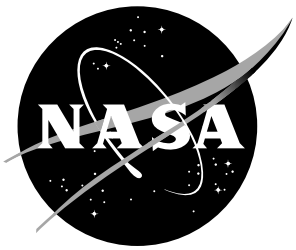
NASA–TM–20210021111



ANOPP2's Farassat Formulations Internal Functional Modules (AFFIFMs) Reference Manual

Version 1.4

General Public Release

Debug Build
No Parallelization or Multithreading

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

December 2021

NASA STI Program. . . in Profile

Since its founding, NASA has been dedicated to the advancement of aeronautics and space science. The NASA scientific and technical information (STI) program plays a key part in helping NASA maintain this important role.

The NASA STI Program operates under the auspices of the Agency Chief Information Officer. It collects, organizes, provides for archiving, and disseminates NASA's STI. The NASA STI Program provides access to the NASA Aeronautics and Space Database and its public interface, the NASA Technical Report Server, thus providing one of the largest collection of aeronautical and space science STI in the world. Results are published in both non-NASA channels and by NASA in the NASA STI Report Series, which includes the following report types:

- TECHNICAL PUBLICATION. Reports of completed research or a major significant phase of research that present the results of NASA programs and include extensive data or theoretical analysis. Includes compilations of significant scientific and technical data and information deemed to be of continuing reference value. NASA counterpart of peer-reviewed formal professional papers, but having less stringent limitations on manuscript length and extent of graphic presentations.

- TECHNICAL MEMORANDUM. Scientific and technical findings that are preliminary or of specialized interest, e.g., quick release reports, working papers, and bibliographies that contain minimal annotation. Does not contain extensive analysis.

- CONTRACTOR REPORT. Scientific and technical findings by NASA-sponsored contractors and grantees.

- CONFERENCE PUBLICATION. Collected papers from scientific and technical conferences, symposia, seminars, or other meetings sponsored or co-sponsored by NASA.

- SPECIAL PUBLICATION. Scientific, technical, or historical information from NASA programs, projects, and missions, often concerned with subjects having substantial public interest.

- TECHNICAL TRANSLATION. English-language translations of foreign scientific and technical material pertinent to NASA's mission.

Specialized services also include organizing and publishing research results, distributing specialized research announcements and feeds, providing information desk and personal search support, and enabling data exchange services.

For more information about the NASA STI Program, see the following:

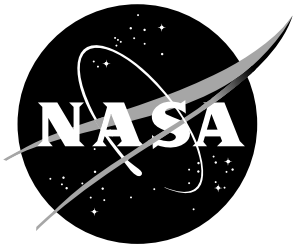
- Access the NASA STI program home page at `http://www.sti.nasa.gov`

- E-mail your question to `help@sti.nasa.gov`

- Phone the NASA STI Information Desk at 757-864-9658

- Write to:
  NASA STI Information Desk
  Mail Stop 148
  NASA Langley Research Center
  Hampton, VA 23681-2199

NASA–TM–20210021111



ANOPP2's Farassat Formulations Internal Functional Modules (AFFIFMs) Reference Manual

Version 1.4

General Public Release

Debug Build
No Parallelization or Multithreading

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

National Aeronautics and
Space Administration

Langley Research Center

December 2021

Acknowledgments

This work is funded by the NASA Aeronautic Research Mission Directorate (ARMD), specifically the Transformational Tools and Technologies (TTT) Project in the Transformative Aeronautics Concepts Program (TACP) and the Advanced Air Transport Technology (AATT) Project, Commercial Supersonic Technology (CST) Project, and Revolutionary Vertical Lift Technology (RVLT) Project in the Advanced Air Vehicles Program (AAVP).

Available from:

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

Abstract

This manual documents version 1.4.0 of ANOPP2's Farassat's Formulations Internal Functional Modules (AFFIFMs) developed by NASA Langley Research Center's Aeroacoustics Branch. The AFFIFMs provide the capability of calculating an acoustic pressure time history (APTH), or similar metric, provided one or more Ffowcs Williams and Hawkings (FWH) surfaces. AFFIFMs also allow for compact line sources.

This application programming interface (API) is part of a larger toolkit called the Aircraft NOise Prediction Program 2 (ANOPP2). The goal of ANOPP2 is to provide the ability to independently: (1) assess aircraft system noise; (2) assess aircraft component noise; and (3) evaluate aircraft noise reduction technologies and flight procedures. Additionally, ANOPP2 is designed to provide a capability for understanding the fundamental physics involved in noise generation to support experiments and flight demonstration activities.

As a component of ANOPP2, ANOPP2's Farassat's Formulations Internal Functional Modules and this document may be included as part of the ANOPP2 distribution, or they may be provided independent of that distribution.

Contents

List of Symbols

English letters

A_m	Fourier transform coefficients
A, B, C	noncompact propagation coefficients
$\mathscr{A}, \mathscr{B}, \mathscr{C}$	compact propagation coefficients
$\mathcal{A}, \mathcal{B}, \mathcal{C}$	noncompact field coefficients
$\mathbb{A}, \mathbb{B}, \mathbb{C}$	compact field coefficients
a_u, a_l	linear interpolation coefficients
C_d	coefficient of drag
C_l	coefficient of lift
C_T	coefficient of thrust
C_{l_α}	lift curve slope, $C_{l_\alpha} = 2\pi\alpha_e$
c	speed of sound
dB	decibel
$\boldsymbol{F}_i$	lifting line source strength
f	frequency
f	integration surface defined by $f = 0$
F	integration line defined by $F = 0$
G	Green's function
G_{pp}	single sided spectral density
g	collapsing sphere
$H(x)$	Heaviside function
J	Jacobian of differential surface area
K	Jacobian of differential line length
k_i	key at time step i
L	characteristic size of surface
$\boldsymbol{L}_i$	dipole source term
M	Mach number
$\boldsymbol{M}_i$	Mach vector $\boldsymbol{M}_i = \boldsymbol{v}_i / c_\infty$
ρu_i	fluid momentum
N	number of polynomial coefficients
$\hat{\boldsymbol{n}}_i$	surface normal vector
$\boldsymbol{P}'$	vector intregrand of acoustic pressure
$\boldsymbol{p}_{ij}$	compressive stress tensor
p	fluid pressure
Q	monopole source term
R	propagation function
r	nondimensional radial position
r	radiation distance $r = \lvert\boldsymbol{r}_i\rvert$
$\boldsymbol{r}_i$	radiation vector
S	surface area
$\boldsymbol{T}_{ij}$	Lighthill stress tensor
$\tilde{T}$	period of characteristic fluctuation
T	period
t	maximum airfoil thickness
t	observer time
t	time
$\boldsymbol{U}_i$	fluid mass flux
$\boldsymbol{u}_i$	Cartesian flow velocity
u	time independent line coordinates
u_1, u_2	time independent surface coordinates
$\boldsymbol{V}'$	vector intregrand of acoustic velocity
$\boldsymbol{v}_i$	Cartesian surface velocity
$\boldsymbol{x}_i$	observer coordinates
$\boldsymbol{x}_i$	Cartesian coordinates
x	nondimensional chord position

y_t	airfoil half thickness
$\boldsymbol{y}_i$	source coordinates

Greek letters

α_e	effective angle of attack
$\delta(x)$	Dirac delta function
$\boldsymbol{\delta}_{ij}$	Kronecker-delta
ε	additional time factor
ϵ	source time tolerance
Γ	intersection $f = 0$ and $g = 0$
μ	rotor advance ratio
Φ	acoustic velocity potential
Ψ	cross sectional area
$\mathbf{\Phi}$	vector intregrand of velocity potential
ρ	fluid density
Σ	source surface
σ	rotor solidity
$\boldsymbol{\sigma}_{ij}$	viscous stress tensor
$[\Theta_{N/0}]$	frame of reference change
τ	source time $\tau = t - {}^{r}/_{c_\infty}$
ω	angular frequency $\omega = 2\pi f$
ζ	spanwise blade length

Subscripts

x_a	acoustic sources
$x_{\mathbf{rms}}$	root mean square
x_d	dipole term
x_e	estimated value
x_g	specified in kinematics
x_∞	freestream quantity
x_l	lower band limit
x_M	dotted with the Mach vector
x_m	monopole term
x_n	unit normal direction
x_p	previous execution
x_q	quadrupole term
x_r	dotted with radiation hat
$x_{\mathbf{ref}}$	reference value
$x_{\mathbf{ret}}$	evaluated at source time
x_{ss}	spurious signals
x_s	surface terms
x_u	specified in configuration file
x_u	upper band limit
x_v	volume term

Symbols

$\Box^2$	wave operator ${}^{1}/_{c_\infty^2}{}^{\partial^2}/_{\partial t^2} - {}^{\partial^2}/_{\partial x_i^2}$

Accents

$\bar{x}$	generalized derivative
$\dot{x}$	source time derivative $\dot{x} = {}^{\partial x}/_{\partial \tau}$
x^*	local frame of reference
x'	perturbation quantity
$\tilde{x}$	Fourier transform $\tilde{x} = \int x e^{-i\omega t} \mathrm{d}t$
$\hat{x}$	unit vector

List of Figures

List of Tables

Change Log

Version 1.3.0

1. This is the initial instantiation of this reference manual.
2. Updated the observer time interpolation technique to include user defined number of polynomial coefficients (N). See Sec. 3.1.7 for more information.
3. Updated the implicit time algorithm to force the emission time to coincide with the source time defined in the periodic data (such as that extracted from CFD) when present. See Sec. 3.2 for more information.
4. Added frequency domain metadata. Turning this option on will cause the data in the metadata files to be converted to the frequency domain, which allows a user to see where certain frequency content is coming from. The positions of the surface or line are shown as if from the first time step. As a result, for example, surfaces or lines representing rotating, time dependent rotor blade surfaces look stationary.

Version 1.4.0

1. AF1AIFM now supports the FIELD option for strResultsDesired, where the monopole and dipole noise is returned with near field, far field, and (for compact monopole noise) ultra near field results. The returned APTH results are returned in the following order for noncompact: far field monopole, near field monopole, far field dipole, and near field dipole, and for compact: far field monopole, near field monopole, ultra near field monopole, far field dipole, and near field dipole.

1 Overview

The derivation of the Ffowcs Williams and Hawkings (FWH) equation [1] uses generalized function theory [2] and Lighthill's stress tensor [3, 4] to define the acoustic pressure caused by a noise generating mechanism. The FWH equation, Eq. 1, includes two surface source terms and one volume source term. The surface source terms are identified by the delta function, $\delta(x)$, and the volume source term is identified by the Heaviside function, $H(x)$. The source terms on the right-hand-side of the FWH equation often are called the monopole, dipole, and quadrupole terms, respectively.[1] The surface f, shown in Fig. 1, can be impermeable, such as a rotor blade surface, or permeable, such as a computational surface surrounding the entire rotor. The impermeable source terms are shown in Eq. 2, and the permeable source terms are shown in Eq 3. The permeable form of the FWH equation is used frequently with a computational fluid dynamics (CFD) solution, which provides the flow properties at the permeable surface position. The CFD solution calculates the nearfield hydrodynamics, including the necessary acoustic generation and propagation inside the surface. Integral solutions of the FWH equation, often assuming a free space Green's function, calculate the noise propagation from the surface to an observer location in the nearfield or farfield. The CFD/FWH approach has been applied to many different types of geometries, such as helicopter rotors, open rotors, landing gear, slats, flaps, trailing edges, jets, and wind turbines.

$$\bar{\Box}^2 p' = \frac{\partial}{\partial t}\{Q\delta(f)\} - \frac{\partial}{\partial \boldsymbol{x}_i}\{\boldsymbol{L}_i\delta(f)\} + \frac{\bar{\partial}^2}{\partial \boldsymbol{x}_i \partial \boldsymbol{x}_j}\{\boldsymbol{T}_{ij}H(f)\} \tag{1}$$

$$Q = \rho_\infty \boldsymbol{U}_i \hat{\boldsymbol{n}}_i = \rho_\infty \boldsymbol{v}_n, \qquad \boldsymbol{L}_i = \boldsymbol{p}_{ij}\hat{\boldsymbol{n}}_j, \qquad \boldsymbol{T}_{ij} = \rho \boldsymbol{u}_i \boldsymbol{u}_j - \boldsymbol{\sigma}_{ij} + (p' - c_\infty^2 \rho)\boldsymbol{\delta}_{ij} \tag{2}$$

$$\begin{gathered} Q = \rho_\infty \boldsymbol{U}_i \hat{\boldsymbol{n}}_i = \rho_\infty \boldsymbol{v}_n + \rho(\boldsymbol{u}_n - \boldsymbol{v}_n), \qquad \boldsymbol{L}_i = \boldsymbol{p}_{ij}\hat{\boldsymbol{n}}_j + \rho \boldsymbol{u}_i(\boldsymbol{u}_n - \boldsymbol{v}_n) \\ \boldsymbol{T}_{ij} = \rho \boldsymbol{u}_i \boldsymbol{u}_j - \boldsymbol{\sigma}_{ij} + (p' - c_\infty^2 \rho)\boldsymbol{\delta}_{ij} \end{gathered} \tag{3}$$

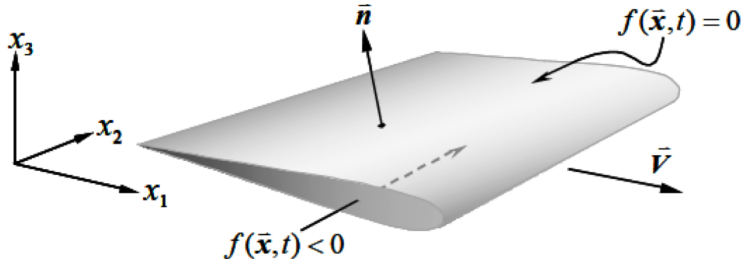


Figure 1: Surface defined by $f = 0$, $\hat{\boldsymbol{n}}_i = {}^{\partial f}/_{\partial \boldsymbol{x}_i}$, $f < 0$ inside, and $f > 0$ outside the surface.

1.1 Description

One such integral solution approach applied to the FWH equation is the application of a free space Green's function by Farassat [5, 6]. This section explains the ANOPP2 [7, 8] capability that

[1] The source terms on the right-hand-side are called the monopole, dipole, and quadrupole terms, respectively, even though this is a misnomer. Technically, they are source time and spatial derivatives of monopole, dipole, and quadrupole noise source terms, leading to confusion about their radiative characteristics.

provides access to an implementation of Farassat's Formulations, so named ANOPP2's Farassat's Formulations Internal Functional Modules (AFFIFMs). In general, all of Farassat's Formulations assume a free space Green's function to solve for the acoustic pressure (Formulations 1, 1A, and Q1A), pressure gradient (Formulations G0, G1, and G1A), acoustic velocity (Formulation V1 and V1A), or acoustic pressure frequency spectrum (Formulation 2B) at a microphone location (sometimes referred to as an observer position). Only a subset of the possible Farassat Formulations are included in ANOPP2; they include Formulation 1, 1A, 2B, G0, G1, G1A and V1A [6, 9–12]. Each formulation is available in a unique ANOPP2's Internal Functional Module (AIFM); there is an AIFM to provide the result from 1A (sometimes referred to as ANOPP2's Formulation 1A Internal Functional Module (AF1AIFM)), another for G1A (sometimes referred to as ANOPP2's Formulation G1A Internal Functional Module (AFG1AIFM)), etc. The inputs for all the AIFMs are very similar, though some differences do exist. All implementations of Farassat's Formulations adhere to the following requirements as outlined by Farassat [13]:

1. There must be no restrictions on the geometry of the noise generator. This means that a formulation cannot utilize the compact source assumption; however, 'compact forms' of existing solutions can be derived.
2. There must be no restrictions on the kinematics of the noise generator. This means that the motion of the source cannot be restricted to lie on a straight line at a constant Mach number, and the surface may be rigid or deforming.
3. The result must be valid in the nearfield and farfield.
4. One should be able to calculate the noise for an observer that is stationary in the medium, in motion with the aircraft, or undergoing any arbitrary motion.

For this reason, Formulation 1C can be more aptly referred to as the rectilinear motion form (or wind tunnel configuration) of Formulation 1A [14]. Furthermore, any compact assumptions made to a Formulation result in a compact form of that Formulation, not a new Formulation.

All AFFIFMs include the capability of handling deforming or nondeforming, node or cell centered, structured or unstructured, single or double precision, and permeable or impermeable surfaces or compact lines. The surface and line deformation and flow quantities can be defined as constant, periodic, or aperiodic in time (independently). The surface or line and observer can be moving (independently) relative to the ground frame of reference by specifying a series of frame transformations as a function of time [15]. All AFFIFMs are optimized for reduced memory and faster computations via Open Message Passing (OpenMP) [16] and are parallelized by observer position and/or FWH surface nodes on the FWH surface using Message Passing Interface (MPI) [17] via ANOPP2's Message Passing Interface Tool (AMPIT) [18]. The integration contains several different options for speed and efficiency. These options include: calculating the derivatives up front or on an as needed basis; reading the surface information as needed for reduced memory requirements; and writing files for analysis and debugging.

All AFFIFMs implemented in ANOPP2 include only the surface terms of the FWH equation. The inputs into all AFFIFMs are the surface or line definition and the perturbation flow quantities, ρ, ρu_i, and p' as a function of time for permeable surfaces, only p' for impermeable surfaces, and Ψ and $\boldsymbol{F}_i$ for compact sources (see Sec. 2). Refer to the ANOPP2's Geometry Data Structure (AGDS) Reference Manual for more information on how to create an input containing the position

and flow quantities of the FWH surface [19], the ANOPP2's Kinematics Utility (AKU) Reference Manual for more information on specifying motion of the FWH surface [15], and the ANOPP2's Observer Data Structure (AODS) Reference Manual for more information on specifying the observer location(s) [20].

Many equations listed during derivatives (see Sec. 2) include surface and line deformation [21] and use the shorthand notation [10] shown in Eq. 4, where m_1 and m_2 are integer coefficients that are defined in each formulation.

$$R(m_1, m_2) = r^{-m_1}(1 - M_r)^{-m_2} \tag{4}$$

The surface or line is assumed to be deforming, where each differential surface or line element is defined as a Jacobian transformation multiplied by a time independent coordinate system fixed to the surface or line [21].

$$\mathrm{d}S = J\mathrm{d}u_1\mathrm{d}u_2 \qquad \mathrm{d}\zeta = K\mathrm{d}u \tag{5}$$

In addition to the above features, all AFFIFMs include the capability to predict combinations of integrals. For example, the calculation may return the total acoustic metric (pressure, pressure gradient, etc.), monopole and dipole terms, or several metrics that may be grouped on Q or $\mathbf{\Psi}$ and $\boldsymbol{L}_i$ or $\boldsymbol{F}_i$, blade deformation, or their source time deivatives.

Lastly, all AFFIFMs contain the ability to export metadata in the form of a Tecplot-compatible binary file that contains a single surface's or line's acoustic, hydrodynamic, and kinematic properties [22]. The metadata can be exported for debugging or flow visualizations. There exist four different settings of metadata: *BASIC* (including the surface or line position and observer and source time of all surface or line nodes), *GEOMETRY* (including the surface or line hydrodynamics and kinematics), *SOURCE* (including all derivatives of the surface or line hydrodynamics and kinematics), and *ALL* (including the acoustic properties for all integrands at all surface or line nodes). Each metadata setting includes all previous ones; for example, the *SOURCE* metadata setting includes the *GEOMETRY* and *BASIC* metadata levels.

1.2 Software Release Level

Currently this code is at software release level (SRL) 3. SRL 3 software has undergone extensive testing by the ANOPP2 software development team and is distributed with complete documentation and several demonstration cases showcasing the available capability. All AFFIFMs are fully supported by the ANOPP2 development team.

2 Derivations

This chapter briefly describes the derivation of each of the Formulations supported within AFFIFMs, for more information refer to Reference [23]. The derivations start with some background material, including application of the free space Green's function to the FWH equation, assumptions made during derivation of the compact equations, and how the terms are grouped for analysis. These techniques are used throughout the derivation of all Formulations and are presented up front. Then, even though Formulation 1A is the most commonly used Formulation, the derivation of Formulation 1 will be described. This is because, mathematically, the derivation of Formulation 1 leads to the derivation of Formulation 1A. Afterward, the derivation of Formulation 1A, the pressure gradient Formulations, and the remaining Formulations will be explained. Along with each Formulation derivation, explanations on how the terms are grouped when used within ANOPP2 are also shown where applicable.

2.1 Common Mathematical Derivations

This section includes common mathematical operations that are applicable to all of the Formulations implemented in AFFIFMs.

2.1.1 Application of Free Space Green's Function

The starting point of the Formulations available in ANOPP2 is the FWH equation without the volume (or quadrupole) term. This is shown in Eq. 6 and is separated into monopole and dipole components in Eq. 7.

$$\bar{\Box}^2 p' = \frac{\partial}{\partial t}\{Q\delta(f)\} - \frac{\partial}{\partial \boldsymbol{x}_i}\{\boldsymbol{L}_i\delta(f)\} \tag{6}$$

$$\bar{\Box}^2 p'_m = \frac{\partial}{\partial t}\{Q\delta(f)\}, \qquad \bar{\Box}^2 p'_d = -\frac{\partial}{\partial \boldsymbol{x}_i}\{\boldsymbol{L}_i\delta(f)\} \tag{7}$$

The free space Green's function introduces the concept of source (also called emission or retarded) space time coordinates ($\boldsymbol{y}_i$, τ) and receiver (or observer) space time coordinates ($\boldsymbol{x}_i$, t). This is shown in Eq. 8.

$$G(\boldsymbol{x}_i(t), t; \boldsymbol{y}_i(\tau), \tau) = \begin{cases} 0 & \tau > t \\ \delta(g)/4\pi r & \tau \leq t \end{cases} \tag{8}$$

where an equation for a sphere that collapses around the observer is defined as

$$g = t - (\tau + r/c_\infty) \tag{9}$$

and

$$r = |\boldsymbol{r}_i| = |\boldsymbol{x}_i(t) - \boldsymbol{y}_i(\tau)| \tag{10}$$

The function $g = 0$ represents a collapsing sphere centered at the observer location where $r = c_\infty(t - \tau)$. The intersection of $g = 0$ and the surface $f = 0$ is the Γ curve, the portion of the

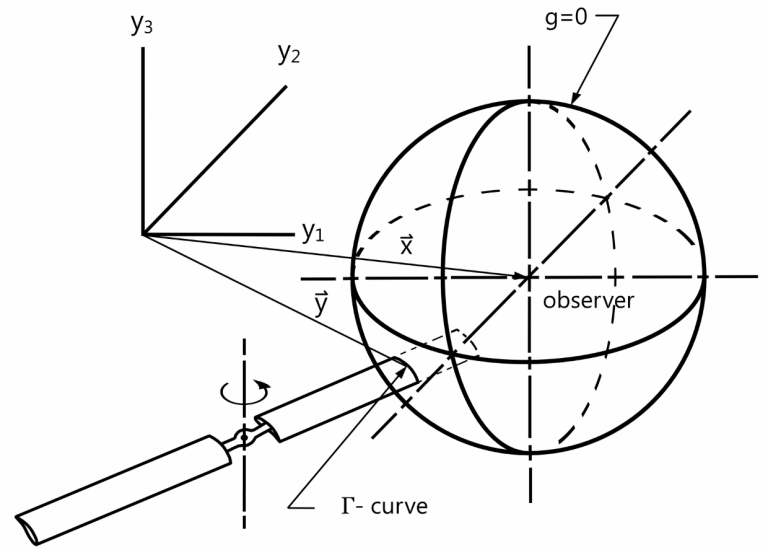


Figure 2: The Γ curve is the intersection of $g = 0$ and $f = 0$.

emitting surface whose noise arrives at the observer at observer time t. This is shown in Fig. 2, which is taken from Reference [24].

Applying the free space Green's function to the FWH equation results in the following integral form for acoustic pressure. Equation 11 involves integrals over space and time up to the observer time; the next steps will take advantage of the $\delta(x)$ functions to limit those integrals.

$$4\pi p'_m(\boldsymbol{x}_i, t) = \frac{\partial}{\partial t} \int_{\infty}^{t} \int_{-\infty}^{\infty} \frac{Q}{r} \delta(f)\delta(g) \mathrm{d}\boldsymbol{y}_i \mathrm{d}\tau \tag{11a}$$

$$4\pi p'_d(\boldsymbol{x}_i, t) = -\frac{\partial}{\partial \boldsymbol{x}_i} \int_{\infty}^{t} \int_{-\infty}^{\infty} \frac{\boldsymbol{L}_i}{r} \delta(f)\delta(g) \mathrm{d}\boldsymbol{y}_i \mathrm{d}\tau \tag{11b}$$

2.1.2 Change in Variables with Generalized Functions

The first step in this process is to replace the gradient in the dipole term with an observer time derivative that is more amenable to numerical applications. This is done by using the identity in Eq. 12 and applying it to Eq. 11.

$$\frac{\partial}{\partial \boldsymbol{x}_i}\left(\frac{\delta(g)}{r}\right) = -\frac{1}{c_\infty}\frac{\partial}{\partial t}\left(\frac{\hat{\boldsymbol{r}}_i \delta(g)}{r}\right) - \frac{\hat{\boldsymbol{r}}_i \delta(g)}{r^2} \tag{12}$$

This results in the following equation for the monopole and dipole acoustic pressure. Many steps are not shown in this derivation but can be found in References [5] and [23]. The result of applying

the identity in Eq. 12 to Eq. 11 is shown in Eq. 13.

$$4\pi p'_m(\boldsymbol{x}_i,t) = \frac{\partial}{\partial t}\int_\infty^t\int_{-\infty}^\infty \frac{Q}{r}\delta(f)\delta(g)\mathrm{d}\boldsymbol{y}_i\mathrm{d}\tau \tag{13a}$$

$$4\pi p'_d(\boldsymbol{x}_i,t) = \frac{1}{c_\infty}\frac{\partial}{\partial t}\int_\infty^t\int_{-\infty}^\infty \frac{\boldsymbol{L}_i\hat{\boldsymbol{r}}_i}{r}\delta(f)\delta(g)\mathrm{d}\boldsymbol{y}_i\mathrm{d}\tau + \int_\infty^t\int_{-\infty}^\infty \frac{\boldsymbol{L}_i\hat{\boldsymbol{r}}_i}{r^2}\delta(f)\delta(g)\mathrm{d}\boldsymbol{y}_i\mathrm{d}\tau \tag{13b}$$

The source time integral can be addressed by changing variables from τ to g, first noting their relationship. Equation 14 shows the derivative of the collapsing sphere with respect to source time, which can be derived using Eqs. 9 and 10 and noting that the surface velocity, $\boldsymbol{v}_i$, is defined as $\partial\boldsymbol{y}_i/\partial\tau$.

$$\mathrm{d}g = |1 - M_r|\mathrm{d}\tau \tag{14}$$

Applying this to Eq. 11 and integrating over g results in an equation for acoustic pressure with only a spatial indefinite integral. This is shown in Eq. 15, which introduces the source time subscript, denoting that the quantity is evaluated at the source time, where τ is fixed so that $g = 0$ (or $\tau = t - r/c_\infty$).

$$4\pi p'_m(\boldsymbol{x}_i,t) = \frac{\partial}{\partial t}\int_{-\infty}^\infty \left[\frac{Q}{r|1-M_r|}\right]_{\mathrm{ret}}\delta(f)\mathrm{d}\boldsymbol{y}_i \tag{15a}$$

$$4\pi p'_d(\boldsymbol{x}_i,t) = \frac{1}{c_\infty}\frac{\partial}{\partial t}\int_{-\infty}^\infty \left[\frac{\boldsymbol{L}_j\hat{\boldsymbol{r}}_j}{r|1-M_r|}\right]_{\mathrm{ret}}\delta(f)\mathrm{d}\boldsymbol{y}_i + \int_{-\infty}^\infty \left[\frac{\boldsymbol{L}_j\hat{\boldsymbol{r}}_j}{r^2|1-M_r|}\right]_{\mathrm{ret}}\delta(f)\mathrm{d}\boldsymbol{y}_i \tag{15b}$$

The remaining indefinite integral can be addressed by, again, a changing of variables, this time from $\boldsymbol{y}_i$ to f. This is done by defining a new coordinate system aligned with the surface's normal vector, $\hat{\boldsymbol{n}}_i$, and introducing the Jacobian surface area, J, and time independent surface coordinates, u_1 and u_2, shown in Eq. 16.

$$\mathrm{d}\boldsymbol{y}_i = J\mathrm{d}u_1\mathrm{d}u_2\mathrm{d}f \tag{16}$$

Applying the spatial change in variables allows the integration over space to be limited to just the surface defined by $f = 0$. This is shown in Eq. 17.

$$4\pi p'_m(\boldsymbol{x}_i,t) = \frac{\partial}{\partial t}\int_{f=0}\left[\frac{QJ}{r|1-M_r|}\right]_{\mathrm{ret}}\mathrm{d}u_1\mathrm{d}u_2 \tag{17a}$$

$$4\pi p'_d(\boldsymbol{x}_i,t) = \frac{1}{c_\infty}\frac{\partial}{\partial t}\int_{f=0}\left[\frac{\boldsymbol{L}_j\hat{\boldsymbol{r}}_jJ}{r|1-M_r|}\right]_{\mathrm{ret}}\mathrm{d}u_1\mathrm{d}u_2 + \int_{f=0}\left[\frac{\boldsymbol{L}_j\hat{\boldsymbol{r}}_jJ}{r^2|1-M_r|}\right]_{\mathrm{ret}}\mathrm{d}u_1\mathrm{d}u_2 \tag{17b}$$

At this time, the shorthand notation introduced by Lee et al. in Reference [10] and shown in Eq. 4 will be used. The result of applying the shorthand notation is shown in Eq. 18.

$$4\pi p'_m(\boldsymbol{x}_i,t) = \frac{\partial}{\partial t}\int_{f=0}\Big[QAJ\Big]_{\mathrm{ret}}\mathrm{d}u_1\mathrm{d}u_2 \tag{18a}$$

$$4\pi p'_d(\boldsymbol{x}_i,t) = \frac{\partial}{\partial t}\int_{f=0}\Big[\boldsymbol{L}_j\boldsymbol{A}_jJ\Big]_{\mathrm{ret}}\mathrm{d}u_1\mathrm{d}u_2 + \int_{f=0}\Big[\boldsymbol{L}_j\boldsymbol{B}_jJ\Big]_{\mathrm{ret}}\mathrm{d}u_1\mathrm{d}u_2 \tag{18b}$$

where the following propagation coefficients have been defined to allow for further shorthand notation.

$$A = R(1,1), \qquad \boldsymbol{A}_j = c_\infty^{-1} R(1,1)\hat{\boldsymbol{r}}_j, \qquad \boldsymbol{B}_j = R(2,1)\hat{\boldsymbol{r}}_j \tag{19}$$

2.1.3 Compact Formulations

The compact assumption can potentially reduce a costly evaluation of a surface integral into a very efficient evaluation of a line integral. Two assumptions are made: 1) the observer position is very far away with respect to the characteristic size of the surface, L, and 2) the time it takes the collapsing sphere to cross the surface, referred to as $\Delta\tau$, is much larger than the typical period of surface pressure fluctuations, $\tilde{T}$. These are shown in Eq. 20.

$$r_{\min} \gg L \tag{20a}$$

$$\Delta\tau \gg \tilde{T} \tag{20b}$$

The compact assumption breaks the first and third constraints outlined in Section 1; therefore, any application of the compact assumptions is a compact form of that existing Formulation, not a new Formulation in and of itself. The monopole term is accounted for first since that requires more mathematical operations than the dipole term [11]. Following a derivation from Reference [6] and assuming the surface is impermeable, the noise from the monopole term is the solution of Eq. 21.

$$\bar{\Box}^2 p'_m(\boldsymbol{x}_i, t) = \rho_\infty \frac{\partial}{\partial t}\left[v_n \delta(f)\right] \tag{21}$$

The source surface can be written as a volume of sources by recognizing that the rate of change of the volume enclosed by the surface is relatable to the velocity of the surface, shown by Eq. 22 [25].

$$v_n \delta(f) = \frac{\partial}{\partial t}\left[1 - H(f)\right] \tag{22}$$

Therefore, the monopole term is proportional to the second derivative with respect to observer time of the volume encapsulated by the surface (proportional to the acceleration of the surface). This is shown in Eq. 23.

$$\bar{\Box}^2 p'_m(\boldsymbol{x}_i, t) = \rho_\infty \frac{\partial^2}{\partial t^2}\left[1 - H(f)\right] \tag{23}$$

Using a free space Green's function as above, the acoustic pressure at the observer location is a function of the volume encapsulated by the source surface (Σ surface) and the distance from the volume to the observer location. Assuming an elongated body, such as a rotor blade, and applying a geometrically compact assumption ($r_{min} \gg$ blade chord), the monopole term can be reduced to an integration along a line of the cross sectional area, Ψ, where dζ is the differential length of the

compact line placed at the quarter chord defined by $f = 0$.[2]

$$4\pi p'_m(\boldsymbol{x}_i, t) = \rho_\infty \frac{\partial^2}{\partial t^2} \int_{f<0} \left[\mathscr{A} \right]_{\text{ret}} \mathrm{d}\boldsymbol{y}_i \approx \rho_\infty \frac{\partial^2}{\partial t^2} \int_{f=0} \left[\Psi \mathscr{A} \right]_{\text{ret}} \mathrm{d}\zeta = \rho_\infty \frac{\partial^2}{\partial t^2} \int_{f=0} \left[\Psi \mathscr{A} K \right]_{\text{ret}} \mathrm{d}u \quad (24)$$

where

$$\mathscr{A} = R(1,1)$$

We introduce new math script coefficents that parallel the propagation coefficients in the noncompact forms.

