



## **Users Guide**

*InfraMonitor 3.1*

Stephen Arrowsmith  
([arrows@lanl.gov](mailto:arrows@lanl.gov)) and Omar  
Marcillo ([omarcillo@lanl.gov](mailto:omarcillo@lanl.gov))

## Table of Contents

Overview.....	2
Algorithms.....	2
InfraMonitor detection algorithm: The Adaptive F-detector.....	2
InfraMonitor association algorithm.....	3
InfraMonitor location algorithm: The Bayesian Infrasound Source Locator (BISL).....	3
Code.....	5
Installation.....	5
Pipeline processing using InfraMonitor 3.1.....	6
Additional Pipeline Processing Utilities.....	12
Interactive Processing using InfraMonitor 3.1.....	12
Ray tracing using InfraMonitor 3.1.....	18
Input Parameters.....	22
File Input/Output.....	22
Under the Hood.....	23
References.....	27
APPENDIX 1: Change Log.....	28

## Overview

InfraMonitor is a Matlab-based tool for infrasound monitoring that incorporates LANL-developed detection, association, and location algorithms (see descriptions below) in addition to a Matlab implementation of the Tau-P raytracing technique (Garces et al., 1998, Drob et al., 2010). The InfraMonitor package has been developed to run in two different modes: (1) As a pipeline process, and (2) interactively. The pipeline mode is designed for processing large quantities of data, without user interaction, in order to detect and locate infrasound events. The input for pipeline mode comprises data from multiple infrasound arrays, and the output comprises an event catalog (containing calculated origin times and locations). The interactive mode is designed for a more thorough user-driven analysis of individual events (or short time intervals), providing a suite of tools that aid in the analysis of infrasound data. Interactive tools include support for reading different data formats (SAC, CSS), filtering, spectra, spectrograms, array plots, display of ground-truth, arrival time picking, amplitude measurements, and ray tracing.

## Algorithms

### InfraMonitor detection algorithm: The Adaptive F-detector

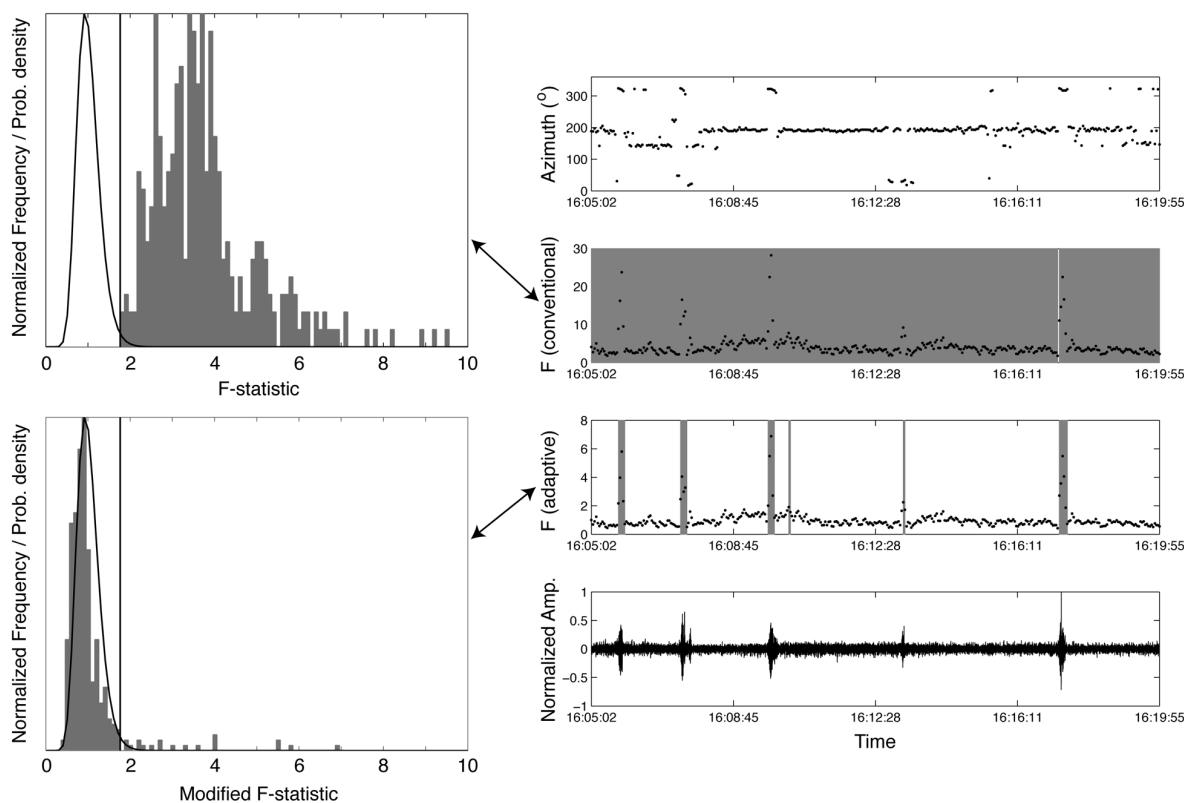
The Adaptive F-detector (AFD) technique is outlined by Arrowsmith et al. (2008, 2009). In contrast with other routinely used infrasound detectors, the AFD utilizes a contextual detection hypothesis that adaptively accounts for temporally variable correlated ambient noise (see Figure 1). This technique reduces the number of false alarms caused by background noise sources such as ocean noise, wind farms, and other continuous wave sources.

### InfraMonitor association algorithm

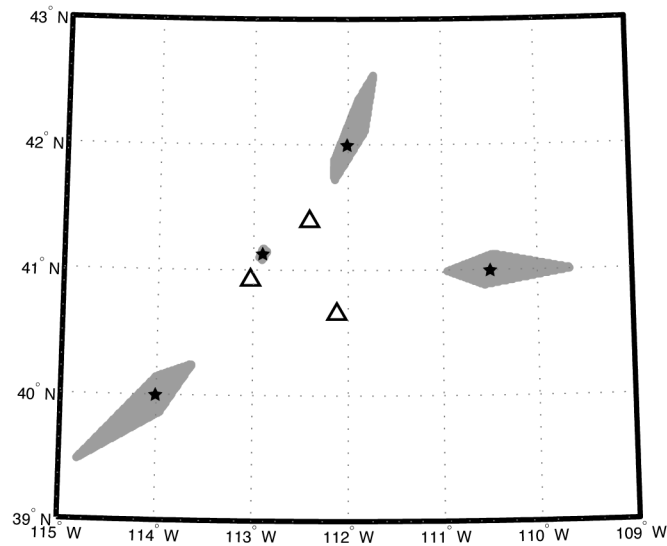
The association technique uses a grid search approach to relate arrivals at multiple arrays (Arrowsmith et al., 2008). InfraMonitor searches for groups of arrivals with backazimuths and inter-array delay times that are consistent with each grid node. When such groups of arrivals are identified, they are associated and input to the localization procedure described below (see Figure 2).

