configurations.

9. Collier, F. S., “Environmentally Responsible Aviation (ERA) Project,” September 29–October 1, 2009, presented at the NASA Fundamental Aeronautics Program, Third Annual Technical Conference.

10. Thomas, R. H., Burley, C. L., and Nickol, C. L., “Assessment of the Noise Reduction Potential of Advanced Subsonic Transport Concepts for NASA’s Environmentally Responsible Aviation Project,” January 4–8, 2016, AIAA Paper No. 2016-0863, presented at the $54^{th}$ AIAA Aerospace Sciences Meeting.

11. Raney, J. P., “Development of a New Computer System for Aircraft Noise Prediction,” March 24–26, 1975, AIAA Paper No. 75-536, presented at the $2^{nd}$ Aero-acoustics Conference.

12. Zorumski, W. E., “Aircraft Noise Prediction Program Theoretical Manual, Parts 1 and 2,” Tech. Rep. NASA/TM-83199-PT-1 and PT-2, National Aeronautics and Space Administration, Langley Research Center, Hampton, VA., February 1982.

13. Smith, M., Carrilho, J., Molin, N., Piet, J.-F., and Chow, L., “Modeling Landing Gear Noise With Installation Effects,” May 21–23, 2007, AIAA Paper No. 2007-3472, presented at the 13$^{th}$ AIAA/CEAS Aeroacoustics Conference (28$^{th}$ AIAA Aeroacoustics Conference).

14. Lopes, L. V., Brentner, K. S., and Morris, P. J., “Airframe Noise Prediction with Installed Landing Gear for a Complete Aircraft,” May 11–13, 2009, AIAA Paper No. 2009-3155, presented at the 15$^{th}$ AIAA/CEAS Aeroacoustics Conference (30$^{th}$ AIAA Aeroacoustics Conference).

15. Yueping Guo, C. L. B. and Thomas, R. H., “Landing Gear Noise Prediction and Analysis for Tube-And-Wing and Hybrid-Wing-Body Aircraft,” January 4–8, 2016, AIAA Paper No. 2016-1273, presented at the 54$^{th}$ AIAA Aerospace Sciences Meeting.

16. Chec, L., “Closing the Chasm Between Engineers and Simulation Data Management,” November 15, 2011, presented at NAFEMS European Conference on Simulation Process and Data Management, retrieved from *http://www.nafems.org/*.

17. Gray, J. S., Moore, K. T., and Naylor, B. A., “OpenMDAO: An Open Source Framework for Multidisciplinary Analysis and Optimization,” August 2010, AIAA Paper No. 2010-9101, presented at the 13$^{th}$ AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference.

18. Gross., D. C., “Report from the Fidelity Definition and Metrics Implementation Study Group (FDM-ISG),” March 1999, No. 99S-SIW-167, presented at the Spring Simulation Interoperability Workshop (SIW), Simulation Interoperability Standards Organization (SISO).

19. Ffowcs Williams, J. E. and Hawkings, D. L., “Sound Generated by Turbulence and Surfaces in Arbitrary Motion,” *Philosophical Transactions of the Royal Society*, Vol. A264, No. 1151, 1969, pp. 321–342.

20. Farassat, F., Dunn, M. H., and Spence, P. L., “Advanced Propeller Noise Prediction in the Time Domain,” *AIAA Journal*, Vol. 30, No. 9, September 1992, pp. 2337–2340.

21. Lockard, D. P., “A Comparison of Ffowcs Williams - Hawkings Solvers for Airframe Noise Applications,” June 17–19, 2002, AIAA Paper No. 2002-2580, presented at the 8$^{th}$ AIAA/CEAS Aeroacoustics Conference and Exhibit (23$^{rd}$ AIAA Aeroacoustics Conference).

22. Brès, G. A., Brentner, K. S., Perez, G., and Jones, H. E., “Maneuvering Rotorcraft Noise Prediction,” *Journal of Sound and Vibration*, Vol. 275, No. 3–5, August 2004, pp. 719–738.

