Waymark Based Underwater Acoustic Channel Model - MATLAB code description

I. Introduction

THESE notes describe the Matlab code for the Waymark based underwater acoustic propagation model [1][2]. Figs. 5 and 6 show a block diagram of the channel model. In Fig. 5, an ADC and DAC are shown only to emphasise that the channel model is for signal transmission, the Waymark simulation has $s(nT_s)$ as an input, and $y(nT_s)$ as an output. Fig. 6 shows internal variables in the Matlab code with reference to the model in Fig. 5.

II. prerequisites

The waymark model uses the Acoustic Toolbox from Heat, Light, and Sound Research, Inc. [3] for the underwater environment modelling. Therefore, this needs to be installed before proceeding further with the Waymark model.

The current version of the channel model was tested using Matlab R2016a, with Windows 7.

III. Main code loop

The function `waymarkModelMain.m` is the main section of code where the calculation and construction of the time varying channel takes place. The signal is downshifted to the baseband, passed through the channel and upshifted back to the passband. The code is designed to process the channel model and signal on a rolling, per waymark basis; this is designed to reduce the memory requirements for very long simulations. It means that the sections of code described in this note are iterated through for each waymark. The function `waymarkModelMain.m` is evoked with parameters from an experiment script `runExperimentLocal.m`, and it is this script that should be executed to start the example simulation.

A. Trajectory generation

[`waymarkTrajGenBellhopFunc200.m` (example file)]
In order to calculate the acoustic pressure field, the input to any wave propagation modeller would be the basic environment and the trajectory of the transmitter or the receiver within that environment. The trajectory generation specifies the depth of the transmitter, the depth of the receiver, and the horizontal range between them. This information is generated at every waymark time point along the trajectory. Additional waymarks are added before and after the time interval of interest for the channel filter and the spline approximation. The trajectory path can be generated analytically or can be interpolated from a table; because of this, each trajectory is defined in it's own function for which the handle is specified for each experiment. Fig. 1, shows a plot of a typical receiver/transmitter trajectory over time. The trajectory is defined with a function, the handle for which is a top level input into `waymarkModelMain.m`.

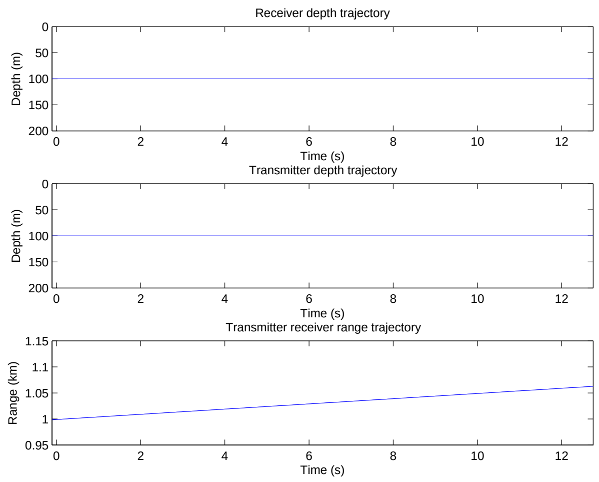

Fig. 1. A plot showing the relative trajectory of the transmitter (source) and receiver. This is in terms of the two depths and the range between them (`waymarkTrajGenBellhopFunc200.m`).

B. Surface generation

[`surfaceGenSine.m` (example file)]
For experiments with a dynamic surface wave, the shape of the wave is defined as a function that generates the total surface wave for the time instant associated with each waymark. The handle for the function is a top level input into `waymarkModelMain.m`.

C. Waymark field generation

[`waymarkFieldGenBellhop.m`]
[`editBellhopEnvFile.m`] This function takes a set of trajectory parameters for each waymark point and modifies the BELLHOP input files [4][5]. BELLHOP is then run and the arrival times and complex valued amplitudes are read in and organised into a structure to be returned to the main code. More details of the setup for the BELLHOP wave propagation modeller can be found in Appendix B.