Using the assumptions outlined in Eq. 20 on the dipole term allows integration in the chord direction of the surface pressure. This integration changes a blade surface pressure distribution into a line of lifting sources, $\boldsymbol{F}_i$. Applying the compact assumption to the dipole term results in the compact form of the dipole equation, which is shown in Eq. 25.

$$4\pi c_\infty p'_d(\boldsymbol{x}_i, t) = \frac{\partial}{\partial t} \int_{f=0} \left[\boldsymbol{F}_i \mathscr{B}_i K \right]_{\text{ret}} \mathrm{d}u + \int_{f=0} \left[\boldsymbol{F}_i \mathscr{C}_i K \right]_{\text{ret}} \mathrm{d}u \quad (25)$$

where

$$\mathscr{B}_i = R(1,1)\hat{\boldsymbol{r}}_i, \qquad \mathscr{C}_i = c_\infty R(2,1)\hat{\boldsymbol{r}}_i$$

2.1.4 Grouping Integrals

Before continuing to the Formulations, an explanation of the grouping of terms is required to communicate what is available from AFFIFMs. One of the main mathematical operations employed when deriving the Formulations is bringing the observer time derivative from the outside of the surface integration to the inside. Several steps involved in the derivation of taking the observer time derivative inside the surface integration are not shown here. Applying this mathematical operation on the monopole equation in Eq. 26 is presented here as an example.

$$\frac{\partial}{\partial t} \int_{f=0} \left[QAJ \right]_{\text{ret}} \mathrm{d}u_1 \mathrm{d}u_2 = \int_{f=0} \left[R(0,1) \frac{\partial}{\partial \tau} \left\{ QAJ \right\} \right]_{\text{ret}} \mathrm{d}u_1 \mathrm{d}u_2 \quad (26)$$

After a simple chain rule, three terms emerge, each with a different component of the integrand with a source time derivative. Since this operation is one of the steps to go from one Formulation to another, subscripts are used to identify which Formulation the coefficient corresponds to. This is shown in Eq. 27.

$$\begin{aligned} &\int_{f=0} \left[R(0,1) \frac{\partial}{\partial \tau} \left\{ QA_1J \right\} \right]_{\text{ret}} \mathrm{d}u_1 \mathrm{d}u_2 = \\ &\qquad \int_{f=0} \left[\dot{Q} A_{1A} J \right]_{\text{ret}} \mathrm{d}u_1 \mathrm{d}u_2 + \int_{f=0} \left[Q B_{1A} J \right]_{\text{ret}} \mathrm{d}u_1 \mathrm{d}u_2 + \int_{f=0} \left[Q A_{1A} \dot{J} \right]_{\text{ret}} \mathrm{d}u_1 \mathrm{d}u_2 \end{aligned} \quad (27)$$

[2]The quarter chord line, where the aerodynamic center of subsonic airfoils is located, is convenient due to any accompanying reduced order blade loading calculations; however, the geometric center of the airfoil would be a more accurate location for the compact line [11].

where

$$A_{1A} = R(0,1)A_1, \qquad B_{1A} = R(0,1)\dot{A}_1$$

The above example shows one of the operations to derive Formulation 1A from Formulation 1. In AF1AIFM (and some other AFFIFMs), these terms may be returned to the user as part of a group. This is shown in Eq. 28 and can be grouped by *COMPONENT*, *GEOMETRY*, and *COEFFICIENTS*.

$$\textit{COMPONENT}: \quad \int_{f=0}\left[\dot{Q}\{A_{1A}J\}\right]_{\text{ret}} \mathrm{d}u_1\mathrm{d}u_2 + \int_{f=0}\left[Q\{B_{1A}J + A_{1A}\dot{J}\}\right]_{\text{ret}} \mathrm{d}u_1\mathrm{d}u_2 \tag{28a}$$

$$\textit{GEOMETRY}: \quad \int_{f=0}\left[J\{\dot{Q}A_{1A} + QB_{1A}\}\right]_{\text{ret}} \mathrm{d}u_1\mathrm{d}u_2 + \int_{f=0}\left[\dot{J}\{QA_{1A}\}\right]_{\text{ret}} \mathrm{d}u_1\mathrm{d}u_2 \tag{28b}$$

$$\textit{COEFFICIENTS}: \quad \int_{f=0}\left[A_{1A}\{\dot{Q}J + Q\dot{J}\}\right]_{\text{ret}} \mathrm{d}u_1\mathrm{d}u_2 + \int_{f=0}\left[B_{1A}\{QJ\}\right]_{\text{ret}} \mathrm{d}u_1\mathrm{d}u_2 \tag{28c}$$

This type of grouping cannot occur for all Formulations, particularly when there are observer time derivatives still employed. Also, in some Formulations, there is an additional option to group by *FIELD*, which groups the terms into far field $(1/r)$, near field $(1/r)^2$, and, sometimes, ultra near field $(1/r^3)$ integrals. When grouping by *FIELD*, the coefficients, like those listed in Eq. 19, are not used, but are rather expanded as functions of $\boldsymbol{r}_i$ and M. In addition to the grouping of integrals, all AFFIFMs also provide the ability to return the sum of the monopole and dipole term, denoted as *TOTAL*, the sum of all monopole and dipole terms, respectively denoted as *COMPONENT*, and every integral by itself without summing, denoted as *INTEGRALS*. When using these options, the order of the results of AFFIFMs is the same order noted in the equations below, i.e., monopole then dipole or monopole integrals then dipole integrals. Since they are relatively straightforward, all of the options (with the exception of *FIELD*) are not explicitely described here, just that which would be returned by the *INTEGRALS* option.

2.1.5 Source Time Derivatives of Radiation Functions

The derivations shown below rely heavily on source time derivatives of complicated propagation functions and radiation vectors. Rather than show the Formulations with those derivatives taken out analytically, the propagation coefficients of a given Formulation may rely on the coefficients from other Formulations. It is left to the reader to rederive the analytical propagation functions, if necessary. This significantly simplifies the expressions shown here for the Formulations and allows the reader to understand the connection between them. To aid in the derivation of Formulations with the analytical derivatives expanded, this section contains some common analytical derivatives.

The first analytical source time derivative is that of the radiation distance, r.

$$\begin{aligned}\frac{\partial r^2}{\partial \tau} =& \frac{\partial (\boldsymbol{x}_i - \boldsymbol{y}_i)^2}{\partial \tau} \\ 2r\frac{\partial r}{\partial \tau} =& -2(\boldsymbol{x}_i - \boldsymbol{y}_i)\frac{\partial \boldsymbol{y}_i}{\partial \tau} \\ r\frac{\partial r}{\partial \tau} =& -\boldsymbol{r}_i \frac{\partial \boldsymbol{y}_i}{\partial \tau} \\ \frac{\partial r}{\partial \tau} =& -\boldsymbol{v}_i \hat{\boldsymbol{r}}_i \\ \dot{r} =& -\boldsymbol{v}_r\end{aligned} \tag{29}$$

The next is the source time derivative of the radiation vector, $\hat{\boldsymbol{r}}_i$.

$$\dot{\hat{\boldsymbol{r}}}_i = \frac{\partial \hat{\boldsymbol{r}}_i}{\partial \tau} = \frac{\partial}{\partial \tau}\left(\frac{\boldsymbol{r}_i}{r}\right) = -\frac{\boldsymbol{v}_i}{r} - \frac{\boldsymbol{r}_i}{r^2}\frac{\partial r}{\partial \tau} = -\frac{c_\infty}{r}(\boldsymbol{M}_i - M_r \hat{\boldsymbol{r}}_i) \tag{30}$$

Next is the analytical source time derivative of the Mach vector in the radiation direction, M_r.

$$\frac{\partial M_r}{\partial \tau} = \frac{\partial \boldsymbol{M}_i \boldsymbol{r}_i}{\partial \tau} = \boldsymbol{M}_i \frac{c_\infty}{r}(M_r \hat{\boldsymbol{r}}_i - \boldsymbol{M}_i) + \frac{\partial \boldsymbol{M}_i}{\partial \tau}\hat{\boldsymbol{r}}_i = \frac{c_\infty}{r}(M_r^2 - M^2) + \dot{M}_r \tag{31}$$

The last expression shown here is the source time derivative of the most common radiation function, $R(1,1)$.

$$\dot{R}(1,1) = \dot{M}_r R(1,2) + c_\infty (M_r - M^2) R(2,2) \tag{32}$$

This was generalized by Lee et al. to any powers, m_1 and m_2, of r and $|1 - M_r|$, respectively [10].

$$\begin{aligned}\dot{R}(m_1, m_2) =& m_2 \dot{M}_r R(m_1, m_2 + 1) + c_\infty m_2 (M_r - M^2) R(m_1 + 1, m_2 + 1) + \\ & c_\infty (m_1 - m_2) M_r R(m_1 + 1, m_2)\end{aligned} \tag{33}$$

2.2 Formulation 1

Farassat's Formulation 1 of the FWH equation is implemented in ANOPP2's Formulation 1 Internal Functional Module (AF1IFM). Below are the noncompact and compact forms of Formulation 1. Since Formulation 1 contains observer time derivatives outside the surface integration, it is not grouped by the source time derivatives or field terms as described above. Only *TOTAL*, *COMPONENT*, and *INTEGRALS* are available.

Noncompact Formulation 1

Since this is the first Formulation, the derivation for the noncompact form has already been shown above in Eq. 18. It is repeated here for reference.

$$\begin{aligned}4\pi p'_m(\boldsymbol{x}_i, t) =& \frac{\partial}{\partial t}\int_{f=0}\left[QA_1 J\right]_{\text{ret}} \mathrm{d}u_1 \mathrm{d}u_2 \\ 4\pi c_\infty p'_d(\boldsymbol{x}_i, t) =& \frac{\partial}{\partial t}\int_{f=0}\left[\boldsymbol{L}_j \boldsymbol{B}_{1,j} J\right]_{\text{ret}} \mathrm{d}u_1 \mathrm{d}u_2 + \int_{f=0}\left[\boldsymbol{L}_j \boldsymbol{C}_{1,j} J\right]_{\text{ret}} \mathrm{d}u_1 \mathrm{d}u_2\end{aligned} \tag{34}$$

where

$$A_1 = R(1,1), \qquad \boldsymbol{B}_{1,j} = R(1,1)\hat{\boldsymbol{r}}_j,$$
$$\boldsymbol{C}_{1,j} = c_\infty R(2,1)\hat{\boldsymbol{r}}_j$$

Compact Formulation 1

Applying the compact assumption and the technique employed above, the compact Formulation 1 contains a second observer time derivative for the monopole noise.

$$\begin{aligned}\frac{4\pi}{\rho_\infty}p'_m(\boldsymbol{x}_i,t) =& \frac{\partial^2}{\partial t^2}\int_{f=0}\left[\Psi\mathscr{A}_1 K\right]_{\text{ret}}\mathrm{d}u \\ 4\pi c_\infty p'_d(\boldsymbol{x}_i,t) =& \frac{\partial}{\partial t}\int_{f=0}\left[\boldsymbol{F}_i\mathscr{B}_{1,i}K\right]_{\text{ret}}\mathrm{d}u + \int_{f=0}\left[\boldsymbol{F}_i\mathscr{C}_{1,i}K\right]_{\text{ret}}\mathrm{d}u\end{aligned} \tag{35}$$

where

$$\mathscr{A}_1 = R(1,1), \qquad \mathscr{B}_{1,i} = R(1,1)\hat{\boldsymbol{r}}_i, \qquad \mathscr{C}_{1,i} = c_\infty R(2,1)\hat{\boldsymbol{r}}_i$$

2.3 Formulation 1A

Using the mathematical technique in Eq. 26, the observer time derivatives are turned into source time derivatives. This is Farassat's Formulation 1A of the FWH equation and is implemented in AF1AIFM. Since this is one of the most commonly used Formulations and includes no observer time derivatives, all options of grouping are available. This includes *TOTAL*, *SOURCE*, *COMPONENT*, *GEOMETRY*, *COEFFICIENTS*, and *FIELD*. In the below equations, only the three integral options (*COMPONENT*, *GEOMETRY*, *COEFFICIENTS*) and *FIELD* option is shown.

Noncompact Formulation 1A

The noncompact form of Formulation 1A, with all the integrals separated (i.e., *INTEGRALS*), is shown in Eq. 36.

$$\begin{aligned}4\pi p'_m(\boldsymbol{x}_i,t) =& \int_{f=0}\left[\dot{Q}A_{1A}J\right]_{\text{ret}}\mathrm{d}u_1\mathrm{d}u_2 + \int_{f=0}\left[QA_{1A}\dot{J}\right]_{\text{ret}}\mathrm{d}u_1\mathrm{d}u_2 + \\ & \int_{f=0}\left[QB_{1A}J\right]_{\text{ret}}\mathrm{d}u_1\mathrm{d}u_2 \\ 4\pi c_\infty p'_d(\boldsymbol{x}_i,t) =& \int_{f=0}\left[\dot{\boldsymbol{L}}_j\boldsymbol{C}_{1A,j}J\right]_{\text{ret}}\mathrm{d}u_1\mathrm{d}u_2 + \int_{f=0}\left[\boldsymbol{L}_j\boldsymbol{C}_{1A,j}\dot{J}\right]_{\text{ret}}\mathrm{d}u_1\mathrm{d}u_2 + \\ & \int_{f=0}\left[\boldsymbol{L}_j\boldsymbol{D}_{1A,j}J\right]_{\text{ret}}\mathrm{d}u_1\mathrm{d}u_2\end{aligned} \tag{36}$$

where

$$A_{1A} = R(0,1)A_1, \qquad B_{1A} = R(0,1)\dot{A}_1$$
$$\boldsymbol{C}_{1A,j} = R(0,1)B_{1,j}, \qquad \boldsymbol{D}_{1A,j} = R(0,1)\dot{B}_{1,j} + \boldsymbol{C}_{1,j}$$

When grouping by *FIELD*, the propagation coefficients are replaced by equations of $\boldsymbol{r}_i$ and M. The integrals are grouped into near field and far field integrals for thickness and loading noise. This is shown in Eq. 37.

$$
\begin{aligned}
4\pi p'_m(\boldsymbol{x}_i,t) &= \int_{f=0}\left[\{\dot{Q}J+Q\dot{J}\}\mathcal{A}_{1A}+QJ\dot{M}_r\mathcal{B}_{1A}\right]_{\text{ret}}\mathrm{d}u_1\mathrm{d}u_2+c_\infty\int_{f=0}\left[QJ\mathcal{C}_{1A}\right]_{\text{ret}}\mathrm{d}u_1\mathrm{d}u_2\\
4\pi c_\infty p'_d(\boldsymbol{x}_i,t) &= \int_{f=0}\left[\{\dot{\boldsymbol{L}}_jJ+\boldsymbol{L}_j\dot{J}\}\mathcal{D}_{1A,j}+\boldsymbol{L}_jJ\mathcal{E}_{1A,j}\right]_{\text{ret}}\mathrm{d}u_1\mathrm{d}u_2+\\
&\qquad c_\infty\int_{f=0}\left[\boldsymbol{L}_iJ\mathcal{F}_{1A,j}\right]_{\text{ret}}\mathrm{d}u_1\mathrm{d}u_2
\end{aligned}
\tag{37}
$$

where

$$
\begin{gathered}
\mathcal{A}_{1A}=R(1,2),\qquad \mathcal{B}_{1A}=R(1,3),\qquad \mathcal{C}_{1A}=c_\infty(M_r-M^2)R(2,3)\\
\mathcal{D}_{1A,j}=R(1,2)\hat{\boldsymbol{r}}_j,\qquad \mathcal{E}_{1A,j}=\dot{M}_rR(1,3)\hat{\boldsymbol{r}}_j\\
\mathcal{F}_{1A,j}=\hat{\boldsymbol{r}}_iR(2,1)+(\hat{\boldsymbol{r}}_i-\boldsymbol{M}_i)R(2,2)+(M_r-M^2)\hat{\boldsymbol{r}}_iR(2,3)
\end{gathered}
$$

Compact Formulation 1A

The compact form of Formulation 1A follows the same technique as Formulation 1 when applying the compact assumption except this time there is a second-order observer time derivative that needs to be brought inside of the line integral. This can be done one observer time derivative at a time, similar to the derivation of Formulation G1 (see below), but this was avoided as the delineation of this intermediary Formulation is unnecessary.

$$
\begin{aligned}
\frac{4\pi}{\rho_\infty}p'_m(\boldsymbol{x}_i,t) &= \int_{\digamma=0}\left[\ddot{\Psi}\mathscr{A}_{1A}K\right]_{\text{ret}}\mathrm{d}u+\int_{\digamma=0}\left[2\dot{\Psi}\mathscr{A}_{1A}\dot{K}\right]_{\text{ret}}\mathrm{d}u+\int_{\digamma=0}\left[\Psi\mathscr{A}_{1A}\ddot{K}\right]_{\text{ret}}\mathrm{d}u+\\
&\qquad \int_{\digamma=0}\left[\dot{\Psi}\mathscr{B}_{1A}K\right]_{\text{ret}}\mathrm{d}u+\int_{\digamma=0}\left[\Psi\mathscr{B}_{1A}\dot{K}\right]_{\text{ret}}\mathrm{d}u+\int_{\digamma=0}\left[\Psi\mathscr{C}_{1A}K\right]_{\text{ret}}\mathrm{d}u\\
4\pi c_\infty p'_d(\boldsymbol{x}_i,t) &= \int_{\digamma=0}\left[\dot{\boldsymbol{F}}_i\mathscr{D}_{1A,i}K\right]_{\text{ret}}\mathrm{d}u+\int_{\digamma=0}\left[\boldsymbol{F}_i\mathscr{D}_{1A,i}\dot{K}\right]_{\text{ret}}\mathrm{d}u+\int_{\digamma=0}\left[\boldsymbol{F}_i\mathscr{E}_{1A,i}K\right]_{\text{ret}}\mathrm{d}u
\end{aligned}
\tag{38}
$$

where

$$
\begin{gathered}
\mathscr{A}_{1A}=R(0,2)\mathscr{A}_1,\qquad \mathscr{B}_{1A}=2R(0,2)\dot{\mathscr{A}}_1+R(0,1)\dot{R}(0,1)\mathscr{A}_1\\
\mathscr{C}_{1A}=R(0,2)\ddot{\mathscr{A}}_1+R(0,1)\dot{R}(0,1)\dot{\mathscr{A}}_1\\
\mathscr{D}_{1A,i}=R(0,1)\mathscr{B}_{1,i},\qquad \mathscr{E}_{1A,i}=R(0,1)\dot{\mathscr{B}}_{1,i}+\mathscr{C}_{1,i}
\end{gathered}
$$

When grouping by *FIELD*, the propagation coefficients are replaced by equations of $\boldsymbol{r}_i$ and M. The integrals are grouped into near field and far field integrals for thickness and loading noise. This is

shown in Eq. 39.

$$\begin{aligned}
\frac{4\pi}{\rho_\infty}p'_m(\boldsymbol{x}_i,t) &= \int_{F=0}\left[\{\ddot{\Psi}K+2\dot{\Psi}\dot{K}+\Psi\ddot{K}\}\mathbb{A}_{1A}+\{\dot{\Psi}K+\Psi\dot{K}\}\mathbb{B}_{1A}+\Psi K\mathbb{C}_{1A}\right]_{\text{ret}}\mathrm{d}u+\\
&\quad\int_{F=0}\left[\{\ddot{\Psi}K+\Psi\dot{K}\}\mathbb{D}_{1A}+\Psi K\mathbb{E}_{1A}\right]_{\text{ret}}\mathrm{d}u+\int_{F=0}\left[\Psi K\mathbb{F}_{1A}\right]_{\text{ret}}\mathrm{d}u\\
4\pi c_\infty p'_d(\boldsymbol{x}_i,t) &= \int_{F=0}\left[\{\dot{\boldsymbol{F}}_jK+\boldsymbol{F}_j\dot{K}\}\mathbb{G}_{1A,j}+\boldsymbol{F}_jK\mathbb{H}_{1A,j}\right]_{\text{ret}}\mathrm{d}u+\\
&\quad c_\infty\int_{F=0}\left[\boldsymbol{F}_jK\mathbb{I}_{1A,j}\right]_{\text{ret}}\mathrm{d}u
\end{aligned}\tag{39}$$

where

$$\mathbb{A}_{1A}=R(1,3),\qquad \mathbb{B}_{1A}=2\dot{M}_rR(1,4),\qquad \mathbb{C}_{1A}=\ddot{M}_rR(1,4)+3\dot{M}_r^2R(1,5)$$
$$\mathbb{D}_{1A}=c_\infty(M_r+M_r^2-2M^2)R(2,4)$$
$$\mathbb{E}_{1A}=c_\infty\dot{M}_rR(2,3)+c_\infty(\dot{M}_r-3M_i\dot{M}_i)R(2,4)+c_\infty\dot{M}_r(5(M_r-M^2)+(M_r^2-M^2))R(2,5)$$
$$\mathbb{F}_{1A}=c_\infty^2(M_r^2-M^2)R(3,4)+c_\infty^2(2(M_r-M^2)^2+(M_r-M^2)(M_r^2-M^2))R(3,5)$$
$$\mathbb{G}_{1A,j}=R(1,2)\hat{\boldsymbol{r}}_j,\qquad \mathbb{H}_{1A,j}=\dot{M}_rR(1,3)\hat{\boldsymbol{r}}_j$$
$$\mathbb{I}_{1A,j}=\hat{\boldsymbol{r}}_iR(2,1)+(\hat{\boldsymbol{r}}_i-\boldsymbol{M}_i)R(2,2)+(M_r-M^2)\hat{\boldsymbol{r}}_iR(2,3)$$

2.4 Formulation G0

Pressure gradient formulations are used in conjunction with acoustic scattering algorithms, such as the Fast Scattering Code (FSC) [26], to account for scattering and shielding of noise caused by an object, such as an aircraft in proximity to the sound source. Pressure gradient formulations provide the incident field on the scattering body and at the observer, while the scattering codes provide the scattered field [10]. The sum of the incident and scattered fields is then the total noise at the observer. There are currently three pressure gradient Formulations implemented in ANOPP2: Formulations G0, G1, and G1A. Formulation G0, first introduced by Lopes [11], is derived by applying a spatial gradient to both sides of Eq. 11 and then using Eq. 12 to turn the spatial gradient into a time derivative.

Noncompact Formulation G0

Equation 40 shows Formulation G0 and is implemented in ANOPP2 via ANOPP2's Formulation G0 Internal Functional Module (AFG0IFM). Since Formulation G0 contains observer time derivatives outside the surface integration, AFG0IFM is only able to group on *TOTAL*, *COMPONENT*, and *INTEGRALS*.

$$\begin{aligned}
4\pi c_\infty\frac{\partial p'_m}{\partial \boldsymbol{x}_i}(\boldsymbol{x}_i,t) &= -\frac{\partial^2}{\partial t^2}\int_{f=0}\left[Q\boldsymbol{A}_{G0,i}J\right]_{\text{ret}}\mathrm{d}u_1\mathrm{d}u_2-\frac{\partial}{\partial t}\int_{f=0}\left[Q\boldsymbol{B}_{G0,i}J\right]_{\text{ret}}\mathrm{d}u_1\mathrm{d}u_2\\
4\pi c_\infty^2\frac{\partial p'_d}{\partial \boldsymbol{x}_i}(\boldsymbol{x}_i,t) &= -\frac{\partial^2}{\partial t^2}\int_{f=0}\left[\boldsymbol{L}_j\boldsymbol{C}_{G0,ij}J\right]_{\text{ret}}\mathrm{d}u_1\mathrm{d}u_2+\frac{\partial}{\partial t}\int_{f=0}\left[\boldsymbol{L}_j\boldsymbol{D}_{G0,ij}J\right]_{\text{ret}}\mathrm{d}u_1\mathrm{d}u_2+\\
&\quad\int_{f=0}\left[\boldsymbol{L}_j\boldsymbol{E}_{G0,ij}J\right]_{\text{ret}}\mathrm{d}u_1\mathrm{d}u_2
\end{aligned}\tag{40}$$

where

$$\boldsymbol{A}_{G0,i} = R(1,1)\hat{\boldsymbol{r}}_i, \qquad \boldsymbol{B}_{G0,i} = c_\infty R(2,1)\hat{\boldsymbol{r}}_i$$

$$\boldsymbol{C}_{G0,ij} = R(1,1)\hat{\boldsymbol{r}}_i\hat{\boldsymbol{r}}_j, \qquad \boldsymbol{D}_{G0,ij} = c_\infty R(2,1)(\boldsymbol{\delta}_{ij} - 3\hat{\boldsymbol{r}}_i\hat{\boldsymbol{r}}_j), \qquad \boldsymbol{E}_{G0,ij} = c_\infty^2 R(3,1)(\boldsymbol{\delta}_{ij} - 3\hat{\boldsymbol{r}}_i\hat{\boldsymbol{r}}_j)$$

Compact Formulation G0

The compact form of Formulation G0 is found by following the steps outlined in the above section; which is shown in Eq. 41. Similar to the noncompact form of Formulation G0, only *TOTAL*, *COMPONENT*, and *INTEGRAL* are available for grouping options.

$$\begin{aligned}
4\pi\frac{c_\infty}{\rho_\infty}\frac{\partial p'_m}{\partial \boldsymbol{x}_i}(\boldsymbol{x}_i,t) =& -\frac{\partial^3}{\partial t^3}\int_{F=0}\left[\Psi\mathscr{A}_{G0,i}K\right]_{\text{ret}}\mathrm{d}u - \frac{\partial^2}{\partial t^2}\int_{F=0}\left[\Psi\mathscr{B}_{G0,i}K\right]_{\text{ret}}\mathrm{d}u \\
4\pi c_\infty^2\frac{\partial p'_d}{\partial \boldsymbol{x}_i}(\boldsymbol{x}_i,t) =& -\frac{\partial^2}{\partial t^2}\int_{F=0}\left[\boldsymbol{F}_j\mathscr{C}_{G0,ij}K\right]_{\text{ret}}\mathrm{d}u + \frac{\partial}{\partial t}\int_{F=0}\left[\boldsymbol{F}_j\mathscr{D}_{G0,ij}K\right]_{\text{ret}}\mathrm{d}u + \\
& \int_{F=0}\left[\boldsymbol{F}_j\mathscr{E}_{G0,ij}K\right]_{\text{ret}}\mathrm{d}u
\end{aligned} \tag{41}$$

where

$$\mathscr{A}_{G0,i} = R(1,1)\hat{\boldsymbol{r}}_i, \qquad \mathscr{B}_{G0,i} = c_\infty R(2,1)\hat{\boldsymbol{r}}_i$$

$$\mathscr{C}_{G0,ij} = R(1,1)\hat{\boldsymbol{r}}_i\hat{\boldsymbol{r}}_j, \qquad \mathscr{D}_{G0,ij} = c_\infty R(2,1)(\boldsymbol{\delta}_{ij} - 3\hat{\boldsymbol{r}}_i\hat{\boldsymbol{r}}_j), \qquad \mathscr{E}_{G0,ij} = c_\infty^2 R(3,1)(\boldsymbol{\delta}_{ij} - 3\hat{\boldsymbol{r}}_i\hat{\boldsymbol{r}}_j)$$

2.5 Formulation G1

Formulation G1 was first introduced by Lee [10] and is an intermediary step between Formulations G0 and G1A. The intermediary step is important because Formulation V1A can be derived from G1A, and because the propagation coefficients between Formulations G1 and V1A are very similar. Formulation G1 is implemented in ANOPP2 in ANOPP2's Formulation G1 Internal Functional Module (AFG1IFM). Since Formulation G1 contains observer time derivatives outside the surface integration, AFG1IFM is only able to group on *TOTAL*, *COMPONENT*, and *INTEGRALS*.