23. Perez, G., Brentner, K. S., Brès, G. A., and Jones, H. E., “A First Step Toward the Prediction of Rotorcraft Maneuvering Noise,” *Journal of the American Helicopter Society*, Vol. 50, No. 3, July 2005, pp. 230–237.

24. Farassat, F. and Brentner, K. S., “Supersonic Quadrupole Noise Theory For High-Speed Helicopter Rotors,” *Journal of Sound and Vibration*, Vol. 218, No. 3, 1998, pp. 481–500.

25. Bodony, D. and Lele, S., "Current Status of Jet Noise Predictions Using Large-Eddy Simulation," *AIAA Journal*, Vol. 46, No. 2, 2008, pp. 364–380.

26. Tinetti, A. F., Dunn, M. H., and Pope, D. S., *Fast Scattering Code (FSC) User's Manual, Version 2.0*, October 2006, NASA CR 2006-214510.

27. Redonnet, S., *Simlation de la propagation acoustique en présence d'écoulements quelconques et de structures solides, par résolution numérique des équations d'Euler*, Ph.D. thesis, Université Bordeaux I, December 2001.

28. Redonnet, S., Desquesnes, G., Manoha, E., and Parzani, C., "Numerical Study of Acoustic Installation Effects with a CAA Method," *AIAA Journal*, Vol. 48, No. 5, May 2010, pp. 929–937.

29. Hu, F. Q., "Further Development of a Time Domain Boundary Integral Equation Method for Aeroacoustic Scattering Computation," June 16–20 2014, AIAA 2014-3194, presented at the $20^{th}$ AIAA/CEAS Aeroacoustics Conference ($35^{th}$ AIAA Aeroacoustics Conference).

30. Farassat, F., "Linear Acoustic Formulas for Calculation of Rotating Blade Noise," *AIAA Journal*, Vol. 19, No. 9, September 1981, pp. 1122–1130.

31. Unidata, "NetCDF (Network Common Data Form)," retrieved from *http://www.unidata.ucar.edu*, February 2013.

32. Lopes, L. V., *ANOPP2 Command Executive API Reference Manual (Distributed with ANOPP2)*, National Aeronautics and Space Administration, Hampton, VA 23681.

33. Lopes, L. V., *ANOPP2 Plugin System Manual (Distributed with ANOPP2)*, National Aeronautics and Space Administration, Hampton, VA 23681.

34. Lopes, L. V., *ANOPP2 Functional Modules Manual (Distributed with ANOPP2)*, National Aeronautics and Space Administration, Hampton, VA 23681.

35. Lopes, L. V., Iyer, V. R., and Jones, J. J., *ANOPP2 Observer API Reference Manual (Distributed with ANOPP2)*, National Aeronautics and Space Administration, Hampton, VA 23681.

36. McCullers, L. A., "FLOPS User's Guide," Tech. Rep. Release 6.02, NASA Langley Research Center, March 2003.

37. Lopes, L. V., *ANOPP2 Flight Path API Reference Manual (Distributed with ANOPP2)*, National Aeronautics and Space Administration, Hampton, VA 23681.

38. Lopes, L. V., *ANOPP2 Atmosphere API Reference Manual (Distributed with ANOPP2)*, National Aeronautics and Space Administration, Hampton, VA 23681.

39. Lytle, J. K., "The Numerical Propulsion System Simulation: An Overview," Tech. Rep. NASA/TM-2000-209915, National Aeronautics and Space Administration, June 2000.

40. Stauber, L. J. and Naiman, C. G., "Numerical Propulsion System Simulation (NPSS): An Award Winning Propulsion System Simulation Tool," June 7–9, 2002.

41. Lopes, L. V., *ANOPP2 Propulsion System API Reference Manual (Distributed with ANOPP2)*, National Aeronautics and Space Administration, Hampton, VA 23681.

42. Buning, P. G., "User's Manual for OVERFLOW 2.2," March 20, 2015, Version 2.2, retrieved from *http://overflow.larc.nasa.gov/*.

