Updated
13 Feb 2025

User Manual

xDDx: numerical toolbox for ultrasound transducer characterization and diagnostics

Table of Contents

INTRODUCTION

Two parts of the toolbox

The functionality of xDDx is divided into two parts: “Holography Toolbox” and “Simulation Toolbox”. The core algorithm is based on the Rayleigh integral implemented in C++ executables for graphics and central processing units (GPUs and CPUs).

The first part, “Holography”, is designed for forward or backward projection of transient acoustic holography measurements to identify transducer surface defects and reveal structural details of the radiated acoustic field. The second part, “Simulation,” utilizes the same projection algorithm as the first part to simulate fields radiated by user-defined transducer designs in a uniform medium without attenuation. Each part is described in a corresponding chapter.

The Holography chapter assumes that users have a transducer of interest, a hydrophone with an acquisition system, and a positioning system to perform and record planar pressure scans with this hydrophone. The Holography part of xDDx uses these recorded scans to project them.

The Simulation chapter, in contrast, does not require any measurements and is intended for simulating fields of computer-modeled transducers. However, it can be combined with the Holography part, as holography-based projections can be used in the Simulation part as boundary conditions.

Installation guide

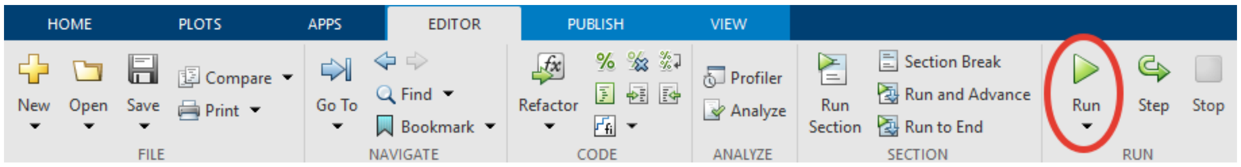
The xDDx toolbox is located in the archive “xDDx.zip”, which is available from the link https://github.com/pavrosni/xDDx/releases

The archive contains a single folder with the same name, “xDDx”. Users can unpack the archive into a folder of their choice and refer to the “xDDx\examples” folder to access the tools.

As the toolbox consists of two parts, the tools for each part are organized into the following folders:
Holography Toolbox (“xDDx\examples\holography_toolbox”)
Simulation Toolbox (“xDDx\examples\simulation_toolbox”)

The xDDx Toolbox has its tools sorted by categories, with certain subfolders present in the corresponding folder. For example:
“xDDx\examples\holography_toolbox\simple_projection_tools”.

The xDDx tools are accessible through scripts e.g. “xDDx\examples\holography_toolbox\quick_start_spherical.m”, and GUI tools are available to assist with some of the interactive steps. To use a tool, a corresponding script (*.m) should be opened and run in MATLAB/Octave environment:



It is good practice to make a local copy of each example before using a tool for a particular case, to keep the original examples unchanged.

How to read this manual

Descriptions of xDDx tools are presented in various examples. For clarity, the examples are divided into three subsections:

1) "General input parameters" – Describes the most important parameters of the tool,
2) "Simulation output" – Provides a brief overview of the simulation results,
3) "Details" – Explains the input and output of the software in greater detail, especially in cases with flexible input/output formats or specific formats for input/output variables.

The text includes portions of MATLAB/Octave code relevant to the topics discussed, presented in a `different font`.

For convenience, a quick explanation of the tools and corresponding variables used in these tools is provided directly in the code as **`%MATLAB/Octave comments%`**. Some users may benefit from reading the comments within a tool script before running it, as the comments summarize the tool's capabilities and might be sufficient to understand its input and output. Note that all examples are available "out of the box" and can be run immediately after unpacking the archive, with no further setup required.

For Holography Toolbox users, it is recommended to read the General Concepts section (page 9) before running the examples. This section provides valuable information about transient holography measurements and establishes relationships between physical measurements and the toolbox parameters.

Users interested in a quick overview of the software's capabilities may start with:

1) The Quick Start example for the Holography Toolbox "xDDx\examples\holography_toolbox\quick_start_spherical.m", and the
2) The Single-Frequency Simulation Tool for the Simulation Toolbox "xDDx\examples\simulation_toolbox\transducer_simulation_sf.m"

These tools provide a general overview of the xDDx interface, its inputs, and outputs.

xDDx Tools

The scripting interface of each tool can be operated using:

- MATLAB/Octave variables for simple scalar parameters that will be described below: numerical, logical, etc. (e.g. `frequencyNominal`).

- MATLAB/Octave data files (*.mat) for complex parameters presented as MATLAB/Octave structures that will be described below (e.g. `Medium`, `HologramTr`, `TransducerSf`, etc.) These structures contain specific fields for different parameters and should be created and saved as MAT-files by the user before using the toolbox. For convenience, the manual pages describing each input structure are listed below:

`Geometry` – page 10
`Medium` – page 11
`HologramTr` – page 13
`HologramSf` – page 16
`TransducerTr` – page 18 or 80
`TransducerSf` – page 19 or 77

An alternative way to define some of the xDDx structures is by using (*.xlsx) templates. The structure of MAT and XLSX files will be discussed in detail in the corresponding sections. MAT and XLSX files used in the example scripts are available in the directory "xDDx\examples\data_for_examples", and empty templates can be found in "xDDx\examples\data_for_examples\xlsx_templates". Note that some cells of the XLSX-templates like text labels and sheet names are protected from changes. Editable cells are highlighted green.

The scripting interface of the tools (M-files) can be accessed in the editable portions of code marked as:

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%EDITABLE CODE%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Changes of the Editable Code do not affect the post-processing part of the scripts and just change input parameters. Nevertheless, users can feel free to modify the rest of the code if deep changes in post processing are required. Note that inappropriate changing this part may result in a tool malfunctioning, so backup is recommended prior editing the tool.

To execute a script, users need to run it script using MATLAB/Octave interface.

The program will run a combination of MATLAB/Octave and C++ executables, and the simulation progress will be displayed in the MATLAB/Octave command line. Some scripts perform several simulation sets, so the percentage counter can appear several times. Note that the percentage counter progress can be non-uniform in time. Once the script is over the message

```
xDDx simulation completed!
```

appears in the MATLAB/Octave Command Window.

All scripts have the default stimulation device flag:

```
simulationDevice = 'cuda'; %simulation device: 'cuda' or 'cpu'
```

which corresponds to PCs with CUDA-compatible GPUs. If your PC does not have a CUDA-compatible GPU, all simulations can be performed using the central processor unit (CPU). In this case the flag should be changed on

```
simulationDevice = 'cpu'; %simulation device: 'cuda' or 'cpu'
```

If you are unsure whether your graphic card supports CUDA, you can try to run the script with the `'cuda'` flag. In the case of failure, you will receive the following MATLAB/Octave error message:

```
Error: The simulation has not been performed! The most likely issue is that your graphics card is not CUDA-compatible. Try running your simulation in CPU mode (simulationDevice = 'cpu'). If you are sure that your GPU is CUDA-compatible, please check that you have the latest version of your graphics drivers.
```

Each tool refers to functions for the xDDx library, so every script includes the lines to add this library in MATLAB/Octave path:

```
libraryDir = '..\..\xDDx_lib'; %lib directory
```

```
addpath(genpath(libraryDir));
```

Note that for the example scripts, the `xDDx_lib` is located one or two directories above the script, which is why the relative directory notation ".." is used. Users can modify the path as necessary if the tool script has been moved from the example folder.

HOLOGRAPHY TOOLBOX

The first part of this toolbox – "Holography Toolbox" – is developed to characterize existing ultrasound transducers in water using the acoustic holography technique. The general idea of this approach is to record the acoustic pressure field at the vertices of a spatial grid (typically 20k – 40k vertices) within a transverse plane in front of the ultrasound source. This 2D record of the acoustic pressure, termed a "hologram", can be numerically projected to either reconstruct the vibrational velocity distribution at the surface of the transducer ("back-projection") or reconstruct the 3D acoustic field in front of the hologram ("forward-projection"). Knowledge of the vibrational velocity pattern at the transducer surface can reveal manufacturing defects manifesting as amplitude and phase nonuniformity, potentially leading to poor focusing or device failure, and 3D HIFU field simulations help to ensure that sufficient pressure levels are achieved in the target region and that no side and grating lobes or hot spots are present outside the target volume.

One of the toolbox's capabilities is performing automatic postprocessing of raw acoustic pressure measurement data and its numerical projections using the Rayleigh integral method. The toolbox includes an automated procedure for correcting geometrical misalignments in the experimental setup to avoid reconstruction errors. To improve the readability of this manual, the theoretical details related to the implementation of this toolbox are not included in the manual. A more detailed description of the methods and algorithms can be found in the paper [Rosnitskiy 2025].

GENERAL CONCEPTS

This section describes the general concepts of the acoustic holography measurement procedure and introduces the toolbox parameters related to them. Please note that all input parameters of the toolbox are expressed in the International System of Units (SI).

Transducer nominal geometry

The software is designed to work with spherically shaped (Fig. 1a) and flat (Fig. 1b) ultrasound transducers. The automatic correction procedures implemented in the toolbox

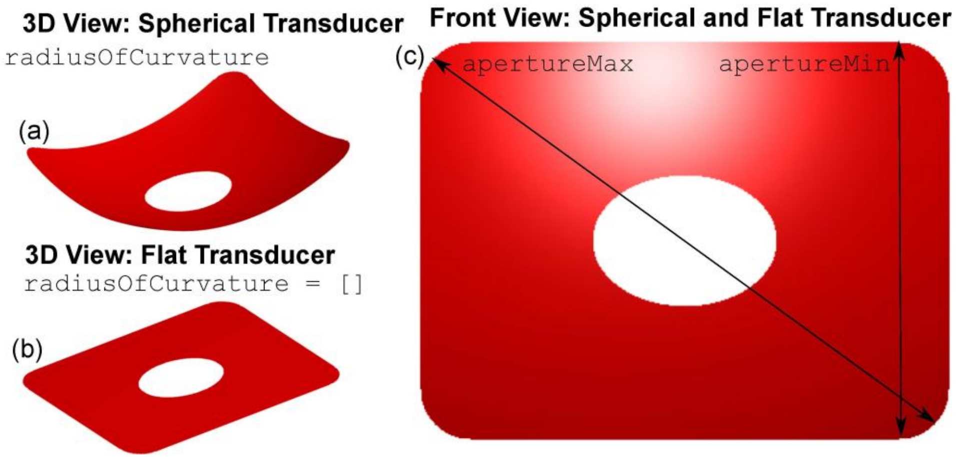


Fig. 1. General sketch of a transducer with a central opening.

rely on a set of nominal parameters for the transducer, intended prior its manufacturing. These geometrical parameters can be expressed using three values: the maximum transverse size `apertureMax`, the minimum transverse size `apertureMin`, and the radius of curvature `radiusOfCurvature` of the transducer's spherical surface (Fig. 1c). For flat transducers, the `radiusOfCurvature` variable can either be omitted or set as an empty vector: `radiusOfCurvature = []`. It is clear that for circular transducers, `apertureMax = apertureMin`. For convenience, these three geometric parameters are combined into a single MATLAB/Octave structure, `Geometry`, which describes the transducer's nominal geometric parameters:

```
% 'Geometry' struct with fields:
%   'radiusOfCurvature': nominal radius of curvature of the transducer in m
%   'apertureMin': characteristic minimum aperture of the transducer in m
%   'apertureMax': characteristic maximum aperture of the transducer in m
%                  (apertureMax = apertureMin for circular transducers)
```

Note that the described geometric parameters do not need to be precise, as they are used only as an initial approximation for the automated alignment procedure and can be adjusted when working with the toolbox based on the projection results.

Medium

The measurements should be performed in a tank filled with degassed and deionized water at a known temperature. Given the temperature, one can determine the sound speed and density of the water using the results of various studied [Sapozhnikov 2015]. For an estimated simulation, “default values” of a sound speed of 1500 m/s and density of 1000 kg/m^3 can be used. The propagation medium parameters can be defined as a structure:

```
% 'Medium' struct with fields:
%   'soundSpeed': sound speed in m/s
%   'density': density in kg/m^3
```

Holography scan

This software is optimized for transient holography, which is based on the excitation of a transducer by a short pulse. In transient mode, the recorded waveforms have a wide frequency spectrum. This allows for broadband characterization of the transducer through hydrophone signal decomposition into spectral components [Sapozhnikov 2006, Tsysar 2021, Maxwell 2022].

The central frequency of the pulse should match the transducer’s nominal operating frequency, `frequencyNominal` (typically around 1 MHz), and the excitation burst should be as short as possible to ensure pulse stability and repeatability, typically 2 cycles in

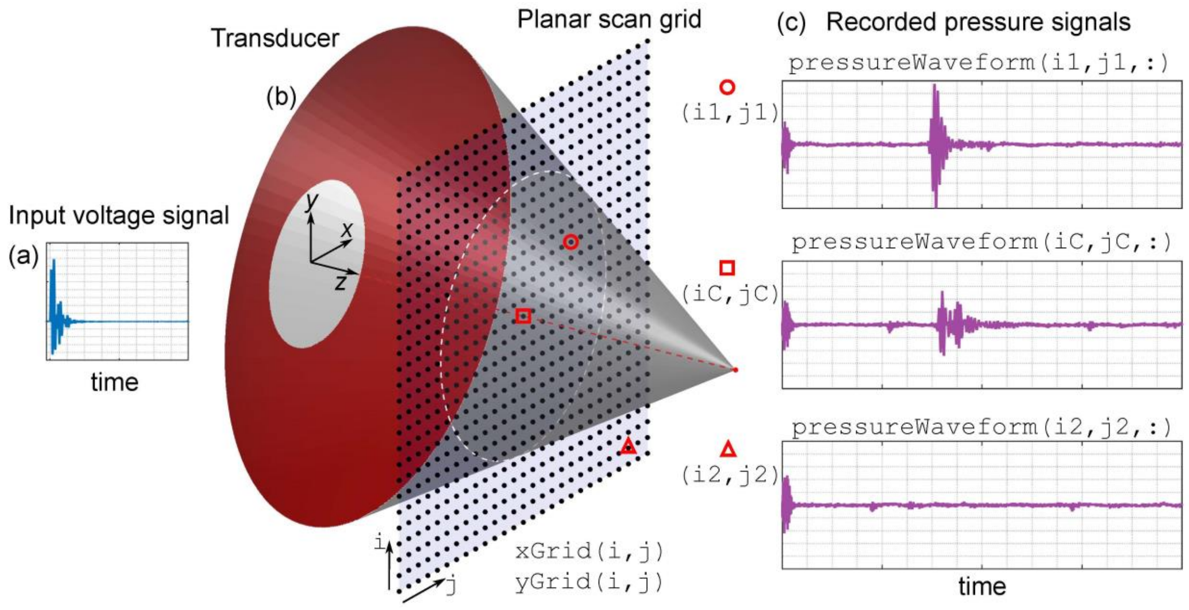


Fig. 2. Holography scan sketch. This sketch is for demonstration purposes and does not represent a real-case holography grid.

duration (Fig 2a). The pulses should be sent with a pulse repetition frequency (PRF) based on the burst length, hydrophone safety limits, and performance of the positioning/acquisition system (typically 5 – 500 Hz).

Fig. 2b shows a typical sketch of holography measurements. A hydrophone, attached to a 3D positioning system is subsequently located at the vertices of a Cartesian grid in a planar (2D) scan region *xy* perpendicular to the transducer axis *z*. The x- and y-grid steps are defined as `dx, dy`.

The most illustrative way to define a hologram scan grid is by using two matrices `xGrid(i,j)` and `yGrid(i,j)`, where the row indices `i` and the column indices `j` correspond to the respective axes directions. One of the most common ways to define these matrices in MATLAB/Octave is with the function `meshgrid`, which maps the *y*-direction with rows (i-index) and the x-direction with columns (j-index), as shown in Fig. 2b. Although this representation is used by default in the toolbox, users can map their vertex sets `xGrid` and `yGrid` in any convenient order and indexing directions. For example, they can be presented as 1D vectors `xGrid(iLin)`, `yGrid(iLin)`, as long as the 1D index `iLin` traverses all hologram vertices. However, for convenience, we consider the `meshgrid` notation.

At each spatial point, a hydrophone signal is recorded using an oscilloscope. A good practice is to acquire several pulses at each vertex of the hologram (typically up to 16) and then average them to minimize inherent noise. The scan results can be represented as a 3D matrix, `pressureWaveform(i,j,s)`, where the third dimension `s` corresponds to the number of time samples in the acquired signal. The time samples are stored in the time window vector, `time = [time(1), time(2), ... , time(end)]`, where the first sample `time(1) = 0`, corresponds to the start of the acoustic signal. The `time` vector should have a uniform time step, `timeStep = time(s) - time(s-1)`. Examples of three hydrophone signals acquired at different grid vertices, `(i1,j1)`, `(iC,jC)`, `(i2,j2)` are shown in Fig. 2c ("circle", "square", and "triangle" markers respectively). Thus, in MATLAB/Octave notation, a pressure signal at a vertex `(i,j)`, can be represented as a 1D vector, `pressureWaveform(i,j,:)`. Similarly, in the case of 1D spatial indexing, the scan result matrix will be 2D: `pressureWaveform(iLin,s)`.

Fig. 2c shows that the recorded signals occupy only a portion of the acquisition time. It is important to choose the correct time window to ensure that the recorded signals do not include reflections from the hydrophone holder or the water tank walls, or electrical pickup noise of a length `tPickupNoise` at the beginning of the signal recorded at the center `(iC,jC)` of the hologram (Fig. 3a). The simplest way to approximate the time window is through estimates of the propagation path for the nominal transducer parameters and the ringing time of the recorded signal, `tRinging` (Fig. 3a).

First, the `zPosition` can be estimated as `zPosition = soundSpeed*tReceive`, where `tReceive` is the time delay of the signal when the hydrophone is positioned at the center of the hologram (Fig. 3). The maximum distance between the transducer and the hologram is shown in Fig. 3b for a circular transducer is denoted as `distLong`. The specific distance depends on the transducer's shape; however, the distance can be approximated as:

```
distLong = sqrt(apertureMax^2 + zPosition^2)
```

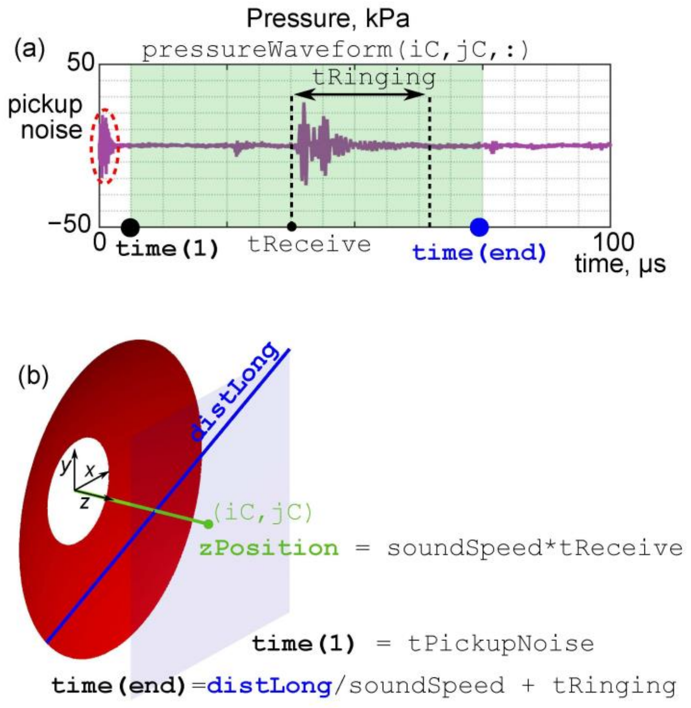


Fig. 3. Example of selecting a time window for a scan.

Given the sound speed, the ringing time `tRinging`, and the pickup noise time `tPickupNoise` estimates, we can approximate the first and last time samples as:

```
time(1) = tPickupNoise

time(end) = distLong/soundSpeed + tRinging
```

The chosen time window, is shown in Fig. 3a as a green rectangle within the initial full-acquisition time window.

It is important to note that while selecting the correct time window improves the quality of the results by excluding recorded reflections and pickup noise, choosing the wrong window is not critical to the outcome of transient hologram post-processing. This is due to the time separation between the recorded transient holography burst and the reflections.

Finally, the parameters described above allow the definition of a transient holography planar scan as a structure:

```
% 'HologramTr' struct with fields:

%     'xGrid': vector or matrix with x-coordinates of the Cartesian grid in
%              m at each grid node of the hologram
%      'yGrid': vector or matrix with y-coordinates of the Cartesian grid in
%              m at each grid node of the hologram
%      'dx': x-step of the hologram Cartesian grid in m
%      'dy': y-step of the hologram Cartesian grid in m
%      'zPosition': approximate z-position in m of the hologram, assuming
%                  zero at the apex of the transducer
%      'time': vector with the time samples in s at each time grid nodes
%             for the recorded waveforms
%      'pressureWaveforms': 2D or 3D matrix with the acoustic pressure
%                          waveforms in Pa at corresponding grid nodes of the
%                          hologram
%                          if xGrid/yGrid is a 2D matrix, then
%                          size(pressureWaveforms) = [size(xGrid, 1)
%                          size(xGrid, 2) length(time)]
%                          if xGrid/yGrid is a vector, then
%                          size(pressureWaveforms) = [length(xGrid)
%                          length(time)]
```

Single-frequency data representation

The matrix `HologramTr.pressureWaveforms` presents scan data in a time-domain format which can be displayed a set of time frames for different time samples. An example of this representation is shown in Fig. 4a for three time samples `s1`, `s2`, `s3`. The recorded waveforms have a wide frequency spectrum, which, upon decomposing the hydrophone signal into spectral components, allows for the broadband characterization of the transducer. By applying the Fourier transform to the time-domain signal, one can present it in the frequency domain as a sum of independent single-frequency holograms at different operating frequencies (Fig. 4b). This is an important feature of transient holography, as a single scan provides information about various single-frequency regimes that are widely used in therapeutic applications.

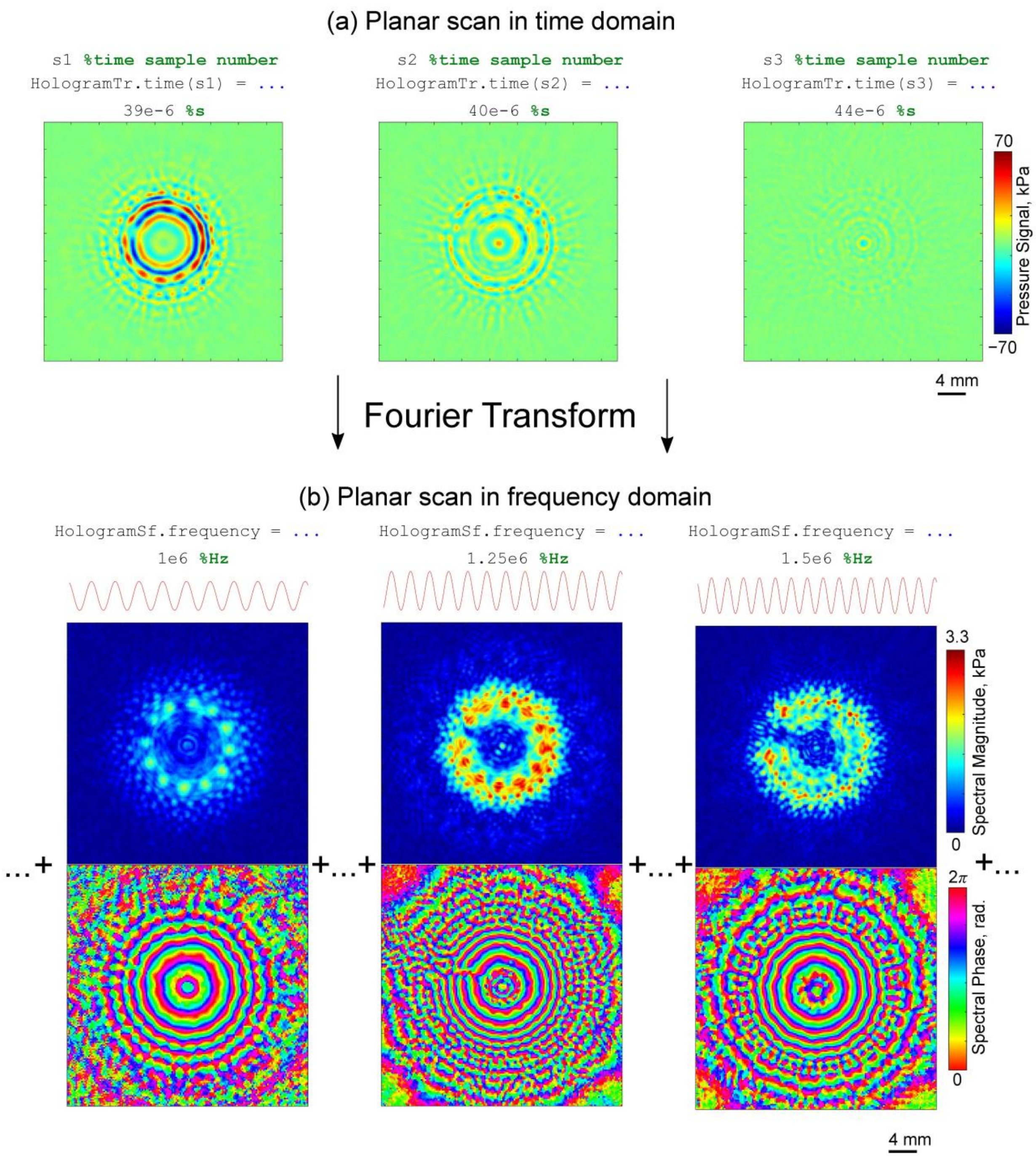


Fig. 4. Example of time-domain and frequency-domain representations of the hologram.

Each single-frequency component includes the pressure magnitude and phase distribution, which can be presented as a matrix of complex numbers `complexPressureAmplitude(i,j)`. Note that this complex representation depends on the exponent sign convention `exp(+2*pi*1i*frequency*time)` or `exp(-2*pi*1i*frequency*time)` in the discrete Fourier transform. Here `1i` is the MATLAB/Octave denotation of the imaginary unit. The toolbox utilizes the exponent sign parameter `expSign = +1` or `−1` to specify the convention used. For example, the `fft` function in MATLAB/Octave utilizes the `exp(+2*pi*1i*frequency*time)` convention so the default exponent convention used by the toolbox is `expSign = +1`. An easy way to check your exponent sign is provided below in Example 5, Details.

The software includes an automatic tool to extract a single-frequency component at a desired `frequency`, which will be described in detail later. It presents a single-frequency hologram as a structure similar to the transient one:

```
% 'HologramSf' struct with fields:
%   'expSign': +1 or -1, depending on the exponent sign convention
%               exp(+1i * omega * t) or exp(-1i * omega * t). E.g., the "fft"
%               function in MATLAB utilizes the exp(+1i * omega * t)
%               convention, so in this case, expSign is +1
%   'frequency': the operating frequency of the transducer in Hz
%   'xGrid': vector or matrix with the x-coordinates in m at each grid node
%       of the hologram
%   'yGrid': vector or matrix with the y-coordinates in m at corresponding
%             grid nodes of the hologram
%   'zPosition': z-position in m of the hologram, assuming zero at the apex
%                 of the transducer
%     'dx': x-step of the hologram Cartesian grid in m
%     'dy': y-step of the hologram Cartesian grid in m
%   'complexPressureAmplitude': vector or matrix with the complex pressure
%                                 amplitude values in Pa at corresponding grid
%                                 nodes of the hologram
```

Holography projection

A transient or single-frequency holograms (`HologramTr` or `HologramSf`) can be used as a boundary condition for the Rayleigh integral or the angular spectrum method to perform numerical projection and reconstruct the 3D acoustic field (Fig. 5, “Forward-Projection”)

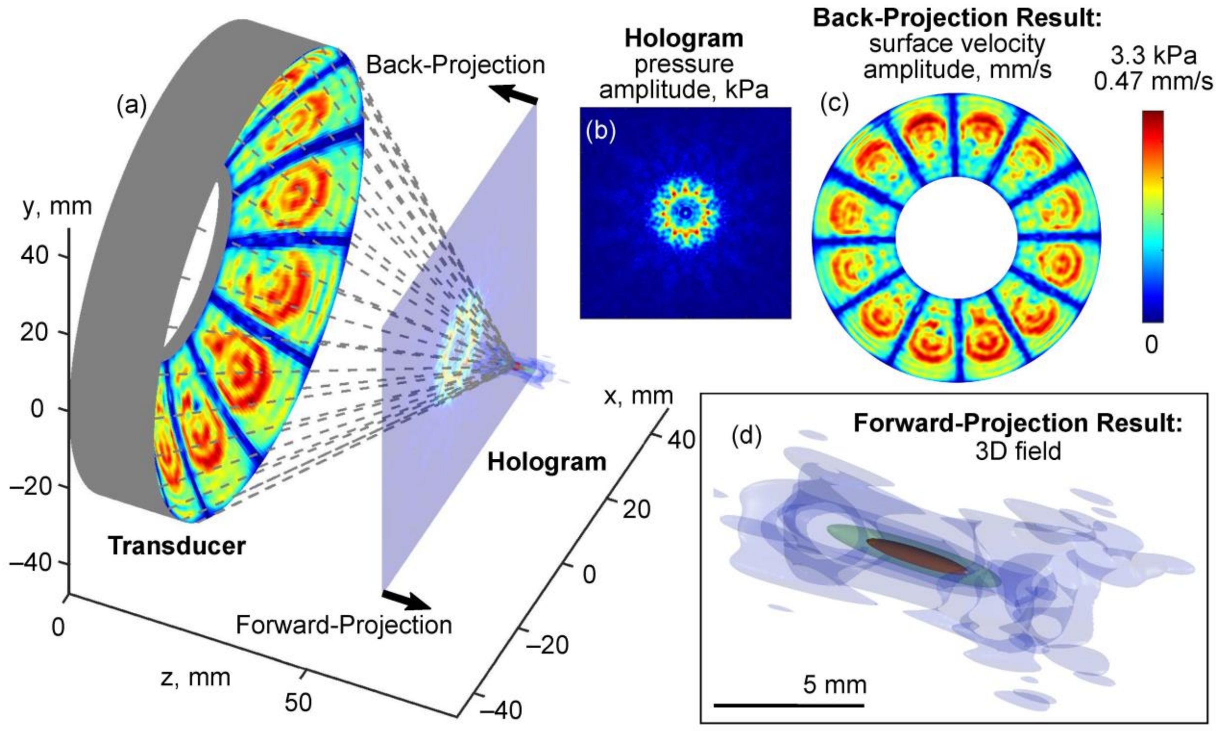


Fig. 5. Rayleigh integral-based hologram projections.

[O'Neil1 949, Sapozhnikov 2015]. Additionally, the hologram-originated field projection can be simulated in the reverse direction (Fig. 5, "Back-Projection"), allowing not only for the calculation of the 3D field but also the reconstruction of the vibrational velocity distribution at the surface of the transducer (Fig. 5, "Transducer"). The vibrational velocity or pressure distribution at the flat or spherical surface of a transducer can also be used as a boundary condition for forward/back projections to either simulate the entire transducer field or set a boundary condition for the transducer at a plane for subsequent numerical simulations.

Although the reconstructed 3D acoustic field is not entirely derived from direct hydrophone measurements, it has been demonstrated to have an error smaller than the hydrophone measurement error [Sapozhnikov 2015].

The core function of the toolbox, `rayleigh_simulator`, is a universal tool applicable to Rayleigh integral-based forward/back projection of either a single-frequency boundary condition or a set of single-frequency components. This function has flexible input and output options for either pressure or velocity on either a spherically curved or a flat surface. The technical details of this core function are described in the corresponding chapter Make Your Own Tools.

Transducer Grid Definition

As described in the previous section, the toolbox can project the field not only for planar holography scans but also for spherically curved and flat transducers. To provide this capability, structure describing a single-frequency transducer is used, which is similar to single-frequency hologram structures `HologramTr` and `HologramSf`:

```
% 'TransducerTr' struct with fields:
%     'radiusOfCurvature': (optional, can be omitted) radius of curvature of
%                          the transducer in m. Omit this field or set it as
%                          an empty vector if your transducer is flat
%     'xGrid': vector or matrix with x-coordinates of the Cartesian grid in
%              m at each grid node of the transducer surface
%     'yGrid': vector or matrix with y-coordinates of the Cartesian grid in
%              m at each grid node of the transducer surface
%     'zGrid': vector or matrix with z-coordinates of the Cartesian grid in
%              m at each grid node of the transducer surface. Users can omit
%              this variable, in such cases, the toolbox will automatically
%              determine the transducer type—spherical or flat. For a
%              spherical transducer, zGrid nodes will be set according to the
%              sphere equation:
%              zGrid = radiusOfCurvature - sqrt(radiusOfCurvature^2 -
%              xGrid.^2 - yGrid.^2) For a flat transducer, all zGrid nodes
%              will be set to zero:
%              zGrid = zeros(size(xGrid))
%     'dx': x-step of the transducer Cartesian grid in m
%     'dy': y-step of the transducer Cartesian grid in m
%     'time': vector with the time samples in s at each time grid nodes
%             for the recorded waveforms
%     'velocity': 2D or 3D matrix with the vibrational velocity
%                             waveforms in m/s at corresponding grid nodes of
%                             the transducer
%                           if xGrid/yGrid is a 2D matrix, then
%                           size(velocity) = [size(xGrid, 1)
%                           size(xGrid, 2) length(time)]
%                           if xGrid/yGrid is a vector, then
%                           size(velocity) = [length(xGrid)
%                           length(time)]

% 'TransducerSf' struct with fields:
%   'expSign': +1 or -1, depending on the exponent sign convention
%              exp(+1i * omega * t) or exp(-1i * omega * t). E.g., the "fft"
%              function in MATLAB utilizes the exp(+1i * omega * t)
%              convention, so in this case, expSign is +1
%   'frequency': the operating frequency of the transducer in Hz
%   'radiusOfCurvature': (optional, can be omitted) radius of curvature of
                         the transducer in m. Omit this field or set it as
                         an empty vector if your transducer is flat
%   'xGrid': vector or matrix with the x-coordinates in m at each grid node
%            of the transducer surface (Cartesian grid only)
%   'yGrid': vector or matrix with the y-coordinates in m at each grid node
%            of the transducer surface (Cartesian grid only)
%   'zGrid': vector or matrix with the z-coordinates in m at each grid node
%            of the transducer surface (Cartesian grid only). Users can omit
%            this variable, in such cases, the toolbox will automatically
%            determine the transducer type—spherical or flat. For a
%            spherical transducer, zGrid nodes will be set according to the
%            sphere equation:
%            zGrid = radiusOfCurvature - sqrt(radiusOfCurvature^2 -
%            xGrid.^2 - yGrid.^2) For a flat transducer, all zGrid nodes
%            will be set to zero:
%            zGrid = zeros(size(xGrid))
%   'dx': x-step of the transducer Cartesian grid in m
%   'dy': y-step of the transducer Cartesian grid in m
%   'complexVelocityAmplitude': complex velocity amplitude in m/s at each
%                                node of the transducer surface grid
```

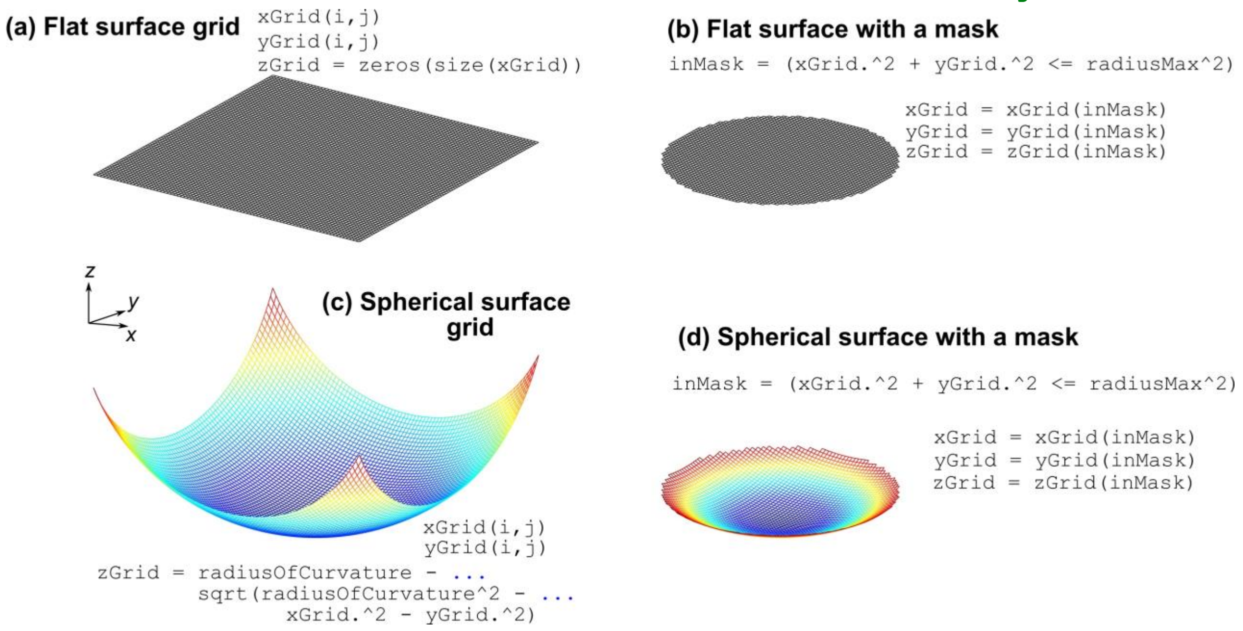

(a) Flat surface grid
xGrid(i,j)
yGrid(i,j)
zGrid = zeros(size(xGrid))
(b) Flat surface with a mask
inMask = (xGrid.^2 + yGrid.^2 <= radiusMax^2)
xGrid = xGrid(inMask)
yGrid = yGrid(inMask)
zGrid = zGrid(inMask)
z
y
x
(c) Spherical surface grid
xGrid(i,j)
yGrid(i,j)
zGrid = radiusOfCurvature - ...
sqrt(radiusOfCurvature^2 - ...
xGrid.^2 - yGrid.^2)
(d) Spherical surface with a mask
inMask = (xGrid.^2 + yGrid.^2 <= radiusMax^2)
xGrid = xGrid(inMask)
yGrid = yGrid(inMask)
zGrid = zGrid(inMask)

Fig. 6. Grid definition for transducers.