D. Waymark delays and adjustment

[`waymarkDelayCalc.m` (example file)]
The aim of this section of the code is to find the delays between the waymarks for a time varying channel. Although the waymarks are equally spaced in time along the trajectory, because of the variation in the speed of the sound propagation they are generally not equally spaced relative to the signal. The delay estimation is performed from the first waymark to the latest, building up the composite delay of each waymark. The delays are estimated in the frequency domain. The common propagation delay is removed from the arrival time of all of

the waymarks; this common arrival time being the first arrival at the first waymark, τ_{min}. The frequency response $P_m(k)$ at waymark m is given by:

$$P_m(k) = \sum_{n=0}^{N-1} c_n e^{-j2\pi(\tau_n - \tau_{min}) f_k} \tag{1}$$

where:
$P_m(k)$: frequency response for waymark index m,
c_n: baseband complex amplitude of the multipath arrival,
τ_n: multipath arrival delay,
τ_{min}: minimun delay for all paths,
n: index for the multipath arrival,
k: index $k = 0, \ldots, K-1$ for each frequency component in the search or channel frequency range.
As described in [2] the delay shift is found by shifting the impulse response in time (or multiplying in the frequency domain) and looking for the best correlation. More formally, from [2]:

$$\Delta = \arg\max_{\Theta} J(\Theta) \tag{2}$$

where:

$$J(\Theta) = \left| \sum_{k=0}^{K-1} p_m(\omega_k) p^*_{m-1}(\omega_k) e^{j\omega_k \Theta} \right|^2 \tag{3}$$

where:
Θ: the shift in time,
p_m: frequency response for waymark index m.
This process estimates the delay between waymarks, these delays are accumulated, the sum representing the composite delay from the first waymark. This delay needs to be offset to the waymark.

Fig. 2 shows an example of a waymark impulse response. The response is long enough to encompass all of the multipath components; this is dependent upon the environment that is being modelled.

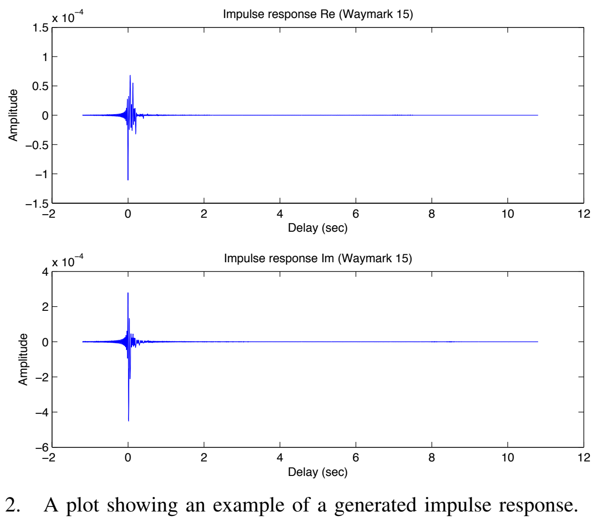

Fig. 2. A plot showing an example of a generated impulse response.

E. Shifting the input signal to the baseband

[`waymarkModelMain.m`]
In this section, the input signal is converted to an analytic signal and decimated. The downshift in the frequency domain is achieved by multiplying the input signal samples by $e^{-j2\pi f_c t}$, therefore:

$$s_e(nT_s) = s(nT_s) e^{-j2\pi f_c n T_s} \tag{4}$$

where:
n: sample index,
$s_e(nT_s)$: frequency-shifted signal,
$s(nT_s)$: input signal,
f_c: carrier frequency,
T_s: sample period.

This signal is then passed through a low pass filter to remove all but the frequencies centred around zero. For this a raised-cosine filter is used where the filter impulse response is given by:

$$r(nT_s) = \text{sinc}(f_0 n T_s) \frac{\cos(\pi f_0 \alpha n T_s)}{1 - (2 f_0 \alpha n T_s)^2} \tag{5}$$

where:
f_0: a cut-off frequency of interest,
α: roll-off factor.

Fig. 3, shows the spectrum of the original signal, the downshifted signal and the filtered and decimated signal. It is this processed signal that is the input to the channel model.

