ERDC SR-07-3

Engineer Research and Development Center

US Army Corps of Engineers®
Engineer Research and Development Center

Acoustic Propagation Through a Forest Edge

Data Report for Camp Ripley, Minnesota

Michelle E. Swearingen, Michael J. White, Patrick J. Guertin, Jeffrey A. Mifflin, Timothy E. Onder, Donald G. Albert, Stephen N. Decato, and Arnold Tunick

July 2007

ERDC SR-07-3
July 2007

Acoustic Propagation Through a Forest Edge

Data Report for Camp Ripley, Minnesota

Michelle E. Swearingen, Michael J. White, Patrick J. Guertin, Jeffery A. Mifflin, and Timothy E. Onder

Construction Engineering Research Laboratory
U.S. Army Engineer Research and Development Center
P.O. Box 9005
Champaign, IL 61826-9005

Donald G. Albert and Stephen N. Decato

Cold Regions Research and Engineering Laboratory
U.S. Army Engineer Research and Development Center
72 Lyme Road
Hanover, NH 03755-1290

Arnold Tunick

U.S. Army Research Laboratory
2800 Powder Mill Road
Adelphi, MD 20783-1197

Final report

Prepared for U.S. Army Corps of Engineers
Washington, DC 20314-1000

Under Work Unit 31D7FK

Abstract: Acoustic propagation and diffraction of high-amplitude, short duration signals through a forest edge have implications for noise mitigation and battlefield acoustic sensors. While the acoustic significance of this unique environment has been noted in the past, it has not been studied in any detail. Acoustic signals that have propagated through a forest edge yield complicated pressure-time histories for receivers both within and outside the forest. Several physical processes contribute to this complexity, including the physical structures of the biomass and ground and the microclimate. A deep understanding of acoustic propagation through this unique environment may lead to strategic placement of fire breaks for noise mitigation and improved signal processing algorithms for use with acoustic detection, and direction-finding and range-finding sensors. Because of the broad scope of issues that could be addressed once acoustic propagation and diffraction at a forest edge is understood, it is important to study this unique environment in detail. This report provides documentation of a field experiment conducted as part of a study of the acoustic properties of the forest edge environment.

Contents

Figures and Tables

Figures

Preface

This study was conducted for Headquarters, U.S. Army Corps of Engineers under the ERDC Basic Research program. The technical monitor was Martin J. Savoie, CERL Technical Director for the Installations business area.

The Ecological Processes Branch (CN-N) of the Installation Division (CN), Construction Engineering Research Laboratory (CERL) managed and executed the work. The CERL principal investigator was Dr. Michelle Swearingen. Field personnel during this experiment included Dr. Swearingen, Dr. Michael White, Patrick Guertin, Jeffery Mifflin, and Timothy Onder from CERL and Dr. Donald G. Albert and Stephen Decato from the Signature Physics Branch, Cold Regions Research and Engineering Laboratory (CRREL). Thanks go to CPT Keith Ferdon, Bill Brown, Tim Notch, CW2 Derek Lindberg, SSG Justin Piepenburg, and all of the personnel at Camp Ripley who provided test support and logistical assistance.

Alan Anderson is Chief, CN-N, and Dr. John T. Bandy is Chief, CN. The Deputy Director of CERL is Dr. Kirankumar V. Topudurti, and the Director is Dr. Ilker R. Adiguzel. Dr. Thomas S. Anderson is Acting Chief of the Signature Physics Branch at CRREL. The Deputy Director of CRREL is Dr. Lance D. Hansen, and the Director is Dr. Robert E. Davis.

CERL and CRREL are elements of the U.S. Army Engineer Research and Development Center (ERDC), U.S. Army Corps of Engineers. The Commander and Executive Director of ERDC is COL Richard B. Jenkins, and the Director of ERDC is Dr. James R. Houston.

Unit Conversion Factors

Multiply	By	To Obtain
acres	4,046.873	square meters
inches	0.0254	meters
pounds (mass)	0.45359237	kilograms
square feet	0.09290304	square meters

1 Introduction

The experiment described in this data report took place on Range 79 at Camp Ripley, MN during 19–27 June 2006. Data recordings were obtained on 22–25 June 2006. The location was chosen because of its many favorable attributes. It is a uniform pine stand with a distinct, straight edge. The ground is flat with a sufficient open field adjacent to the stand. The site was easily accessible and of sufficient size.

Objectives

This experiment had several goals, as stated below:

1. Obtain acoustic measurements in the open field, within the forest edge area, and within deep forest. Measure at more than one source and receiver height combination. Use these measurements to track the acoustic propagation through these three distinct media.
2. Determine the changes in ground impedance characteristics and the effect on acoustic propagation as sound travels through the forest edge.
3. Obtain measurements of the acoustic backscatter from the forest edge into the open field.
4. Investigate the changes in meteorological profiles as one moves from the open field, through the forest edge, and into the forest. Take sufficient measurements to verify the theory and computer model calculations of Arnold Tunick (Army Research Laboratory meteorologist).
5. Determine the importance of the biomass structure at the forest edge on acoustic propagation.
6. Determine a characteristic size for the effective transition zone (i.e., the distance needed to transition from open field to forest conditions and vice versa).

Approach

Four acoustic sources were used during the course of this experiment:

1. 1.25-lb bricks of Composition C-4,
2. a propane cannon,
3. a loudspeaker using a white noise signal, and
4. a .45 cal pistol with blanks.

These sources were chosen to represent a broad variety of source types (impulsive and continuous, linear and nonlinear) and the full range of frequencies of interest. Detonation of Composition C-4 supplied a nonlinear, low-frequency-rich source. The propane cannon provided a more linear impulsive source, with ample low-frequency energy. The loudspeaker provided a continuous-type source for a broad range of frequencies. The propane cannon and loudspeaker were portable, allowing for the use of four separate source locations with minimal fuss. The pistol was used to characterize ground impedance. Limited data are available for this sound source.

Scope

This report is intended to provide a full description of the data set from the experiment described within it.* A quality evaluation of the data is included in this report, as well as a description of data reduction. Analysis of the data is reserved for a later report.

* Data sets are available on DVD upon request. Send requests to Dr. Michelle Swearingen, PO Box 9005, Champaign, IL 61826-9005 or email her at Michelle.E.Swearingen@erdc.usace.army.mil.

2 Test Overview

The impact of a forest edge on acoustic propagation is currently unknown. It is hypothesized that the forest edge contributes significant attenuation when in a propagation path. This attenuation will be caused by changes in the microclimate, ground, and biomass compared to an open area. This basic research project is designed to isolate and study each factor that could influence the acoustic propagation.

The experimental measurements were conducted on Range 79 at Camp Ripley, MN. Figure 1 contains an aerial photograph of the site with sensor locations marked. Figure 2 is a schematic of the sensor layout.

Acoustical data recording and reduction

Acoustical data were recorded on two model DL750 16-channel 16-bit Yokogawa digital recording oscilloscopes, using type 701251 voltage input cards. The four four-element microphone arrays were recorded on Trailer scope (located in the trailer), and the other microphones (S1-5, L1-3) were recorded on the Forest scope (located in the forest).

Composition C-4 records were 5 seconds long. The Trailer scope was triggered by microphone A2a; the Forest scope was triggered by microphone S2. In each case, the trigger signal contains a 1 second pre-trigger and 4 seconds post-trigger. The sampling rate was set to 100k samples/second for each channel. Other microphone channels are recorded simultaneously and show the blast at increasingly later times with increasing distance from the blast point.

Propane cannon records were in the same format and sampling rate as the Composition C-4 records. However, the trigger microphone varied according to blast point. Table 1 indicates the blast point and trigger microphone for each oscilloscope.

Loudspeaker records were manually started and stopped at each oscilloscope in order to accommodate the lower received levels and to avoid recording only wind noise. Records were 100 seconds in length with a sampling rate of 50,000 samples/second.

Pistol shot data were only recorded on four 1/2-in. microphones, placed on the ground at locations A1–A4. These data were recorded with a record length of 2 seconds and a sampling rate of 5,000 samples/second. Triggering was accomplished manually.

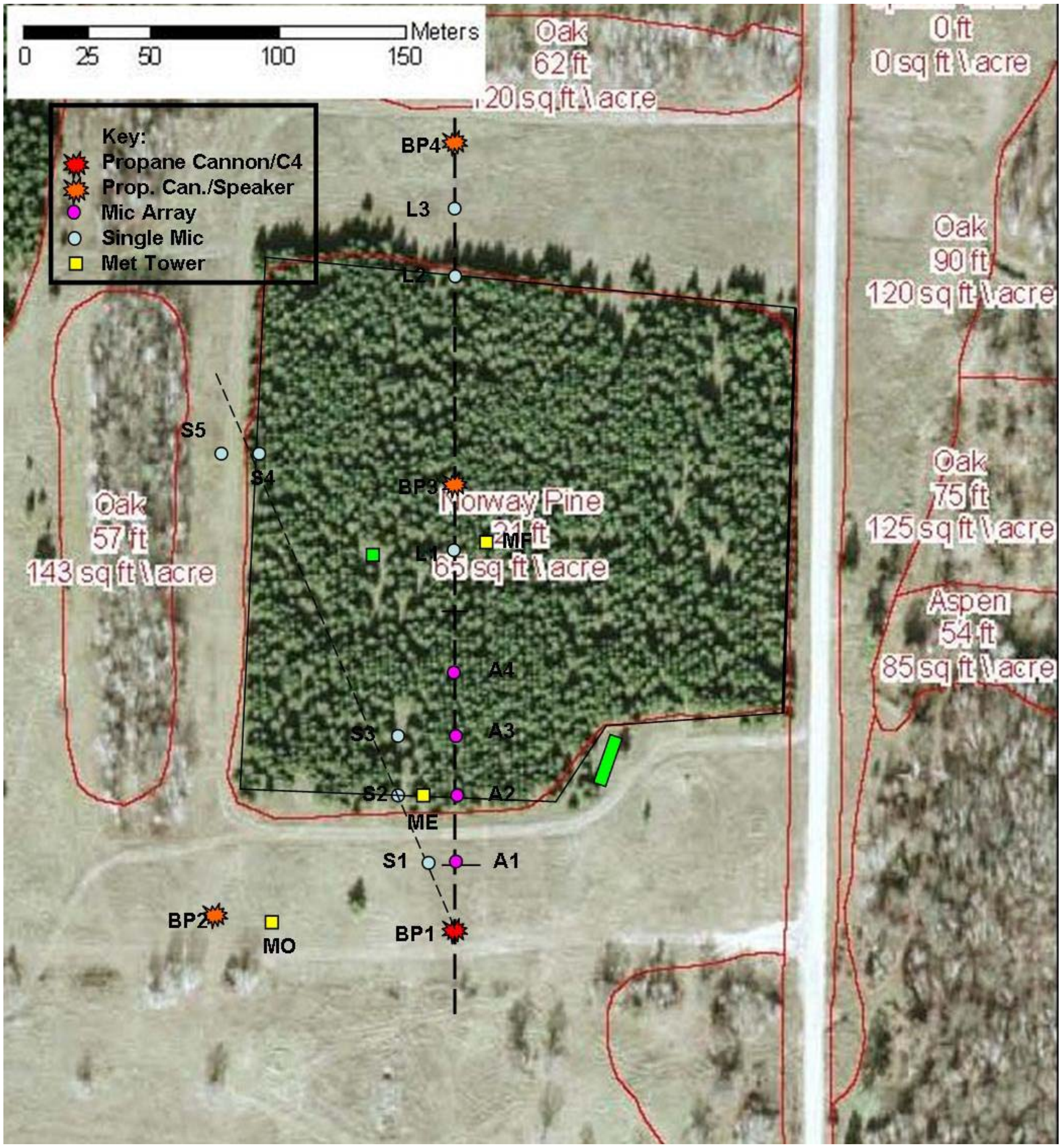

Figure 1. Aerial view of test site with sources, sensors, and meteorological instrumentation locations overlaid.

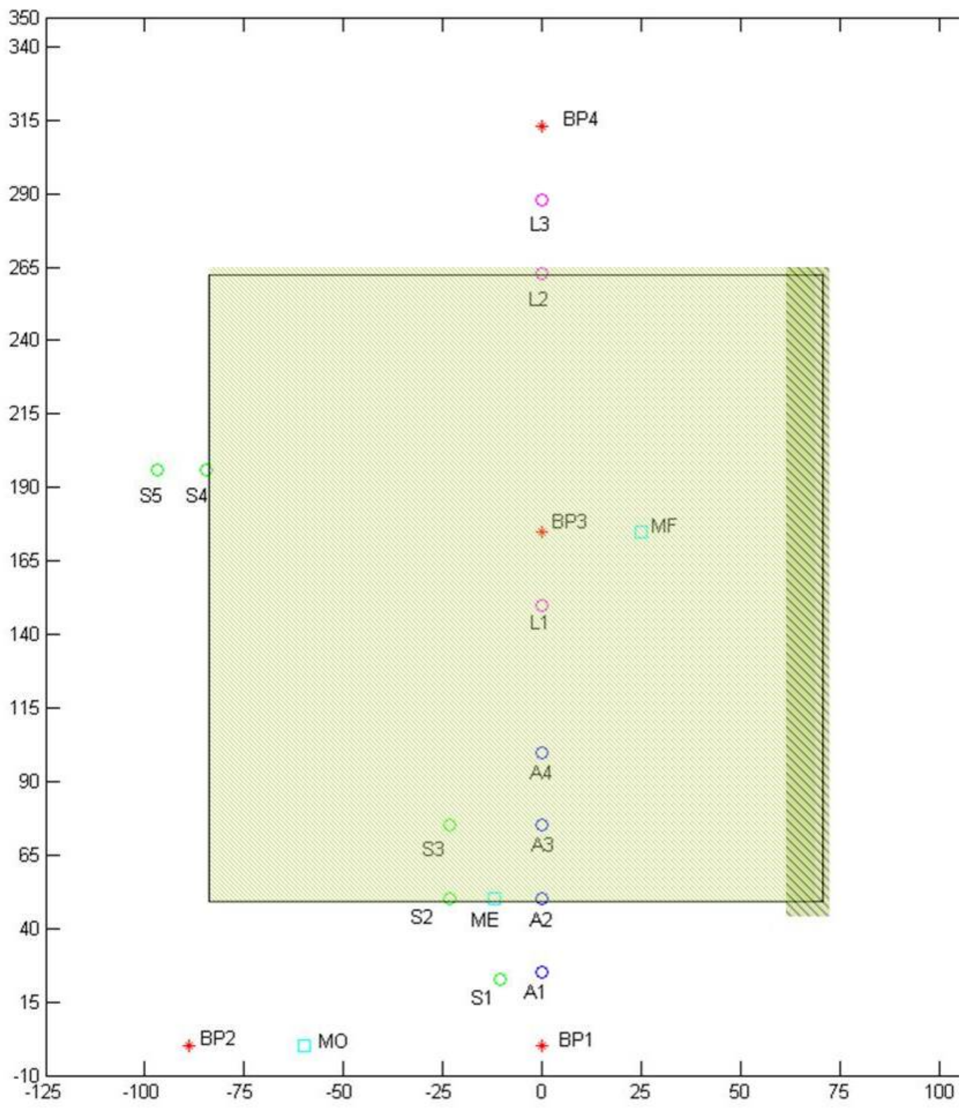

Figure 2. Schematic of sensor layout. Green hashmarked box represents a portion of the forest. Forest continues in the +x direction.

Table 1. Trigger microphone used for each propane cannon blast point.

Blast Point	Trailer	Forest
BP1	A1a	S1
BP2	A1a	S1
BP3	A4a	L1
BP4	A4a	L3

Acoustical data in all cases were reduced using Matlab scripts (attached in the appendix). As the data were all recorded digitally as integer-time series, the only transformation necessary was to convert the files from the binary oscilloscope format into floating-point Matlab format as voltage-time series. Waveforms remain in volts but are converted to Pascals by applying pressure voltage sensitivities within plotting scripts.

List of signature records

The inserted Excel spreadsheet contains the following information for all recordings: Yokogawa scope file number, Date, Time (1 hour fast), Source Type, Source Location, Array Configuration, Matlab file name/location, and Comments. The comments contain information on data quality. Two worksheets are in the file, one for the Trailer Scope and one for the Forest Scope. Table 2 lists the microphone locations and appropriate scope and channel.

Table 2. Microphone location, recording oscilloscope, and channel matrix.