Note that, unlike the `Hologram` structures, the `Transducer` structures use a surface definition format consisting of three variables `xGrid`, `yGrid`, `zGrid`. This is necessary because a transducer can have either a flat or spherical shape. The shape is represented in the `zGrid` variable for flat (Fig. 6a) and spherical (Fig. 6c) surfaces, assuming the transducer is positioned at the origin of the *z*-coordinate. Even though it is convenient to define a transducer surface as a rectangular grid `(i,j)`, the number of points can be based on the transducer's geometry. For example, a circular mask can be applied to the transducer located inside a circle with radius `radiusMax`, which helps to significantly reduce the number of simulation points, as shown in Fig. 6c, d. The software automatically applies these masks in certain cases by setting up the value basing on the maximum size of the nominal geometry of the transducer

`radiusMax = 0.5*(1+apertureReserve)*Geometry.apertureMax`,

where `apertureReserve` is a preset fraction of the aperture used to increase the nominal output window. The transducer structures, `TransducerTr` and `TransducerSf`, can be saved using the GUI of the “Holography Toolbox” and utilized as boundary conditions in the “Simulation Toolbox” for single-frequency and transient field simulations.

XLSX Templates for Single-Frequency Structures

`HologramSf`, `TransducerSf`, `Geometry`, and `Medium` structures can be created for cases of interest using “.xlsx” templates located in “xDDx\examples\data_for_examples\xlsx_templates”. These templates are divided into sheets containing scalar parameters (such as `expSign` and `frequency`) and separate sheets for each matrix parameter (such as `xGrid`). Complex parameters, like `complexPressureAmplitude`, are split into two sheets: one for the amplitude matrix and another for the phase matrix. Each .xlsx file includes a sheet named “Info”, which provides detailed explanations of all parameters.

Misalignment problem

An important challenge when using measured holograms to reconstruct a transducer’s surface vibrational velocity is that the transducer is typically positioned manually in a water tank, making its orientation relative to the axes of the positioning system uncertain. Specifically, the axis of the radiated ultrasound beam (the "acoustical" axis) is often slightly tilted and shifted (Fig. 7) relative to the corresponding axis of the positioning system (the "mechanical" axis). Given a typical rotational error of 1° (0.018 radians) and a hologram axial distance of 50 wavelengths from the source, the approximate linear displacement after back-projection would be `0.018 * 50 ≈ 1` wavelength, resulting in noticeable phase errors at the surface of the reconstructed source [Kaloev 2024]. The software includes tools for both manual and automated correction of this misalignment.

Fully automated correction is possible for a focused ultrasound source with a diameter much larger than a wavelength and a focal distance comparable to the diameter. In this case there is a high-amplitude (focal) region, shaped like an elongated ellipsoid. For such an ellipsoid, the axis of symmetry represents the acoustic axis of the transducer, and the point of maximum pressure amplitude corresponds with high accuracy to the center of curvature of the radiating surface (Fig. 7a).

The automatic tool detects the position and shape of this focal lobe by forward-projecting the hologram, then uses the 3D projection results to determine the center of curvature and the acoustic *z*-axis of the transducer. The Hologram is then forward-projected to obtain the Aligned Hologram (Fig. 7b), which is back-projected onto the transducer surface to reconstruct the velocity distribution.

The detailed description of the algorithm can be found in the paper [Rosnitskiy 2025].

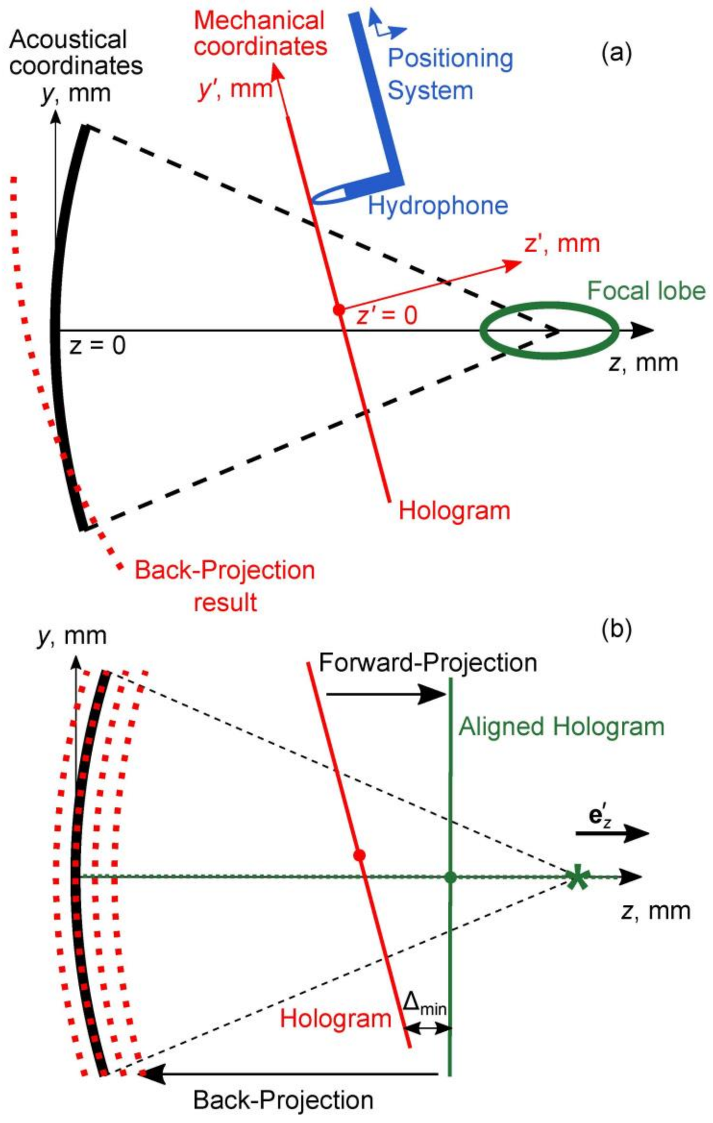


Fig. 7. Rotational misalignment between the positioning system coordinates and the acoustical coordinates of the transducer.

Holography measurements SOP

The tools for correcting geometrical misalignments in the experimental setup allow for a simplified holography measurement procedure, which can be outlined as the following standard operating procedure (SOP):

1) Select the hologram z-position. For a focused transducer, an approximate optimal distance is about half of the nominal radius of curvature [Sapozhnikov 2015]. For a flat transducer any z-position can be chosen, provided it is sufficient to avoid reflections from the hydrophone holder in the time window (typically 15 – 20 wavelengths from the transducer surface).

2) Manually approximate the hydrophone x and y positions to align with the center of the transducer (Fig. 8).

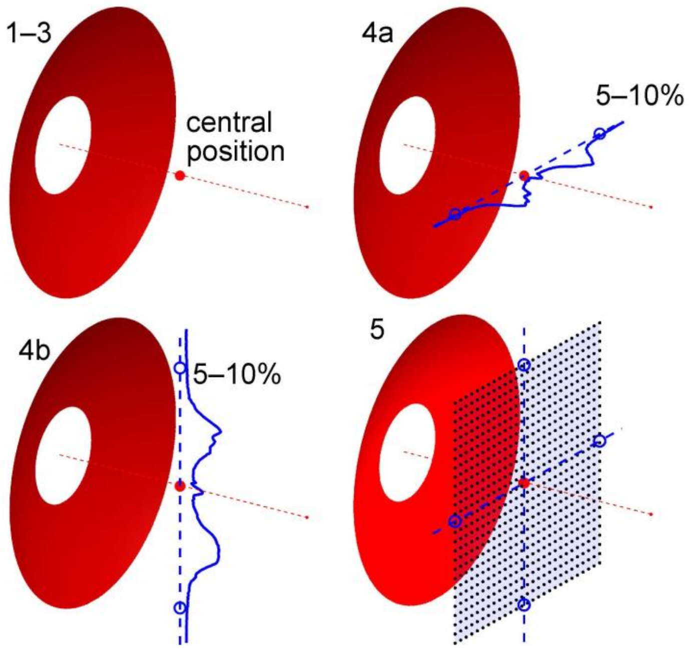


Fig. 8. Step-by-step set up of holography measurements.

3) Estimate the z-position of the hologram using the signal delay `tReceive` at the chosen central position: `zPosition = soundSpeed*tReceive`.

4) Perform two linear scans along the *x* and *y* axes, centered relative to the chosen central position to determine the hologram transverse size that captures the entire field. A useful empirical rule is that the peak pressure levels at the edges of the region should be less than 5–10% of the maximum value in the holography scan.

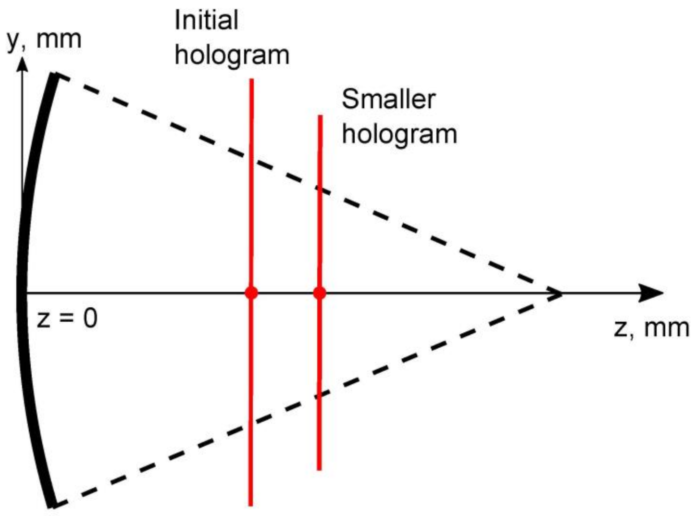


Fig. 9. Selecting a smaller hologram scan position.

5) Plan the holography grid for the selected planar region. A scanning grid step can be set to 3 grid points per wavelength of the ultrasound wave in the propagation medium at the central frequency of the burst. This grid step has been confirmed as reliable for transient holography [Sapozhnikov 2003, Sapozhnikov 2006]. For example, in idealized water with a sound speed of 1500 m/s and a transducer with a central frequency of 1 MHz, the wavelength is 1.5 mm, so the x- and y-steps for the holography grid should be 0.5 mm.

6) If the planar scan with the chosen grid parameters contains too many points and the estimated scan time is too long, the center of the hologram can be shifted toward the focus in the case of a focused transducer. This adjustment can help meet the 5–10% requirement with a more manageable grid size (Fig. 9). While this reduces the accuracy of the hologram, it remains sufficient for detailed field projection.

Examples

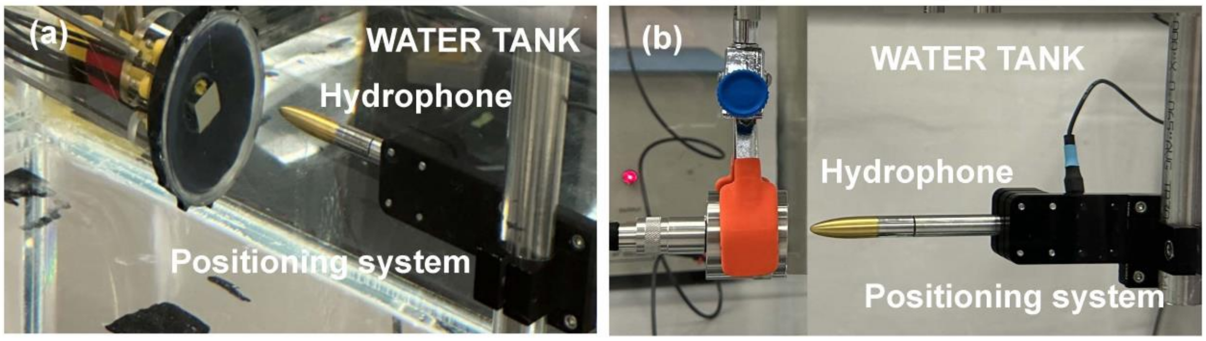
This manual presents two examples of holography scans: one for a 12-element spherically shaped focused transducer and another for a flat transducer (Fig. 10a and b respectively). The figure shows photos of the experimental setups, and the table below summarizes all parameters of the experiments.

The following equipment was used:

3-D positioning system: Velmex (Velmex Inc., Bloomfield, NY)

Hydrophone: Onda HGL-0200 with an AH-2020 preamplifier set at 20 dB gain (Onda Corp., Sunnyvale, CA)

Oscilloscope: Gage Razor 14 digitizer (DynamicSignals LLC, Lockport, IL, USA).



Focused Transducer

```
frequencyNominal = 1.25e6 %Hz
apertureMin = 87e-3  %m
apertureMax = 87e-3 %m
radiusOfCurvature = 87e-3 %m
```

Flat Transducer

```
frequencyNominal = 1e6 %Hz
apertureMin = 38e-3  %m
apertureMax = 38e-3 %m
radiusOfCurvature = []
```

Fig. 10. Experimental setups.

Parameter	**Focused Transducer**	**Flat Transducer**
Burst Parameters		
Central frequency	1.25 MHz	1 MHz
Number of cycles	2	1
Pulse Repetition Frequency (PRF)	20	400
Scan Parameters		
z-position	58 mm	26 mm
x-range	–34 to 34 mm	–30 to 30 mm
x-step	0.4 mm	0.5 mm
y-range	–34 to 34 mm	–30 to 30 mm
y-step	0.4 mm	0.5 mm
Time Window Parameters		
Time range	27 to 76 μs	4 to 60 μs
Time step	12.5 ns	20 ns
Nominal Transducer Geometric Parameters		
Radius of curvature	87 mm	N/A
Minimum aperture	87 mm	38 mm
Maximum aperture	87 mm	38 mm
Medium Parameters		
Sound speed	1489 m/s	1493 m/s
Density	997 kg/m^3	996 kg/m^3

QUICK START

Quick Start Tool

The Quick Start Tool is designed for hologram back-projection and reconstruction of the transducer's surface vibrational velocity pattern. The reconstruction can be performed with or without alignment, in either the single-frequency or transient regime. It requires the minimum possible number of input parameters, while most simulation parameters are defined automatically. This feature is useful for quick checks of holography measurements. For more precise results when reconstructing the vibrational pattern of the transducer surface, use the step-by-step approach described in section Advanced Transducer Diagnostics.

***Example 1:** Quick Start for a Spherical Transducer*

Path: "xDDx\examples\quick_start_spherical.m"

This example shows the most common case for a focused transducer's diagnostic: reconstruction of the surface vibrational velocity pattern for a single frequency (`frequencyNominal`) case with hologram alignment.

General input parameters

- Simulation regime parameters:

```
simulationDevice = 'cuda'; %simulation device: 'cuda' or 'cpu'
```

```
doTransientCase = false; %true for transient case, false for single-frequency case
```

```
doAlignment = true; %set true to perform alignment, or false to perform back-projection with no alignment
```

- Input data-file:

A MAT file containing two structures `HologramTr` (page 13) and `Medium` (page 11)

```
inputParametersFileName =
'..\data_for_examples\sector\holo_data_sector_transient.mat'; %path to input data with transient hologram and medium parameters
```

- Nominal transducer parameters:

```
frequencyNominal = 1.25e6; %desired frequency of the output single-frequency hologram in Hz

apertureMin = 87e-3; %characteristic minimum aperture of the transducer in m

apertureMax = 87e-3; %characteristic maximum aperture of the transducer in m (apertureMax = apertureMin for circular transducers)

radiusOfCurvature = 87e-3; %(optional) nominal radius of curvature of the transducer in m. Omit this variable or define it as an empty value [] if your transducer is flat
```

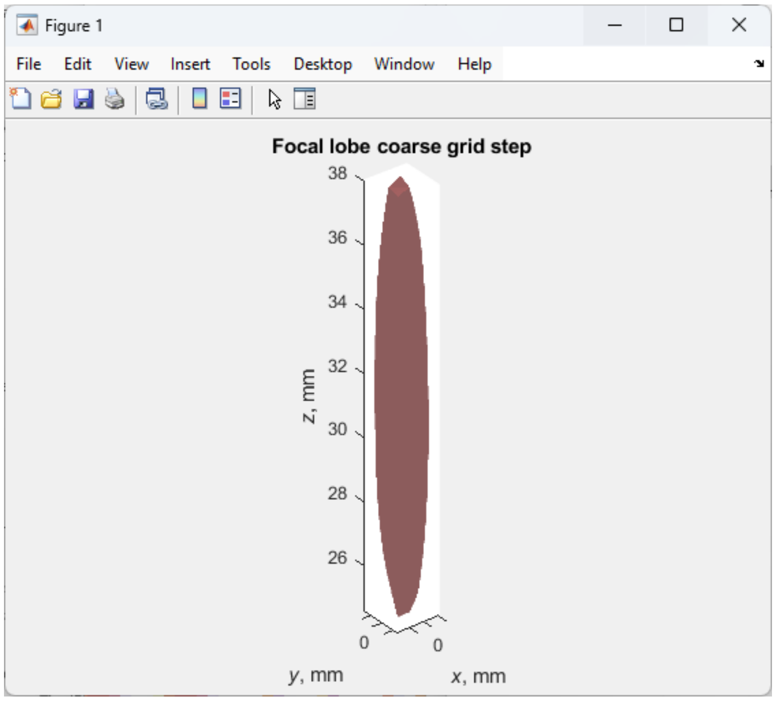
- Save back-projection results:

```
saveTransducer = true; %true to save the boundary condition for the transducer at frequencyNominal as a 'TransducerSf' structure
```

The other parameters of the script are technical and are selected so that they can be set to default for most transducer cases. These will be described in detail later.

Simulation output

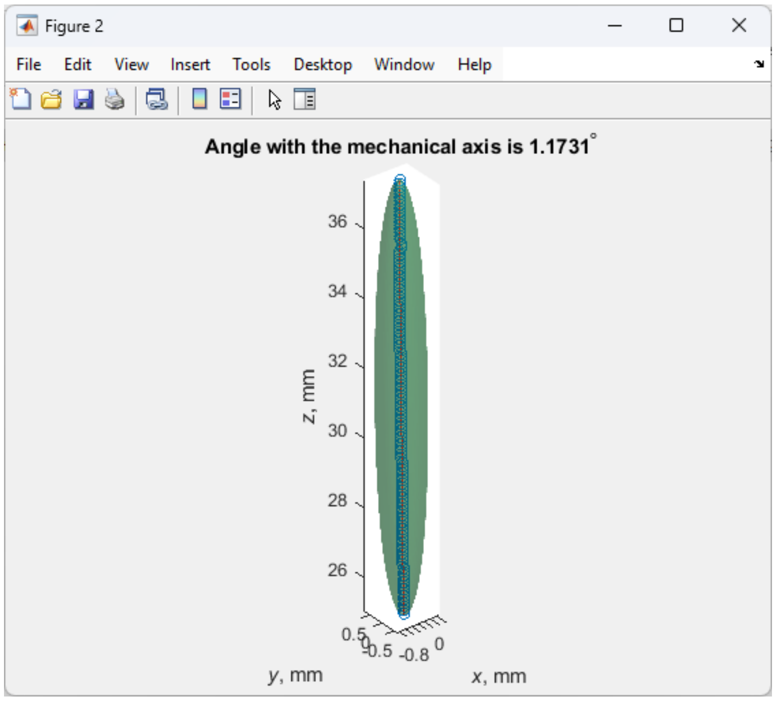
- Coarse-grid shape of the 3D focal lobe simulated for alignment (Fig. 7).



This shape presents an iso-surface of points with equal pressure values `focalLobeLevel` normalized by the maximum pressure amplitude. In the Quick Start Tool, the default level 0.6 is used. It can be changed in the input parameters if needed:

```
focalLobeLevel = 0.6; %isolevel related to the pressure maximum for extracting
the isosurface of the main focal lobe
```

- Fine-grid shape of the same 3D focal lobe simulated for alignment (Fig. 7). The solid line passing through the circular markers provides the ordinary least squares-estimated direction of the acoustic axis of the transducer and reveals a rotation angle between the acoustic and mechanical axes



- Reconstructed vibrational velocity amplitude (in m/s) at the spherical surface of the transducer after alignment, sampled with a default step of half the wavelength

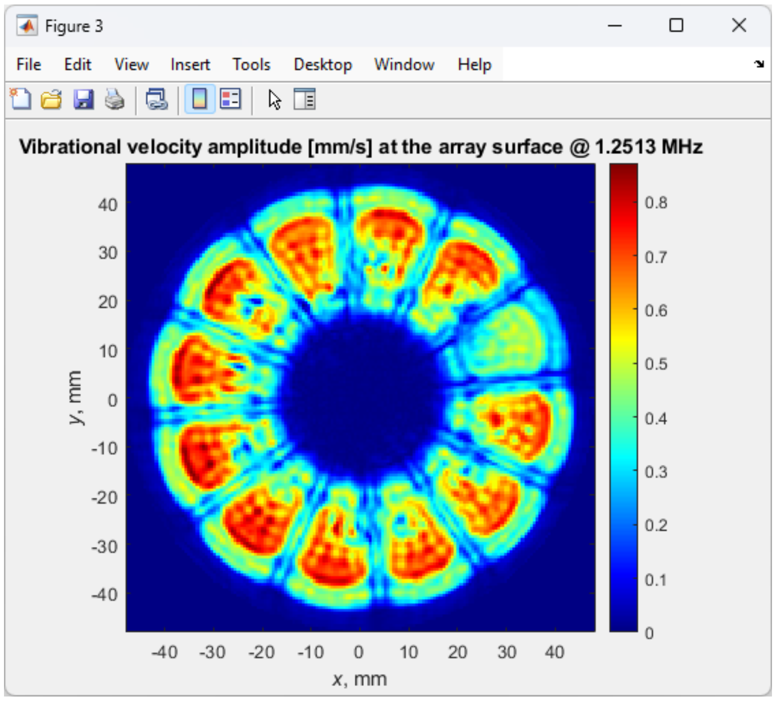


Figure 3
File Edit View Insert Tools Desktop Window Help
Vibrational velocity amplitude [mm/s] at the array surface @ 1.2513 MHz
y, mm
x, mm
40
30
20
10
0
-10
-20
-30
-40
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0

- Reconstructed vibrational velocity phase (in radians) at the spherical surface of the transducer after alignment

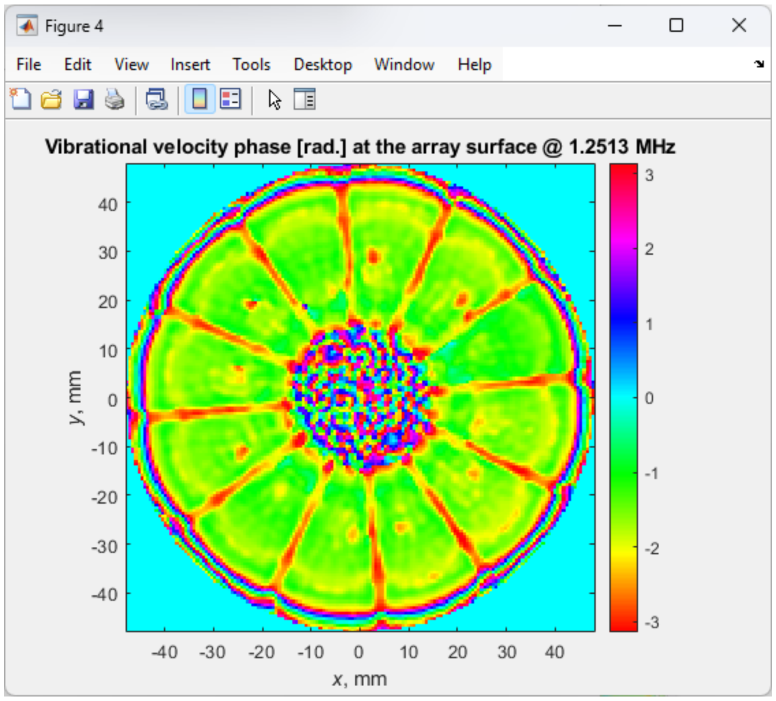


Figure 4
File Edit View Insert Tools Desktop Window Help
Vibrational velocity phase [rad.] at the array surface @ 1.2513 MHz
y, mm
x, mm
40
30
20
10
0
-10
-20
-30
-40
3
2
1
0
-1
-2
-3

The reconstruction is automatically performed for the frequency sample closest to `frequencyNominal`. The amplitude plot shows a distinguishable pattern of 12 elements.

However, the gaps between the elements have nonzero amplitudes, indicating that the reconstruction surface is closer to the focus than the actual surface, and thus the reconstructed field is slightly narrowed due to focusing. A specific tool to adjust the surface position will be presented later (Automatic Alignment Tool, page 53).

An important detail of the reconstruction results is that even though the array elements oscillate as uniform pistons, the obtained amplitude distribution is not uniform due to Lamb waves that propagate along the surface of the elements and manifest themselves in continuous wave (CW) regimes (including the single-frequency regime considered here) by affecting locally the vibrational velocity of the plates.

- If the logical flag `saveTransducer` is set to `true`, the `TransducerSf` structure (see page 19) for the reconstructed vibrational pattern will be saved in the same directory, with the filename formatted to reflect the simulation completion date and time as:

"transducer_yyyy_mm_dd_hh_mm_ss.mat"

Details

Simulation parameters can be changed to perform back-projection without alignment:

```
doAlignment = false; %set true to perform alignment, or false to perform back-projection with no alignment
```

In this case, the simulation output will only show the reconstructed vibrational velocity pattern at the tilted surface without alignment (Fig. 7a, dotted line):

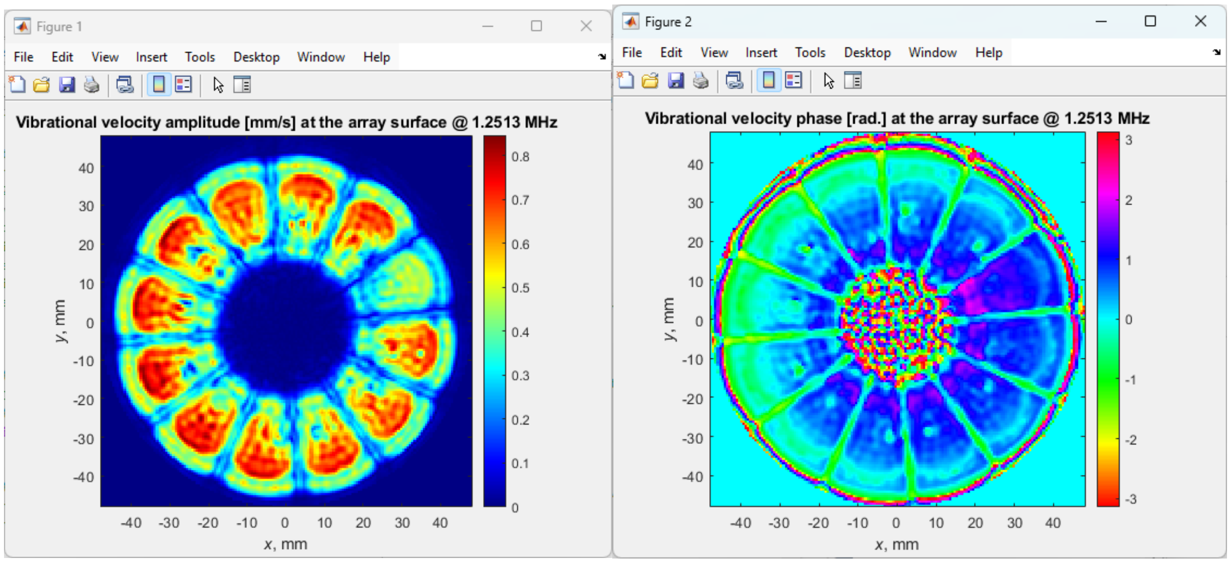


Figure 1
File Edit View Insert Tools Desktop Window Help
Vibrational velocity amplitude [mm/s] at the array surface @ 1.2513 MHz
y, mm
x, mm
40
30
20
10
0
-10
-20
-30
-40
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
Figure 2
Vibrational velocity phase [rad.] at the array surface @ 1.2513 MHz
3
2
1
0
-1
-2
-3

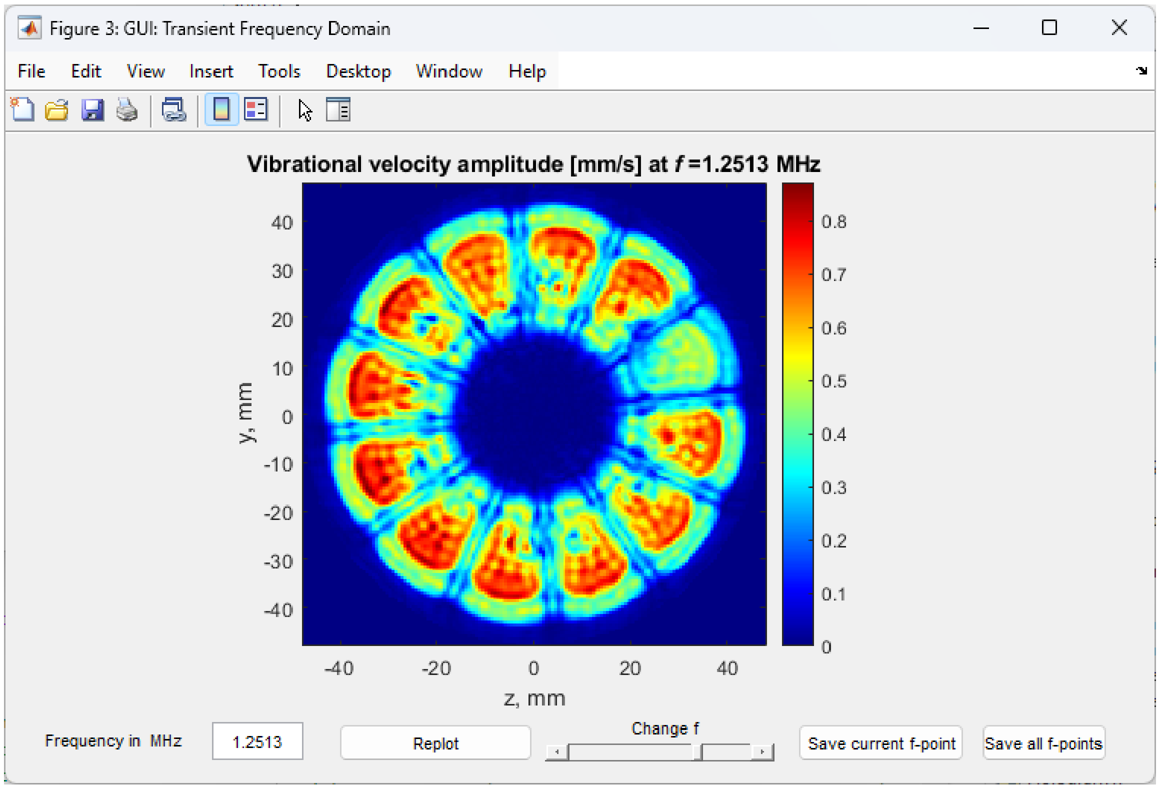
The rotational misalignment between the mechanical and acoustic axes is clearly seen in the phase distribution. The obtained distribution has a nonuniform structure, clearly indicating the rotation of the reconstruction surface. As a result, the obtained mismatched phase distribution does not provide information on phase uniformity at the transducer surface.

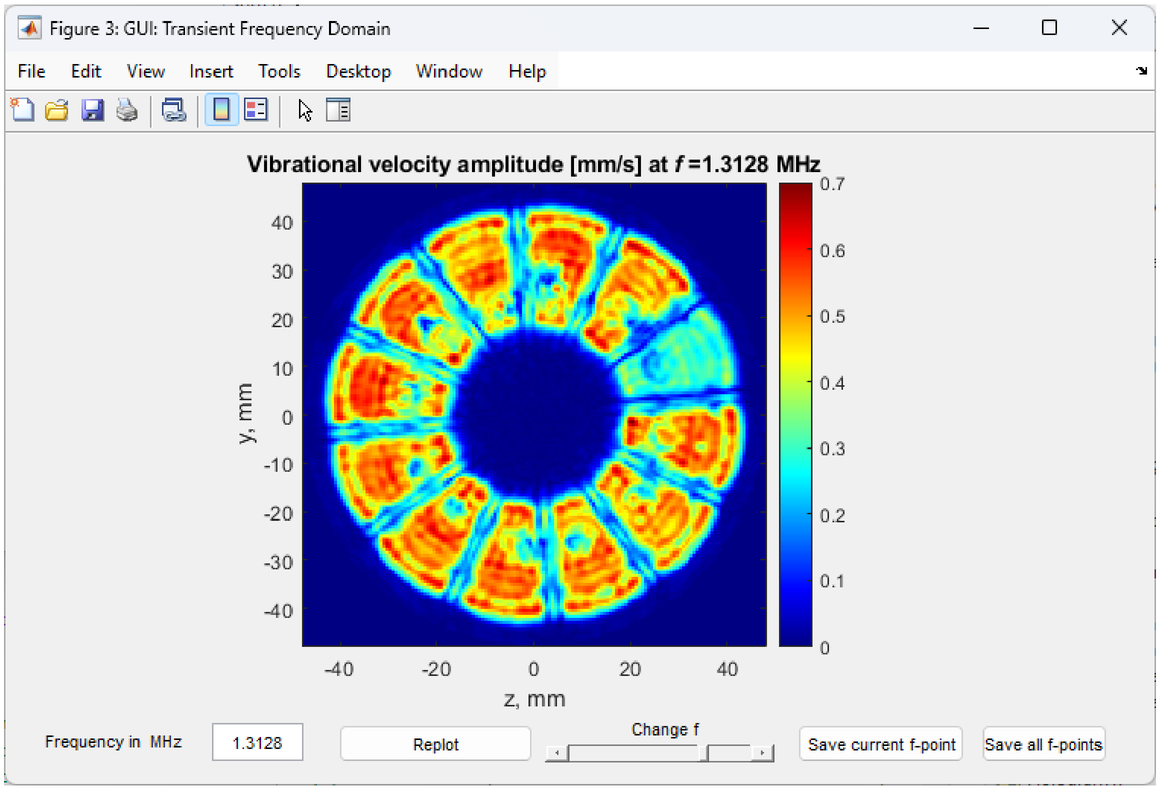
The most general option is transient reconstruction with alignment:

```
doTransientCase = true; %true for transient case, false for single-frequency case

doAlignment = true; %set true to perform alignment, or false to perform back-projection with no alignment
```

Simulation time in this regime is noticeably longer than in the single-frequency case. However, the results are presented in both the frequency and time domains. Each domain has its own GUI to display results for different time and frequency samples, save results, or manually select the frequency/time samples. The format of the saved data is explained in detail in Example 10.

1) Frequency domain output





One application of the frequency-domain GUI is related to selection of optimal operating frequencies for CW regimes. The pattern of transducer vibrations can be visualized over the range of frequencies present in the measured hologram. In this example case, the back-projected vibrational velocity distribution at the nominal frequency of 1.25 MHz was less uniform than a slightly higher frequency of 1.31 MHz. Therefore, an optimized transducer manufacturing procedure could involve performing holography-based analysis before designing the electrical impedance matching.