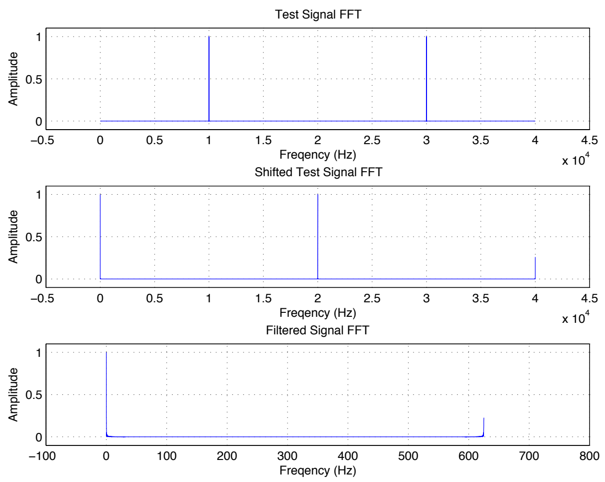

Fig. 3. A plot showing the spectrum of the input signal (f_s = 40kHz), the downshifted signal spectrum (f_s = 40kHz) and spectrum of the filtered and decimated signal (f_d = 625Hz)

F. Channel Model

[`waymarkModelMain.m`]
This section of code creates the channel impulse response at the decimated sampling rate. This is done using a spline approximation between the waymark impulse responses. Based on the estimated waymark delays, the total delay to the signal is calculated for each decimated sample point. These delays

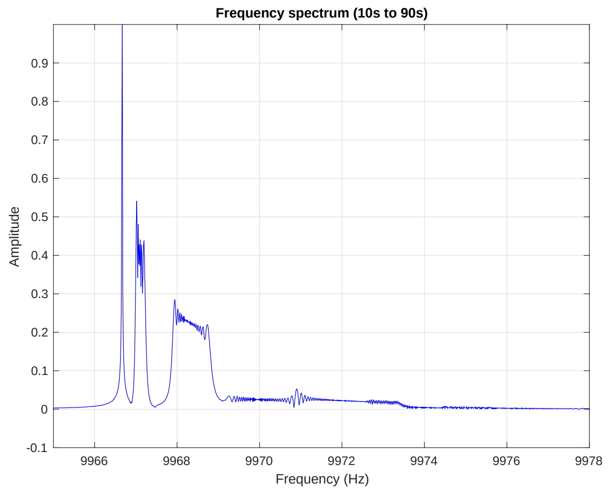

Fig. 4. A plot showing the spectrum of the signal over an example experiment (the transitions at the beginning and end of the signal have been removed).

are then used to interpolate a correctly timed signal for every sample point allowing it to be convolved with the generated channel impulse response to produce an output sample.

G. *Shifting the output signal to the passband*

[`waymarkModelMain.m`]
The output signal from the channel model is upsampled back to the original sampling frequency using a raised-cosine low pass filter with the same parameters as the one used in the downsampling. Thus the baseband signal is restored at the original sampling rate.

The signal is shifted back to the passband by multiplying each sample by $e^{j2\pi f_c t}$, however the time t for each sample nT_s takes into account τ_n:

$$y(nT_s) = \Re\left\{y_e(nT_s)e^{j2\pi f_c(nT_s-\tau_n)}\right\} \tag{6}$$

where:
n: sample index,
$y(nT_s)$: output signal,
$y_e(nT_s)$: low frequency equivalent signal,
f_c: carrier frequency,
T_s: sample period,
τ_n: estimated additional delay,
$\Re\{\cdot\}$: real part.

For the final output wav file the initial delay is restored by zero padding the beginning of the file with the appropriate number of samples.

IV. Example Simulation

[`runExperimentLocal.m`]
A 10kHz signal of duration 10s is passed through the channel whist the range between the transmitter and receiver is increasing by 5m/s; giving a theoretical Doppler shift of 33.33Hz. The transmitter and receiver were both at a depth of 100m and the initial range between them is 1000m. The environment was 200m deep, with the speed of sound at 1500m/s over the whole depth, the bottom was a simple single layer with a single speed of sound of 1600m/s. Fig. 4 shows the spectrum of the received signal. The direct path can be seen with the multipath components having an altered Doppler shift due to the angle of incident of the reflections.