Noncompact Formulation G1

The noncompact form of Formulation G1 is shown in Eq. 42 and is implemented in AFG1IFM.

$$\begin{aligned}
4\pi c_\infty\frac{\partial p'_m}{\partial \boldsymbol{x}_i}(\boldsymbol{x}_i,t) =& -\frac{\partial}{\partial t}\int_{f=0}\left[\dot{Q}\boldsymbol{A}_{G1,i}J\right]_{\text{ret}}\mathrm{d}u_1\mathrm{d}u_2 - \frac{\partial}{\partial t}\int_{f=0}\left[Q\boldsymbol{B}_{G1,i}J\right]_{\text{ret}}\mathrm{d}u_1\mathrm{d}u_2 \\
& - \frac{\partial}{\partial t}\int_{f=0}\left[Q\boldsymbol{A}_{G1,i}\dot{J}\right]_{\text{ret}}\mathrm{d}u_1\mathrm{d}u_2 \\
4\pi c_\infty^2\frac{\partial p'_d}{\partial \boldsymbol{x}_i}(\boldsymbol{x}_i,t) =& -\frac{\partial}{\partial t}\int_{f=0}\left[\dot{\boldsymbol{L}}_j\boldsymbol{C}_{G1,ij}J\right]_{\text{ret}}\mathrm{d}u_1\mathrm{d}u_2 - \frac{\partial}{\partial t}\int_{f=0}\left[\boldsymbol{L}_j\boldsymbol{D}_{G1,ij}J\right]_{\text{ret}}\mathrm{d}u_1\mathrm{d}u_2 \\
& - \frac{\partial}{\partial t}\int_{f=0}\left[\boldsymbol{L}_j\boldsymbol{C}_{G1,ij}\dot{J}\right]_{\text{ret}}\mathrm{d}u_1\mathrm{d}u_2 + \int_{f=0}\left[\boldsymbol{L}_j\boldsymbol{E}_{G1,ij}J\right]_{\text{ret}}\mathrm{d}u_1\mathrm{d}u_2
\end{aligned} \tag{42}$$

where

$$\boldsymbol{A}_{G1,i} = R(0,1)\boldsymbol{A}_{G0,i}, \qquad \boldsymbol{B}_{G1,i} = R(0,1)\dot{\boldsymbol{A}}_{G0,i} + \boldsymbol{B}_{G0,i}$$
$$\boldsymbol{C}_{G1,ij} = R(0,1)\boldsymbol{C}_{G0,ij}, \qquad \boldsymbol{D}_{G1,ij} = R(0,1)\dot{\boldsymbol{C}}_{G0,ij} - \boldsymbol{D}_{G0,ij}, \qquad \boldsymbol{E}_{G1,ij} = \boldsymbol{E}_{G0,ij}$$

Compact Formulation G1

The compact form of Formulation G1 is shown in Eq. 43 and is implemented in AFG1IFM.

$$\begin{aligned}
4\pi\frac{c_\infty}{\rho_\infty}\frac{\partial p'_m}{\partial \boldsymbol{x}_i}(\boldsymbol{x}_i,t) =& -\frac{\partial}{\partial t}\int_{f=0}\left[\ddot{\Psi}\mathscr{A}_{G1,i}K\right]_{\text{ret}}\mathrm{d}u - \frac{\partial}{\partial t}\int_{f=0}\left[\dot{\Psi}\mathscr{B}_{G1,i}K\right]_{\text{ret}}\mathrm{d}u \\
&-\frac{\partial}{\partial t}\int_{f=0}\left[2\dot{\Psi}\mathscr{A}_{G1,i}\dot{K}\right]_{\text{ret}}\mathrm{d}u - \frac{\partial}{\partial t}\int_{f=0}\left[\Psi\mathscr{A}_{G1,i}\ddot{K}\right]_{\text{ret}}\mathrm{d}u \\
&-\frac{\partial}{\partial t}\int_{f=0}\left[\Psi\mathscr{B}_{G1,i}\dot{K}\right]_{\text{ret}}\mathrm{d}u - \frac{\partial}{\partial t}\int_{f=0}\left[\Psi\mathscr{C}_{G1,i}K\right]_{\text{ret}}\mathrm{d}u \qquad (43)\\
4\pi c_\infty^2\frac{\partial p'_d}{\partial \boldsymbol{x}_i}(\boldsymbol{x}_i,t) =& -\frac{\partial}{\partial t}\int_{f=0}\left[\dot{\boldsymbol{F}}_j\mathscr{D}_{G1,ij}K\right]_{\text{ret}}\mathrm{d}u - \frac{\partial}{\partial t}\int_{f=0}\left[\boldsymbol{F}_j\mathscr{E}_{G1,ij}K\right]_{\text{ret}}\mathrm{d}u \\
&-\frac{\partial}{\partial t}\int_{f=0}\left[\boldsymbol{F}_j\mathscr{D}_{G1,ij}\dot{K}\right]_{\text{ret}}\mathrm{d}u + \int_{f=0}\left[\boldsymbol{F}_j K_{G1,ij}\right]_{\text{ret}}\mathrm{d}u
\end{aligned}$$

where

$$\mathscr{A}_{G1,i} = R(0,2)\mathscr{A}_{G0,i}, \qquad \mathscr{B}_{G1,i} = R(0,1)\dot{R}(0,1)\mathscr{A}_{G0,i} + R(0,1)\mathscr{B}_{G0,i}$$
$$\mathscr{C}_{G1,i} = R(0,2)\ddot{\mathscr{A}}_{G0,i} + R(0,1)\dot{R}(0,1)\dot{\mathscr{A}}_{G0,i} + R(0,1)\dot{\mathscr{B}}_{G0,i}$$
$$\mathscr{D}_{G1,ij} = R(0,1)\mathscr{C}_{G0,ij}, \qquad \mathscr{E}_{G1,ij} = R(0,1)\dot{\mathscr{C}}_{G0,ij} - \mathscr{D}_{G0,ij}, \qquad \mathscr{F}_{G1,ij} = \mathscr{E}_{G0,ij}$$

2.6 Formulation G1A

Formulation G1A is the final form of the pressure gradient formulation. It requires a second observer time derivative (for the noncompact) or a third-order observer time derivative (for the compact) to be brought inside the integral and changed into a source time derivative. As a result, the Formulations involve a large number of terms. Since these Formulations include no observer time derivatives outside the integrals, all options of grouping are available. This includes *TOTAL*, *SOURCE*, *COMPONENT*, *GEOMETRY*, *COEFFICIENTS* and *FIELD*.

Noncompact Formulation G1A

Equation 44 shows the noncompact form of Formulation G1A. This was derived by taking Eq. 42 and using Eq. 26 to change the observer time derivatives into source time derivatives and bringing

them into the line integral.

$$
\begin{aligned}
4\pi c_\infty \frac{\partial p'_m}{\partial \boldsymbol{x}_i}(\boldsymbol{x}_i,t) = &-\int_{f=0}\left[\ddot{Q}J\boldsymbol{A}_{G1A,i}J\right]_{\text{ret}}\mathrm{d}u_1\mathrm{d}u_2-\int_{f=0}\left[2\dot{Q}\boldsymbol{A}_{G1A,i}\dot{J}\right]_{\text{ret}}\mathrm{d}u_1\mathrm{d}u_2\\
&-\int_{f=0}\left[\dot{Q}\boldsymbol{B}_{G1A,i}J\right]_{\text{ret}}\mathrm{d}u_1\mathrm{d}u_2-\int_{f=0}\left[Q\boldsymbol{A}_{G1A,i}\ddot{J}\right]_{\text{ret}}\mathrm{d}u_1\mathrm{d}u_2\\
&-\int_{f=0}\left[Q\boldsymbol{B}_{G1A,i}\dot{J}\right]_{\text{ret}}\mathrm{d}u_1\mathrm{d}u_2-\int_{f=0}\left[Q\boldsymbol{C}_{G1A,i}J\right]_{\text{ret}}\mathrm{d}u_1\mathrm{d}u_2\\
4\pi c_\infty^2 \frac{\partial p'_d}{\partial \boldsymbol{x}_i}(\boldsymbol{x}_i,t) = &-\int_{f=0}\left[\ddot{\boldsymbol{L}}_j\boldsymbol{D}_{G1A,ij}J\right]_{\text{ret}}\mathrm{d}u_1\mathrm{d}u_2-\int_{f=0}\left[2\dot{\boldsymbol{L}}_j\boldsymbol{D}_{G1A,ij}\dot{J}\right]_{\text{ret}}\mathrm{d}u_1\mathrm{d}u_2\\
&-\int_{f=0}\left[\dot{\boldsymbol{L}}_j\boldsymbol{E}_{G1A,ij}J\right]_{\text{ret}}\mathrm{d}u_1\mathrm{d}u_2-\int_{f=0}\left[\boldsymbol{L}_j\boldsymbol{D}_{G1A,ij}\ddot{J}\right]_{\text{ret}}\mathrm{d}u_1\mathrm{d}u_2\\
&-\int_{f=0}\left[\boldsymbol{L}_j\boldsymbol{E}_{G1A,ij}\dot{J}\right]_{\text{ret}}\mathrm{d}u_1\mathrm{d}u_2-\int_{f=0}\left[\boldsymbol{L}_j\boldsymbol{H}_{G1A,ij}J\right]_{\text{ret}}\mathrm{d}u_1\mathrm{d}u_2
\end{aligned}
\tag{44}
$$

where

$$
\boldsymbol{A}_{G1A,i}=R(0,1)\boldsymbol{A}_{G1,i},\qquad \boldsymbol{B}_{G1A,i}=R(0,1)(\dot{\boldsymbol{A}}_{G1,i}+\boldsymbol{B}_{G1,i}),\qquad \boldsymbol{C}_{G1A,i}=R(0,1)\dot{\boldsymbol{B}}_{G1,i}
$$

$$
\boldsymbol{D}_{G1A,ij}=R(0,1)\boldsymbol{C}_{G1,ij},\qquad \boldsymbol{E}_{G1A,ij}=R(0,1)(\dot{\boldsymbol{C}}_{G1,ij}+\boldsymbol{D}_{G1,ij})
$$

$$
\boldsymbol{H}_{G1A,ij}=R(0,1)\dot{\boldsymbol{D}}_{G1,ij}-\boldsymbol{E}_{G1,ij}
$$

Compact Formulation G1A

Equation 45 shows the compact form of Formulation G1A. This was derived by taking Eq. 43 and using Eq. 26 to change the observer time derivatives into source time derivatives and bringing them

into the line intregral.

$$
\begin{aligned}
4\pi\frac{c_\infty}{\rho_\infty}\frac{\partial p'_m}{\partial \boldsymbol{x}_i}(\boldsymbol{x}_i,t) = &-\int_{\mathcal{F}=0}\left[\ddot{\Psi}\mathscr{A}_{G1A,i}K\right]_{\text{ret}}\mathrm{d}u-\int_{\mathcal{F}=0}\left[3\ddot{\Psi}\mathscr{A}_{G1A,i}\dot{K}\right]_{\text{ret}}\mathrm{d}u\\
&-\int_{\mathcal{F}=0}\left[\ddot{\Psi}K\mathscr{B}_{G1A,i}K\right]_{\text{ret}}\mathrm{d}u-\int_{\mathcal{F}=0}\left[3\dot{\Psi}\mathscr{A}_{G1A,i}\ddot{K}\right]_{\text{ret}}\mathrm{d}u\\
&-\int_{\mathcal{F}=0}\left[2\dot{\Psi}\mathscr{B}_{G1A,i}\dot{K}\right]_{\text{ret}}\mathrm{d}u-\int_{\mathcal{F}=0}\left[\dot{\Psi}\mathscr{C}_{G1A,i}K\right]_{\text{ret}}\mathrm{d}u\\
&-\int_{\mathcal{F}=0}\left[\Psi\mathscr{A}_{G1A,i}\dddot{K}\right]_{\text{ret}}\mathrm{d}u-\int_{\mathcal{F}=0}\left[\Psi\mathscr{B}_{G1A,i}\ddot{K}\right]_{\text{ret}}\mathrm{d}u\\
&-\int_{\mathcal{F}=0}\left[\Psi\mathscr{C}_{G1A,i}\dot{K}\right]_{\text{ret}}\mathrm{d}u-\int_{\mathcal{F}=0}\left[\Psi\mathscr{D}_{G1A,i}K\right]_{\text{ret}}\mathrm{d}u\\
4\pi c_\infty^2\frac{\partial p'_d}{\partial \boldsymbol{x}_i}(\boldsymbol{x}_i,t) = &-\int_{\mathcal{F}=0}\left[\ddot{\boldsymbol{F}}_j\mathscr{E}_{G1A,ij}K\right]_{\text{ret}}\mathrm{d}u-\int_{\mathcal{F}=0}\left[2\dot{\boldsymbol{F}}_j\mathscr{E}_{G1A,ij}\dot{K}\right]_{\text{ret}}\mathrm{d}u\\
&-\int_{\mathcal{F}=0}\left[\dot{\boldsymbol{F}}_j\mathscr{F}_{G1A,ij}K\right]_{\text{ret}}\mathrm{d}u-\int_{\mathcal{F}=0}\left[\boldsymbol{F}_j\mathscr{E}_{G1A,ij}\ddot{K}\right]_{\text{ret}}\mathrm{d}u\\
&-\int_{\mathcal{F}=0}\left[\boldsymbol{F}_j\mathscr{F}_{G1A,ij}\dot{K}\right]_{\text{ret}}\mathrm{d}u-\int_{\mathcal{F}=0}\left[\boldsymbol{F}_jK\mathscr{G}_{G1A,ij}K\right]_{\text{ret}}\mathrm{d}u
\end{aligned}
\tag{45}
$$

where

$$
\begin{gathered}
\mathscr{A}_{G1A,i}=R(0,1)\mathscr{A}_{G1,i},\qquad \mathscr{B}_{G1A,i}=R(0,1)(\dot{\mathscr{A}}_{G1,i}+\mathscr{B}_{G1,i})\\
\mathscr{C}_{G1A,i}=R(0,1)(\dot{\mathscr{B}}_{G1,i}+\mathscr{C}_{G1,i}),\qquad \mathscr{D}_{G1A,i}=R(0,1)\dot{\mathscr{C}}_{G1,i}\\
\mathscr{E}_{G1A,ij}=R(0,1)\mathscr{D}_{G1,ij},\qquad \mathscr{F}_{G1A,ij}=R(0,1)(\dot{\mathscr{D}}_{G1,ij}+\mathscr{E}_{G1,ij})\\
\mathscr{G}_{G1A,ij}=R(0,1)\dot{\mathscr{E}}_{G1,ij}-\mathscr{F}_{G1,ij}
\end{gathered}
$$

2.7 Formulation V1

Although Formulation V1 is a Formulation reported in Reference [12], it has not yet been implemented in ANOPP2.

2.8 Formulation V1A

Formulation V1A was first introduced in Reference [12] by defining an acoustic velocity potential from Formuation 1A, then integrating over time to get the acoustic velocity. The steps taken in the derivation are not shown here and the authors suggest referring to Reference [12] for the mathematical derivation. There are several ways to derive Formulation V1A, one of which begins by starting with Formulation G0. Formulation V1A is implemented in ANOPP2's Formulation V1A Internal Functional Module (AFV1AIFM). Since Formulation V1A contains an observer time integral outside of a surface integral, AFV1AIFM supports only the following groupings: *TOTAL*, *COMPONENT*, and *INTEGRAL*.

Noncompact Formulation V1A

The noncompact form of Formulation V1A is shown in Eq. 46.

$$\begin{aligned}
4\pi\rho_\infty c_\infty \boldsymbol{u}'_{i,m}(\boldsymbol{x}_i,t) = &\int_{f=0}\left[\dot{Q}\boldsymbol{A}_{V1A,i}J\right]_{\text{ret}}\mathrm{d}u_1\mathrm{d}u_2 + \int_{f=0}\left[Q\boldsymbol{B}_{V1A,i}J\right]_{\text{ret}}\mathrm{d}u_1\mathrm{d}u_2 + \\
&\int_{f=0}\left[Q\boldsymbol{A}_{V1A,i}\dot{J}\right]_{\text{ret}}\mathrm{d}u_1\mathrm{d}u_2 \\
4\pi\rho_\infty c_\infty^2 \boldsymbol{u}'_{i,d}(\boldsymbol{x}_i,t) = &\int_{f=0}\left[\dot{\boldsymbol{L}}_j\boldsymbol{C}_{V1A,ij}J\right]_{\text{ret}}\mathrm{d}u_1\mathrm{d}u_2 + \int_{f=0}\left[\boldsymbol{L}_j\boldsymbol{D}_{V1A,ij}J\right]_{\text{ret}}\mathrm{d}u_1\mathrm{d}u_2 + \\
&\int_{f=0}\left[\boldsymbol{L}_j\boldsymbol{C}_{V1A,ij}\dot{J}\right]_{\text{ret}}\mathrm{d}u_1\mathrm{d}u_2 - \int_{\infty}^{t}\int_{f=0}\left[\boldsymbol{L}_j\boldsymbol{E}_{V1A,ij}J\right]_{\text{ret}}\mathrm{d}u_1\mathrm{d}u_2\mathrm{d}t'
\end{aligned} \tag{46}$$

where

$$\boldsymbol{A}_{V1A,i} = R(0,1)\boldsymbol{A}_{G0,i}, \qquad \boldsymbol{B}_{V1A,i} = R(0,1)\dot{\boldsymbol{A}}_{G0,i} + \boldsymbol{B}_{G0,i},$$
$$\boldsymbol{C}_{V1A,ij} = R(0,1)\boldsymbol{C}_{G0,ij}, \qquad \boldsymbol{D}_{V1A,ij} = R(0,1)\dot{\boldsymbol{C}}_{G0,ij} - \boldsymbol{D}_{G0,ij}, \qquad \boldsymbol{E}_{V1A,ij} = \boldsymbol{E}_{G0,ij}$$

Compact Formulation V1A

The compact form of Formulation V1A is similar to the previous compact Formulations. This is shown in Eq. 47 and is implemented in AFV1AIFM.

$$\begin{aligned}
4\pi c_\infty \boldsymbol{u}'_{i,m}(\boldsymbol{x}_i,t) = &\int_{F=0}\left[\ddot{\Psi}\mathscr{A}_{V1A,i}K\right]_{\text{ret}}\mathrm{d}u + \int_{F=0}\left[2\dot{\Psi}\mathscr{A}_{V1A,i}\dot{K}\right]_{\text{ret}}\mathrm{d}u + \\
&\int_{F=0}\left[\dot{\Psi}\mathscr{B}_{V1A,i}K\right]_{\text{ret}}\mathrm{d}u + \int_{F=0}\left[\Psi\mathscr{A}_{V1A,i}\ddot{K}\right]_{\text{ret}}\mathrm{d}u + \\
&\int_{F=0}\left[\Psi\mathscr{B}_{V1A,i}\dot{K}\right]_{\text{ret}}\mathrm{d}u + \int_{F=0}\left[\Psi\mathscr{C}_{V1A,i}K\right]_{\text{ret}}\mathrm{d}u \\
4\pi\rho_\infty c_\infty^2 \boldsymbol{u}'_{i,d}(\boldsymbol{x}_i,t) = &\int_{F=0}\left[\dot{\boldsymbol{F}}_j\mathscr{D}_{V1A,ij}K\right]_{\text{ret}}\mathrm{d}u + \int_{F=0}\left[\boldsymbol{F}_j\mathscr{E}_{V1A,ij}K\right]_{\text{ret}}\mathrm{d}u + \\
&\int_{F=0}\left[\boldsymbol{F}_j\mathscr{D}_{V1A,ij}\dot{K}\right]_{\text{ret}}\mathrm{d}u - \int_{\infty}^{t}\int_{F=0}\left[\boldsymbol{F}_j\mathscr{F}_{V1A,ij}K\right]_{\text{ret}}\mathrm{d}u\mathrm{d}t'
\end{aligned} \tag{47}$$

where

$$\mathscr{A}_{V1A,i} = R(0,2)\mathscr{A}_{G0,i}, \qquad \mathscr{B}_{V1A,i} = R(0,1)(\dot{R}(0,1)\mathscr{A}_{G0,i} + \mathscr{B}_{G0,i})$$
$$\mathscr{C}_{V1A,i} = R(0,2)\ddot{\mathscr{A}}_{G0,i} + R(0,1)\dot{R}(0,1)\dot{\mathscr{A}}_{G0,i} + R(0,1)\dot{\mathscr{B}}_{G0,i}$$
$$\mathscr{D}_{V1A,ij} = R(0,1)\mathscr{C}_{G0,ij}, \qquad \mathscr{E}_{V1A,ij} = R(0,1)\dot{\mathscr{C}}_{G0,ij} - \mathscr{D}_{G0,ij}, \qquad \mathscr{F}_{V1A,ij} = \mathscr{E}_{G0,ij}$$

2.9 Formulation 2B

Formulation 2B is a broadband formulation proposed by Farassat and Casper that computes the acoustic pressure spectrum via the acoustic velocity potential by way of a Fourier Transform [13].

This is shown in Eq. 48. Although Formulation 2B does not contain any observer time derivatives outside of the surface integrals, it does contain an observer time integration. For this reason, returning all the grouping options is not possible. Formulation 2B supports grouping by *TOTAL*, *COMPONENT*, and *INTEGRAL* only. The compact and noncompact forms of Formulation 2B are implemented in ANOPP2's Formulation 2B Internal Functional Module (AF2BIFM).

$$\tilde{p}(\boldsymbol{x}_i, f) = -2\pi i f \rho_\infty \tilde{\Phi}(\boldsymbol{x}_i, f) \tag{48}$$

Noncompact Formulation 2B

Applying Eq. 48 to Formulation 1, shown in Eq. 34, produces Formulation 2B, which is shown in Eq. 49.

$$\begin{aligned} 4\pi\rho_\infty \Phi_m(\boldsymbol{x}_i, t) &= -\int_{f=0} \Big[Q J A_{2B} \Big]_{\text{ret}} \mathrm{d}u_1 \mathrm{d}u_2 \\ 4\pi\rho_\infty c_\infty \Phi_d(\boldsymbol{x}_i, t) &= -\int_{f=0} \Big[\boldsymbol{L}_i J B_i \Big]_{\text{ret}} \mathrm{d}u_1 \mathrm{d}u_2 - \int_{-\infty}^{t} \int_{f=0} \Big[\boldsymbol{F}_i J C_i \Big]_{\text{ret}} \mathrm{d}u_1 \mathrm{d}u_2 \mathrm{d}t' \end{aligned} \tag{49}$$

where

$$A_{2B} = R(1,1), \qquad \boldsymbol{B}_{2B,i} = R(1,1)\hat{\boldsymbol{r}}_i, \qquad \boldsymbol{C}_{2B,i} = c_\infty R(2,1)\hat{\boldsymbol{r}}_i$$

Compact Formulation 2B

Following the same process for deriving the compact form of the Formulations results in the compact form of Formulation 2B. This is shown below in Eq. 50.

$$\begin{aligned} 4\pi\Phi_m(\boldsymbol{x}_i, t) &= -\frac{\partial}{\partial t} \int_{F=0} \Big[\Psi K \mathscr{A}_{2B} \Big]_{\text{ret}} \mathrm{d}u \\ 4\pi\rho_\infty c_\infty \Phi_d(\boldsymbol{x}_i, t) &= -\int_{F=0} \Big[\boldsymbol{F}_i K \mathscr{B}_i \Big]_{\text{ret}} \mathrm{d}u - \int_{-\infty}^{t} \int_{F=0} \Big[\boldsymbol{F}_i K \mathscr{C}_i \Big]_{\text{ret}} \mathrm{d}u \mathrm{d}t' \end{aligned} \tag{50}$$

$$\mathscr{A}_{2B} = R(1,1), \qquad \mathscr{B}_{2B,i} = R(1,1)\hat{\boldsymbol{r}}_i, \qquad \mathscr{C}_{2B,i} = c_\infty R(2,1)\hat{\boldsymbol{r}}_i$$

3 Process Summary

All Formulations implemented in AFFIFMs follow the same general code process. This chapter summarizes that code process, defining the steps taken to predict the noise, the methodology for determining implicit time ranges, interpolation into a common reception time, and other general code processes.

The first section presented here is for the kernel operation, which is the part of the code that predicts the APTH from the FWH surface or line. The kernel requires some information that may be specified a number of different ways. This includes the source and observer time ranges and resolution, surface or line properties, and additional user options. After the kernel is explained in the first section, these additional code processes will be explained in the following sections.

3.1 Formulation Kernel

The Formulations involve one integral, the FWH surface or line; however, the source time assumption within the integrand implies a second integral over source or observer time. This integral can be performed through a source time dominant or observer time dominant approach. Reference [27] explains the difference between a source time and observer time dominant approach in terms of computational efficiency. Their conclusion is that when there is complex motion, i.e., multiple frame changes between the FWH surface or line and the ground frame, it is more efficient to compute the noise using a source time dominant approach. Since AFFIFMs do not assume any source motion, i.e., it is entirely up to the user to specify how the surfaces or lines move, the source time dominant algorithm is more efficient and therefore implemented in AFFIFMs. The source time dominant algorithm of the kernel, including intermediary steps, for all AFFIFMs, is shown in Fig. 3.

There are two iterations involved in calculating the noise; the first is over source time and the second is over source position. There are two important facets to these iterations:

1. The first facet is that the source time iteration loop does not have to be the same source time as that defined in the FWH surface or line. For example, if the FWH surface is for a rotor blade and that blade is defined at every half degree, the source time iteration loop can be set at twice the resolution, at every quarter degree. An interpolation in the blade frame of reference of the FWH surface would then occur. Also, if the FWH surface or line is defined for a very long time range, for example four rotor revolutions, but the noise from only the last two revolutions is desired, the source time range can be limited so that only the last two revolutions are computed. Another example of this is if an FWH surface represents a rotor blade whose properties are defined as periodic over one revoluion, but the simulation is for an entire rotorcraft flyover of an oberver containing many revolutions. The surface or line is defined at a limited number of time steps, for a single period of the rotor revolution, but the source time is for the flyover event. We introduce a new term here, *key*, to clarify the difference. A key is a time step in the instantiation of the FWH surface or line held within AGDS. A source time (also called retarded or emission time) is that time in which the noise is computed and may occur between two keys of the FWH surface or line.

2. The second facet is that the source position in the inner iteration may be over every node on the FWH surface or line or every cell on the FWH surface or line depending on whether it is node based or cell based. This is recorded when the user creates the FWH surface or line

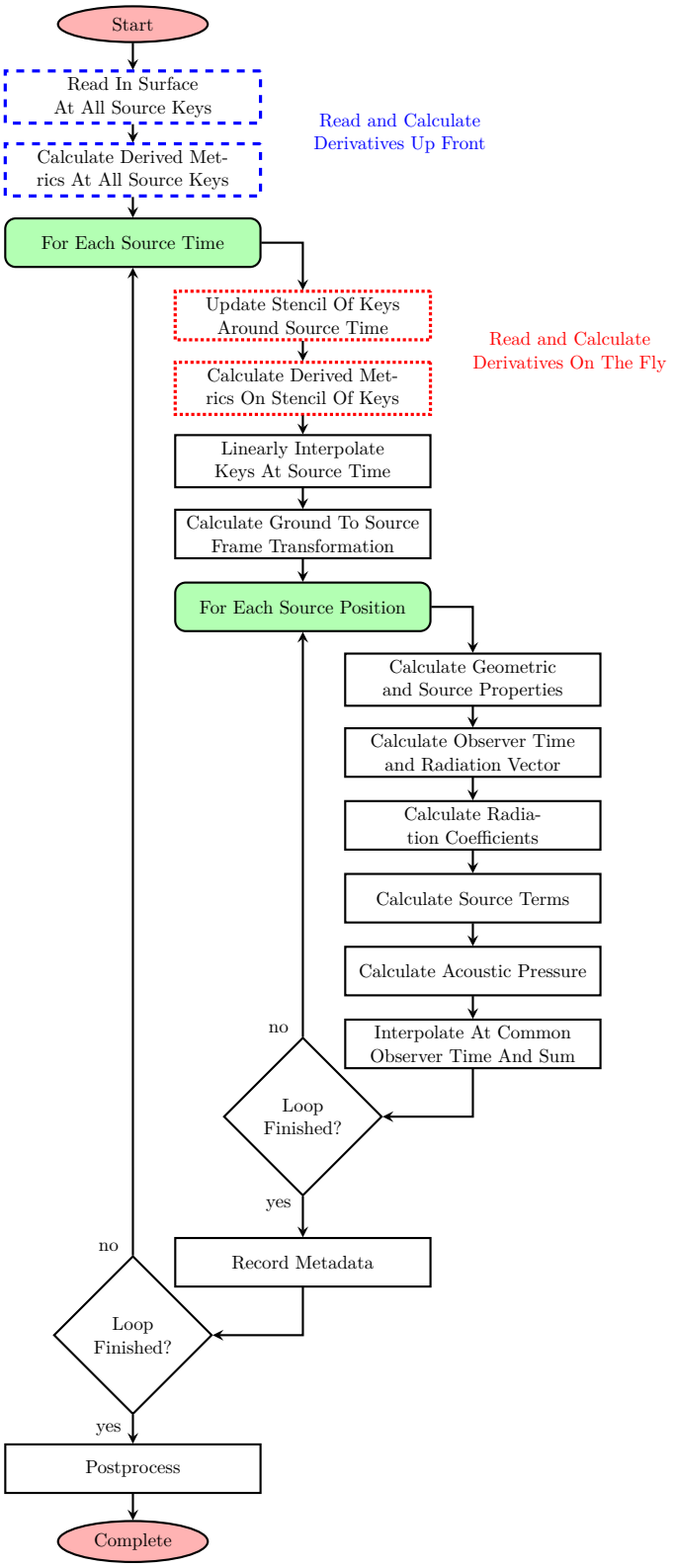


Figure 3: Diagram showing the kernel of the AFFIFMs.

using AGDS. If the surface or line is cell based, then an average of the nodes that define the cell are used as the source position.