43. Biedron, B. T., "Fun3D: Fully Unstructured Navier-Stokes," November 10, 2015, Version 12.8-dc8fb20, retrieved from *http://fun3d.larc.nasa.gov/*.

44. Lopes, L. V., *ANOPP2 Geometry API Reference Manual (Distributed with ANOPP2)*, National Aeronautics and Space Administration, Hampton, VA 23681.

45. Lopes, L. V., Jones, J. J., and Rawls, J. W., *ANOPP2 Acoustic Analysis API Reference Manual (Distributed with ANOPP2)*, National Aeronautics and Space Administration, Hampton, VA 23681.

46. Lopes, L. V., *ANOPP2 Kinematics API Reference Manual (Distributed with ANOPP2)*, National Aeronautics and Space Administration, Hampton, VA 23681.

47. Lopes, L. V., *ANOPP2 Math API Reference Manual (Distributed with ANOPP2)*, National Aeronautics and Space Administration, Hampton, VA 23681.

48. Lopes, L. V., *ANOPP2 Tag API Reference Manual (Distributed with ANOPP2)*, National Aeronautics and Space Administration, Hampton, VA 23681.

49. Lopes, L. V., *ANOPP2 Error API Reference Manual (Distributed with ANOPP2)*, National Aeronautics and Space Administration, Hampton, VA 23681.

50. Lopes, L. V. and Iyer, V. R., *ANOPP2 Model Center Interface Code (AMCIC) Manual (Distributed with ANOPP2)*, National Aeronautics and Space Administration, Hampton, VA 23681.

51. Lopes, L. V., *ANOPP2 Fortran Coding Standard (Distributed with ANOPP2)*, National Aeronautics and Space Administration, Hampton, VA 23681.

52. Lopes, L. V., *ANOPP2 Quick Start Guide (Distributed with ANOPP2)*, National Aeronautics and Space Administration, Hampton, VA 23681.

53. Lighthill, M. J., "On Sound Generated Aerodynamically, I: General Theory," *Proceedings of the Royal Society. A, Mathematical, Physical, and Engineering Sciences*, Vol. 211, 1952, pp. 564–587.

54. Lighthill, M. J., "On Sound Generated Aerodynamically, II: Turbulence as a Source of Sound," *Proceedings of the Royal Society. A, Mathematical, Physical, and Engineering Sciences*, Vol. 222, 1954, pp. 1–32.

# Acronyms

| | |
|---|---|
| **ADS** | acoustic data surface |
| **AMCIC** | ANOPP2 Model Center Interface Code |
| **ANOPP** | Aircraft NOise Prediction Program |
| **ANOPP2** | Aircraft NOise Prediction Program 2 |
| **AOIC** | ANOPP2 OpenMDAO Interface Code |
| **API** | Application Programming Interface |
| **APTH** | Acoustic Pressure Time History |
| **CFD** | Computational Fluid Dynamics |
| **EPNL** | Effective Perceived Noise Level |
| **FLOPS** | FLight OPtimization Software |
| **FW-H** | Ffowcs Williams and Hawkings |
| **HSI** | High Speed Impulsive |
| **NetCDF** | Network Common Data Form |
| **NPSS** | Numerical Propulsion System Simulation |
| **PAA** | Propulsion Airframe Aeroacoustics |
| **PSD** | Power Spectral Density Spectrum |
| **SPL** | Sound Pressure Level |

# Glossary

**Acoustic Analysis API**

An API for the calculation of derived acoustic metrics commonly used in aircraft noise prediction. The Acoustic Analysis API defines the functions, enumerators, and constants made available through the acoustic analysis utility portion of the ANOPP2 library.

**acoustic code**

Software for predicting noise provided a set of information. An acoustic code is wrapped in a functional module which interfaces with the ANOPP2 framework to provide the information required by the acoustic code and to receive the noise prediction from the acoustic code.