The main experiment setup parameters are as follows: Setup the path to the Acoustic Toolbox [3] and include the sub-folders in the current directory. This path needs to be appropriate for the operating system.

```
path(path,genpath('~/Documents/MATLAB/at'));
path(path,genpath('./'));
```

Specify a experiment number and name; this is useful for batch processing as the results will be put into a folder with this name.

```
EXPERIMENT_NUM = 1;
EXPERIMENT_NAME = ['exp', num2str(EXPERIMENT_NUM),
'Waymark200Flat'];
```

Specify the input waveform filename.

```
SRC_DATA.waveformName = 'cw_10kHz_100s';
SRC_DATA.waveformExt = 'wav';
```

Add some extra information about the input waveform. The `fCarrier` parameter is the frequency shift to move the signal to the baseband, `fSample` and `bandwidth` define the amount the signal may be decimated.

```
SRC_DATA.fCarrier = 10000;
SRC_DATA.fSample = 40000;
SRC_DATA.bandwidth = 2*156.25;
```

Setup the surface generation parameters; `range`, distance of the surface to generate (m); `numRangePts`, the number of sample points over the surface distance; `SURFACE_FUNCTION`, the matlab function that is used to generate the surface heights as a function distance (range).

```
SURFACE_PARAM.range = 10000.0;
SURFACE_PARAM.numRangePts = 512;
SURFACE_FUNCTION = @surfaceGenFlat;
```

The matlab function that is used to describe the relative position of the transmitter and receiver as a function of time.

```
TRAJECTORY_FUNCTION = @waymarkTrajGenBellhopFunc200;
```

The name of the file that is used to setup the environment parameters for the BELLHOP underwater propagation simulator [4].

```
ENVIRONMENT_FILE_NAME = 'iso200';
```

Parameters to define the search space for the delays between waymarks. The maximum speed of sound `c0`, and the maximum relative speed between the transmitter and the receiver `RXTX_VEL_MAX`.

```
MISC.c0 = 1500;
MISC.RXTX_SPEED_MAX = 6;
```

The roll off factor for the low-pass filter:

```
MISC.ROLL_OFF = 0.25;
```

The total simulation time (s). If the simulation time is longer than the input waveform then a zero amplitude signal is input after the waveform has ended.

```
SIMULATION_TIME = 120;
```

The interval between waymarks (s).

```
T_WAYMARK = 0.0512;
```

References

[1] B. Henson, J. Li, Y. V. Zakharov, and C. Liu, "Waymark baseband underwater acoustic propagation model," in *Underwater Communications and Networking (UComms), 2014*. IEEE, 2014, pp. 1–5.

[2] C. Liu, Y. V. Zakharov, and T. Chen, "Doubly Selective Underwater Acoustic Channel Model for a Moving Transmitter/Receiver," *IEEE Transactions on Vehicular Technology*, vol. 61, no. 3, pp. 938–950, 2012.

[3] (2010, Accessed 3, December, 2016) Heat, Light & Sound Research, Inc. website. [Online]. Available: http://oalib.hlsresearch.com/Modes/AcousticsToolbox/

[4] M. B. Porter, "The bellhop manual and user's guide: Preliminary draft," *Heat, Light, and Sound Research, Inc., La Jolla, CA, USA, Tech. Rep*, 2011.

[5] O. C. Rodriguez, "General description of the BELLHOP ray tracing program," *Physics Department Signal Processing Laboratory Faculty of Sciences and the University of the Algarve Tecnologia (Galician), Version*, vol. 1, 2008.

[6] M. B. Porter, "The KRAKEN normal mode program," DTIC Document, Tech. Rep., 1992.