3.1.1 Linear Interpolation of FWH Surface or Line Data

The difference between source keys and source time steps was explained in the previous section. In this section, the linear interpolation of the source keys at the source time is explained. It is important to note that higher order interpolation methods, like those mentioned in Section 3.1.7, have not yet been explored. As mentioned above, the source keys are used to determine the source properties at a source time. The source properties, such as position, velocity, and normals, are interpolated between an upper and lower key, denoted as k_u and k_l, using linear interpolation coefficients a_u and a_l. When calculating source time derivative metrics, such as velocity, a second-order central stencil is used to calculate the derivative, $\dot{k}_j$, at the source keys that comprise the second-order central stencil, k_{j-1} and k_{j+1}. As a result of the derivative and the linear interpolation, in order to calculate the interpolated time derivative quantity, a stencil of source keys is required. A schematic of a required source key stencil to calculate source velocity at source time, τ_i, is shown in Fig. 4. It is obvious that higher order interpolations would result in a more accurate value for the source property; however, the influence on the noise has not been quantified.

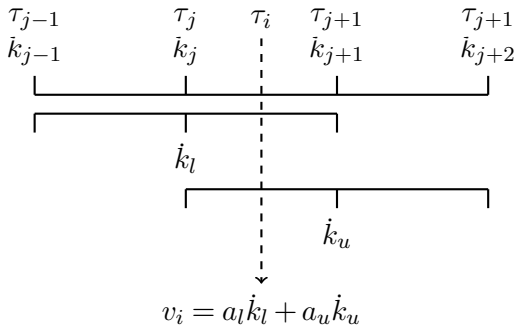


Figure 4: Schematic for calculating a time derivative of a source property.

At this portion of the kernel, the interpolation does not yet occur, only the determination of the linear interpolation coefficients and the derived metrics at the required upper and lower key indices.

3.1.2 Memory Efficiency and Optimization

Figure 3 shows the kernel of AFFIFMs and Fig. 4 shows how several key steps are used to define a source property at a given source time. There are two parts of Fig. 3 that are highlighted showing the options for memory efficiency and speed: the option to read the FWH surface or line 'up front' (before the source time iteration) or 'on the fly' (as needed during the source time iteration). The 'up front' option means the entire surface or line is read in and the derived metrics are calculated for every key. This way, the linear interpolation occuring in the source time iteration does not have to calculate these derived metrics at the upper and lower keys before interpolating. The 'on the fly' option means the entire surface or line is not read in, and only what is required for interpolation of the derived metrics is read in. This option is important if the source is a very large data set, such as that which would be required to capture broadband noise. The FWH surface may be of

such a size that holding it in memory, including derived metrics such as normal vectors, areas, line length, and velocity, is not feasible. In this case, the kernel, using the 'on the fly' option can read only that which is required to calculate the source properties at the source time, e.g., the source key stencil like that shown in Fig. 4. Conversely, if the surface is periodic and the source time is for several periods, reading 'on the fly' would calculate the same derivatives at source keys for every iteration of the source period, potentially increasing computation time significantly with redundant computations.

It is also important to note that when reading 'on the fly', the key stencil is updated at each source time iteration. So, if the source time step associated with τ_i requires keys k_{j-1}, k_j, k_{j+1}, and k_{j+2}, and the following source time step associated with τ_{i+1} requires keys k_j, k_{j+1}, k_{j+2}, and k_{j+3}, then key k_{j-1} is deleted, and key k_{j+3} is read in and used to calculated the derived metrics. The source keys that are required for both source time steps, k_j, k_{j+1} and k_{j+2}, are kept in place, including their derived metrics.

3.1.3 Source to Ground and Observer To Ground Transformations

The FWH surface or line specified using AGDS may be defined in a different frame of reference than the observer, such as the blade frame. Therefore, it is important to specify how both the source position and observer position are defined in the ground frame (or base frame) of reference so that we may calculate the radiation vector. This is done through a tranformation $[\Theta_{N/0}]$, which relates how a vector defined in the local frame N can be defined in the ground frame, shown in Eq. 51, where coordinates defined in their local frame are denoted with a star superscript, such as the source position $\boldsymbol{y}_i^*$ [28].

$$\boldsymbol{y}_i(\tau) = [\Theta_{N/0}(\tau)]\boldsymbol{y}_i^*(\tau) \tag{51}$$

The frame change transformation can be chained together when there are multiple frame changes, such as the aircraft center of gravity position, orientation angles, rotor offset, rotor tilt, and blade pitch, flap, and lead lag. This is shown in Eq. 52. However, to combine two transformations together, they themselves must be defined in the same frame. The frame upon which a transformation is defined is denoted by a superscript.

$$[\Theta^0_{N/0}(\tau)] = [\Theta^0_{N/N-1}(\tau)][\Theta^0_{N-1/N-2}(\tau)]...[\Theta^0_{i/i-1}(\tau)]...[\Theta^0_{2/1}(\tau)][\Theta^0_{1/0}(\tau)] \tag{52}$$

To define a frame change transformation in the ground frame, it is reoriented by way of a transformation as well. This is shown in Eq. 53.

$$[\Theta^0_{i/0}(\tau)] = [\Theta^0_{i-1/0}(\tau)][\Theta^{i-1}_{i/i-1}(\tau)] \tag{53}$$

It is important to note that a transformation is not simply a position rotation and offset, but also may contain velocity, acceleration, jerk, and snap functionality, allowing computation of properties such as the velocity in the ground frame of a point fixed on a blade. Each of AFFIFMs require a certain source time derivative, and only up to the metric required by the AIFM is computed (for example, an acceleration transformation will include velocity transformation). These are tabulated in Table 1. For all of AFFIFMs, the observer only requires a position transformation. Also, the transformation may take into account whether the quantity in the local frame is time dependent, such as local velocity of a node on a deforming surface, $\dot{\boldsymbol{y}}_i^*$. The frame changes themselves may be

Table 1: Transformations used by each of the Formulations available in AFFIFMs.

Formulation	Noncompact	Compact
Formulation 1	Velocity	Velocity
Formulation 1A	Acceleration	Jerk
Formulation G0	Velocity	Velocity
Formulation G1	Acceleration	Jerk
Formulation G1A	Jerk	Snap
Formulation V1A	Accleration	Jerk
Formulation 2B	Velocity	Velocity

time dependent. The options for time dependency include constant frame change, aperiodic frame change, periodic frame change, or a polynomial frame change of an angle rotation or position offset. A list of time dependent frame changes is referred to as a frame of reference list.

A significant amount of derivation for these operations is left out of this document for brevity; however, ANOPP2 includes a utility, AKU, to relate frames of reference and chain frame changes together as well as to define the positions, velocities, accelerations, jerks and snaps in the local and global frame of reference. A user specifies a list of time dependent local frame changes, $[\Theta^{i-1}_{i/i-1}(\tau)]$, and AFFIFMs then use the list of time dependent frame changes, i.e., a frame of reference list, to define the source in the ground frame. Then, source position and other derived metrics defined in the local frame are used with the source to ground frame change transformation to define these metrics in the ground frame. A similar operation occurs for the observer position. See the AKU Reference Manual for more details [15].

3.1.4 Geometric and Source Properties

The geometric and source properties portion of the kernel is the first to occur within the source position iteration. A geometric property is a quantity that is or is derived from the positions of the surface or line. This includes source position, velocity, acceleration, jerk, snap, normal, source time derivatives of the normal, area, and length. Source properties include density, momentum, surface pressure, cross-sectional area, and lifting line force, and their source time derivatives. At this part of the process, the linear interpolation coefficients, derived metrics at the neighboring upper and lower keys, and frame transformations are available for a given source time step. This portion of the code performs the linear interpolation of the source property at a given source position (node or cell) and transforms to the ground frame. The required geometric and source properties for the noncompact Formulations are shown in Table 2 and for the compact Formulations are shown in Table 3.

3.1.5 Retarded Time Calculation and Radiation Coefficients

Arguably, one of the most critical parts of the process for AFFIFMs is the source time calculation, where, given a source time, τ, and a source position, $\boldsymbol{y}_i(\tau)$, an observer time is calculated via $t = \tau + r/c_\infty$. The challenge faced is that, if the observer is moving, the radiation vector is a function of observer time, $t = \tau + r(t)/c_\infty$. Since there is no way to form a closed solution, this must be solved iteratively. Furthermore, since this is at the very heart of the kernel, any and all

Table 2: Geometric and Source Properties for noncompact Formulations.

Formulation	Geometric Properties			Source Properties		
Formulation 1	$\boldsymbol{y}_i, \boldsymbol{v}_i$	$\hat{\boldsymbol{n}}_i$	J	ρ'	$(\boldsymbol{\rho u})_i$	p'
Formulation 1A	$\boldsymbol{y}_i, \boldsymbol{v}_i, \dot{\boldsymbol{v}}_i$	$\hat{\boldsymbol{n}}_i, \dot{\hat{\boldsymbol{n}}}_i$	$J, \dot{J}$	$\rho', \dot{\rho}'$	$(\boldsymbol{\rho u})_i, (\dot{\boldsymbol{\rho u}})_i$	$p', \dot{p}'$
Formulation G0	$\boldsymbol{y}_i, \boldsymbol{v}_i$	$\hat{\boldsymbol{n}}_i$	J	ρ'	$(\boldsymbol{\rho u})_i$	p'
Formulation G1	$\boldsymbol{y}_i, \boldsymbol{v}_i, \dot{\boldsymbol{v}}_i$	$\hat{\boldsymbol{n}}_i, \dot{\hat{\boldsymbol{n}}}_i$	$J, \dot{J}$	$\rho', \dot{\rho}'$	$(\boldsymbol{\rho u})_i, (\dot{\boldsymbol{\rho u}})_i$	$p', \dot{p}'$
Formulation G1A	$\boldsymbol{y}_i, \boldsymbol{v}_i, \dot{\boldsymbol{v}}_i, \ddot{\boldsymbol{v}}_i$	$\hat{\boldsymbol{n}}_i, \dot{\hat{\boldsymbol{n}}}_i, \ddot{\hat{\boldsymbol{n}}}_i$	$J, \dot{J}, \ddot{J}$	$\rho', \dot{\rho}', \ddot{\rho}'$	$(\boldsymbol{\rho u})_i, (\dot{\boldsymbol{\rho u}})_i, (\ddot{\boldsymbol{\rho u}})_i$	$p', \dot{p}', \ddot{p}'$
Formulation V1A	$\boldsymbol{y}_i, \boldsymbol{v}_i, \dot{\boldsymbol{v}}_i$	$\hat{\boldsymbol{n}}_i, \dot{\hat{\boldsymbol{n}}}_i$	$J, \dot{J}$	$\rho', \dot{\rho}'$	$(\boldsymbol{\rho u})_i, (\dot{\boldsymbol{\rho u}})_i$	$p', \dot{p}'$
Formulation 2B	$\boldsymbol{y}_i, \boldsymbol{v}_i$	$\hat{\boldsymbol{n}}_i$	J	ρ'	$(\boldsymbol{\rho u})_i$	p'

Table 3: Geometric and Source Properties for compact Formulations.

Formulation	Geometric Properties		Source Properties	
Formulation 1	$\boldsymbol{y}_i, \boldsymbol{v}_i$	K	Ψ	$\boldsymbol{F}_i$
Formulation 1A	$\boldsymbol{y}_i, \boldsymbol{v}_i, \dot{\boldsymbol{v}}_i, \ddot{\boldsymbol{v}}_i$	$K, \dot{K}, \ddot{K}$	$\Psi, \dot{\Psi}, \ddot{\Psi}$	$\boldsymbol{F}_i, \dot{\boldsymbol{F}}_i$
Formulation G0	$\boldsymbol{y}_i, \boldsymbol{v}_i$	K	Ψ	$\boldsymbol{F}_i$
Formulation G1	$\boldsymbol{y}_i, \boldsymbol{v}_i, \dot{\boldsymbol{v}}_i, \ddot{\boldsymbol{v}}_i$	$K, \dot{K}, \ddot{K}$	$\Psi, \dot{\Psi}, \ddot{\Psi}$	$\boldsymbol{F}_i, \dot{\boldsymbol{F}}_i$
Formulation G1A	$\boldsymbol{y}_i, \boldsymbol{v}_i, \dot{\boldsymbol{v}}_i, \ddot{\boldsymbol{v}}_i, \dddot{\boldsymbol{v}}_i$	$K, \dot{K}, \ddot{K}, \dddot{K}$	$\Psi, \dot{\Psi}, \ddot{\Psi}, \dddot{\Psi}$	$\boldsymbol{F}_i, \dot{\boldsymbol{F}}_i, \ddot{\boldsymbol{F}}_i$
Formulation V1A	$\boldsymbol{y}_i, \boldsymbol{v}_i, \dot{\boldsymbol{v}}_i, \ddot{\boldsymbol{v}}_i$	$K, \dot{K}, \ddot{K}$	$\Psi, \dot{\Psi}, \ddot{\Psi}$	$\boldsymbol{F}_i, \dot{\boldsymbol{F}}_i$
Formulation 2B	$\boldsymbol{y}_i, \boldsymbol{v}_i$	K	Ψ	$\boldsymbol{F}_i$

optimization techniques should be implemented and investigated. That being said, a simple fixed point iterative approach is implemented in AFFIFMs, and limited attempts at using higher accuracy optimization techniques have occurred. In the approach, an initial estimate of the observer time, denoted by t_e, is determined from the previous source time step's calculation of observer time. The iterative process for determining the observer time implementd in AFFIFMs is shown in Fig. 5.

An important component of the source time calculation is the tolerance upon which a valid observer time is found. This tolerance, ϵ, is set as a fraction of the observer time step size, Δt. The tolerance is set by the user, which is shown in Sections 5 and 6.

Once the radiation vector, $\boldsymbol{r}_i$, and radiation distance, r, are known, it is straight forward to calculate the radiation direction vector. Also, once the radiation direction is known, terms like the Mach vector dotted with the radiation direction, $\dot{\boldsymbol{M}}_r$, are easily found. This is shown in Eqs. 54 and 55. After these operations, finding the radiation coefficients is also straight forward.

$$\hat{\boldsymbol{r}}_i = \frac{\boldsymbol{r}_i}{r} \tag{54}$$

$$M_r = \boldsymbol{M}_i \hat{\boldsymbol{r}}_i = \frac{\boldsymbol{v}_i}{c_\infty} \hat{\boldsymbol{r}}_i \tag{55}$$

3.1.6 Source Terms and Acoustic Pressure

The next step in the kernel is to calculate the acoustic source terms, Q, $\boldsymbol{L}_i$, Ψ, and $\boldsymbol{F}_i$, and their source time derivatives. For the compact formulations, this is straight forward even for higher order source time derivatives of the compact source terms, Ψ and $\boldsymbol{F}_i$. Source time derivatives of

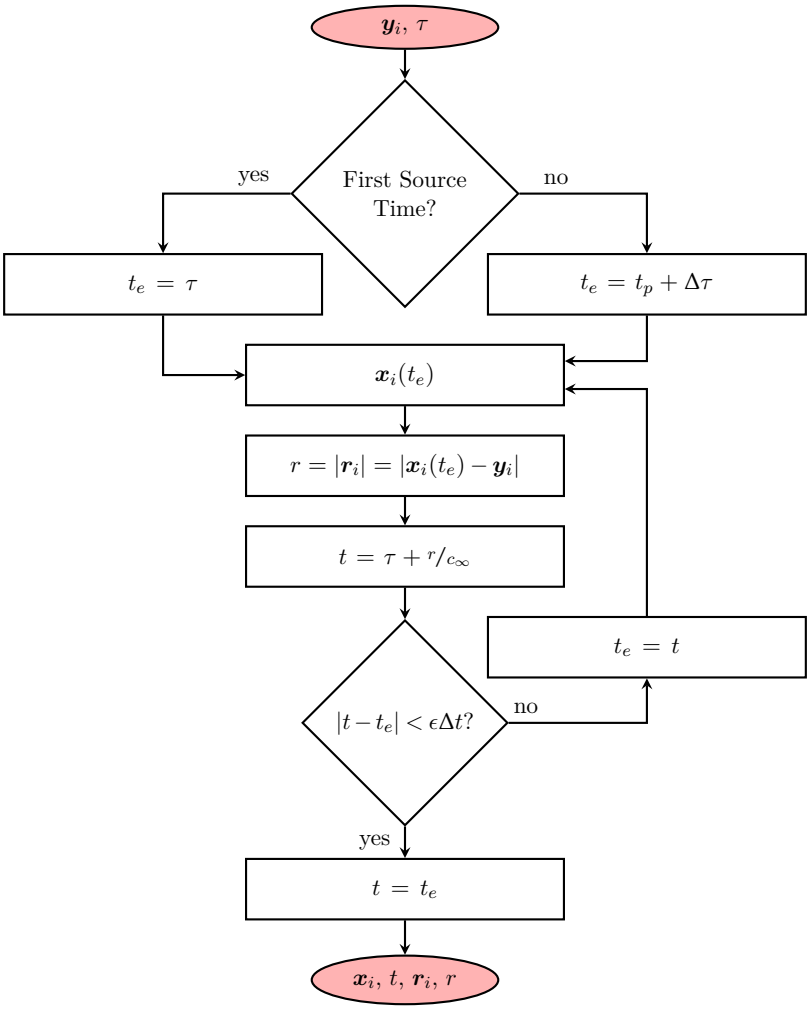


Figure 5: Diagram showing the source time calculation.

the noncompact source terms, Q and $\boldsymbol{L}_i$, are found via the chain rule. For instance, the first-order source term derivative of Q for a permeable surface is shown in Eq. 56.

$$\dot{Q} = \frac{\partial}{\partial \tau}\left\{\rho_\infty v_n + \rho(u_n - v_n)\right\} = \rho_\infty(\dot{v}_n + v_{\dot{n}}) + \dot{\rho}(u_n - v_n) + \rho(\dot{u}_n - \dot{v}_n + u_{\dot{n}} - v_{\dot{n}}) \tag{56}$$

After finding the source time derivatives of the source terms, the result metric can be found by multiplying the corresponding source term (or its source time derivative), surface area or line length (or its source term derivative), and the radiation coefficient. The result is a vector of results (acoustic pressures, pressure gradients, acoustic potentials, or acoustic velocities) corresponding to each of the integrals in the Formulation of interest.

3.1.7 Interpolation At Observer Time

The acoustic pressure, observer position, and observer time have been calculated for a given source position and source time; however, every source position on the surface with the same source time may have different observer times. Also, since the source may be moving toward or away from the observer, the observer times from a series of evenly spaced source times may be unevenly spaced. In order to sum the acoustic metric from an entire surface and also to perform frequency analyses (Fourier transform), an interpolation occurs to find the acoustic pressure at an observer time in an evenly spaced array, denoted p'_i. This operation occurs during the source time loop in order to reduce memory footprint by storing a stencil of N observer time - acoustic metric pairs, shown in Fig. 6, denoted by subscript j. A polynomial interpolation is performed to find the acoustic metric, p_i, shown in Eq. 57. The size of the polynomial interpolation is set by the user in the AFFIFMs configuration file.

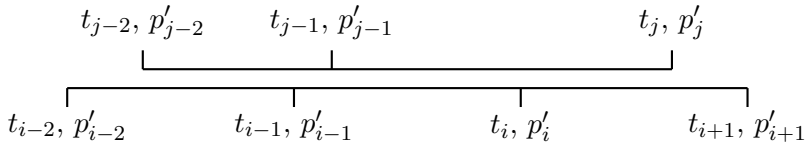


Figure 6: Stencil of acoustic metric and observer time pairs used for observer time interpolation where N is 3.

$$p'_i = \sum_{n=0}^{N-1} a_n t_i^n \tag{57}$$

where the coefficients are found by solving the linear system of equations shown in Eq. 58.

$$\begin{bmatrix} 1 & t_j & \cdots & t_j^{N-1} \\ 1 & t_{j-1} & \cdots & t_{j-1}^{N-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & t_{j-(N-1)} & \cdots & t_{j-(N-1)}^{N-1} \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_{N-1} \end{bmatrix} = \begin{bmatrix} p'_j \\ p'_{j-1} \\ \vdots \\ p'_{j-(N-1)} \end{bmatrix} \tag{58}$$

Once the metric from a given source position and source time is found at the appropriate observer time, it is accumulated with any other source position and source time whose pressure was recieved

at that observer time. Since the acoustic pressure is weighted by the surface area or line length, this means that our surface integration is essentially a midpoint Reimann sum over the surface. Farassat, in Reference [23], mentioned that, whenever possible, a higher order integration scheme, such as a Guassian integration, should be utilized. This has not been investigated at this time and, to the author's knowledge, has never been investigated.

3.1.8 Metadata

The last step in the Formulation kernel is storage of any metadata that the user may want. The metadata for AFFIFMs is exported in Tecplot binary format, but during the iterations it is stored in temporary files that are then reorganized into the format Tecplot requires [22]. As mentioned in Sec. 1, there are four levels of metadata exported by all AFFIFMs; however, the data in those levels may be different depending on AIFM and compactness. The four levels are *BASIC* (including the surface or line position and observer and source time of all surface or line nodes), *GEOMETRY* (including the surface or line geometric properties and kinematics), *SOURCE* (including all fluid dynamics, their derivatives, and source terms), and *ALL* (including the acoustic properties for all integrands at all surface or line nodes). Each metadata setting includes all previous ones; for example, the *SOURCE* metadata setting includes the *GEOMETRY* and *BASIC* metadata levels. In the special case of *BASIC*, all compact and noncompact AFFIFMs export the same information, as shown in Table 4. Tables 5 and 6 show the variables exported when *GEOMETRY* and *SOURCE* are active, respectively, for noncompact Formulations. Table 7 shows the variables exported when *GEOMETRY* and *SOURCE* metadata are active in compact AFFIFMs, respectively. Lastly, Tables 8 and 9 show the variables exported when *ALL* metadata are active in noncompact and compact AIFMs, respectively. Examples of metadata being exported will be shown in Section 6.

Table 4: Metadata variables exported when the *BASIC* setting is selected.

Formulation	Noncompact Metadata	Compact Metadata
All Formulations	$\boldsymbol{y}_i$, τ, t, $\boldsymbol{r}_i$, M	$\boldsymbol{y}_i$, τ, t, $\boldsymbol{r}_i$, M

Table 5: Additional metadata variables for metadata setting *GEOMETRY* for noncompact Formulations.

Formulation	*GEOMETRY*		
Formulation 1	$\boldsymbol{v}_i$	$\hat{\boldsymbol{n}}_i$	J
Formulation 1A	$\boldsymbol{v}_i, \dot{\boldsymbol{v}}_i$	$\hat{\boldsymbol{n}}_i, \dot{\hat{\boldsymbol{n}}}_i$	$J, \dot{J}$
Formulation G0	$\boldsymbol{v}_i$	$\hat{\boldsymbol{n}}_i$	J
Formulation G1	$\boldsymbol{v}_i, \dot{\boldsymbol{v}}_i$	$\hat{\boldsymbol{n}}_i, \dot{\hat{\boldsymbol{n}}}_i$	$J, \dot{J}$
Formulation G1A	$\boldsymbol{v}_i, \dot{\boldsymbol{v}}_i, \ddot{\boldsymbol{v}}_i$	$\hat{\boldsymbol{n}}_i, \dot{\hat{\boldsymbol{n}}}_i, \ddot{\hat{\boldsymbol{n}}}_i$	$J, \dot{J}, \ddot{J}$
Formulation V1A	$\boldsymbol{v}_i, \dot{\boldsymbol{v}}_i$	$\hat{\boldsymbol{n}}_i, \dot{\hat{\boldsymbol{n}}}_i$	$J, \dot{J}$
Formulation 2B	$\boldsymbol{v}_i$	$\hat{\boldsymbol{n}}_i$	J

Table 6: Additional metadata variables for metadata setting *SOURCE* for noncompact Formulations.

Formulation	*SOURCE*					
Formulation 1	ρ'	$(\boldsymbol{\rho}u)_i$	p'	Q	$\boldsymbol{U}_i$	$\boldsymbol{L}_i$
Formulation 1A	$\rho', \dot{\rho}'$	$(\boldsymbol{\rho}u)_i, (\dot{\boldsymbol{\rho}u})_i$	$p', \dot{p}'$	$Q, \dot{Q}$	$\boldsymbol{U}_i, \dot{\boldsymbol{U}}_i$	$\boldsymbol{L}_i, \dot{\boldsymbol{L}}_i$
Formulation G0	ρ'	$(\boldsymbol{\rho}u)_i$	p'	Q	$\boldsymbol{U}_i$	$\boldsymbol{L}_i$
Formulation G1	$\rho', \dot{\rho}'$	$(\boldsymbol{\rho}u)_i, (\dot{\boldsymbol{\rho}u})_i$	$p', \dot{p}'$	$Q, \dot{Q}$	$\boldsymbol{U}_i, \dot{\boldsymbol{U}}_i$	$\boldsymbol{L}_i, \dot{\boldsymbol{L}}_i$
Formulation G1A	$\rho', \dot{\rho}', \ddot{\rho}'$	$(\boldsymbol{\rho}u)_i, (\dot{\boldsymbol{\rho}u})_i, (\ddot{\boldsymbol{\rho}u})_i$	$p', \dot{p}', \ddot{p}'$	$Q, \dot{Q}, \ddot{Q}$	$\boldsymbol{U}_i, \dot{\boldsymbol{U}}_i, \ddot{\boldsymbol{U}}_i$	$\boldsymbol{L}_i, \dot{\boldsymbol{L}}_i, \ddot{\boldsymbol{L}}_i$
Formulation V1A	$\rho', \dot{\rho}'$	$(\boldsymbol{\rho}u)_i, (\dot{\boldsymbol{\rho}u})_i$	$p', \dot{p}'$	$Q, \dot{Q}$	$\boldsymbol{U}_i, \dot{\boldsymbol{U}}_i$	$\boldsymbol{L}_i, \dot{\boldsymbol{L}}_i$
Formulation 2B	ρ'	$(\boldsymbol{\rho}u)_i$	p'	Q	$\boldsymbol{U}_i$	$\boldsymbol{L}_i$

Table 7: Additional metadata variables for metadata setting *GEOMETRY* and *SOURCE* for compact Formulations.

Formulation	*GEOMTRY*		*SOURCE*	
Formulation 1	$\boldsymbol{y}_i, \boldsymbol{v}_i$	K	Ψ	$\boldsymbol{F}_i$
Formulation 1A	$\boldsymbol{y}_i, \boldsymbol{v}_i, \dot{\boldsymbol{v}}_i, \ddot{\boldsymbol{v}}_i$	$K, \dot{K}, \ddot{K}$	$\Psi, \dot{\Psi}, \ddot{\Psi}$	$\boldsymbol{F}_i, \dot{\boldsymbol{F}}_i$
Formulation G0	$\boldsymbol{y}_i, \boldsymbol{v}_i$	K	Ψ	$\boldsymbol{F}_i$
Formulation G1	$\boldsymbol{y}_i, \boldsymbol{v}_i, \dot{\boldsymbol{v}}_i, \ddot{\boldsymbol{v}}_i$	$K, \dot{K}, \ddot{K}$	$\Psi, \dot{\Psi}, \ddot{\Psi}$	$\boldsymbol{F}_i, \dot{\boldsymbol{F}}_i$
Formulation G1A	$\boldsymbol{y}_i, \boldsymbol{v}_i, \dot{\boldsymbol{v}}_i, \ddot{\boldsymbol{v}}_i, \dddot{\boldsymbol{v}}_i$	$K, \dot{K}, \ddot{K}, \dddot{K}$	$\Psi, \dot{\Psi}, \ddot{\Psi}, \dddot{\Psi}$	$\boldsymbol{F}_i, \dot{\boldsymbol{F}}_i, \ddot{\boldsymbol{F}}_i$
Formulation V1A	$\boldsymbol{y}_i, \boldsymbol{v}_i, \dot{\boldsymbol{v}}_i, \ddot{\boldsymbol{v}}_i$	$K, \dot{K}, \ddot{K}$	$\Psi, \dot{\Psi}, \ddot{\Psi}$	$\boldsymbol{F}_i, \dot{\boldsymbol{F}}_i$
Formulation 2B	$\boldsymbol{y}_i, \boldsymbol{v}_i$	K	Ψ	$\boldsymbol{F}_i$

Table 8: Additional metadata variables exported when the *ALL* setting is selected. $\boldsymbol{P}'$, $\boldsymbol{V}_i'$, and Φ are vectors of integrands for each integral in the associated noncompact Formulation.