### InfraMonitor location algorithm: The Bayesian Infrasound Source Locator (BISL)

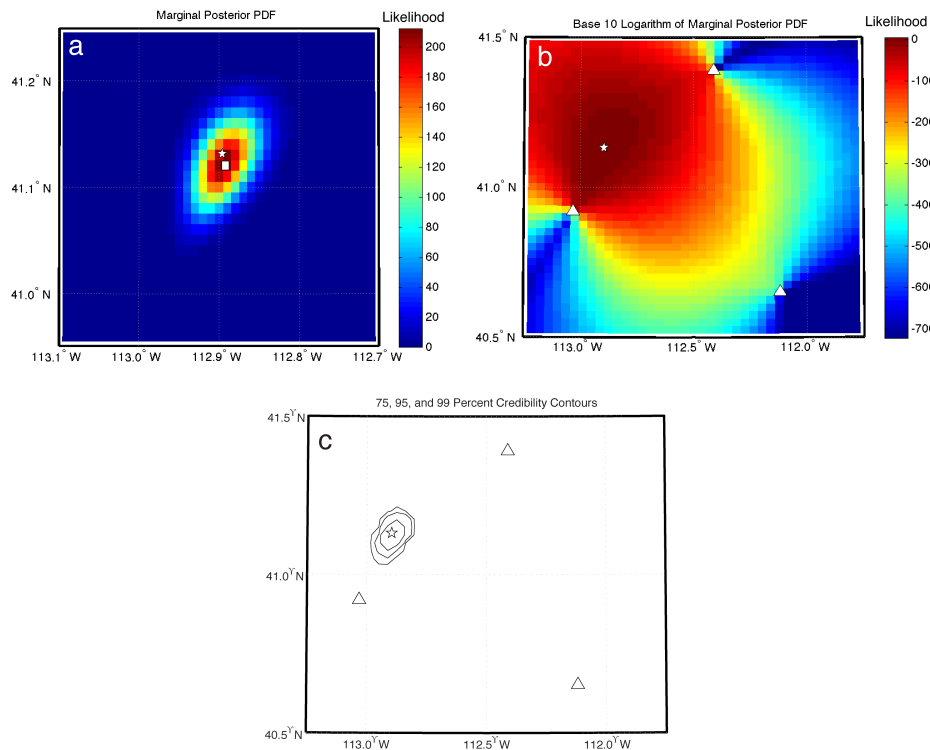
The BISL technique (Modrak et al., 2010) uses a statistical approach for infrasound location that can appropriately account for both model and measurement uncertainties inherent to infrasound. Phase identification is replaced by the use of a Bayesian prior on group velocity, with both arrival times and backazimuths contributing to the estimation of a location polygon for a specified confidence level (see Figure 3).



**Figure 1: Illustration of the AFD method in comparison with a conventional F-detector for infrasound data containing five transient signals from mine shots but contaminated by wind farm noise. The conventional detector identifies the full data segment as a detection (represented by shaded gray areas on the right panel) whereas the AFD only detects the mining explosion signals.**



**Figure 2: Illustration of the association algorithm. Gray areas represent grid nodes associated with test events (stars) at three arrays.**



**Figure 3: An illustration of the application of BISL to a rocket motor detonation conducted at the Utah Test and Training Range and recorded by three arrays. The location polygons are calculated by taking slices through the posterior PDF (shown as a color plot).**

## Code

InfraMonitor 3.1 is a Matlab GUI interface and toolbox for regional infrasound monitoring. Features include:

- Array detection (Adaptive F-detector)
- Interactive Signal Analysis
- Signal Association
- Event location
- BISL method
- Polygon location method
- Tau-P infrasound ray-tracing model
- Graphical display and Interactive GUI interface
- Command-line alternatives to GUI functionality

The full InfraMonitor 3.1 package comprises Matlab scripts, Matlab GUI templates, Python scripts, SAC data, CSS3.0 tables, and a users manual (pdf file). This distribution does not contain binary (compiled) code.

The following supporting software is needed to run this tool:

- Matlab
  - Signal Processing Toolbox
  - Statistics Toolbox
- Python

## Installation

Note that InfraMonitor 3.1 has been tested on Mac OS-X, Sun Solaris, and Linux operating systems. The current version is not supported for Microsoft Windows.

The following files are provided as part of the InfraMonitor 3.1 package: `InfraMonitor_3.1.tar.gz`, `ExampleData.tar.gz`

To get started, uncompress and untar the source folder:

```
gunzip InfraMonitor_3.1.tar.gz
tar xvf InfraMonitor_3.1
```

This will generate a folder (the root folder, the whole path to which is referred to hereafter as `<InfraROOT>`). Add `<InfraROOT>` to your Matlab path:

```
addpath(genpath(<InfraROOT>))
```

The best idea is to add this line to your Matlab startup.m file (see the Matlab documentation for more details)

Uncompress and untar the example data folder (this can be in a totally separate location from the source code):

```
gunzip ExampleData.tar.gz
```

## InfraMonitor 3.1 Users Guide

tar xvf ExampleData.tar

Now, enter Matlab and type InfraMonitor3 (or select InfraMonitor from Matlab's start menu).

### Pipeline processing using InfraMonitor 3.1

Running InfraMonitor 3.1 in pipeline mode is based on two separate functions. To test these with the ExampleData folder, run:

```
InfraMonitor_Detector('.', 'utah.wfdisc', [1 5], 2, 'css3.0');
```

This processes all the data defined in a wfdisc file for detections, outputting .arrival and .detect files. A site file with the same name as the wfdisc file must exist in the same location as the wfdisc file. Both wfdisc and site files must conform to either the CSS3.0 or NNSA formats, provided below.

Input parameters to InfraMonitor\_Detector are, in order:

- Path to the wfdisc file (Use '.' if calling function from the directory containing the wfdisc file)
- Name of wfdisc file
- Frequency band [f\_min f\_max]
- Filter order (InfraMonitor uses the Butterworth filter in the Signal Processing Toolbox)
- Schema type (Use either 'css3.0' or 'nnsa')

The precise NNSA and CSS3.0 formats are defined in the following four pages for the wfdisc and site files. InfraMonitor uses the refsta parameter to associate individual elements into arrays. It is strongly recommended that the user populates the refsta parameter (see utah.site in the example dataset).

The output files include a .detect file and .arrival file. The .detect file has the following (free-form format):

- Station name
- start date of detection
- start time of detection
- end date of detection
- end time of detection
- backazimuth
- trace velocity (km/s)
- Cross-correlation coefficient.

## InfraMonitor 3.1 Users Guide

The .arrival file conforms to the CSS3.0 arrival format, specified below (NA values are important).

#	COLUMN	STORAGE TYPE	EXT FORMAT	CHAR POS	NA VALUE	DESCRIPTION
1	<a href="#"><u>sta</u></a>	varchar2(6)	a6	1-6	Not Allowed	station identifier
2	<a href="#"><u>time</u></a>	float(53)	f17.5	8-24	Not Allowed	epoch time
3	<a href="#"><u>arid</u></a>	number(8)	i8	26-33	Not Allowed	arrival id
4	<a href="#"><u>jdate</u></a>	number(8)	i8	35-42	-1	julian date
5	<a href="#"><u>stassid</u></a>	number(8)	i8	44-51	-1	stassoc id
6	<a href="#"><u>chanid</u></a>	number(8)	i8	53-60	-1	instrument id
7	<a href="#"><u>chan</u></a>	varchar2(8)	a8	62-69	-	channel code
8	<a href="#"><u>iphase</u></a>	varchar2(8)	a8	71-78	-	reported phase
9	<a href="#"><u>stype</u></a>	varchar2(1)	a1	80-80	-	signal type
10	<a href="#"><u>deltim</u></a>	float(24)	f6.3	82-87	-1.0	delta time
11	<a href="#"><u>azimuth</u></a>	float(24)	f7.2	89-95	-1.0	observed azimuth
12	<a href="#"><u>delaz</u></a>	float(24)	f7.2	97-103	-1.0	delta azimuth
13	<a href="#"><u>slow</u></a>	float(24)	f7.2	105-111	-1.0	observed slowness (s/deg)
14	<a href="#"><u>delslo</u></a>	float(24)	f7.2	113-119	-1.0	delta slowness
15	<a href="#"><u>ema</u></a>	float(24)	f7.2	121-127	-1.0	emergence angle
16	<a href="#"><u>rect</u></a>	float(24)	f7.3	129-135	-1.0	rectilinearity
17	<a href="#"><u>amp</u></a>	float(24)	f10.1	137-146	-1.0	amplitude, instrument corrected, nm
18	<a href="#"><u>per</u></a>	float(24)	f7.2	148-154	-1.0	period
19	<a href="#"><u>logat</u></a>	float(24)	f7.2	156-162	-999.0	log (amp/per)
20	<a href="#"><u>clip</u></a>	varchar2(1)	a1	164-164	-	clipped flag
21	<a href="#"><u>fm</u></a>	varchar2(2)	a2	166-167	-	first motion
22	<a href="#"><u>snr</u></a>	float(24)	f10.2	169-178	-1.0	signal to noise ratio
23	<a href="#"><u>qual</u></a>	varchar2(1)	a1	180-180	-	signal onset quality
24	<a href="#"><u>auth</u></a>	varchar2(15)	a15	182-196	-	source/originator
25	<a href="#"><u>commid</u></a>	number(8)	i8	198-205	-1	comment id
26	<a href="#"><u>lddate</u></a>	date	a17:YY/MM/DD HH24:MI:SS	207-223	Not Allowed	load date