Microphone	Scope Loc.	Channel
A1a	Trailer	CH1
A1b	Trailer	CH2
A1c	Trailer	CH3
A1d	Trailer	CH4
A2a	Trailer	CH5
A2b	Trailer	CH6
A2c	Trailer	CH7
A2d	Trailer	CH8
A3a	Trailer	CH9
A3b	Trailer	CH10
A3c	Trailer	CH11
A3d	Trailer	CH12
A4a	Trailer	CH13
A4b	Trailer	CH14
A4c	Trailer	CH15
A4d	Trailer	CH16
L1	Forest	CH1
L2	Forest	CH2
L3	Forest	CH3
S1	Forest	CH4
S2	Forest	CH5
S3	Forest	CH6
S4	Forest	CH7
S5	Forest	CH8

The embedded spreadsheet (also included on the DVD) contains a list of all recordings. Quality control on the data has been completed in the sense that all missing and completely unusable signals are marked. Noisy signals are not necessarily marked. The following abbreviations are used:

Pre-Cal: Pre-calibration signal

Post-Cal: Post-calibration signal

C-4: Composition C-4 explosive

PC: Propane Cannon

N:\Research\
Acoustics\Camp Riple

Specifications on sources

C-4	1 Brick	1.25 lb (0.57 kg)
0.45 Pistol Blank	Overall Length	0.89 in. (22.6 mm)
	Case Length	0.89 in. (22.6 mm)
	Cartridge Weight	7.6 g
	Case Weight	5.8 g
	Powder Weight	31 gr
	Powder Type	Black, FFFFG, CCI
	Primer Type	Magnum
	Manufacturer	Custom made
Propane Cannon	Manufacturer	Reed-Joseph International Co.
	Model	SCARE AWAY M-4 Cannon
Loudspeaker	Manufacturer	Cerwin-Vega!
	Model	PSX-122
	Amplifier	Crown Macro-Tech 600

Configuration details

Tables 3 through 11 contain sensor and source locations, as well as sensor details and changes/issues.

Table 3. Locations of microphones and blast points. Distances are to BP1.

Location Name	Coordinates	Distance to BP1 (m)
BP1	(0,0)	0
BP2	(-88.85,0)	88.85
BP3	(0,175)	175
BP4	(0,313)	313
A1(a-d)	(0,25)	25
A2(a-d)	(0,50)	50
A3(a-d)	(0,75)	75
A4(a-d)	(0,100)	100
S1	(-10.6,22.7)	25
S2	(-23.2,50)	55.1
S3	(-23.2,75)	78.5
S4	(-84.5,196)	213.4
S5	(-97,196)	218.7
L1	(0,150)	150
L2	(0,263)	263
L3	(0,288)	288

Table 4. Initial set of meteorological sensors on the open field tower.

Location	Height (m)	Height (ft)	Sensor Type	Serial #
Open	Ground	Ground	Soil Moisture	1001458
Open	N/A	N/A	Barometric Pressure	718072
Open	~1	3.28	Net Radiometer	730735
Open	2.59	8.5	Anemometer	716311
Open	2.59	8.5	8-bit temp/humidity	725373
Open	2.59	8.5	12-bit temperature	732365
Open	5.18	17	Anemometer	981368
Open	5.18	17	8-bit temp/humidity	725365
Open	5.18	17	12-bit temperature	732762
Open	7.77	25.5	Anemometer	991685
Open	7.77	25.5	8-bit temp/humidity	725370
Open	7.77	25.5	12-bit temperature	1000055
Open	10.36	34	Anemometer	713662
Open	10.36	34	8-bit temp/humidity	725372
Open	10.36	34	12-bit temperature	732773
Open	12.95	42.5	Anemometer	991684
Open	12.95	42.5	8-bit temp/humidity	725363

Location	Height (m)	Height (ft)	Sensor Type	Serial #
Open	12.95	42.5	12-bit temperature	732766
Open	~1	3.28	Rain Gauge	668179
Open	N/A	N/A	Data Logger –temperature	724611
Open	N/A	N/A	Data Logger – wind	724610
Open	N/A	N/A	Data Logger – other	724615

Table 5. Initial set of meteorological sensors on the forest edge tower.

Location	Height (m)	Height (ft)	Sensor Type	Serial #
Edge	Ground	Ground	Soil Moisture	1001459
Edge	N/A	N/A	Barometric Pressure	718077
Edge	~1	3.28	Net Radiometer	730732
Edge	2.59	8.5	Anemometer	981384
Edge	2.59	8.5	8-bit temp/humidity	725371
Edge	2.59	8.5	12-bit temperature	732761
Edge	5.18	17	Anemometer	981383
Edge	5.18	17	8-bit temp/humidity	725356
Edge	5.18	17	12-bit temperature	732759
Edge	7.77	25.5	Anemometer	713665
Edge	7.77	25.5	8-bit temp/humidity	725360
Edge	7.77	25.5	12-bit temperature	732760
Edge	10.36	34	Anemometer	981385
Edge	10.36	34	8-bit temp/humidity	999197
Edge	10.36	34	12-bit temperature	732764
Edge	12.95	42.5	Anemometer	713660
Edge	12.95	42.5	8-bit temp/humidity	725367
Edge	12.95	42.5	12-bit temperature	732768
Edge	~1	3.28	Rain Gauge	668173
Edge	N/A	N/A	Data Logger –temperature	535230
Edge	N/A	N/A	Data Logger – wind	550714
Edge	N/A	N/A	Data Logger – other	595411

Table 6. Initial set of meteorological sensors on the forest tower.

Location	Height (m)	Height (ft)	Sensor Type	Serial #
Forest	Ground	Ground	Soil Moisture	986131
Forest	N/A	N/A	Barometric Pressure	718071
Forest	~1	3.28	Net Radiometer	730731
Forest	2.59	8.5	Anemometer	713657

Location	Height (m)	Height (ft)	Sensor Type	Serial #
Forest	2.59	8.5	8-bit temp/humidity	725358
Forest	2.59	8.5	12-bit temperature	732763
Forest	5.18	17	Anemometer	713658
Forest	5.18	17	8-bit temp/humidity	725357
Forest	5.18	17	12-bit temperature	732767
Forest	7.77	25.5	Anemometer	965682
Forest	7.77	25.5	8-bit temp/humidity	725359
Forest	7.77	25.5	12-bit temperature	732770
Forest	10.36	34	Anemometer	713666
Forest	10.36	34	8-bit temp/humidity	725366
Forest	10.36	34	12-bit temperature	732772
Forest	12.95	42.5	Anemometer	716310
Forest	12.95	42.5	8-bit temp/humidity	725364
Forest	12.95	42.5	12-bit temperature	732769
Forest	~1	3.28	Rain Gauge	668171
Forest	N/A	N/A	Data Logger –temperature	724627
Forest	N/A	N/A	Data Logger – wind	724625
Forest	N/A	N/A	Data Logger – other	724608

Table 7. Issues with meteorological sensors during the experiment.

Location	Height (m)	Sensor Type	Serial #	Date Failed	Time Failed	Replaced?
Open	N/A	Data Logger	724611	unknown	unknown	724619 on 06/24/06 15:55
Edge	2.59	8-bit temp/humidity	725371	N/A	N/A	No. Manufacturer evaluation indicates failure
Forest	12.95	12-bit temperature	732769	24 June	09:20	No. Manufacturer evaluation indicates no defect
Open	12.95	Anemometer	991684	24 June	05:10	No. Manufacturer evaluation indicates no defect.

Table 8. Composition C-4 microphone list. Distances are from BP1.

Location	Distance (m)	Mic. Type	Microphone #	Power Supply #	Changed?
A1a	25	Blast Pencil	5872	8014	N
A1b	25	Blast Pencil	5870	8012	N
A1c	25	Blast Pencil	5875	8197	N
A1d	25	Blast Pencil	5869	8013	N
A2a	50	1/8 in.	2515896	1466052	N
A2b	50	1/8 in.	45967	1466052	N
A2c	50	1/8 in.	56385	1466063	N
A2d	50	1/8 in.	56396	1466063	N
A3a	75	1/8 in.	2515907	1533630	N
A3b	75	1/8 in.	2515897	1533630	N
A3c	75	1/8 in.	2515914	1466060	N
A3d	75	1/8 in.	2515901	1466060	N
A4a	100	1/4 in.	991054	1533663	N
A4b	100	1/4 in.	38681	1533663	N
A4c	100	1/4 in.	41576	2424025	N
A4d	100	1/4 in.	991206	2424025	N
S1	25	Blast Pencil	5871	8015	N
S2	55.1	1/8 in.	56378	555809	N
S3	78.5	1/8 in.	2515888	1466091	N
S4	213.4	1/4 in.	38671	1533661	N
S5	218.7	1/4 in.	35968	1533632	N
L1	150	1/4 in.	41574	1466051	N
L2	263	1/4 in.	35976	936299	N
L3	288	1/4 in.	35979	1466048	N

Table 9. Propane cannon microphone list. Distances are from BP1.

Location	Distance (m)	Mic. Type (in.)	Microphone #	Power Supply #	Changed?
A1a	25	1/4	41584	761782	N
A1b	25	1/4	41572	761782	N
A1c	25	1/4	41571	1533633	N
A1d	25	1/4	35975	1533633	N
A2a	50	1/4	41577	1466052	N
A2b	50	1/4	318293	1466052	N

Location	Distance (m)	Mic. Type (in.)	Microphone #	Power Supply #	Changed?
A2c	50	1/4	48592	1466063	N
A2d	50	1/4	41579	1466063	N
A3a	75	1/4	41520	1533630	N
A3b	75	1/4	38674	1533630	N
A3c	75	1/4	35966	1466060	N
A3d	75	1/4	52399	1466060	Y
A4a	100	1/4	991054	1533663	N
A4b	100	1/4	38681	1533663	Y
A4c	100	1/4	41576	2424025	N
A4d	100	1/4	991206	2424025	N
S1	25	1/4	52398	1466055	N
S2	55.1	1/4	41578	555809	N
S3	78.5	1/4	41573	1466091	N
S4	213.4	1/4	38671	1533661	N
S5	218.7	1/4	35968	1533632	N
L1	150	1/4	41574	1466051	N
L2	263	1/4	35976	936299	N
L3	288	1/4	35979	1466048	N

Table 10. Loudspeaker microphone list. Distances are from BP1.

Location	Distance (m)	Mic. Type (in.)	Microphone #	Power Supply #	Changed?
A1a	25	1/4	41584	761782	N
A1b	25	1/4	41572	761782	N
A1c	25	1/4	41571	1533633	N
A1d	25	1/4	35975	1533633	N
A2a	50	1/2	1857571	1466052	N
A2b	50	1/2	1357654	1466052	N
A2c	50	1/2	1881050	1466063	N
A2d	50	1/2	2335415	1466063	N
A3a	75	1/2	1881043	1533630	N
A3b	75	1/2	1881046	1533630	N
A3c	75	1/2	1881049	1466060	N
A3d	75	1/2	2068999	1466060	N
A4a	100	1/2	1857589	1533663	N

Location	Distance (m)	Mic. Type (in.)	Microphone #	Power Supply #	Changed?
A4b	100	1/2	1848766	1533663	N
A4c	100	1/2	1848782	2424025	N
A4d	100	1/2	1783511	2424025	N
S1	25	1/2	52398	1466055	N
S2	55.1	1/2	1838395	555809	N
S3	78.5	1/4	41573	1466091	N
S4	213.4	1/4	38671	1533661	N
S5	218.7	1/4	35968	1533632	N
L1	150	1/4	41574	1466051	N
L2	263	1/4	35976	936299	N
L3	288	1/4	35979	1466048	N

Table 11. Issues with microphones during propane cannon test.

Location	Mic Type (in.)	Serial #	Date Failed	Time Failed	Replaced?
A3d	1/4	52399	24 June	13:16	2239253 on 6/24/06 14:00
A4b	1/4	38681	24 June	13:16	2239254 on 6/24/06 14:00

Microphone array

The microphones set up at locations A1–A4 all included a four-microphone array set in a minimum baseline redundancy array configuration. A schematic of this configuration is shown in Figure 3. Figure 4 shows a microphone array in a parallel configuration.

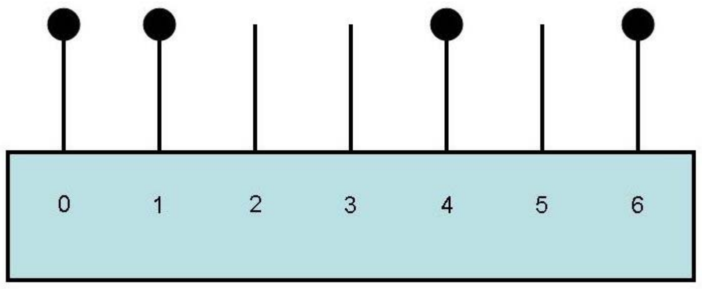

Figure 3. Minimum baseline-redundancy array. This array provides six baselines per four microphones. Locations 2, 3, and 5 are not used.

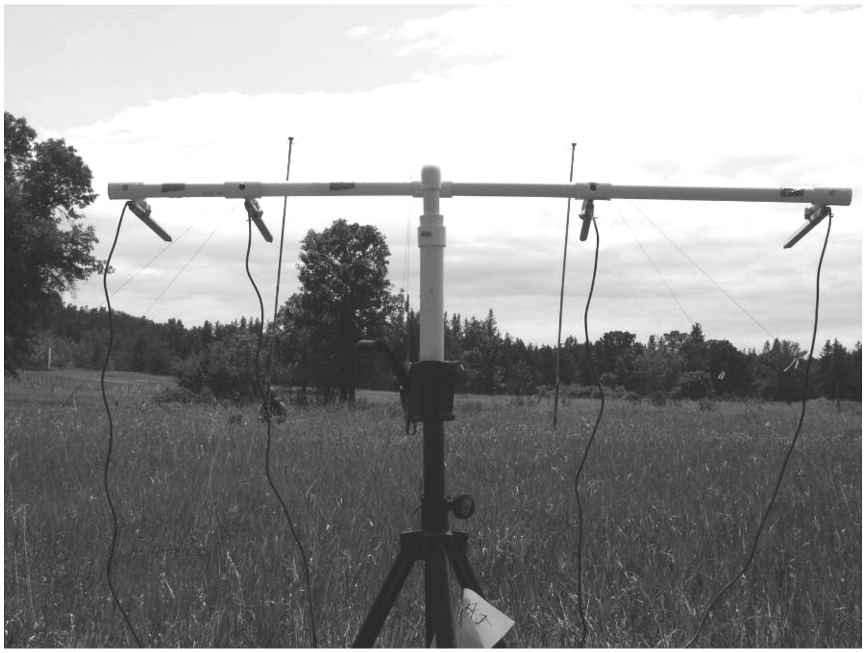
Figure 4. A microphone array is shown in the parallel configuration.

In this configuration, the following distances were used: (a) 66.5 cm to the left of the center point, (b) 41 cm to the left of the center point, (c) 34.5 cm to the right of the center point, and (d) 84 cm to the right of the center point. In the perpendicular configuration, (a) was to the left of the propagation line. In the parallel configuration, (a) was closest to the source point. In the vertical configuration, (a) was the highest microphone. The height was set so that in the perpendicular and parallel configurations, all microphones were 1.4 m above the ground surface.

Microphone issues

During the experiment, some of the microphones failed to produce a clean calibration signal. When this problem occurred, the microphone, preamplifier, and/or power supply were changed and the new serial number of each new component was used for all of the following recordings (see Table 11). In no instances did a microphone fail during a test. All issues were resolved during pre-calibrations, and none were found during post-calibrations.

Meteorological sensor issues

During the experiment, the temperature data logger on the open field tower failed. It was replaced on 24 June and all temperature and temperature/humidity sensors checked in as working. The failed data logger was sent to the manufacturer in the hopes that the data could be recovered.

However, the manufacturer was not able to find anything wrong with the logger or retrieve any data. This particular data logger was facing the Composition C-4 blasts from a distance of approximately 50 m. The proximity to the blast is a possible cause for the failure.

During the experiment, there were occasional dropped recordings. It is possible that the 1-minute polling delay on the sensors was not long enough for the number of sensors in the data loggers. Two of the sensors (listed in Table 7) failed during the experiment. They were sent to the manufacturer and found to be free of defects. The reason for failure is unknown. One additional sensor (also listed in Table 7) appeared to give questionable data. It was sent to the manufacturer and determined to be faulty and not repairable.