Formulation	*ALL*	
Formulation 1	A_1, B_1, $\boldsymbol{C}_{1,i}$	$\boldsymbol{P}'$
Formulation 1A	A_1, B_1, $\boldsymbol{C}_{1,i}$, $\boldsymbol{D}_{1,i}$	$\boldsymbol{P}'$
Formulation G0	$\boldsymbol{A}_{G0,i}$, $\boldsymbol{B}_{G0,i}$, $\boldsymbol{C}_{G0,ij}$, $\boldsymbol{D}_{G0,ij}$, $\boldsymbol{E}_{G0,ij}$	$\partial \boldsymbol{P}_i'/\partial \boldsymbol{x}_i$
Formulation G1	$\boldsymbol{A}_{G1,i}$, $\boldsymbol{B}_{G1,i}$, $\boldsymbol{C}_{G1,ij}$, $\boldsymbol{D}_{G1,ij}$, $\boldsymbol{E}_{G1,ij}$	$\partial \boldsymbol{P}_i'/\partial \boldsymbol{x}_i$
Formulation G1A	$\boldsymbol{A}_{G1A,i}$, $\boldsymbol{B}_{G1A,i}$, $\boldsymbol{C}_{G1A,ij}$, $\boldsymbol{D}_{G1A,ij}$, $\boldsymbol{E}_{G1A,ij}$, $\boldsymbol{H}_{G1A,ij}$	$\partial \boldsymbol{P}_i'/\partial \boldsymbol{x}_i$
Formulation V1A	A_{V1A}, B_{V1A}, $\boldsymbol{C}_{V1A,i}$, $\boldsymbol{D}_{V1A,i}$, $\boldsymbol{E}_{V1A,i}$	$\boldsymbol{V}_i'$
Formulation 2B	A_{2B}, B_{2B}, $\boldsymbol{C}_{2B,i}$	Φ

Table 9: Additional metadata variables exported when the *ALL* setting is selected. $\boldsymbol{P'}$, $\boldsymbol{V}_i'$, and Φ are vectors of integrands for each integral in the associated compact Formulation.

Formulation	*ALL*	
Formulation 1	$\mathscr{A}_1$, $\mathscr{B}_1$, $\mathscr{C}_{1,i}$	$\boldsymbol{P'}$
Formulation 1A	$\mathscr{A}_{1A}$, $\mathscr{B}_{1A}$, $\mathscr{C}_{1A}$, $\mathscr{D}_{1A,i}$, $\mathscr{E}_{1A,i}$	$\boldsymbol{P'}$
Formulation G0	$\mathscr{A}_{G0,i}$, $\mathscr{B}_{G0,i}$, $\mathscr{C}_{G0,ij}$, $\mathscr{D}_{G0,ij}$, $\mathscr{E}_{G0,ij}$	$\partial \boldsymbol{P}_i'/\partial \boldsymbol{x}_i$
Formulation G1	$\mathscr{A}_{G1,i}$, $\mathscr{B}_{G1,i}$, $\mathscr{C}_{G1,i}$, $\mathscr{D}_{G1,ij}$, $\mathscr{E}_{G1,ij}$, $\mathscr{H}_{G1,ij}$	$\partial \boldsymbol{P}_i'/\partial \boldsymbol{x}_i$
Formulation G1A	$\mathscr{A}_{G1A,i}$, $\mathscr{B}_{G1A,i}$, $\mathscr{C}_{G1A,i}$, $\mathscr{D}_{G1A,i}$, $\mathscr{E}_{G1A,ij}$, $\mathscr{F}_{G1A,ij}$, $\mathscr{G}_{G1A,ij}$	$\partial \boldsymbol{P}_i'/\partial \boldsymbol{x}_i$
Formulation V1A	$\mathscr{A}_{V1A,i}$, $\mathscr{B}_{V1A,i}$, $\mathscr{C}_{V1A,i}$, $\mathscr{D}_{V1A,ij}$, $\mathscr{E}_{V1A,i,}$, $\mathscr{F}_{V1A,i,}$	$\boldsymbol{V'}$
Formulation 2B	$\mathscr{A}_{2B}$, $\mathscr{B}_{2B}$, $\mathscr{C}_{2B,i}$	Φ

In addition to time dependent data, AFFIFMs have the option to export frequency dependent metadata. If this option is turned on, a Fourier transform of all variables is taken before exporting. This allows a user to analyze frequency content from the source surface or line.

3.1.9 Postprocessing

After the source time iterations are complete, the last step in the AFFIFMs is the postprocessing step. Three operations may occur during this step:

1. Numerical observer time derivative calculations for Formulations, such as Formulation 1 and G0 (AF1IFM and AFG0IFM). The numerical observer time derivatives (first and second) are both second-order central schemes for evenly spaced discretization.

2. Formulation 2B (AF2BIFM) requires a conversion of the acoustic velocity potential from the time domain to the frequency domain before applying Eq. 48. This may require windowing before the Fourier Transform can be applied. See the ANOPP2's Acoustic Analysis Utility (AAAU) reference manual for more information on windowing time series [29].

3. Adding integrals into the groups desired by the user, such as summing monopole and dipole terms together (see Sec. 2.1.4).

3.2 Implicit and Explicit Source and Observer Times

Oftentimes, it is inconvenient to explicitly define source and observer time as it requires knowledge of the geometric setup of both the source and observer, as a function of time, and an estimate of the propagation time for all source nodes to all observers. AFFIFMs allow for the implicit declaration of source and observer time such that the entire emitting surface or line arrives at an observer whose definition includes those reception times. This is useful in situations such as when a rotorcraft blade defined by a surface or line is periodic over one rotor revolution (i.e., exists for all time but is only defined for one revolution) but the noise metric is desired only for one rotor period. The emitting surface must exist for larger than one period, since the propagation across the source surface or line elongates the source time range required to fill an observer time range. Another example is when the source is aperiodic (i.e., exists for only a clearly defined source time range) and the observer time range should be limited to only those times when the observer hears

the entire source. In both these examples, only one time range is required (observer or source) and the other is determined implicitly.

All AFFIFMs include capability for a user to specify either source time range or observer time range, and have the undefined time range determined implicitly. The source or observer time range is defined by three, potentially overlapping, time definitions: explicitly by the user in a configuration file, in the geometric definition of a time dependent geometry and/or flow data, and through the kinematics (motion) of the source or observer. The process for determining implicit source time range and number of source time steps, taking into account any aperiodic time range associated with the FWH surface or line including kinematics, is shown in Figs. 7 and 8, respectively. Also, the process for determining implicit observer time range and number of source time steps is shown in Figs. 9 and 10, respectively. Figures 7 through 10 include an additional time factor, ε, which is used to ensure that any implicit time range calculated is free of any numerical artifacts that might lead to spurious noise at the first and last portion of the noise prediction. It essentially lengthens (or shortens) the source (or observer) time so that the beginning and end of the observer time hears the entire surface.

3.3 Multiple FWH Surfaces or Lines and Multiple Observers

The Formulation kernel shown in Fig. 3 is for a single FWH surface or line and a single observer location (where an observer location is a time dependent coordinate). More often than not, AFFIFMs are called with multiple FWH surfaces or lines and, potentially, multiple observers. This section explains how multiple FWH surfaces or lines and multiple observer locations are handled by AFFIFMs.

All AFFIFMs take in an observer data structure defined by AODS. The observer data structure may be a single location or many locations with or without connectivity. Examples of two observer data structures are 1) a structured ground surface of observer locations and 1) an unconnected point cloud of observer locations. See the AODS Reference Manual for more information on specifying observer data structures [20]. For the purpose of AFFIFMs, every observer location is independent of every other observer location. Furthermore, every FWH surface or line is independent of every other FWH surface or line. These two iterations, one over observer location and one over FWH surface and/or line, occur outside the kernel process shown in Fig. 3. Figure 11 shows the iteration loops over observer position and FWH surface or line in relation to the kernel. These loops occur in different portions of ANOPP2. The first loop, over observer location, occurs in ANOPP2's Command Executive (ACE), a common process to all AIFMs. The second loop, for each FWH surface or line, occurs within the AFFIFMs. Shown in Fig. 11 is the step to calculate implicit source or observer time, which is explained in Section 3.2.

3.4 Parallelization

There are two methods of parallelization of AFFIFMs: Message Passing Interface (MPI) and Open Multiprocessing (OpenMP).

3.4.1 Parallelization with MPI

All of the AFFIFMs are capable of running in parallel using MPI via AMPIT [17, 18]. MPI is only available on Linux-based clusters. AMPIT uses schedulers in the AODS and AGDS as well

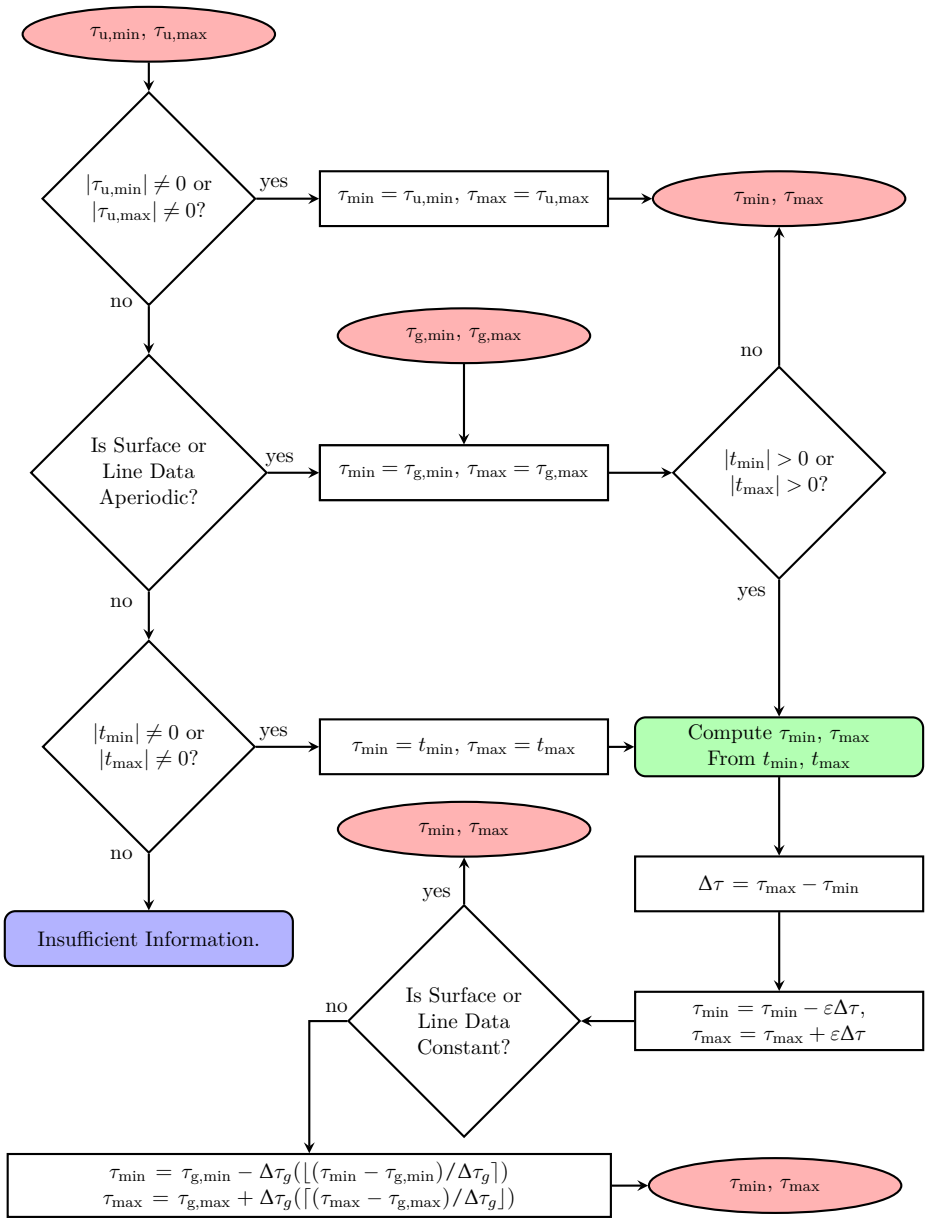


Figure 7: Schematic of implicit source time calculation. Subscript 'u' denotes user defined in a configuration file, and subscript 'g' denotes defined by the periodic or aperiodic FWH surface or line, including kinematics.

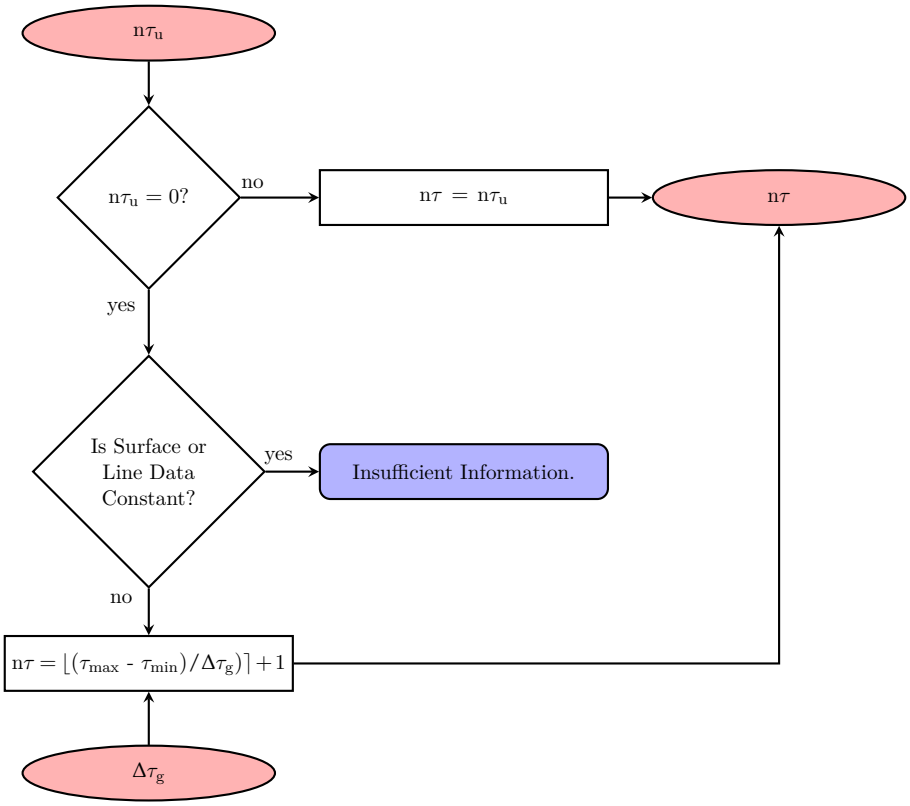


Figure 8: Schematic of calculation of the number of source time steps.

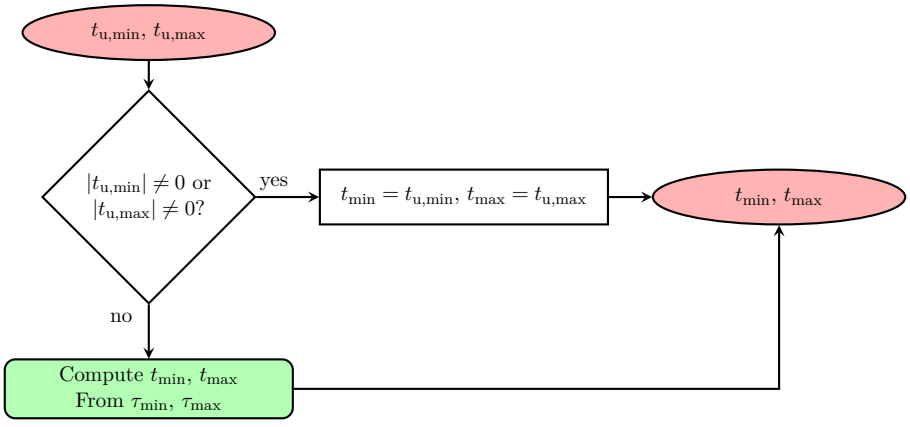


Figure 9: Schematic of implicit observer time calculation.

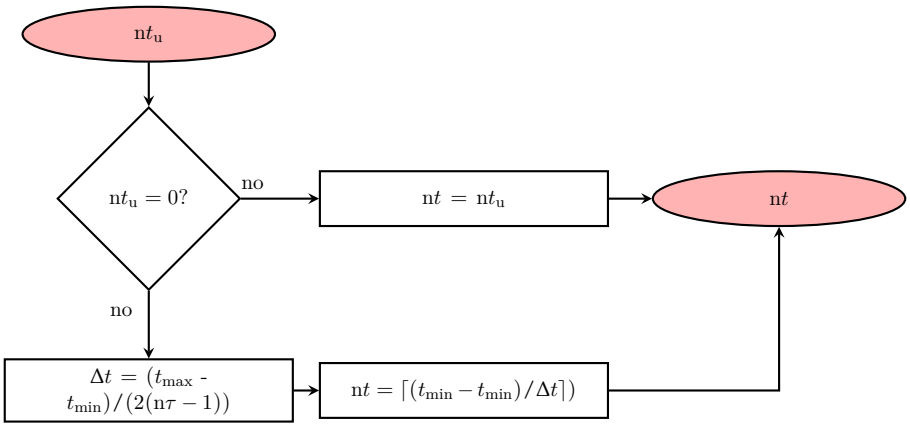


Figure 10: Schematic of calculation of the number of observer time steps.

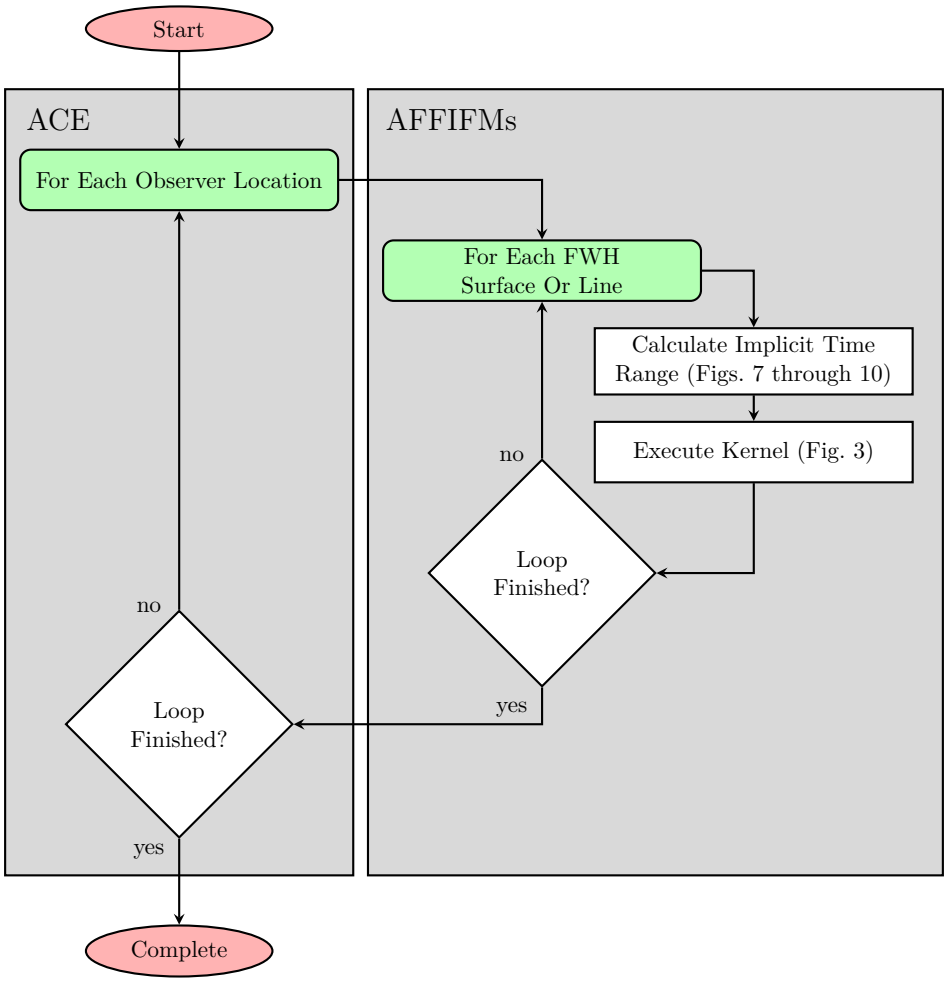


Figure 11: Diagram showing the iterations over observer position and FWH surfaces or lines.

as the computational resources (computational processors available) to limit the loops shown in Fig. 11 [19,20]. AMPIT also allows for accumulating the results back onto a single rank for analysis, export, and/or further computations [18].

Two scenarios where MPI is beneficial are when running many observer locations or a single observer location but a very large source surface. In the first example, AMPIT distributes the observers across the available computational resources and combines the results afterward. In the second example, AMPIT distributions the source surface across the available computational resources and combines the results afterward. See the AMPIT API Reference Manual for more information [18].

3.4.2 Parallelization with OpenMP

OpenMP entails using threads on a single computer to calculate more than one iteration at a time [16]. At the time of writing this manual, only a limited number of AFFIFMs support OpenMP in the source position iteration loop shown in Fig. 3. As such, it is not supported at this time.

3.5 Waypoints

A final important feature of AFFIFMs is the ability to calculate the noise for short periods of time starting at different aircraft flight path waypoint times. Waypoints, such as an aircraft manuever that occurs over many seconds or minutes, can be defined using ANOPP2's Flight Path Data Structure (AFPDS) [30]. In the case of a rotorcraft in steady level flight over many rotor revolutions, the same data can be used to calculate a single rotor revolution of noise centered around the first and last waypoint, separately. This allows for noise predictions centered at several waypoints, offset by waypoint time, which can be analyzed spectrally, and then provided to a propagation algorithm which can account for absorption and ground reflection. Furthermore, if the manuever is not a steady manuever, two or more sets of FWH data can be used by 'skipping' a number of surfaces per waypoint. An example of this is a four bladed rotor where each blade is an FWH surface undergoing a manuever that includes three waypoints. There would be a total of twelve FWH surfaces and the skip number would be four because the second waypoint would require a different set of blade data than the first waypoint. The operation of waypoints is shown schematically in Fig. 12.

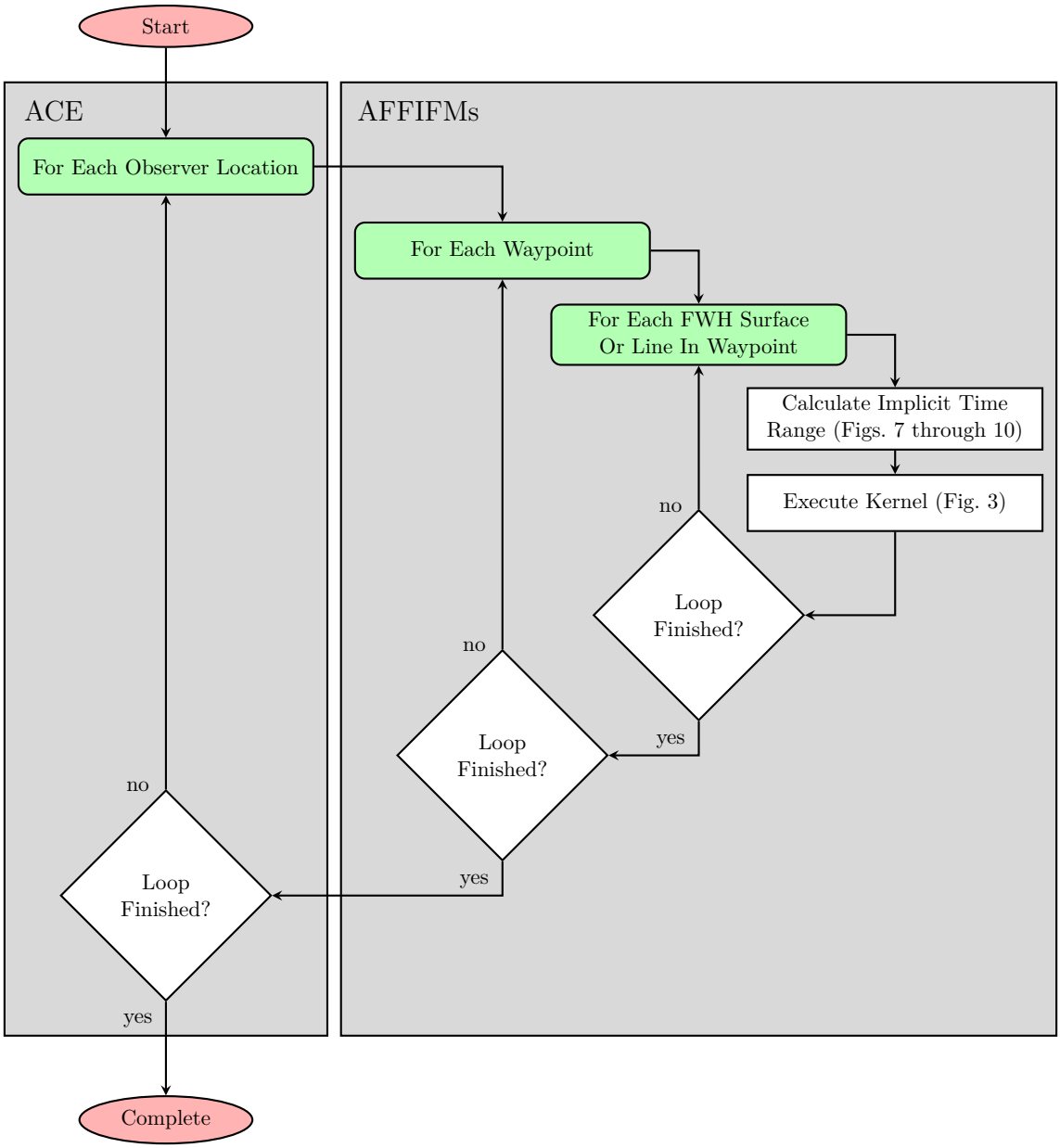


Figure 12: Diagram showing the iterations over observer position and FWH surfaces or lines including waypoints.

4 Interface Description

Unlike the Aircraft NOise Prediction Program (ANOPP), which is distributed as an executable and requires an input file called an ANOPP input deck, ANOPP2 is distributed as a library, the interface into which is defined by an API. This means that a user of ANOPP2 is required to develop a 'user code' in order to execute certain processes that the library can perform. This user code can be written in Fortran, C++, or Python. To execute, AFFIFMs requires certain inputs, such as an FWH surface or line. These can be created using other components of ANOPP2, such as AGDS (see the AGDS Reference Manual for more information) [19]. Other inputs include an atmosphere (defined by ANOPP2's Atmosphere Data Structure (AADS)), a flight path (defined by AFPDS), and an observer (defined by AODS) [20, 30, 31]. In addition to these inputs, executing AFFIFMs requires input files (explained in Section 5). Once all these inputs are created, the user code uses constants and enumerators and calls routines that pass the inputs to the AFFIFMs for execution. These constants, enumerators, and routines are explained in ACE [32]. In addition to those that are supplied in ACE, some AIFMs present their own API. For AFFIFMs, those additional constants, enumerators, and routines are presented here.

4.1 Constants

At this time, AFFIFMs do not provide any additional constants.

4.2 Enumerators

At this time, AFFIFMs do not provide any additional enumerators.

4.3 Routines

At this time, AFFIFMs do not provide any additional routines.

5 Setup And Execution

As mentioned previously, ANOPP2 is a library, an interface (called an API) is provided to the user for use in their user code. Therefore, it is up to the user to create the required inputs and provide them to AFFIFMs to be used during execution. This section of the manual explains how to construct a user code to create the inputs, create AFFIFMs, execute AFFIFMs, and process the results. In the below sections, only Fortran code will be shown. Similar calls can be made in C++ or Python.

When using ANOPP2, all AIFMs within ANOPP2 follow the same general process. The sequence of steps to execute an AIFM include:

1. Initialize the ANOPP2 Library
2. Create Input for AFFIFMs
3. Create AFFIFMs
4. Execute AFFIFMs

5. Postprocess and Export

These steps, including what API routines are used within and what occurs within the ANOPP2 database backend, are shown in Figs. 13a and 13b.

5.1 Step 1: Initialize

The ANOPP2 library requires initialization of the backend, which must occur before any other calls to the ANOPP2 library. This step initializes the backend of ANOPP2, most of which is invisible to the user and not relevant to their operation. There are some parts of the initialization process that the user should be aware of, including initialization of verbosity. To initialize the ANOPP2 library, a call must be made to the initialize routine, an example of which is shown below in Ex. 1.

Example 1: Initializing ANOPP2 in a Fortran user code.

```
! Initialize the backend of ANOPP2.
call a2f_exec_init_api ()
```

5.1.1 Verbosity

The verbosity settings allow a user to monitor the progress of components of ANOPP2, such as ACE, AGDS, and AFFIFMs. This is useful when debugging the binary input files explained in Sec. 5.2. Similar to AFFIFMs, the output of ACE or AGDS, when verbosity is set to *VERBOSE*, can be extensive and the user may wish to suppress those outputs. Included in the verbosity of AGDS are the input data of FWH surfaces or lines and the observer loop shown in Fig. 11. There are three levels of verbosity for all ANOPP2 components: *QUIET*, *STANDARD*, and *VERBOSE*. As their names indicate, these are in order of increasing amount of text written to standard out (usually the terminal if running in a Linux or Mac environment or the command window if running in a Windows environment). To set the verbosity, an environment variable is set for each component, an example of which, in a bash terminal, is shown in Ex. 2, where the `_A2_` indicates an ANOPP2 environment variable and the `_ACE_` indicates the ACE component of ANOPP2.

Example 2: Setting ACE verbosity level in a bash shell to *VERBOSE*.

```
-bash-4.2: export _A2_EXEC_VERBOSITY_=VERBOSE
```

As for the result of setting the verbosity, the *QUIET* verbosity setting for all ANOPP2 components outputs nothing to standard out (completely silent), the *STANDARD* verbosity setting outputs the observer index (Sec. 3.3) under which the integration is occuring. This is shown in Ex. 3. Currently, *VERBOSE* setting in ACE is the same as *STANDARD*.