**NNSA, wfdisc file:**

#	COLUMN	STORAGE TYPE	EXT FORMAT	CHAR POS	NA VALUE	DESCRIPTION
1	<a href="#"><u>sta</u></a>	varchar2(6)	a6	1-6	Not Allowed	station code
2	<a href="#"><u>chan</u></a>	varchar2(8)	a8	8-15	Not Allowed	channel code
3	<a href="#"><u>time</u></a>	float(53)	f17.5	17-33	Not Allowed	epoch time
4	<a href="#"><u>wfid</u></a>	number(9)	i9	35-43	Not Allowed	waveform identifier
5	<a href="#"><u>chanid</u></a>	number(8)	i8	45-52	-1	channel identifier
6	<a href="#"><u>jdate</u></a>	number(8)	i8	54-61	-1	julian date
7	<a href="#"><u>endtime</u></a>	float(53)	f17.5	63-79	9999999999.999	ending time
8	<a href="#"><u>nsamp</u></a>	number(8)	i8	81-88	Not Allowed	number of samples
9	<a href="#"><u>samprate</u></a>	float(24)	f11.7	90-100	Not Allowed	sampling rate
10	<a href="#"><u>calib</u></a>	float(24)	f16.6	102-117	1	calibration factor
11	<a href="#"><u>calper</u></a>	float(24)	f16.6	119-134	-1	calibration period
12	<a href="#"><u>instype</u></a>	varchar2(6)	a6	136-141	-	instrument type
13	<a href="#"><u>segtype</u></a>	varchar2(1)	a1	143-143	-	segment type
14	<a href="#"><u>datatype</u></a>	varchar2(2)	a2	145-146	-	numeric data storage
15	<a href="#"><u>clip</u></a>	varchar2(1)	a1	148-148	-	clipped data flag
16	<a href="#"><u>dir</u></a>	varchar2(64)	a64	150-213	Not Allowed	directory
17	<a href="#"><u>dfile</u></a>	varchar2(32)	a32	215-246	Not Allowed	data file
18	<a href="#"><u>foff</u></a>	number(10)	i10	248-257	Not Allowed	byte offset of data segment within file
19	<a href="#"><u>commid</u></a>	number(9)	i9	259-267	-1	comment identifier
20	<a href="#"><u>lddate</u></a>	date	a17:YY/MM/DD HH24:MI:SS	269-285	Not Allowed	load date



**NNSA, site file:**

#	COLUMN	STORAGE TYPE	EXT FORMAT	CHAR POS	NA VALUE	DESCRIPTION
1	<a href="#"><u>sta</u></a>	varchar2(6)	a6	1-6	Not Allowed	station code
2	<a href="#"><u>ondate</u></a>	number(8)	i8	8-15	-1	turn on date
3	<a href="#"><u>offdate</u></a>	number(8)	i8	17-24	2286324	turn off date
4	<a href="#"><u>lat</u></a>	float(53)	f11.6	26-36	-999	geographic latitude
5	<a href="#"><u>lon</u></a>	float(53)	f11.6	38-48	-999	geographic longitude
6	<a href="#"><u>elev</u></a>	float(24)	f9.4	50-58	-999	elevation
7	<a href="#"><u>staname</u></a>	varchar2(50)	a50	60-109	-	station name/description
8	<a href="#"><u>statype</u></a>	varchar2(4)	a4	111-114	-	station type (single station, array)
9	<a href="#"><u>refsta</u></a>	varchar2(6)	a6	116-121	-	reference station for array members
10	<a href="#"><u>dnorth</u></a>	float(24)	f9.4	123-131	Not Allowed	north offset from array reference (km)
11	<a href="#"><u>deast</u></a>	float(24)	f9.4	133-141	Not Allowed	east offset from array reference (km)
12	<a href="#"><u>lddate</u></a>	date	a17:YY/MM/DD HH24:MI:SS	143-159	Not Allowed	load date

**CSS3.0, wfdisc file:**

#	COLUMN	STORAGE TYPE	EXT FORMAT	CHAR POS	NA VALUE	DESCRIPTION
1	<a href="#"><u>sta</u></a>	varchar2(6)	a6	1-6	Not Allowed	station identifier
2	<a href="#"><u>chan</u></a>	varchar2(8)	a8	8-15	Not Allowed	channel code
3	<a href="#"><u>time</u></a>	float(53)	f17.5	17-33	Not Allowed	epoch time
4	<a href="#"><u>wfid</u></a>	number(8)	i8	35-42	Not Allowed	waveform id
5	<a href="#"><u>chanid</u></a>	number(8)	i8	44-51	-1	instrument id
6	<a href="#"><u>jdate</u></a>	number(8)	i8	53-60	-1	julian date
7	<a href="#"><u>endtime</u></a>	float(53)	f17.5	62-78	+9999999999.999	end epoch time
8	<a href="#"><u>nsamp</u></a>	number(8)	i8	80-87	Not Allowed	number of samples
9	<a href="#"><u>samprate</u></a>	float(24)	f11.7	89-99	Not Allowed	sampling rate in samples/second
10	<a href="#"><u>calib</u></a>	float(24)	f16.6	101-116	Not Allowed	nominal calibration
11	<a href="#"><u>calper</u></a>	float(24)	f16.6	118-133	Not Allowed	calibration period
12	<a href="#"><u>instype</u></a>	varchar2(6)	a6	135-140	-	instrument type
13	<a href="#"><u>segtype</u></a>	varchar2(1)	a1	142-142	-	indexing method
14	<a href="#"><u>datatype</u></a>	varchar2(2)	a2	144-145	-	numeric storage
15	<a href="#"><u>clip</u></a>	varchar2(1)	a1	147-147	-	clipped flag
16	<a href="#"><u>dir</u></a>	varchar2(64)	a64	149-212	Not Allowed	directory
17	<a href="#"><u>dfile</u></a>	varchar2(32)	a32	214-245	Not Allowed	data file
18	<a href="#"><u>foff</u></a>	number(10)	i10	247-256	Not Allowed	byte offset
19	<a href="#"><u>commid</u></a>	number(8)	i8	258-265	-1	comment id
20	<a href="#"><u>lddate</u></a>	date	a17:YY/MM/DD HH24:MI:SS	267-283	Not Allowed	load date

**CSS3.0, site file:**

#	COLUMN	STORAGE TYPE	EXT FORMAT	CHAR POS	NA VALUE	DESCRIPTION
1	<a href="#"><u>sta</u></a>	varchar2(6)	a6	1-6	Not Allowed	station identifier
2	<a href="#"><u>ondate</u></a>	number(8)	i8	8-15	Not Allowed	Julian start date
3	<a href="#"><u>offdate</u></a>	number(8)	i8	17-24	-1	Julian off date
4	<a href="#"><u>lat</u></a>	float(24)	f9.4	26-34	-999.0	estimated latitude
5	<a href="#"><u>lon</u></a>	float(24)	f9.4	36-44	-999.0	estimated longitude
6	<a href="#"><u>elev</u></a>	float(24)	f9.4	46-54	-999.0	elevation
7	<a href="#"><u>staname</u></a>	varchar2(50)	a50	56-105	-	station description
8	<a href="#"><u>statype</u></a>	varchar2(4)	a4	107-110	-	station type: single station, virt. array, etc.
9	<a href="#"><u>refsta</u></a>	varchar2(6)	a6	112-117	-	reference station for array members
10	<a href="#"><u>dnorth</u></a>	float(24)	f9.4	119-127	Not Allowed	offset from array reference (km)
11	<a href="#"><u>deast</u></a>	float(24)	f9.4	129-137	Not Allowed	offset from array reference (km)
12	<a href="#"><u>lddate</u></a>	date	a17:YY/MM/DD HH24:MI:SS	139-155	Not Allowed	load date

## InfraMonitor 3.1 Users Guide

In InfraMonitor 3.1 the detector and associator/locator are made up of separate functions. This provides flexibility for running these functions using your own scripts.