APPENDIX A
UNDERWATER ACOUSTIC UNDERWATER CHANNEL SIMULATOR

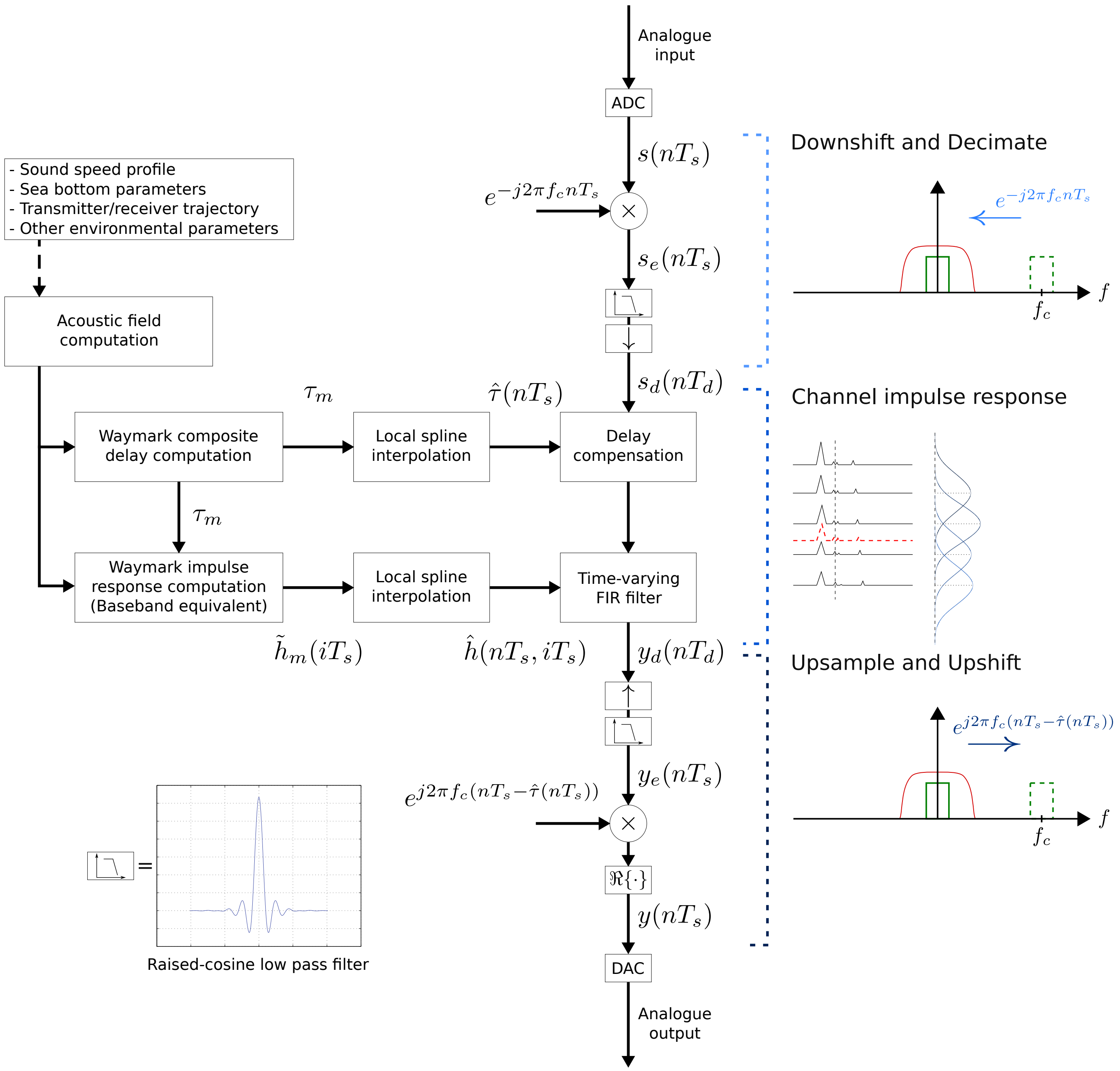

Fig. 5. A block diagram of the underwater acoustic simulator as a development on the system presented in [1].