Sensor calibration

Microphones were calibrated both before and after each recording session using a B&K type 4228 pistonphone calibrator with appropriate adapters. The 4228 calibrators produce an equivalent level equal to 124 dB root mean square (peak sound pressure level equal to 127 dB) at 250 Hz. The calibration procedure required that the calibrator and microphone lie on the ground to minimize extraneous vibrations. Clean 5-second calibration signals at a sampling rate of 100,000 samples/second were recorded for each microphone.

The calibration procedure for the blast gauges was much more primitive. Blast gauges were struck with a hard object to induce a signal. If a clear signal was detected, the blast gauge was considered operational. Calibration values for each blast gauge were obtained from calibration sheets provided during the last manufacturer calibration. Previous experience indicates that the calibration values of the blast gauges do not change significantly over time.

Representative acoustic data

Figures 5 through 8 show samples of the pressure-time series data collected during this experiment. In each figure, the source location is BP1. Microphone locations are listed on the figures. All waveforms are calibrated pressure signatures. Each of the four source types is represented: Composition C-4, Propane Cannon, Loudspeaker, and Pistol Shot.

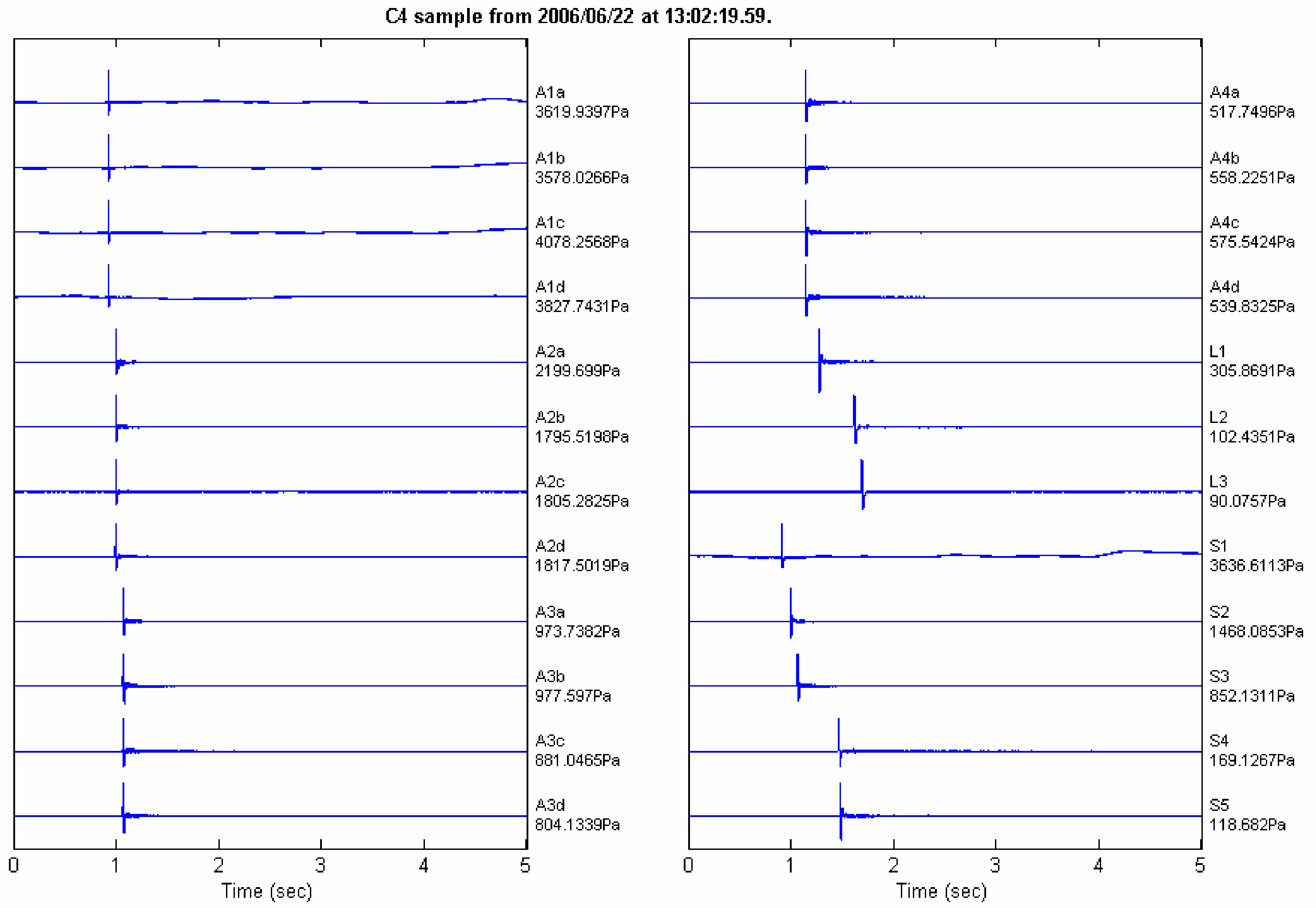

Figure 5. Sample pressure-time histories from a single shot of Composition C-4 at shot location BP1.

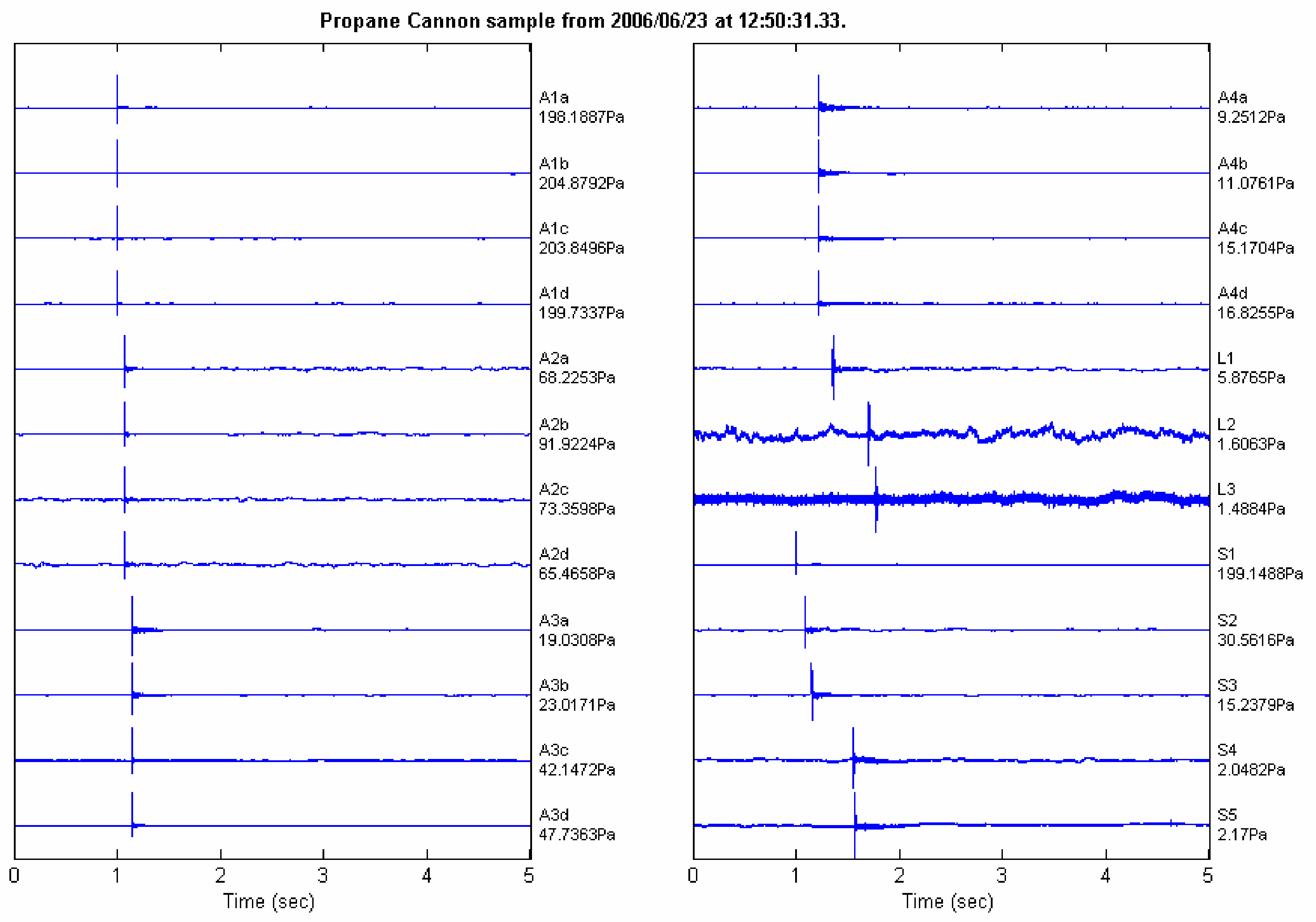

Figure 6. Sample pressure-time histories from a single propane cannon shot at shot location BP1.

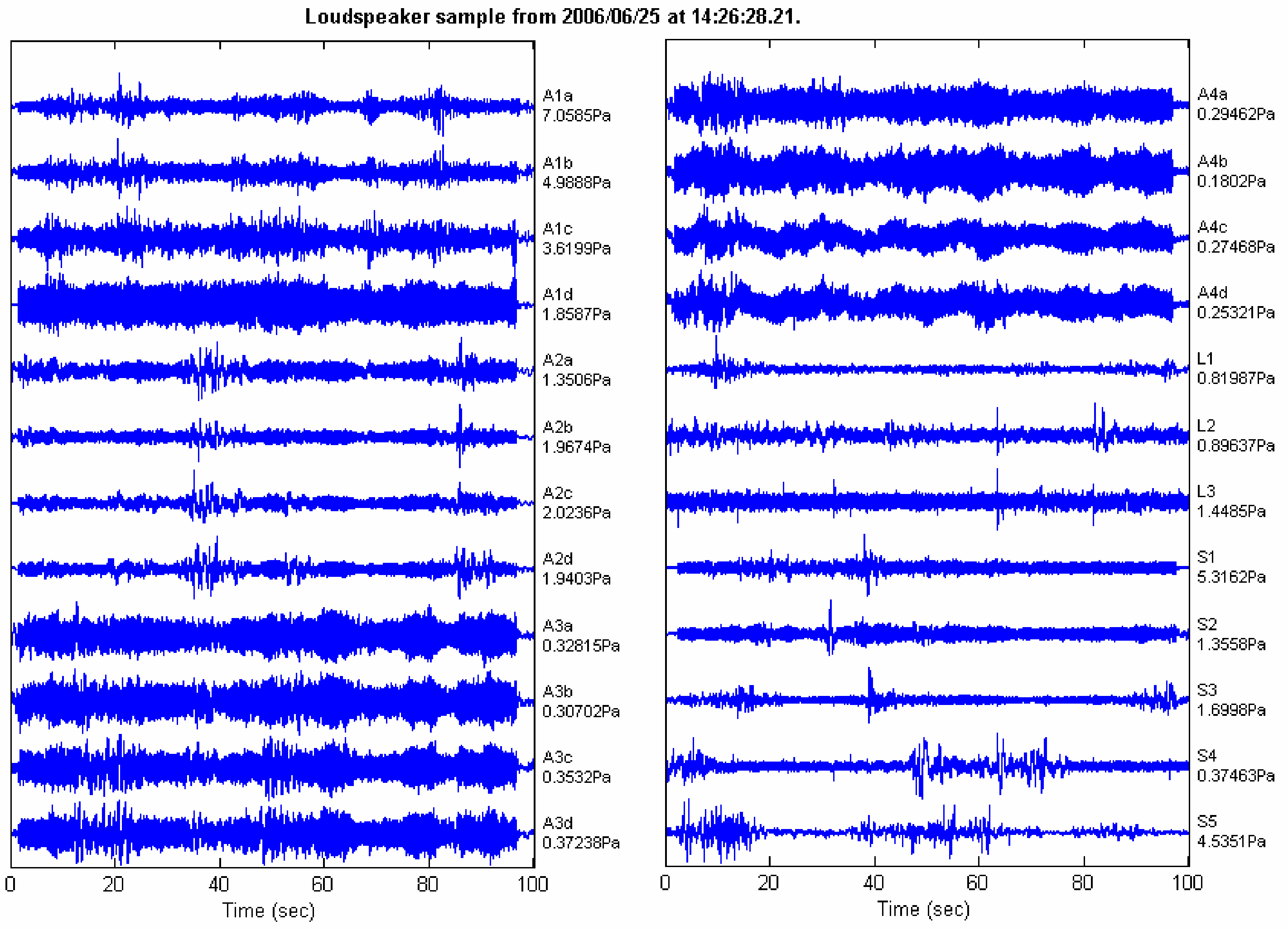

Figure 7. Sample pressure-time histories from a single loudspeaker run at shot location BP1.

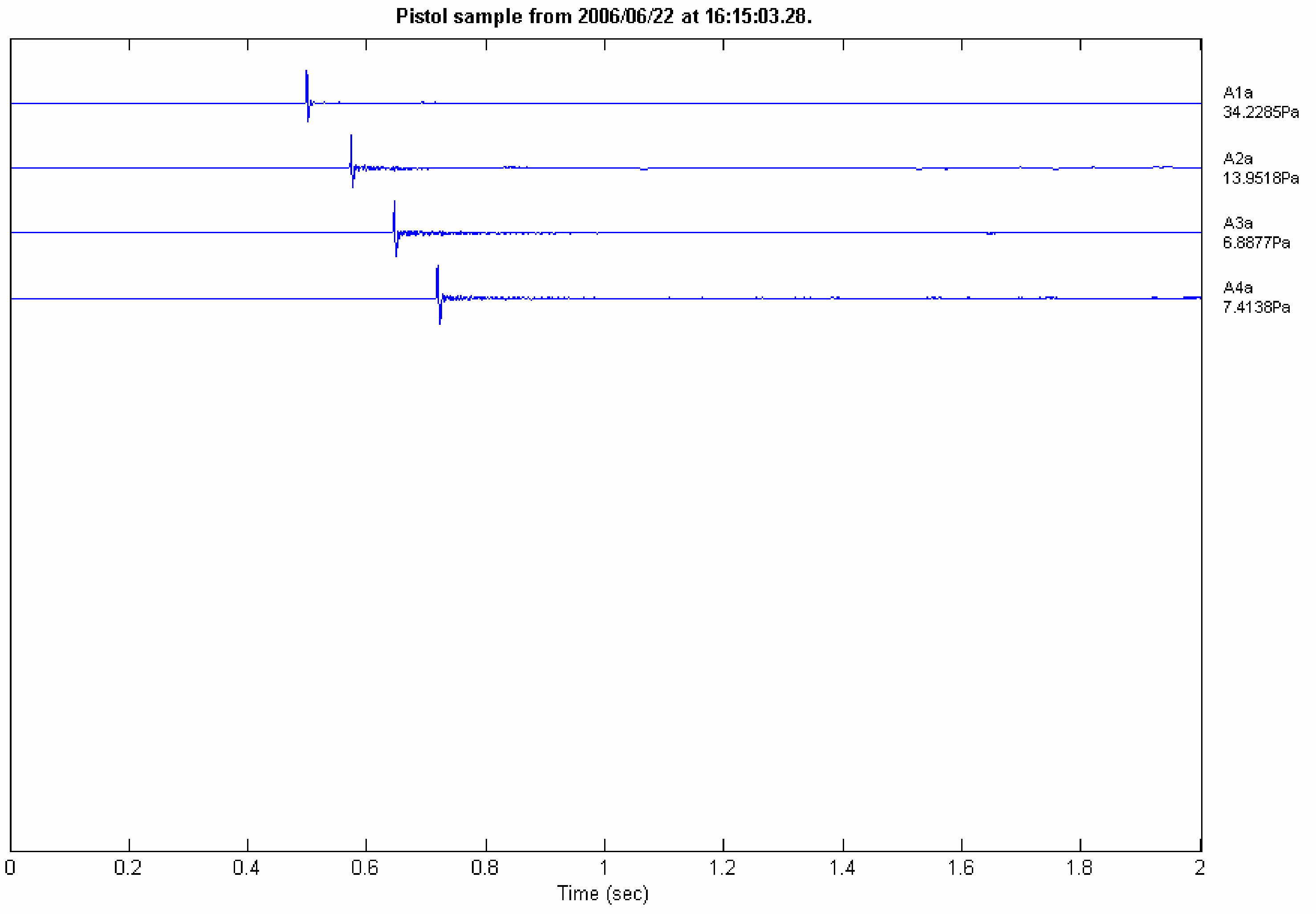

Figure 8. Sample pressure-time histories from a single pistol shot.