To run the association/location routines for the example event, run:

```
InfraMonitor_Locator('utah',[37.0 43.0 -115.0 -109.0],[0.1 0.1],4)
```

Input parameters to InfraMonitor\_Locator are, in order:

- Database name (name of the wfdisc and site files, without any extension added)
- Grid bounds [min\_lat max\_lat min\_lon max\_lon]
- Grid spacing [min\_spac max\_spac]
- Minimum number of arrays for association.

The output files include a .origin file, a .IMassoc, file and a .origin file. The .IMassoc file and .IMpoly files have the following formats:

Table Extension	Description	How to generate
IMassoc	Associates location polygons (IMPoly table) with events (origin table) Format: %17.5f (Arrival time) %8d (EventID/PolyID) %8d (ArrayID)	Automatically generated by location routine
IMPoly	Contains information on location polygons Format: %9d (PolyID) %9d (PointID) %11.6f (lat) %11.6f (lon)	Automatically generated by location routine

## InfraMonitor 3.1 Users Guide

The .origin file uses the CSS3.0 origin format, provided below.

#	COLUMN	STORAGE TYPE	EXT FORMAT	CHAR POS	NA VALUE	DESCRIPTION
1	<a href="#"><u>lat</u></a>	float(24)	f9.4	1-9	-999.0	estimated latitude
2	<a href="#"><u>lon</u></a>	float(24)	f9.4	11-19	-999.0	estimated longitude
3	<a href="#"><u>depth</u></a>	float(24)	f9.4	21-29	-999.0	estimated depth
4	<a href="#"><u>time</u></a>	float(53)	f17.5	31-47	Not Allowed	epoch time
5	<a href="#"><u>orid</u></a>	number(8)	i8	49-56	Not Allowed	origin id
6	<a href="#"><u>evid</u></a>	number(8)	i8	58-65	-1	event id
7	<a href="#"><u>jdate</u></a>	number(8)	i8	67-74	-1	julian date
8	<a href="#"><u>nass</u></a>	number(4)	i4	76-79	-1	number of associated phases
9	<a href="#"><u>ndef</u></a>	number(4)	i4	81-84	-1	number of locating phases
10	<a href="#"><u>ndp</u></a>	number(4)	i4	86-89	-1	number of depth phases
11	<a href="#"><u>grn</u></a>	number(8)	i8	91-98	-1	geographic region number
12	<a href="#"><u>srn</u></a>	number(8)	i8	100-107	-1	seismic region number
13	<a href="#"><u>etype</u></a>	varchar2(7)	a7	109-115	-	event type
14	<a href="#"><u>depdp</u></a>	float(24)	f9.4	117-125	-999.0	estimated depth from depth phases
15	<a href="#"><u>dtype</u></a>	varchar2(1)	a1	127-127	Not Allowed	depth method used
16	<a href="#"><u>mb</u></a>	float(24)	f7.2	129-135	-999.0	body wave magnitude
17	<a href="#"><u>mbid</u></a>	number(8)	i8	137-144	-1	mb magid
18	<a href="#"><u>ms</u></a>	float(24)	f7.2	146-152	-999.0	surface wave magnitude
19	<a href="#"><u>msid</u></a>	number(8)	i8	154-161	-1	ms magid
20	<a href="#"><u>ml</u></a>	float(24)	f7.2	163-169	-999.0	local magnitude
21	<a href="#"><u>mlid</u></a>	number(8)	i8	171-178	-1	ml magid
22	<a href="#"><u>algorithm</u></a>	varchar2(15)	a15	180-194	-	location algorithm used
23	<a href="#"><u>auth</u></a>	varchar2(15)	a15	196-210	-	source/originator
24	<a href="#"><u>commid</u></a>	number(8)	i8	212-219	-1	comment id
25	<a href="#"><u>lddate</u></a>	date	a17:YY/MM/DD HH24:MI:SS	221-237	Not Allowed	load date

## **Additional Pipeline Processing Utilities**

A set of Python scripts are provided in the PYTHON subdirectory for performing various useful processing functions in analyzing pipeline results. These Python scripts are provided as examples only and will likely take some tweaking to get these working on your system. These utilities are summarized below:

<b>1. Scripts for pre-processing</b>	MakeWfdisc.py	Make a wfdisc file from a bunch of SAC files
	MakeSite.py	Make a site file from a wfdisc file and SAC files
	RobustSiteReader.py	Formats a wfdisc file with nnsa format (which InfraMonitor can read)
	RobustWfdiscReader.py	Formats a site file with nnsa format (which InfraMonitor can read)
<b>2. Scripts for post-processing</b>	CleanEvents.py	Removes events from IMassoc, IMpoly, origin tables with duplicate arrivals
	CatTables.py	Concatenates a bunch of separate IMassoc, IMpoly, origin tables
	RemoveEvents.py	Removes events from IMassoc, IMpoly, origin tables based on the area of uncertainty
	MakePolygonKML.py	Makes a Google Earth kml file for a given IMpoly table

For additional help on using these scripts, type the script name with the -h option from your UNIX terminal window.

## **Interactive Processing using InfraMonitor 3.1**

This tutorial guides you through using the InfraMonitor GUI using the example dataset included with this package. The goal of the tutorial is not to cover every detail, but to provide a broad introduction to using InfraMonitor 3.1 and to some of the key features.

Reading in and filtering data:

- Change into the ExampleData directory that you created in the installation. Open Matlab and run InfraMonitor3.
- Select File>Open>CSS3.0 from the menu bar.
- Select utah.wfdisc in the file dialog box and click the Open button.
- In the dialog box that comes up highlight all the traces by clicking and dragging, and select OK (you should note that there are 6 four-channel arrays – i.e., 24 channels – with 1 hour of data per waveform).
- Once the data are read in, they are displayed in the main InfraMonitor window as white traces on a black background. The labels on the left hand side denote the channel names. Individual arrays are not explicitly highlighted but can be identified because they typically have the same channel names (with different digits appended).
- Select Tools>Filtering>Apply on the menu bar.

## InfraMonitor 3.1 Users Guide

- Select No to the Filter only selected traces? dialog and click OK.
- Use the default filter coefficients (Filter band = 1 – 5 Hz, Order = 4) by clicking OK in the Filtering dialog box. Wait until the screen refreshes with the filtered data.

Processing data using the adaptive F-detector:

- Process all the data using the Adaptive F-detector by selecting Detect>Process Everything from the menubar. This will cycle through all the arrays that are read into InfraMonitor and add detections.
- *If you just wish to process a single array, you first need to select the array. Imagine you wish to select only BGU (Channels 1-4). You can do this in two different ways. Either press 's' on your keyboard and then click and drag over the waveforms associated with channels 1-4, or click on the Select button on the screen and choose channels 1-4 (some people find this second step easier on certain operating systems). Once channels 1-4 are highlighted, they will appear red in the main GUI window (see Figure 4). You could now process this array by selecting Detect>Process Array from the menu bar.*
- While the detector is running, a text-based progress bar will appear in the Matlab Command Window (it will appear as though nothing is happening in the InfraMonitor window!). When the detector has finished running, a pop-up box will display. Detections are denoted by gray boxes.