2) Time domain output

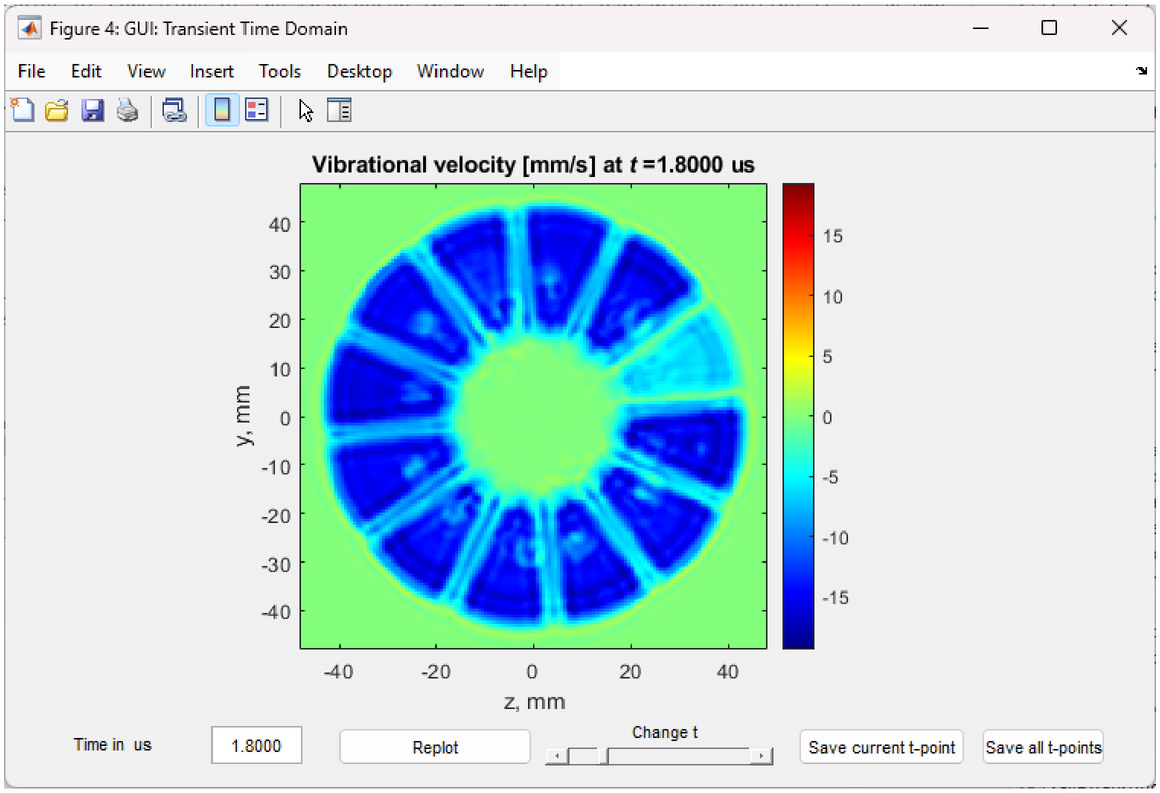


Figure 4: GUI: Transient Time Domain
File Edit View Insert Tools Desktop Window Help
Vibrational velocity [mm/s] at *t* =1.8000 us
y, mm
z, mm
Time in us
1.8000
Replot
Change t
Save current t-point
Save all t-points

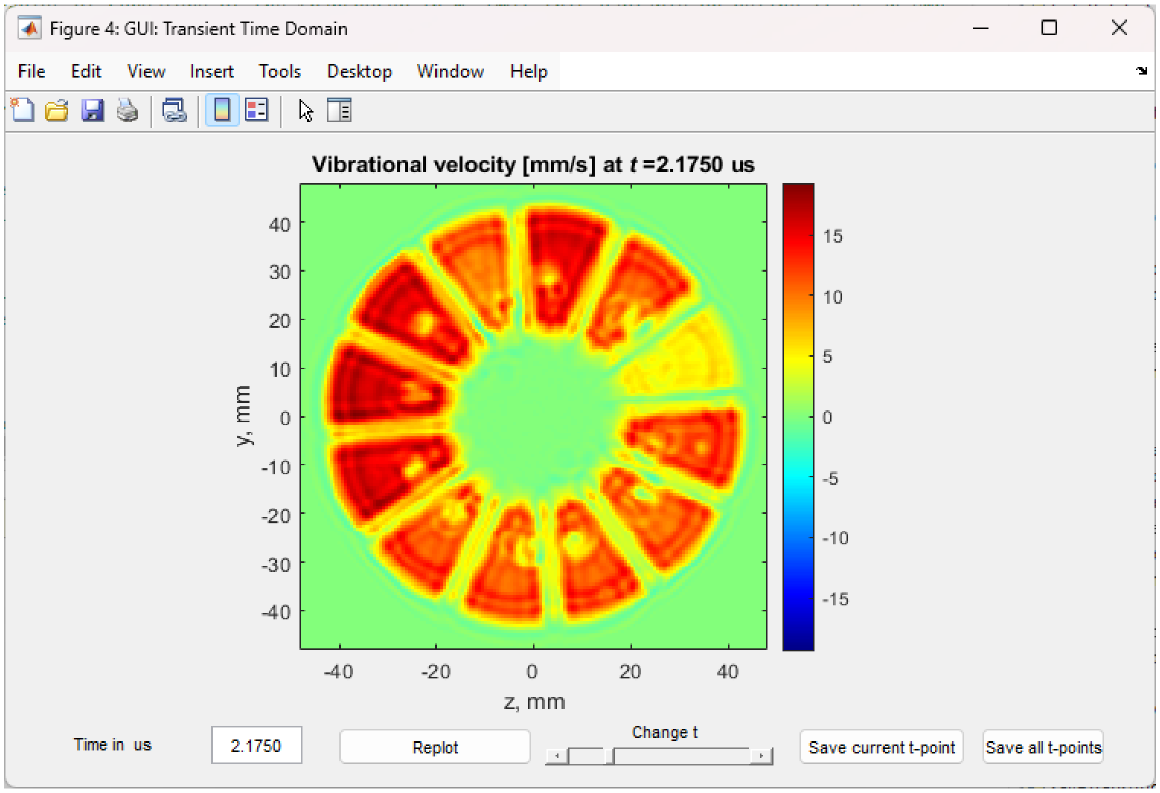


Figure 4: GUI: Transient Time Domain
File Edit View Insert Tools Desktop Window Help
Vibrational velocity [mm/s] at *t* =2.1750 us
y, mm
z, mm
Time in us
2.1750
Replot
Change t
Save current t-point
Save all t-points

A significant advantage of time domain images for short transient pulses is that the Lamb waves do not manifest themselves as they do in CW regimes, allowing for visualization of small-scale surface defects and geometrical details of the radiating surface.

Example 2: *Quick Start for a Flat Transducer*

Path: “xDDx\examples\quick_start_flat.m”

The Quick Start Tool can also be used for flat transducer cases. The main difference is that the automatic alignment procedure is not available for this type of transducer, as it does not have a tight focal lobe. The process of manual alignment will be described in Example 11.

General input parameters

Since the transducer does not have a `radiusOfCurvature` parameter, the input variables should be adjusted accordingly:

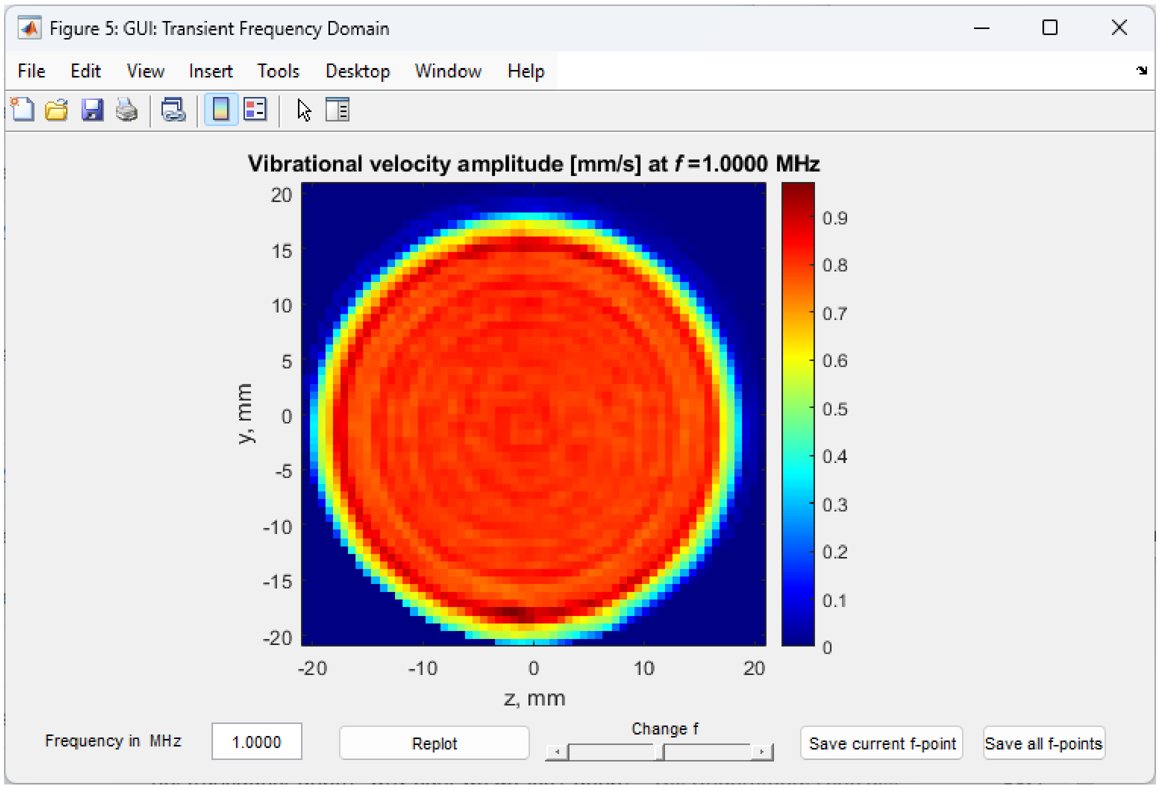
```
frequencyNominal = 1e6; %desired frequency of the output single-frequency hologram in Hz

apertureMin = 38e-3; % characteristic minimum aperture of the transducer in m

apertureMax = 38e-3; % characteristic maximum aperture of the transducer in m (apertureMax = apertureMin for circular transducers)

radiusOfCurvature = []; % (optional) nominal radius of curvature of the transducer in m. Omit this variable or define it as an empty value [] if your transducer is flat
```

Examples of transient reconstruction

```
doTransientCase = true; %true for transient case, false for single-frequency case

doAlignment = false; %set true to perform alignment, or false to perform backprojection with no alignment
```

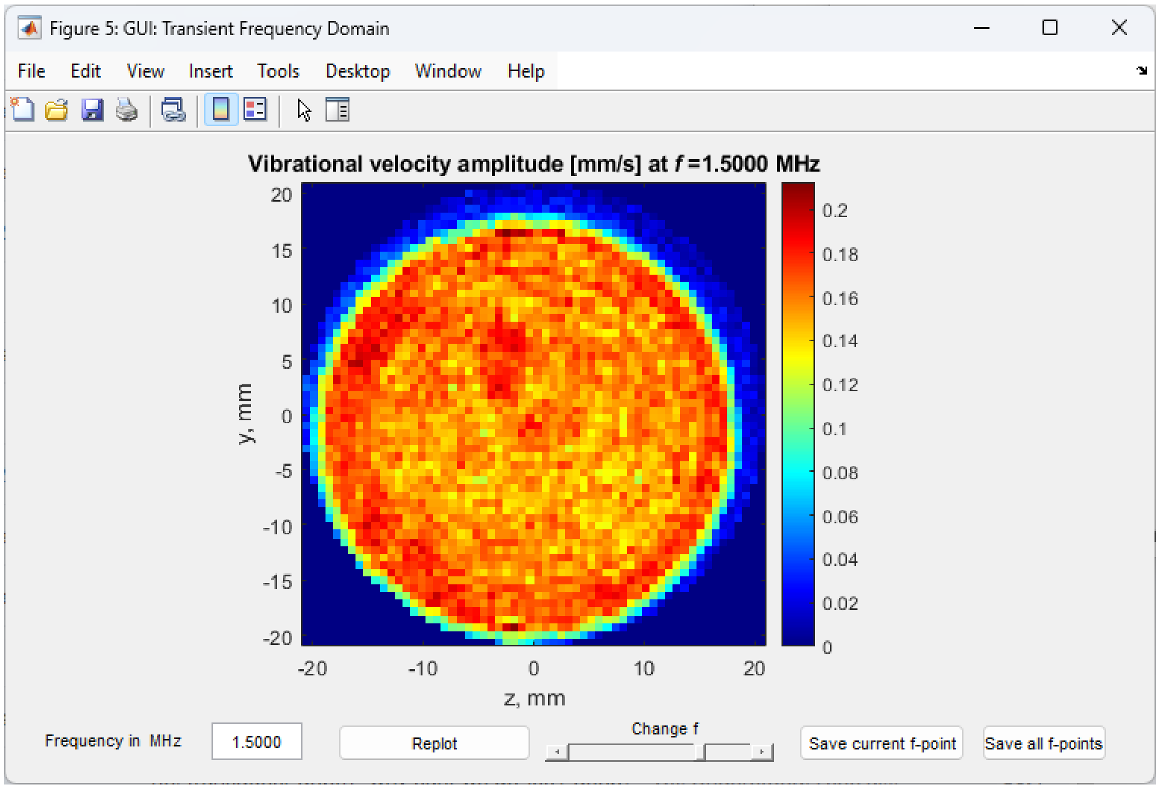
are shown below in frequency domain

Simulation results



Vibrational velocity amplitude [mm/s] at f =1.0000 MHz

Frequency in MHz 1.0000



Vibrational velocity amplitude [mm/s] at f =1.5000 MHz

Frequency in MHz 1.5000

and in time domain

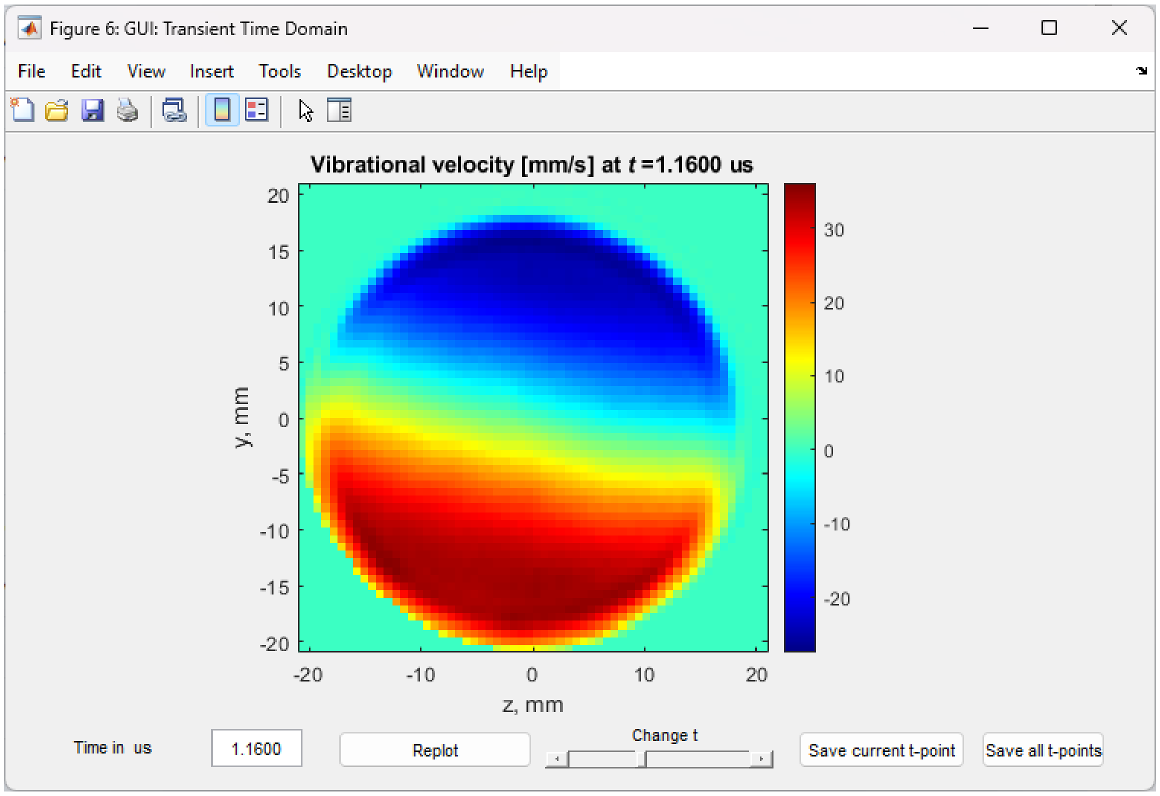


Figure 6: GUI: Transient Time Domain
File Edit View Insert Tools Desktop Window Help
Vibrational velocity [mm/s] at t =1.1600 us
y, mm
z, mm
Time in us
1.1600
Replot
Change t
Save current t-point
Save all t-points

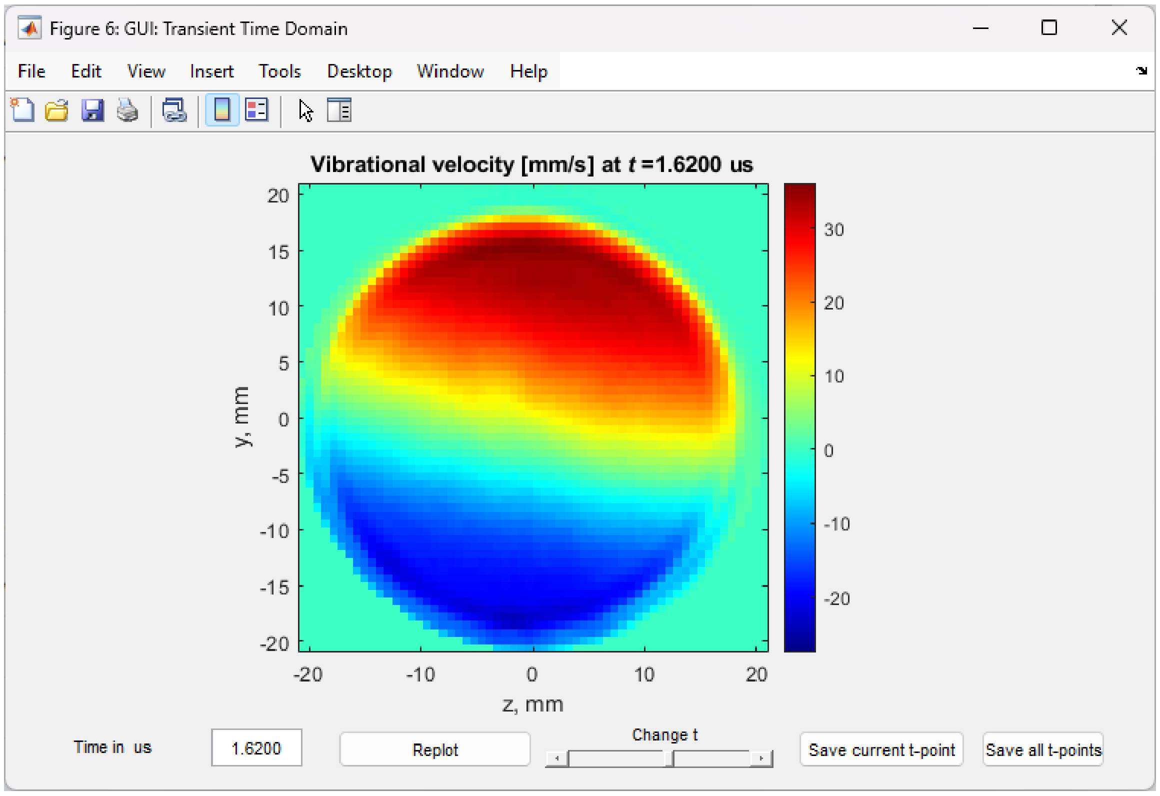


Figure 6: GUI: Transient Time Domain
File Edit View Insert Tools Desktop Window Help
Vibrational velocity [mm/s] at t =1.6200 us
y, mm
z, mm
Time in us
1.6200
Replot
Change t
Save current t-point
Save all t-points

As the alignment has not been performed and the transducer is tilted the fibrational velocity is nonuniform on the reconstruction plane, clearly indicating that the hologram plane is tilted relative to the acoustical axis.

ADVANCED TRANSDUCER DIAGNOSTICS

Even though the Quick Start Tool can be useful for a brief test in holography-based transducer diagnostics, there are methods for performing a deeper analysis of transducer performance and gaining a clearer understanding of potential defects. The tools presented below enable this advanced analysis. All examples below will be given for the case of a focused transducer (Fig. 10a). The tools and corresponding examples are divided into three sections based on the procedures they represent.

Data Preparation

The first set of tools can be used to analyze transient holography data prior to projection and extract single-frequency components from the transient scan.

Power Spectrum Tool

This tool is designed to calculate and plot a power curve for different spectral components of a transient hologram. It helps determine the maximum significant frequency for transient hologram post-processing. The tool uses an angular spectrum-based method for power calculation in a single-frequency planar scan [Sapozhnikov 2013].

Example 3: *Spectral Power Curve for a Transient Hologram*

Path: “xDDx\examples\data_preparation_tools\power_of_hologram.m”

General input parameters

- A MAT file containing two structures `HologramTr` (see page 13) and `Medium` (see page 11).

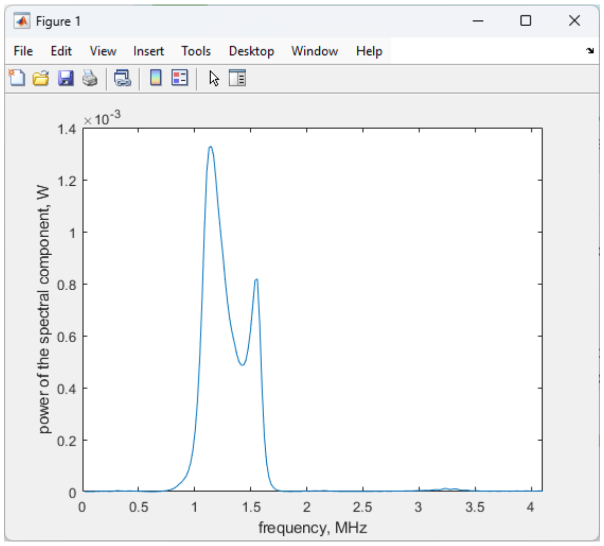
```
inputParametersFileName =
'..\..\data_for_examples\sector\holo_data_sector_transient.mat';   %path to input data with transient hologram and medium parameters
```

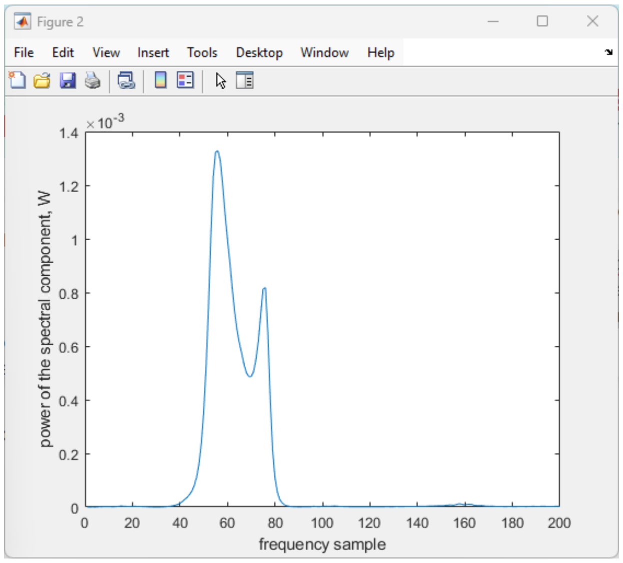
Note that for the angular spectrum calculations, `HologramTr.xGrid` should be a 2D matrix, and `HologramTr.pressureWaveforms` should be 3D: `size(pressureWaveforms) = [size(xGrid, 1) size(xGrid, 2) length(time)]`

- Spatial step parameters:

```
stepDim1 = 0.4e-3; %step size in m along the 1st dimension of the HologramTr.pressureWaveforms 3D matrix

stepDim2 = 0.4e-3; %step size in m along the 2nd dimension of the HologramTr.pressureWaveforms 3D matrix
```

Simulation output





The output graphs display spectral power distributions in two ways: by frequency in MHz and by FFT frequency samples, which are determined based on the time window `[HologramTr.time(1) HologramTr.time(end)]`. This distribution is useful for future spectral analysis, as it indicates the maximum significant frequency. In the case considered, it can be assumed that frequency samples smaller than 30 and greater than 90 do not contain significant beam energy. This provides frequency sample range parameters

```
minFrequencySample = 30; %frequency samples ranging (minFrequencySample : maxFrequencySample)*frequencyStep, where frequencyStep = 1 / "time window for HologramTr.time"
```

```
maxFrequencySample = 90; %set 'auto' to determine the values automatically
based on the powerSignificanceLevel
```

These parameters will be used later in transient simulations.

Single-Frequency Extraction Tool

Example 4: *Single-Frequency Extraction*

Path: “xDDx\examples\data_preparation_tools\extract_sf_from_transient.m”

This tool reads transient hologram data from a file, specified by `inputParametersFileName`, and extracts a single-frequency component at a user-specified frequency, `frequencyNominal`. The output frequency is as close to the desired `frequencyNominal` as the spectral sampling of the discrete signal allows. The result is saved as a MAT file `outFilename`, which can be used for automatic alignment tools.

General input parameters

- A MAT file containing two structures `HologramTr` (see page 18) and `Medium` (see page 11)

```
inputParametersFileName =

'..\..\data_for_examples\sector\holo_data_sector_transient.mat';    %path    to
input data with transient hologram and medium parameters
```

- Output filename:

```
outFilename = 'holo_data_sector_sf.mat'; %output data with transducer, single-
frequency hologram, and medium parameters
```

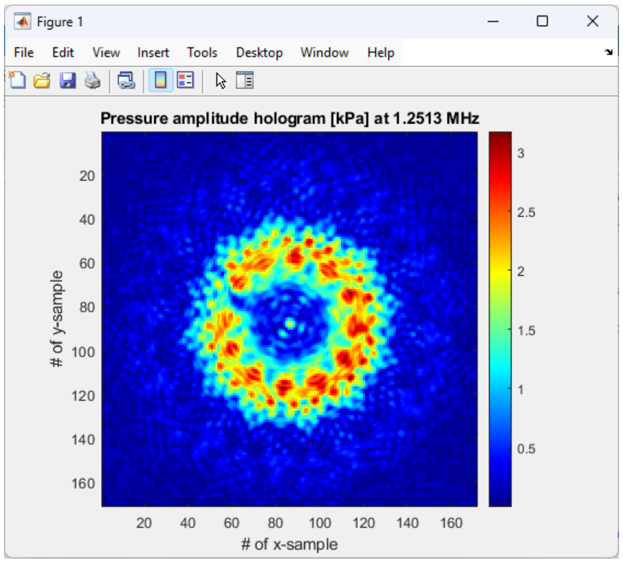
- Nominal transducer parameters:

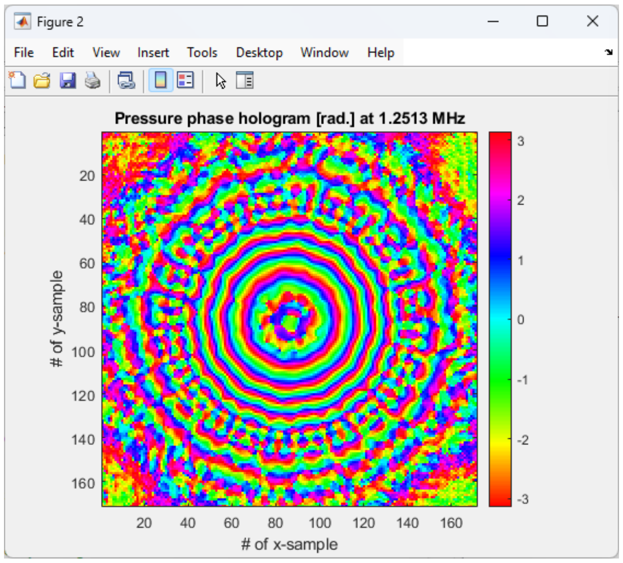
```
frequencyNominal = 1.25e6; %desired frequency of the output single-frequency
hologram in Hz

apertureMin = 87e-3; %characteristic minimum aperture of the transducer in m

apertureMax = 87e-3; %characteristic maximum aperture of the transducer in m
(apertureMax = apertureMin for circular transducers)

radiusOfCurvature = 87e-3; %(optional) nominal radius of curvature of the
transducer in m. Omit this variable or define it as an empty value [] if your
transducer is flat
```

-Exponent convention parameter. The default value of +1 can be used for all xDDx tools.

```
expSign = +1; %+1 or -1, depending on the exponent sign convention exp(+ 1i *
omega * t) or exp(- 1i * omega * t), +1 by default.
```

Simulation output





Two output windows show the amplitude and phase distributions at the hologram plane for the frequency `frequencyNominal`. Users can examine the amplitude distribution to ensure that the holography scan captures the entire beam.
The MAT file with the output results, `outFilename` is saved in the same directory and contains three structures: `Geometry` (see page 10), `HologramSf` (see page 16), and `Medium` (see page 11). The `Geometry` structure is included to provide an initial approximation of the focal lobe position and shape for the automated alignment procedure.

Simple Projections

Simple Projection tools are applicable to all types of flat and spherically shaped transducers and are useful for back and forward projections of holograms without alignment.

Simple Back-Projection Tool

Example 5: *Simple Back-Projection for Single-Frequency*

Path: “xDDx\examples\simple_projection_tools\bp_sf.m”

This tool back-projects a measured single-frequency hologram, assuming it is perpendicular to the transducer's axis of symmetry. The output shows the complex velocity amplitude and phase at the surface of the transducer, plotted in two figures.

General input parameters

- A MAT or XLSX file containing three structures: `Geometry` (see page 10), `HologramSf` (see page 16), and `Medium` (see page 11). This file can be generated using the Single-Frequency Extraction Tool.

```
inputParametersFileName =

'..\..\data_for_examples\sector\holo_data_sector_sf.mat'; %path to input data with transducer, single-frequency hologram, and medium parameters in '.mat' or '.xlsx' format
```

An alternative XLSX way is

```
inputParametersFileName =

'..\..\data_for_examples\sector\holo_data_sector_sf.xlsx'; %path to input data with transducer, single-frequency hologram, and medium parameters in '.mat' or '.xlsx' format
```

- Grid parameters for the surface of the transducer

```
nxSource = 201; %number of points for the Cartesian grid of the source along the x- and y-dimension

nySource = 201;

dxSource = 0.5e-3; %x and y grid step in m

dySource = 0.5e-3;
```

A symmetrical rectangular grid (Fig. 6 a, c) is generated based on the input parameters, covering the range from – `(nxSource-1)*dxSource /2` to `(nxSource-1)*dxSource /2` with a step size of `dxSource`. The same pattern applies in the y-direction.

For example, the x- and y-grid for the case considered spans from -100 mm to 100 mm with a step size of 0.5 mm.

- Reserved size for the circular mask

```
apertureReserve = 0.1; %fraction of the aperture used to increase the nominal output window
```

As shown in (Fig. 6d, e), a circular mask is automatically applied to the rectangular simulation region using the formula

```
radiusMax = 0.5*(1+apertureReserve)*Geometry.apertureMax
```

to improve simulation speed.

The default value of `apertureReserve = 0.1` is suitable for most cases.

- GPU threads

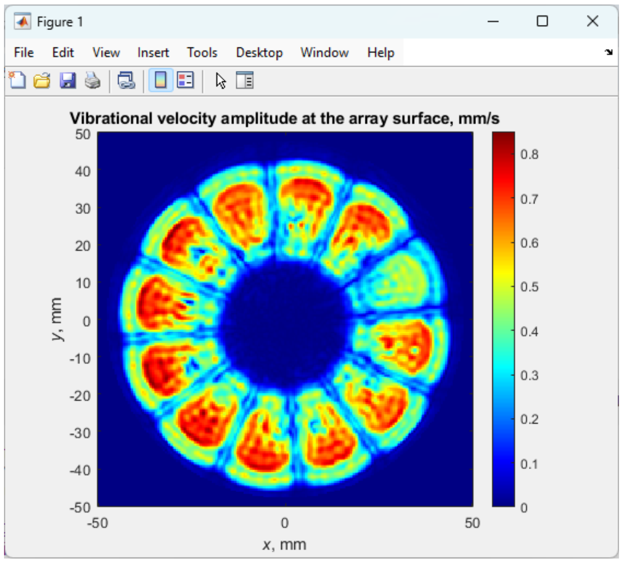
```
ServiceParameters.threadsPerBlockGPU = 128; %number of threads per block for GPU (if applicable)
```

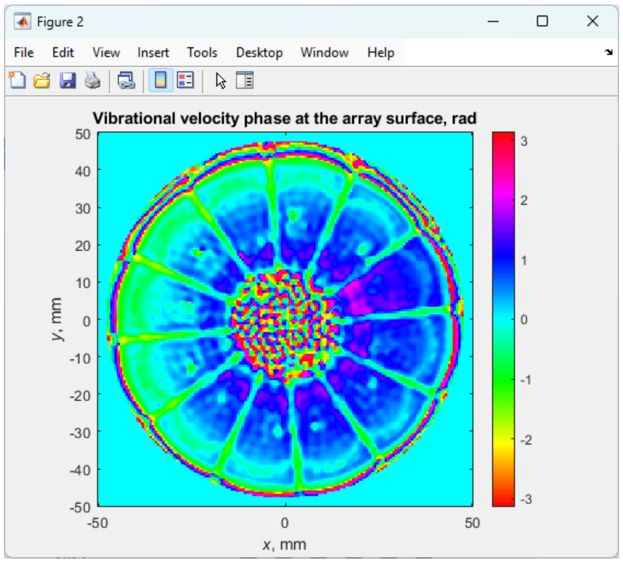
This parameter is ignored if the CPU mode is chosen. The default value of 128 provides good performance for most GPU and simulation cases.

- Save back-projection results:

```
saveTransducer = true; %true to save the boundary condition for the transducer at HologramSf.frequency as a 'TransducerSf' structure
```

Simulation output





Note that the amplitude and phase are set to zero outside the circle defined by the `apertureReserve` parameter. The results are identical to those obtained with the Quick Start Tool when no automatic alignment is used. However, the Simple Back-Projection Tool allows for control over the source surface grid, making it useful for more precise transducer diagnostics.

Details

In some cases, when users have pre-processed a single-frequency hologram without using the xDDx Single-Frequency Extraction Tool, determining the correct exponent sign can be challenging. The Simple Back Projection Tool can assist with a quick check. Users can simulate two cases: one with `expSign = 1` and another with `expSign = -1`. The correct exponent sign will produce an output pattern that reflects the expected transducer surface shape, while the incorrect one will result in a completely unrealistic distribution. Examples of “correct” and “incorrect” results for the case considered are shown below.

`expSign = 1` (**correct**) `expSign = -1` (**wrong**)

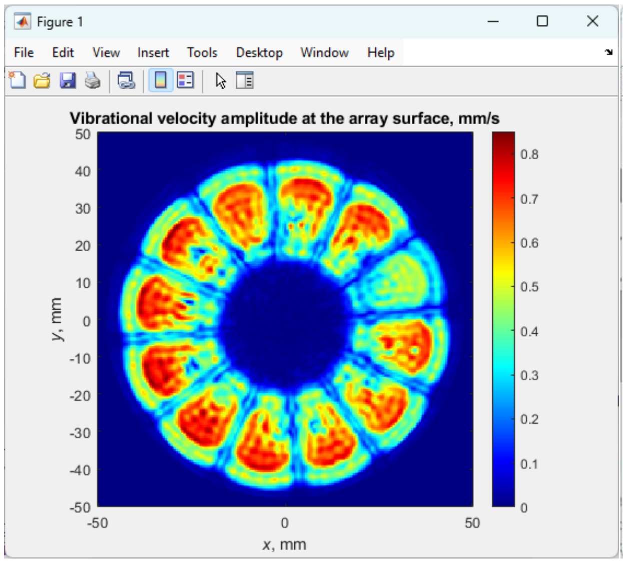


Figure 1
File Edit View Insert Tools Desktop Window Help
Vibrational velocity amplitude at the array surface, mm/s
y, mm
x, mm
50 40 30 20 10 0 -10 -20 -30 -40 -50
-50 0 50
0.8 0.7 0.6 0.5 0.4 0.3 0.2 0.1 0

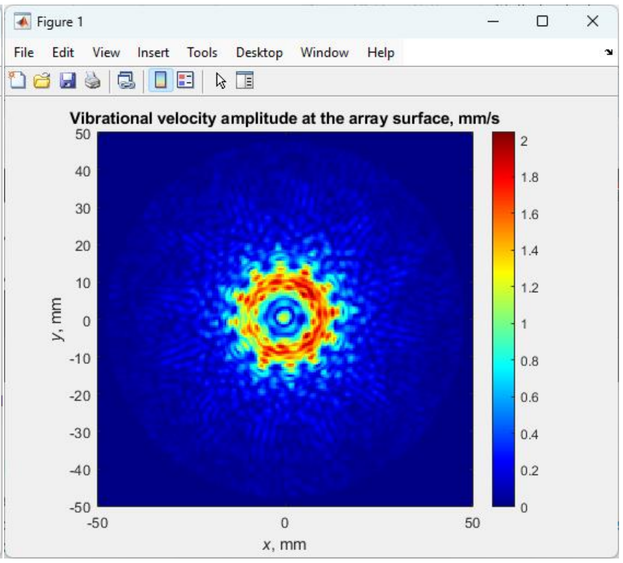



Figure 1
File Edit View Insert Tools Desktop Window Help
Vibrational velocity amplitude at the array surface, mm/s
y, mm
x, mm
50 40 30 20 10 0 -10 -20 -30 -40 -50
-50 0 50
2 1.8 1.6 1.4 1.2 1 0.8 0.6 0.4 0.2 0

Single-Frequency Forward-Projection Tool

This tool is designed to forward-project a measured single-frequency hologram to obtain a 3D field within a specified spatial region. The output acoustic pressure amplitude can be plotted in 3D, 2D, 1D, or 0D with an option for slice-by-slice representation.