Example 3: Output from AF1AIFM execution with ACE *STANDARD* verbosity and AFFIFMs and AGDS verbosities set to *QUIET*.

```
ANOPP2's Command Executive API: Version 1.3 (r20043)
 Scanning for Plugins
 No Plugin manifest files were found
 0 Plugin(s) Available
 Calculating the noise on observer geometry index 1 of 1
```

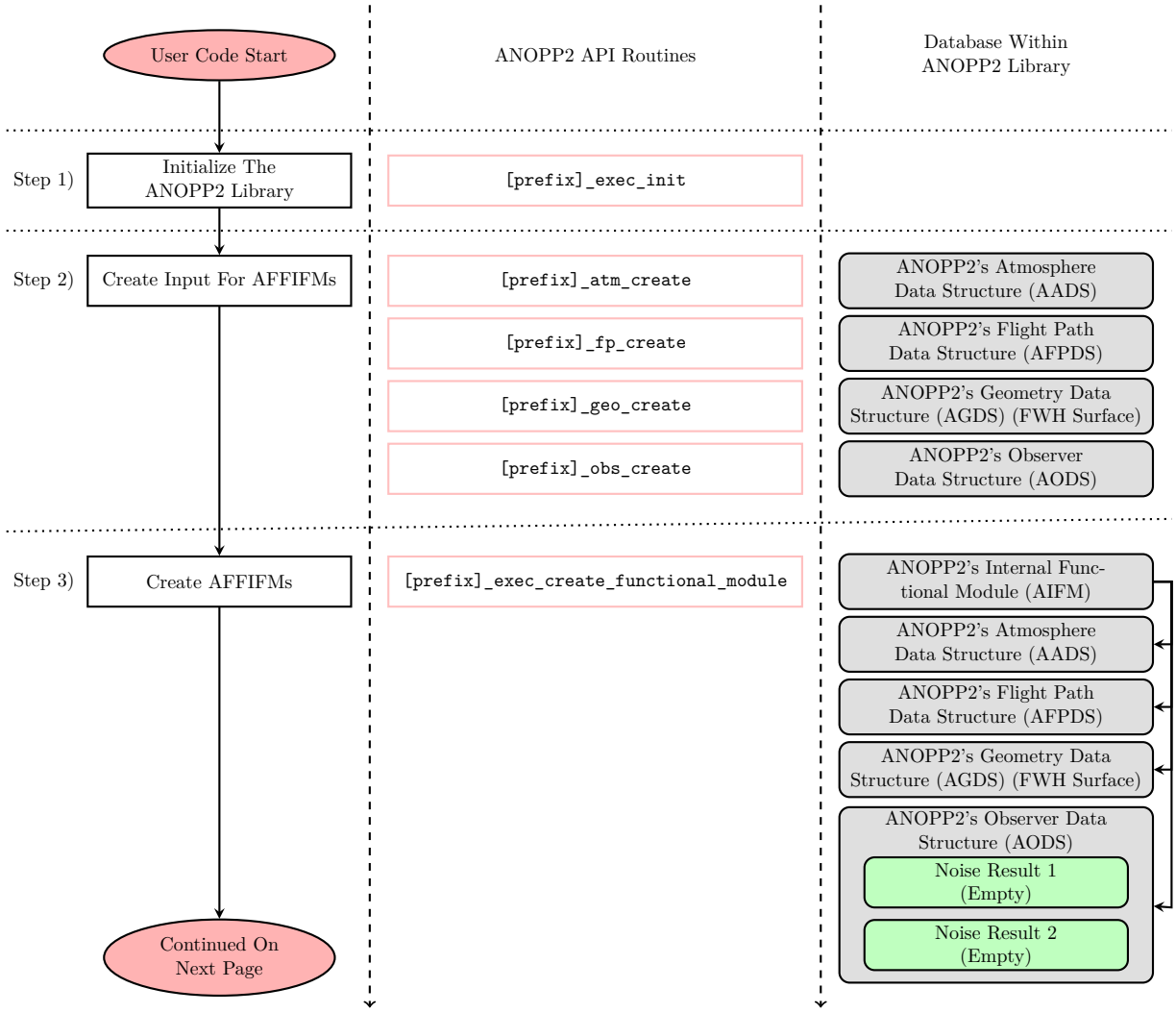


Figure 13a: Diagram showing the general process required in a user's user code in order to execute AFFIFMs. Continued on next page.

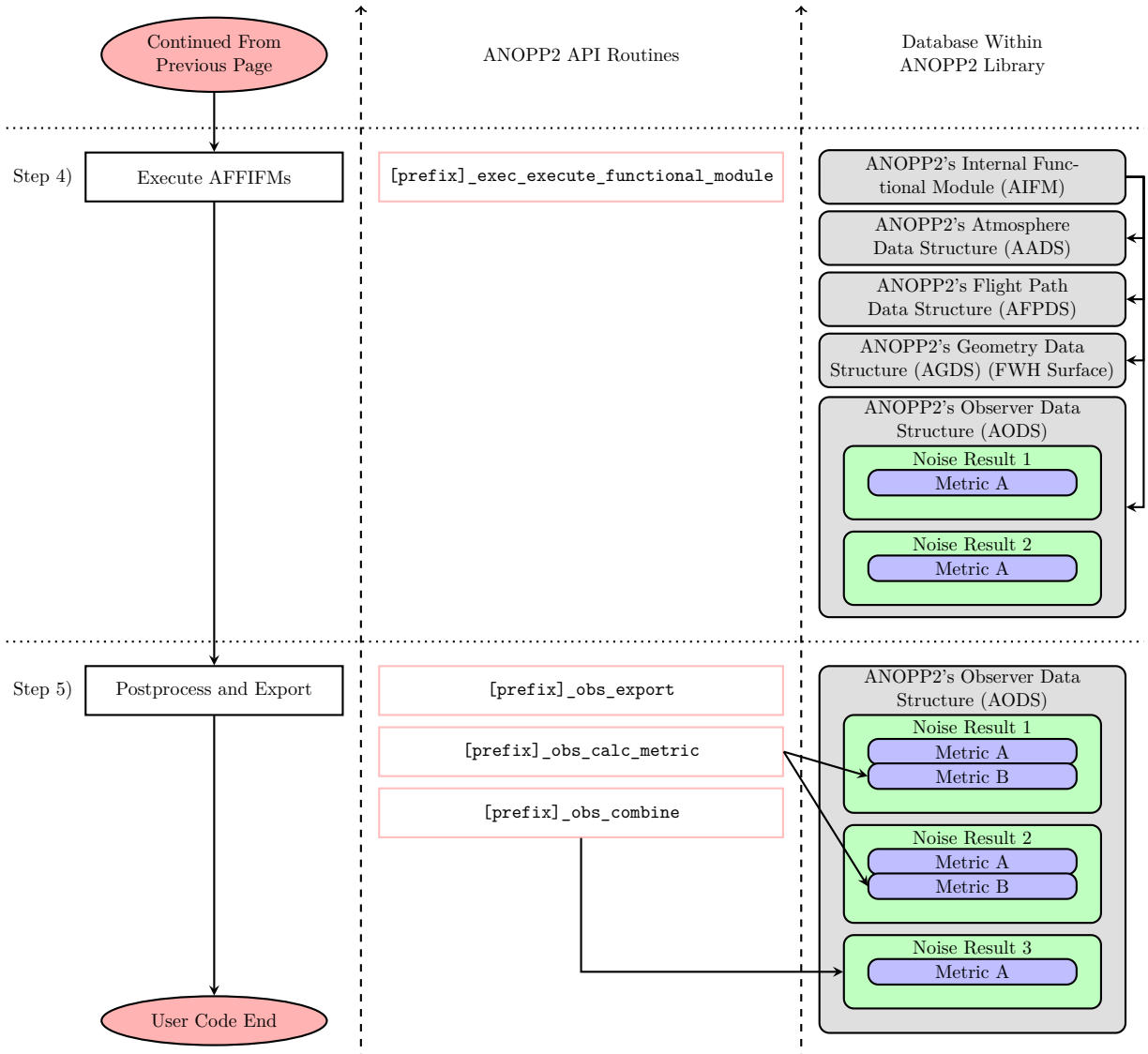


Figure 13b: Diagram showing the general process required in a user's user code in order to execute AFFIFMs. Continued from previous page.

A common problem with executing AFFIFMs is creating a valid binary file defining the FWH surface or line, which is read in by AGDS. The ACE *VERBOSE* setting allows a user to examine the contents of the binary file as it is read in, but it does significantly increase the amount of output written to standard out to the terminal during the startup phase. Example 4 shows what the initalization prints out when the verbosity setting for AGDS is set to *VERBOSE* (by setting the environment variable `_A2_AGDS_VERBOSITY_=VERBOSE`). Specifically, Example 4 shows the output as AGDS reads the binary file, echos the headers in the binary file, and shows how the headers are interpreted, the key time information for both the geometry and flow data, and the boundary box of the surface as a function of key.

Example 4: Output from AF1AIFM execution with AGDS *VERBOSE* verbosity, with the AFFIFM verbosity setting of *QUIET*, and with ACE verbosity set to *STANDARD*.

```
ANOPP2's Command Executive API: Version 1.3 (r20043)
 ANOPP2 Internal Distribution
 Double Precision
 Debugging Library
 Scanning for Plugins
 No Plugin manifest files were found
 0 Plugin(s) Available
 Successfully opened connected geometry file that is big endian.
 Creating surface with header information:
         [42  1  1  3  1  2  2  1  0]
 The version of the surface in the input file is: 1.1
 Time type determined as aperiodic.
 Norm type determined as node based.
 Grid type determined as unstructured.
 Surface will be read in with REAL64.
 Normals will be calculated from the positions.
 Time range of aperiodic geometry: 0.0000000000000000E+00 --> 0.5765730212470423E-01
 Finished initializing an aperiodic time dependence.
 Unstructured grid dimensions: [nNodes, nFaces] [5014, 10024]
 Reading in the connectivity for this unstructured surface.
 Creating data surface with header information:
         [1  1  3  1  2  2  1  1  0  0]
 The version of the data surface in the input file is: 1.1
 Data time type determined as aperiodic.
 Number of data values read in as: 1
 Data on surface will be read in with REAL64.
 Data surface input file contains sequential format.
 Data surface will be extended with the following settings: 1  1  0  0
 Number of data keys: 360
 Time range of aperiodic data: 0.0000000000000000E+00 --> 0.5765730212470423E-01
 Delta time: 0.1606052983975048E-03 s.
 Finished initializing an aperiodic data surface.
 Bounding box of instantaneous surface:
         0.1004636956421382E+01  -0.8283837834188222E-01  0.7867479013686146E+00 -->
         0.2766478475468262E+01  0.6513178972765239E-01  0.8055564252501223E+00
 Creating data surface at index 1 at file position: 280849
 Reading aperiodic data surface at data key 1
 Read aperiodic data surface at data key 1
 Bounding box of instantaneous surface:
         0.1004000978031930E+01  -0.7512029779249869E-01  0.7867479013686146E+00 -->
         0.2767147342109322E+01  0.1000121584118863E+00  0.8055564252501223E+00
 Creating data surface at index 2 at file position: 441301
 Reading aperiodic data surface at data key 2
 Read aperiodic data surface at data key 2
<snip>
```

Similar to the verbosity of ACE and AGDS, the verbosity of AFFIFMs includes *QUIET*, *STAN-*

DARD, and *VERBOSE* and is set via an environment variable. The environment variable is `_A2_AFFIFM_VERBOSITY=VERBOSE` and is set in a similar manner as AGDS and ACE. Example 5 shows representative output when verbosity is set to *STANDARD*, and Ex. 6 shows representative output when verbosity is set to *VERBOSE*.

Example 5: Output from AF1AIFM execution with ACE and AGDS set to *QUIET* verbosity and the AFFIFM verbosity setting of *STANDARD*.

```
Creating a Farassat Formulation 1A solver (AF1AIFM).
Number of passed over geometries is 0 and number of acoustic data geometries is 4
Performing integration for geometry 1 of 4 acoustic data geometries.
Performing integration for geometry 2 of 4 acoustic data geometries.
Performing integration for geometry 3 of 4 acoustic data geometries.
Performing integration for geometry 4 of 4 acoustic data geometries.
```

Example 6: Output from AF1AIFM execution with ACE and AGDS set to *QUIET* verbosity and AFFIFMs verbosity setting of *VERBOSE*.

```
Creating a Farassat Formulation 1A solver (AF1AIFM).
&AF1AIFMNAMELIST
NEMISSIONTIMES =          179,
FLTEMISSIONTIMERANGE     = 2*0.000000000000000E+000   ,
FLTRECEPTIONTIMERANGE    = 2*0.000000000000000E+000   ,
NDATAINDATAGROUP         =           0,
STRMETADATASETTING       = NONE                       ,
FLTRETARDEDTIMETOLERANCE       =  1.000000000000000E-005,
STRMETADATAIDENTIFIER    = Metadata                   ,
NRECEPTIONTIMES =          179,
STRRESULTSDESIRED        = TOTAL                      ,
STRVERBOSITY     =                                ,
BLNCALCULATEONTHEFLY     = F,
STRTIMEHISTORYGROUP      = COMPLETE                   ,
BLNINCLUDESFLIGHTEFFECTS       = 2*T,
NGEOMETRIESPERWAYPOINT   =           4, 999*0,
STRCOORDINATESYSTEM      = BODY                       ,
FLTADDITIONALIMPLICITTIMEFACTOR =  1.000000000000000E-002
/
Calculate noise for observer node 1.
Calculate noise for observer node 1 and waypoint 0.
The waypoint is 0 and number of geometries per waypoint is 4
Number of passed over geometries is 0 and number of acoustic data geometries is 4
Performing integration for geometry 1 of 4 acoustic data geometries.
FWH surface is impenetrable and deforming.
Source Node Range For This Process: [0001, 5014]
Source loop contains 5014 data.
Emission time index 001 of 179 with time: 0.0000000000000000E+00
<snip>
Emission time index 179 of 179 with time: 0.5765730212470423E-01
Performing integration for geometry 2 of 4 acoustic data geometries.
FWH surface is impenetrable and deforming.
Source Node Range For This Process: [0001, 5014]
Source loop contains 5014 data.
Emission time index 001 of 179 with time: 0.0000000000000000E+00
<snip>
```

5.2 Step 2: Create Inputs

Step 2 in Fig. 13a shows four calls to create routines, one for each of the required data structures. Each of the create routines are similar in that they require a configuration file and return a tag that

can be used in subsequent calls in the user code. Descriptions of the configuration files and data structures can be found in their reference manuals and will not be discussed here [19, 20, 30, 31]. Representative calls to the four create routines are shown in Ex. 7.

Example 7: Representative calls to create routines for the required data structures.

```
! Call the routine that will create the atmosphere.
intSuccess = a2f_atm_create (intAadsTag, 'AADS.config')

! Call the routine that will create the flight path from the configuration file.
intSuccess = a2f_fp_create (intAfpdsTag, 'AFPDS.config', intAadsTag)

! Call the routine that will create the observer.
intSuccess = a2f_obs_create (intAodsTag, 'AODS.config')

! Use the ANOPP2 routine to create the FWH data structure.
intSuccess = a2f_geo_create (intAgdsTag, 'AGDS.config')
```

5.3 Step 3: Create AFFIFM

To create an instance of an AFFIFM, a call to an ANOPP2 routine, very similar to the calls in Ex. 7, must occur. This is shown in Ex. 8. The create routine of ACE is used for AIFMs as well as for plugins (more explanation of plugins and the plugin system can be found in Reference [33]). There are some key differences between the create routines shown in Ex. 7 and in Ex. 8. For the overall purpose, inputs, and outputs of the create functional module, refer to ACE manual [32]. Specifically, in the case of AFFIFMs, the input tags `[intAadsTag, intAgdsTag]` include a tag for the atmosphere as well as a list of tags, one for each FWH surface or line. The last argument of the create functional module routine is a list of observer result tags returned by the call. This array is allocated by the routine and has an entry for every observer result returned by AFFIFMs. The number of observer results returned by AFFIFMs depends on the grouping of integrals (desired results) explained in Sec. 2.1.4. This is set in the configuration file, `AFFIFM.config` in Ex. 8, explained next.

Example 8: Representative call to create an AFFIFM.

```
! Create the function module by passing in the appropriate tags.
intSuccess =                                                    &
  a2f_exec_create_functional_module                             &
    (intAffifmTag, 'AFFIFM.config', [intAadsTag, intAgdsTag], &
     intAodsTag, intAodsResultTags)
```

5.3.1 Configuration File

The input in the create functional module call is a configuration file that contains a list of inputs in Fortran namelist form. Example 9 shows a representative configuration file for a call to create an AF1AIFM. Default values are shown in parenthesis.

Example 9: An example configuration file for AF1AIFM.

```
! This portion of the config file tells ANOPP2 what AFFIFM is being created. In this
! example, an AF1AIFM is being created. There is a unique namelist name for each
! AFFIFM, they all follow the same pattern.
&Af1aifmNamelist
  nEmissionTimes              = 179
  nReceptionTimes             = 179
  fltEmissionTimeRange        = [0, 0]
  fltReceptionTimeRange       = [0, 0]
  nInterpolationCoefficients = 3
  blnCalculateOnTheFly        = .FALSE.
  strResultsDesired           = "TOTAL"
  strIntegrandDesired         = ""
  strMetadataSetting          = "NONE"
  strMetadataIdentifier       = ""
  blnFrequencyDomainMetadata = .FALSE.
  fltRetardedTimeTolerance    = 0.00001
  nGeometriesPerWayPoint      = 4
  strTimeHistoryGroup         = "COMPLETE"
  strCoordinateSystem         = "BODY"
/
```

There are many inputs in the configuration file. Below is an itemized list of each input and a brief description.

- `nEmissionTimes` (*-1*)
  The number of time steps for the source surface or line. All surfaces or lines will contain the same number of time steps if this parameter is nonzero (and positive). If this number is 0, then the number of source times is calculated from the time resolution of each surface and the source time range (defined below). This is $n\tau$ in Figs. 7 through 10. The "-1" indicates that this must be set by the user in the configuration file.
- `nReceptionTimes` (*0*)
  This parameter is the number of receptions times. All source noise will be interpolated into a common time range (reception time range defined below) at the resolution defined by this parameter. This is nt in Figs. 7 through 10.
- `fltEmissionTimeRange` (*[0,0]*)
  This is the time range of all surfaces or lines. If this is set to [0, 0], then the source time range is set implicitly by either the observer (first) or by the aperiodic kinematics or flow properties of the surface or line (second). This is $\tau_{\text{u,min}}$ and $\tau_{\text{u,max}}$ in Figs. 7 through 10.
- `fltReceptionTimeRange` (*[0,0]*)
  This is the time range of the observer position. If this is set to [0, 0], then the reception time range is set implicetly by the source time range. This is $t_{\text{u,min}}$ and $t_{\text{u,max}}$ in Figs. 7 through 10.
- `nInterpolationCoefficients` (*3*)
  This is the number of polynomial interpolation coefficients, N, shown in Sec. 3.1.7.
- `fltAdditionalImplicitTimeFactor` (0.01)
  This is the amount of additional time that will be added to the implicit source or reception time range. This is ε in Figs. 7 through 10.
- `blnCalculateOnTheFly` (*.FALSE.*)
  This sets whether the formulation will calculate the time derivatives of the surface or line and flow quantities up front or on an as needed basis. A value of true means only a small window (5 point stencil) in time is stored in memory at any 1 source time step. This significantly reduces the memory required but adds a small amount of overhead. Refer to Fig. 3 for information on calculating on the fly.

– `strResultsDesired` (*TOTAL*)
  This string communicates to the prediction method the number of results desired by the user. There are 6 options: *TOTAL*, *SOURCE*, *COMPONENT*, *GEOMETRY*, *COEFFICIENT*, and *FIELD*. If *TOTAL* is specified, only one acoustic metric time history is provided per surface or line: the total acoustic metric. If *SOURCE* is specificed, two acoustic metrics are provided: monopole and dipole. If one of the *INTEGRAND* options are specified, a number of results are returned: one for each integrand in the formulation. The number of integrands is set by the formulation and the group setting below.
– `strMetadataSetting` (*NONE*)
  The metadata setting is the level that determines the amount of information that is written to a metadata output file. Options are *NONE*, *BASIC*, *GEOMETRY*, *SOURCE*, and *ALL*. Caution, level of *ALL* may write out a very large data set. The size of the data set is a function of the source size and number of time steps.
– `strMetadataIdentifier` (*"FormulationMetadata"*)
  This is the name of the metadata file that will be placed in the *metadata* folder. This is stored in the formulation until the module is executed. The metadata files are written out in the *metadata* folder with the number of the surfaces appended to the end of the file name (plus .plt for Tecplot binary format) [22]
– `blnFrequencyDomainMetadata` (*.FALSE.*)
  In addition to time dependent data, AFFIFMs have the option to export frequency dependent metadata. If this option is turned on, a Fourier transform of all variables is taken before the data are written to the output files.
– `fltRetardedTimeTolerance` (*0.00001*)
  This is the tolerance when performing a source time calculation. It is ϵ in Fig. 5. A value of 0.00001 should be sufficient in most applications.
– `nGeometriesPerWayPoint` (*0*)
  This is the number of surfaces or lines to skip for each waypoint when doing a several flight path waypoint calculation, explained in Sec. 3.5.
– `strTimeHistoryGroup` (*COMPLETE*)
  All AFFIFMs produce an acoustic metric time history, such as an APTH. These time histories could be for very long flight operations, such as a flyover event, or for relatively short durations, such as a steady state condition along a flight path defined for many waypoints. A long duration flight operation can be separated into short segments, each of which can have signal processing operations performed. When doing a long duration flight operation, the acoustic metric is said to be defined for the 'complete' time history. If using many waypoints, then the acoustic metric is defined at a segment. The options for this setting are *COMPLETE* and *SEGMENT*.
– `strCoordinateSystem` (*""*)
  Results within ANOPP2 can only be summed if they are in the same coordinate system. For instance, an aircraft engine noise source may be defined in the aircraft frame (or body frame) but an airframe noise source, such as landing gear, is typically defined in the wind frame. The purpose of this entry in the configuration file is to tag the result's coordinate system. Options include *BODY*, *WIND*, or *GROUND*.
– `strWindowFunction` (*NONE*)
  The windowing function applied to the time domain acoustic velocity potential before a Fourier transform in AF2BIFM in the postprocessing step of the kernel (explained in Sec. 3.1.9). Options include *NONE*, *FLAT TOP*, *HAMMING*, *BLACKMAN*, and *HANNING*. See AAAU reference manual for more information on windowing [29].

.

5.4 Step 4: Execute AFFIFM

The call to perform the noise prediction, Step 4 in Fig. 13a, is straightforward. Example 10 shows the call to ACE routine that executes a functional module.

Example 10: Representative call to execute an AFFIFM.

```
! Execute the functional module.  This performs the noise prediction and inserts
! the noise into the associated observer data structure.
intSuccess = &
  a2f_exec_execute_functional_module (intAffifmTag, intAadsTag, intFpdsTag)
```

5.5 Step 5: Postprocess and Export

Once AFFIFMs have been executed, AODS contains the noise results generated by the execution. Step 5 in Fig. 13a includes postprocessing and exporting of acoustic results. These operations occur independent of AFFIFMs and are accomplished by interacting with AODS directly. Tags from creating AODS and result tags from creating the AFFIFMs are passed back to AODS for operation. A common operation is to combine results, for example, combining thickness (monopole) and loading (dipole) noise into a total noise. Example 11 demonstrates how AODS tag and the observer result tag can be used to create a total observer result. Example 12 demonstrates how AODS can be used to export a result into a Tecplot formatted file [22]. Example 13 demonstrates how AODS can be used to calculate a derived metric, such as a power spectral density spectrum (PSD) from an APTH. See AODS reference manual for more information on the combine, calculate metric, and export routines [20].

Example 11: Representative call to combine APTH into a new observer result.

```
! Call the combine results function in ANOPP2's Observer Data Structure to sum up
! each blade contribution to a total.
intSuccess =              &
  a2f_obs_combine_results &
    (intAodsTag, intAodsResultTags, a2_aa_apth, 'Total', intAodsTotalResultTag)
```

Example 12: Representative call to export an APTH from an observer result.

```
! Example of exporting an observer result to a Tecplot formatted file.
intSuccess =                                        &
  a2f_obs_export                                    &
    (intAodsTag, intAodsResultTags(1), 'export.dat', &
     a2_aa_apth, a2_local, a2_formatted, a2_tecplot)
```

Example 13: Representative call to calculate a PSD from an APTH.

```
! Example of using the AODS to calculate a derived metric.
intSuccess = &
  a2f_obs_calc_metric (intAodsTag, intAodsResultTags, a2_aa_psd, 'COMPLETE')
```

6 Demonstration

This chapter provides two demonstrations of using AFFIFMs to predict an APTH. Both examples presented here are for rotorcraft cases. The first uses a lower fidelity method to calculate the blade loads and compact lifting lines for a three bladed rotor. The second uses CFD data and impermeable FWH surfaces for a four bladed rotor. Code snippets of some steps during the prediction will be inserted in the text here for reference but can be found in the demonstrations directory of the ANOPP2 distribution. Even though Fortran code snippets are shown here, the demonstration codes are provided also in C++ and Python. Both demonstrations show results of the acoustic predictions as well as metadata. All data files exported by these demonstrations are in Tecplot format. This manual assumes that the user is familiar with Tecplot. For additional help with Tecplot, please refer to the Tecplot User's Manual [22].

6.1 Hovering Rotor With Ideally Twisted Blades Using BEMT

The first demonstration case presented here is for a hovering three bladed rotor with ideally twisted blades consisting of National Advisory Committee for Aeronautics (NACA) 0012 airfoil cross sections. This demonstration will utilize the compact form of AF1AIFM, which requires the cross sectional area and force on each section of the blades. These properties could be provided from a comprehensive analysis code, such as CAMRAD II [34], but, for demonstration purposes, a reduced order model can be used to signficantly reduce the complexity of the blade sectional force computation. Estimating the blade forces follows Chapter 3 of Leishman [35] where blade element momentum theory (BEMT) is used to estimate the coefficient of lift on the rotor, shown in Eq. 59, where C_{l_α} is 2π. The coefficient of drag is determined using an empirical model based on Bailey and Gustafson, shown in Eq. 60 [36]. The cross sectional areas are analytically derived from integrating the definition of the NACA 0012 airfoil, shown in Eq. 61, taken from Reference [37].

$$C_l(r) = C_{l_\alpha}\alpha_e = \frac{4C_T}{\sigma r} \tag{59}$$

$$C_d = 0.0087 - 0.0216\alpha_e + 0.4\alpha_e^2 \tag{60}$$

$$y_t = 5t(0.2969\sqrt{x} - 0.1260x - 0.3516x^2 + 0.2843x^3 - 0.1015x^4) \tag{61}$$

The above equations require thrust coefficient and solidity. For this example, we will be using a thrust coefficient of 0.008 and a solidity of 0.1. Example 14 shows a code snippet from the three bladed demonstration case provided with the ANOPP2 distribution. In the example, chord lengths, cross sectional areas, effective angles of attack, and lift, drag, and span forces are calculated for every blade segment. Since the rotor is in hover, each azimuthal station will utilize the same blade section properties. This information is imported into AGDS through a series of routine calls shown in Exs. 15 and 16. This results in a single AGDS for the first blade. The last step in the setup of the inputs required by AFFIFMs is creating the other two blades by using the copy routine in AGDS. Since all three blades have the same data, just offset in space and time, one blade is created and the other two are copied by offsetting spatially and temporally around the rotor. This is shown in Ex. 17 which is, again, taken from the three bladed rotor demonstration. The following two steps are similar to those shown in Exs. 8 and 10.