**acoustic data surface (ADS)**

A surface of acoustic properties. For instance, a point source may be represented as a sphere shaped acoustic data surface where the acoustic properties are represented as a power spectral density spectrum (PSD). Another example of an acoustic data surface is a Ffowcs Williams and Hawkings surface, where acoustic pressure, velocity, and density time histories fully define the sources within the surface include their distribution.

**acoustic pressure time history (APTH)**

A time history of acoustic pressures. An APTH can be specified as a function of waypoint (multiple segments) or the entire flight event (long duration).

**Aircaft NOise Prediction Program (ANOPP)**

A NASA computer program to predict aircraft component and system noise. ANOPP has been implemented in three ANOPP2 functional modules. See the Functional Modules Manual for more information. [34].

**Aircraft NOise Prediction Program 2 (ANOPP2)**

Second generation ANOPP. ANOPP2 employs a mixed-fidelity framework to incorporate methods of differing fidelity allowing semi-empirically based, low- resolution methods, such as those found in ANOPP, to be combined with higher resolution, CFD-based methods required for better understanding of the noise-generating mechanisms.

**ANOPP2 Model Center Interface Code (AMCIC)**

An ANOPP2 peripheral application that facilitates easy inputs to and use of ANOPP2 within the Model Center environment.

**ANOPP2 OpenMDAO Interface Code (AOIC)**

An ANOPP2 peripheral that facilitates easy integration of ANOPP2 within the OpenMDAO environment.

**application programming interface (API)**

A set of rules or protocol to access public functionality, variables, and classes of a precompiled library. The same interface exists regardless of operating system and configuration during compilation of the library.

**Atmosphere API**

An API for the atmosphere data structure. The atmosphere API defines the functions, enumerators, and constants made available through the atmosphere portion of the ANOPP2 library.

**Command Executive API**

An API into the top-level capability of ANOPP2, in which functional modules, data structures, and utilities are organized into a single, unified system. The command executive contains functionality for plugins, functional modules, and missions as well as hooks to the other APIs available within ANOPP2.

**constant**

Value that cannot change. Many components of ANOPP2, such as the acoustic analysis utility, come with constants that are available for a user. These values are the same values as those that are used internally within the library.

**data structure**

Common or related information organized into a single unit. ANOPP2 includes several data structures including the atmosphere, flight path, and observer. A data structure is create when requested by the user code and can be accessed by the user code using a tag provided by ANOPP2.

**enumerator**

Constant integer whose value is unique when compared to other enumerators in its enumerator group. These are used to convey information to ANOPP2 such as specific units, metrics, formatting options, and orientations.

**enumerator group**

A group of enumerators used to convey a set of options. These are used in ANOPP2 to convey information selection of a user such as units, metrics, options, and orientations.

**Error API**

A utility API within ANOPP2 for the handling, storing, and reporting of errors.

**far-surface**

A group of observer nodes that are far away from the noise source (e.g. a location on the ground). Although named a surface, the far-surface can be a point, line, plane, or surface. The name “far-surface” is used to communicate similarities with the source- and mid-surface.

**Ffowcs Williams and Hawkings (FW-H)**

An integration method in computational aeroacoustics introduced by John Ffowcs Williams and David L. Hawkings. [19] It is based on the acoustic analogy of James Lighthill. [53, 54]

**Fidelity**

The accuracy of the representation when compared to the real world. Many characteristics define fidelity, including error, precision, sensitivity, resolution, tolerance, and validity.

**flight condition**

An aggregation of a waypoint and flight properties for a corresponding source time. Many different properties can be defined in the flight condition including position, Mach number, throttle setting, and aircraft orientation. These properties are defined at the source times of the flight path.

**Flight Path API**

An API for the flight path data structure. The Flight Path API defines the functions, enumerators, and constants made available through the flight path portion of the ANOPP2 library.

**functional module**

A capability within ANOPP2 for computing acoustic data by means of an acoustic code. All functional modules within ANOPP2 take in a series of data structures, such as a flight path and observer, and predict noise which is then cast into the observer data structure. The information provided by the functional module is then made available to a user via the observer data structure. Functional modules may either be internal functional modules (contained within the ANOPP2 library) or plugins, separate entities of compiled code provided by the ANOPP2 team or developed by a user and loaded at run time.