Example 6: *Forward-Projection for Single-Frequency*

Path: “xDDx\examples\simple_projection_tools\fp_sf.m”

General input parameters

- A MAT or XLSX file containing two structures: `HologramSf` (see page 16), and `Medium` (see page 11). This file can be generated using the Single-Frequency Extraction Tool.

```
inputParametersFileName =

'..\..\data_for_examples\sector\holo_data_sector_sf.mat'; %path to input data with transducer, single-frequency hologram, and medium parameters in '.mat' or '.xlsx' format
```

An alternative XLSX way is

```
inputParametersFileName =

'..\..\data_for_examples\sector\holo_data_sector_sf.xlsx'; %path to input data with transducer, single-frequency hologram, and medium parameters in '.mat' or '.xlsx' format
```

- Rectangular window for field simulation

The parameters listed below specify boundaries and steps for 3D, 2D, 1D, or 0D (single point) simulation regions. The tool automatically detects the region type and plots the result in different ways based on the number of dimensions.

```
xFieldBegin = -10e-3; %x, y, and z limits in m for the rectangular simulation region. Set "Begin" equal to "End" for specific coordinates to reduce the number of dimensions.

xFieldEnd   =  10e-3;

yFieldBegin = -10e-3;

yFieldEnd   =  10e-3;

zFieldBegin = 70e-3;

zFieldEnd   = 110e-3;


dxField = 0.25e-3; %x, y, and z grid step in m for the rectangular simulation region. Steps for singleton dimensions (where "Begin" = "End") are ignored and can be omitted.

dyField = 0.25e-3;

dzField = 0.5e-3;
```

- Isosurface parameters for 3D (ignored for 2D/1D/0D cases.)

For 3D output regions, the results are presented as set of iso-surfaces of points with equal pressure amplitude levels normalized by the maximum pressure amplitude. The levels are specified in the variable `levelArray`, and the transparency of each iso-surface is controlled by `transparencyArray`. Users can set an arbitrary number of iso-surfaces. The `'jet'` colormap, ranging from 0 to 1, is used by default to assign colors to the iso-surfaces in `levelArray`.

```
levelArray = [0.8 0.5 0.1]; %isolevels related to the pressure maximum for extracting the isosurface of the simulated 3D field

transparencyArray = [0.6 0.5 0.1]; %transparency of the isosurfaces for the levels from levelArray
```

Simulation output

- Iso-surface representation of the field:

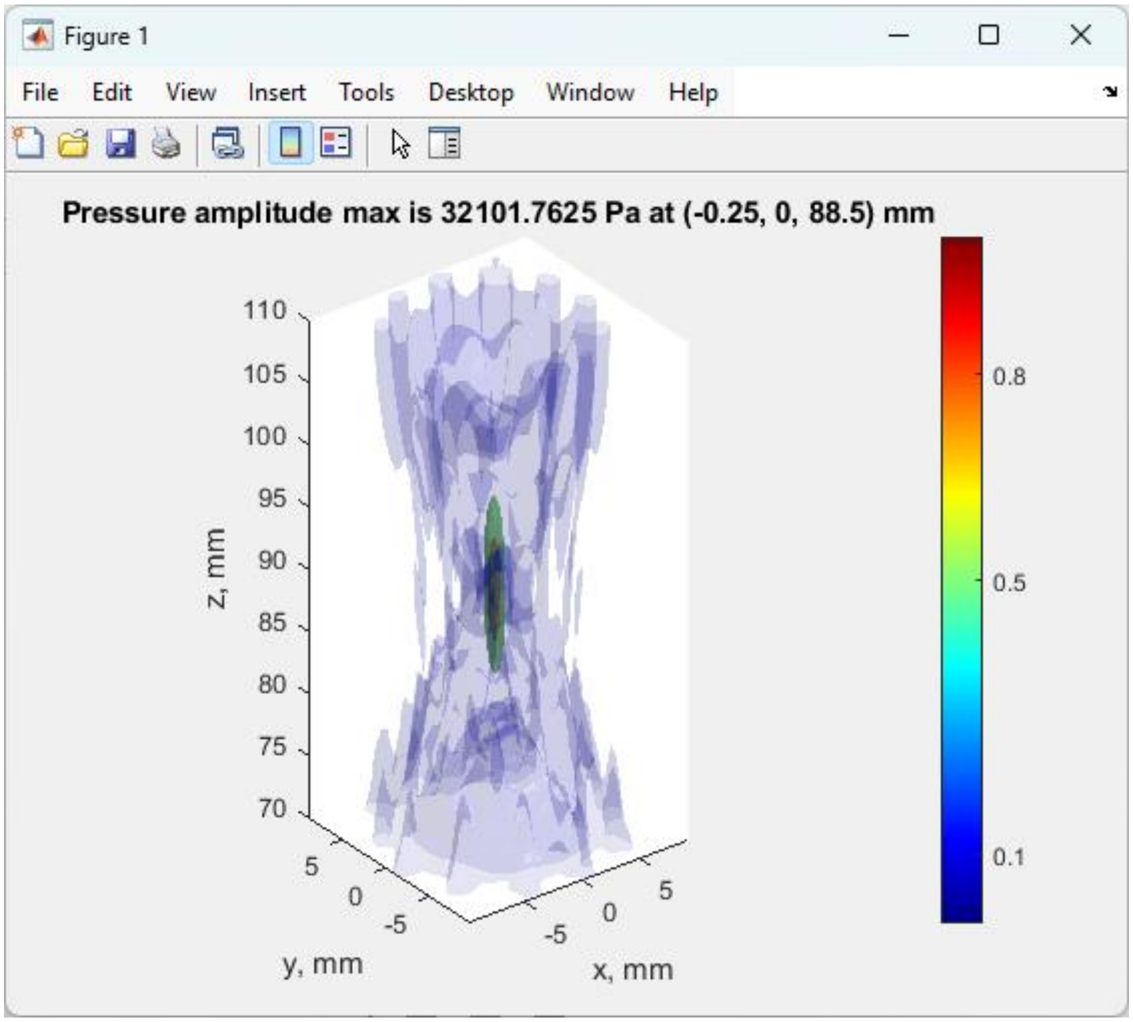


Figure 1
File Edit View Insert Tools Desktop Window Help
Pressure amplitude max is 32101.7625 Pa at (-0.25, 0, 88.5) mm
110
105
100
95
90
85
80
75
70
z, mm
5
0
-5
y, mm
-5
0
5
x, mm
0.8
0.5
0.1

The maximum pressure amplitude and its position are indicated in the plot title. Note that the focal lobe may be shifted and rotated due to hologram misalignment.

- Slice-by-slice representation of the 3D field

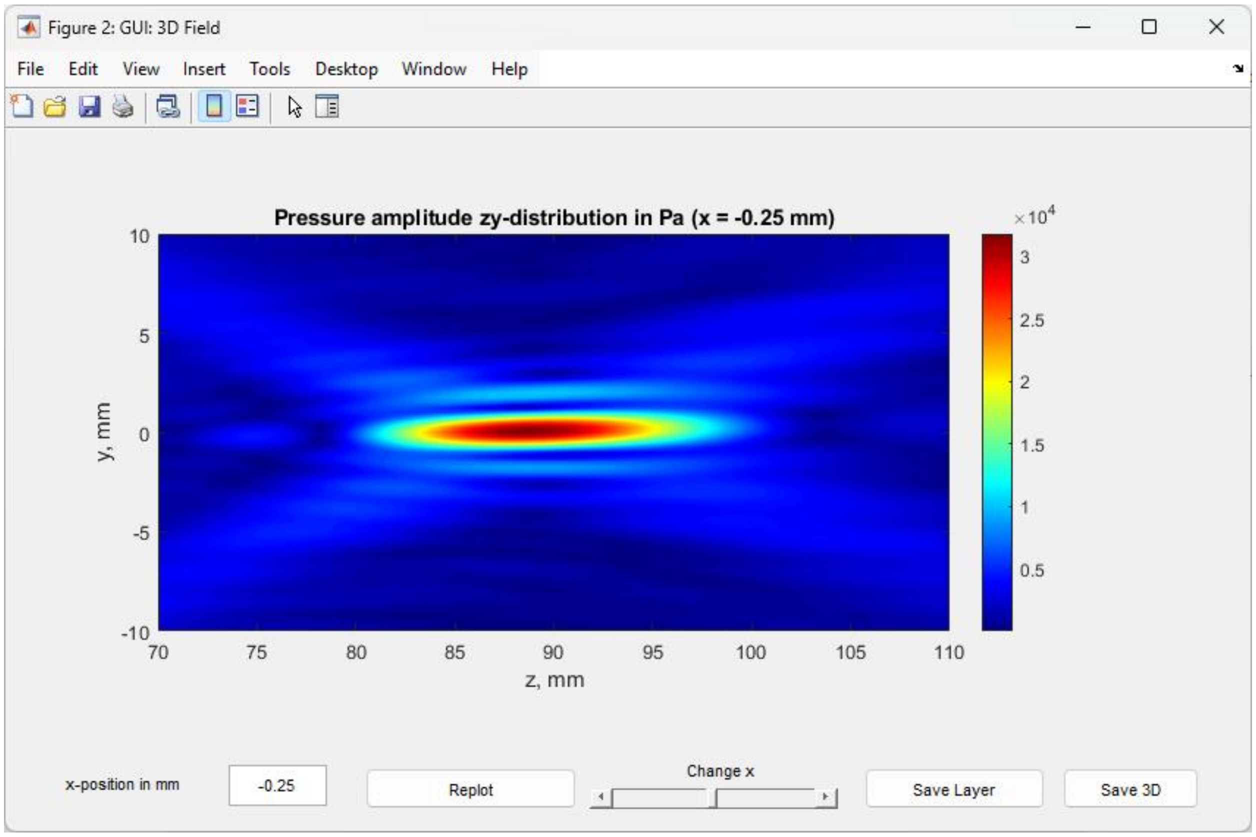


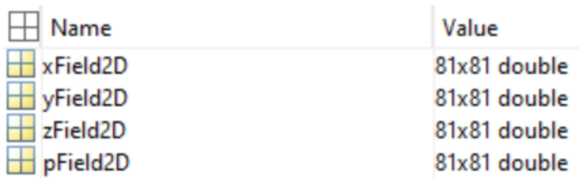
Figure 2: GUI: 3D Field
File Edit View Insert Tools Desktop Window Help
Pressure amplitude zy-distribution in Pa (x = -0.25 mm)
y, mm
z, mm
×10^4
x-position in mm
-0.25
Replot
Change x
Save Layer
Save 3D

The window with GUI controls displays the pressure amplitude in the *zy*-plane for different *x* positions. The default *x*-position is set at the pressure maximum but can be changed using either the "Change x" slider or the "x-position in mm" text box.

Details

The simulation results can be saved for future post-processing in two modes: the current single layer or the entire 3D field.

The "Save Layer" button saves the variables for the current layer:



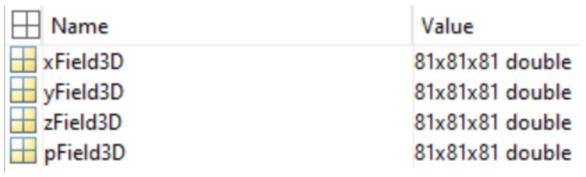
Name	Value
xField2D	81x81 double
yField2D	81x81 double
zField2D	81x81 double
pField2D	81x81 double

In this case, all elements of the `xField2D` matrix are identical and represent the x-position of the slice. The result can be displayed using the `contourf` function:

```
figure;
```

```
contourf(zField2D, yField2D, abs(pField2D));

axis equal;
```

The “Save 3D” button saves all points in the simulation region as 3D matrices:



Name	Value
xField3D	81x81x81 double
yField3D	81x81x81 double
zField3D	81x81x81 double
pField3D	81x81x81 double

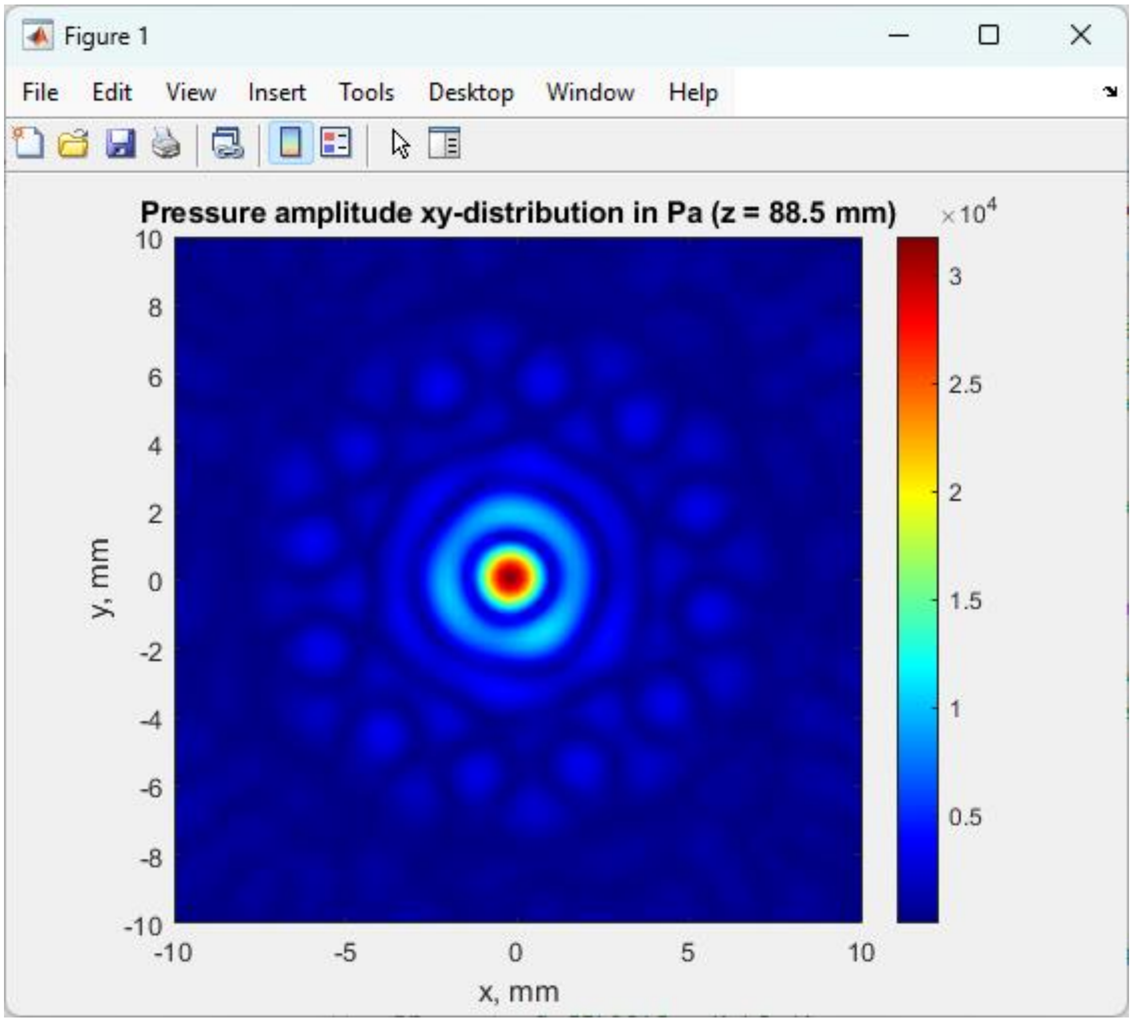
This data can be visualized using the `isosurface` function. The example below shows how to plot an iso-surface for `isoLevel = 0.8`:

```
figure;

isoLevel = 0.8; %select the iso-level

pFieldMax = max(abs(pField3D(:))); %find pressure max

pField3DNormalized = abs(pField3D) / pFieldMax; %normalize pressure

isosurface(xField3D, yField3D, zField3D, abs(pField3DNormalized), isoLevel);
%plot results

axis equal;
```

Input parameters can be adjusted to perform a 2D simulation. The example below simulates the field for an *xy*-slice at a *z*-position of 88.5 mm:

```
xFieldBegin = -10e-3; %m

xFieldEnd   =  10e-3;

yFieldBegin = -10e-3;

yFieldEnd   =  10e-3;

zFieldBegin = 88.5e-3;

zFieldEnd   = 88.5e-3;


dxField = 0.25e-3;  %dzField is omitted

dyField = 0.25e-3;
```

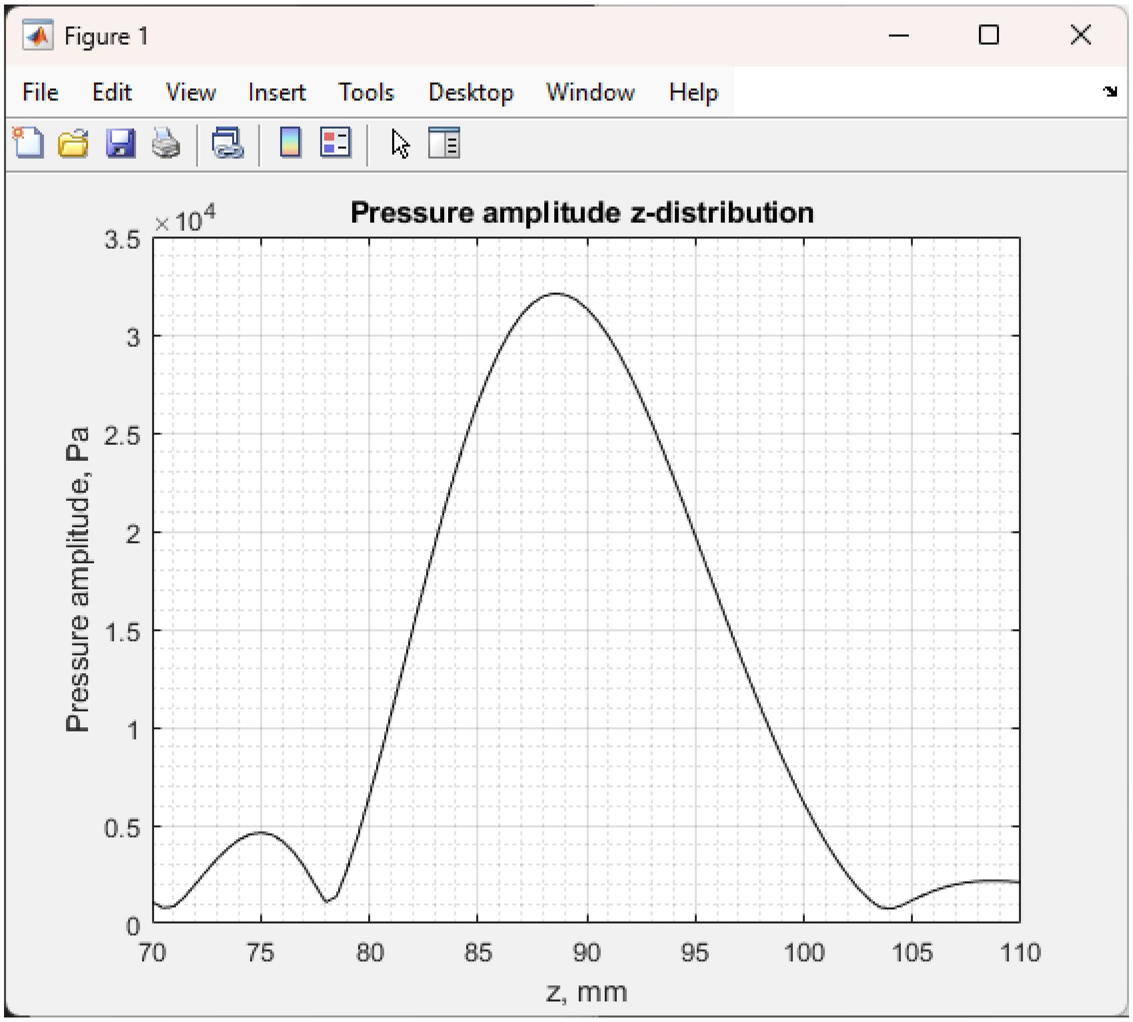
The simulation output in this case is a simple 2D distribution:



The input can be further simplified for a 1D case:

```
 xFieldBegin = -0.25e-3; %m

xFieldEnd    = -0.25e-3;

yFieldBegin = 0e-3;

yFieldEnd    = 0e-3;

zFieldBegin = 70e-3;

zFieldEnd    = 110e-3;


dzField = 0.5e-3; %dxField and dyField are omitted
```

where the output is displayed as a 1D plot



Finally, the simplest 0D case simulates the field at a single point

```
xFieldBegin = -0.25e-3; %m
xFieldEnd   = -0.25e-3;
yFieldBegin = 0e-3;
yFieldEnd   = 0e-3;
zFieldBegin = 88.5e-3;
zFieldEnd   = 88.5e-3;
%dxField, dyField, and dzField are omitted
```

with the output displayed directly in the MATLAB/Octave command line:

```
Pressure amplitude at x = -0.25 mm, y = 0 mm, z = 88.5 mm is 32101.7625 Pa
```

In all cases, the simulation results are stored in the following variables

```
xField3D, yField3D, zField3D, pField3D
xField2D, yField2D, zField2D, pField2D
xField1D, yField1D, zField1D, pField1D
xField0D, yField0D, zField0D, pField0D
```

These variables can be accessed from the MATLAB/Octave workspace after the simulation.

If the logical flag `saveTransducer` is set to `true`, the `TransducerSf` structure (see page 19) for the reconstructed vibrational pattern will be saved in the same directory, with the filename formatted to reflect the simulation completion date and time as:

“transducer_yyyy_mm_dd_hh_mm_ss.mat”

Transient Forward-Projection Tool

This tool is designed for the forward projection of a transient hologram to obtain the pressure waveform at user-defined points.

Example 7: *Forward-Projection for Transient Signal*

Path: “xDDx\examples\simple_projection_tools\fp_transient.m”

General parameters

- A MAT file containing two structures `HologramTr` (see page 13) and `Medium` (see page 11)

```
inputParametersFileName =
'..\..\data_for_examples\sector\holo_data_sector_transient.mat';    %path    to
input data with transient hologram and medium parameters
```

- Vectors specifying the coordinates of the points of interest

```
xFieId = [0; -0.25e-3; 1e-3]; %x, y, and z coordinates in m of the field points
of interest

yField = [0; 0; 1e-3];

zField = [88e-3; 88.5e-3; 90e-3];
```

In this example, the field will be simulated at three (*x*, *y*, *z*) points: (0, 0, 88) mm, (–0.25, 0, 88.5) mm, and (1, 1, 90) mm. The second point corresponds to the pressure maximum, obtained in Example 6. An arbitrary number of points can be set by a user.

- Frequency-domain parameters. Range of frequency samples for the transient hologram projection

```
minFrequencySample = 30; %frequency samples ranging (minFrequencySample :
maxFrequencySample)*frequencyStep, where frequencyStep = 1 / "time window for
HologramTr.time"

maxFrequencySample = 90; %set 'auto' to determine the values automatically
based on the powerSignificanceLevel
```

This value can be obtained using the Power Spectrum Tool or set in the automatic regime. In this regime, band-pass filtering is applied using the condition that the total power of the hologram spectrum at the minimum (`minFrequencySample`) and maximum (`maxFrequencySample`) frequency is less than a given level (`powerSignificanceLevel = 0.01` by default) of the maximum power among all spectral components.

```
powerSignificanceLevel = 0.01; %minimum significant power level of a spectral
component relative to the maximum power (used in 'auto' regime)

minFrequencySample = 'auto'; %frequency samples ranging
(minFrequencySample:maxFrequencySample)*frequencyStep, where frequencyStep =
1 / "time window for HologramTr.time"

maxFrequencySample = 'auto'; %set 'auto' to determine the values
automatically based on the powerSignificanceLevel
```

Simulation output

Results are displayed as separate plots of the waveforms for each simulation point

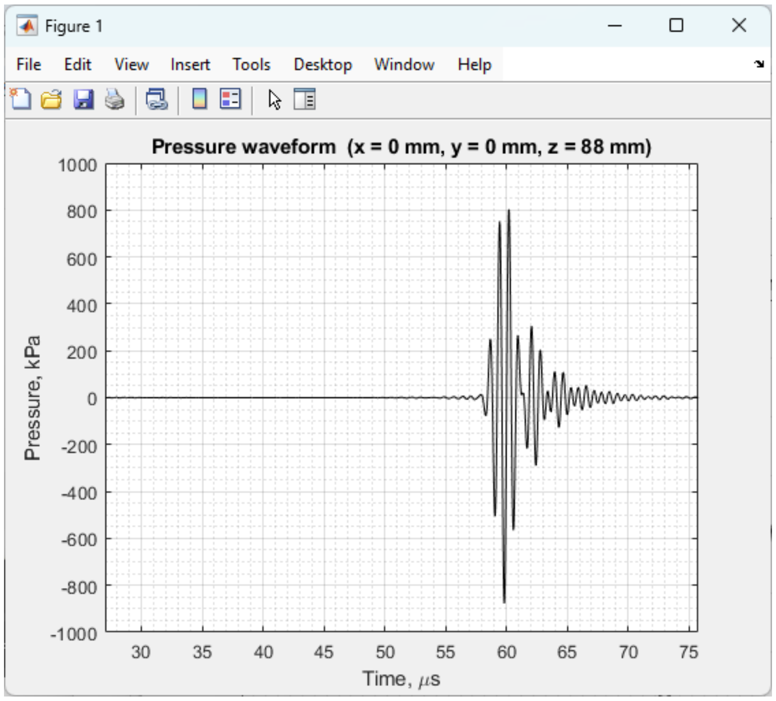


Figure 1
File Edit View Insert Tools Desktop Window Help
Pressure waveform (x = 0 mm, y = 0 mm, z = 88 mm)
Pressure, kPa
Time, μs
1000
800
600
400
200
0
-200
-400
-600
-800
-1000
30
35
40
45
50
55
60
65
70
75

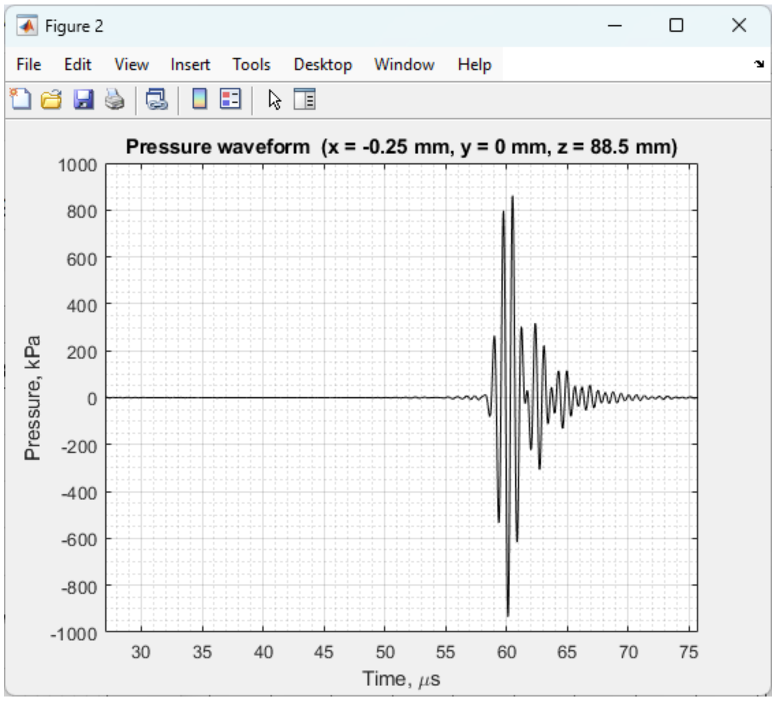
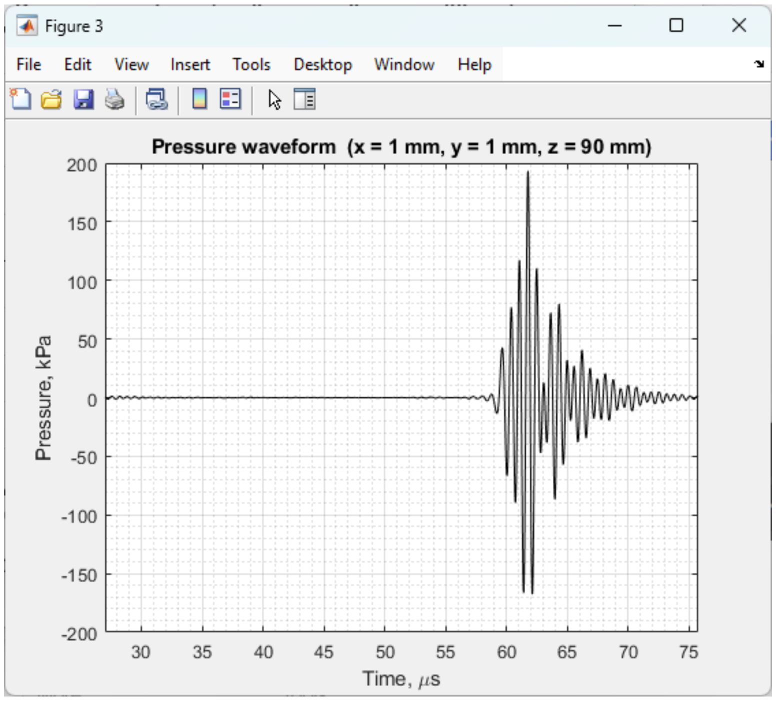


Figure 2
File Edit View Insert Tools Desktop Window Help
Pressure waveform (x = -0.25 mm, y = 0 mm, z = 88.5 mm)
Pressure, kPa
Time, μs
1000
800
600
400
200
0
-200
-400
-600
-800
-1000
30
35
40
45
50
55
60
65
70
75



Alignment Tools

These tools can be used to align holograms, back-project the aligned holograms, and reconstruct the distribution at the actual transducer surface.

Automatic Alignment Tool

The Automatic Alignment Tool is designed for focused transducers shaped like a spherical cup with a known nominal radius of curvature.

Assumptions:
- The transducer's field has a symmetrical focal lobe to find the axis of symmetry
- The focal maximum is located at the center of curvature (true for strongly focused transducers)

This tool rotates and shifts a measured single-frequency hologram so that the aligned hologram is perpendicular to the transducer’s axis of symmetry, then back-projects it to the transducer surface. The output GUI figure allows adjustment of the transducer’s radius of curvature with re-backprojection of the field. The actual radius of curvature corresponds to the sharpest backprojected image. GUI controls are available for saving results. Technical details of the automatic alignment algorithm are discussed in [Rosnitskiy 2025].

Example 8: *Automatic Alignment for Single-Frequency Hologram*

Path: “xDDx\examples\alignment_tools\bp_auto_alignment_sf.m”

General input parameters

- A MAT or XLSX file containing three structures: `Geometry` (see page 10), `HologramSf` (see page 16), and `Medium` (see page 11). This file can be generated using the Single-Frequency Extraction Tool.

```
inputParametersFileName =

'..\..\data_for_examples\sector\holo_data_sector_sf.mat'; %path to input data
with transducer, single-frequency hologram, and medium parameters in '.mat' or
'.xlsx' format
```

An alternative XLSX way is

```
inputParametersFileName =

'..\..\data_for_examples\sector\holo_data_sector_sf.xlsx'; %path to input data
with transducer, single-frequency hologram, and medium parameters in '.mat' or
'.xlsx' format
```

- Grid parameters for the surface of the transducer

```
nxSource = 201; %number of points for the Cartesian grid of the source along
the x- and y-dimension

nySource = 201;

dxSource = 0.5e-3; %x and y grid step in m

dySource = 0.5e-3;

xZeroPhase = -25e-3; %(optional parameter) x-coordinate of the zero-phase point
on the surface of the source

yZeroPhase = -25e-3; %(optional parameter) y-coordinate of the zero-phase point
on the surface of the source
```

The wave phase is always relative to a selected zero point. This tool provides optional grid parameters `xZeroPhase` and `yZeroPhase` to define a zero-phase on the transducer surface. These parameters can be set for visualization convenience and can be omitted. If omitted, the zero-phase point remains unspecified and depends on the propagation distance.

- Focal lobe parameters

```
focalLobeLevel = 0.6; %isolevel related to the pressure maximum for extracting the isosurface of the main focal lobe
```

This parameter is described above in the Quick Start Tool.

Two forward-projection simulations are performed to determine the 3D shape of the focal lobe. One is performed in a large region of interest using a `Coarse` grid step, and another in a smaller region with a `Fine` grid step for improved spatial resolution. Steps are defined in points per wavelength (PPW = "wavelength" / "step"):

```
ppwRadialCoarse3D = 3; %number of points per wavelength (ppw) in the radial direction for identifying the main lobe with a coarse grid step
```

```
ppwAxialCoarse3D = 2; %number of points per wavelength (ppw) in the axial direction for identifying the main lobe with a coarse grid step
```

```
ppwRadialFine3D = 20; %number of points per wavelength (ppw) in the radial direction for identifying the main lobe with a fine grid step
```

```
ppwAxialFine3D = 10; %number of points per wavelength (ppw) in the axial direction for identifying the main lobe with a fine grid step
```

- Technical parameters

```
errorPositioningWl = 10; %expected error of the positioning of the hologram in the axial and radial directions in wavelengths of HologramSf.frequency
```

This parameter represents the expected error in determining the position of the hologram center relative to the acoustical axis.

```
minHoloShiftWl = 4; % minimum shift between the initial and aligned holograms, measured in wavelengths of the central frequency
```

This parameter is illustrated in Fig. 7b and denoted as Δ_{min}. Default values for the focal lobe and technical parameters are generally suitable for most cases.

Simulation output

- The first and second figures show the focal lobes (coarse-grid and fine-grid) along with the ordinary least squares-estimated direction of the acoustic axis. These figures correspond to those described on page 27 (Quick Start Tool).

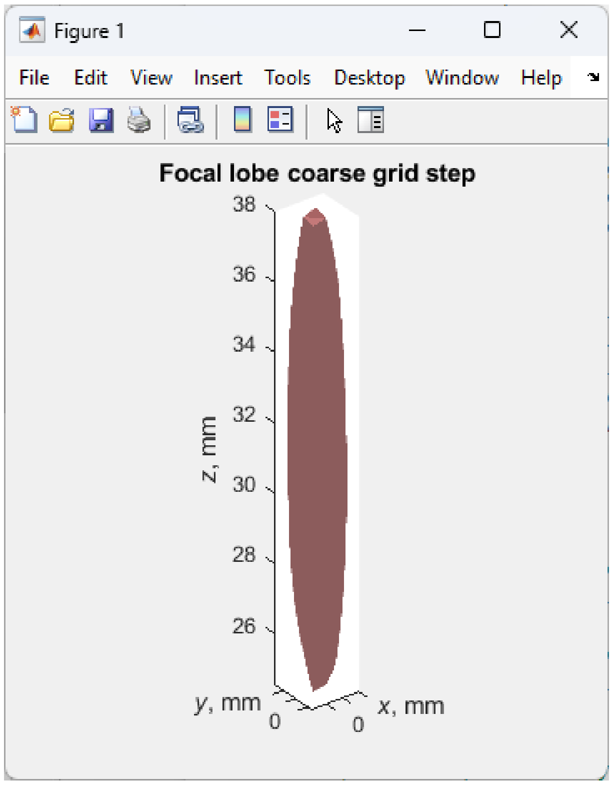


Figure 1
File Edit View Insert Tools Desktop Window Help
Focal lobe coarse grid step
z, mm
38
36
34
32
30
28
26
y, mm
0
0
x, mm

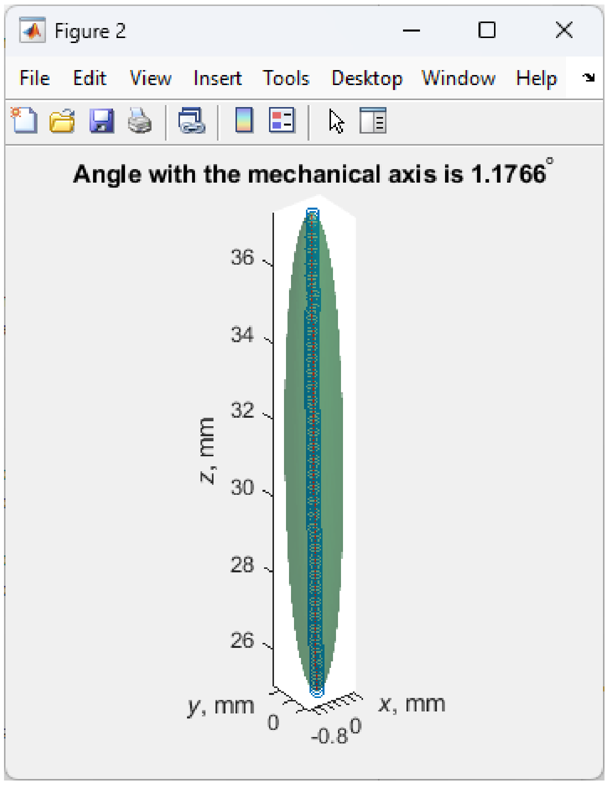


Figure 2
File Edit View Insert Tools Desktop Window Help
Angle with the mechanical axis is 1.1766°
z, mm
36
34
32
30
28
26
y, mm
0
-0.8
0
x, mm

- The next GUI window presents the amplitude and phase distributions after automatic alignment. The edit field labeled "Radius of Curvature in mm", initially contains the nominal value of 87 mm. This value can be adjusted, and the back-projection repeated using the button "Backproject and Replot".