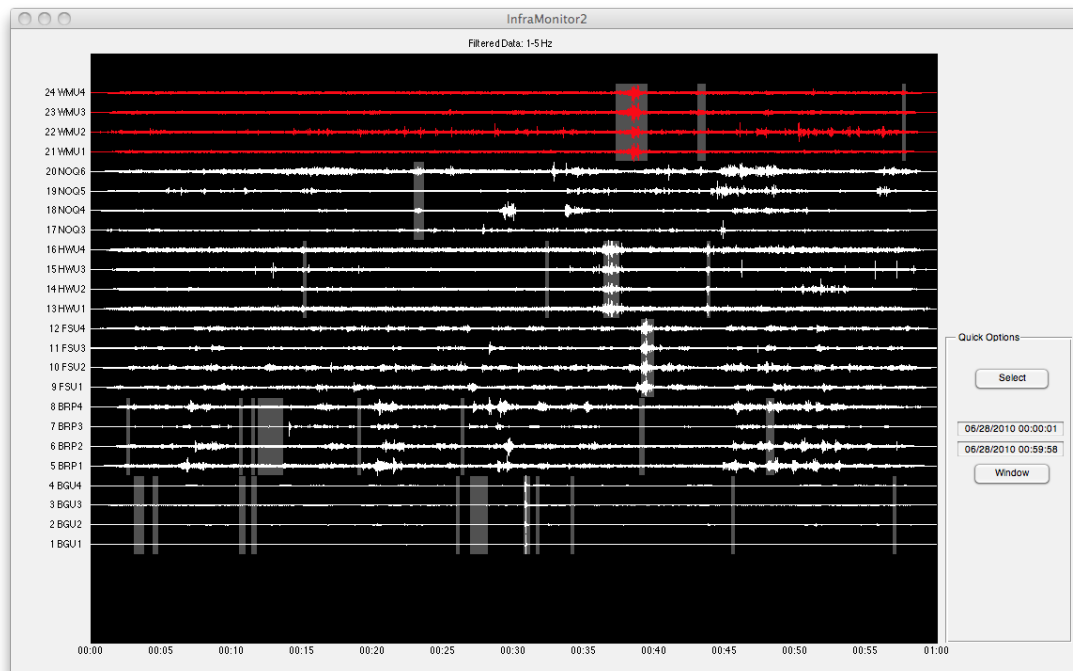


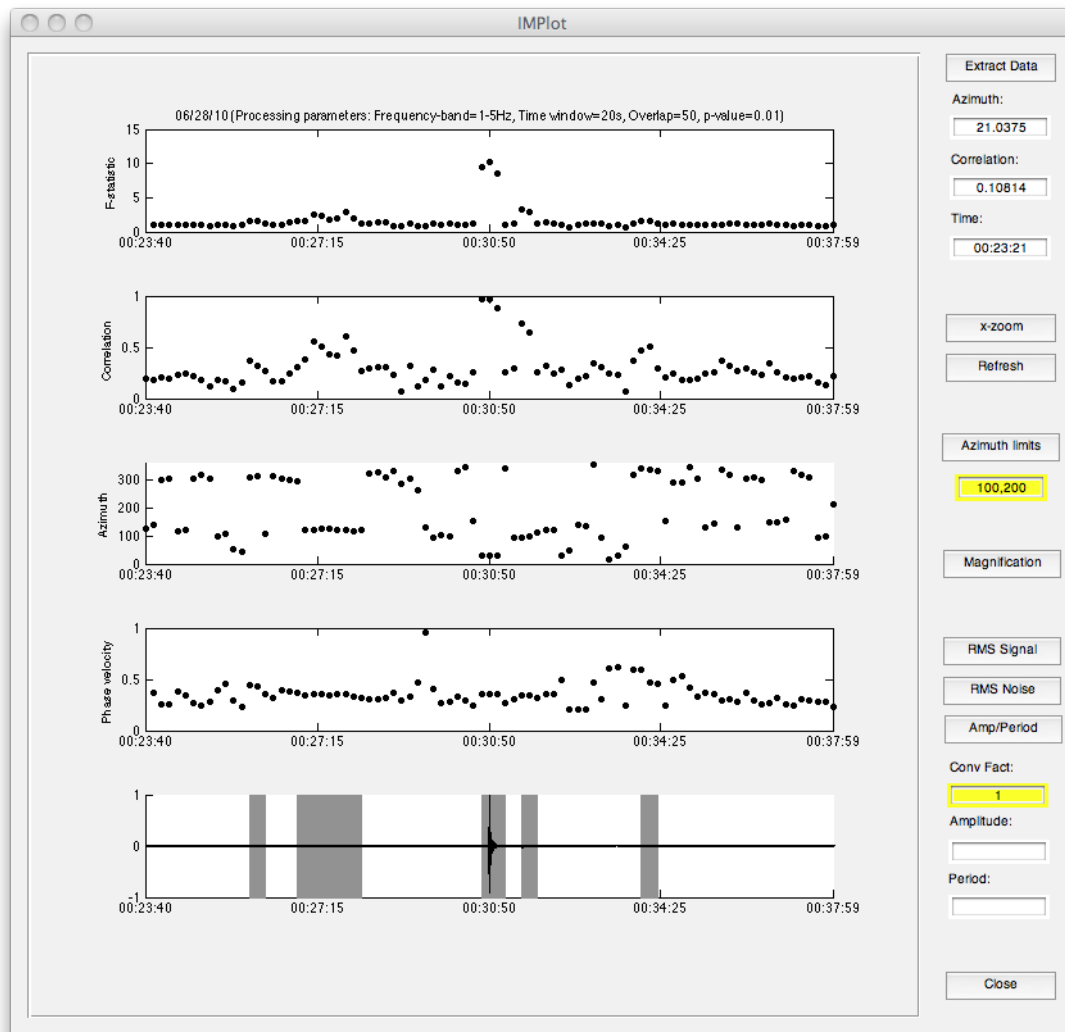
Figure 5. After running the detector on all 6 arrays.

- When the detector has completed processing all arrays your window should look like Figure 5. Select Detect>View from the menu bar and select WMU1 as the channel to plot in the popup window that opens. A plot (entitled IMPlot) containing the detection processing results will appear (detections are denoted by gray boxes).

### Analyzing the detection results:

- The detection plot window (IMPlot) has a number of different functions enabling you to analyze the detection results. Try some of these features out now. Start by clicking the Extract Data button, then click on a point on the time axis that you are interested in (e.g., the highest adaptive F-statistic point). An FK plot will appear, and the azimuth, correlation, and time will be displayed in the IMPlot display (Note: the trace velocity is also output in the Matlab command window).
- To zoom in on the IMPlot window, click the x-zoom button then click and drag over the time period you wish to zoom over. To return to displaying the full waveform, click the Refresh button (Figure 6).
- To zoom on the y-axis, click the y-zoom button then click and drag over the y limits you want to zoom on using whichever panel you want to zoom on. To zoom on a different panel, click the y-zoom button again and repeat the process for that panel.
- To magnify the waveform, click on the Magnification button and enter a magnification factor (e.g., 10). This enables you to view small signals that may be saturated by a larger signal (similar to the y-zoom feature).
- There are three features that enable you to measure amplitudes. These features assume that the data have been instrument corrected to physical units, however, you may simply input a nominal calibration factor into the yellow box titled Conv Fact.
  - To measure an RMS signal amplitude, click on a point in time centered on your signal (the RMS amplitude in a 10 s window centered on the signal will be displayed).
  - To measure the RMS noise amplitude, click on the RMS noise button and then click at the start, then the end of a noise interval.
  - To measure the peak to peak amplitude and period, zoom in on the signal of interest, click the Amp/Period button, a window will display. Select a peak, then a trough, then a peak to obtain measurements of the peak-to-peak amplitude and period.





**Figure 6. IMPlot window showing adaptive F-detector detections as gray boxes.**

Modifying the input parameters:

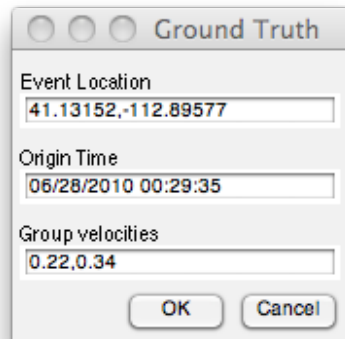
- Select Tools>Preferences from the menu bar on the main InfraMonitor window.
- The preferences window is broken down into three categories: Detector parameters, Associator parameters, BISL Locator parameters. These parameters are discussed in further detail in the Input Parameters section. For now, try changing the detection parameters and re-running the detector on a single array to get a feel for the effect of the input parameters on the results.

Some other features before we get to associating and locating events:

- With the channels for WMU highlighted (Figure 5) select Tools>Plot>Array to plot a map of the array configuration. In the figure that appears, select the data cursor tool and click on one of the points to get the array element name associated with that point.
- Now select Tools>Plot>Spectrogram with the same four channels highlighted, choose an nfft equal to 512, click on a start point, then an end point for plotting the spectrograms. Four plots

will appear – one for each waveform that is selected.