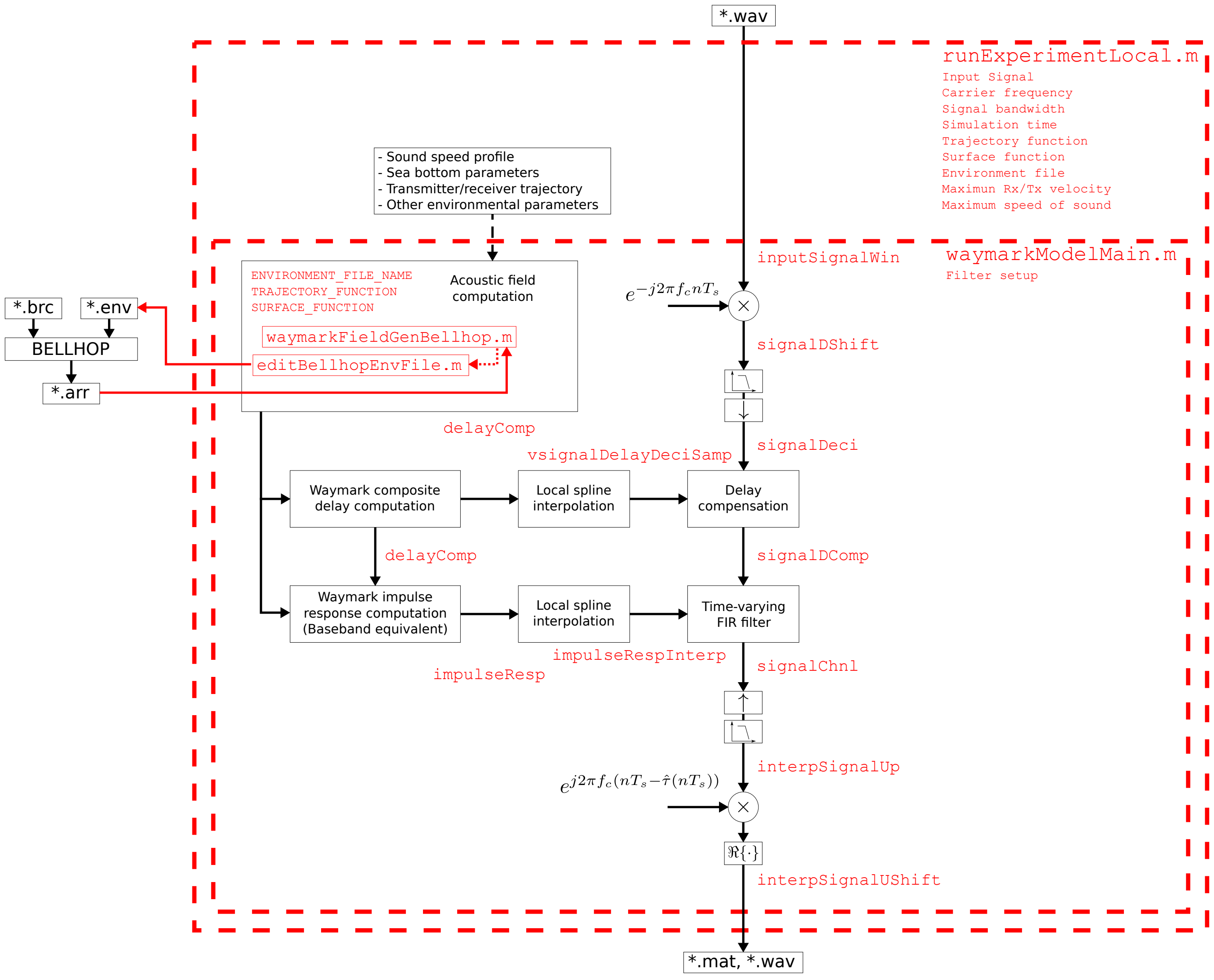

Fig. 6. A block diagram of the underwater acoustic simulator detailing the key variables and the main division in the code.

Appendix B
BELLHOP Setup

The underwater propagation modeller BELLHOP [4] is part of the Acoustic Toolbox created by HLS Research [3]. Listing 1 shows the environment file for the SwellEx-96 Shallow water experiment, the full speed of sound table has been truncated to shorten the listing but it shows the options for BELLHOP and the Speed of Sound Profile (SSP) for the water column. The BELLHOP program is unable to model the sub-bottom profile directly, however it is able to process a set of tabulated complex bottom reflection coefficients. This is not as accurate as a it would be with a full-wave model, however, it does improve the treatment of a layered bottom [4]. The tabulated reflection coefficients are produced by creating an environment file with the sub-bottom profile only, then from the Acoustic Toolbox the BOUNCE program is run with the file as an input. This creates a bottom reflection coefficient file that can then be used as an input the the BELLHOP modeller. Listing 2 shows the option and the SSP for the sub-bottom sediment layers. A brief description of BOUNCE can be found in the report for the KRAKEN normal mode program [6]. The option ’F’ on line 35 of Listing 1 indicates that the bottom reflection coefficient should be read from a file. Another option to note is the ’B’ on line 42, this indicates that Gaussian beam bundles should be modelled. More detail about the options set may be found in the BELLHOP descriptions [4] and [5].