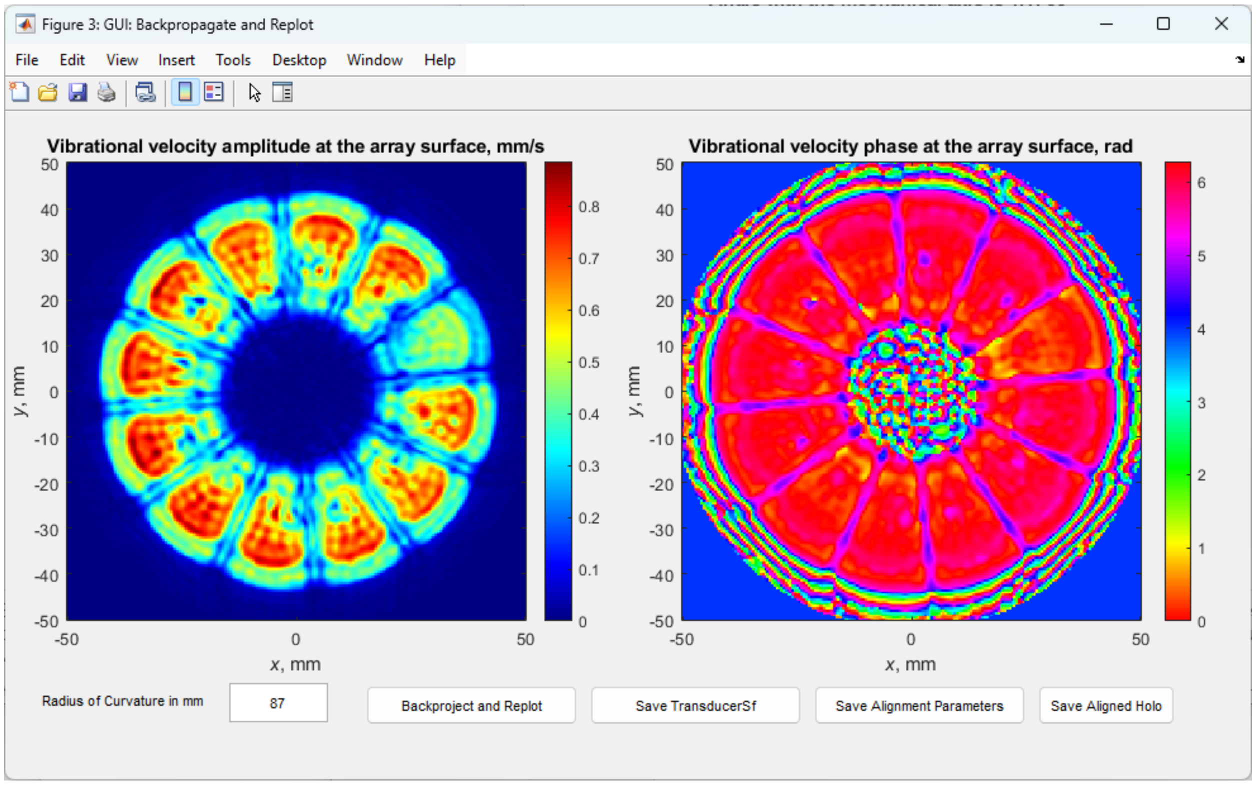


Figure 3: GUI: Backpropagate and Replot
File Edit View Insert Tools Desktop Window Help
Vibrational velocity amplitude at the array surface, mm/s
Vibrational velocity phase at the array surface, rad
y, mm
x, mm
Radius of Curvature in mm
87
Backproject and Replot
Save TransducerSf
Save Alignment Parameters
Save Aligned Holo

Two other buttons allow saving the back-projected vibrational velocity distribution and alignment parameters, respectively.

After alignment, the phase distribution is predominantly uniform, aside from minor local defects caused by lens manufacturing. As noted, the nominal radius of curvature was initially shorter than the actual radius. Using the GUI, this radius was adjusted iteratively to determine the effective value, a process akin to focusing an optical image: the appropriate radius of curvature (Fig. 7b) aligns the reconstruction with the transducer surface and yields the sharpest distribution ("image"). In this case, the optimal radius was found to be 92 mm.

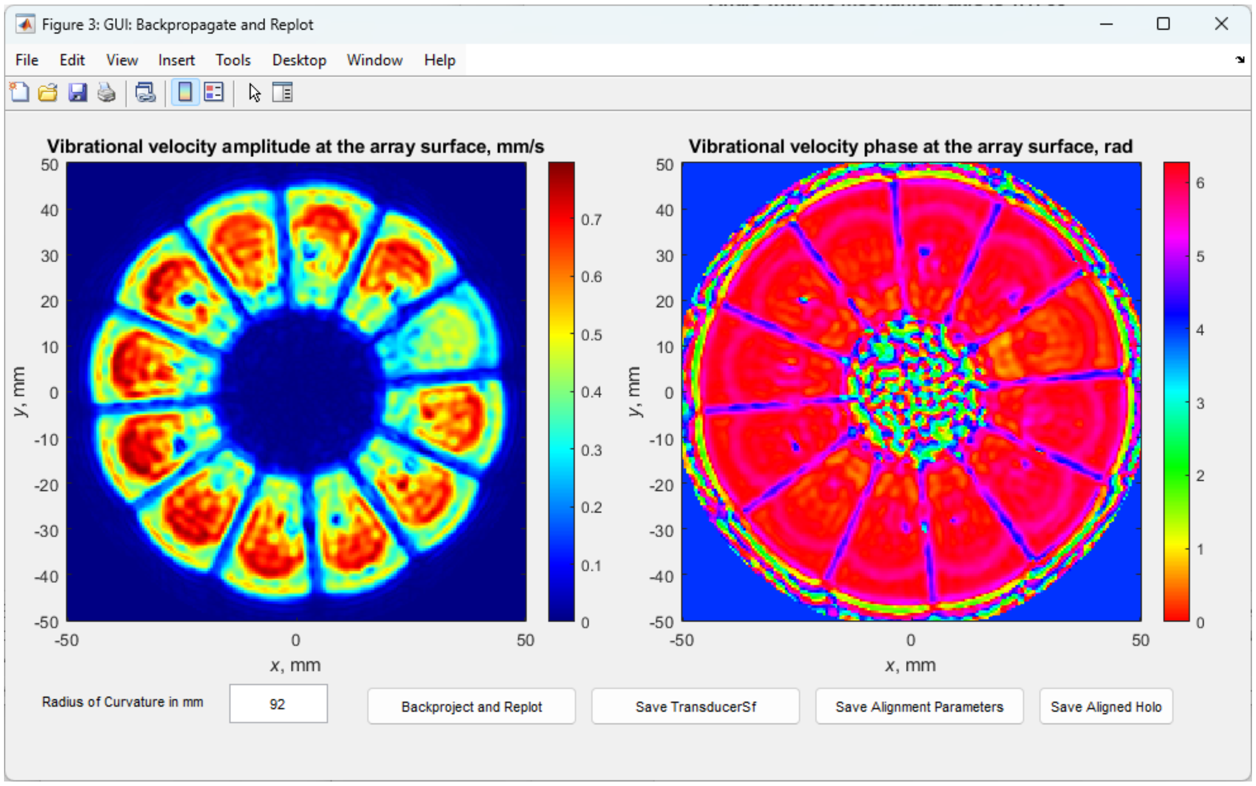
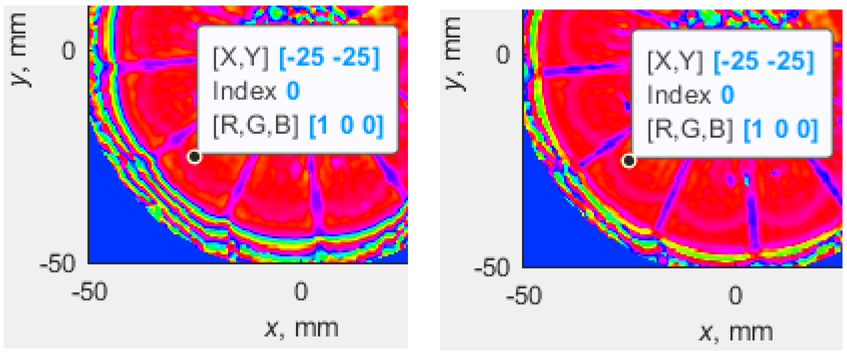


Figure 3: GUI: Backpropagate and Replot
File Edit View Insert Tools Desktop Window Help
Vibrational velocity amplitude at the array surface, mm/s
Vibrational velocity phase at the array surface, rad
y, mm
x, mm
Radius of Curvature in mm
92
Backproject and Replot
Save TransducerSf
Save Alignment Parameters
Save Aligned Holo

The vibrational velocity amplitude distribution is significantly improved, with sharper contours of the elements and clearer gaps between them. The phase distribution remains nearly identical to that of the nominal radius of curvature case, as the wavefront shape generated by the array remains spherical. Note that, in both cases, the phase is zero at $x = y = -25$ mm, as indicated by the `xZeroPhase` and `yZeroPhase` variables.



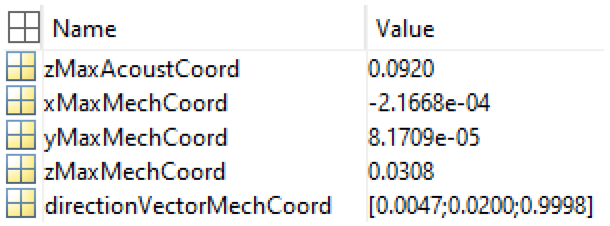
This consistency makes it easier to compare phase distributions across different radius of curvature values.

Details

Other buttons in the GUI allow for saving the simulation results in various formats:

“Save TransducerSf” saves the back-projected vibrational velocity distribution as a `TransducerSf` structure (page 19). This distribution can be used for realistic field simulations of the current transducer, as explained in detail in Example 12, page 88.

“Save Alignment Parameters” saves the parameters that identify misalignment of the hologram. An example for the case considered is shown below:



Name	Value
zMaxAcoustCoord	0.0920
xMaxMechCoord	-2.1668e-04
yMaxMechCoord	8.1709e-05
zMaxMechCoord	0.0308
directionVectorMechCoord	[0.0047;0.0200;0.9998]

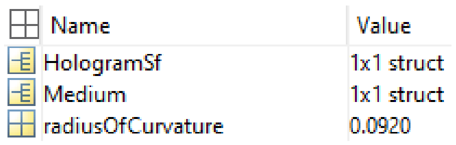
`zMaxAcoustCoord` is the acoustical z-coordinate of the maximum pressure point (marked as “*” in Fig. 7b). For strongly focused transducers, this coordinate equals the radius of curvature.

`xMaxAcoustCoord`, `yMaxAcoustCoord`, `zMaxAcoustCoord` are the mechanical coordinates *x*′, *y*′, *z*′ of this point

`directionVectorMechCoorsd` is the unit vector direction of the transducer's acoustic axis in the mechanical coordinates (denoted $\mathbf{e}_z{}'$ in Fig. 7b).

These parameters allow other tools to perform pre-defined hologram alignment. The example file is saved in the example-data folder `'xDDx\data_for_examples\sector\alig_param_sector.mat'`.

“Save Aligned Holo” saves the aligned hologram as a hologram structure `HologramSf`, medium parameters structure `Medium`, and the selected radius of curvature



Name	Value
HologramSf	1x1 struct
Medium	1x1 struct
radiusOfCurvature	0.0920

These saved parameters are useful for future projections of the aligned hologram.

Pre-Defined Alignment Tool: Single Frequency

This tool rotates and shifts a measured single-frequency hologram so that the aligned hologram is perpendicular to the axis of symmetry of the transducer and back-project it to its surface. The script reads predetermined position parameters 'alignmentParametersFileName' from a file. This file can be either saved using the Automatic Alignment Tool or constructed manually (see Manual Alignment section below).

***Example 9:** Pre-Defined Alignment for Single-Frequency Hologram*

Path: “xDDx\examples\alignment_tools\bp_predefined_alignment_sf.m”

General input parameters

- A MAT or XLSX file containing three structures: `Geometry` (see page 10), `HologramSf` (see page 16), and `Medium` (see page 11). This file can be generated using the Single-Frequency Extraction Tool.

```
inputParametersFileName = 
'..\..\data_for_examples\sector\holo_data_sector_sf.mat'; %path to input data with transducer geometry, single-frequency hologram, and medium parameters in '.mat' or '.xlsx' format
```

An alternative XLSX way is

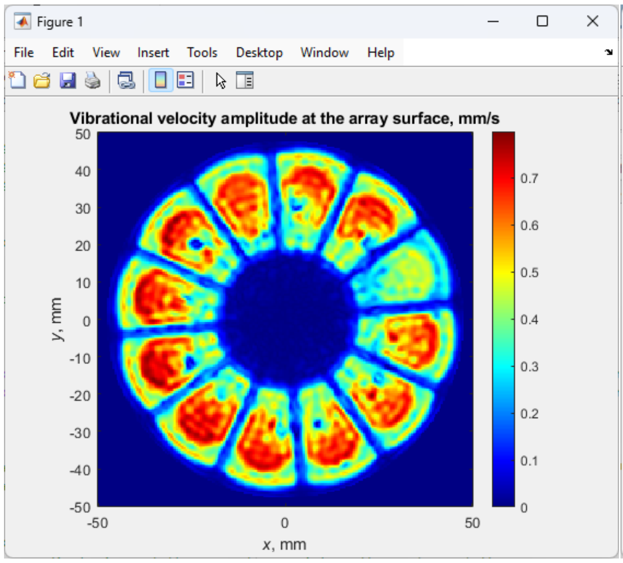
```
inputParametersFileName = 
'..\..\data_for_examples\sector\holo_data_sector_sf.xlsx'; %path to input data with transducer geometry, single-frequency hologram, and medium parameters in '.mat' or '.xlsx' format
```

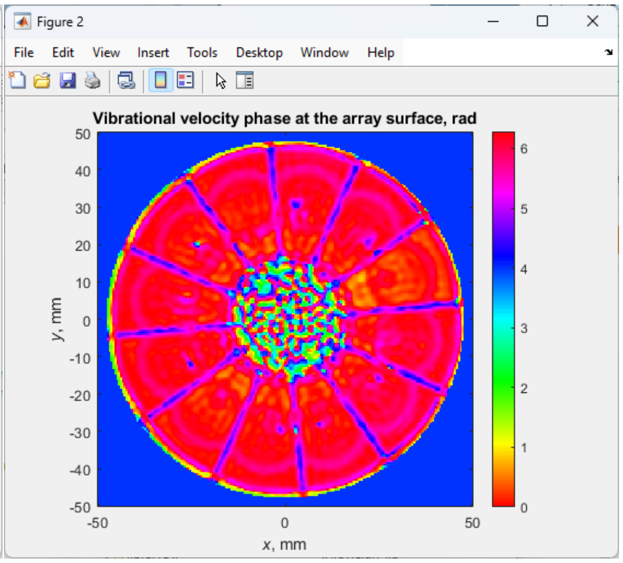
- A MAT file containing alignment parameters (see Automatic Alignment Tool)

```
alignmentParametersFileName = 
'..\..\data_for_examples\sector\alig_param_sector.mat'; %data structure with position parameters (can be saved using 'bp_auto_alignment_sf.m')
```

The other parameters are similar to the Automatic Alignment Tool.

Simulation output





The output in this example matches the result obtained with the Automatic Alignment GUI for a radius of curvature of 92 mm (page 58).

Details

One possible application of this tool is the rotation of holograms for multi-element arrays with varying phase patterns at each element. Once alignment parameters are saved for one configuration with a tight focal lobe, holograms for other field patterns without a tight focal lobe can be measured and aligned for the same transducer position.

Pre-Defined Alignment Tool: Transient Case

This tool rotates and shifts a measured transient hologram so that the aligned hologram is perpendicular to the transducer's axis of symmetry, then back-projects it to the transducer's surface. The script reads predetermined position parameters from the file 'alignmentParametersFileName'. This file can either be saved using the Automatic Alignment Tool or constructed manually (see the Manual Alignment section below).

The output frame-by-frame GUI displays results in both frequency and time domains, with an option to render the results in video format.

Note: transient simulations may require more processing time compared to single-frequency simulations.

Example 10: *Pre-Defined Alignment for Transient Hologram*

Path: “xDDx\examples\alignment_tools\bp_predefined_alignment_transient.m”

General input parameters

- Hologram parameters:

A MAT file containing three structures: `Geometry` (see page 10), `HologramTr` (see page 13) and `Medium` (see page 11)

```
inputParametersFileName =
'..\data_for_examples\sector\holo_data_sector_transient.mat'; %path to input data with transducer geometry, transient hologram, and medium parameters

frequencyCentral = 1.25e6; %approximate central frequency of the transducer in Hz
```

- Grid parameters for the transducer surface (same as used in the Automatic Alignment Tool)

```
nxSource = 201; %number of points for the Cartesian grid of the source along the x- and y-dimension

nySource = 201;

dxSource = 0.5e-3; %x and y grid step in m

dySource = 0.5e-3;

xZeroPhase = -25e-3; %(optional parameter) x-coordinate of the zero-phase point on the surface of the source

yZeroPhase = -25e-3; %(optional parameter) y-coordinate of the zero-phase point on the surface of the source
```

- Range of frequency samples for the transient hologram projection

```
minFrequencySample = 30; %frequency samples ranging (minFrequencySample : maxFrequencySample)*frequencyStep, where frequencyStep = 1 / "time window for HologramTr.time"

maxFrequencySample = 90; %set 'auto' to determine the values automatically based on the powerSignificanceLevel
```

This value can be obtained using the Power Spectrum Tool or set in the automatic regime (Example 7).

- Technical parameters

```
levelSignalToSave = 0.05; %minimum velocity waveform level relative to the maximum velocity to be saved in the output video

showRingingDown = false; %true to show all output time points, false to show only the part of the signal greater than levelSignalToSave
```

These parameters simplify time-domain output display. The software automatically detects the start of the back-projected signal as the first instance when the maximum vibrational velocity at the transducer's surface exceeds the `levelSignalToSave` threshold. The default level of 5% is suitable for most cases.

The parameter `showRingingDown` should be set to `true` to display not only the reconstructed transient velocity burst at the transducer surface but the "ringing down" throughout the time window. Although typically unnecessary, this feature can help identify minor defects. When `showRingingDown` is set to `false`, the "ringing down" portion of the signal, if below `levelSignalToSave`, is ignored.

- Output video parameters

Optionally, the tool can save the frequency domain ("signal") and time domain ("spectrum") outputs in the same directory, with user-specified filenames. Frame rates for time-domain and frequency-domain videos can be set separately.

```
saveSurfSpectrum = false; %set true to save a video with the back-projected velocity amplitudes at different frequencies

outputVideoSurfSpectrum = 'out_v_spec_source.mp4'; %output filename

frameRateVideoSurfSpectrum = 4; %frame rate in the output video


saveSurfSignal = false; %set true to save a video with the back-projected velocity waveforms at the surface of the transducer

outputVideoSurfSignal = 'out_v_sgnl_source.mp4'; %output filename

downsampleCoeffTime = 4; %downsampling factor for the output back-projected velocity waveform related to the input 'HologramTr.time' samples

frameRateVideoSurfSignal = 20; %frame rate in the output video
```

To manage the high sample count, which may result in lengthy video rendering, users can downsample the time-domain video using the `downsampleCoeffTime` parameter. By default, videos are not saved, as the GUI can be used to present results without rendering.

Simulation output

The results are displayed in both frequency and time domains using the same GUI described for the Quick Start Tool.

1) Frequency-domain output

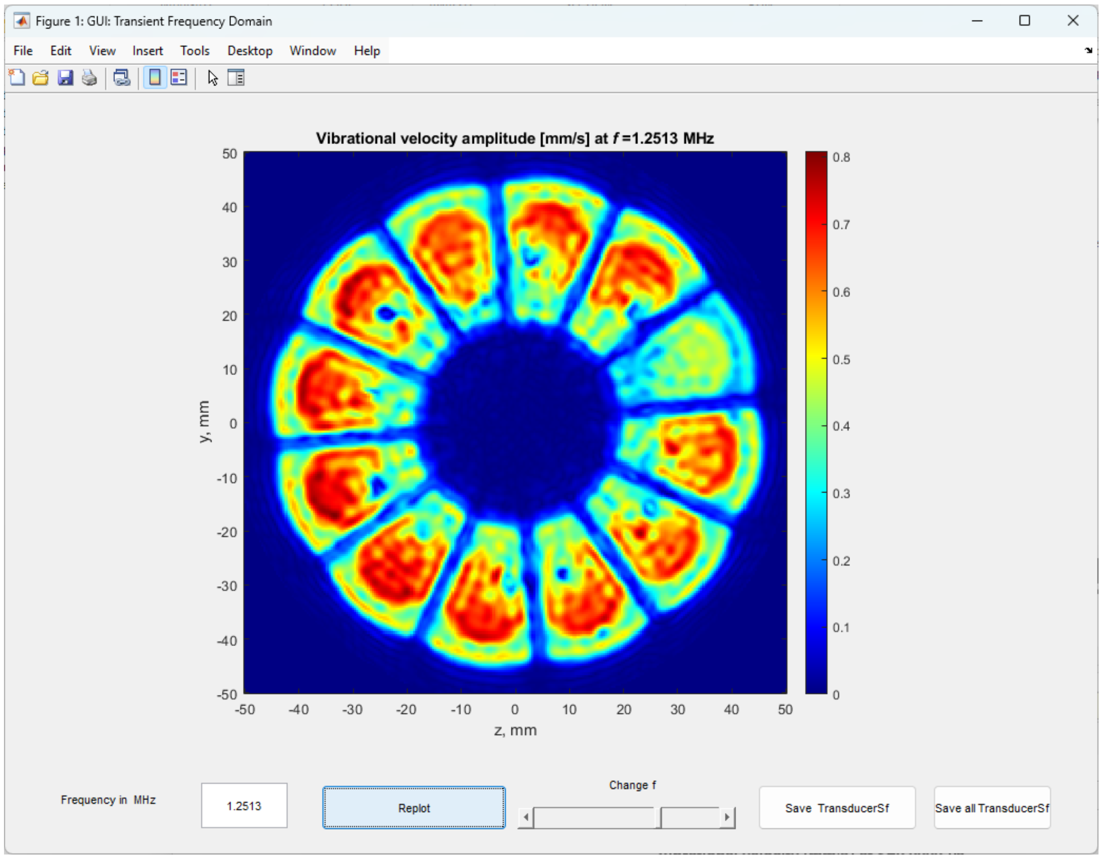


Figure 1: GUI: Transient Frequency Domain
File Edit View Insert Tools Desktop Window Help
Vibrational velocity amplitude [mm/s] at f =1.2513 MHz
y, mm
z, mm
50
40
30
20
10
0
-10
-20
-30
-40
-50
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
Frequency in MHz
1.2513
Replot
Change f
Save TransducerSf
Save all TransducerSf

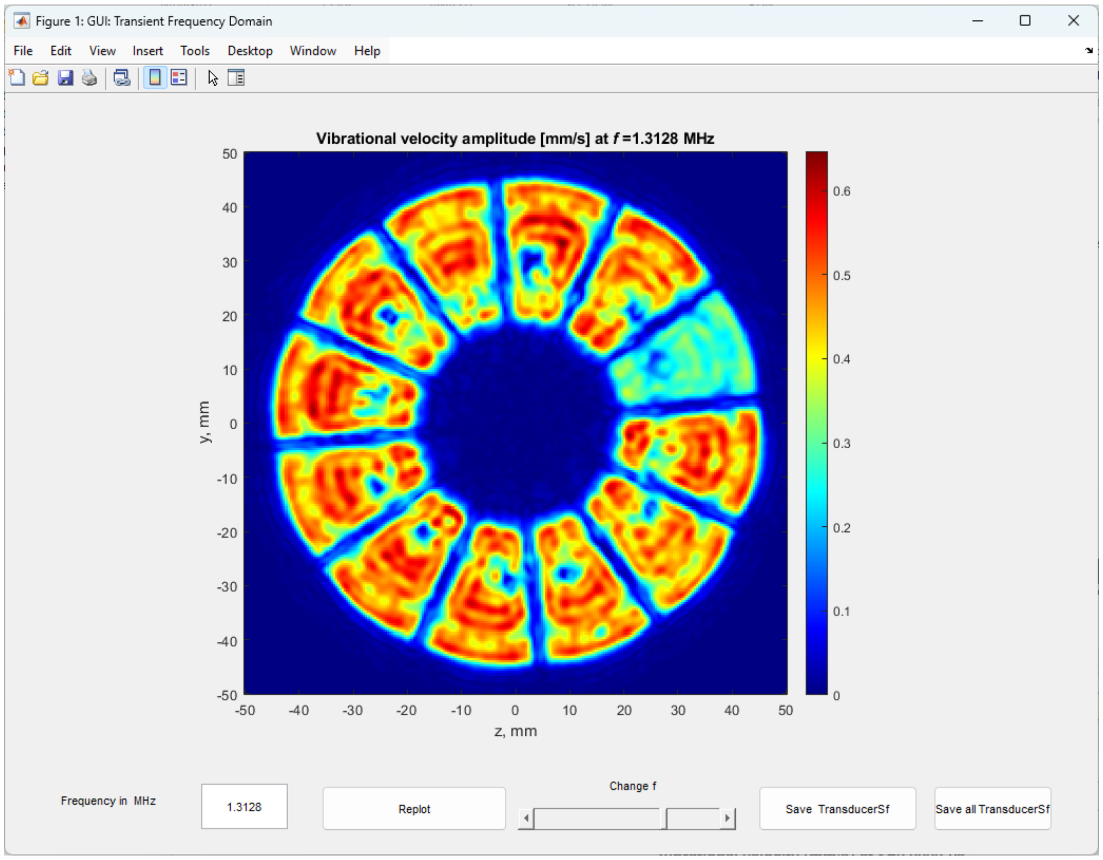


Figure 1: GUI: Transient Frequency Domain
File Edit View Insert Tools Desktop Window Help
Vibrational velocity amplitude [mm/s] at f =1.3128 MHz
y, mm
z, mm
Frequency in MHz
1.3128
Replot
Change f
Save TransducerSf
Save all TransducerSf

The “Change f” slider or the “Frequency in MHz” text box can allows frequency adjustments, and the “Save” buttons provide options to save results in various formats:

“Save TransducerSf” saves the back-projected vibrational velocity complex amplitude for the currently selected frequency as a `TransducerSf` structure (page 19).

“Save all TransducerSf” saves the Back-Projected vibrational velocity complex amplitude for all frequency samples as a cell array `TransducersSf`. Each cell, `TransducersSf{iFreqSample}`, is a `TransducerSf` structure at the frequency that corresponds to the frequency range `(minFrequencySample : maxFrequencySample)*frequencyStep`

The saved structures can be used for realistic field simulations for the current transducer (explained further in Example 12, page 88).

2) Time domain

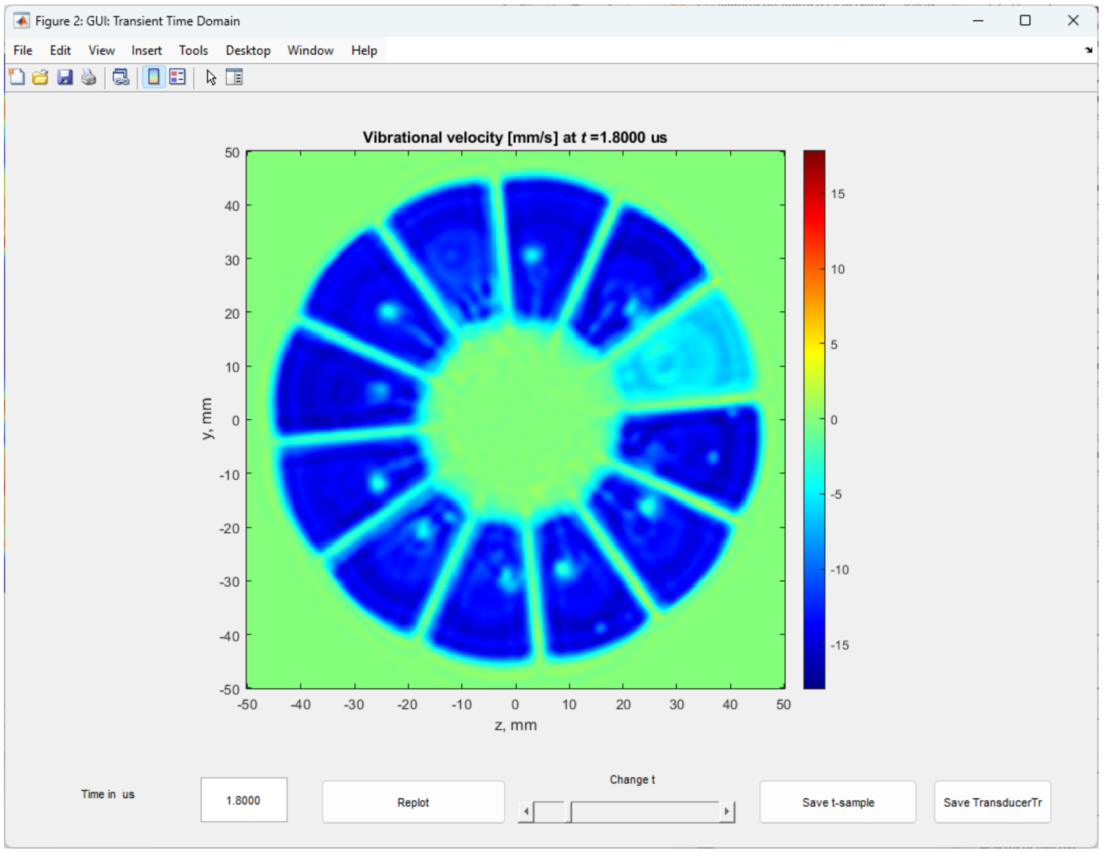


Figure 2: GUI: Transient Time Domain
File Edit View Insert Tools Desktop Window Help
Vibrational velocity [mm/s] at t =1.8000 us
y, mm
z, mm
Time in us
1.8000
Replot
Change t
Save t-sample
Save TransducerTr

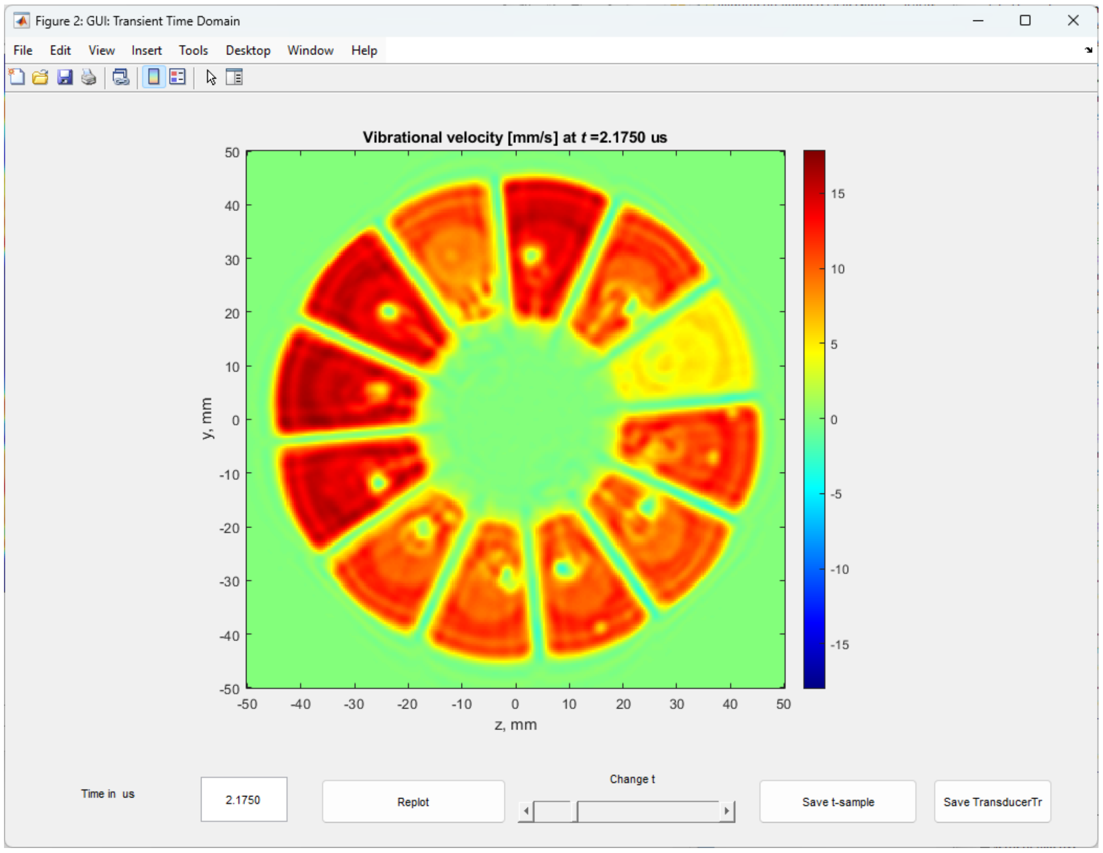
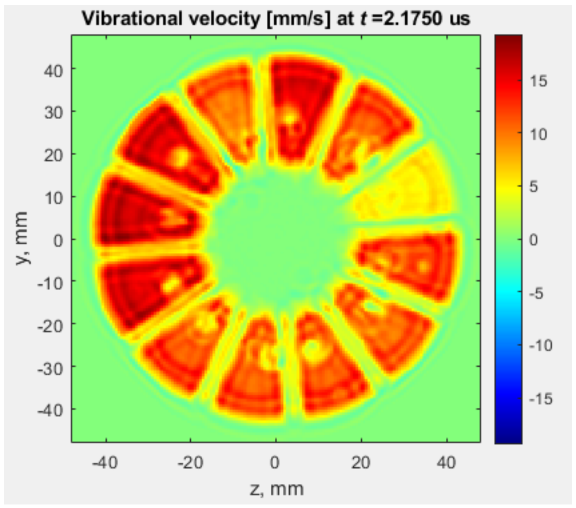
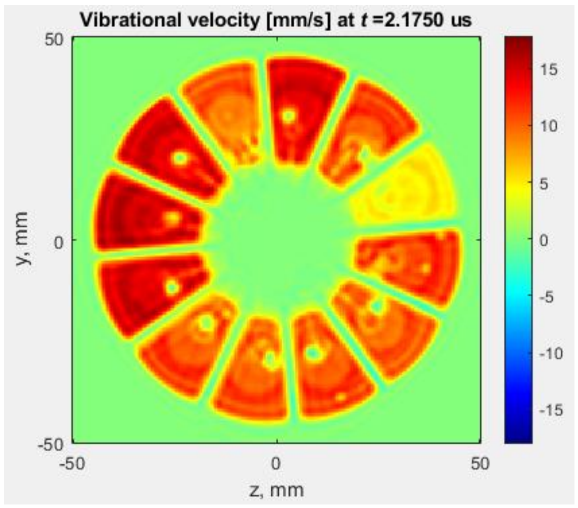


Figure 2: GUI: Transient Time Domain
File Edit View Insert Tools Desktop Window Help
Vibrational velocity [mm/s] at t =2.1750 us
y, mm
z, mm
Time in us
2.1750
Replot
Change t
Save t-sample
Save TransducerTr

Details

The vibrational velocity distribution obtained using this alignment tool (shown in the right figure below) is noticeably improved compared to the Quick Start Tool output (shown in the left figure below). This improvement is due to selecting a realistic radius of curvature, which enhances the sharpness of the transducer surface image and visualizes surface defects more clearly.





The slider and text box function similarly to the frequency domain case, and the “Save” buttons offer the following options:

“Save t-sample” saves the current “frame” for a selected time sample. The saved structure `SurfaceVelocityFrame` is similar to the `'TransducerTr'` structure, as shown in the example below:

```
SurfaceVelocityFrame =

  struct with fields:

                 time: 1.8000e-06

    radiusOfCurvature: 0.0920

                xGrid: [201×201 double]

                yGrid: [201×201 double]

                zGrid: [201×201 double]
```

```
            dx: 5e-04
            dy: 5e-04
            velocity: [201×201 double]
```

The following code demonstrates how to plot saved results:

```
xGrid = SurfaceVelocitySample.xGrid;
yGrid = SurfaceVelocitySample.yGrid;
velocity = SurfaceVelocitySample.velocity;
figure;
contourf(xGrid,yGrid,velocity);
axis equal;
```

"Save TransducerTr" saves the back-projected vibrational velocity in time domain as a `TransducerTr` as structure (page 18). The saved structures can be used for realistic field simulations for the current transducer (explained further in Example 13, page 88).

Manual Alignment Algorithm

The tools described above can be combined to manually align holograms measured for transducers without tight focal lobes. The example below demonstrates this process for a flat transducer (Fig. 10b), which shows some field symmetry but lacks a focal lobe, as it is unfocused.