**Geometry API**

An API for the geometry data structure. The geometry API defines the functions, enumerators, and constants made available through the geometry portion of the ANOPP2 library.

**Kinematics API**

An API for the kinematics utility. The kinematics API defines the functions, enumerators, and constants made available through the kinematics portion of the ANOPP2 library.

**library**

A library is a collection of compiled functionality. A library contains public and private functionality, where only the public functionality can be accessed via an API. The library compilation is dependent on the operating system and the compiler directives used during its creation (such as parallel capability).

**Math API**

A utility API that includes common mathematical operations. This includes such operations as interpolation and equation operations.

**mid-surface**

An observer data stucture that forms an envelope around a combination of several noise sources (e.g., a spherical envelope surrounding an aircraft, that moves with the aircraft). The mid-surface can be of differing fidelity, i.e., can include noise metrics of differing fidelity, and can be of any shape depending on the fidelity

**mission manager**

A manager, made available through the Command Executive API, that collects, organizes, and executes the functional modules.

**Network Common Data Form (NetCDF)**

Set of software libraries and self-describing, machine-independent data formats that support the creation, access, and sharing of scientific data. [31]

**Numerical Propulsion System Simulation (NPSS)**

An object-oriented, multi-physics, engineering design and simulation environment that enables development, collaboration and seamless integration of system models, including the aerothermodynamic behavior of an aircraft's engines. [39, 40]

**Observer API**

An API for the observer data structure. The Observer API includes functions, enumerators, and constants made available through the observer portion of the ANOPP2 library.

**plugin**

A functional module that is created by a user and linked in via the ANOPP2 plugin system component of the command executive.

**plugin system**

A portion of the command executive that loads ANOPP2 plugins at run time.

**power spectral density spectrum (PSD)**

A noise metric of pressure squared as a function of frequency with units of $Pa^2/\text{Hz}$. It is computed by first Fourier transforming the pressure time signal and computing from the Fourier coefficients. Where

$$G_{PP,m} = \frac{|A_m|^2 \Delta t^2}{T} \quad \text{for} \quad m = 0$$

$$G_{PP,m} = 2\frac{|A_m|^2 \Delta t^2}{T} \quad \text{for} \quad 1 \leq m \leq (N/2-1)$$

Here, $G_{PP}$ is the single sided PSD, $A_m$ are Fourier transform coefficients, $T$ is the period of acoustic pressure time history, and $\Delta t$ is time step size of the APTH.

**Propulsion API**

An API for the propulsion system data structure. The propulsion API defines the functions, enumerators, and constants made available through the propulsion system portion of the ANOPP2 library.

**psuedo-observer**

A sphere of nodes surrounding a noise source where acoustic information is stored. The psuedo-observer typically includes acoustic information in the form of 1/3-octave band SPL spectra and includes flight effects (Doppler frequency shift and convective amplification).

**routine**

A single function or subroutine call that executes a series of computations. The routine is called from a user code using input specified by the user, performs some operation, and returns output to the user code. An integer is typically returned as well to communicate success of the operation or to indicate if any problems occurred.

**sound pressure level (SPL)**

A logarithmic measure of the effective sound pressure relative to a reference value.

$$\text{SPL} = 20\log_{10}\left(\frac{p_{\text{rms}}}{p_{\text{ref}}}\right)\text{dB}$$

where, $p_{\text{ref}}$ is a reference sound pressure ($2 \times 10^{-5}$ Pa) and $p_{\text{rms}}$ is the root mean square of the sound pressure being measured.

**source time**

A time value associated with a unique flight condition (a series of which define the aircraft properties through all time). This differs from emission time which is defined as the time at which the noise is emitted. Typically there exist many emissions times and relatively few source times.

**source-surface**

An observer data structure that forms an envelope around each noise source, e.g., a surface surrounding an engine and propeller, that moves with the noise source.