Listing 1. SWellEx experiment setup file example

```
’SwellEx Environment  Bach2geo output for Range = 0.’
10000.000  | Frequency
1          | Number of Layers
’NVW’      | SS interp. option, Upper half opt, atten
0 0.0  220 | 1st layer (ssp)
-10  1521.95    0.0 1.0 0.0 0.0 /
0    1521.95    0.0 1.0 0.0 0.0 /
0.5  1521.95    0.0 1.0 0.0 0.0 /
1    1521.95    0.0 1.0 0.0 0.0 /
1.5  1521.935   0.0 1.0 0.0 0.0 /
...
Truncated table for example listing
...
190  1488.46997 0.0 1.0 0.0 0.0 /
200  1488.30005 0.0 1.0 0.0 0.0 /
220  1488.30005 0.0 1.0 0.0 0.0 /
’F’ 0.0      | Bottom halfspace option, RMS Rough
1            ! NSD
155.99 /     ! SD(1:NSD) (m)
1            ! NRD
 52.00 /     ! RD(1:NRD) (m)
1            ! NR
9.22977 /    ! R(1:NR ) (km)
’AB R’       ! Run-type:’R/C/I/S’
1000         ! NBeams
-85.0 85.0 / ! ALPHA1,2 (degrees)
0.0  220.0  10.0  ! STEP (m), ZBOX (m), RBOX (km)
```

Listing 2. SWellEx experiment setup file example with the sub-bottom reflection coefficients specified

```
’SwellEx Environment  Bach2geo output for Range =   0.’
10000.000      | Frequency
2              | Number of Layers
’NVW’          | SS interp. option, Upper half opt, atten
0 0.0  19      | 2nd layer (sediment)
0   1549.99487 0.0   1.73236012  0.322001904 0.0 /
1   1549.99487 0.0   1.73236012  0.322001904 0.0 /
2   1570.45447 0.0   1.73336411  0.290452182 0.0 /
3   1575.71143 0.0   1.73436713  0.285457075 0.0 /
4   1579.02234 0.0   1.73536932  0.282913357 0.0 /
5   1581.53076 0.0   1.73637068  0.2812877   0.0 /
6   1583.59741 0.0   1.73737097  0.280136436 0.0 /
7   1585.38354 0.0   1.73837054  0.279272139 0.0 /
8   1586.97559 0.0   1.73936915  0.278598756 0.0 /
9   1588.42542 0.0   1.74036694  0.278060824 0.0 /
10  1589.76611 0.0   1.74136376  0.277623594 0.0 /
11  1591.02087 0.0   1.74235976  0.277263939 0.0 /
12  1592.20581 0.0   1.7433548   0.276965588 0.0 /
13  1593.33289 0.0   1.744349    0.276716828 0.0 /
14  1594.41162 0.0   1.74534225  0.276508957 0.0 /
15  1595.44897 0.0   1.74633467  0.276335329 0.0 /
16  1596.45068 0.0   1.74732614  0.276190668 0.0 /
17  1597.42126 0.0   1.74831676  0.276070833 0.0 /
18  1598.36438 0.0   1.74930656  0.275972456 0.0 /
19  1599.28333 0.0   1.7502954   0.275892854 0.0 /
0 0.0  819                   | 3rd layer (mudstone)
19 1881 0.0   2.05999994  0.0599999987 0.0 /
819 3245 0.0   2.05999994  0.0599999987 0.0 /
’A’ 0.0                      | Bottom halfspace option, RMS Rough
819 5200.0  0.0 2.66 0.02 0. | Halfspace acoutic properties
1400.0 1.0E9                 | Lower and upper phase speed cutoff
10.0                         | Maximum range
```