Example 11: *Manual Alignment for a Flat Transducer*

1. Detecting field symmetry patterns

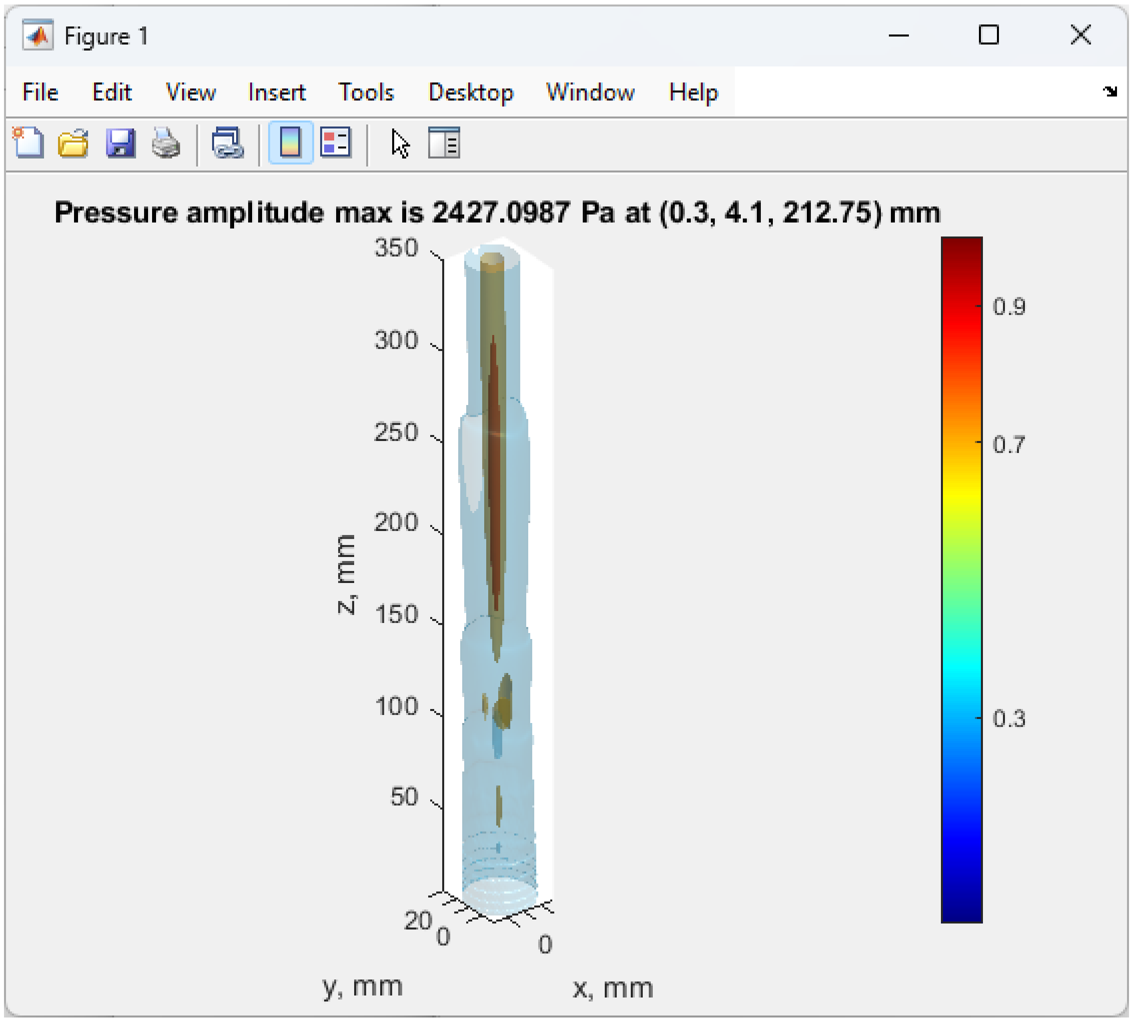
First, the Single-Frequency Forward-Projection Tool is used to visualize the field pattern. In the example case, a single-frequency hologram at 1 MHz was used:

```
inputParametersFileName =
'..\data_for_examples\flat\holo_data_flat_transient.mat'; %path to input data
with transducer, single-frequency hologram, and medium parameters
```

Importantly, the simulation should be conducted in the mechanical coordinates (Fig. 7a), so the hologram z-position was set to zero:

```
HologramSf.zPosition = 0;
```

The Simulation reveals the field pattern, highlighting the global field maximum and indicating that `isoLevel = 0.9` has sufficient symmetry to identify the acoustical *z*-axis of the transducer.



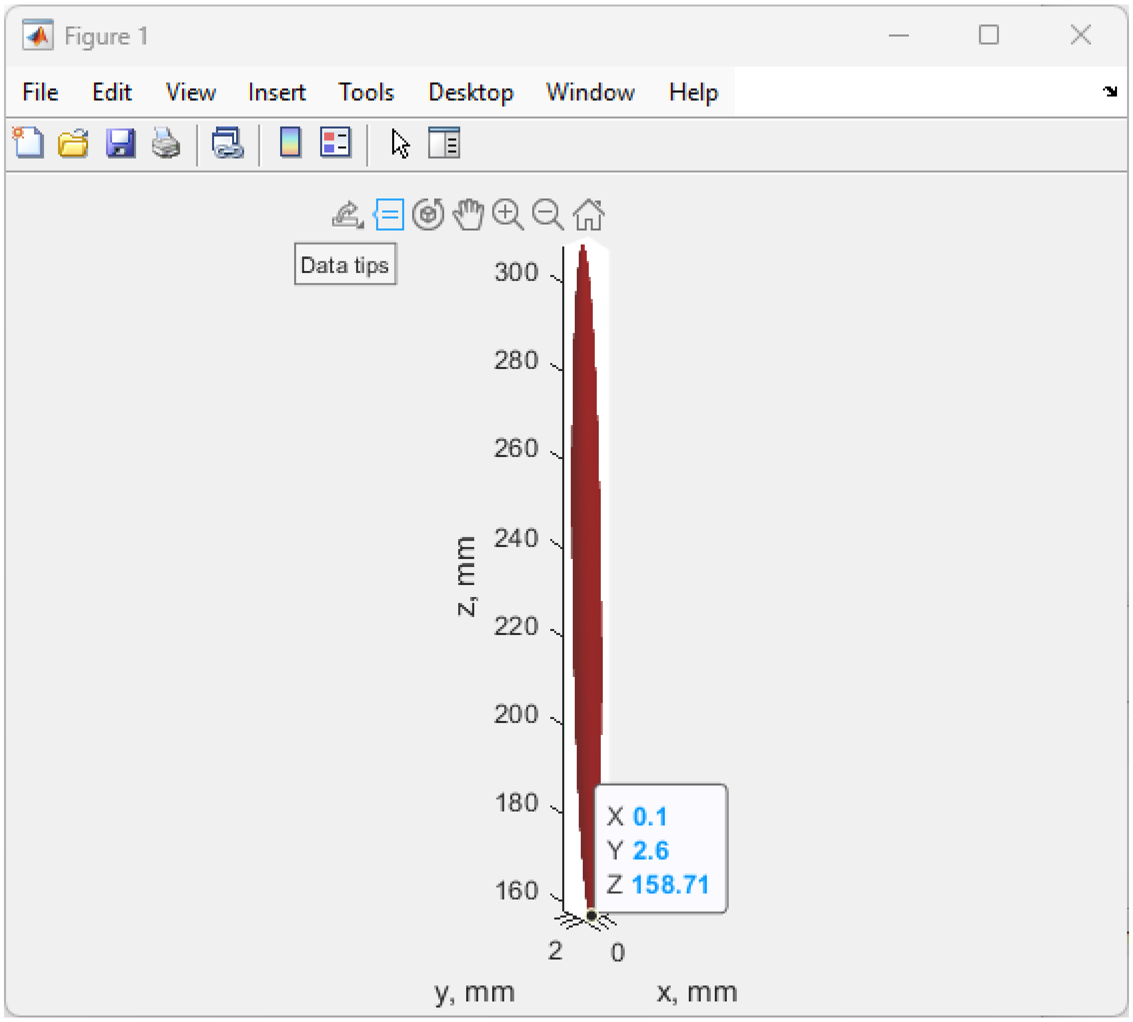
2. Determining alignment parameters

The first three alignment parameters can be derived directly from the forward-projection:

```
xMaxMechCoord = 0.3e-3; %m

yMaxMechCoord = 4.1e-3;
```

```
zMaxMechCoord = 212.75e-3;
```

To determine the direction vector, it is necessary to locate the second point of symmetry `[xBaseMechCoord; yBaseMechCoord; zBaseMechCoord]` of the 0.9 iso-level lobe. This is easiest to do using the "Data Tips" feature in MATLAB/Octave:



```
xBaseMechCoord = 0.1e-3; %m

yBaseMechCoord = 2.6e-3;

zBaseMechCoord = 158.71e-3;
```

Thus, the direction-vector can be found as:

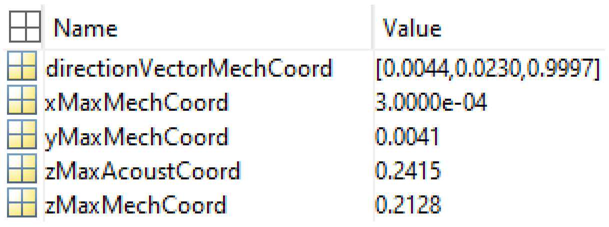
```
directionVectorMechCoord = [xMaxMechCoord - xBaseMechCoord; yMaxMechCoord - yBaseMechCoord; zMaxMechCoord - zBaseMechCoord]; %find the direction-vector
directionVectorMechCoord =
directionVectorMechCoord/norm(directionVectorMechCoord); %normalize the
direction-vector
```

Lastly, the acoustical *z*-coordinate of the maximum pressure point, `zMaxAcoustCoord`, can be determined. For non-spherical transducers, prior knowledge of the transducer geometry is useful. In this case, the idealized analytical formula for the field maximum position is applied [O'Neil 1949]. Given a circular transducer with a known aperture of 38 mm and wavelength of 1.5 mm:

```
zMaxAcoustCoord = aperture^2 / (4*wavelength) - wavelength / 4
```

This results in `zMaxAcoustCoord` = 241.5 mm.

By compiling all parameters into a single alignment-parameters data file, we obtain the following MAT-data file, saved in the example-data folder:
`'xDDx\data_for_examples\flat\alig_param_flat.mat'`.



Name	Value
directionVectorMechCoord	[0.0044,0.0230,0.9997]
xMaxMechCoord	3.0000e-04
yMaxMechCoord	0.0041
zMaxAcoustCoord	0.2415
zMaxMechCoord	0.2128

3. Applying Pre-Defined Alignment

Finally, the saved alignment-parameters file is used with the Pre-Defined Alignment Tool for either the single-frequency or transient cases. An example of time-domain results in the transient case is shown for two time-samples below.

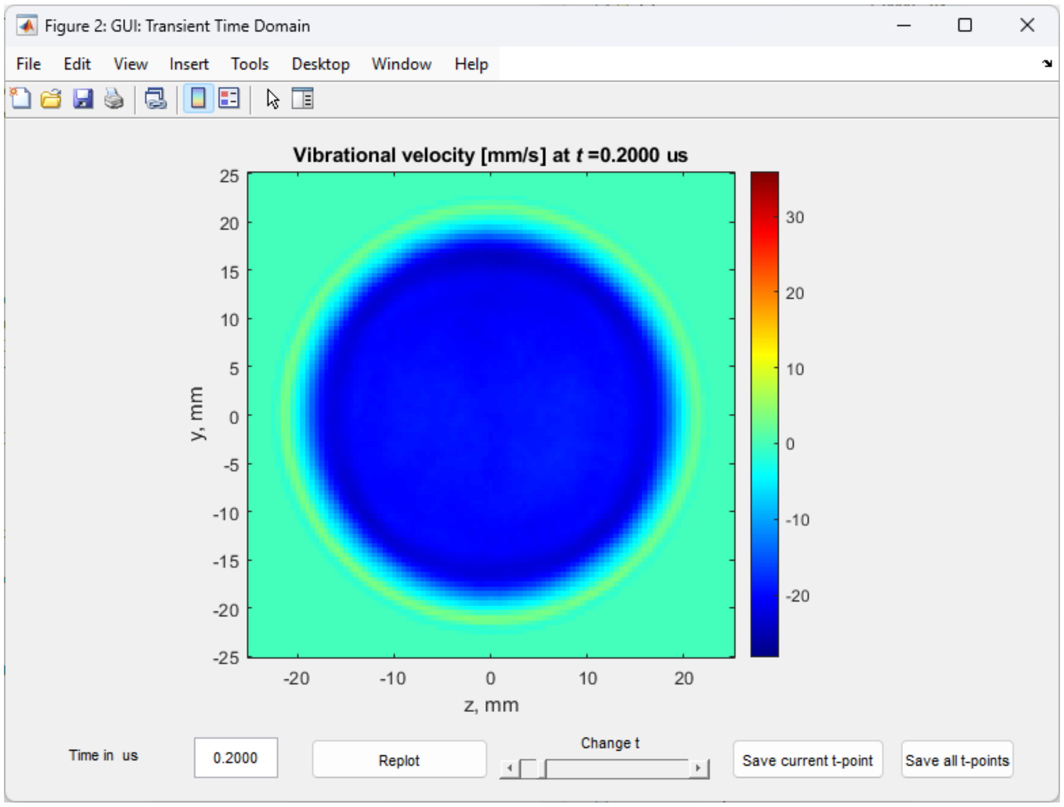


Figure 2: GUI: Transient Time Domain
File Edit View Insert Tools Desktop Window Help
Vibrational velocity [mm/s] at t =0.2000 us
y, mm
z, mm
Time in us
0.2000
Replot
Change t
Save current t-point
Save all t-points

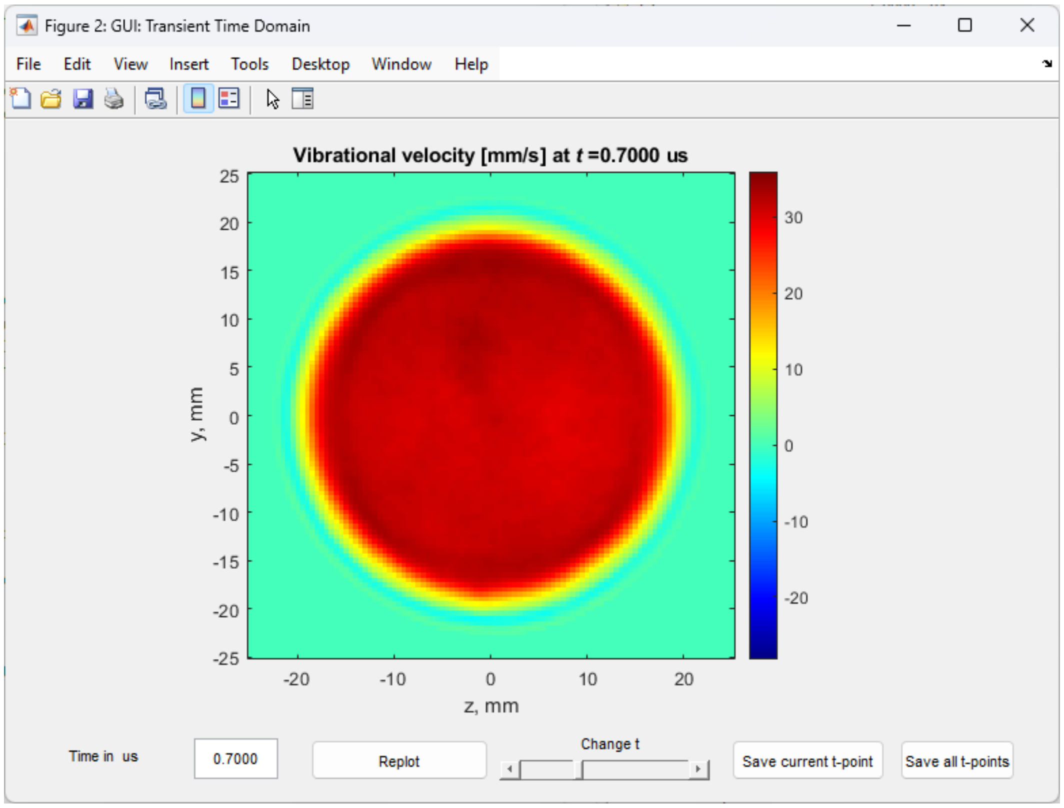
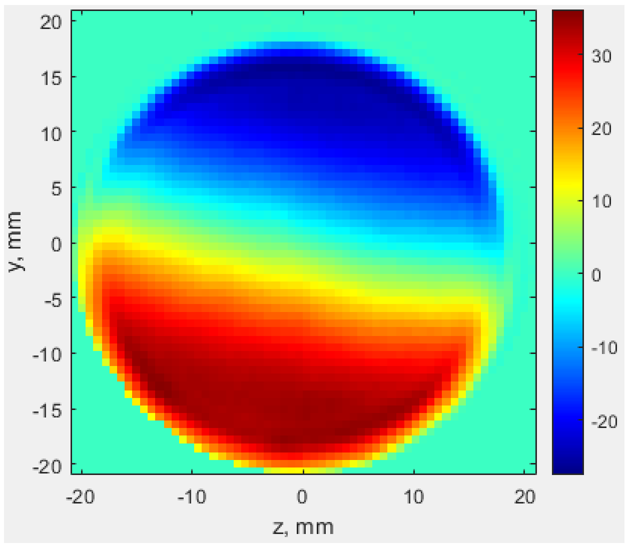
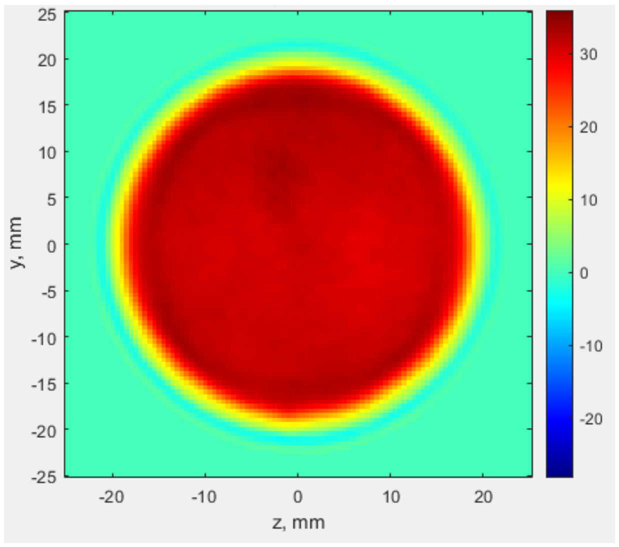


Figure 2: GUI: Transient Time Domain
File Edit View Insert Tools Desktop Window Help
Vibrational velocity [mm/s] at t =0.7000 us
y, mm
z, mm
Time in us
0.7000
Replot
Change t
Save current t-point
Save all t-points

It is clear that the tilted hologram plane issue present in the Quick Start Tools without alignment (shown in the left figure below) is fully resolved. The post-alignment vibrational velocity distribution is centered and uniform (shown in the right figure below):





SIMULATION TOOLBOX

The second part of this toolbox can be used independently of the first to simulate fields of computer models of transducers in uniform propagation media (usually water). The same field projection concepts as those in the "Holography Toolbox" are implemented here using the Rayleigh integral method. A more detailed description of the numerical simulation method can be found in the paper [Rosnitskiy 2025].

Although this chapter, "Simulation Toolbox," relies on the data formats introduced in the first chapter, "Holography Toolbox," the simulation procedure is entirely independent of holography field scans and does not require any measurements. Nonetheless, some general points described in the first chapter will be repeated below, allowing users to skip the first chapter if they are focused solely on simulations rather than hologram post-processing.

GENERAL CONCEPTS

The text below includes portions of MATLAB/Octave code relevant to the topics discussed, presented in a `different font` for convenience. Please note that all input parameters of the toolbox are expressed in the International System of Units (SI).

Single-Frequency Transducer Definition

A general and flexible transducer model is implemented for a single-frequency transducer surface with a known `frequency`, operating in the continuous wave (CW) regime. The transducer surface is considered a set of discretized point sources located at the nodes of a Cartesian grid, each with a known normal component of the vibrational velocity.

It is known that discrete Rayleigh integral summation in a single-frequency case with spatial step sizes `dx` and `dy` equal to one-half wavelength or less can accurately represent the acoustic field [Sapoznikov 2015]. Given possible floating-point errors, it is recommended to select the discretization step slightly smaller than the half-wavelength limit. For example, in idealized water with a sound speed of 1500 m/s and a transducer with a central frequency of 1 MHz, the wavelength is 1.5 mm, making the half wavelength 0.75 mm. Therefore the x- and y-steps for the discretization grid can be chosen as be 0.7 mm.

The most illustrative way to define a transducer grid is by using three matrices: `xGrid(i,j)`, `yGrid(i,j)`, and `zGrid(i,j)`, where the row indices `i` and the column indices `j` correspond to the respective axes directions. The x- and y-grid steps are defined as `dx` and `dy`. A common way to define these matrices in MATLAB/Octave is with the `meshgrid` function, which maps the *y*-direction with rows (i-index) and the x-direction with columns (j-index), as shown in Fig. 11a. It is convenient to define the coordinate system so that the origin is at the surface of the transducer. Thus, in the case of a flat transducer,

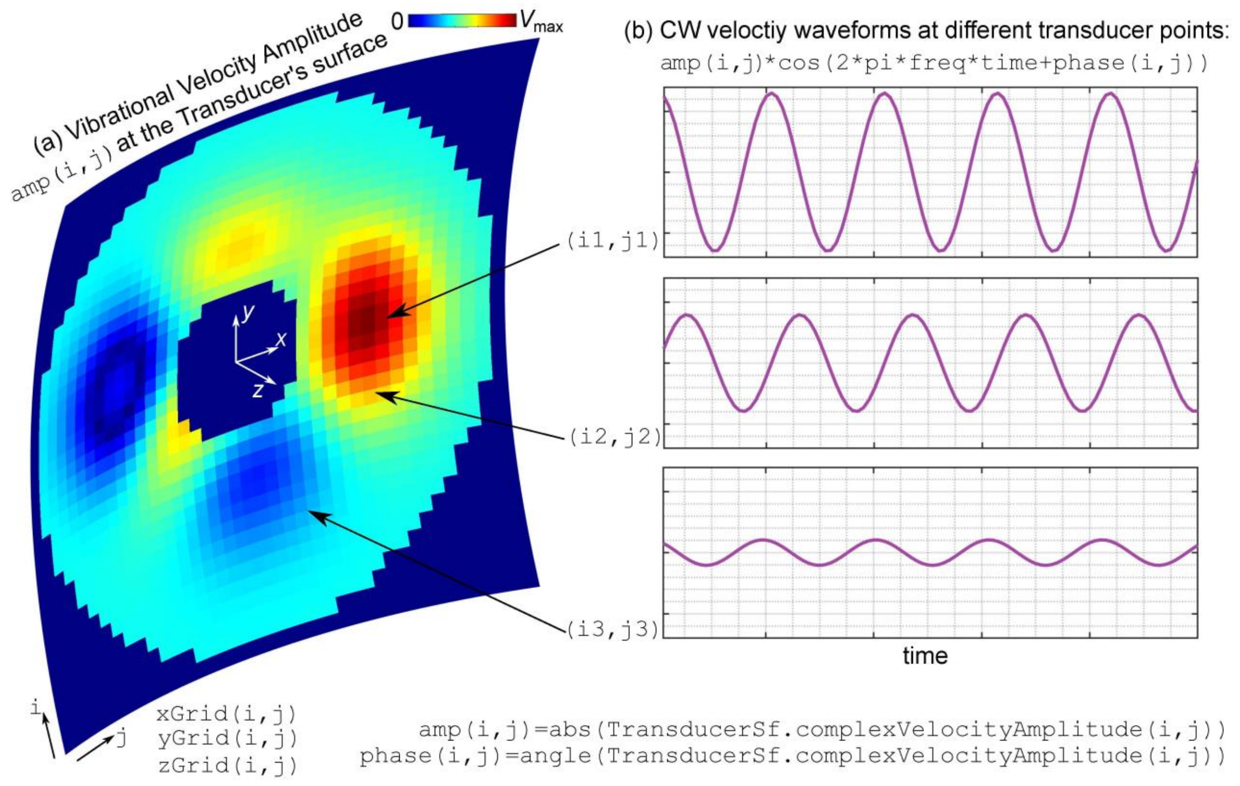

(a) Vibrational Velocity Amplitude at the Transducer's surface
amp(i,j)
0
V_{max}
(b) CW veloctiy waveforms at different transducer points:
amp(i,j)*cos(2*pi*freq*time+phase(i,j))
(i1,j1)
(i2,j2)
(i3,j3)
time
y
x
z
i
j
xGrid(i,j)
yGrid(i,j)
zGrid(i,j)
amp(i,j)=abs(TransducerSf.complexVelocityAmplitude(i,j))
phase(i,j)=angle(TransducerSf.complexVelocityAmplitude(i,j))

Fig. 11. Transducer representation sketch for the single-frequency regime.

`zGrid(i,j) = 0` for all `i` and `j`. For a spherically shaped transducer with a known `radiusOfCurvature`, the *z*-coordinates can be calculated using the formula

```
zGrid(i,j) =
radiusOfCurvature - sqrt(radiusOfCurvature^2 - xGrid(i,j)^2 - yGrid(i,j)^2);
```

In addition to the `(i,j)` index notation, users can map their vertex sets `xGrid` and `yGrid` using 1D indexing in any convenient order and direction. This is explained in detail in the Holography Toolbox chapter, under the "*Holography Scan*" and "*Transducer Grid Definition*" sections. Using these `(i,j)` index notations, the time-related vibrational velocity of each discretized point source can be represented as an amplitude `amp(i,j)` and phase `phase(i,j)`:

```
amp(i,j)*cos(2*pi*frequency*time+phase(i,j))
```

where the variable `time` represents the moment of interest. Fig. 11b illustrates three such waveforms at different points `(i1,j1)`, `(i2,j2)`, and `(i3,j3)` on a transducer with a nonuniform vibrational velocity distribution. All waveforms have the same frequency but different amplitudes `amp(i,j)` and phase shifts `phase(i,j)`. The amplitude and phase are typically presented as a complex amplitude

```
complexVelocityAmplitude(i,j)=amp(i,j)*exp(±1i*phase(i,j))
```

Thus,

```
amp(i,j) = abs(complexVelocityAmplitude(i,j))

phase(i,j) = angle(complexVelocityAmplitude(i,j))
```

Here, `1i` is the MATLAB/Octave notation for the imaginary unit. Note that the "±" sign in the complex representation is determined by the exponent sign convention `exp(+2*pi*1i*frequency*time)` or `exp(-2*pi*1i*frequency*time)`.

The toolbox utilizes the exponent sign parameter `expSign = +1` or `-1` to specify the convention used. For example, the `fft` function in MATLAB/Octave utilizes the `exp(+2*pi*1i*frequency*time)` convention so the default exponent convention used by the toolbox is `expSign = +1`.

Finally, all transducer parameters can be combined into the structure `TransducerSf` that was introduced in the first chapter. It is detailed here again for convenience:

```
% 'TransducerSf' struct with fields:
%   'expSign': +1 or -1, depending on the exponent sign convention
%              exp(+1i * omega * t) or exp(-1i * omega * t). E.g., the "fft"
%              function in MATLAB utilizes the exp(+1i * omega * t)
%              convention, so in this case, expSign is +1
%   'frequency': the operating frequency of the transducer in Hz
%   'radiusOfCurvature': (optional, can be omitted) radius of curvature of
                          the transducer in m. Omit this field or set it as
                          an empty vector [] if your transducer is flat
%   'xGrid': vector or matrix with the x-coordinates in m at each grid node
%            of the transducer surface (Cartesian grid only)
%   'yGrid': vector or matrix with the y-coordinates in m at each grid node
%            of the transducer surface (Cartesian grid only)
%   'zGrid': vector or matrix with the z-coordinates in m at each grid node
%            of the transducer surface (Cartesian grid only). Users can omit
%            this variable, in such cases, the toolbox will automatically
%            determine the transducer type—spherical or flat. For a
%            spherical transducer, zGrid nodes will be set according to the
%             sphere equation:
%             zGrid = radiusOfCurvature - sqrt(radiusOfCurvature^2 -
%             xGrid.^2 - yGrid.^2) For a flat transducer, all zGrid nodes
%             will be set to zero:
%             zGrid = zeros(size(xGrid))
%    'dx': x-step of the transducer Cartesian grid in m
%    'dy': y-step of the transducer Cartesian grid in m
%    'complexVelocityAmplitude': complex velocity amplitude in m/s at each
%                                node of the transducer surface grid
```

`TransducerSf` structures can be created for the cases of interest using ".xlsx" templates located in "xDDx\examples\data_for_examples\xlsx_templates". These templates are divided into sheets containing scalar parameters (such as expSign and frequency) and separate sheets for each matrix parameter (such as xGrid). Complex parameters, like complexVelocityAmplitude, are split into two sheets: one for the amplitude matrix and another for the phase matrix. Each .xlsx file includes a sheet named "Info", which provides detailed explanations of all parameters.

Transient Transducer Definition

The transducer model described in this subsection is a generalization of the single-frequency model `TransducerSf`. This generalized model represents a similar idea of the discretized transducer surface, but in this case each node of the transducer grid, `xGrid(i,j)`, `yGrid(i,j)`, `zGrid(i,j)`, is associated with the vector of vibrational velocity values at different moments of time (Fig. 12a). The data for all points of the transducer grid can be represented as a 3D matrix, `velocity(i,j,s)`, where the third dimension `s` corresponds to the number of time samples in the acquired signal.- The time samples are stored in the time window vector, `time = [time(1), time(2), ... , time(end)]`, where the first sample `time(1) = 0`, corresponds to the start of the acoustic signal, and the last sample `time(end)` marks the end of the signal, ensuring that the velocity `velocity(i,j,sEnd)` is below the noise level for all spatial grid points `(i,j)`. The `time` vector should have a uniform time step, `timeStep = time(s) - time(s-1)`. Examples of three hydrophone signals acquired at different grid vertices, `(i1,j1)`, `(i2,j2)`, `(i3,j3)` are shown in Fig. 12b. Thus, in MATLAB/Octave notation, a signal at a vertex `(i,j)`, can be represented as a 1D vector, `velocity(i,j,:)`. Similarly, in the case of 1D spatial indexing, the scan result matrix will be 2D: `velocity(iLin,s)`.

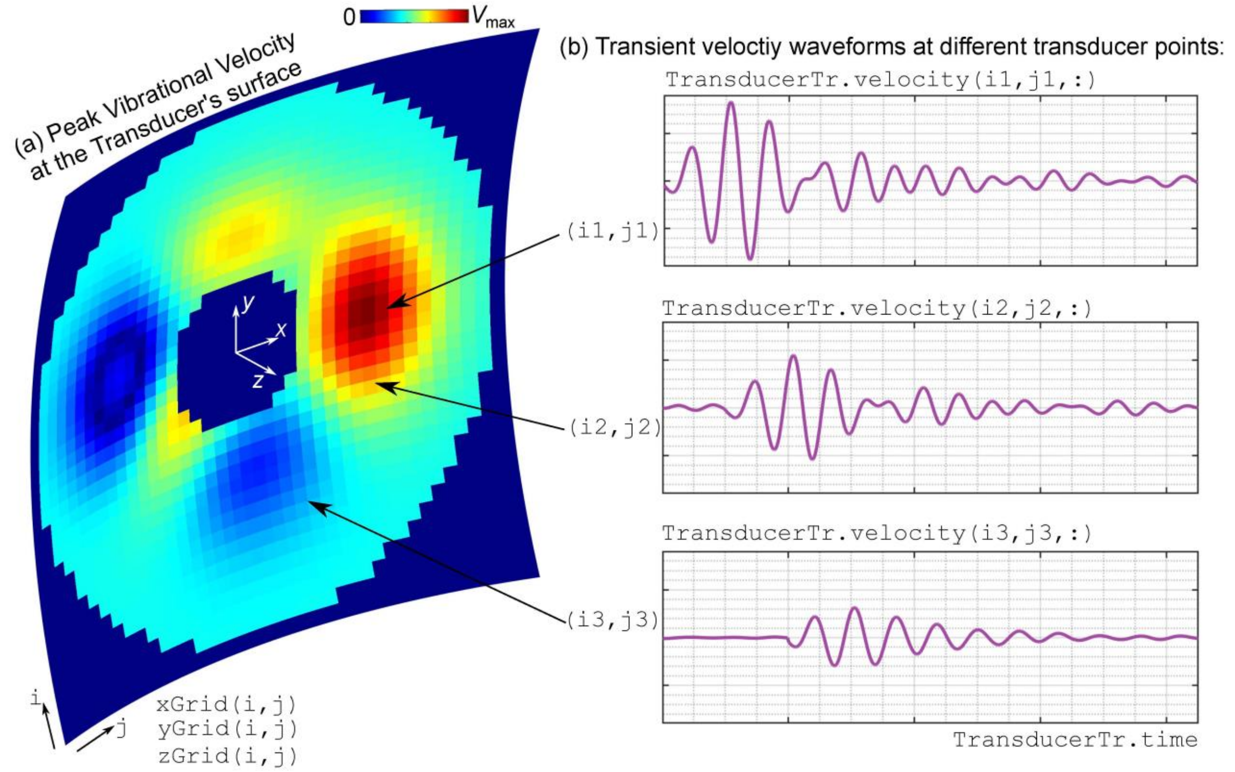

0
V_{max}
(a) Peak Vibrational Velocity at the Transducer's surface
(b) Transient velocity waveforms at different transducer points:
TransducerTr.velocity(i1,j1,:)
TransducerTr.velocity(i2,j2,:)
TransducerTr.velocity(i3,j3,:)
(i1,j1)
(i2,j2)
(i3,j3)
y
x
z
i
j
xGrid(i,j)
yGrid(i,j)
zGrid(i,j)
TransducerTr.time

Fig. 12. Transducer representation sketch for the transient regime.

Thus, the transient transducer parameters can be combined into the structure `TransducerTr` that was introduced in the first chapter. It is detailed here again for convenience:

```
% 'TransducerTr' struct with fields:
%   'radiusOfCurvature': (optional, can be omitted) radius of curvature of
%                        the transducer in m. Omit this field or set it as
%                        an empty vector if your transducer is flat
%     'xGrid': vector or matrix with x-coordinates of the Cartesian grid in
%              m at each grid node of the transducer surface
%     'yGrid': vector or matrix with y-coordinates of the Cartesian grid in
%              m at each grid node of the transducer surface
%     'zGrid': vector or matrix with z-coordinates of the Cartesian grid in
%              m at each grid node of the transducer surface. Users can omit
%              this variable, in such cases, the toolbox will automatically
%              determine the transducer type—spherical or flat. For a
%              spherical transducer, zGrid nodes will be set according to the
%              sphere equation:
%              zGrid = radiusOfCurvature - sqrt(radiusOfCurvature^2 -
%              xGrid.^2 - yGrid.^2) For a flat transducer, all zGrid nodes
%              will be set to zero:
%              zGrid = zeros(size(xGrid))
%     'dx': x-step of the transducer Cartesian grid in m
%     'dy': y-step of the transducer Cartesian grid in m
%     'time': vector with the time samples in s at each time grid nodes
%             for the recorded waveforms
%     'velocity': 2D or 3D matrix with the vibrational velocity
%                         waveforms in m/s at corresponding grid nodes of
%                         the transducer
%                         if xGrid/yGrid is a 2D matrix, then
%                         size(velocity) = [size(xGrid, 1)
%                         size(xGrid, 2) length(time)]
%                              if xGrid/yGrid is a vector, then
%                              size(velocity) = [length(xGrid)
%                              length(time)]
```

In the case of a transient transducer, the time window for simulation is an important parameter. The maximum limit of the time window `timeMax` should account for the entire duration required for the signal to travel from the transducer surface to the edge of the simulation region of interest, plus the ringing time of the recorded signal `tRinging= TransducerTr.time(end)`. An example of selecting this simulation window for a flat rectangular simulation region is shown in Fig. 13. Users have the option to calculate this window automatically based on the equations shown in the figure or to set it manually in seconds. The automatic option can also include zero-padding of the signal for time points beyond `TransducerTr.time(end)`.

Transducer Examples

Four transducer models are used as examples for the Simulation Toolbox (Fig. 14).

Simulation Transducer 1: "Flat Single-Element". An idealized piston flat transducer with a circular aperture of 20 mm operating in a single-frequency regime at 1 MHz. The vibrational velocity amplitude is uniformly distributed over the surface (Fig. 14a). For convenience, the amplitude is selected so that the characteristic surface pressure equals 1 Pa, where

```
characteristicPressureAmplitude =
soundSpeed * density * abs(complexVelocityAmplitude)
```

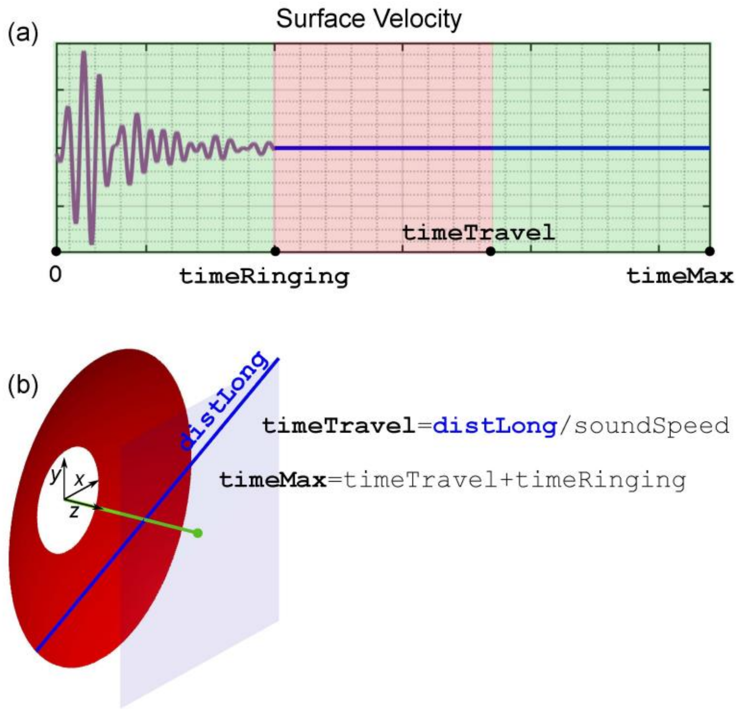


Fig. 13. Example of selecting a time window for a transient simulation.

Here, "default values" of a sound speed of 1500 m/s and density of 1000 kg/m^3 are used, so

```
abs(complexVelocityAmplitude) = 6.7e-7 %m/s
```

As the transducer is a piston, the phase is uniform across all points on the source and is equal to zero (Fig. 14b). Fig. 14c shows a 3D distribution of the vibrational velocity amplitude. Given the frequency and the sound speed, the "less than half-wavelength" discretization step in x and y direction was 0.7 mm.