- Click on Tools>GT and enter the parameters as shown in Figure 7.

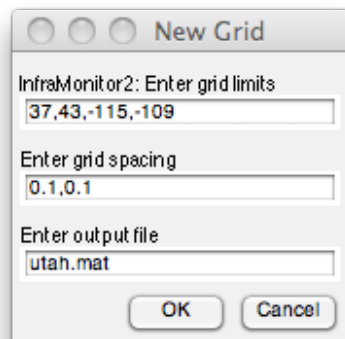


**Figure 7. Ground truth parameters for example data.**

- Lines should appear on the main window corresponding to predicted arrival times for seismic arrivals (red, magenta, and purple lines) and infrasound (green lines denote the predicted time window for this event).
- Try viewing the detection results again. The ground truth predicted backazimuth is denoted by the horizontal gray bar in the azimuth window.

Running the associator and locator:

- Select Locate>Grid>New and enter the parameters as shown in Figure 8.

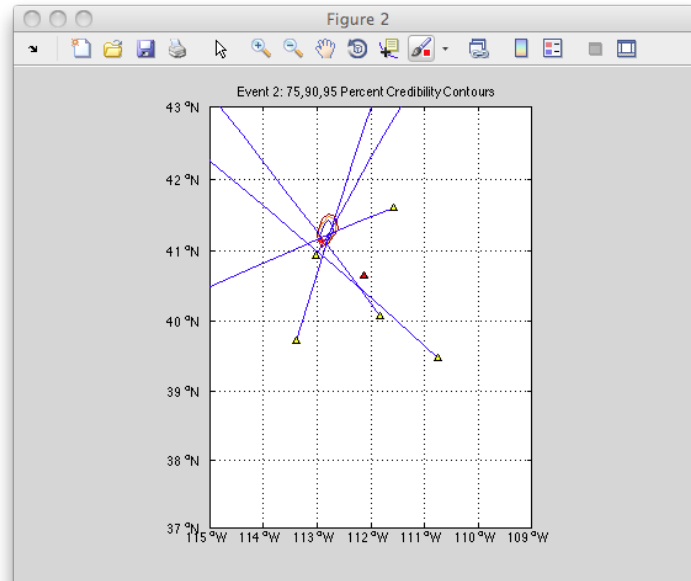


**Figure 8. Generating a grid for a rectangular region to use for associating and locating events.**

- Select Locate>Run>BISL, and use the default minimum number of arrays of 3.
- Some progress bars will appear in the Matlab command window, once the association and location processing has finished, a dialog box will appear.
- Select Locate>View>BISL and click on the Select all button followed by OK.
- Select yes in the Backazimuths dialog box and click OK.

## InfraMonitor 3.1 Users Guide

- Two map plots – one for each event found – will be displayed. The ground-truth location you entered previously is denoted by the red star. You should get a solution corresponding to the example in Figure 9.



**Figure 9. InfraMonitor localization solution for the ground-truth event in the example dataset.**

- This example illustrates an important point because one false alarm is obtained at three stations near the boundary of the box. One must be careful with associations where the network resolution is poor (i.e., all arrays are in the same direction) and with only a few stations. As the number of stations associated with the event increases, and the network resolution improves, one can have more confidence in the solution (e.g., Figure 9 would indicate a highly confident solution).

Introducing the output files:

- The InfraMonitor GUI writes out some output files as you go along (Note: to write out an arrival file you must manually select Detect>Save from the file menu). Other output files that are written out automatically include an origin file, an IMassoc file, and an IMpoly file. The arrival and origin files conform to the CSS3.0 schema guidelines, the IMassoc and IMpoly files conform to a structure unique to InfraMonitor 3.1 (see the section below entitled File Input/Output). All output files are written into the current Matlab working directory (check in your directory to make sure they are there).

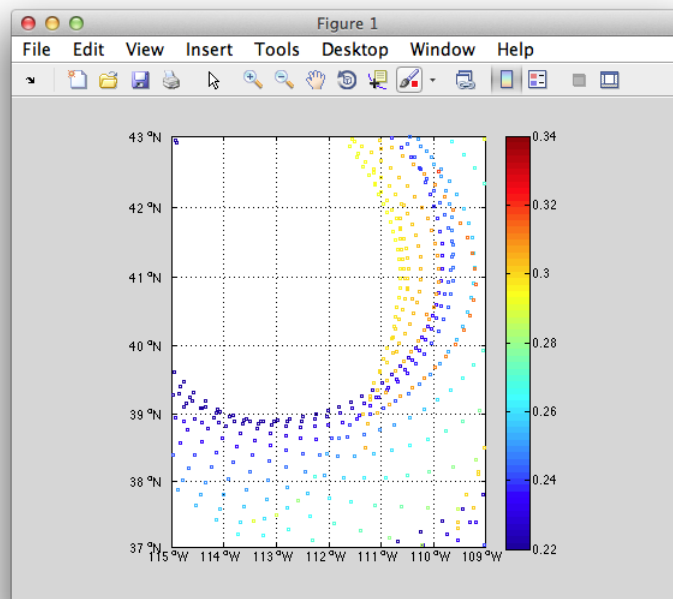
Saving the configuration:

- Select File>Save and save the configuration as Config.mat (you will use this later). Saving a configuration allows you to load your InfraMonitor session as it was without repeating all the same steps as before. To open a configuration file, select File>Open>Configuration.

## Ray tracing using InfraMonitor 3.1

InfraMonitor 3.1 incorporates the Tau-P prediction model.

- To illustrate the implementation of Tau-P, open InfraMonitor3 in Matlab and load the configuration you saved earlier (Config.mat) by selecting File>Open>Configuration<sup>1</sup>.
- Change the grid file coordinates by selecting Locate>Grid>New and inputting the following parameters:
  - Grid limits: 36,46,-120,-108
  - Grid spacing: 0.1,0.1
  - Output file: utah.mat
- Select Predict>Bounce Points from the menu bar and select the met file included in the Example data directory – exampleProfile.met.
- A met file is a simple 1D atmospheric profile with the following format: Elevation (km), Temperature (K), Zonal wind (+E) (m/s), Meridional wind (+N) (m/s)
- After the run is finished, the plot shown in Figure 10 should appear, providing predicted bounce points color coded by group velocity (or celerity) for the ground-truth event parameters that you provided earlier.



**Figure 10. Predicted bounce points for the example ground-truth event, color coded by group velocity.**

<sup>1</sup> Your configuration must include ground-truth information in addition to a grid file generated by Locate>Grid>New

## InfraMonitor 3.1 Users Guide

A much more complete set of raytracing functions can be obtained by using the InfraTaup GUI interface (run in Matlab using the command: infrataup).

For any run using InfraTaup, the following basic parameters should be set: Met-file, Resolution, Azimuth, Launch angle, Range

- Met-file: This should be set to the name of an appropriate ASCII file in the current (working) directory.
- Resolution: Ensure that the resolution corresponds to the vertical sampling in your met-file (but note that it should be entered in meters).
- Azimuth: The azimuth in degrees clockwise from North. For shooting rays at multiple azimuths, enter the minimum azimuth, increment, and maximum azimuth (e.g., 0:1:359 will shoot rays from 1° to 359° in increments of 1°).
- Launch angle: The launch angle in degrees from the horizontal. For shooting rays at multiple launch angles, enter the minimum launch angle, increment, and maximum launch angle (e.g., 1:1:89 will shoot rays from 1° to 89° in increments of 1°).
- Range: The maximum range in kilometers for which to compute the ray path.

Shooting Direction:

- Choose 'Forward from source' or 'Backward from receiver'. For the latter option, the meteorological profiles will be reversed such that shooting backward from the receiver at a particular azimuth will simulate a ray arriving at the receiver from that direction.

Methodology:

- The final solution should always be run using the 'Formal integration method' which uses the approach outlined by Drob et al. (2010). The 'Quick/Dirty method' is not recommended for anything other than a quick survey of a large simulation before running the more computationally expensive 'Formal integration method'.