Representative meteorological data

The following three tables show samples of the meteorological data collected during this experiment. The three tower locations are open, edge, and forest. Data are presented for temperature and wind speed for a single time. Relative humidity and wind direction were also recorded. During the experiment, data from each sensor were recorded once every minute. Every 5 minutes the data logger performed an average over the previous five recordings and saved the 5-minute average to its storage device. All of the data available from this experiment are in the form of 5-minute averages.

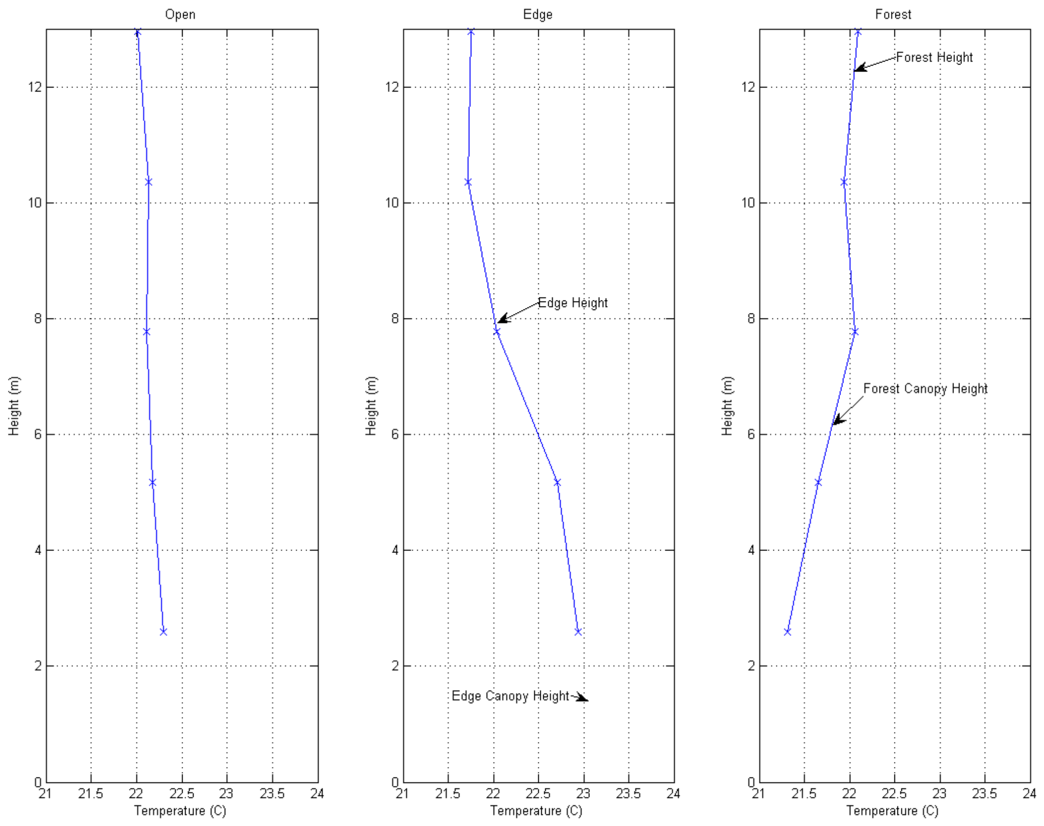

Figure 9. Temperature profiles at all three towers at the same time.

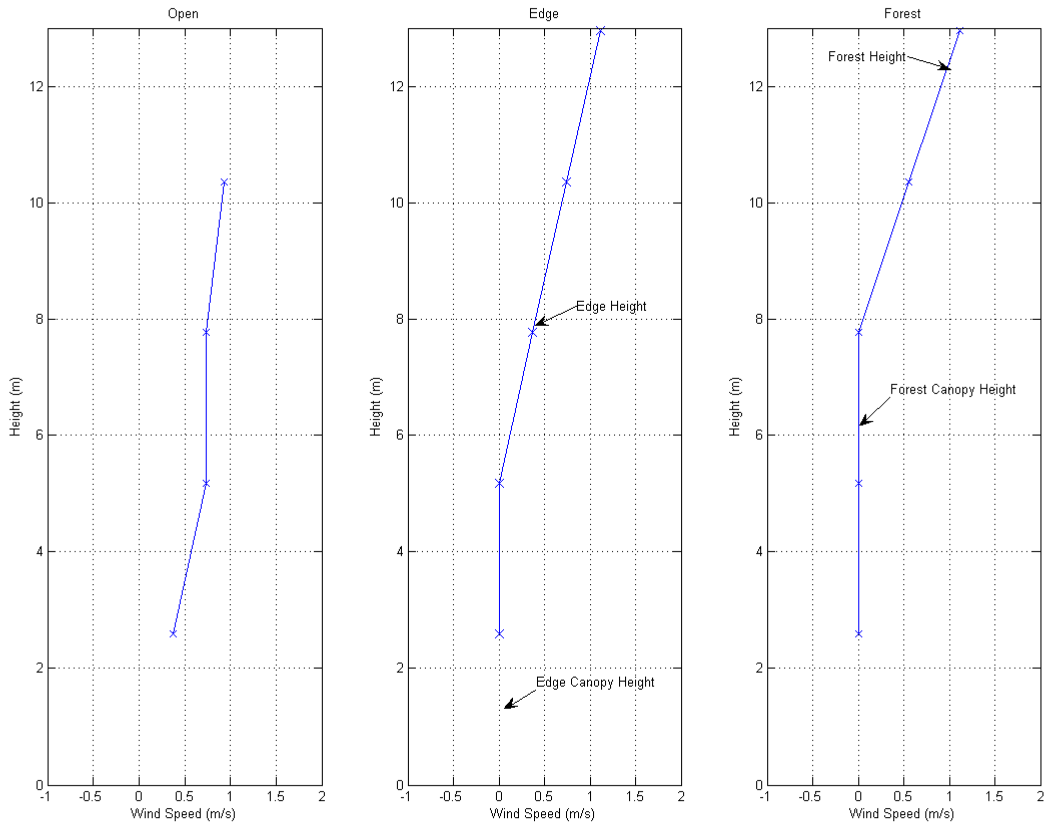

Figure 10. Wind speed profiles from all three towers at the same time.

Forest/plantation characterization

Characterization of the forest vegetation will be conducted at two levels. The primary level is the forest edge (Edge Characterization); the secondary is the forest properties in general (Stand Characterization). Edge characterization will include the measurement and characterization of all woody and foliage components from the edge of the plantation (where the trees meet the adjacent grassland) to the where the canopy height/density normalizes (non-edge plantation). There is a lack of woody/shrub component along the edge so there is no need to concentrate on forest under story components. Stand characterization will mainly focus on the corridor of trees along the microphone array, with secondary measurements taken to account for the areas not on in the corridor. The forest consisted entirely of Red Pine (*Pinus resinosa*). Figures 11–14 show varying views of the forest.

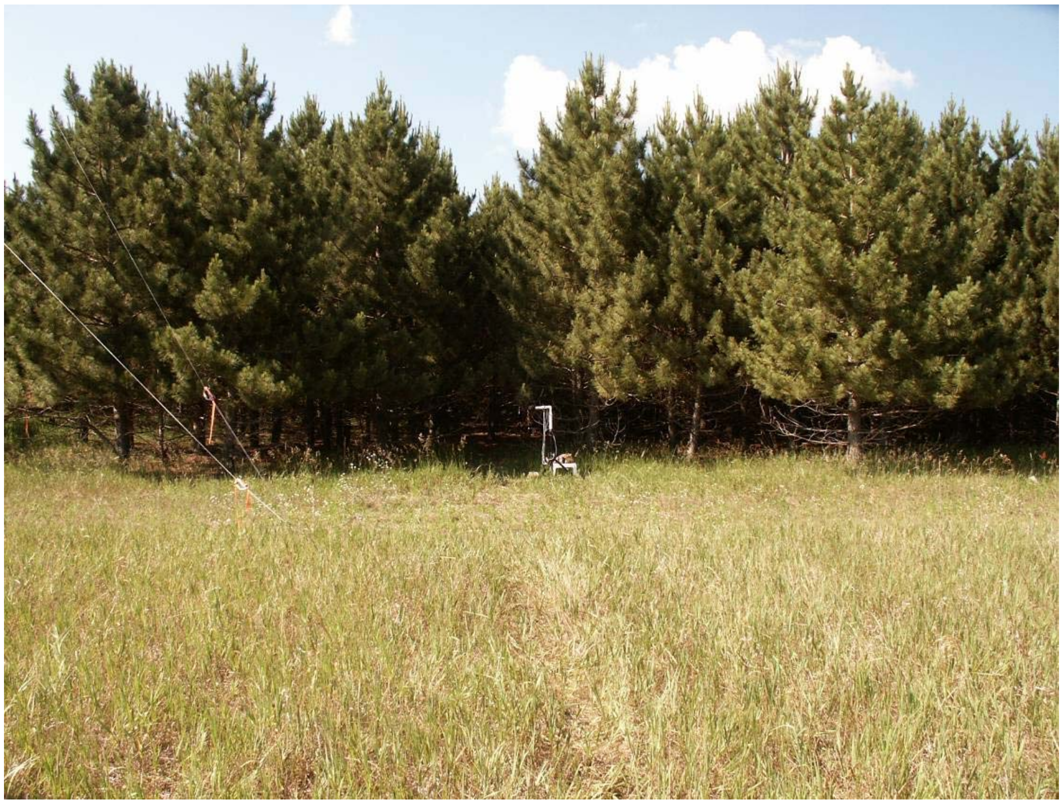
Figure 11. Forest edge photographed from the open field.

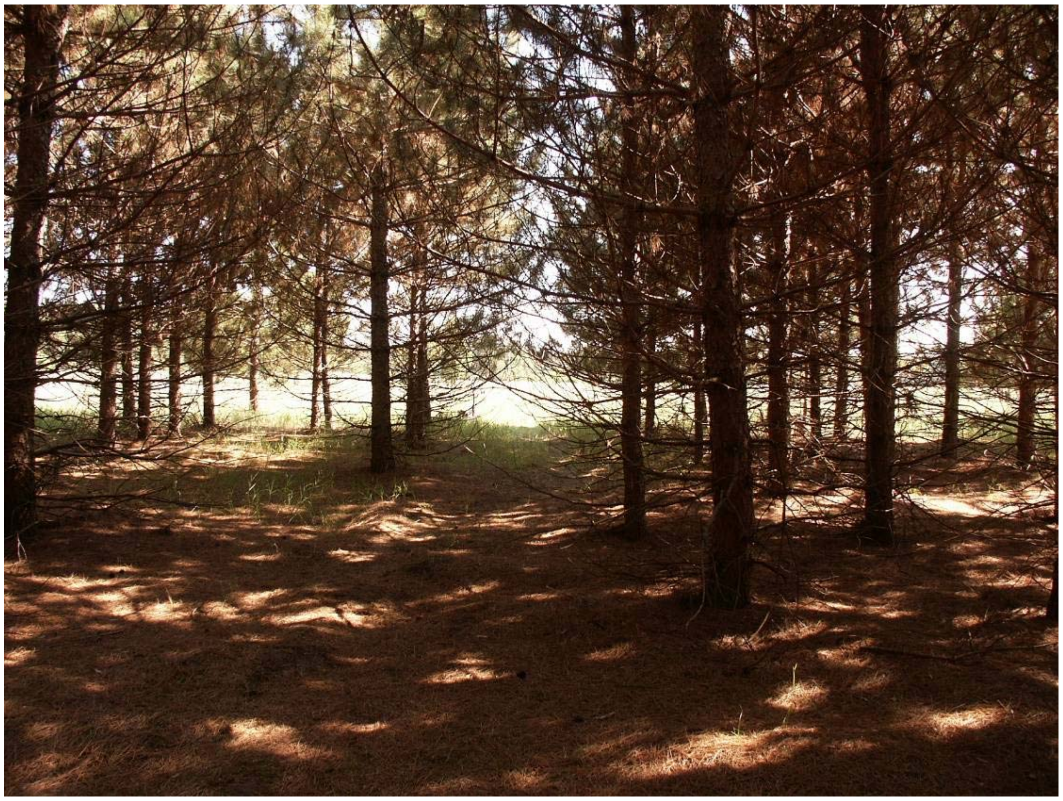
Figure 12. Forest edge photographed from inside the forest.

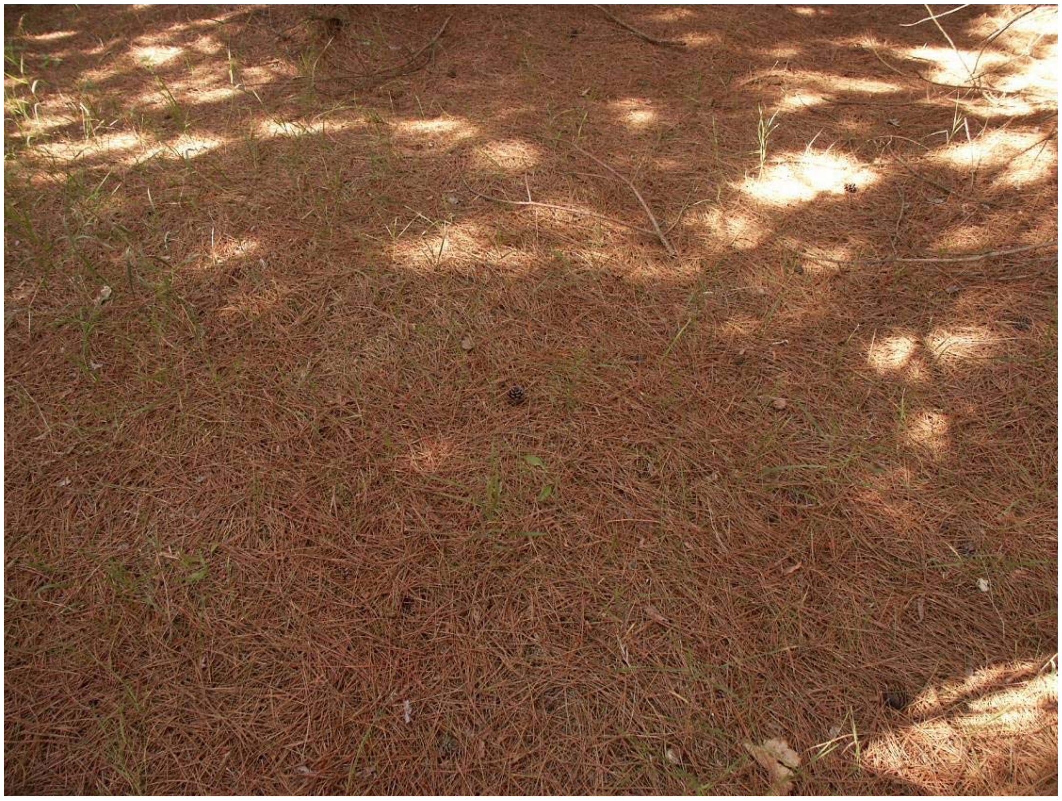
Figure 13. Typical forest floor at the test site.