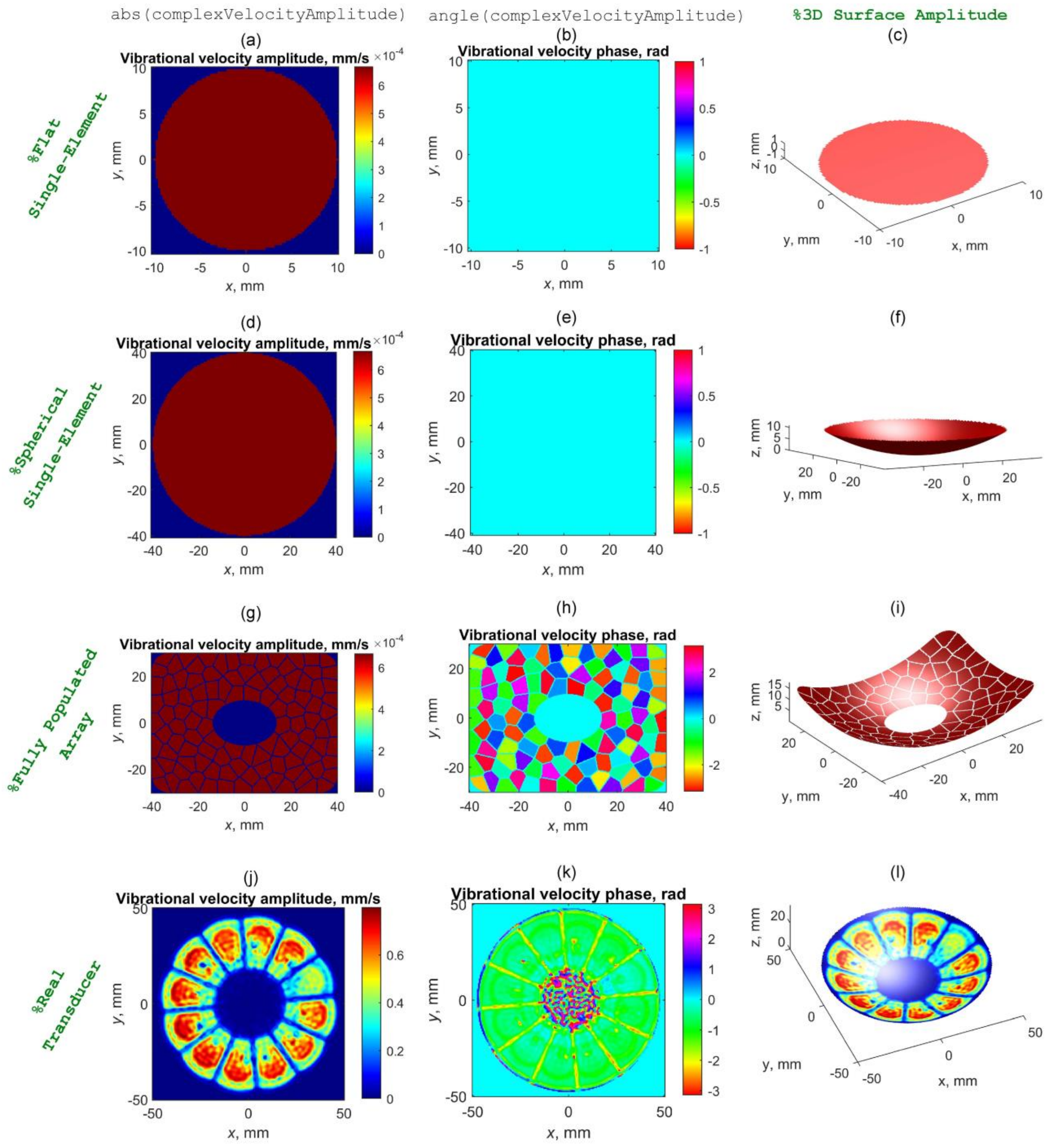


Fig. 14. Transducers used as examples for the Simulation Toolbox.

Simulation Transducer 2: “Spherical Single-Element”. An idealized piston spherically shaped transducer with a circular aperture of 80 mm and a radius of curvature of 80 mm, operating in a single-frequency regime at 1 MHz (Fig. 14d–f). The same vibrational velocity and discretization steps as those for “Simulation Transducer 1” were set.

Simulation Transducer 3: “Fully-Populated-Array”. A multi-element transducer with a complex and irregular vibrational velocity pattern: a 109-element 1-MHz spherically shaped array with a rectangular aperture, an oval opening and a randomized fully populated pattern of elements and 0.5-mm gap between the elements (see design method in [Rosnitskiy 2018]). The same vibrational velocities at the elements as that for “Simulation Transducer 1” were set (Fig. 14g) Phases at the array elements were specifically set to provide a 10-mm electronic steering of the focus in the x-direction (Fig. 14h).

Simulation Transducer 4: “Real Transducer”. The case presents the holography back-projection results saved for a real transducer using the GUI of the Automatic Alignment Tool (Example 8, Holography Toolbox). Two regimes are considered: a transient regime with a short 2-cycle 1.25-MHz pulse (`TransducerTr` structure), and a single-frequency regime at 1.25 MHz (`TransducerSf` structure).

SIMULATION TOOLS

Single-Frequency Simulation Tool

This tool is designed to simulate 3D, 2D, 1D, or 0D fields of transducer operation in a single-frequency regime. Transducer parameters are provided by the TransducerSf structure. The output acoustic pressure amplitude can be visualized in 3D, 2D, 1D, or 0D, with an option for slice-by-slice representation.

Example 12: *Single-Frequency Field Simulation for a Multi-Element Array*

Path: “xDDx\examples\simulation_toolbox\transducer_simulation_sf.m”

General input parameters

The interface of this tool is identical to the one used for forward-projection of a single-frequency hologram (Example 6, Holography Toolbox). The only difference is in the first two parameters:

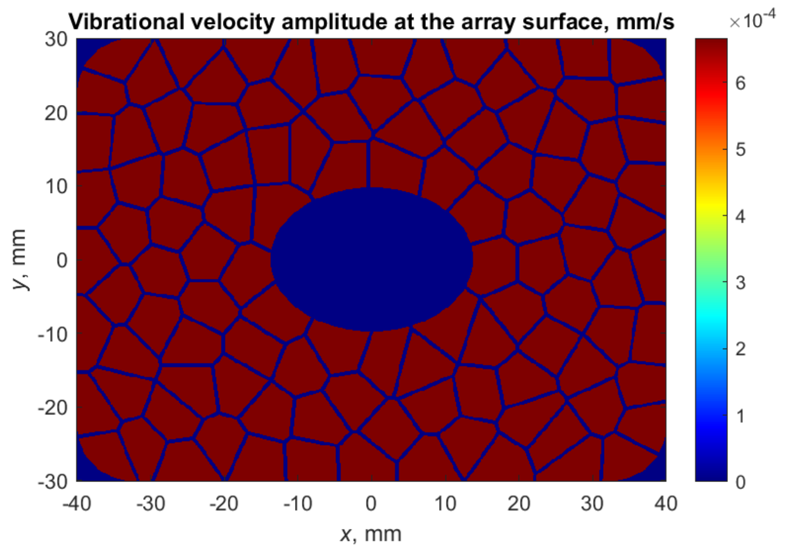
- Medium parameters:

```
Medium.soundSpeed = 1500; %sound speed in m/s

Medium.density = 1000; %density in kg/m^3
```

-Transducer parameters. A MAT or XLSX file containing a single-frequency transducer structure: `TransducerSf` (see page 77). This example uses the modes of Simulation Transducer 3 (Fully-Populated-Array). Three models are available for testing. The standard grid step model is used by default.

1) Fine grid step model (0.125-mm spatial step)

```
inputParametersFileName =

'..\..\data_for_examples\simulation\fpa_steering_10mm_fine_grid.mat';   %input
with the transducer TransducerSf data structure in '.mat' or '.xlsx' format
```

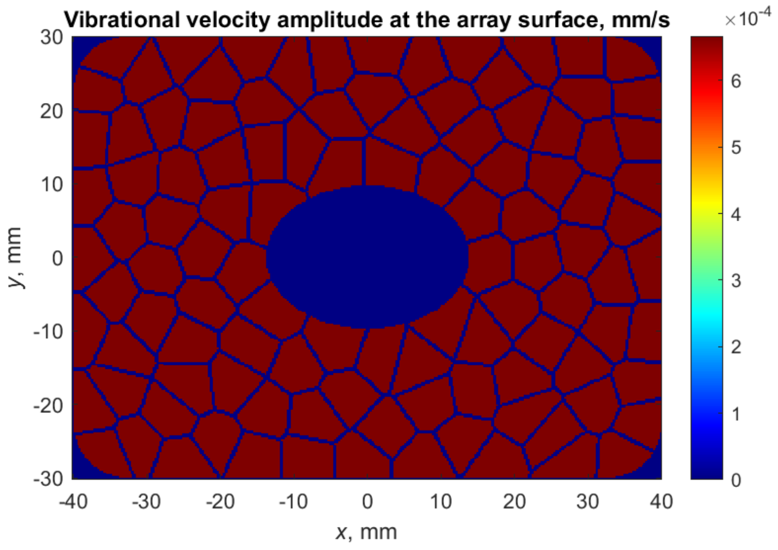


2) Standard grid step model (0.25-mm spatial step)

```
inputParametersFileName =

'..\..\data_for_examples\simulation\fpa_steering_10mm_standard_grid.mat';
%input with the transducer TransducerSf data structure in '.mat' or '.xlsx'
format
```

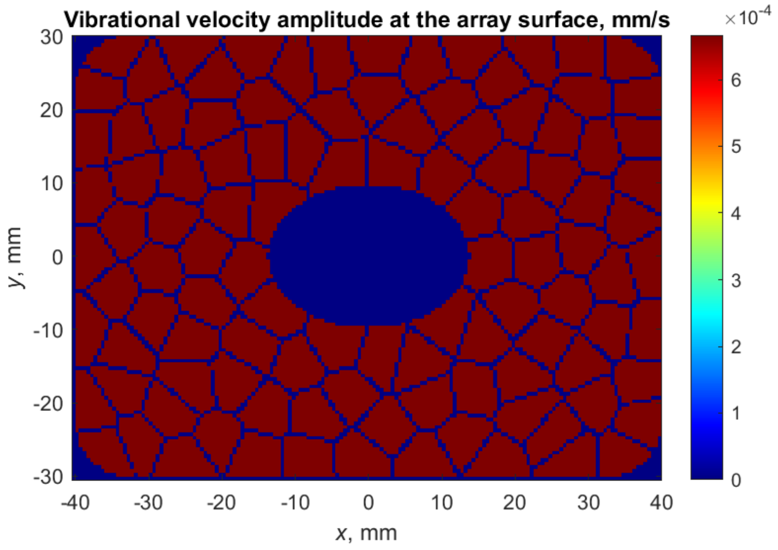
An XLSX-file can also be used as an input for the simulation:

```
inputParametersFileName =
'..\..\data_for_examples\simulation\fpa_steering_10mm_standard_grid.xlsx';
%input with the transducer TransducerSf data structure in '.mat' or '.xlsx'
format
```



3) Coarse grid step model (0.5-mm spatial step)

```
inputParametersFileName =
'..\..\data_for_examples\simulation\fpa_steering_10mm_coarse_grid.mat';      %
input with the transducer TransducerSf data structure in '.mat' or '.xlsx'
format
```



A description of all other input and output parameters, along with visualization tools, can be found in Example 6 of the Holography Toolbox.

Simulation output

- 3D simulation results for the standard grid step (0.25 mm) are shown, along with the velocity distribution at the transducer surface. Users can utilize MATLAB/Octave tools to hide or delete the transducer surface.

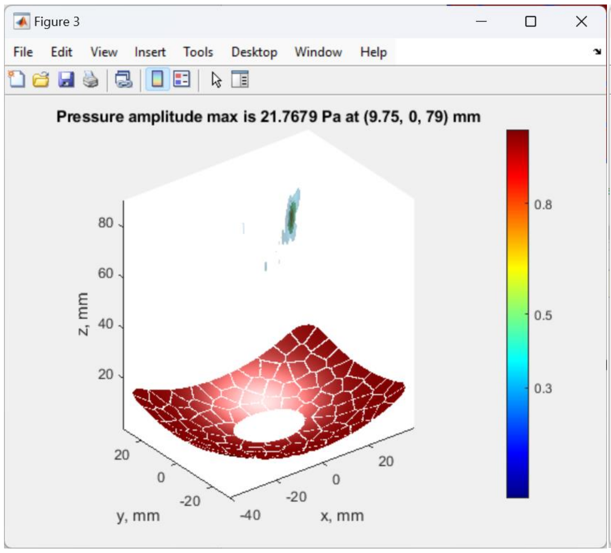


Figure 3
File Edit View Insert Tools Desktop Window Help
Pressure amplitude max is 21.7679 Pa at (9.75, 0, 79) mm
z, mm
y, mm
x, mm
0.8
0.5
0.3

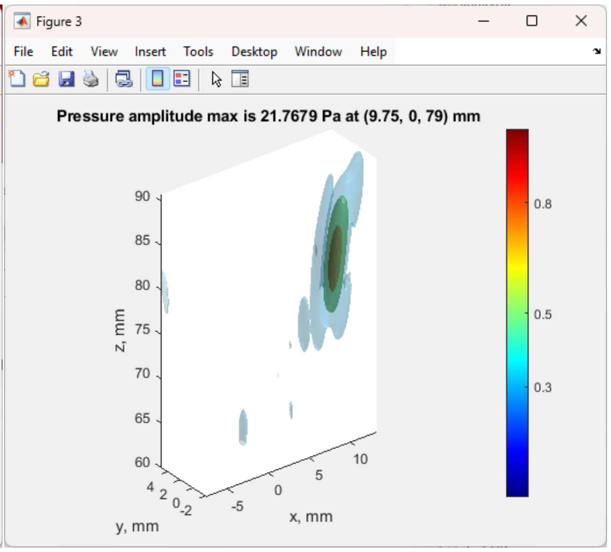


Figure 3
File Edit View Insert Tools Desktop Window Help
Pressure amplitude max is 21.7679 Pa at (9.75, 0, 79) mm
z, mm
y, mm
x, mm
0.8
0.5
0.3

-Slice-by-slice 3D representation of the simulated field (see details in Example 6, Holography Toolbox)

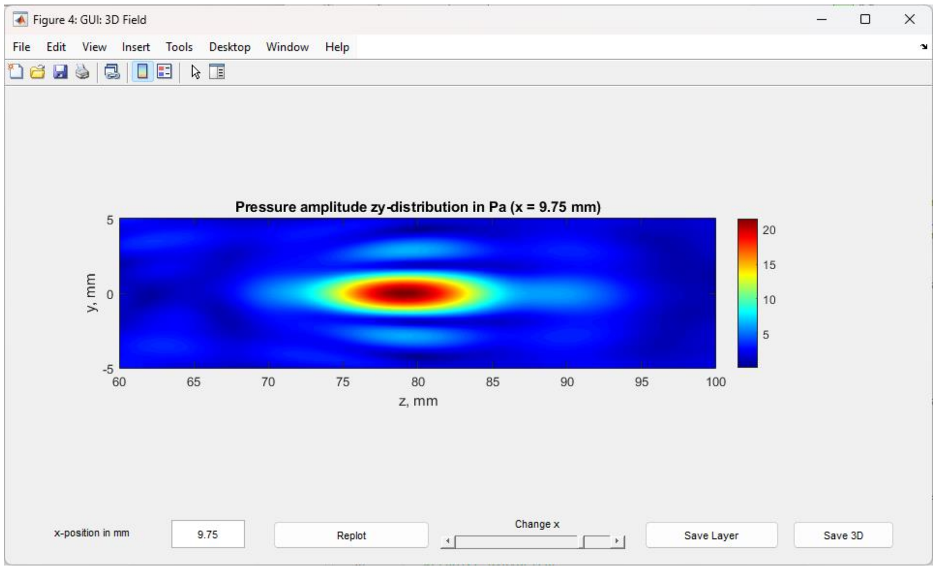


Figure 4: GUI: 3D Field
File Edit View Insert Tools Desktop Window Help
Pressure amplitude zy-distribution in Pa (x = 9.75 mm)
y, mm
z, mm
x-position in mm
9.75
Replot
Change x
Save Layer
Save 3D

Details

The results for fine and coarse grid steps are similar to those of the standard grid step, demonstrating that sub-wavelength differences in the boundary condition do not result in noticeable differences in the far-field pattern.

Fine grid step model (0.125-mm spatial step)

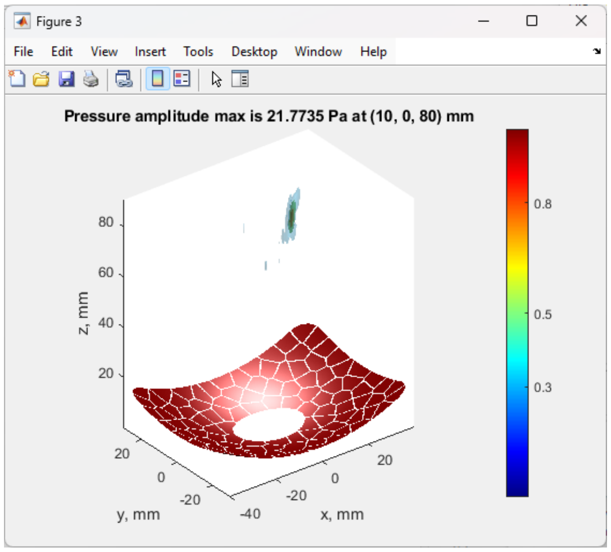


Figure 3
File Edit View Insert Tools Desktop Window Help
Pressure amplitude max is 21.7735 Pa at (10, 0, 80) mm
z, mm
80
60
40
20
20
0
-20
y, mm
-40
-20
0
20
x, mm
0.8
0.5
0.3

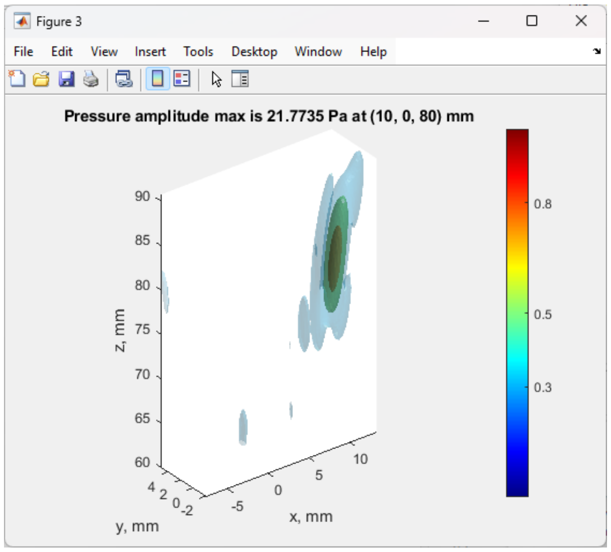


Figure 3
File Edit View Insert Tools Desktop Window Help
Pressure amplitude max is 21.7735 Pa at (10, 0, 80) mm
z, mm
90
85
80
75
70
65
60
4
2
0
-2
y, mm
-5
0
5
10
x, mm
0.8
0.5
0.3

Coarse grid step model (0.5-mm spatial step)

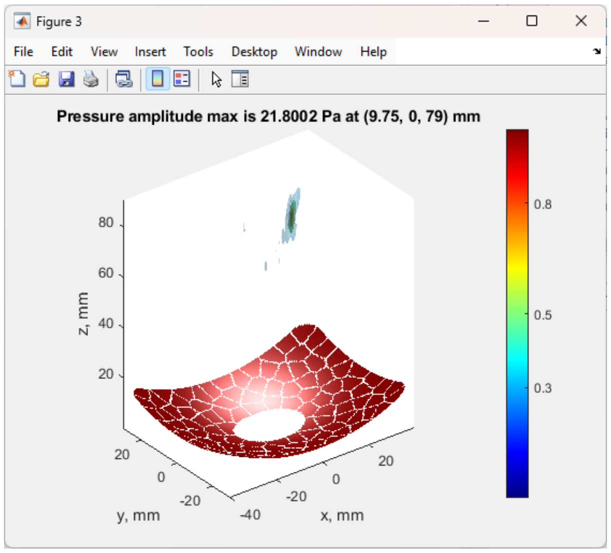


Figure 3
File Edit View Insert Tools Desktop Window Help
Pressure amplitude max is 21.8002 Pa at (9.75, 0, 79) mm
z, mm
80
60
40
20
20
0
-20
y, mm
-40
-20
0
20
x, mm
0.8
0.5
0.3

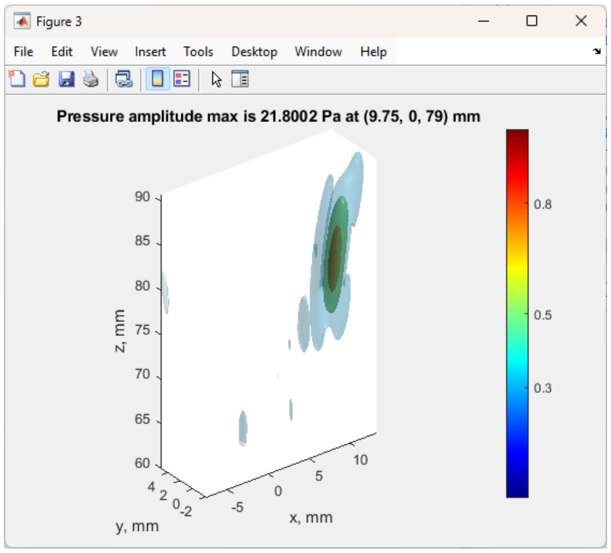
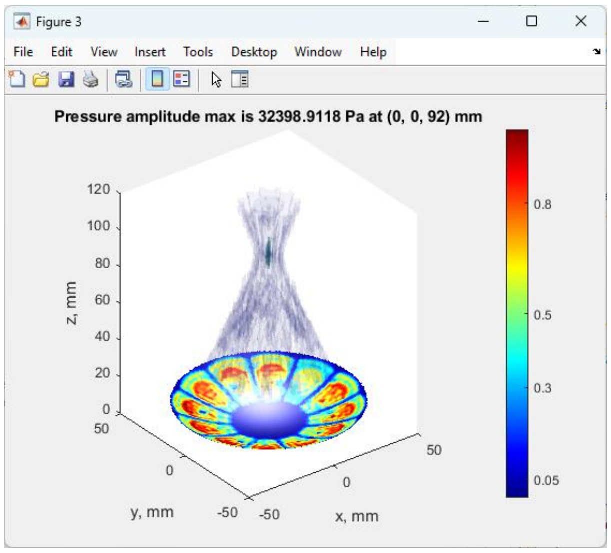
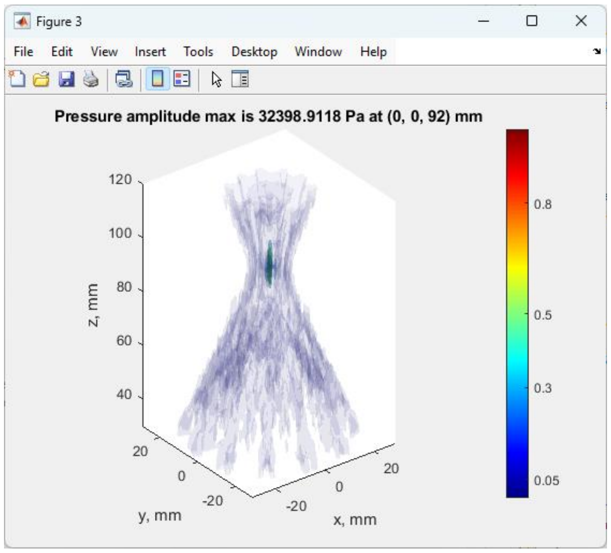


Figure 3
File Edit View Insert Tools Desktop Window Help
Pressure amplitude max is 21.8002 Pa at (9.75, 0, 79) mm
z, mm
90
85
80
75
70
65
60
4
2
0
-2
y, mm
-5
0
5
10
x, mm
0.8
0.5
0.3

Single-Frequency Field Simulation for a Saved Transducer Data

Simulation data for the Single-Frequency Simulation Tool can be loaded from distributions saved in the Holography Toolbox using the "Save TransducerSf" buttons (see Examples 8, 10). An example of a single-frequency simulation using the TransducerSf distribution saved in Example 8 (Simulation Transducer 4: "Real Transducer") is shown below:


Pressure amplitude max is 32398.9118 Pa at (0, 0, 92) mm


Pressure amplitude max is 32398.9118 Pa at (0, 0, 92) mm

Transient Simulation Tool

This tool is designed to simulate 3D, 2D, 1D, or 0D fields of a transducer operating in a transient regime. Transducer parameters are provided by the 'TransducerTr' structure. The output includes a frame-by-frame GUI displaying simulated pressure patterns in 2D simulation regions at different moments in time. An option to render results in video format is also available. For 0D, 1D, and 3D simulations, raw results can be saved by the user and visualized as needed.

Example 13: *Transient Field Simulation for a Saved Transducer Data*

Path: "xDDx\examples\simulation_toolbox\transducer_simulation_transient.m"

General input parameters

- A MAT file containing a transient transducer structure: `TransducerTr` (see page 80).

```
inputParametersFileName = 
'..\data_for_examples\simulation\backprojected_spherical_transient.mat';     % input with the transducer TransducerTr data structure in '.mat' format
```

This example uses the modes `TransducerTr` saved in Example 10 (Simulation Transducer 4: "Real Transducer")

- The rectangular window for field simulation parameters is identical to the one used for forward-projection of a single-frequency hologram (Example 6, Holography Toolbox). Here, we consider an example of a 2D spatial output region *zy*:

```
xFieldBegin = 0e-3; %x, y, and z limits in m for the rectangular simulation region. Set "Begin" equal to "End" for specific coordinates to reduce the number of dimensions.

xFieldEnd   = 0e-3;

yFieldBegin = -5e-3;

yFieldEnd   =  5e-3;

zFieldBegin = 80e-3;

zFieldEnd   = 100e-3;


dxField = []; %x, y, and z grid step in m for the rectangular simulation region. Steps for singleton dimensions (where "Begin" = "End") are ignored and can be omitted.

dyField = 0.2e-3;

dzField = 0.2e-3;
```

- Time domain parameters

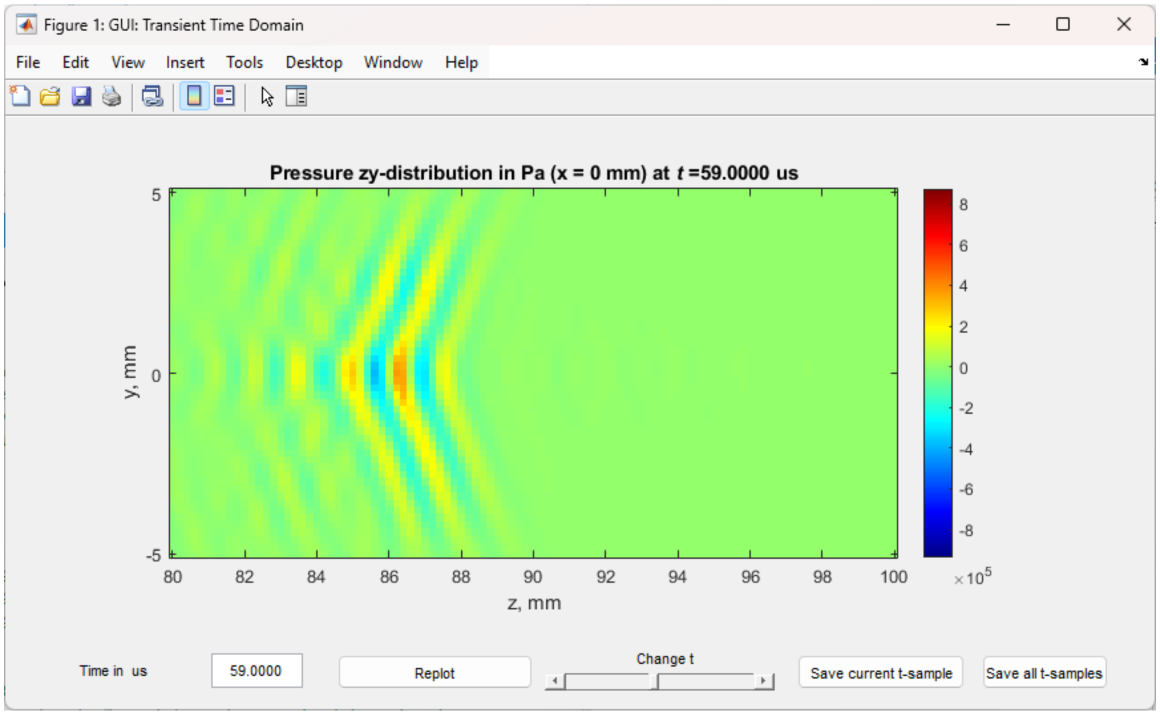
```
timeMax = 'auto';  % the maximum limit of the time window, which can be set either as a numeric value in s or as 'auto' to be determined automatically based on the simulation geometry.

zeroPadData = true; % perform zero-padding of the data if the edge of the input time window is less than timeMax.
```

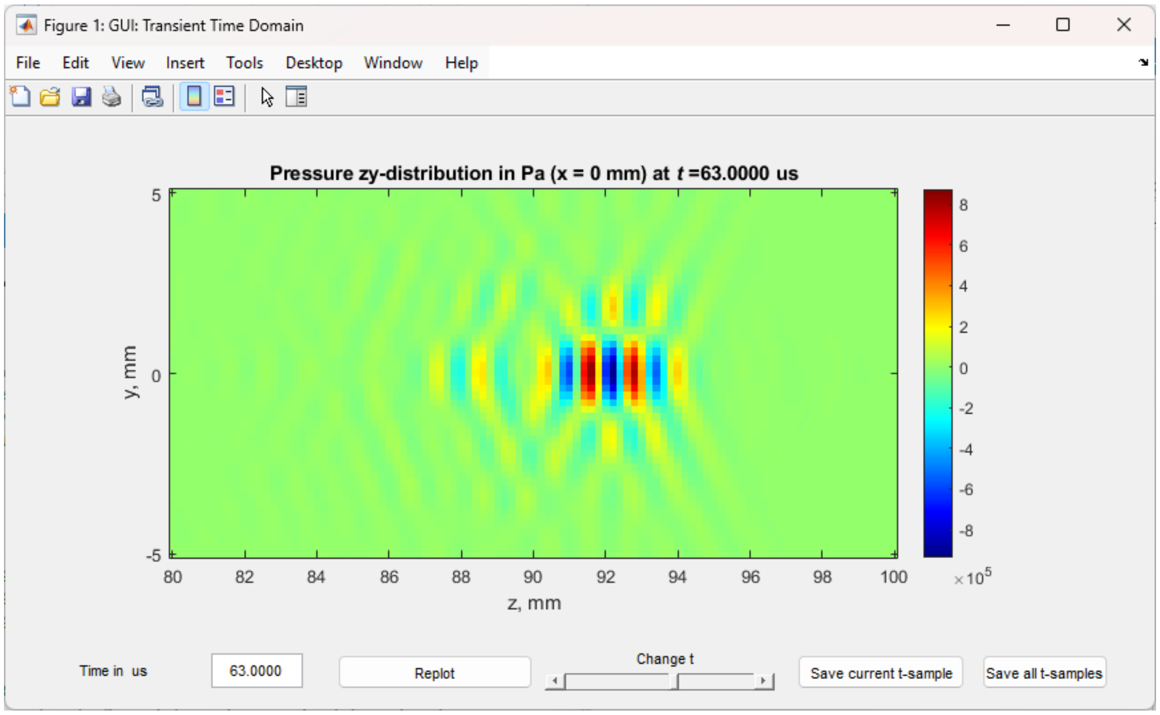
The geometric ideas for time window selection are explained in Fig. 13.

- “Frequency Domain” and “Output Video” parameters are identical to those used in the Holography Toolbox (Examples 6, and 10).

Simulation output

Frame-by-frame representation of simulation results is implemented using the GUI window.

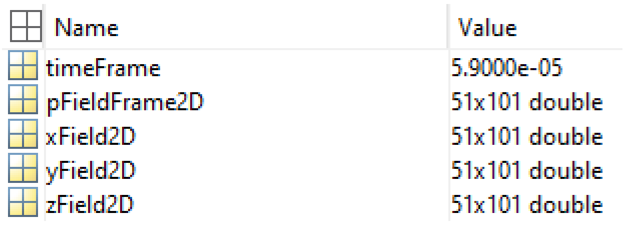




The “Change t” slider or the “Time in us” text box allows adjustments for time selection, and the “Save” buttons provide options to save results in various formats.

Details

“Save current t-samples” saves the parameters that identify the current “frame” for a selected time sample. An example for the case considered is shown below:



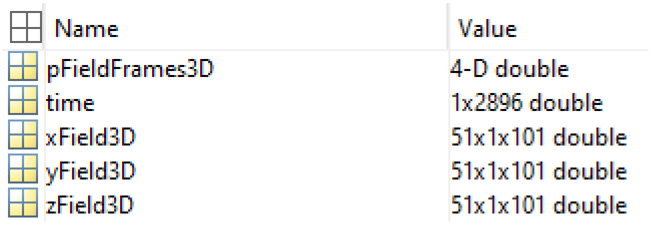
Name	Value
timeFrame	5.9000e-05
pFieldFrame2D	51x101 double
xField2D	51x101 double
yField2D	51x101 double
zField2D	51x101 double

where `timeFrame` is the time sample in s, and the other variables define the saved field in `meshgrid` format.

In this case, all elements of the `xField2D` matrix are identical and represent the x-position of the slice (*x* = 0). The result can be displayed using the `contourf` function:

```
figure;

contourf(zField2D, yField2D, abs(pFieldFrame2D));

axis equal;
```

The “Save all t-samples” button saves all time samples in the most generalized form:



Name	Value
pFieldFrames3D	4-D double
time	1x2896 double
xField3D	51x1x101 double
yField3D	51x1x101 double
zField3D	51x1x101 double

Here, the 4D array `pFieldFrames3D` has dimensions corresponding to the y-, x-, z-, and t-directions, which gives a size of `[51, 1, 101, 2896]` in the given case. In the general case, `size(pFieldFrames3D) = [size(xField3D), length(time)]`. The variable time is a vector of `time` samples, and the other variables are output spatial 3D coordinates in `meshgrid` format.

Boundary Condition Tool

This tool may be useful for back-projecting a boundary condition from a spherical surface to a flat surface can be used to set a boundary condition for spherically shaped transducers on a plane. It can serve as one of the possible simple methods for setting flat boundary conditions for spherically shaped transducers in other simulation software such as k-Wave, FOCUS, and mSOUND.

Transducer parameters are provided by a 'TransducerSf' structure. The script back-projects the vibrational velocity at the surface of the transducer to a plane at the apex of the transducer and then numerically simulates the 3D field based on the obtained flat boundary condition to test its accuracy.

Example 14: *Setting Flat Boundary Condition for a Multi-Element Array*

Path: "xDDx\examples\simulation_toolbox\set_flat_boundary_condition_sf.m"

General input parameters

- Transducer parameters.

A MAT file containing a single-frequency transducer structure: `TransducerSf` (see page 77). This example uses the modes of Simulation Transducer 3 (Fully-Populated-Array). Three models are available for testing. The standard grid step model is used by default. This input is identical to the Single-Frequency Simulation Tool (see Example 12).

- Medium parameters:

```
Medium.soundSpeed = 1500; %sound speed in m/s

Medium.density = 1000; %density in kg/m^3
```

- Flat Boundary Parameters

```
nxFlatBoundary = 181; %number of points for the Cartesian grid of the flat
boundary condition along the x-dimension

nyFlatBoundary = 141; %number of points for the Cartesian grid of the flat
boundary condition along the y-dimension

dxFlatBoundary = 0.7e-3; %x and y grid step in m for the flat boundary condition

dyFlatBoundary = 0.7e-3;
```

In this case the spatial step of 0.7 mm was chosen to satisfy the half-wavelength criteria in water for 1-MHz frequency.

- Rectangular window for field simulation.

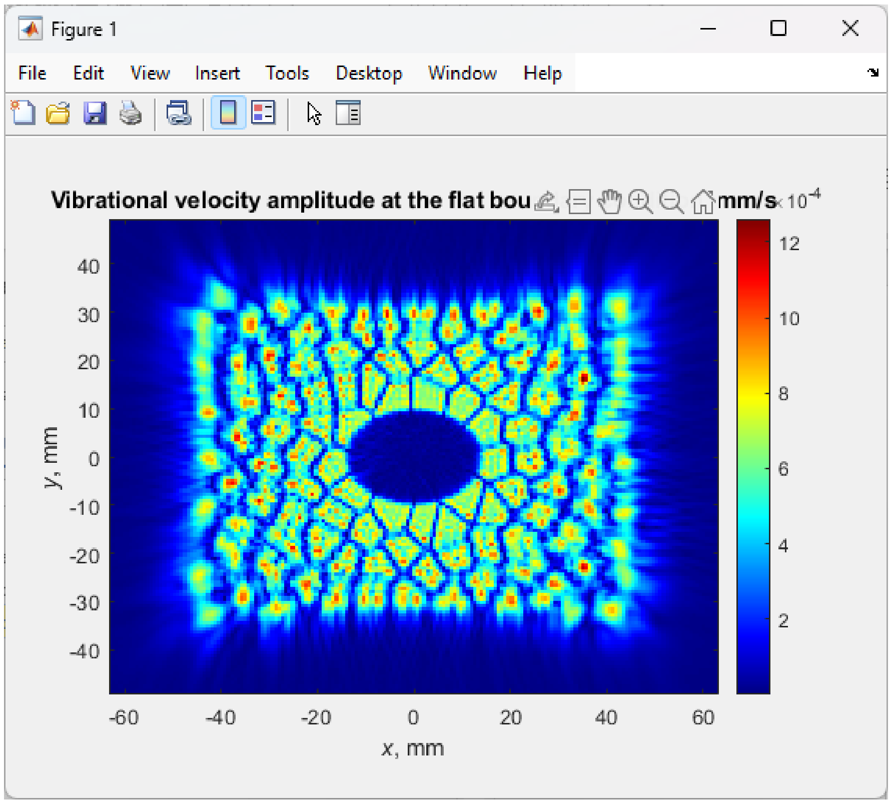
The rectangular window parameters are identical to those used for the forward projection of a single-frequency hologram (Example 6, Holography Toolbox).