Example 14: Calculation of the blade sectional properties required by the compact form of AFFIFMs.

```
! For this blade, all the cross sectional areas are the same.  This was found by
! algebraically integrating the equation for a 0012 airfoil and scaling by chord squared.
fltCrossSectionalAreas =      &
  (fltChordLengths(1) ** 2) * &
  (1.2_A2_RK *                &
     (0.2969_A2_RK * 2 / 3 -  &
      0.1260_A2_RK / 2 -      &
      0.3516_A2_RK / 3 +      &
      0.2843_A2_RK / 4 -      &
      0.1015_A2_RK / 5))

! The effective angles of attack are found from the thrust coefficient, solidity,
! 2 * pi * alpha lift curve slope assumption and the normalized radial position.
fltEffectiveAngleOfAttacks =                                  &
  ((4 * fltThrustCoefficient) / (fltSolidity * 2 * a2_const_pi)) * &
  (fltSpanLineCoordinates / fltRadius)

! Calculate the drag and lift force.  Since this is an ideal twist rotor, the
! coefficients of lift and drag are constant for the whole blade.  We can then
! calculate the force with the chord length and blade section velocity.  We are
! going to assume a 2 pi alpha lift curve slope and 0.0087 - 0.0216 * alpha +
! 0.4 * (alpha ** 2) for the drag coefficient.
fltLiftCoefficients = 2 * a2_const_pi * fltEffectiveAngleOfAttacks
fltDragCoefficients =                          &
  0.0087_A2_RK -                               &
  0.0216_A2_RK * fltEffectiveAngleOfAttacks + &
  0.4000_A2_RK * (fltEffectiveAngleOfAttacks ** 2)

! Now that we have the lift and drag coefficients, we can calculate the forces.
! First calculate the dynamic pressures.
fltDynamicPressures = &
  0.5_A2_RK * fltAmbientDensity * (fltOmega * fltSpanLineCoordinates) ** 2

! Calculate the drag, lift, and span forces.
fltLiftForces = fltDynamicPressures * fltChordLengths * fltLiftCoefficients
fltDragForces = fltDynamicPressures * fltChordLengths * fltDragCoefficients
fltSpanForces = a2_const_zero

! Collect the blade properties into a single array.  Negate the forces since we need
! the force on the fluid, not the force on the blades.
fltProperties(1,:) = fltCrossSectionalAreas
fltProperties(2,:) = -fltDragForces
fltProperties(3,:) = -fltSpanForces
fltProperties(4,:) = -fltLiftForces
```

Example 15: Part 1: Example of exporting the sectional forces and cross sectional areas to an AGDS using from memory routines.

```
! First step to create an AGDS from memory is to create an empty data structure.  This
! is done via the create from memory with header routine.  We must communicate what
! type of AGDS we are trying to create.  This takes in a series of arguments. 1) first
! is the type of AGDS we are going to create.  In this case, it is an acoustic data line.
! Other options include a2_geo_impermeable_data_surface or a2_geo_line.  Next argument
! is the time dependence of the nodal locations on the AGDS.  For this case, we are
! constant and use the AKU enumerator a2_kine_constant.  Next is where the data on
! the AGDS will be located.  This is a node-based structure.  The other option is for a
! a2_geo_cell_based line.  And lastly, structuredness of the AGDS.  This argument
! is meaningless for a line, but if this AGDS was a surface, we could have a
! structured (a2_geo_structured) or unstructured (a2_geo_unstructured) surface.
intSuccess =                                &
  a2f_geo_create_from_memory_with_header &
    (intLiftingLineTags(1),              &
     a2_geo_acoustic_data_line,          &
     a2_kine_constant,                   &
     a2_geo_node_based,                  &
     a2_geo_structured)

! The next step to augment this AGDS with time information about the nodal locations.
! In this example, we are constant, so the time is [0,0] (or nonexistent).  We could
! have a time range like [0,T] where T is a period of a periodic AGDS, or [-5,10]
! for an aperiodic AGDS.  Within this time range, a number of keys (or time steps)
! is specified.  This is constant, so this value is 1.
intSuccess =                                  &
  a2f_geo_augment_from_memory_with_grid_time &
    (intLiftingLineTags(1), 1, a2_const_zero * [1, 1])

! Now that time is specified, we augment the AGDS with the grid.  This is done by
! passing an array of values for each grid key (only 1 here).  Since this is a
! constant AGDS, we only do this for the first (and only) grid key.
intSuccess =                                  &
  a2f_geo_augment_from_memory_with_positions &
    (intLiftingLineTags(1), 1, fltGridPositions)

! Next step is to augment the AGDS with data (cross sectional area and force as a
! function of time).  This is done by first specifying data time information, similar
! to what was done for grid time.  The first argument is the number of data keys
! (azimuth stations here).  The next argument is the number of data values (or
! parameters that will be cast on the surface).  Here, this is 4 for force (3) and
! cross sectional area (1).  The next argument is the time dependence of that data
! cast on the surface.  Here, it is periodic information.  And finally, the time range
! of that data is periodic from 0 to period.
intSuccess =                                  &
  a2f_geo_augment_from_memory_with_data_time &
    (intLiftingLineTags(1),                  &
     nAzimuthStations,                       &
     4,                                      &
     a2_kine_periodic,                       &
     [a2_const_zero, fltPeriod])
```

Example 16: Part 2: Example of exporting the sectional forces and cross sectional areas to an AGDS using from memory routines.

```
! The next step, similar to the grid positions, we have to augment the AGDS with the
! cross sectional area and forces.  This is done similar to the grid positions but,
! this time, the informatin is time dependent, so we call the augment function for every
! radial station.  This is a constant blade, the properties for azimuth are the same,
! but for demonstration purposes, we are going to make it time dependent by using
! the same data at every time step.
do iAzimuthStation = 1, nAzimuthStations

  ! Augment the AGDS with data at this azimuth station.
  intSuccess =                                &
    a2f_geo_augment_from_memory_with_data &
      (intLiftingLineTags(1), iAzimuthStation, fltProperties)

end do

! The last step before we have a complete AGDS is to add kinematics (or motion) to the
! structure.  This is done by passing a kinematics tag to the structure.  We first
! have to create the kinematics.  This routine creates an empty list and returns a tag
! associated to a kinematics frame change list.
intSuccess = a2f_kine_load (intAkuTag, a2_kine_trivial, '')

! Now that we have an empty list, we are going to append a single frame change.  This
! frame change is a polynomial frame change that includes a second coefficient (velocity)
! angle.  This is the omega of the blade around the rotor.  The append function takes in
! a list of translation coefficients (empty in this case), an axis of rotation, which will
! be [0, 0, 1] for the z axis, a list of coefficients for angles (0, omega), and a flag
! that tells the AKU to use this for any periodic frame changes that may follow (there
! will be none in this case but, if there were, this would be the frame change to use
! for periodicity.
intSuccess =                                          &
  a2f_kine_append_polynomial_frame_change             &
    (intAkuTag,                                       &
     fltEmptyTranslationCoefficients,                 &
     [a2_const_zero, a2_const_zero, a2_const_one], &
     [a2_const_zero, fltOmega],                       &
     .TRUE.)

! And, now that we have the kinematics constructed, we augment the AGDS with it.
intSuccess = &
  a2f_geo_augment_from_memory_with_kinematics (intLiftingLineTags(1), intAkuTag)
```

Example 17: Creating three blades by creating the first blade and copying, with offset, the other two blades.

```
! will allow us to perform the calculation of a rotor with multiple blades by using
! only the values of 1 blade (including computations of temporal derivatives of all
! flow and geometric properties).
CopyBlades: &
do iBlade = 2, nBlades

  ! There are a few things that we have to be aware of when making a copy.  The first is
  ! that the geometry is time dependent, so we have to offset the grid and data.  Since
  ! the grid is time independent, there is no offset (0.0).  However, the data are
  ! offset by ((iBlade - 1) / nBlades) * fltPeriod.  There is also a global position
  ! offset around the rotor of 2 pi * (iBlade - 1) / nBlades.
  intSuccess =                                                &
    a2f_geo_copy_ads                                          &
      (intLiftingLineTags(iBlade), intLiftingLineTags(1), &
       (REAL (iBlade - 1, A2_RK) / nBlades) * fltPeriod,  &
       a2_const_zero,                                         &
       2 * a2_const_pi * REAL (iBlade - 1, A2_RK) / nBlades)

end do CopyBlades
```

After AF1AIFM has been executed, the APTHs stored in AODS, include the monopole and dipole APTHs, for each AGDS given to AF1AIFM. In this case, an AGDS was provided for each blade (three), meaning there are a total of six APTHs now available in AODS. The three bladed rotor demonstration case presented here exports each of the APTH into a file, sums all APTH, and exports the total to a file. This is shown in Ex. 18.

Example 18: Exporting monopole and dipole APTHs from each blade, summing all APTHs, and exporting the total APTH.

```
! Loop over each blade and export the monopole and dipole acoustic pressure terms, and
! export the geometry
ExportBladePressure: &
do iBlade = 1, nBlades

  ! Set the filename to include the blade number for monopole noise.
  WRITE (strExportFileName, '(A6,I1,A17)') 'Blade.', iBlade, '.monopole.out.dat'

  ! Now export the monopole pressure for this blade.
  intSuccess =                                                                    &
    a2f_obs_export                                                                &
      (intObserverTag, intResultTags((iBlade - 1) * 2 + 1), TRIM (strExportFileName), &
       a2_aa_apth, a2_local, a2_formatted, a2_tecplot)

  ! Do the same for the dipole noise file name.
  WRITE (strExportFileName, '(A6,I1,A15)') 'Blade.', iBlade, '.dipole.out.dat'

  ! And export the dipole pressure for this blade.
  intSuccess =                                                                    &
    a2f_obs_export                                                                &
      (intObserverTag, intResultTags((iBlade - 1) * 2 + 2), TRIM (strExportFileName), &
       a2_aa_apth, a2_local, a2_formatted, a2_tecplot)

  ! Write the file name for the geometry export file
  WRITE (strExportFileName, '(A6,I1,A9)') 'Blade.', iBlade, '.geometry'

  ! Export the geometry
  intSuccess =     &
    a2f_geo_export &
      (intLiftingLineTags(iBlade), strExportFileName, a2_formatted, a2_tecplot)

end do ExportBladePressure

! Combine all the results into a total.
intSuccess =              &
  a2f_obs_combine_results &
    (intObserverTag, intResultTags, a2_aa_apth, 'Total', intTotalResultTag)

! Export the total noise.
intSuccess =                                                &
  a2f_obs_export                                            &
    (intObserverTag, intTotalResultTag, strOutputFileName, &
     a2_aa_apth, a2_local, a2_formatted, a2_tecplot)
```

Figure 14 shows the APTH for the monopole and dipole term APTH for the second blade and the total after summing monopole and dipole for all blades. For this case, only monopole term 6 and dipole term 3 of Eq. 38 are nonzero. Figures 15 and 16 show the metadata output from AFFIFMs showing these two terms on the rotor disk.

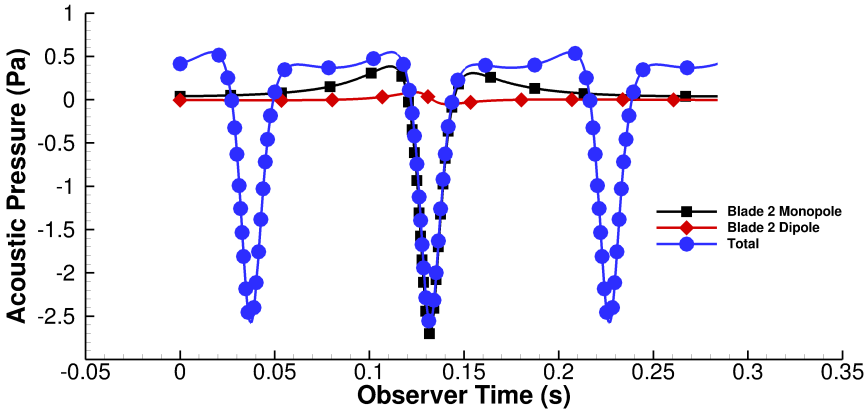


Figure 14: Monopole and dipole APTHs from blade 2. Also shown, the total APTH from all blades.

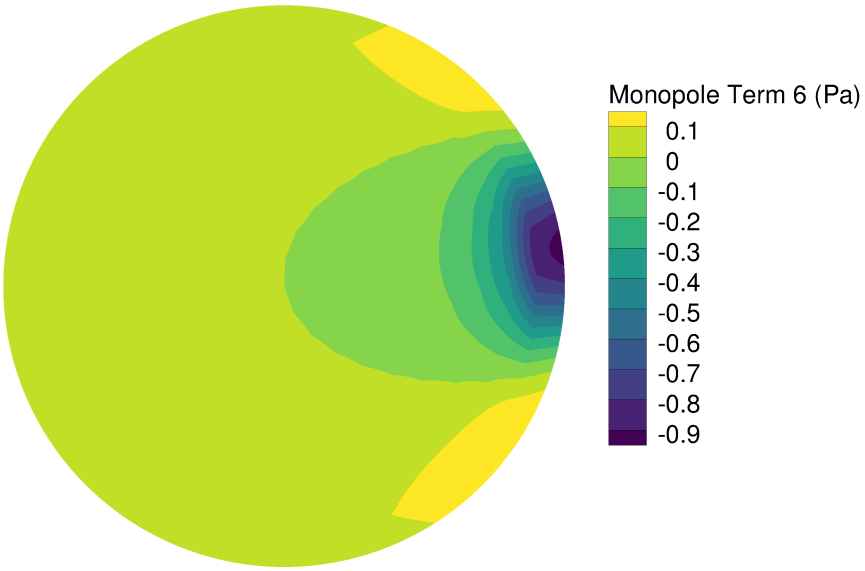


Figure 15: Metadata output showing monopole term across the disk (through source time).

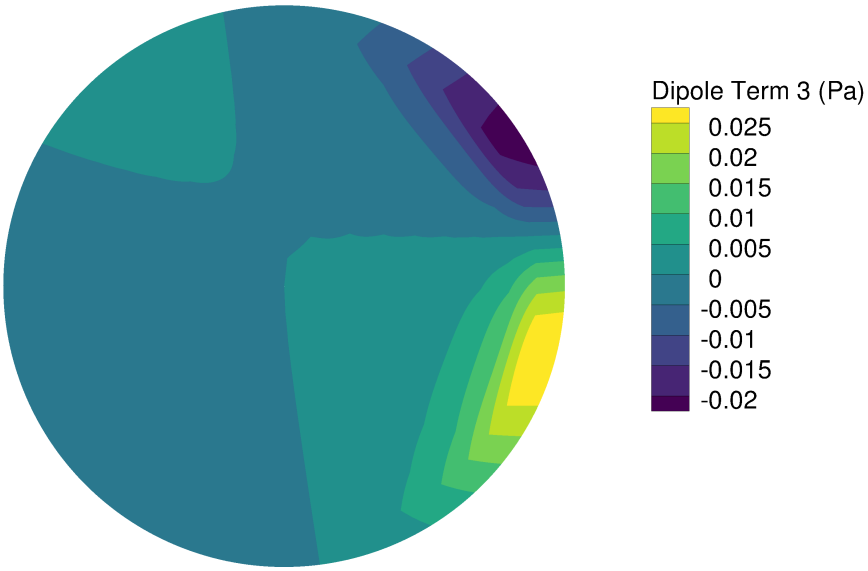


Figure 16: Metadata output showing dipole term across the disk (through source time).

6.2 HART II Noise Prediction Using CFD

The next demonstration case is a noise prediction of the HART-II [38] rotor based on CFD calculations using the FUN3D software [39]. Unlike the demonstration presented in Sec. 6.1, since this demonstration depends on CFD, the binary files are provided to the user and accompany the user code in the ANOPP2 distribution. Also, since the binary files are provided in the ANOPP2 distribution, in order to keep their file size down, the resolution of the CFD solution was reduced and, therefore, the tip vortices generated by each blade are underresolved. This is deemed acceptable since our observer, for this demonstration, is in the rotor plane, avoiding the regions where blade vortex interaction (BVI) is typically dominant.

The HART-II noise prediction using CFD data uses a flow solution that results in an unstructured surface mesh, i.e., the surface is made of many tetrahedrons and the connectivity is provided with the surface pressures and grid locations. See AGDS Reference Manual for more information on structuredness and how to provide structured and unstructured FWH surface data to AFFIFMs [19]. A configuration file is provided to AGDS that communicates the type of surface, whether or not to read the data into memory, and how the surface moves in relation to the AFPDS. Also, unlike the previous demonstration, this demonstration includes a forward flight velocity of the rotor. This is handled through AFPDS, the configuration file of which is shown in Ex. 19, and includes a frame of reference list in which a forward flight velocity is specified. Since this velocity is in AFPDS, all surface motions include the contribution of motion. The rotation of the blades and their offset on the rotor hub is handled by the time dependent node positions of the surface data specified in each binary file.

Figure 17: Example metadata output from HART II demonstration case. Isosurface of observer time, colored by Mach number.

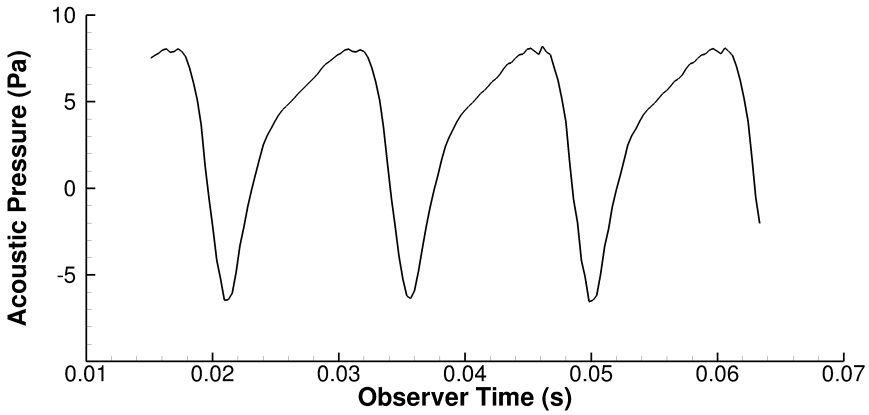


Figure 18: Total APTH from the HART-II noise prediction.

Example 19: Configuration file for AFPDS in which the vehicle is moving forward at a constant velocity.

```
&KinematicsFlightPathNamelist
  nFrameChanges = 1
/
  &PolynomialFrameChangeNamelist
    fltTranslationCoefficient =
    (0,0), (0,0), (0,0), (34.029410657356, 0.000000000000), (0,0), (0,0)
  /
```

Figure 17 shows an isosurface plot in observer time of the metadata output from AF1AIFM. This is the Σ-surface for a given observer time. The surfaces are colored by Mach number. Figure 18 shows the APTH predicted for the HART-II demonstration case.

7 Common Problems, Mistakes, Inaccuracies, and Techniques

AFFIFMs have been used in government, industry, and academia for several years and the author has noticed a set of common issues that users encounter. In order to provide guidance in avoiding some of these common issues, this section notes several common problems, mistakes, and inaccuracies; and suggested techniques to improve predictions and avoid problems are presented.

7.1 Outward Oriented Surface Normal Vectors

One of the most common problems in using AFFIFMs is getting the normal vectors correctly oriented outward into the flow region. The metadata provided by AFFIFMs include the normal vectors. The author strongly suggests checking the setup by plotting normal vectors such as those shown in Fig. 19, which is taken from the second demonstration case in Sec. 6. If the surface data files are very large, consider running at fewer emission times (`nEmissionTimes` setting in the AFFIFMs configuration file) to reduce the amount of data in the metadata files for debugging. Once the surfaces have been adequately verified, increase the number of emission times to their original resolution. If, upon inspection, the normal vectors are pointed inward, AGDS can be used to negate the normals without having to regenerate the normals. See the AGDS Reference Manual for more information [19].

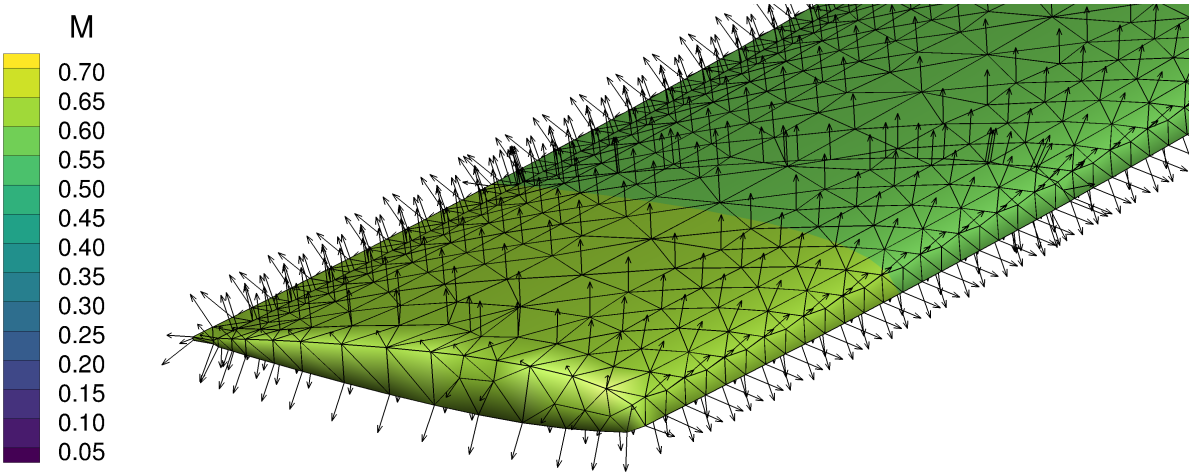


Figure 19: Example metadata output from HART II demonstration case. Isosurface of observer time, colored by Mach number with normal vectors and connectivity included.

7.2 Spurious Signals From Permeable FWH Surfaces

This explanation is derived from Reference [40]. Equation 1 is the general form of the FWH equation, where, for a permeable surface, the source terms are defined by Eq. 3 and include the monopole, dipole, and quadrupole terms, respectively. The right hand side of the FWH equation can be separated into two types of source terms: surface and volume. The surface terms are the combination of monopole and dipole terms and the volume term is the quadrupole term. These are shown in Eqs. 62 and 63. The total acoustic pressure is the sum of the surface and volume

pressures, $p' = p'_s + p'_v$.

$$\bar{\Box}^2 p'_s = \bar{\Box}^2 p'_m + \bar{\Box}^2 p'_d = \frac{\partial}{\partial t}\{Q\delta(f)\} - \frac{\partial}{\partial \boldsymbol{x}_i}\{\boldsymbol{F}_i \delta(f)\} \tag{62}$$

$$\bar{\Box}^2 p'_v = \bar{\Box}^2 p'_q = \frac{\bar{\partial}^2}{\partial \boldsymbol{x}_i \partial \boldsymbol{x}_j}\{\boldsymbol{T}_{ij} H(f)\} \tag{63}$$

Figure 20 shows a diagram of a source geometry causing a wake region and acoustic sources. This is a simple representation of a CFD solution that would be used as input into the FWH equation. The freestream flow velocity is from left to right. The noise is generated by the volume displacement caused by the body (monopole noise), pressures on the physical surface (dipole noise), and from turbulent mixing inside the wake, as well as propagation effects (quadrupole noise). In Fig. 21, the same configuration is shown with a permeable FWH surface that encapsulates the source geometry and a portion of the wake region near the body. The monopole and dipole noise emitted by the permeable surface replaces the noise from the sources inside the surface, including any quadrupoles and propagation effects inside, and hence the sources and wake inside are not shown in the figure. The monopole and dipole noise can be calculated by applying a free space Green's function to a surface integration over the permeable surface, such as Farassat's Formulation 1A [5]. The quadrupole noise generated outside the surface is calculated by a volume integration of the entire region outside the surface. The noise at the observer is the combination of the acoustic pressure from the permeable surface, p'_s, and from the volume, p'_v. The noise everywhere inside the surface, by definition of the generalized functions employed in the derivation of the FW-H equation, is exactly zero (i.e., if the acoustic pressure is calculated inside the surface, the sum of the surface and volume terms equates to exactly zero).

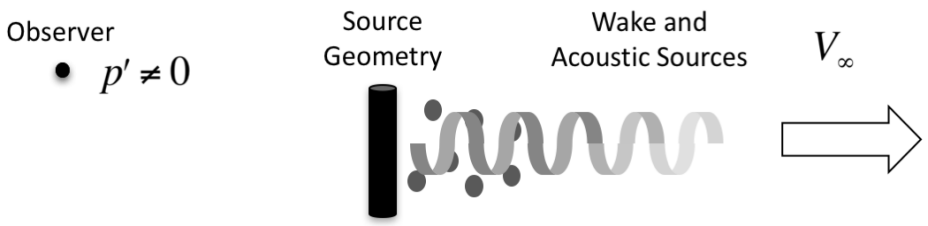


Figure 20: Diagram of source geometry placed in uniform flow generating a wake and acoustic sources. Observer is also shown.

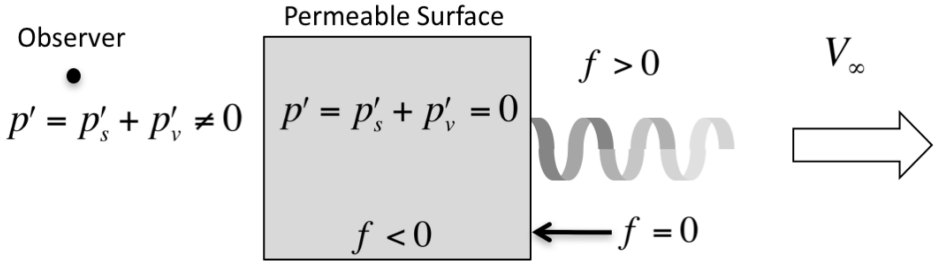


Figure 21: Diagram of permeable surface placed around source geometry and part of the wake and acoustic sources. Noise at an observer is the combination of the surface and volume terms. p' is zero everywhere inside the surface.

The volume term, p'_v, is computationally intensive to calculate. In many FWH solvers, p'_v is neglected with the assumption that its contribution to the noise is small. This assumption is argued to be valid because the majority of the noise generation occurs near the body, which is inside the permeable surface. However, the FWH equation is an exact rearrangement of the Navier-Stokes equations and includes all aerodynamic flow and acoustic phenomena. Because the aerodynamic flow is included in the right hand side of the FWH equation and solutions to the FWH equation employ a free space Green's function, the aerodynamic wake passing through the permeable surface radiates from the surface as an acoustic wave. The volume term, when included, cancels out acoustics that radiate due to the aerodynamics passing through the surface via Lighthill's stress tensor. When the volume term is excluded from the FWH solution, the aerodynamic contribution from the surface source terms is not cancelled out properly. Figure 22 shows a diagram of the case where the volume term is not included in the computation, hence the noise sources and wake outside the surface are not included in the diagram. The noise at the observer is no longer equal to the combination of the surface and volume terms. The noise inside the surface is no longer zero and is equal to only the surface terms. Because the noise inside is zero when all sources are accounted for, when the volume term is not accounted for, the noise inside is equal to the negative of the volume term that was excluded.

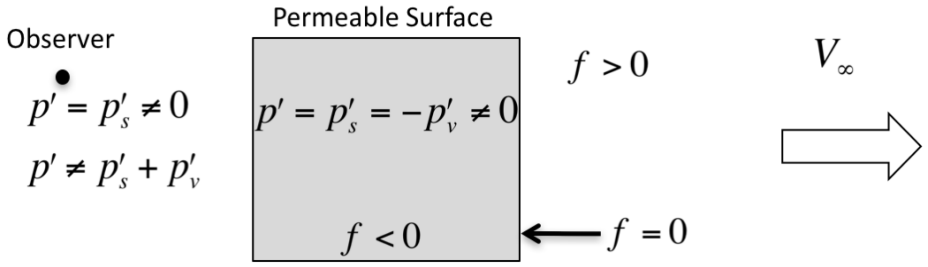


Figure 22: Diagram of permeable surface placed around source geometry without volume term. Noise inside the surface is no longer zero and is equal to the negative of the volume term that was excluded.

The noise predicted from the surface terms includes acoustic sources that are located inside the permeable surface, shown as small circles in Fig. 20, and whose noise at the observer is denoted by

p'_a, as well as contributions from flow that crosses the surface that would have been cancelled by the quadrupole term if it had been included. The noise from the permeable surface caused by the acoustic sources inside the surface is zero inside the surface and nonzero outside. The contributions from the surface that should have been cancelled by the quadrupole term are identified as spurious signals, p'_{ss}. The spurious signals, because they should have been cancelled out by the volume term, are equal to the negative of the quadrupole noise, $p'_{ss} = -p'_v = -p'_q$. This error occurs inside and outside the permeable surface, shown in Fig 23. It is also important to note that the reciprocal is also possible; i.e., if a noise source is outside the FW-H surface, it will radiate inward and be zero everywhere outside. However, this does not change the spurious signal, which will radiate inward and outward.

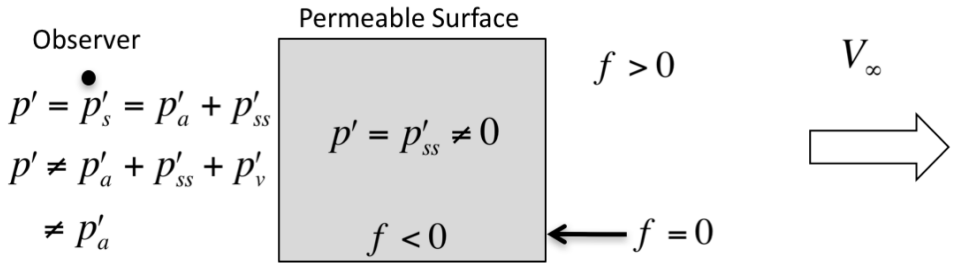


Figure 23: Diagram of spurious signals caused by not including the quadrupole term. These occur inside and outside the permeable surface.

Without predicting the quadrupole noise, the noise outside the permeable surface contains acoustic pressure from the sources inside the surface as well as spurious signals. It is impossible to determine if the acoustic pressure predicted outside the surface contains spurious signals without knowledge of the flow field inside because the noise generation inside the surface is unknown; i.e., it is unclear whether flow phenomena on the surface are acoustic or aerodynamic in nature. However, spurious signals can be identified by predicting the noise from the permeable surface inside the surface where the noise from acoustic sources is calculated as zero. Because the spurious signals are the negative of the quadrupole noise, the nonzero noise inside the surface provides guidance to the inaccuracy incurred when ignoring the quadrupole term. If the quadrupole noise is small, the spurious signal noise inside the surface will also be small. If the noise inside the surface is large, the spurious signal noise is large and the noise predicted outside may contain large amounts of spurious signals.

7.3 Observer and Source Motion for Wind Tunnel Configuration Using CFD

When predicting the noise using AFFIFMs, it is important to remember that the assumption made by Farassat, so that the free space Green's function could be used, is that the surrounding fluid is stationary. This often causes trouble when using CFD in a wind tunnel configuration; that is, the source is stationary and the fluid is moving at a uniform velocity at the external boundaries. This is similar to a wind tunnel environment where the model is stationary and the flow is moving. To correct for this, the source in AFFIFMs must be moving in the opposite direction; that is, the source surfaces or lines must move at the negative of the flow velocity seen at the external boundaries. Also, since the source is now moving, to keep relative position to the source, the observer must also be moved similarly.

7.4 Improving Compact Thickness

Section 2.1.3 explained that the compact thickness formulation can reduce a costly surface integration into a very fast line integration at the cost of accuracy in the noise prediction. This line integral is normally taken at the quarter chord location of the airfoil. However, by increasing the number of line integrals, the accuracy of the thickness noise can be improved. This does increase the computational cost by approximately the number of compact lines; however, the noise results improve in the higher frequencies and are still potentially orders of magnitude less costly than a full surface integration. Figure 24 depicts an NACA 0012 airfoil being sectioned into one, two, and three compact sections, the centers of which are also shown. Each compact point is associated with a different cross sectional area [11]. Figure 25 shows representative noise calcuations in plane of a rotor moving at an advance ratio, μ, of 0.3. Notice how increasing the number of compact line sources from one to three descreases the overprediction in the higher frequencies.