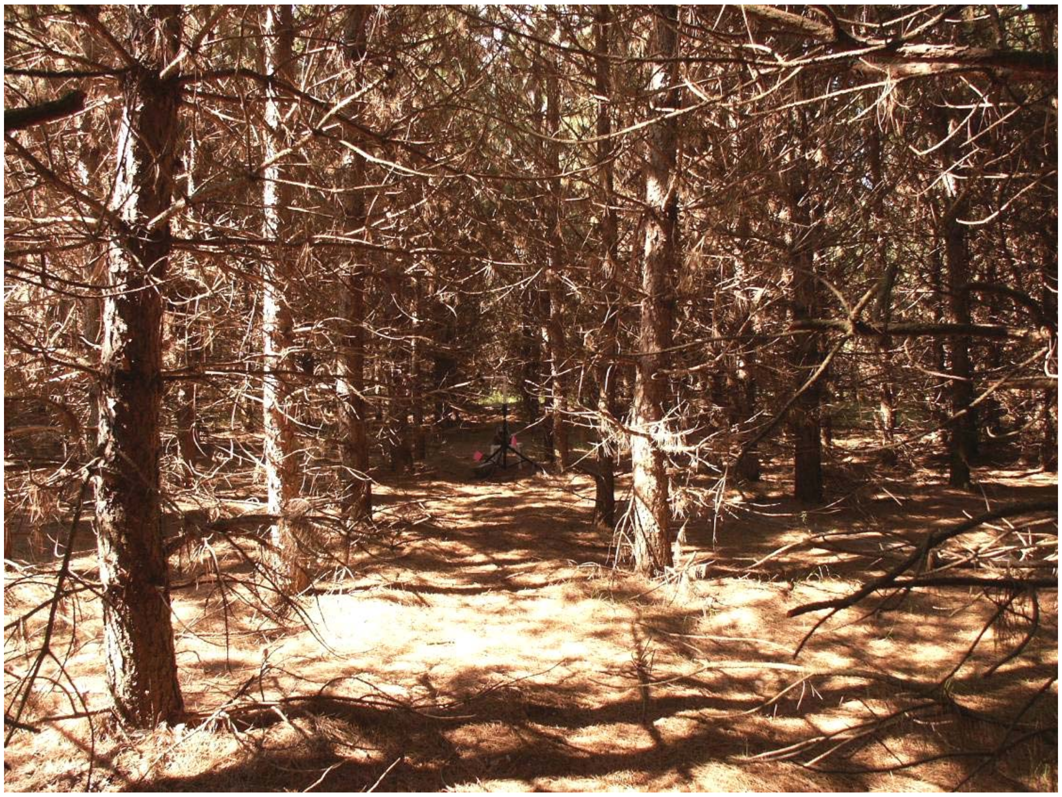
Figure 14: Forest interior with microphone set in the center of the photograph.

Edge characterization

The structure of the plantation's border between the trees and adjacent grassland is very simple. There are no shrubs/woody species along the edge other than trees, there is no tree recruitment (all trees are the same age), and there are minimal numbers of grasses and nonwoody vegetation. This structure is mainly due to the active management practices (fire) used to manage the adjacent grassland. Additionally, all trees are in distinct rows.

1. Individual tree measurements:

Individual trees in the edge were measured according to their grouping in rows parallel from the edge (i.e., first row measured, second row, etc.). Information recorded:

   a. Diameter at Breast Height (DBH)
   b. Species
   c. Location
   d. Height
   e. Height from ground to canopy

2. Tree component measurements:

For each row of trees, the region from the edge to a point where the canopy normalizes the structural components of the trees was quantified (volume of foliage, branches, and twigs (fine branches). A representative sample of trees was measured to establish parameters for the density and volumes of these materials.

Selected trees were chosen, their limbs were removed up to the canopy height, and materials were separated into branches, foliage, and twigs. Branch diameters were measured at both ends and at the center with a caliper. Additionally, heights were measured (up to the maximum height of limb removal) and a measurement was taken to account for the total radius of the on-tree foliage. Taking these steps allowed for determining the area of foliage. The three components separated from the tree were measured to account for volume of foliage, branches, and twigs. These volumes were determined by bulk immersion in a 30-gallon trash-can filled with water and measuring the displaced volume.

3. Digital imagery analysis:

In addition to the physical measurements of forest edge components; the density of vegetation on the edge was measured using digital imagery analysis. A white cloth sheet was placed at intervals behind trees along the edge and a photograph was taken. These intervals were selected based on the parallel rows of trees along the edge. Analysis software will be used to delineate the percentage of foliage and woody materials visible at each interval.

Stand characterization measurements

In addition to characterizing the forest edge, sampling was conducted to characterize stand parameters. A 100% survey was conducted along the microphone/instrument array. Based on estimates of the time interval needed to account for sound reverberations from C-4 charges, a corridor was delineated along the array and all trees were tallied. The following information was recorded for all trees within the corridor:

1. Diameter at Breast Height (DBH) (DBH Tape)
2. Species
3. Location

(Transect Width: 20-m width [10 m each side of transect line].)

Stand parameters within the corridor were measured to generally characterize the forested environment. These parameters (and methods of measurement) are:

1. Crown Closure (optical point sampler)
2. Tree Height (clinometer)
3. Canopy Height (clinometer)
4. Basal Area (calculation)
5. Trees per Acre (calculation).

The forested area outside the measurement corridor was sampled using a systemic random methodology. Variable radius plots were established within these areas using a 10 basal area factor prism. The same individual and stand level parameters were measured within these plots.

Additionally, the volume density of dead branches still attached to trunks was estimated in the plantation using methods similar to those for component measurement on the edge. Most of the trees within the plantation have retained their dead understory branches.

Forest environment

The forest was characterized thoroughly. The following section contains details of these measurements.

General forest stats

Basal Area:	125 sq ft/acre
Trees Per Acre (TPA):	327
Average DBH:	8.2 in.

Array transect stats

Transect was 20 m by 125 m centered on centerline of array. The starting (0 meter mark) point was the microphone stand on the forest edge.

Average DBH: 7.7 in.
TPA (along total array): 429

Tree Heights: 36 ft
Height to Canopy: 19 ft

Tree Height EDGE: 26 ft
Canopy Height EDGE: 5 ft

Litter characteristics

Litter was sampled at each of the microphone points along the array up to L1. Additional samples were taken at midpoints between the microphones. One-meter square areas of litter were collected to measure volume.

Table 12. Litter layer characteristics.

Sample Point	Depth (mm)	Volume (cubic cm)
A2	13	946
	31	2,120
A3	30	5,413
	41	6,852
A4	36	5,186
	17	3,066
L1	31	4,732
Woods Blast Point	28	2,839

Crown closure

Crown closure was measured at 25-m intervals starting 25 m into the woods from A2. An optical device was used.

Table 13. Crown closure.

Point	Closure (%)
25 meters in	55
A3	70
	60
A4	70
	50
flag	40
	70
L1	75
	65
Blast Point Woods	80

Tree biomass data

Several trees were destructively sampled to determine their characteristics. The following information describes these characteristics.

Branches removed up to 10 ft from trees within the woods; 15 ft for edge tree.

Tree 1

Branch radius: 7.5 ft		
DBH: 8.9 in.		
Ht: 41 ft		
Canopy: 20 ft		
Dead Branches:	Number:	22
	Avg. Length:	7.03 ft
	Avg. Fat Diameter:	1.29 in.
	Avg. Mid-point Diameter:	0.83 in.
	Avg. End-point Diameter:	0.38 in.
Biomass Volumes:	Branches: 10,410 cubic centimeter [cc])	
	Twigs: 11,356 cc	

Tree 2

Branch radius: 4.0 ft		
DBH: 6.3 in.		
Ht: 33 ft		
Canopy: 16 ft		
Dead Branches:	Number:	25
	Avg. Length:	5.12 ft
	Avg. Fat Diameter:	79 in.
	Avg. Mid-point Diameter:	0.48 in.
	Avg. End-point Diameter:	0.17 in.
Biomass Volumes:	Branches: 5,678 cc	
	Twigs: 2,839 cc	

Tree 3

Branch radius: 6.0 ft		
DBH: 8.7 in.		
Ht: 45 ft		
Canopy: 26 ft		
Dead Branches:	Number:	31
	Avg. Length:	5.08 ft
	Avg. Fat Diameter:	1.12 in.

	Avg. Mid-point Diameter:	0.74 in.
	Avg. End-point Diameter:	0.43 in.

Biomass Volumes:	Branches: 11,356 cc
	Twigs: 8,517 cc

Tree 4

Branch radius: 9.0 ft
DBH: 9.4 in.
Ht: 38 ft
Canopy: 20 ft

Dead Branches:	Number:	39
	Avg. Length:	5.07 ft
	Avg. Fat Diameter:	0.95 in.
	Avg. Mid-point Diameter:	0.65 in.
	Avg. End-point Diameter:	0.39 in.

Biomass Volumes:	Branches: 13,249 cc
	Twigs: 12,303 cc

Edge Tree

Branch radius: 7.3 ft (~3.5 at 15 in.)
DBH: 8.0 in.
Ht: 26 ft
Canopy: 5 ft

Dead Branches:	Number:	10
	Avg. Length:	6.32 ft
	Avg. Fat Diameter:	1.3 in.
	Avg. Mid-point Diameter:	0.81 in.
	Avg. End-point Diameter:	0.38 in.

Live Branches:	Number:	21
	Avg. Length:	7.86 ft
	Avg. Fat Diameter:	1.6 in.
	Avg. Mid-point Diameter:	1.03 in.
	Avg. End-point Diameter:	0.48 in.

Biomass Volumes:	Live Branches: 31,230 cc
	Live Twigs: 16,088 cc
	Needles: 16,088 cc
	Dead Branches: 7,571 cc
	Dead Twigs: 1,893 cc

Given the edge extends 5 m in (material collected within 15 ft of ground, 10 trees within edge), frontage 20 m (100 m^2 area):

0.007 m^3 (7287 cc) of canopy material per square meter
0.003 m^3 (3123 cc) of live branch material per square meter
0.002 m^3 (1609 cc) of needle/live twig material per square meter

Archival data

All binary acoustic files from the Yokogawa oscilloscope, along with Matlab scripts to reduce the data into Matlab format with physical units, are included on the available data disk (see footnote, p 2). Raw weather data in Microsoft Excel format are included. An archive of site photographs and a readme file that describes file structure and simple directions on using the Matlab scripts are also included.

3 Summary and Conclusions

The experiment was successful in obtaining excellent quality acoustical data illuminating the effect of a forest edge on acoustic propagation. No acoustic equipment (e.g., sensors, power supplies, recording devices) failed during data-taking sessions. The vast majority of acoustic signals recorded are easily distinguishable above the background noise, and only one data point out of several thousand was clipped.

The weather was good during the bulk of the experiment, with rain causing only minimal delays. Winds were generally light, probably a desirable attribute for this measurement. While there are some minor deficiencies in the meteorological data, they are not insurmountable. Analysis of the data will determine whether the experiment has answered the question of "what is the acoustic effect of a forest edge."

References

The field procedures and measurement practices used here were adapted from the general guidance found in American National Standards Institute (ANSI) S12.7, ANSI S12.18, American Society for Testing and Materials (ASTM) E1503, and ASTM E1779.

ANSI S12.7-1986 (R2006) – American National Standard Methods for Measurements of Impulse Sound, Acoustical Society of America, New York, 1986.

ANSI S12.18-1994 (R2006) – American National Standard Methods for Outdoor Measurement of Sound Pressure Level, Acoustical Society of America, New York, 1986.

ASTM E-1503-05 – Standard Test Method for Conducting Outdoor Sound Measurements Using a Digital Statistical Analysis System. ASTM International, 2005.

ASTM E-1779-96a (Reapproved 2004) – Standard Guide for Preparing a Measurement Plan for Conducting Outdoor Sound Measurements. ASTM International, 2005.

Guice et al. 1998. Impulsive noise measurements in a forest during summer and winter conditions. *NCEJ* 46, 185-189.

Appendix: Matlab Programs

The first set of scripts shown below was used to convert the raw data from the Yokogawa to a Matlab-friendly format. These scripts were developed for previous projects using the same Yokogawa digital oscilloscopes. The data from the Yokogawa included a file for each waveform and a header with pertinent information such as sampling rate, shot time, etc. The functions `organizeWaveforms`, `parseHeader`, `writeSensitivities`, and `findSensitivity` were used to extract this information and reorganize it for easy analysis in Matlab.

The second set of scripts (beginning p 43) was written specifically for this project. The function `ploton1` is used to view all microphone signals for a single shot simultaneously. This function is useful in observing the change in the waveform as it travels through the air. The function `plotsingle` plots the waveform for a single microphone. This function offers a quick and easy way to view any one channel's wave. `WaveandSpectra` similarly plots a single waveform, but also includes a one-third octave band frequency analysis. The functions `getPeaks`, and `plotPeaks` were designed to analyze changes in peak levels versus distance from the source. `getPeaks` determined the peak levels for each waveform and saved them in a matrix called `peaksTable.mat`, with columns indicating which source, blast point, and array configuration were used. The function `plotPeaks` loads the matrix created by `getPeaks` and then creates graphs relating these peak levels to each microphone's distance from the source. `PeaksPerShot` provides a view of the peak level of each microphone location for individual shots of a given source, source location, and array configuration.

To observe effects on frequency content as the waves traveled through the forest, another set of scripts was developed. The function `SELtest2` takes a single waveform and performs a one-third octave band frequency analysis. `SELavg2` provides a way to calculate sound exposure levels for all shots for a particular source type and blast point. The results can either be returned for each shot or can be averaged then returned. `collectSEL2` was written to assemble the matrix `tobSEL_all.mat` which contains data for all the C-4 and propane cannon shots, along with their overall SEL and one-third octave band spectra. `SELperShot` works like `PeaksPerShot,` except that it displays overall SEL values instead of peak pressures.