Perspective:

- Set the perspective to x-z or x-y before clicking on the Compute button to define what perspective the ray solutions are plotted from. Note that the perspective can be modified by clicking on the Rotate button after the full solution has been obtained. Clicking Rotate will bring up the Matlab rotate tool, enabling the user to rotate the axes by clicking and dragging on the ray figure. Clicking Rotate a second time will deactivate the rotate tool.

Turning Rays Pop-up Menu:

- Select 'Yes' to compute and plot only rays that turn and are refracted back to the surface. Select 'No' to compute and plot all rays, including those that do not refract back to the surface, noting that this option is slower and may overcomplicate the plot in some cases.

Geographic Coordinates Pop-up Menu:

## InfraMonitor 3.1 Users Guide

- The default coordinate system is (range, transverse offset). Selecting ‘Yes’ in the Geographic Coordinates pop-up menu will instruct InfraTaup to convert all coordinates to a geographic coordinate system (longitude, latitude). InfraTaup will then use the parameters specified in the ‘Geographic Parameters’ box (Lat and Lon correspond to an event location – or receiver location for shooting rays backwards) This option is recommended when computing multiple azimuths because otherwise all solutions will be projected onto the same azimuth. Selecting this option also instructs InfraTaup to generate a map of predicted bounce points, color coded by group velocity, for the region specified by Lat limits and Lon limits.

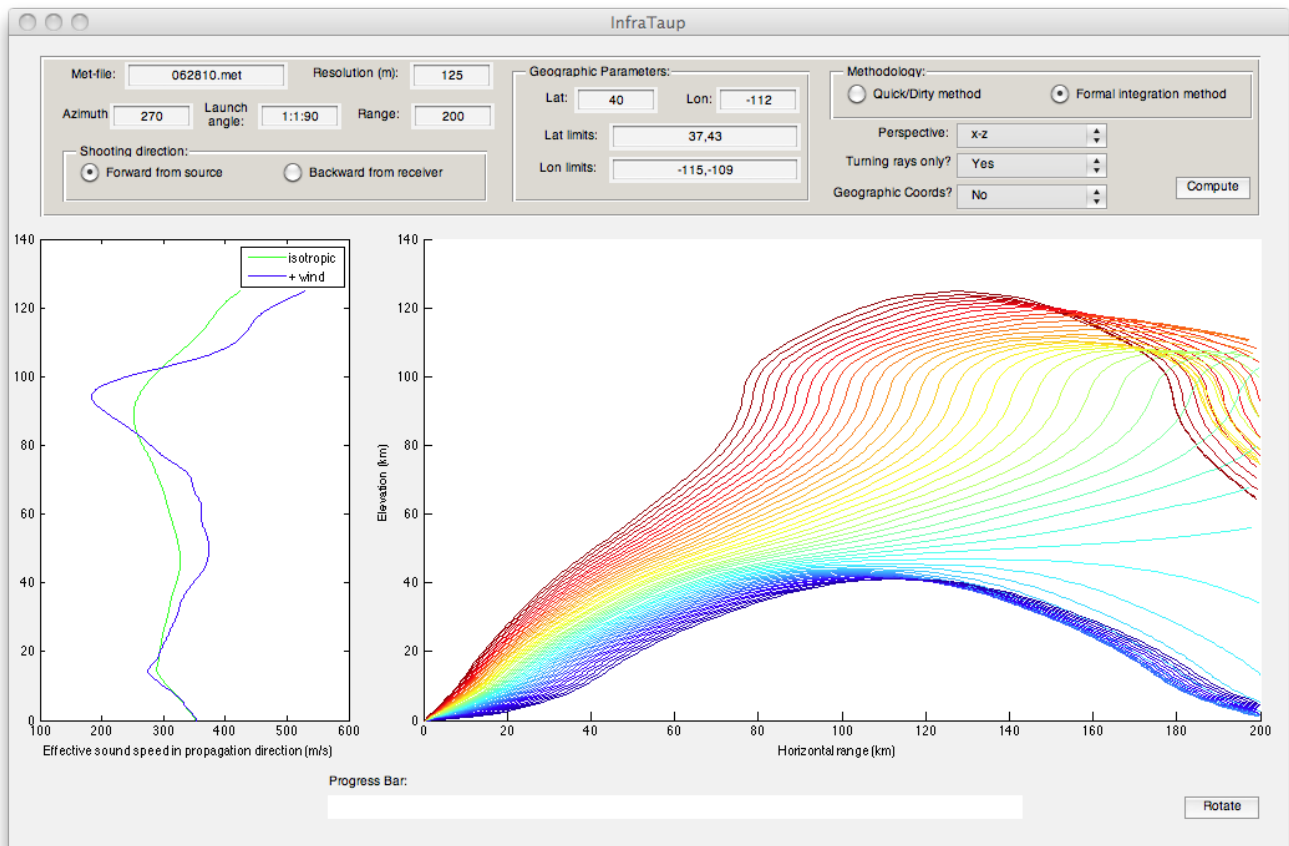
### Output Data Structure:

- InfraTaup creates a global Matlab structure array called ray. Each element in the structure array corresponds to a different ray. To view this structure array (only after performing a computation), type:

`global ray`

in the Matlab command window. This will make available to the user all the ray parameters calculated by InfraTaup. These include:

- `ray.phi`: The launch azimuth in degrees for the ray.
- `ray.theta`: The launch angle in degrees for the ray.
- `ray.x`: A vector containing either (i) the range in km at discrete points along the ray, or (ii) the longitude at discrete points along the ray (depending on whether Geographic Coordinates is selected as No (i) or Yes (ii)).
- `ray.y`: A vector containing either (i) the transverse offset in km at discrete points along the ray, or (ii) the latitude at discrete points along the ray (depending on whether Geographic Coordinates is selected as No (i) or Yes (ii)).
- `ray.z`: A vector containing the elevation in km at discrete points along the ray.
- `ray.v_g`: The group velocity of the ray.
- `ray.bp`: Bounce points for the ray (only gets computed for a geographic coordinate system).



**Figure 11. InfraTaup GUI interface with ray tracing simulations in (range, transverse offset) coordinates.**

Equivalent Command Line Functions:

- Ray computations can also be run from the Matlab command line. This option enables the user to call InfraTaup functions from their own Matlab scripts.

To shoot a ray, use the taup function:

**taup(metfile,dz,h,phi,theta,max\_range,plotflag,l,reverse,fullray,turningRays)**

Where the input parameters are:

- metfile: An ASCII file containing a vertical profile of winds and temperatures [format: elevation (km), temperature (K), zonal wind (m/s), meridional wind (m/s)]%
- dz: Vertical sampling distance for meteorological profiles (m)
- h: Ground elevation (km)
- phi: Launch azimuth (degrees)
- theta: Launch angle from horizontal (degrees)
- max\_range: Maximum straight line distance from source (km)

## InfraMonitor 3.1 Users Guide

- plotflag: 1 for plotting meteorological data, 0 otherwise
- I: Ray number (set I=1 if processing a single ray)
- reverse: Forward propagation from source (=0) or reverse propagation from receiver (=1)
- fullray: Flag to indicate whether or not to compute the full ray coordinates (0) or just the bounce-points (1)
- turningRays: Flag to indicate whether to compute only turning rays (=1) or all rays, including those that do not turn (=2)

To convert a ray to geographic coordinates:

### **BouncePoints(ray(i))**

Where i is the index of the ray that you wish to transform. To transform all rays, simply call BouncePoints in a for-loop.

## Input Parameters

All the free parameters used by InfraMonitor 3.1 can be viewed and edited by selecting: Tools>Preferences in the InfraMonitor GUI. Note that when you make a change, clicking ‘Save’ will retain the parameters for future runs (either GUI or command-line runs).

Table 1 provides a summary of the free parameters.

## File Input/Output

InfraMonitor 3.1 can read waveform data in SAC (typically .sac or .s) or CSS3.0/NNSA format (.w files). In either case, the waveform data must be linked to the following CSS3.0/NNSA table files: wfdisc & site. If wfdisc and site files are not available, they can easily be generated for .sac waveform data using the MakeWfdisc.py and MakeSite.py utilities provided in the PYTHON subdirectory.