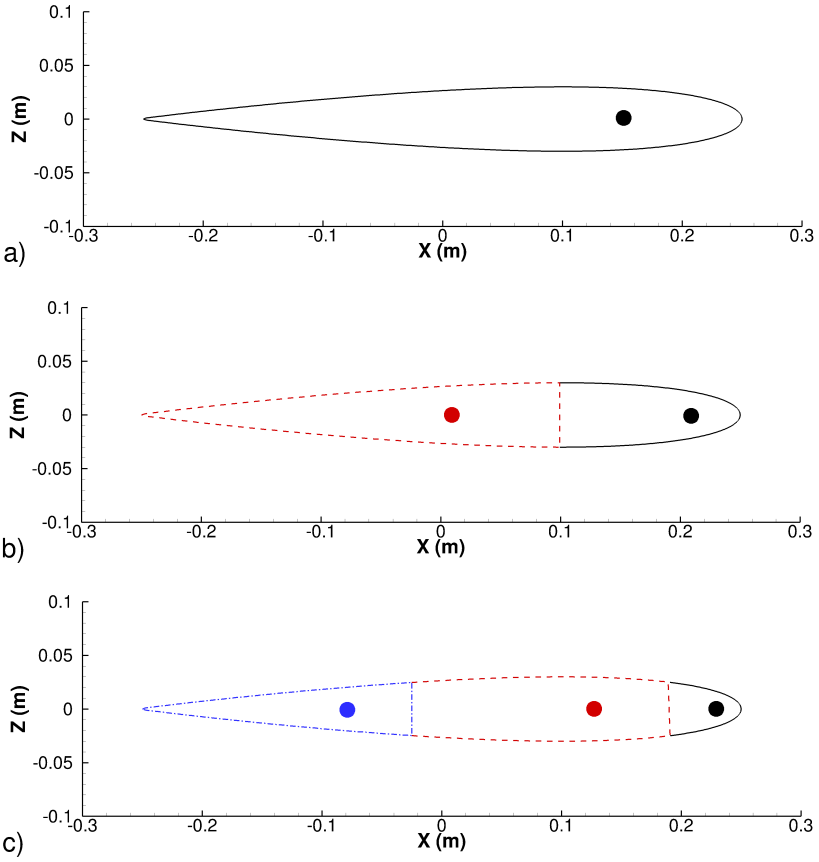


Figure 24: Cross section of an NACA 0012 airfoil with compact line for three compact assumption configurations.

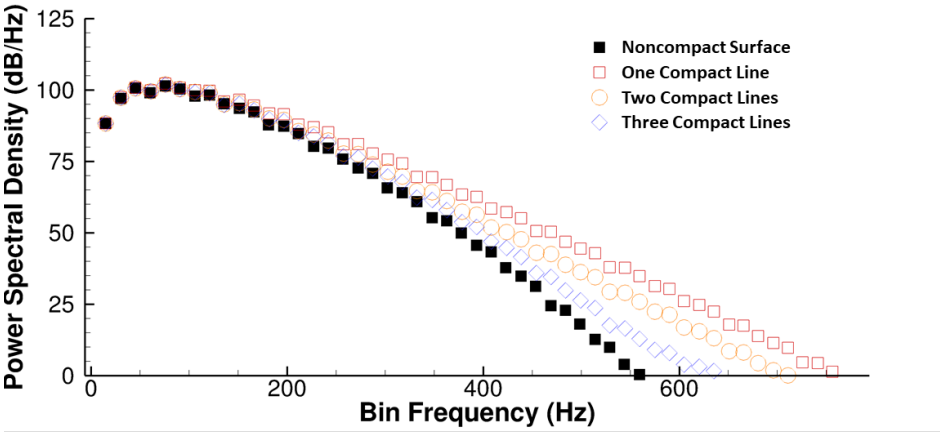


Figure 25: Power spectral density from noncompact and compact predictions of representative rotor with μ of 0.3 and an NACA 0012 airfoil cross section.

7.5 Contents of AGDS Binary Files

A common problem with running AFFIFMs is creating the binary files containing the correct flow data information (i.e., units and/or perturbation quantities). If the FWH surface is permeable, the flow data must contain perturbation density (ρ'), momentum (ρu_i), and perturbation pressure (p') in standard metric units (kg/m^3, kg/m^{2}s, and Pa). If the FWH surface is impermeable, the flow data contain only perturbation pressure (p'). Another common question is whether the permeable flow data contain the free stream flow velocity when calculating momentum. Since the free space Green's function assumes stationary ambient flow, the surface must move at a velocity equal to the negative of the CFD ambient flow velocity. The momentum in the binary file then must include the flow velocity used in the CFD solution.

References

1. 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.

2. Gel'fand, I. M. and Shilov, G. E., *Generalized Functions: Properties and Operations*, Vol. 1, Academic Press, Inc., 11 Fifth Avenue, New York 3, New York, 1964, Translated by Eugene Saletan, Department of Physics, Northeastern University, Boston, Massachuesetts.

3. 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.

4. 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.

5. Farassat, F., "Theory of Noise Generation From Moving Bodies With an Application to Helicopter Rotors," Tech. Rep. NASA TR R-451, National Aeronautics and Space Administration, 1975.

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

7. 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).

8. Lopes, L. V. and Burley, C. L., "ANOPP2's User's Manual," Tech. Rep. NASA/TM-2016-219342, National Aeronautics and Space Administration, 2016.

9. Brentner, K. S., "An efficient and robust method for predicting helicopter rotor high-speed impulsive noise," *Journal of Sound and Vibration*, Vol. 203, No. 1, 1997, pp. 87–100.

10. Lee, S., Brentner, K. S., and Farassat, F., "Analytic Formulation and Numerical Implementation of an Acoustic Pressure Gradient Prediction," *Journal of Sound and Vibration*, Vol. 319, No. 3-5, 2009, pp. 1200–1221.

11. Lopes, L. V., "Compact Assumption Applied to the Monopole Term of Farassat's Formulations," June 22-26, 2015, AIAA Paper No. 2015-2673, presented at the 21^{st} AIAA/CEAS Aeroacoustics Conference.

12. Ghorbaniasl, G., Carley, M., and Lacor, C., "Acoustic Velocity Formulation for Sources in Arbitrary Motion," *AIAA Journal*, Vol. 51, No. 3, 2013, pp. 632–642.

13. Farassat, F. and Casper, J., "Broadband noise prediction when turbulence simulation is available–Derivation of Formulation 2B and its statistical analysis," *Journal of Sound and Vibration*, Vol. 331, 2012, pp. 2203–2208.

14. Najafi-Yazidi, A., Brès, G., and Mongeau, L., "An Acoustic Analogy Formulation for Moving Sources in Uniformly Moving Media," *Proceedings of the Royal Society. A, Mathematical, Physical, and Engineering Sciences*, Vol. 467, 2011, pp. 144–165.

15. Lopes, L. V., *ANOPP2's Kinematics Utility (AKU) Reference Manual (Distributed with ANOPP2)*, National Aeronautics and Space Administration, Hampton, VA 23681.

16. OpenMP, "The OpenMP API Specification for Parallel Programming," `http://openmp.org/`, 2013.

17. Barney, B., "Message Passing Interface (MPI)," `https://hpc-tutorials.llnl.gov/mpi/`, 2013.

18. Lopes, L. V., *ANOPP2's MPI Tool (AMPIT) Reference Manual (Distributed with ANOPP2)*, National Aeronautics and Space Administration, Hampton, VA 23681.

19. Lopes, L. V., *ANOPP2's Geometry Data Structure (AGDS) Reference Manual (Distributed with ANOPP2)*, National Aeronautics and Space Administration, Hampton, VA 23681.

20. Lopes, L. V., Iyer, V. R., and Jones, J. J., *ANOPP2's Observer Data Structure (AODS) Reference Manual (Distributed with ANOPP2)*, National Aeronautics and Space Administration, Hampton, VA 23681.

21. Hennes, C. C. and Brentner, K. S., "The Effect of Blade Deformation on Rotorcraft Acoustics," presented at the 31^{st} European Rotorcraft Forum, September 13–15, 2005.

22. Tecplot, "Tecplot 360," retrieved from `http://www.tecplot.com`, February 2013.

23. Farassat, F., "Derivation of Formulations 1 and 1A of Farassat," Tech. Rep. NASA/TM—2007-214853, National Aeronautics and Space Administration, Langley Research Center, Hampton, VA., March 2007.

24. Farassat, F., Pegg, R. J., and Hilton, D. A., "Thickness Noise of Helicopter Rotors at High Tip Speeds," March 24–26, 1975, AIAA Paper No. 75-453, presented at the 2^{nd} AIAA Aero-Acoustics Conference.

25. Succi, G. P., "Design of Quiet Efficient Propellers," *SAE Business Aircraft Meeting, Society of Automotive Engineers Paper SAE79-0584*, SAE International, April 1979.

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. 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.

28. Rahnejat, H., "Multi-body dynamics: Vehicle, Machines, and Mechanisms," *Professional Engineering Publishing*, Vol. 2, No. 152, 1998, pp. 59.

29. Lopes, L. V., Jones, J. J., and Rawls, J. W., *ANOPP2's Acoustic Analysis Utility (AAAU) Reference Manual (Distributed with ANOPP2)*, National Aeronautics and Space Administration, Hampton, VA 23681.

30. Lopes, L. V., *ANOPP2's Flight Path Data Structure (AFPDS) Reference Manual (Distributed with ANOPP2)*, National Aeronautics and Space Administration, Hampton, VA 23681.

31. Lopes, L. V., *ANOPP2's Atmosphere Data Structure (AADS) Reference Manual (Distributed with ANOPP2)*, National Aeronautics and Space Administration, Hampton, VA 23681.

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

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

34. Johnson, W., "Rotorcraft Aerodynamics Models for a Comprehensive Analysis," presented at the American Helicopter Society 54^{th} Annual Forum, May 20–22, 1998.

35. Leishman, J. G., *Principles of Helicopter Aerodynamics*, Cambridge Aerospace Series, 40 West 20th St., New York, New York, 10011-4211, 2000.

36. F. J. Bailey, J. and Gustafson, F. B., "Charts for Estimation of the Characteristics of a Helicopter Rotor in Forward Flight." Tech. Rep. NACA WR L-110, 1944.

37. Abbott, I. H. and von Doenhoff, A. E., *Theory of Wing Sections*, McGraw-Hill, 1949.

38. van der Wall, B. G., Burley, C. L., Yu, U. H., Pengel, K., and Beaumier, P., "The HART II Test - Measurement of Helicopter Rotor Wakes," *Aerospace Science and Technology*, Vol. 8, No. 4, June 2004, pp. 273—284.

39. Biedron, R. T., Carlson, J. R., Derlaga, J. M., Gnoffo, P. A., Hammond, D. P., Jones, W. T., Kleb, B., Lee-Rausch, E. M., Nielsen, E. J., Park, M. A., Rumsey, C. L., Thomas, J. L., and Wood, W. A., "FUN3D Manual: 13.2," Tech. Rep. NASA/TM-2017-219661, National Aeronautics and Space Administration, Langley Research Center, Hampton, VA., 2017.

40. Lopes, L. V., Boyd, D. D., Nark, D. M., and Wiedemann, K. E., "Identification of Spurious Signals from Permeable Ffowcs Williams and Hawkings Surfaces," May 9–11 2017, presented at the American Helicopter Society 73rd Annual Forum.

41. Morfey, C. L., *Dictionary of Acoustics*, Academic Press, Harcourt Place, 32 Jamestown Road, London NW1 7BY, UK, 2000.

42. U.S. Federal Aviation Administration (FAA), "Noise Standards: Aircraft Type and Airworthiness Certification," Tech. Rep. AC 36–4C, July 2003, retrieved from *http://www.faa.gov*.

43. Committee on Aviation Environmental Protection (CEAP), "Environmental Technical Manual on the Use of Procedures in the Noise Certification of Aircraft," Tech. Rep. Doc 9501 AN/929, International Civil Aviation Organization, July 2003, retrieved from *http://www.icao.int*.

Acronyms and Abbreviations

AAAU . . . . ANOPP2's Acoustic Analysis Utility

AADS . . . . ANOPP2's Atmosphere Data Structure

AATT . . . . Advanced Air Transport Technology

AAVP . . . . Advanced Air Vehicles Program

ACE . . . . . ANOPP2's Command Executive

AF1AIFM . ANOPP2's Formulation 1A Internal Functional Module

AF1IFM . . ANOPP2's Formulation 1 Internal Functional Module

AF2BIFM . ANOPP2's Formulation 2B Internal Functional Module

AFFIFMs . . ANOPP2's Farassat Formulations Internal Functional Modules.

AFG0IFM . ANOPP2's Formulation G0 Internal Functional Module

AFG1AIFM ANOPP2's Formulation G1A Internal Functional Module

AFG1IFM . ANOPP2's Formulation G1 Internal Functional Module

AFPDS . . . ANOPP2's Flight Path Data Structure

AFV1AIFM ANOPP2's Formulation V1A Internal Functional Module

AGDS . . . . ANOPP2's Geometry Data Structure

AIFM . . . . ANOPP2's Internal Functional Module

AKU . . . . ANOPP2's Kinematics Utility

AMPIT . . . ANOPP2's Message Passing Interface Tool

ANOPP . . . Aircraft NOise Prediction Program

ANOPP2 . . Aircraft NOise Prediction Program 2

AODS . . . . ANOPP2's Observer Data Structure

API . . . . . Application Programming Interface

APTH . . . . Acoustic Pressure Time History

ARMD . . . Aeronautic Research Mission Directorate

AVTH . . . . Acoustic Velocity Time History

BEMT . . . Blade Element Momentum Theory

BVI . . . . . Blade Vortex Interaction

CFD . . . . . Computational Fluid Dynamics

CST . . . . . Commercial Supersonic Technology

EPNL . . . . Effective Perceived Noise Level

FAR . . . . . Federal Aviation Regulations

FSC . . . . . Fast Scattering Code

FWH . . . . Ffowcs Williams and Hawkings

HART-II . . Higher harmonic control Aeroacoustics Rotor Test

ICAO . . . . International Civil Aviation Organization

MPI . . . . . Message Passing Interface

NACA . . . . National Advisory Committee for Aeronautics

NBS . . . . . Narrowband Spectrum

OpenMP . . Open Multiprocessing

PGTH . . . . Pressure Gradient Time History

PNL . . . . . Perceived Noise Level

PSD . . . . . Power Spectral Density Spectrum

RVLT . . . . Revolutionary Vertical Lift Technology

SPL . . . . . Sound Pressure Level

SRL . . . . . Software Release Level

TACP . . . . Transformative Aeronautics Concepts Program

TTT . . . . . Transformational Tools and Technologies

Glossary

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).

acoustic velocity time history (AVTH)

A time history of acoustic velocities.

Advanced Air Transport Technology

NASA's vision for advanced fixed wing transport aircraft is revolutionary energy efficiency and environmental compatibility. The overarching goal of the AATT Project is to explore and develop technologies and concepts to enable this vision. (taken from `https://www.nasa.gov/aeroresearch/programs/aavp/aatt`).

Advanced Air Vehicles Program

AAVP studies, evaluates and develops technologies and capabilities for new aircraft systems, and also explores far-future concepts that hold promise for revolutionary air-travel improvements. Innovative AAVP design concepts for advanced vehicles integrate technologies that focus on fuel burn, noise, emissions and intrinsic safety. The goal: to enable new aircraft to fly safer, faster, cleaner, quieter, and use fuel far more efficiently. Partnering with industry, academia, and other government agencies, AAVP pursues mutually beneficial collaborations to leverage opportunities for effective technology transition.

Aeronautic Research Mission Directorate (ARMD)

NASA aeronautics has made decades of contributions to aviation. Every U.S. commercial aircraft and U.S. air traffic control tower has NASA-developed technology on board that helps improve efficiency and maintain safety. Research conducted by ARMD directly benefits today's air transportation system, the aviation industry, and the passengers and businesses who rely on aviation every day. ARMD scientists, engineers, programmers, test pilots, facilities managers and strategic planners are focused on aviation's future. They design, develop and test advanced technologies that will make aviation much more environmentally friendly, maintain safety in more crowded skies, and ultimately transform the way we fly. (`https://www.nasa.gov/aeroresearch/about-armd`).

Aircraft NOise Prediction Program (ANOPP)

A NASA computer program to predict aircraft component and system noise. ANOPP has been implemented in three ANOPP2 functional modules.

Aircraft NOise Prediction Program 2 (ANOPP2)

Second generation ANOPP. ANOPP2 employs a mixed-fidelity framework to incorporate methods of differing fidelity allowing semiempirically-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.

ANOPP input deck

A text file containing ANOPP control statements, which are read by ANOPP to direct the ANOPP run.

ANOPP2's Acoustic Analysis Utility (AAAU)

One of the suite of utilities offered by ANOPP2. This utility provides many common acoustic metric calculations such as effective perceived noise level (EPNL), perceived noise level (PNL), windowing, segmenting, filtering, and Fourier transforms [29].

ANOPP2's Atmosphere Data Structure (AADS)

One of the suites of data structures offered by ANOPP2. This data structure provides many common atmospheric properties such as ambient temperature, dynamic viscosity, and speed of sound, all as a function of time and space. The atmospheric properties can be defined as uniform, functions of altitude, or any other configuration of time and space [31].

ANOPP2's Command Executive (ACE)

The command executive allows for the creation, execution, and organization of ANOPP2 internal functional modules (AIFMs and plugins). ACE includes the ability to group internal functional modules into a mission for execution or to execute them individually. ACE also includes a plugin system that assists the user in creating function modules (plugin) [32].

ANOPP2's Farassat Formulations Internal Functional Modules (AFFIFMs)

Provides capability to calculate APTH, pressure gradient time history (PGTH), and acoustic velocity time history (AVTH) as functions of one or more permeable or impermeable FWH surfaces [6, 10–12].

ANOPP2's Flight Path Data Structure (AFPDS)

One of the suites of data structures offered by ANOPP2. This data structure provides parameters as a function of the source time representing way points. These parameters include angle of attack, Mach number, throttle setting, flap settings, gear settings, plus the ability for the user to define new parameters. This data structure also provides flight time parameters, which occur at or close to 0.5 second intervals, including kinematic parameters such as center of gravity location, aircraft velocity, and aircraft orientation (body to wind and earth to body angles) [30].

ANOPP2's Formulation 1 Internal Functional Module (AF1IFM)

Provides capability to calculate APTH as a function of one or more permeable or impermeable FWH surfaces using Farassat's Formulation 1 [6].

ANOPP2's Formulation 1A Internal Functional Module (AF1AIFM)

Provides capability to calculate APTH as a function of one or more permeable or impermeable FWH surfaces using Farassat's Formulation 1A [6].

ANOPP2's Formulation 2B Internal Functional Module (AF2BIFM)

Provides capability to calculate PSD as a function of one or more permeable or impermeable FWH surfaces using Farassat's Formulation 2B [13].

ANOPP2's Formulation G0 Internal Functional Module (AFG0IFM)

Provides capability to calculate PGTH as a function of one or more permeable or impermeable FWH surfaces using Farassat's Formulation G0 [11].

ANOPP2's Formulation G1 Internal Functional Module (AFG1IFM)

Provides capability to calculate PGTH as a function of one or more permeable or impermeable FWH surfaces using Farassat's Formulation G1 [10].

ANOPP2's Formulation G1A Internal Functional Module (AFG1AIFM)

Provides capability to calculate PGTH as a function of one or more permable or impermeable FWH surfaces using Farassat's Formulation G1A [10].

ANOPP2's Formulation V1A Internal Functional Module (AFV1AIFM)

Provides capability to calculate AVTH as a function of one or more permeable or impermeable FWH surfaces using Farassat's Formulation V1A [12].

ANOPP2's Geometry Data Structure (AGDS)

One of the suites of data structures offered by ANOPP2. This data structure provides geometric and data quantities on a geometry consisting of one or more nodes. The geomety can be a single point, line of points, surface, volume, or point cloud. In the case of line, surface, and volume geometric configurations, the list of points can be structured (implied connectivity) or unstructured (explicitly defined connectivity). The points within the data structure can be moving (independently of each other) as a function of time, periodically or aperiodically. Any reasonable number of data variables can be defined on the geometry and are also functions of time (independent of the geometry). The data variables can be defined at the node locations or between them (at the face, cell, etc.). Geometrically derived quantities can be calculated, such as normals, cell area, line length, volume element, as well as the velocity, acceleration, jerk, and snap time derivatives of those variables, plus node position. Those time derivatives are also available for the data variables. The AGDS can be used to defined FWH surfaces (such as permeable or impermeable FWH surfaces) [19].

ANOPP2's internal functional module (IFM)

Functional modules distributed as inseparable pieces within the ANOPP2 library. The internal functional modules are avialable without having to link in an additional library (as is the case for plugins).

ANOPP2's Kinematics Utility (AKU)

One of the suites of utilities offered by ANOPP2. This utility provides many common kinematics and reference frame functionalities such as defining a reference frame with respect to another frame, evaluating a coordinate transformation as a function of time, defining position,

velocity, acceleration, jerk, and snap (position and rotation) of a local frame coordinate in a global frame and vice versa [15].

ANOPP2's Message Passing Interface Tool (AMPIT)

A capability provided with the ANOPP2's framework to facilitate parallel computations via a message passing interface (MPI). The AMPIT tool is provided to a user via source code so that the user may compile in their unique environment. Once compiled into a library, the user can utilize the functions available in conjunction with the ANOPP2 framework to facilitate parallel computations.

ANOPP2's Observer Data Structure (AODS)

One of the suites of data structures offered by ANOPP2. This data structure provides location and one or more acoustic metrics for one or more observer locations. The data structure may consist of a suite of nodes similar to AGDS. Many acoustic metrics (defined in AAAU) can be added (or cast) onto the observer data structure. Examples of an AODS include a single observer location on the ground containing EPNL, many observer locations constituting a large surface on the ground containing time histories of 1/3-Octave sound pressure level (SPL) spectra, or a hemisphere of locations surrounding an aircraft following the flight path with time histories of narrowband spectra (NBSs), PSD spectra, APTHs, and 1/3-Octave SPL spectra. In addition to these metrics, the change in level metrics defined in AAAU can also be cast. There is also the ability to define the frame of reference, calculate acoustic metrics from other metrics, add results together, and leverage MPI capability [20].

aperiodic frame change

A kinematic frame change that varies, in an aperiodic manner, as a function of time.

application programming interface (API)

A set of rules or protocols 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.

base frame

A common frame of reference to which a series of frame changes are referenced. This is typically the ground frame.

Commercial Supersonic Technology (CST) Project

Supersonic vehicle research addresses the development of tools, technologies, and knowledge that will help eliminate today's technical barriers to practical commercial supersonic flight (taken from `https://www.nasa.gov/aeroresearch/programs/aavp/cst`).

configuration file

An ASCII file created by the user that specifies the values of various parameters in a specific namelist format. This file is passed to the data structures and functional modules in order to configure them to a specific purpose.

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.

constant frame change

A kinematic frame change that is constant with time.

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 created when requested by the user code and can be accessed by the user code using a tag provided by ANOPP2.

dynamic viscosity

In a wide range of fluids, the viscous stress is linearly related to the rate of strain; such fluids are called Newtonian. [41] The constant of proportionality relating fluid stress and rate of strain is called the viscosity; it is the factor μ in the equation

$$\tau_{ij} = 2\mu(d_{ij} - \frac{1}{3}\Delta\delta_{ij})$$

where

$$d_{ij} = \frac{1}{2}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right).$$

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 to a user such as units, metrics, options, and orientations.

Fast Scattering Code (FSC)

The Fast Scattering Code is a frequency domain scattering algorithm that calculates the total acoustic pressure at an observer by combining the incident acoustic pressure and the scattered acoustic pressure. The incident field includes the acoustic pressure at the observer and the pressure gradients on the surface of the scattering body. FSC uses equivalent sources inside the body to cancel the pressure gradients on the body. The calibrated equivalent sources are then used to predict the scattered field at the observer.

Ffowcs Williams and Hawkings (FWH)

A technique in computational aeroacoustics introduced by John Ffowcs Williams and David L. Hawkings [1]. It is an exact rearrangement of the Navier-Stokes equations into a wave equation for pressure. It is based on the acoustic analogy of James Lighthill [3, 4].

frame of reference

The coordinate system assigned to an object or location that is typically aligned with some feature of the object and is independent of all other relative systems.

frame of reference list

An ordered list of frame changes which are applied to the tracked object, sequentially, and describe the rotation and position of the object over time.

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 or observer location, 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.

jerk

Third-order time derivative.

Message Passing Interface (MPI)

A standardized and portable message-passing system to operate on a wide variety of parallel computers.

namelist

A Fortran statement, which defines a list of variables as members of a unique group name. Namelists are used in the configuration files as inputs to API routines.

narrowband spectrum (NBS)

A noise metric of pressure squared as a function of frequency with units of Pa^2. It is computed by first taking the Fourier transform of the pressure time signal and computing from the Fourier coefficients.

$$\bar{p}_m^2\left(f_m\right) = G_{pp}\left(f_m\right)\left(f_{u,m} - f_{l,m}\right)$$

where $\bar{p}_m^2$ is the NBS, f_m is the frequency corresponding to m^{th} frequency.

noise metric

Numerations of annoyance that emulate the manner in which humans respond to sound. Noise metrics, such as Effective Perceived Noise Level (EPNL), are used to assess the annoyance of sound on a listener.

observer result

A set of one or more acoustic data (noise metrics) that originate from a single component's or group of component's noise sources. Examples include: landing gear noise, slat noise, core noise, and can also include groups of noise such as engine or airframe noise. Each observer result can be accessed via a tag provided by the Observer API.

Open Multiprocessing (OpenMP)

A programming interface that supports multiplatform shared memory multiprocess programming in C, C++, and Fortran.

perceived noise level (PNL)

A scale developed to measure the perceived noisiness of jet aircraft by observers on the ground. The process of calculating PNL is clearly defined by Federal Aviation Regulations (FAR) Part 36 and International Civil Aviation Organization (ICAO) standards [42, 43].

periodic frame change

An extension to the frame change that varies in a periodic manner as a function of the rotation angle.

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.

polynomial frame change

An extension of the frame change data stucture where the motion of the following frame is defined as a polynomial function of time.

power spectral density (PSD) spectrum

A noise metric of pressure squared as a function of frequency with units of Pa^2/Hz. It is computed by first taking the Fourier transform of the pressure time signal and computing from the Fourier coefficients.

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

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

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

pressure gradient time history (PGTH)

A time history of pressure gradients.

Revolutionary Vertical Lift Technology (RVLT) Project

NASA's vision for vertical lift vehicles is to capitalize and improve unique vertical capabilities to greatly benefit the Nation's growing civil flight requirements. (taken from `https://www.nasa.gov/aeroresearch/programs/aavp/rvlt`).

routine

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

segment

A time range within which acoustic data is not expected to vary drastically and the acoustic data can be linearly interpolated without any loss of fidelity. In ANOPP2, this is also defined as the period between two source times/waypoints.

signal processing

An area of engineering and mathematics that deals with operations on, or analysis of analog as well as digitized signals, representing time-varying or spatially varying physical quantities.

snap

Fourth-order time derivative.

sound pressure level (SPL)

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

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

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

tag

An integer used by the user to communicate to internal ANOPP2 data structures. These are integers declared in the user code but defined by ANOPP2. Their actual values are arbitrary.

Transformational Tools and Technologies (TTT) Project

This project develops state-of-the-art computational and experimental tools and technologies that are vital to ARMD's ability to advance the prediction of future aircraft performance in flight, such as first-of-a-kind tools that isolate the complex turbulent airflow around vehicles and within propulsion systems. TTT creates computer-based tools, models, and associated scientific knowledge that can be applied to the entire ARMD portfolio. It also explores technologies that are broadly critical to advancing ARMD strategic outcomes, such as the understanding of new types of strong and lightweight materials, innovative controls techniques, and experimental methods (taken from `https://www.nasa.gov/aeroresearch/programs/tacp/ttt`).

Transformative Aeronautics Concepts Program (TACP)

TACP solicits and encourages revolutionary concepts, creates the environment for researchers to experiment with new ideas, performs ground and small-scale flight tests, allows failures and learns from them, and drives rapid turnover into potential future concepts to enable aviation

transformation. Research is organized to aggressively engage both the traditional aeronautics community and non-traditional partners. Although TACP focuses on sharply focused studies, the program provides flexibility for innovators to assess new-technology feasibility and provide the knowledge base for radical aeronautics advance (taken from `https://www.nasa.gov/aeroresearch/programs/tacp`).

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.

verbosity

A setting within ANOPP2, which sets the amount of information shown to the user during execution. There are three levels of verbosity: *QUIET*, *STANDARD*, and *VERBOSE*.

windowing

A technique used to "smooth" out a segment function near the end values to minimize the instantaneous jump from the end of one period to the start of the next. Many methods exist for performing the technique, including rectangular, Hanning, Hamming, flat top, and Blackman.

Index

REPORT DOCUMENTATION PAGE	Form Approved OMB No. 0704–0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
01-12-2021	Technical Memorandum	

4. TITLE AND SUBTITLE

ANOPP2's Farassat Formulations Internal Functional Modules (AFFIFMs) Reference Manual

Version 1.4

General Public Release

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Lopes, Leonard V.

Debug Build

No Parallelization or Multithreading

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

109492.02.07.07.03

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

NASA Langley Research Center
Hampton, VA 23681-2199

8. PERFORMING ORGANIZATION REPORT NUMBER

L–XXXXX

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–20210021111

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 version 1.4.0 of AFFIFMs developed by NASA Langley Research Center's Aeroacoustics Branch. The AFFIFMs provide the capability of calculating an APTH, or similar metric, provided one or more FWH surfaces. AFFIFMs also allow for compact line sources.
This API is part of a larger toolkit called the ANOPP2. The goal of ANOPP2 is to provide the ability to independently: (1) assess aircraft system noise; (2) assess aircraft component noise; and (3) evaluate aircraft noise reduction technologies and flight procedures. Additionally, ANOPP2 is designed to provide a capability for understanding the fundamental physics involved in noise generation to support experiments and flight demonstration activities.
As a component of ANOPP2, ANOPP2's Farassat's Formulations Internal Functional Modules and this document may be included as part of the ANOPP2 distribution, or they may be provided independent of that distribution.

15. SUBJECT TERMS

Acoustics, Aeroacoustics, Acoustic Analysis, Aircraft Noise

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			STI Information Desk (help@sti.nasa.gov)
U	U	U	UU	87	19b. TELEPHONE NUMBER *(Include area code)* (757) 864-9658