The comments accompanying each function provide more detailed explanations.

```
function organizeWaveforms(rootDir)
%*************************************************************************
%Organizes and loads the .wvf and .hdr files corresponding to a day of
%events into separate directories, saving the waveform and header
%information into .mat files
%Written for experiments in Edgewood, MD June 2004 and in Blossom Point, MD
%August 2004.
%Version 1.0 completed 24 JUN 2004 by:
%Tim Eggerding
%(t-eggerding@cecer.army.mil)
%
%Update history:
%   12 AUG 2004:    Use of fread on .wvf file changed such that the header
%                   data is read initially and then thrown out, and the
%                   waveform data is read as 16 bit integers and saved as
%                   an array of 16 bit integers.  fread defaults to a
%                   return type of a double array, which resulted in memory
%                   errors.  Casting to double now occurs during the
%                   process of breaking the .wvf file into seperate
%                   waveforms for each channel and converting from 16 bit
%                   integers to 64 bit double representing accurate
%                   voltages.
%                   NOTE:  MATLAB seems to run out of memory at about
%                   50000000 doubles in one array due to contigious memory
%                   issues.  If more than 50000000 samples exist in a
%                   waveform, more than one array must be created in order
%                   to store the samples with double precision.
%
%   17 AUG 2004: Changed handling of filenames to allow for arbitrary
%                 length file names as opposed to restricting to Yokogawa-type
%                 4-character file names.
%   24 SEP 2004: Changed the variable date to shotDate to avoid conflicts
%                 built-in MATLAB function date()
%*************************************************************************
%
%Referring functions:   yokogawaDataGUI.m
%
%
%Required functions:    parseHeader.m
%
%global variables:      none
%
%Inputs:    rootDir     Root directory for the day
%
%Outputs:   shotTimes.mat   Saves an array of times of events
%           channels.mat    Saves a matrix of valid channels for each event
%
cd(rootDir);
D = dir(rootDir);       %Creates a vector listing all the files in the directory
numFiles = length(D);   %numFiles = the number of files in the directory
shotTimeArray = [];
channelArray = [];
for i = 1:numFiles      %Look through all the files
  wvfName = D(i).name;  %Get the current file name
  if(~isempty(strfind(wvfName,'WVF')))  %If the extension is .WVF
      cd(rootDir)
      nameLength=length(wvfName)-4;     %take the filename without the .WVF
      hdrName = [wvfName(1:nameLength)  '.HDR'];  %Compose the name for the corresponding
.HDR %file
      hdrID = fopen(hdrName, 'r');      %Open the .HDR file
      if hdrID == -1                    %Warn that the header was not found if fopen re-
turns -1
          warning(['Header file ' hdrName ' for waveform file ' wvfName ' does not ex-
ist.']);
      else
            %Load the header information using parseHeader.m
          [endian, dataFormat, dataOffset, traceTotalNumber, traceNames, blockSize,
vResolution, vOffset, vDataType, period, dateCell, timeCell] = parseHeader(hdrName);
```

```
          if strcmp(endian, 'Big')
              wvfID = fopen(wvfName,'r', 'ieee-be');    %Use big endian encoding if nec-
essary
          else
              wvfID = fopen (wvfName, 'r', 'ieee-le');  %Use little endian encoding if
necessary
          end
          shotTimeArray = [shotTimeArray; timeCell(1)]; %Append the time of the current
event to %the shotTimeArray
          if(length(traceNames)<16)
              tName16 = cell(1,16);
              tName16(1:length(traceNames)) = traceNames;
          else
              tName16 = traceNames;
          end
          channelArray = [channelArray; tName16];   %Append the channels of the current
event to %the channelArray
          DataType = char(vDataType(1));            %Put the data type into a character
array %(string)
          time = char(timeCell(1));                 %Put the time into a character array
(string)
          shotDate = char(dateCell(1));                 %Put the date into a character
array %(string)
          dataSize = 8*str2num(DataType(3));        %Find the size of the data in bits
          dataString = num2str(dataSize);           %Put that size into a a string
          shotTime = [time(1:2) '.' time(4:5) '.' time(7:9)];     %Use the time informa-
tion to %make a valid directory name
                                                    %In this case, hh.mm.ss (you can't
use a %colon in directory names).
         if ~(exist(shotTime, 'dir') == 7)         %Make sure that this event hasn't al-
ready been %done
             shotTime
             mkdir(shotTime);                      %Make the subdirectory to hold this
event,
             cd(shotTime);                         %and enter it.
              pack;
              headerData = fread(wvfID,dataOffset/2, ['*int' dataString]);   %Read the
header %data
              clear headerData
              for m = 1:length(traceNames)      %For each trace in the header file
                 waveform = fread(wvfID, blockSize(m), ['*int' dataString]);
                 waveform = double(waveform)*vResolution(m);
                 waveform = waveform + vOffset(1);
                 dirName = char(traceNames(m));     %Set the directory name for the cur-
rent %channel's data
                 mkdir(dirName);                    %Make that directory
                 cd(dirName);                       %Enter that directory
                 hdrname =sprintf('eventHeader.mat');    %Make the name for the header
.mat file
                 wvfname = sprintf('eventWaveform.mat');%Make the name for the waveform
.mat file
                 save eventWaveform waveform;               %Save the waveform
                 traceName = char(traceNames(m));           %Make a string out of the
current %channel name
                 samplingRate = 1/period(m);                  %Make a sampling rate in-
stead of %period information
                                                                    %(more useful in
later code)
                 save eventHeader traceName samplingRate shotDate time;  %Save the header
                 clear waveform;                %Delete the waveform variable to save
memory
                 cd ..                          %Go back to the root directory for the
event time
              end
          end
          cd(rootDir)             %Back to the day's root directory
          save shotTimes shotTimeArray;      %Save the list of event times
          save channels channelArray;        %Save the list of channels for the events
          fclose(wvfID);    %close the .WVF file
          fclose(hdrID);    %close the .HDR file
          end
     end
  end
  clear;
  pack;
return;
```

```
function [endian, dataFormat, dataOffset, traceTotalNumber, traceNames, blockSize, vReso-
lution, vOffset, vDataType, period, date, time] = parseHeader(file)
%***************************************************************************
%Loads the header .HDR file for Yokogawa DL750 waveform objects, and
%extracts necessary data out of it.
%Written for experiments in Edgewood, MD June 2004 and in Blossom Point, MD
%August 2004.
%Version 1.0 completed 24 JUN 2004 by:
%Tim Eggerding
%(t-eggerding@cecer.army.mil)
%
%Update history:
%13 JUL 2004 - added "thisTrace" index in order to handle cases where a
%       group is not full.
%
%***************************************************************************
%
%Referring functions:   organizeWaveforms.m
%
%
%Required functions:    none
%
%global variables:      none
%
%Inputs:    file                File name for the header
%
%Outputs:   endian              Holds whether big-endian or little-endian
%                                   format
%           dataFormat          "Trace" or "Block".  "Trace" is assumed
%                               througout the rest of the code, because "Block" program-
ming
%                               does not make sense with the gathered data.  "Block" has
to do
%                               with saving data over a long period of time, which is not
the
%                               concern of this study.  However, this variable is in-
cluded for
%                               completeness.
%           dataOffset          How many bytes within the .WVF file the header informa-
tion takes %up.
%           traceTotalNumber    How many channel's worth of data is saved in this header.
%           traceNames          Array of the channel names
%           blockSize           How many data points each wave consists of
%           vResolution         The resolution of the screen in volts.
%           vOffset             Displacement of the blast wave from 0 volts
%           vDataType           Encoding of the how the data points should
%                               be interpreted (integer/flop, signed/unsigned, number of
bytes)
%           period              The period of sampling
%           date                Date of event
%           time                Time of event
%
%   All data that contains textual information is recorded as a cell array
%   of strings, while numbers are returned as a numeric array.
%
%   See the DL750 operating manual page APP-9 for more information on the
%   structure of the .HDR file.
%   Also see the Yokogawa Technical Information article TI 7000-21E
%   entitled "Understanding the structure of Binary data (xxxxxxxx.WVF)
%   file created by the DL, AR series."  A copy of this article should be
%   with the DL750 manual at CERl, but can also be found at:
%   http://www.yokogawa.com/tm/pdf/tutorial/tm-technical_dlar.pdf
headerText = textread(file, '%s');      %Read in the header file as a cell array of
strings
endian = char(headerText(11));          %Store the endian information as a string
dataFormat = char(headerText(13));      %Store the data format ast a string
groupNumber = str2num(char(headerText(15)));        %Store the number of groups
traceTotalNumber = str2num(char(headerText(17)));   %Store the total number of channels
dataOffset = str2num(char(headerText(19)));         %Store the number of bytes of header
%information
%index variables
%   i: current group
%   j: current position in headerText string array
%   k: current trace number within group
j = 22;     %j points to the point in the headerText array that holds the current group
name
%For loop to iterate through groups
thisTrace = 1;
for i = 1:groupNumber
    traceNumber = str2num(char(headerText(j)));     %The number of traces (channels) in
the group
    blockNumber = str2num(char(headerText(j+2)));   %Number of blocks per trace.  For the
MD %experiments,
                                 %this should always be 1.  See the manual for the dif-
ference b/w
                                 %Block and Trace settings.
   %For loop to iterate through traces within the group
    for k = 1:traceNumber
            traceNames(thisTrace) = headerText(j+3+k);    %Name of the current trace
            blockSize(thisTrace) = str2num(char(headerText(j+3+traceNumber+1+k)));
                %Number of data points saved in this trace's waveform
            vResolution(thisTrace) = str2num(char(headerText(j+3+2*(traceNumber+1)+k)));
                %Resolution of the data in volts
            vOffset(thisTrace) = str2num(char(headerText(j+3+3*(traceNumber+1)+k)));
                %DC offset of the data
            vDataType(thisTrace) = headerText(j+3+4*(traceNumber+1)+k);
                %Code to describe the size and format of data points
            period(thisTrace) = str2num(char(headerText(j+3+11*(traceNumber+1)+k)))  ;
                %Sampling period
            date(thisTrace) = headerText(j+3+14*(traceNumber+1)+k);
                %Date of event
            time(thisTrace) = headerText(j+3+15*(traceNumber+1)+1);
                %Time of event
                thisTrace = thisTrace+1;
    end
    j=j+3+16*(traceNumber+1)+2; %Increment j to point at the next group
end
```

```
function writeSensitivities(calLevel, calFreq,directory)
%**************************************************************************
%Loop through a directory of calibration signals and write the sensitivies
%given by findSensitivity
%Written for experiments in Edgewood, MD June 2004 and in Blossom Point, MD
%August 2004.
%Version 1.0 completed 24 JUN 2004 by:
%Tim Eggerding
%(t-eggerding@cecer.army.mil)
%
%Update history: none.
%
%**************************************************************************
%
%Referring functions:   yokogawaDataGUI.m
%
%
%Required functions:    findSensitivity.m
%                       parseHeader.m
%
%global variables:      none
%
%Inputs:    calLevel        SPL of calibrator in dB
%           calFreq         Frequency of calibrator in Hz
%           directory       Directory containing calibration files
%
%Outputs:   sensDat.mat     Calibration in V/Pa
%
%
cd(directory);
D = dir;                %Creates a vector listing all the files in the directory
numFiles = length(D);   %numFiles = the number of files in the directory
for i = 1:numFiles      %Loop through the files
  wvfName = D(i).name;  %Take the name of the current file
  if(~isempty(strfind(wvfName,'WVF')))  %If this is a .WVF file (thus, a calibration)
      hdrName = [wvfName(1:4) '.HDR'];  %Create the name of the appropriate header file
      hdrID = fopen(hdrName, 'r');      %Open the header file
      if hdrID == -1
          %Warn if the header is not found
          warning(['Header file ' hdrName ' for waveform file' wvfName ' does not ex-
ist.']); %display error message if no header file.
      else
          %Get the necessary data from the header file
          %In the case of a calibration, variables like traceNames, which
          %are usually arrays, will only have one value, since only one cal
          %is saved at a time.
          [endian, dataFormat, dataOffset, traceTotalNumber, traceNames, blockSize,
vResolution, vOffset, vDataType, period, date, time] = parseHeader(hdrName);
          traceNames = char(traceNames)                       %Cast traceNames as a
string
          vDataType = char(vDataType);                         %Cast vDataType as a
string
              if strcmp(endian, 'Big')
                     wvfID = fopen(wvfName,'r', 'ieee-be');         %Big-endian if neces-
sary
                 else
                     wvfID = fopen (wvfName, 'r', 'ieee-le');       %Little-endian if
necessary
                 end
                 headerData = fread(wvfID,dataOffset/2, ['*int16']);   %Read the header
data
              clear headerData
          %and throw it out

          for m=1:size(traceNames,1)
            if(~(exist(traceNames(m,:), 'dir') == 7))                  %If a calibration
for this %channel has not occured

                 mkdir(traceNames(m,:));                        %Make a directory for this
channel
                 cd(traceNames(m,:));                                      %Enter that di-
rectory
                  waveform = fread(wvfID, blockSize(m), ['*int16']);
```

```
                    sensDat = findSensitivity(double(waveform)*vResolution(m), 
1/period(m), calFreq, calLevel);
                     filename = ['sensDat.mat'];
                     save(filename, 'sensDat');
                     cd ..

            end
          end
      end
  end
end
fclose('all')
cd ..
```

```
function [sensitivity]=findSensitivity (data,samplingRate, calFreq, calLevel)
%**************************************************************************
%Takes the FFT of the data and eliminates the high and low frequency data
%of the calibration signal.  The inverse FFT is then taken and the
%sensitivity is found in V/Pa by Vrms/Prms.
%Written for experiments in Edgewood, MD June 2004 and in Blossom Point, MD
%August 2004.
%Adopted from code written by Ryan Lee for the Texarkana experiments.
%Version 1.0 completed 24 JUN 2004 by:
%Tim Eggerding
%(t-eggerding@cecer.army.mil)
%
%Update history: none.
%
%**************************************************************************
%
%Referring functions:   writeSensitivity.m
%
%
%Required functions:    none
%
%global variables:      none
%
%Inputs:    data            Calibration signal waveform given in Volts
%           samplingRate   Rate of sampling from the scope
%           calFreq         Frequency of the calibrator used in Hz
%           calLevel        SPL of the calibrator used in dB
%
%Outputs:   sensitivity     Calibration in V/Pa
%
%

p0 = 20*10^-6;                                  %Reference pressure in Pascals
lengthData = length(data);
lengthFFT = 2^(nextpow2(length(data))-1);       %This will be the length of the FFT
                                                %and the "-1" insures that our data will
be %truncated (rather
                                                %than zero padded) in the FFT algorithm

freqRes = samplingRate/lengthFFT;           %Resolution of frequencies in the frequency
domain


H = fft (data,lengthFFT); % This is the FFT in "H"
                                    % IMPORTANT NOTE: H(1) is really the element of the
                                    % FFT that is at a frequency of 0.  Matlab does not
                                    % index any matricies beginning with 0, it always be-
gins
                                    % with an index of 1.  Therefore H(0) is improper in
                                    % Matlab.
cutoff_1 = round(calFreq*2^(-1/12)/freqRes);        %Calculate cutoff frequencies to
eliminate %noise
cutoff_2 = round(calFreq*2^(1/12)/freqRes);         %calFreq +/- a semitone
cutoff_3 = length(H) - cutoff_2;                    %The negative cutoff frequency of
cutoff_1, %since
cutoff_4 = length(H) - cutoff_1;                    %the values of a DFT greater than pi
are %actually
                                                    %negative frequencies

                            %Zero out all unwanted low and high frequencies from
H(1:cutoff_1) = 0;          %zero to one semitone below our calibration frequency

                            %Zero from one semitone above our calibration frequency to
one
                             %semitone
H(cutoff_2:cutoff_3) = 0;   %below the negative calibration frequency


                            %Zero From one semitone above the negative calibration fre-
quency to
H(cutoff_4:length(H)) = 0;  %the end of the FFT.

data = ifft(H,length(H));   %New time-based signal.  This should be the original
```

```
                            %cal signal minus any unwanted noise.  Its mean should be
zero.

Vrms = sqrt(sum(abs(data.^2))/length(data));    %RMS voltage found by the square root of
the mean
                                                %of the square of the data.
Prms = 10^(calLevel/20)*p0;                     %RMS pressure found by the inverse of the
SPL
                                                %equation
                                                %SPL(dB) = 20*log10(Prms/P0rms)


sensitivity = -Vrms/Prms;                        %Microphone sensitivity in V/Pa.

return
```

```
function ploton1(rootdir)
%************************************************************************
%Input: directory for one shot
%Output: none
%Description: Loads each channel's waveform for a single shot and plots the calibrated
%signals all on one graph in Pa over seconds.
%
%Requires: eventWaveform.mat,eventHeader.mat,sensDat.mat,MicLocations.mat
%
%Written for experiments at Camp Ripley, June 2006
%by Tim Onder (Timothy.E.Onder@erdc.usace.army.mil)
%***********************************************************************

cd (rootdir);                   %Takes as input the directory of one shot
D = dir(rootdir);
numFiles = length(D);           %Finds how many channels in the directory
loc=cell(numFiles-2, 1);        %Preallocate space for legend array
count=0;                        %Initialize count variable

for i = 1:(numFiles)            %Scrolling through all channels
    channel=['CH' num2str(i)];

    if exist(channel,'dir')~=7  %Checks if there's data for the channel
        continue
    end

    count=count+1;
    cd (channel)                %Opens folder for each channel
    load eventWaveform;         %and loads waveform in Volts/Sample
    load eventHeader;           %as well as samplingRate, shotDate, and time
    cd (['../../PRECAL/' channel])
    load sensDat;               %Gets sensitivity data from calibration

    cd(['../../..']);                                       %loads mic locations for ei-
ther the
    load MicLocations.mat;                                  %forest scope or the trailer
for
    if length(findstr(rootdir, 'Forest'))~= 0               %display in legend
        loc{count,1} = ForestMicLocations{i,1};

    elseif length(findstr(rootdir, 'Trailer'))~= 0
        loc{count,1} = TrailerMicLocations{i,1};
    end


    timeaxis=[1:(length(waveform))]'/samplingRate;          %Sets up the X-axis to show
time (s)
    calwaveform = waveform/sensDat;                         %Converts waveform to Pascals
    if (mod(i,2)==0)
        plot (timeaxis,calwaveform, 'r')                  %and plots calibrated waveform
    else                                                  %alternately in red and blue
        plot (timeaxis,calwaveform, 'b')
    end

    hold on                 %holds plot so others will plot on top of it
    cd (rootdir)
end

hold off
xlabel('Time (s)');
ylabel('Pressure (Pa)');
title(['Sample from ' shotDate ' at ' time '.']);
legend(loc(1:count),'Location','NorthEastOutside');
```

```
function plotsingle(rootdir, chanNum)
%*********************************************************************
%Input: directory for one shot, channel number
%Output: none
%Description: This function will take as input the directory for a single shot and the
%channel whose plot is desired. It will then plot the calibrated waveform in
%Pascals. It will also indicate the peak level of the waveform.
%
%Requires: eventWaveform.mat,eventHeader.mat,sensDat.mat,MicLocations.mat
%
%Written for experiments at Camp Ripley, June 2006
%by Tim Onder (Timothy.E.Onder@erdc.usace.army.mil)
%*********************************************************************

channel = ['CH' num2str(chanNum)];
cd (channel)
load eventHeader.mat;                                    %Load samplingRate, shotDate,
time
load eventWaveform.mat;                                  %and the uncalibrated waveform

cd (['../../PRECAL/' channel]);
load sensDat.mat;                                        %Load sensitivity data

timeaxis=[1:(length(waveform))]'/samplingRate;           %Sets up the X-axis to show time
(sec)
calwaveform = waveform/sensDat;                          %Converts waveform to Pascals
peakLevel = max(abs(calwaveform));                       %Calculates peak level (pos or
neg)

cd(['../../..']);                                        %loads mic locations for either
the
load MicLocations.mat;                                   %forest scope or the trailer for
if length(findstr(rootdir, 'Forest'))~= 0                %display in legend
    loc = ForestMicLocations{chanNum,1};

elseif length(findstr(rootdir, 'Trailer'))~= 0
    loc = TrailerMicLocations{chanNum,1};
end

plot(timeaxis,calwaveform)                               %Plots calibrated waveform
xlabel('Time (s)');
ylabel('Pressure (Pa)');
title(['Sample from ' shotDate ' at ' time '.   Peak Level = ' num2str(peakLevel) '
Pa.']);
legend(loc,'Location','NorthEastOutside');

cd (rootdir)
```

```
function plotPeaks (srcin, srcloc, config)
%**********************************************************************
%Input:     (all integers) source type, source location, array config. Based on the fol-
lowing table:
%
%       SOURCE      LOCATION    CONFIG
%       1 = C-4      1 = BP1     1 = Parallel
%       2 = PC      2 = BP2     2 = Perpendicular
%                   3 = BP3     3 = Vertical
%                   4 = BP4     4 = All
%
%Ouput: none
%Description:   Creates one subplot that shows the peak level for every mic
%in dB over distance from source. A second subplot shows a horizontal cross
%section of the field (with the forest edges indicated) and shows the peak
%levels at each mic location.
%
%Requires: peaksTable.mat
%
%Written for experiments at Camp Ripley, June 2006
%by Tim Onder (Timothy.E.Onder@erdc.usace.army.mil)
%*********************************************************************

load ('N:\Research\Acoustics\Camp Ripley Matlab Format\Pk vs Dist Analy-
sis\peaksTable.mat');
clf;
A1=zeros(length(peaksTable),2); A1cnt=0;             %Initializes a matrix for each mic
loc where
A2=zeros(length(peaksTable),2); A2cnt=0;             %col 1 = dist from blast point and
col 2 =
A3=zeros(length(peaksTable),2); A3cnt=0;             %SPL in dB
A4=zeros(length(peaksTable),2); A4cnt=0;
L1=zeros(length(peaksTable),2); L1cnt=0;             %The *cnt variable is used later in
plotting
L2=zeros(length(peaksTable),2); L2cnt=0;             %to keep from running into errors
L3=zeros(length(peaksTable),2); L3cnt=0;
S1=zeros(length(peaksTable),2); S1cnt=0;
S2=zeros(length(peaksTable),2); S2cnt=0;
S3=zeros(length(peaksTable),2); S3cnt=0;
S4=zeros(length(peaksTable),2); S4cnt=0;
S5=zeros(length(peaksTable),2); S5cnt=0;