The results generated by InfraMonitor 3.1 can be output as a series of CSS3.0 tables (in addition to specific InfraMonitor tables). A summary of the output tables is given in Table 2.

Detector parameters	Time window	The processing time window should be set to the approximate signal durations of interest.
	Overlap	The overlap between processing time windows. An overlap of 50% is recommended as an appropriate choice for most applications.
	Adaptive window	The adaptive window length must be set long enough to obtain a sufficient sample distribution, but short enough to account for temporal variations in ambient noise. We have found that an adaptive window length of 1 hour is appropriate for typical infrasound data.



	p-value	The $p$ -value affects the numbers and signal-to-noise ratios of detections obtained.
Associator parameters	Azimuthal deviation	The allowed backazimuth deviation should be set sufficiently large to account for the effects of wind on measured backazimuths.
	Min. group velocity	The slowest possible infrasonic group velocity or celerity to use in the Polygon locator
	Max. group velocity	The fastest possible infrasonic group velocity or celerity to use in the Polygon locator
	Picking Error	An estimate of the picking error (s)
BISL Locator parameters	Azimuth std. dev.	An estimate of the standard deviation in azimuth
	Arrival time std. dev.	An estimate of the standard deviation in arrival time
	Vector length	The number of origin-times and group velocities to search over
	Flat Earth	1=flat earth, 2=spherical earth (much slower)

Table 1. Summary of InfraMonitor 3.1 Input Parameters.

## Under the Hood

InfraMonitor stores all pertinent information during a session under a global Matlab variable called `InfraConfig`. `InfraConfig` is a Matlab structure array that contains all the data and derived parameters from each run. To access `InfraConfig`, type:

```
global InfraConfig
```

in the Matlab command window while you have an InfraMonitor session open. To test this, try loading the configuration you made – `Config.mat` – then type `global InfraConfig` in the Matlab command window. You should see the following:

Table Extension	Description	How to generate
arrival	Standard CSS3.0 table containing information on each arrival	After running detector on all arrays, select Detect>Save and the arrival file will be created
origin	Standard CSS3.0 table containing information on each event	Automatically generated by location routine
IMassoc	Associates location polygons (IMPoly table) with events (origin table) Format: %17.5f (Arrival time) %8d (EventID/PolyID) %8d (ArrayID)	Automatically generated by location routine
IMPoly	Contains information on location polygons Format: %9d (PolyID) %9d (PointID) %11.6f (lat) %11.6f (lon)	Automatically generated by location routine

Table 2. Output files generated by InfraMonitor 3.1

```
>> global InfraConfig
>> InfraConfig
InfraConfig =
    db: [1x1 struct]
    trace: [1x1 struct]
    array: [1x1 struct]
    detect: [1x1 struct]
    grid: [1x1 struct]
    BISL: [1x1 struct]
    gt: [1x1 struct]
```

This section describes the InfraConfig structure array to help to access data or solutions for your own codes.

InfraConfig.db: Contains database information acquired from wfdisc and site files.

- dir: Full path to the data directory.
- name: Name of CSS3.0 table.
- wfdisc: Contains wfdisc table data.
- site: Contains site table data.
- select: Contains a list of data read into memory.
- stime: The start-time of the data read into memory (in Matlab datenum format).

## InfraMonitor 3.1 Users Guide

- `etime`: The end-time of data read into memory (in Matlab datenum format).
- `dir_data`: Similar to `dir` with `./` appended.
- `f_band`: The frequency band used in the bandpass filter.

`InfraConfig.trace`: Contains the waveform data.

- `name`: Sensor names corresponding to each waveform.
- `s_f`: Sampling frequency associated with each waveform.
- `fname`: File names corresponding to each waveform.
- `data`: Waveform data associated with each waveform.
- `time`: Time vectors associated with waveform data.
- `select`: The current selected waveforms (waveforms displayed in red in the main window).
- `y_ax`: The locations on the y-axis at which each waveform is plotted.
- `gt1`: Earliest infrasound arrival times based on the ground-truth information.
- `gt2`: Latest infrasound arrival times based on the ground-truth information.
- `gt_azi`: Predicted backazimuths at each sensor based on the ground-truth information.
- `GTTime_Pn`: Predicted Pn arrival times based on the ground-truth information.
- `GTTime_Pg`: Predicted Pg arrival times based on the ground-truth information.
- `GTTime_Lg`: Predicted Lg arrival times based on the ground-truth information.

`InfraConfig.array`: Contains array and arrival information.

- `loc`: Central locations for each array.
- `arr`: Detections obtained at each array.
- `trc`: A listing of trace numbers (to link with `InfraConfig.trace` information) associated with each array.

`InfraConfig.detect`: Contains FK processing results:

- `time`: Times associated with FK results.
- `slofk`: Trace velocities.
- `az`: Azimuths.
- `fstat`: Adaptive F-statistics.
- `corr`: Correlation coefficients.
- `trc`: A listing of trace numbers (to link with `InfraConfig.trace` information) associated with each set of FK results.

`InfraConfig.grid`: Contains the grid used for association and location.

## InfraMonitor 3.1 Users Guide

- glat: Grid latitude boundaries.
- glon: Grid longitude boundaries.
- dmin: Minimum grid spacing.
- dmax: Maximum grid spacing (the default is for dmin = dmax).
- D\_STN: Distances from each array to each grid point.
- BA\_STN: Backazimuths from each array to each grid point.
- LAT: Latitudes at each grid point.
- LON: Longitudes at each grid point.

InfraConfig.BISL: Contains BISL processing results:

- lat: Locations of latitudes used in the BISL grid.
- lon: Locations of longitudes used in the BISL grid.
- len: Dimensions of BISL search [max(lat)-min(lat), max(lon)-min(lon), max(time)-min(time) (days), max(v)-min(v) (km/s)].
- PDF4: Combined likelihood value over 4 dimensions (lat, lon, velocity, origin time)
- PDF3: Combined likelihood value after integrating out velocity.
- PDF2: Combined likelihood value after integrating out velocity and origin time.
- Credibilities: Empty array used for storing temporary credibilities.
- CREDIBILITIES: Stores location credibilities for each event found.
- Baz: Stores backazimuths at each array that were used in the location.
- Meanloc: Stores the mean locations for each event found.

InfraConfig.gt: Contains ground-truth information

- loc0: Origin location.
- t0: Origin time.
- v\_g: Group velocities used for plotting ground-truth predicted arrival times.

## References

- Arrowsmith, S.J., Whitaker, R.W., Katz, C. & Hayward, C., 2009. The F-detector Revisited: An Improved Strategy for Signal Detection at Seismic and Infrasound Arrays, *Bull. Seism. Soc. Am.*, 99, 449-453.
- Arrowsmith, S.J., Whitaker, R.W., Taylor, S.R., Burlacu, R., Stump, B., Hedlin, M.A.H., Randall, G., Hayward, C. & ReVelle, D., 2008. Regional monitoring of infrasound events using multiple arrays: application to Utah and Washington State, *Geophys. J. Int.*, 175, 291-300.
- Drob, D.P., Garces, M., Hedlin, M.A.H. & Brachet, N., 2010. The Temporal Morphology of Infrasound Propagation, *Pure appl. geophys.*
- Garces, M., Hansen, R.A. & Lindquist, K.G., 1998. Traveltimes for infrasonic waves propagating in a stratified atmosphere, *Geophys. J. Int.*, 135, 255-263.
- Modrak, R.T., Arrowsmith, S.J. & Anderson, D.N., 2010. A Bayesian framework for infrasound location, *Geophys. J. Int.*, 181, 399-405.

## **APPENDIX 1: Change Log**

March 26, 2012 – Version 3.1

- New Pipeline processing code, written in Python, that simplifies pipeline processing by providing a single script that takes input as a series of arguments. The script can be pointed to a single wfdisc file that may contain multiple data segments.
- Support for reading e1 (compressed) data provided.
- Updated reading of flat files. Support for switching between css3.0 and nnsa format provided.
- Enables use of refsta information in site file.