**suppression**

The amount of noise reduction compared to the noise from a baseline configuration. This includes the acoustic effect of PAA (such as shielding) and other noise reduction technologies.

**Tag API**

A utility API for the creating, classifying, deleting, and accessing of unique tag identifiers.

**toolkit**

A group of interrelated software packages that are used in synchrony to perform a given task. A toolkit is often accompanied with extensive documentation to aid the user in applying the toolkit to their given problem.

**tools**

Additional software provided with ANOPP2 that are not part of the framework or functional modules.

**user code**

A computer-language text file (Fortran, C++, or Python) that uses features of ANOPP2. This includes the routines, enumerators, and/or constants made available by the API.

**user interface**

A layer between a user and the user's desired functionality. The user must utilize the interface in order to excute the functionality hidden in an encapsulated piece of software.

**utilities**

Smaller libraries that contain common mathematics, error handling, and unique tagging capability to the other components of ANOPP2. The utilities include the Kinematics, Acoustic Analysis, Tag, Math, and Error APIs.

# Index

| REPORT DOCUMENTATION PAGE | *Form Approved*<br>*OMB No. 0704–0188* |
|---|---|

**1. REPORT DATE** *(DD-MM-YYYY)*
01-10-2016

**2. REPORT TYPE**
Technical Memorandum

**3. DATES COVERED** *(From - To)*

**4. TITLE AND SUBTITLE**
ANOPP2 User's Manual
Version 1.2

**5a. CONTRACT NUMBER**

**5b. GRANT NUMBER**

**5c. PROGRAM ELEMENT NUMBER**

**6. AUTHOR(S)**
Lopes, Leonard V. and Burley, Casey L.

**5d. PROJECT NUMBER**

**5e. TASK NUMBER**

**5f. WORK UNIT NUMBER**
109492.02.07.01.10

**7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)**
NASA Langley Research Center
Hampton, VA 23681-2199

**8. PERFORMING ORGANIZATION REPORT NUMBER**
L–20692

**9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)**
National Aeronautics and Space Administration
Washington, DC 20546-0001

**10. SPONSOR/MONITOR'S ACRONYM(S)**
NASA

**11. SPONSOR/MONITOR'S REPORT NUMBER(S)**
NASA/TM–2016–219342

**12. DISTRIBUTION/AVAILABILITY STATEMENT**
Unclassified-Unlimited
Subject Category 00
Availability: NASA STI Program (757) 864-9658

**13. SUPPLEMENTARY NOTES**
An electronic version can be found at http://ntrs.nasa.gov.

**14. ABSTRACT**

This manual documents the Aircraft NOise Prediction Program 2 (ANOPP2). ANOPP2 is a toolkit that includes a framework, noise prediction methods, and peripheral software to aid a user in predicting and understanding aircraft noise. This manual includes an explanation of the overall design and structure of ANOPP2, including a brief introduction to aircraft noise prediction and the ANOPP2 background, philosophy, and architecture. The concept of nested acoustic data surfaces and its application to a mixed-fidelity noise prediction are presented. The structure and usage of ANOPP2, which includes the communication between the user, the ANOPP2 framework, and noise prediction methods, are presented for two scenarios: wind-tunnel and flight. These scenarios serve to provide the user with guidance and documentation references for performing a noise prediction using ANOPP2.

**15. SUBJECT TERMS**
Acoustics, Aeroacoustics, Acoustic Analysis, Aircraft Noise

| 16. SECURITY CLASSIFICATION OF: a. REPORT | b. ABSTRACT | c. THIS PAGE | 17. LIMITATION OF ABSTRACT | 18. NUMBER OF PAGES | 19a. NAME OF RESPONSIBLE PERSON |
|---|---|---|---|---|---|
| U | U | U | UU | 40 | STI Information Desk (help@sti.nasa.gov) |
| | | | | | **19b. TELEPHONE NUMBER** *(Include area code)*<br>(757) 864-9658 |