for i=1:length(peaksTable)

%Goes through table and finds all rows that match the desired source type,
%source location, and either a particular config or all configs. The dist
%from the source and the SPL are stored in the corresponding matrix for
%that mic.

%peaksTable is set up with columns:
%  1=Source type    2=Source Loc    3=Array Config  4=Mic Loc   5=dist from source
%  6=SPL in Pa     7=SPL in dB

    if (peaksTable(i,1)==srcin && peaksTable(i,2)==srcloc && (con-
fig==peaksTable(i,3)||config==4))
        if (1<peaksTable(i,4) && peaksTable(i,4)<=4)
            A1cnt=A1cnt+1;
            A1(A1cnt,1)=peaksTable(i,5); A1(A1cnt,2)=peaksTable(i,7);
        elseif (4<peaksTable(i,4) && peaksTable(i,4)<=8)
            A2cnt=A2cnt+1;
            A2(A2cnt,1)=peaksTable(i,5); A2(A2cnt,2)=peaksTable(i,7);
        elseif (8<peaksTable(i,4) && peaksTable(i,4)<=12)
            A3cnt=A3cnt+1;
            A3(A3cnt,1)=peaksTable(i,5); A3(A3cnt,2)=peaksTable(i,7);
        elseif (12<peaksTable(i,4) && peaksTable(i,4)<=16)
            A4cnt=A4cnt+1;
            A4(A4cnt,1)=peaksTable(i,5); A4(A4cnt,2)=peaksTable(i,7);
        elseif (peaksTable(i,4)==17)
            L1cnt=L1cnt+1;
            L1(L1cnt,1)=peaksTable(i,5); L1(L1cnt,2)=peaksTable(i,7);
        elseif (peaksTable(i,4)==18)
            L2cnt=L2cnt+1;
            L2(L2cnt,1)=peaksTable(i,5); L2(L2cnt,2)=peaksTable(i,7);
        elseif (peaksTable(i,4)==19)
            L3cnt=L3cnt+1;
            L3(L3cnt,1)=peaksTable(i,5); L3(L3cnt,2)=peaksTable(i,7);
        elseif (peaksTable(i,4)==20)
            S1cnt=S1cnt+1;
            S1(S1cnt,1)=peaksTable(i,5); S1(S1cnt,2)=peaksTable(i,7);
        elseif (peaksTable(i,4)==21)
            S2cnt=S2cnt+1;
            S2(S2cnt,1)=peaksTable(i,5); S2(S2cnt,2)=peaksTable(i,7);
        elseif (peaksTable(i,4)==22)
            S3cnt=S3cnt+1;
            S3(S3cnt,1)=peaksTable(i,5); S3(S3cnt,2)=peaksTable(i,7);
        elseif (peaksTable(i,4)==23)
            S4cnt=S4cnt+1;
            S4(S4cnt,1)=peaksTable(i,5); S4(S4cnt,2)=peaksTable(i,7);
        elseif (peaksTable(i,4)==24)
            S5cnt=S5cnt+1;
            S5(S5cnt,1)=peaksTable(i,5); S5(S5cnt,2)=peaksTable(i,7);
        end
   end
end

%First subplot shows level vs actual distance from the blast point
subplot(2,1,1)
plot(A1(1:A1cnt,1),A1(1:A1cnt,2),'bo')
hold on
plot(A2(1:A2cnt,1),A2(1:A2cnt,2),'bs')
plot(A3(1:A3cnt,1),A3(1:A3cnt,2),'bd')
plot(A4(1:A4cnt,1),A4(1:A4cnt,2),'bp')
plot(L1(1:L1cnt,1),L1(1:L1cnt,2),'mo')
plot(L2(1:L2cnt,1),L2(1:L2cnt,2),'ms')
plot(L3(1:L3cnt,1),L3(1:L3cnt,2),'md')
plot(S1(1:S1cnt,1),S1(1:S1cnt,2),'go')
plot(S2(1:S2cnt,1),S2(1:S2cnt,2),'gs')
plot(S3(1:S3cnt,1),S3(1:S3cnt,2),'gd')
plot(S4(1:S4cnt,1),S4(1:S4cnt,2),'gp')
plot(S5(1:S5cnt,1),S5(1:S5cnt,2),'g+')

grid on
legend('A1','A2','A3','A4','L1','L2','L3','S1','S2','S3','S4','S5',...
    'Location','NorthEastOutside')
xlabel('Distance from Source (meters)')
ylabel('SPL (dB)')
if srcin==1
    source='C-4';
elseif srcin==2
    source='PC';
end
if config==1
    arrayConfig='Parallel';
elseif config==2
    arrayConfig='Perpendicular';
elseif config==3
    arrayConfig='Vertical';
elseif config==4
    arrayConfig='All';
end
title([source ' - BP' num2str(srcloc) ' : ' arrayConfig ' Config'])
hold off

%Second subplot shows a horizontal view where the mic locations are
%stationary and the blast point moves along the horizontal axis

BP1xy=[0 0;0 200];  %Define x and y coordinates for the blast points and the forest edges
BP2xy=[0 0;0 200];
BP3xy=[175 0;175 200];
BP4xy=[313 0; 313 200];
edge1=[50 0; 50 200];
edge2=[263 0; 263 200];

%Define x and y coordinates for mic locs along main prop line
A1xy=[0 25];
```

```
A2xy=[0 50];
A3xy=[0 75];
A4xy=[0 100];

L1xy=[0 150];
L2xy=[0 263];
L3xy=[0 288];

subplot(2,1,2)

plot(edge1(:,1),edge1(:,2),'k--');
hold on
plot(edge2(:,1),edge2(:,2),'k--');
if srcloc==1
    plot(BP1xy(:,1),BP1xy(:,2),'r-.');
elseif srcloc==2
    plot(BP2xy(:,1),BP2xy(:,2),'r-.');
elseif srcloc==3
    plot(BP3xy(:,1),BP3xy(:,2),'r-.');
elseif srcloc==4
    plot(BP4xy(:,1),BP4xy(:,2),'r-.');
end

legend ('Edge','Edge','BP',...
            'Location','NorthEastOutside');
stem(A1xy(1,2)*ones(A1cnt,1),A1(1:A1cnt,2),'ob');
stem(A2xy(1,2)*ones(A2cnt,1),A2(1:A2cnt,2),'sb');
stem(A3xy(1,2)*ones(A3cnt,1),A3(1:A3cnt,2),'db');
stem(A4xy(1,2)*ones(A4cnt,1),A4(1:A4cnt,2),'pb');
stem(L1xy(1,2)*ones(L1cnt,1),L1(1:L1cnt,2),'om');
stem(L2xy(1,2)*ones(L2cnt,1),L2(1:L2cnt,2),'sm');
stem(L3xy(1,2)*ones(L3cnt,1),L3(1:L3cnt,2),'dm');

hold off
xlabel('Horizontal Distance (meters)');ylabel('SPL (dB)');
```

```
function peaksOut=getPeaks(rootdir)
%*********************************************************************
%Input:     directory for one day's sample set
%Output:    matrix indicating shot date, shot time, channel number, and peak levels
%for each waveform from that day
%Description: For use in the Peak level vs Distance Analysis:
%Takes as input the directory for one day's samples, and returns a matrix
%indicating the peak levels for each waveform from that day.
%
%Requires:  eventWaveform.mat,eventHeader.mat,sensDat.mat
%
%Written for experiments at Camp Ripley, June 2006
%by Tim Onder (Timothy.E.Onder@erdc.usace.army.mil)
%********************************************************************

cd(rootdir)
D=dir(rootdir); numfiles=length(D);
count=0; peaks=cell((numfiles-2)*16, 4);

for j = 1:numfiles
    if(D(j,1).isdir==1 && strcmp(D(j,1).name,'.')==0 && strcmp(D(j,1).name,'PRECAL')==0
&& strcmp(D(j,1).name,'POSTCAL')==0)

        cd(D(j,1).name)

        for i = 1:16            %Scrolling through all channels
            channel=['CH' num2str(i)];

            if exist(channel,'dir')~=7  %Checks if there's data for the channel
                continue
            end

            count=count+1;
            cd (channel)                %Opens folder for each channel
            load eventWaveform;         %and loads waveform in Volts/Sample
            load eventHeader;           %as well as samplingRate, shotDate, and time

            cd (['../../PRECAL/' channel])
            load sensDat;               %Gets sensitivity data from calibration
            calwaveform = waveform/sensDat;                         %Converts waveform to
Pascals

            peaks{count, 1} = shotDate;
            peaks{count, 2} = time;
            peaks{count, 3} = traceName;
            peaks{count, 4} = max(abs(calwaveform));

            cd(['../../' D(j,1).name])
        end
    end
    cd (rootdir)

end

peaksOut=peaks(1:count , 1:4);
```

```
function WaveandSpectra (rootdir, chanNum)
%**************************************************************************
%Input:  directory for one shot, channel number of desired mic
%Output:  none
%Description:  This function will take as input the directory for a single shot and the
%channel whose plot is desired. It will then plot the calibrated waveform in
%Pascals and the 1/3 octave band spectra. Default beginning and end times for the FFT are
0.7s
%and 3.7s respectively. It will also indicate the peak level of the waveform.
%
%Requires:  eventHeader.mat, eventWaveform.mat, sensDat, MicLocations.mat
%Calls:  SELtest2.m, semilogxBar.m
%
%Written for experiments at Camp Ripley, June 2006
%by Tim Onder (Timothy.E.Onder@erdc.usace.army.mil)
%************************************************************************
channel = ['CH' num2str(chanNum)];
cd (channel)
load eventHeader.mat;                                   %Load samplingRate, shotDate,
time
load eventWaveform.mat;                                 %and the uncalibrated waveform

cd (['../../PRECAL/' channel]);
load sensDat.mat;                                       %Load sensitivity data

timeaxis=[1:(length(waveform))]'/samplingRate;          %Sets up the X-axis to show time
(sec)
calwaveform = waveform/sensDat;                         %Converts waveform to Pascals
peakLevel = max(abs(calwaveform));                      %Calculates peak level (pos or
neg)

cd(['../../..']);                                       %loads mic locations for either
the
load MicLocations.mat;                                  %forest scope or the trailer for
if length(findstr(rootdir, 'Forest'))~= 0               %display in legend
    loc = ForestMicLocations{chanNum,1};
end
if length(findstr(rootdir, 'Trailer'))~= 0
    loc = TrailerMicLocations{chanNum,1};
end

subplot(2,1,1), plot(timeaxis,calwaveform)              %and plots calibrated waveform
xlabel('Time (s)');
ylabel('Pressure (Pa)');
title(['Sample from ' shotDate ' at ' time '.   Peak Level = ' num2str(peakLevel) '
Pa.']);
legend(loc,'Location','NorthEastOutside');

t1=0.7;         %define start and stop times for FFT
t2=3.7;
[tobSEL, Fcenter, totalSEL] = SELtest2(rootdir, chanNum, t1, t2);

subplot(2,1,2), semilogxBar(Fcenter, tobSEL);           %Generate and plot the sound ex-
posure %level
xlabel('Frequency (Hz)');
ylabel('SEL (dB)');
axis([sqrt(10^(-2/5)), sqrt(10^((length(Fcenter)+1)/5)), 10*floor(.1*min(tobSEL)),
10*ceil(max(tobSEL)*.1)])

cd (rootdir)
```

```
function [tobSEL,Fcenter,totalSEL]=SELtest2(rootdir, chanNum,t1,t2)
%*********************************************************************
%Input: directory for one shot, desired channel, start time, stop time
%Output: array of the 1/3 octave band SEL levels, center freqs of bands
%Description: Calculates and returns the 1/3 octave band SEL for one shot
%waveform. The user defines the beginning and ending times to be considered
%for this calculation, where time is from 0 to 5 sec.
%
%Requires: eventWaveform.mat, eventHeader.mat, sensDat.mat
%Calls: generateBands.m, semilogxBar.m
%
%Written for experiments at Camp Ripley, June 2006
%by Tim Onder (Timothy.E.Onder@erdc.usace.army.mil)
%**********************************************************************
cd (rootdir)
channel = ['CH' num2str(chanNum)];

%If there is no data for that channel, return zeros for all outputs
if exist(channel,'dir')==0
    tobSEL=zeros(1,47);Fcenter=zeros(1,47);totalSEL=0;
    return
end

cd (channel)
load eventHeader.mat;                                  %Load samplingRate, shotDate,
time
load eventWaveform.mat;                                %and the uncalibrated waveform

cd (['../../PRECAL/' channel]);
load sensDat.mat;                                      %Load sensitivity data
cd (rootdir)

if t1>t2
    temp=t1;                                           %Makes sure t1<t2
    t1=t2;
    t2=temp;
end

calwaveform = waveform/sensDat;                        %Converts waveform to Pascals

%signal will be the segment of the wave we want to evaluate
signal=calwaveform(round(t1*samplingRate)+1:round(t2*samplingRate),1);

p0 = 20e-6;                                             %Define reference pressure


%**********************************************************************
%Get 1/3 Oct Band SEL for signal
%**********************************************************************
N = 2^nextpow2(samplingRate*(t2-t1));  %Find the size of the FFT necessary for t2-t1 sec-
onds of   %data
freqRes = samplingRate/N;          %The resolution of frequencies in the frequency domain

lowRelFreq  = floor(samplingRate/N);    %Lowest reliable frequency in the FFT

%Generate bands for the SEL using generateBands.m
[Flo,Fcenter,Fhi] = generateBands(lowRelFreq,samplingRate);

%Calculate FFT
H = fft (signal,N);

%Compute Narrow Band Sound Exposure
exposure = abs(H).^2/(samplingRate*N);
totalSEL = 10*log10(2*sum(exposure)/p0^2);
%Calculate 1/3-Octave Band Sound Exposure
indexLow = round(Flo(1)/freqRes);   %The first frequencies added up will be
                                    %starting at the lowest Flo
if round(Fhi(1)/freqRes) > 1                %Must take care of the first band calcula-
tions
    indexHigh = round(Fhi(1)/freqRes)-1;    %Separately here, to avoid indexing by 0
    tobShot(1+lowRelFreq) = 2*sum(exposure(indexLow:indexHigh));
    indexLow = indexHigh + 1;
end;
```

```
for k=2:length(Flo)                     % For k = 2:# of Bands, for the rest of the bands
    indexHigh = round(Fhi(k)/freqRes)-1;
    if indexHigh >= indexLow
        if round(Fhi(k)/freqRes) > round(Fhi(k-1)/freqRes)
            tobShot(k+lowRelFreq) = 2*sum(exposure(indexLow:indexHigh)); %T.O.B. is one-
Third
                                                                          %Octave Band
sound exposure
            indexLow = indexHigh + 1;
        end;
    end;
end;

%Compute 1/3-Octave Band SEL
tobSEL = 10*log10(tobShot/p0^2); %1/3 octave band
                                  %sound exposure level = 10 log (E_i/P_o^2)
```

```
 function [Fcenter,totalSEL,tobSEL]=SELavg2(srcType,srcLoc,scope,channel)
%**********************************************************************
%Input: srcType= 'C-4' or 'PC'
%       srcLoc= 1,2,3,4
%       scope= 'Trailer' or 'Forest' (must be entered exactly like these)
%       channel= 1 thru 16
%Output:  Fcenter= 47 element vector with center freqs for the 1/3 oct bands
%         totalSEL= total SEL for each shot
%         tobSEL= each row represents one shot, where the cols are the SEL
%           for each band corresponding to the same col in Fcenter.
%Description:  Loads DataLog.mat and filters to gather all shots matching
%           the inputs. Then runs each shot through SELtest2.m to get
%           tob SEL values. Levels for each shot are compiled into tobSEL.
%           Default values for FFT start and stop times are 0.7sec &
%           3.7sec. Average SEL level across each shot are also computed
%           and can be returned or plotted.
%
%Requires: DataLog.mat
%Calls:    SELtest2.m, semilogxBar.m
%
%Written for experiments at Camp Ripley, June 2006
%by Tim Onder (Timothy.E.Onder@erdc.usace.army.mil)
%*********************************************************************
cd ('N:\Research\Acoustics\Camp Ripley Matlab Format');
load DataLog.mat;
count=0;
x=[];
%DataLog contains information for every shot. Its columns are:
%   1)Scope     2)Yok file number 3)Date    4)Time
%   5)Source type  6)Source location   7)Array config
%   8)file location(within N:\Research\Acoustics\Camp Ripley Matlab Format)

%Shots in DataLog that match the desired inputs are collected into array x
%which will have length of count
for i=1:length(DataLog)
    if strcmp(DataLog{i,6},['BP' num2str(srcLoc)])==1 ...
            && strcmp(DataLog{i,5}, srcType)==1 ...
            && strcmp(DataLog{i,1}, scope)==1
        count=count+1;
        for z=1:8
            x{count,z}=DataLog{i,z};
        end
    end
end

if isempty(x)==1
    disp('There are no waveforms that match the input values.')
    tobSEL=zeros(1,47);Fcenter=zeros(1,47);totalSEL=0;
    return
end

t1=0.7;             %Define start and stop times for the FFT calculation
t2=3.7;

%Fcenter contains the center frequencies for the 1/3 octave bands
%For each shot in x, there will be a row in tobSEL that lists the amplitude
%for each corresponding frequency band.
%The total SEL for each shot will be collected in array totalSEL
for k=1:count
   [tobSEL(k,:),Fcenter,totalSEL(k)]=SELtest2(['N:\Research\Acoustics\Camp Ripley Matlab
Format' x{k,8}], ...
   channel, t1,t2);
end
%To find average SEL across all the shots, all the columns in tobSEL are summed
%and divided by the number of shots
for a=1:length(tobSEL')
    tobSELavg(a)=sum(tobSEL(:,a))/count;
end

totalSELavg=sum(totalSEL)/length(totalSEL);
semilogxBar(Fcenter, tobSELavg)
```

```
function [results]=collectSEL2
%*************************************************************************
%Input: None (user defines srctype, srcloc, and scope within m-file:
%       line 23-26)
%Output:  Matrix containing source type, source location, mic location,
%       total SEL (in dB), and 1/3 octave band SEL levels for the center frequencies
%       given in row 1 cols 5+
%Description:  For the user defined values, the function runs SELavg2 to
%       get the tob SEL for every channel of that Yokogawa. It assembles
%       all the tob SELs for a particular source type and location into the
%       output matrix, "results". This function was used multiple times to make
%       tobSEL_all.mat
%
%Requires:  Fcenter.mat, MicLocations.mat
%Calls: SELavg2.m
%
%Written for experiments at Camp Ripley, June 2006
%by Tim Onder (Timothy.E.Onder@erdc.usace.army.mil)
%*************************************************************************

srcType= {'C-4' 'PC'};
srcLoc=(1:4);
scope={'Trailer' 'Forest'};

cd ('N:\Research\Acoustics\Camp Ripley Matlab Format\Spectrum Analysis');
load Fcenter.mat;
cd ..
load MicLocations.mat;

results{1,1}='SrcType'; results{1,2}='SrcLoc'; results{1,3}='MicLoc';
results{1,4}='TotalSEL';
for i=1:47
    results{1,i+4}=Fcenter(i);
end

count=2;

for i=1                  %srctype
    for j=1               %srcloc
        for k=2           %scope
            if strcmp(scope{k},'Trailer')==1

                for m=1:16
                   [Freq,totalSEL,bands]=SELavg2(srcType(i),srcLoc(j),scope(k),m);

                    for p=1:length(totalSEL)
                        results{count+p-1,1}=srcType{i};
                        results{count+p-1,2}=srcLoc(j);
                        results{count+p-1,3}=TrailerMicLocations{m};
                        results{count+p-1,4}=totalSEL(p);
                        for n=1:47
                            results{count+p-1,n+4}=bands(p,n);
                        end
                    end

                    results{count,3}
                    count=count+length(totalSEL);
                end


            elseif strcmp(scope{k},'Forest')==1
                for m=1:8
                    [Freq,totalSEL,bands]=SELavg2(srcType(i),srcLoc(j),scope(k),m);

                    for p=1:length(totalSEL)
                        results{count+p-1,1}=srcType{i};
                        results{count+p-1,2}=srcLoc(j);
                        results{count+p-1,3}=ForestMicLocations{m};
                        results{count+p-1,4}=totalSEL(p);
                        for n=1:47
                            results{count+p-1,n+4}=bands(p,n);
                        end
                    end
                    results{count,3}
                    count=count+length(totalSEL);
                end

            end
        end
    end
end
```

```
function [results]=SELperShot(srcstr,srcloc,config)
%**********************************************************************
%INPUT:     srcstr = 1=C-4, 2=PC
%           srcloc = 1 thru 4
%           config = 1=Parallel, 2=Perpendicular, 3=Vertical, 4=all
%OUTPUT:    "results" matrix containing the total SEL for all shots matching
%the desired input values. Columns are Mic Location, total SEL, and shot #,
%respectively.
%DESCRIPTION:  Filters through the tobSEL_all matrix and collects all the
%shots matching the desired source type, source location, and array config.
%These are plotted to show the total SEL for each mic location per shot.
%
%Requires: tobSEL_all.mat
%
%Written for experiments at Camp Ripley, June 2006
%by Tim Onder (Timothy.E.Onder@erdc.usace.army.mil)
%*********************************************************************

load ('N:\Research\Acoustics\Camp Ripley Matlab Format\Spectrum Analy-
sis\tobSEL_all.mat');

if srcstr==1                            %must change the input integers to
    srctype='C-4';                       %match the values in the cells of tobSEL_all
elseif srcstr==2                        %in order to scan it later
    srctype='PC';
end

if config==1
    array='Parallel';
elseif config==2
    array='Perpendicular';
elseif config==3
    array='Vertical';
end

rescnt=0;           %rescnt will be the total number of matches

for i=2:length(tobSEL_all)
    if (srcloc==tobSEL_all{i,2} && strcmp(srctype,tobSEL_all{i,1})==1 && ...
            (strcmp(array,tobSEL_all{i,3})==1||config==4))
        rescnt=rescnt+1;
        results{rescnt,1}=tobSEL_all{i,5};   %mic loc to column1
        results{rescnt,2}=tobSEL_all{i,6};   %totalSEL to column2
        results{rescnt,3}=tobSEL_all{i,4};   %shotnum to column3
    end
end

figure
hold on
hand=[];

for k=1:length(results)
    if (strcmp(results(k,1),'A1a')==1||strcmp(results(k,1),'A1b')==1 || ...
            strcmp(results(k,1),'A1c')==1|| strcmp(results(k,1),'A1d')==1)          %A1a-
A1d
        hand(1)=plot(results{k,3},results{k,2},'bo');   %hand vector will be used for
legend

    elseif (strcmp(results(k,1),'A2a')==1||strcmp(results(k,1),'A2b')==1 || ...
            strcmp(results(k,1),'A2c')==1|| strcmp(results(k,1),'A2d')==1)      %A2a-A2d
       hand(2)=plot(results{k,3},results{k,2},'bs');

    elseif (strcmp(results(k,1),'A3a')==1||strcmp(results(k,1),'A3b')==1 || ...
            strcmp(results(k,1),'A3c')==1|| strcmp(results(k,1),'A3d')==1)     %A3a-A3d
        hand(3)=plot(results{k,3},results{k,2},'bd');

    elseif (strcmp(results(k,1),'A4a')==1||strcmp(results(k,1),'A4b')==1 || ...
            strcmp(results(k,1),'A4c')==1|| strcmp(results(k,1),'A4d')==1)    %A4a-A4d
        hand(4)=plot(results{k,3},results{k,2},'bp');

    elseif (strcmp(results(k,1),'L1')==1)                        %L1
        hand(5)=plot(results{k,3},results{k,2},'mo');
    elseif (strcmp(results(k,1),'L2')==1)                        %L2
        hand(6)=plot(results{k,3},results{k,2},'ms');

    elseif (strcmp(results(k,1),'L3')==1)                        %L3
        hand(7)=plot(results{k,3},results{k,2},'md');

    elseif (strcmp(results(k,1),'S1')==1)                        %S1
        hand(8)=plot(results{k,3},results{k,2},'go');

    elseif (strcmp(results(k,1),'S2')==1)                        %S2
        hand(9)=plot(results{k,3},results{k,2},'gs');

    elseif (strcmp(results(k,1),'S3')==1)                        %S3
        hand(10)=plot(results{k,3},results{k,2},'gd');

    elseif (strcmp(results(k,1),'S4')==1)                        %S4
        hand(11)=plot(results{k,3},results{k,2},'gp');

    elseif (strcmp(results(k,1),'S5')==1)                        %S5
        hand(12)=plot(results{k,3},results{k,2},'g+');
    end
end


hold off
xlabel('Shot Number')
ylabel('Total SEL (dB)')
set(gca,'XTick',1:rescnt)
title([srctype ' - BP' num2str(srcloc) ' : ' array ' Config'])
leg-
end(hand,'A1','A2','A3','A4','L1','L2','L3','S1','S2','S3','S4','S5','Location','NorthEas
tOutside')
```

```
function [results]= PeaksPerShot(srcin,srcloc,config)
%**********************************************************************
%INPUT:     srcstr = 1=C-4, 2=PC
%           srcloc = 1 thru 4
%           config = 1=Parallel, 2=Perpendicular, 3=Vertical, 4=all
%OUTPUT:    "results" matrix containing the peak level for all shots matching
%the desired input values. Columns are Mic Location, peak SPL, and shot #,
%respectively.
%DESCRIPTION:  Filters through the peaksTable matrix and collects all the
%shots matching the desired source type, source location, and array config.
%These are plotted to show the peak level for each mic location per shot.
%
%Requires: peaksTable.mat
%
%Written for experiments at Camp Ripley, June 2006
%by Tim Onder (Timothy.E.Onder@erdc.usace.army.mil)
%*********************************************************************

load ('N:\Research\Acoustics\Camp Ripley Matlab Format\Pk vs Dist Analy-
sis\peaksTable.mat');

rescnt=0;

for i=1:length(peaksTable)

    if(peaksTable(i,1)==srcin && peaksTable(i,2)==srcloc && (con-
fig==peaksTable(i,3)||config==4))
        rescnt=rescnt+1;
        results(rescnt,1)=peaksTable(i,4);%mic loc to results col1
        results(rescnt,2)=peaksTable(i,7);%peak(dB) to results col2
        results(rescnt,3)=peaksTable(i,8);%shotnum to results col3
    end
end

figure
hold on
hand=[];
for k=1:length(results)
    if (results(k,1)>=1 && results(k,1)<5)          %A1a-A1d
        hand(1)=plot(results(k,3),results(k,2),'bo');

    elseif (results(k,1)>=5 && results(k,1)<9)      %A2a-A2d
        hand(2)=plot(results(k,3),results(k,2),'bs');

    elseif (results(k,1)>=9 && results(k,1)<13)     %A3a-A3d
        hand(3)=plot(results(k,3),results(k,2),'bd');

    elseif (results(k,1)>=13 && results(k,1)<17)    %A4a-A4d
        hand(4)=plot(results(k,3),results(k,2),'bp');

    elseif results(k,1)==17                         %L1
        hand(5)=plot(results(k,3),results(k,2),'mo');

    elseif results(k,1)==18                         %L2
        hand(6)=plot(results(k,3),results(k,2),'ms');

    elseif results(k,1)==19                         %L3
        hand(7)=plot(results(k,3),results(k,2),'md');

    elseif results(k,1)==20                         %S1
        hand(8)=plot(results(k,3),results(k,2),'go');

    elseif results(k,1)==21                         %S2
        hand(9)=plot(results(k,3),results(k,2),'gs');

    elseif results(k,1)==22                         %S3
        hand(10)=plot(results(k,3),results(k,2),'gd');

    elseif results(k,1)==23                         %S4
        hand(11)=plot(results(k,3),results(k,2),'gp');

    elseif results(k,1)==24                         %S5
        hand(12)=plot(results(k,3),results(k,2),'g+');
    end
end


hold off
xlabel('Shot Number')
ylabel('Peak SPL (dB)')
if srcin==1
    source='C-4';
elseif srcin==2
    source='PC';
end
if config==1
    arrayConfig='Parallel';
elseif config==2
    arrayConfig='Perpendicular';
elseif config==3
    arrayConfig='Vertical';
elseif config==4
    arrayConfig='All';
end
set(gca,'XTick',1:rescnt)
title([source ' - BP' num2str(srcloc) ' : ' arrayConfig ' Config'])
leg-
end(hand,'A1','A2','A3','A4','L1','L2','L3','S1','S2','S3','S4','S5','Location','NorthEas
tOutside')
```

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE (DD-MM-YYYY) 07-2007	**2. REPORT TYPE** Final	**3. DATES COVERED (From - To)**

4. TITLE AND SUBTITLE
Acoustic Propagation Through a Forest Edge:
Data Report for Camp Ripley, Minnesota

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Michelle E. Swearingen, Michael J. White, Patrick J. Guertin, Jeffrey A. Mifflin, Timothy E. Onder, Donald G. Albert, Stephen N. Decato, and Arnold Tunick

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER
31D7FK

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
U.S. Army Engineer Research and Development Center (ERDC)
Construction Engineering Research Laboratory (CERL)
PO Box 9005
Champaign, IL 61826-9005

8. PERFORMING ORGANIZATION REPORT NUMBER
ERDC SR-07-3

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
U.S. Army Corps of Engineers
441 G Street NW
Washington, DC 20314-1000

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

Acoustic propagation and diffraction of high-amplitude, short duration, signals through a forest edge has implications for noise mitigation and battlefield acoustic sensors. While the acoustic significance of this unique environment has been noted in the past, it has not been studied in any detail. Acoustic signals that have propagated through a forest edge yield complicated pressure time histories for receivers both within and outside the forest. Several physical processes contribute to this complexity, including the physical structures of the biomass and ground and the microclimate. A deep understanding of acoustic propagation through this unique environment may lead to strategic placement of fire breaks for noise mitigation and improved signal processing algorithms for use with acoustic detection, direction-finding, and range finding sensors. Because of the broad scope of issues that could be addressed once acoustic propagation and diffraction at a forest edge is understood, it is important to study this unique environment in detail. This report provides documentation of a field experiment conducted as part of a study of the acoustic properties of the forest edge environment.

15. SUBJECT TERMS

Camp Ripley, MN	military training	noise barriers
sound propagation		

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON**
a. REPORT Unclassified	**b. ABSTRACT** Unclassified	**c. THIS PAGE** Unclassified	SAR	66	**19b. TELEPHONE NUMBER (include area code)**