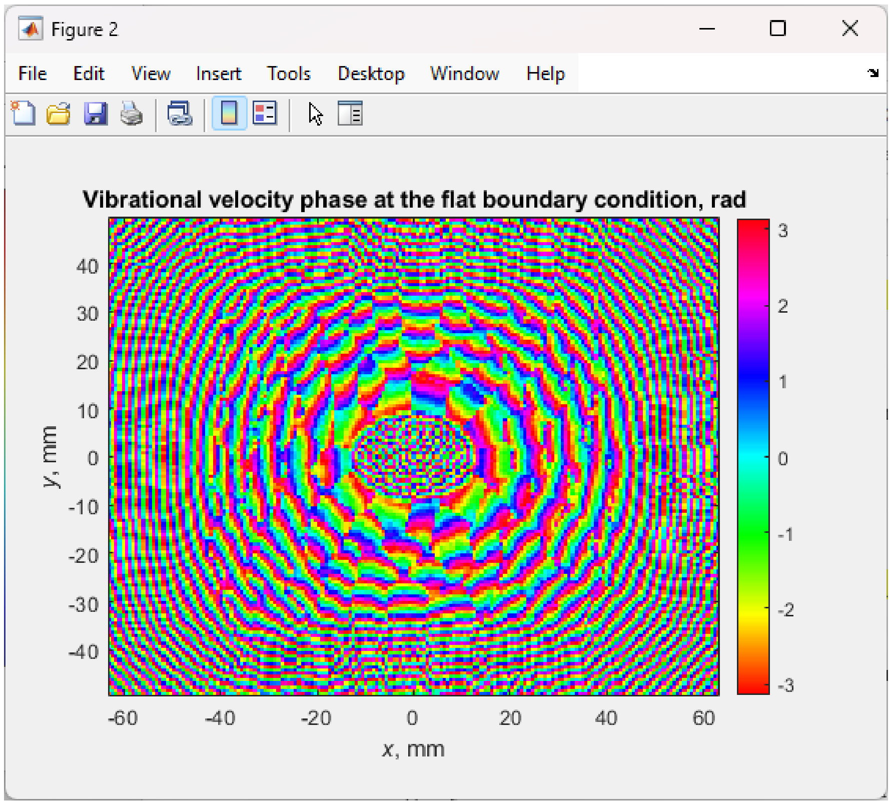
- Service Parameters

```
saveBoundaryCondition = true; %true to save the flat boundary condition
```

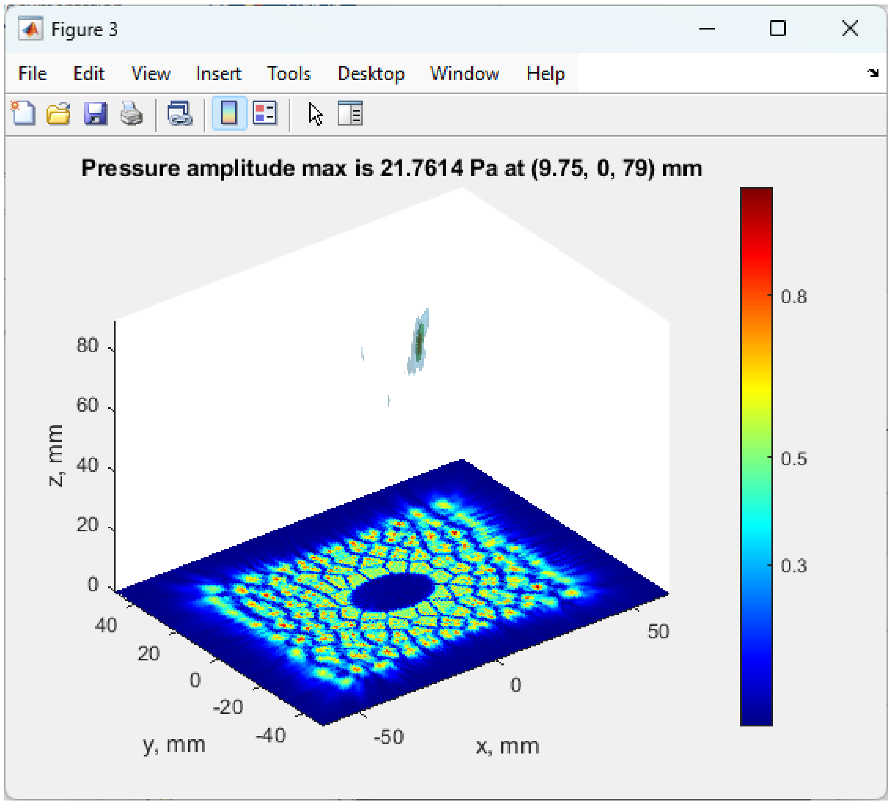
Simulation output

- The simulated boundary condition





-Test field simulated, using the flat boundary condition



Details

- Accuracy

Note that the difference in the maximum pressure compared to the same simulation with the spherical boundary condition (Example 12, standard grid step) is just 0.03%, which is negligible.

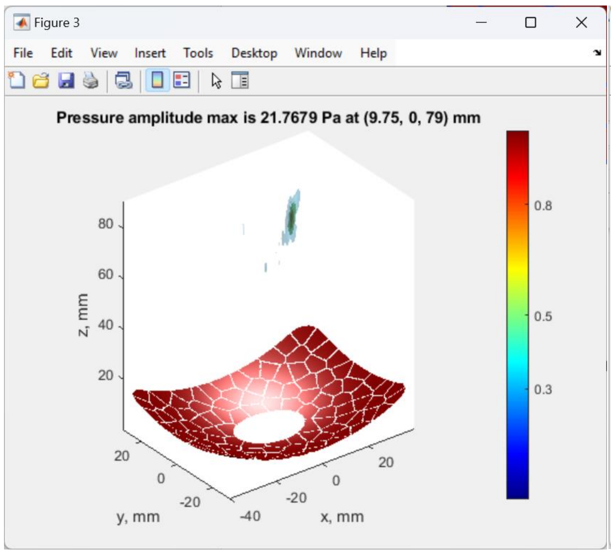


Figure 3
File Edit View Insert Tools Desktop Window Help
Pressure amplitude max is 21.7679 Pa at (9.75, 0, 79) mm
z, mm
y, mm
x, mm

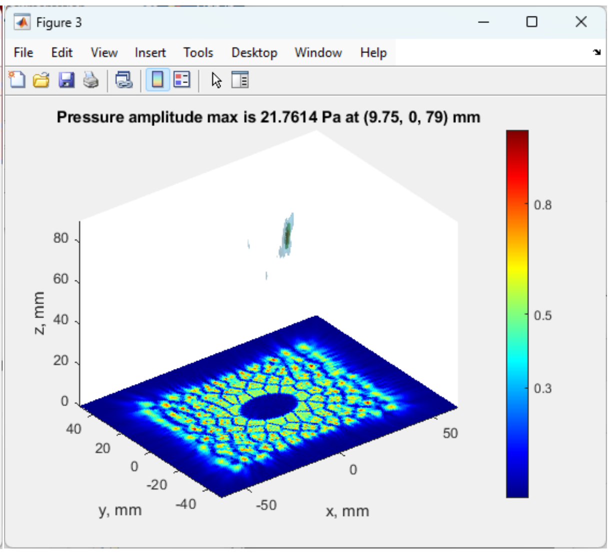


Figure 3
File Edit View Insert Tools Desktop Window Help
Pressure amplitude max is 21.7614 Pa at (9.75, 0, 79) mm
z, mm
y, mm
x, mm

This confirms that the width and length (`nxFlatBoundary * dxFlatBoundary` and `nyFlatBoundary * dyFlatBoundary`) of the flat boundary condition were chosen correctly. If the boundary condition does not capture the entire field, the flat boundary condition will result in a weaker field compared to the initial spherical boundary. Below is an example for `nxFlatBoundary = 141` and `nxFlatBoundary = 101`.

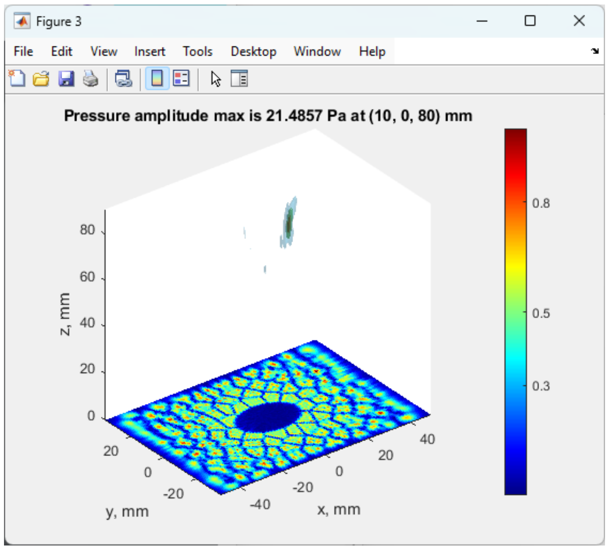


Figure 3
File Edit View Insert Tools Desktop Window Help
Pressure amplitude max is 21.4857 Pa at (10, 0, 80) mm
z, mm
y, mm
x, mm

An important aspect of the flat boundary condition is that the grid step can be larger than the characteristic scale of vibrational velocity heterogeneity at the transducer surface. For instance, the presented array has a gap of 0.5 mm between elements, but the flat boundary condition with a larger step of 0.7 mm still describes the far field accurately. This approach can potentially reduce the total number of grid nodes and thus accelerate the simulation without compromising accuracy.

- Saved boundary condition

If the logical flag `saveBoundaryCondition` is set to true, the boundary condition parameters

`xFlatBoundary`, `yFlatBoundary`, `zFlatBoundary`, `vFlatBoundary`

will be saved along with `TransducerSf` and `Medium` structures in the same directory, with the filename formatted to reflect the simulation completion date and time as:

“flat_boundary_condition_yyyy_mm_dd_hh_mm_ss.mat”

- Using the boundary condition in the k-Wave toolbox

Below is a portion of code that demonstrates how to use this boundary condition in the k-Wave toolbox to simulate acoustic fields in heterogeneous media using the pseudospectral method (http://www.k-wave.org/).

The example assumes a spatial simulation domain with `Nx`, `Ny`, and `Nz` grid points in Cartesian coordinates, with steps equal to the boundary condition step: `dx=dy=dz=dxFlatBoundary=dyFlatBoundary`. The presented grid does not include perfectly matched layer (PML) nodes, which should be set separately around the simulation grid using k-Wave functionalities.

The number of time grid points is denoted as `Nt`, with the time vector defined as `timeVector = (0:(Nt-1))*dt`. Here, `dt` is the time step. It is assumed that, even though the medium might be heterogeneous, the surface of the transducer is located in a homogeneous medium defined by the `Medium` structure.

The k-Wave acoustic source structure `source` in `'additive'` mode is used to set this boundary condition for a k-Wave simulation.

A specific description of the structure can be found in the k-Wave documentation (http://www.k-wave.org/manual/k-wave_user_manual_1.1.pdf).

```
%Select the source mode

source.p_mode = 'additive'; %optional, because this mode is default


%Set the boundary condition mask at the first sample of the z-grid

source.p_mask = false(Nx, Ny, Nz);

source.p_mask(:,:,1) = true;


%Transpose the boundary condition, as k-Wave uses the Nx x Ny dimension order,
%while xDDx uses the transposed Ny x Nx order (meshgrid)

vFlatBoundary = transpose(vFlatBoundary);


%Extract and normalize the amplitude from the boundary condition on the plane.
%A detailed explanation of this normalization is given in [Rosnitskiy et al.,
%IEEE UFFC, DOI: 10.1109/TUFFC.2024.3355390]

density = Medium.density;

soundSpeed = Medium.soundSpeed;

amplitude = density*soundSpeed*transpose(abs(vFlatBoundary));

phase = angle(vFlatBoundary);


%Define the additive acoustic sources as a matrix of size Nx*Ny by Nt

frequency = Transducer.Sf.frequency;

source.p = amplitude(:).*cos(2*pi*frequency*timeVector + phase(:));
```

Validation Tool

This tool is designed to allow users to test the accuracy of simulation algorithms presented in the toolbox. The tool is almost identical to the Single-Frequency Simulation Tool (Example 12), but it automatically generates a model of an idealized spherical (Simulation Transducer 2, Fig. 14), or a flat (Simulation Transducer 1, Fig. 14) piston transducer. The 3D field of the source is simulated numerically using the toolbox, and the numerical result along the transducer axis is compared to the analytical solution [O'Neil 1949]. Users can test different transducer models with varying sizes, frequencies, and grid discretization steps to test simulation precision for cases of interest.

Example 15: *Idealized Spherical Source Simulation*

Path:
“xDDx\examples\simulation_toolbox\validation_tools\spherical_piston_simulation_sf.m”

General input parameters

- Transducer Parameters

```
aperture = 80e-3; %in m

radiusOfCurvature = 80e-3; %in m

frequency = 1e6; %in Hz

sourceStepX = 0.7e-3;  %x grid step for source representation in m

sourceStepY = 0.7e-3;  %y grid step for source representation in m

initialVelocity = 1/Medium.soundSpeed/Medium.density; %vibrational velocity at
the surface of the source in m/s
```

See the description of the default parameters on page 81.

The other input parameters are identical to those in the Single-Frequency Simulation Tool (Example 12).

Simulation output

- The front view projection of the boundary condition with given discretization steps `sourceStepX` and `sourceStepY`

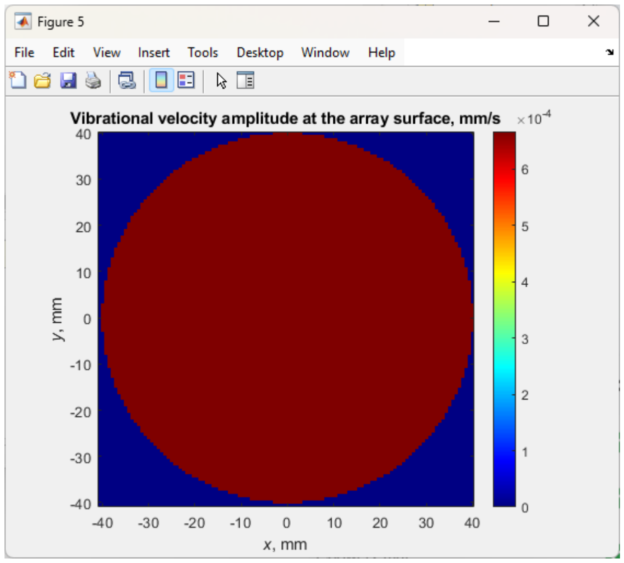


Figure 5
File Edit View Insert Tools Desktop Window Help
Vibrational velocity amplitude at the array surface, mm/s
$\times 10^{-4}$
y, mm
x, mm

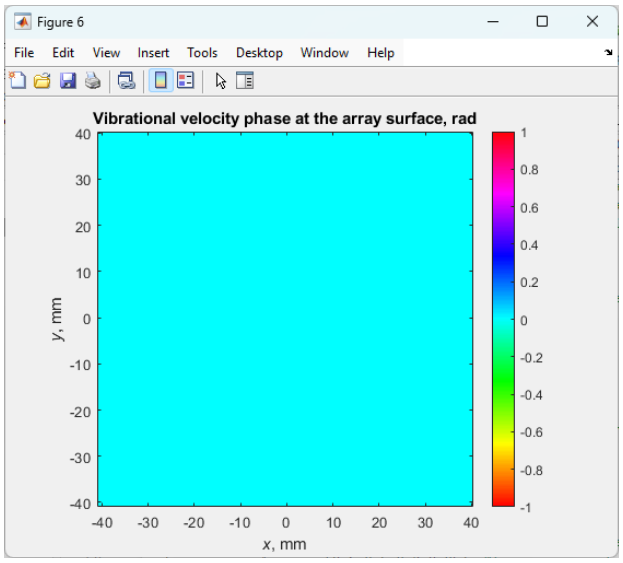


Figure 6
File Edit View Insert Tools Desktop Window Help
Vibrational velocity phase at the array surface, rad
y, mm
x, mm

- Simulated field in 3D

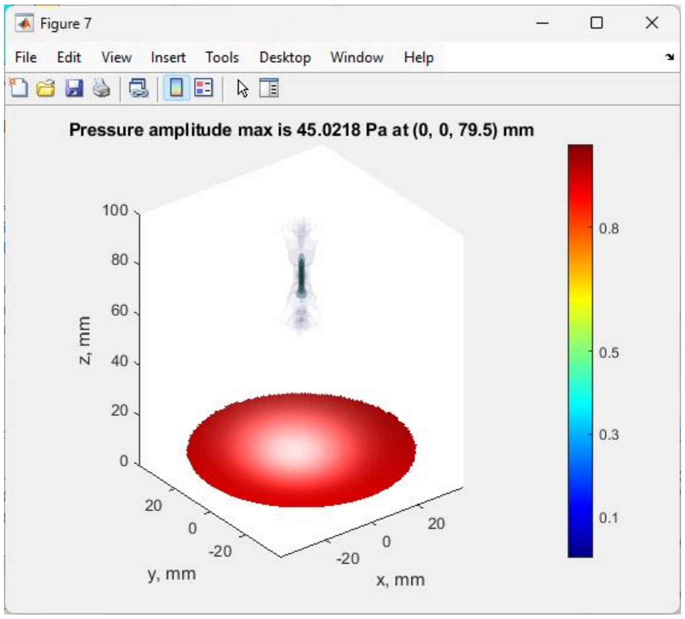
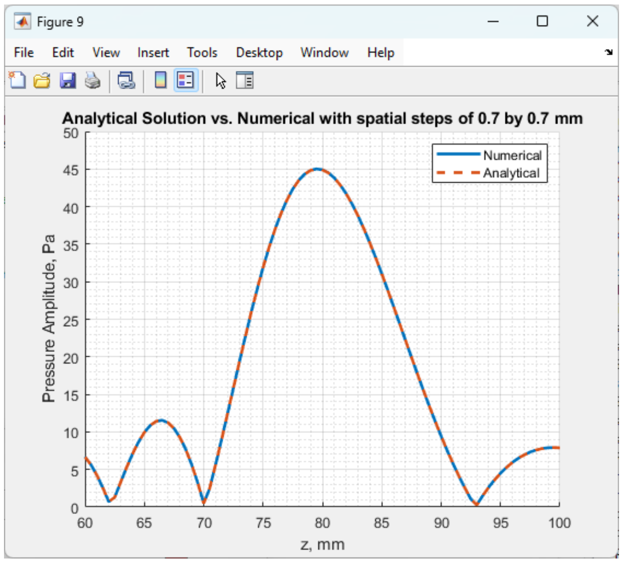
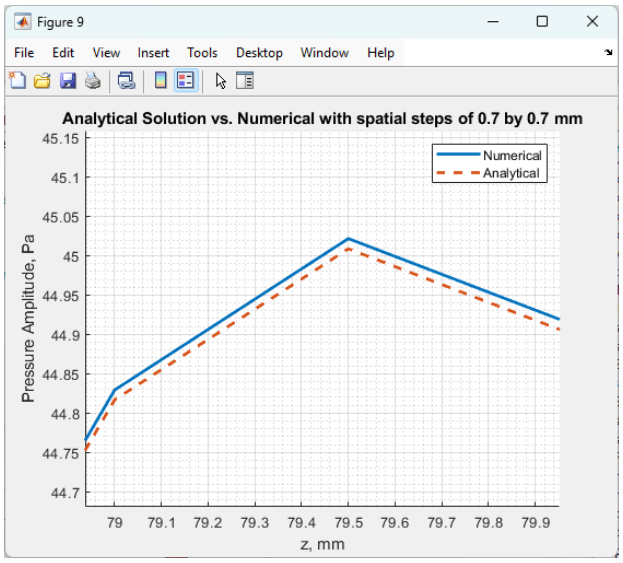


Figure 7
File Edit View Insert Tools Desktop Window Help
Pressure amplitude max is 45.0218 Pa at (0, 0, 79.5) mm
z, mm
y, mm
x, mm
100
80
60
40
20
0
20
0
-20
0.8
0.5
0.3
0.1

- Validation in 1D





The error in the maximum amplitude for the discretization step of 0.7 mm is 0.03% which is negligible.

Example 16: *Idealized Flat Source Simulation*

Path:
"xDDx\examples\simulation_toolbox\validation_tools\spherical_piston_simulation_sf.m"

General input parameters

- Transducer Parameters

```
aperture = 20e-3; %in m

frequency = 1e6; %in Hz

sourceStepX = 0.7e-3;  %x grid step for source representation in m

sourceStepY = 0.7e-3;  %y grid step for source representation in m

initialVelocity = 1/Medium.soundSpeed/Medium.density; %vibrational velocity at
the surface of the source in m/s
```

The other input parameters are identical to the previous Example 15.

Simulation output

- The front view projection of the boundary condition with given discretization steps `sourceStepX` and `sourceStepY`

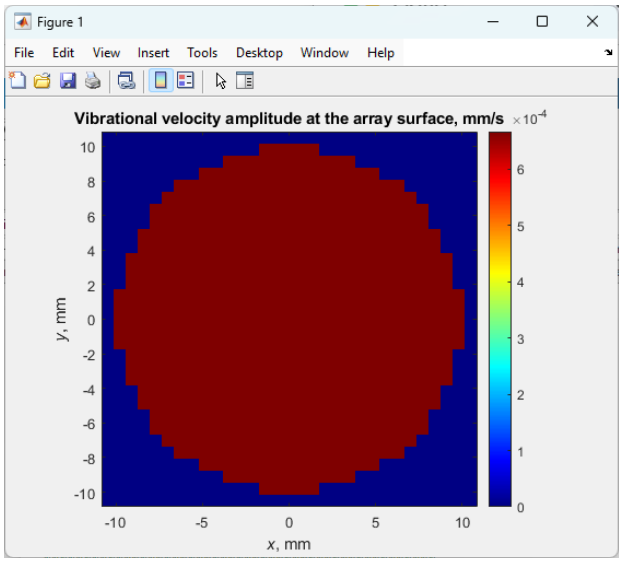


Figure 1
File Edit View Insert Tools Desktop Window Help
Vibrational velocity amplitude at the array surface, mm/s
×10⁻⁴
y, mm
x, mm

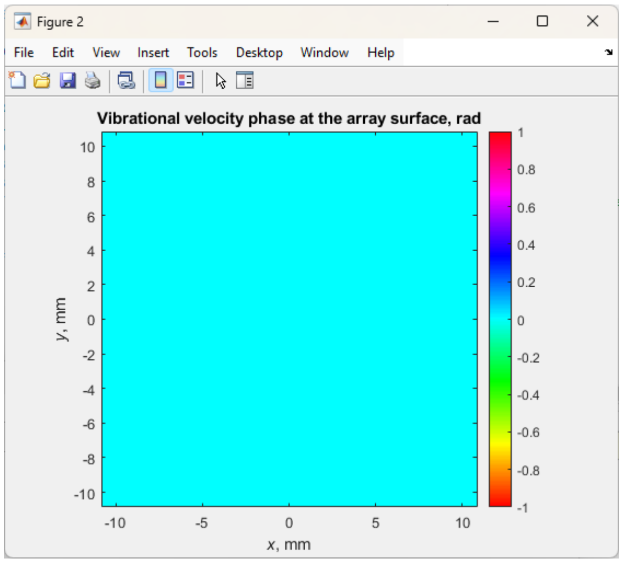


Figure 2
File Edit View Insert Tools Desktop Window Help
Vibrational velocity phase at the array surface, rad
y, mm
x, mm

- Simulated field in 3D

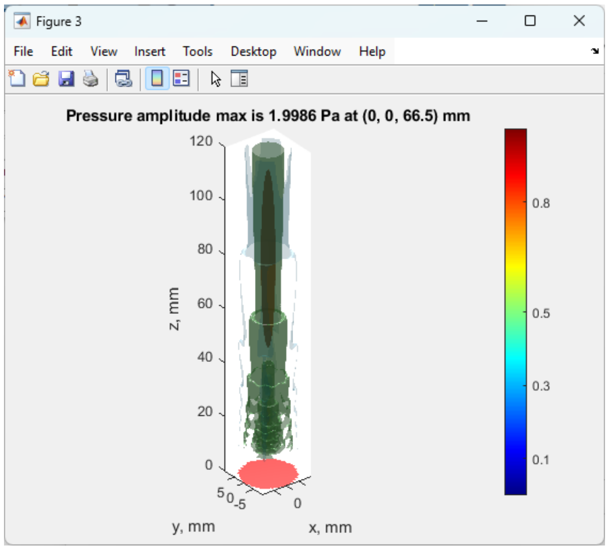
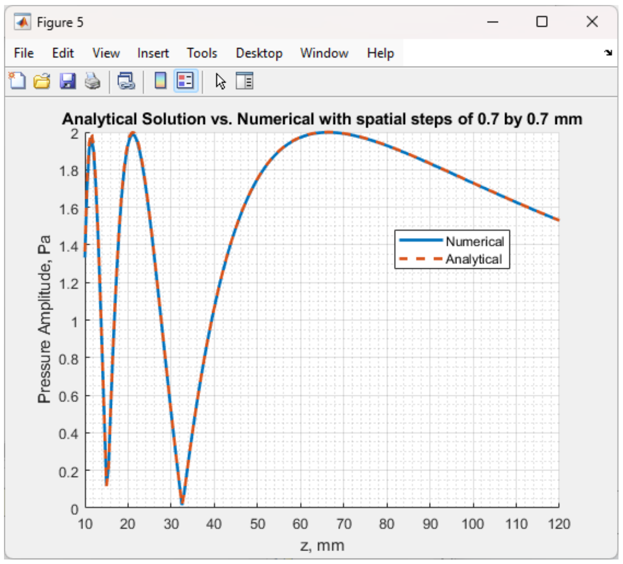
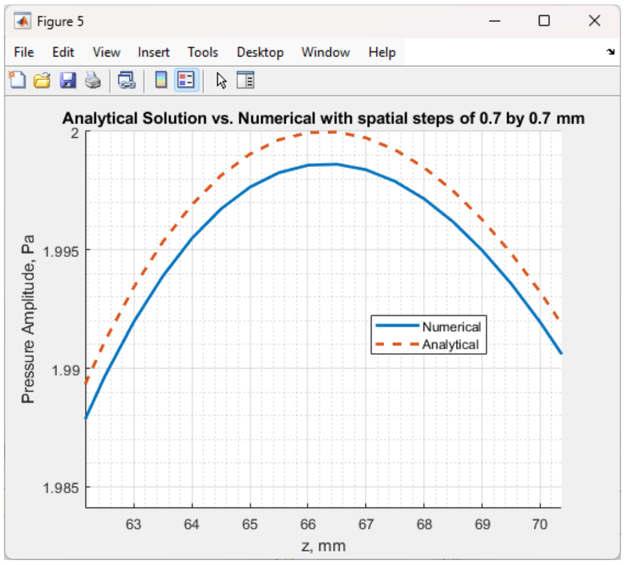


Figure 3
File Edit View Insert Tools Desktop Window Help
Pressure amplitude max is 1.9986 Pa at (0, 0, 66.5) mm
z, mm
y, mm
x, mm
120
100
80
60
40
20
0
5
0
-5
0
0.8
0.5
0.3
0.1

- Validation in 1D





Details

Even though the transducer model with a 0.7-mm discretization step is noticeably pixelated, it still results in a negligible error of 0.06% in the pressure amplitude maximum. A finer discretization of 0.25 mm further improves the pixelization, reducing the error to 0.004%.

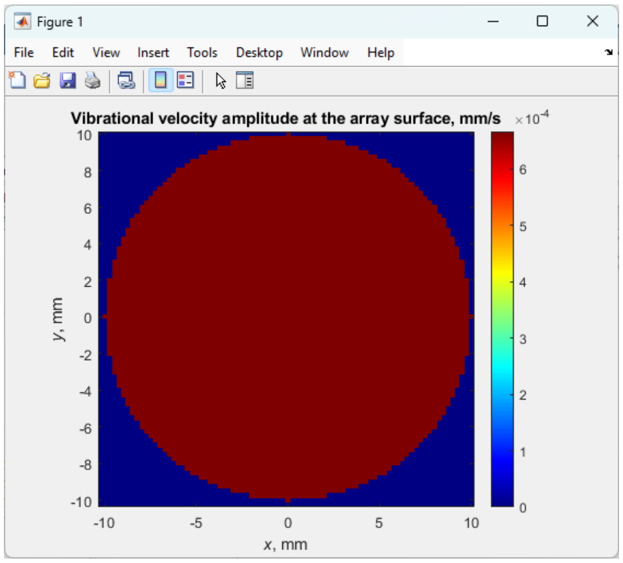


Figure 1
File Edit View Insert Tools Desktop Window Help
Vibrational velocity amplitude at the array surface, mm/s
$\times 10^{-4}$
y, mm
x, mm

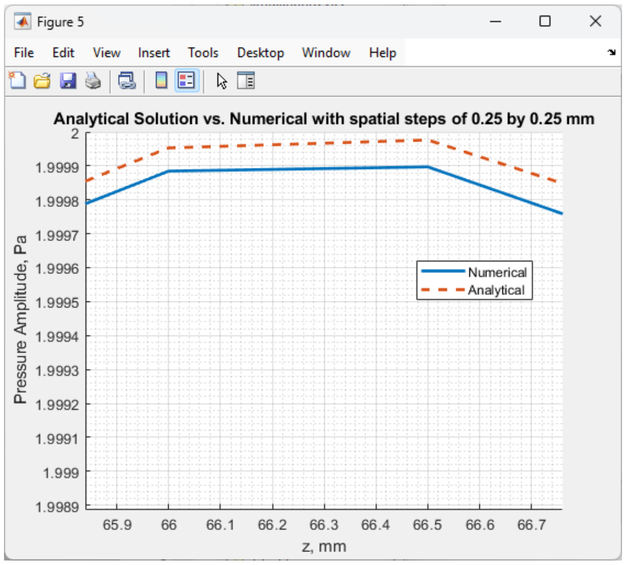


Figure 5
File Edit View Insert Tools Desktop Window Help
Analytical Solution vs. Numerical with spatial steps of 0.25 by 0.25 mm
Pressure Amplitude, Pa
z, mm
Numerical
Analytical

MAKE YOUR OWN TOOLS

This chapter describes the core function of the toolbox `rayleigh_simulator` which is a universal tool applicable to Rayleigh integral-based forward/back projection of either a single-frequency boundary condition or a set of single-frequency components (the "transient regime"). This function has flexible input and output options for either pressure or velocity on either a spherically curved or flat surface.

The core serves as a base for all presented xDDx tools facilitating different sequencies of forward and back projections. Nevertheless, even though the existing tools cover many needs for transducer diagnostics and field simulation, users can create their own tools based on the core function to address specific cases.

The `rayleigh_simulator` is located in the xDDx-library folder "xDDx\xDDx_lib\rayleigh_simulator.m". The text below explains the inputs and outputs of the function, and the explanation is presented in MATLAB/Octave format for consistent representation of variables.

```
USAGE:

[outputField] = rayleigh_simulator(expSign, frequencyParameter, regime, simulationDevice, isTransient, SourceParameters, FieldParameters, Medium, ServiceParameters, radiusOfCurvature)


INPUTS:

expSign              +1 or -1, depending on the exponent sign convention exp(+1i * omega * t) or exp(- 1i * omega * t). E.g., the "fft" function in MATLAB utilizes the  exp(+ 1i * omega * t) convention, so in this case, expSign is +1


frequencyParameter    frequencyParameter = frequency of the transducer in Hz for a single-frequency hologram, or frequencyParameter = frequencyStep for a transient hologram

regime               simulation regime number from 1 to 6, see details below*

simulationDevice     'cuda' - perform simulation using CUDA compatible videocard (GPU); 'cpu'  - perform simulation using the central processor (CPU) of the computer

isTransient            true for the transient regime, false for the single-frequency regime

SourceParameters      input or output (depending on the regime) structure that describes the Source, see details below**

FieldParameters       input or output (depending on the regime) structure that describes the Field, see details below**
```

Medium structure with medium parameters, see details below***

radiusOfCurvature (optional, can be omitted if unnecessary) radius of curvature in m of the input spherical surface of integration in regimes 3 and 6. Omit this variable if your transducer is flat.

ServiceParameters (optional, can be omitted if unnecessary) structure with service parameters, see details below. If omitted, the default ServiceParameters are set.****

OUTPUT:

outputField matrix of acoustic pressure/vibrational velocity complex amplitude at the Source/Field surface (depending on the regime), see details regarding the matrix size below*****

***REGIMES:**

1 Back-projection: P on a plane --> V on a plane

2 Back-projection: P on a plane --> V on a sphere

3 Forward-projection: V on a sphere --> P at an arbitrary set of points

4 Forward-projection: V on a plane --> P at an arbitrary set of points

5 Forward-projection: P on a plane --> P at an arbitrary set of points

6 Back-projection: V on a sphere --> P on a planes

****SIMULATION PARAMETERS FORMAT:**

'SourceParameters' and **'FieldParameters'** structures that may contain the following fields

'xGrid' (necessary field) vector or matrix with the x-coordinates in m at each grid node of the Source or Field region

'yGrid' (necessary field) vector or matrix with the y-coordinates in m at each grid node of the Source or Field region

'zGrid' (necessary field): vector or matrix with the z-coordinates in m at each grid node of the Source or Field region

'dx' (optional field): x-step of the Source or Field Cartesian grid in m

'dy' (optional field): y-step of the Source or Field Cartesian grid in m

```
'input'(optional field): input complex pressure or velocity amplitude area for
integration surface in Pa or m/s, see details regarding the matrix size
below*****


***MEDIUM PARAMETERS:

 'Medium' struct with fields:

    'soundSpeed': sound speed in m/s

     'density': density in kg/m^3


****SERVICE PARAMETERS:

'ServiceParameters'  (optional) struct with fields:

     'threadsPerBlockGPU': number of threads per block for GPU if applicable.
Default value is 128.


*****INPUT/OUTPUT FIELD MATRIX:

for a single-frequency hologram (isTransient = false):

input/output complex amplitudes are given for each node of the the input/output
grid

i.e. size(input) = size(xGrid)


for a transient hologram (isTransient = true):

input/output complex amplitudes are given for each node of the the input/output
grid in a range of frequencies (1:numberOfFrequencySamples)*frequencyStep i.e.
size(input) = [size(xGrid) numberOfFrequencySamples]
```

Thus, all the xDDx tools introduced in previous chapters call the core function one or more times to perform simulations and display the results. For example, the Automatic Alignment Tool includes Regime 5 for focal lobe detection and planar scan repositioning, and Regime 2 for back-projection at the transducer surface.

The portion of code below shows the simplest case of usaing Regime 5 in the Single-Frequency Forward-Projection Tool. It projects the acoustic pressure amplitude of a single-frequency hologram (`HologramSf` structure, page 16) given by the coordinate matrices `xHolo`, `yHolo`, `zHolo` to the nodes of a 3D rectangular region `xField3D`, `yField3D`, `zField3D`.

```
%Set geometric parameters

xHolo = HologramSf.xGrid;

yHolo = HologramSf.yGrid;

zHolo = HologramSf.zPosition * ones(size(xHolo));


%Forward-project the hologram to calculate the complex pressure amplitude at
%the nodes of the grid

SourceParameters = [];

SourceParameters.xGrid = xHolo;

SourceParameters.yGrid = yHolo;

SourceParameters.zGrid = zHolo;

SourceParameters.dx = HologramSf.dx;

SourceParameters.dy = HologramSf.dy;

SourceParameters.input = HologramSf.complexPressureAmplitude;


FieldParameters = [];

FieldParameters.xGrid = xField3D;

FieldParameters.yGrid = yField3D;

FieldParameters.zGrid = zField3D;

regime = 5; % 5 Forward-projection: P on a plane --> P at an arbitrary set of
points

isTransient = false; %set to true for the transient regime, false for the
single-frequency regime


%Rayleigh simulator function with the complex pressure amplitude output

[ pField3D ] = rayleigh_simulator(HologramSf.expSign, HologramSf.frequency,
regime, simulationDevice, isTransient, SourceParameters, FieldParameters,
Medium, ServiceParameters);
```

Users can find examples of other regimes in the scripts describing the existing tools and combine projections as needed for their own specific tasks.

REFERENCES

P. B. Rosnitskiy, O. A. Sapozhnikov, V. A. Khokhlova, W. Kreider, S. A. Tsysar, G. P. L. Thomas, K. Contreras, and T. D. Khokhlova, “xDDx: a Numerical Toolbox for Ultrasound Transducer Characterization and Design with Acoustic Holography,” IEEE Trans. Ultrason., Ferroelectr., Freq. Control (Early Access) 2025.

O. A. Sapozhnikov, S. A. Tsysar, V. A. Khokhlova, and W. Kreider, “Acoustic holography as a metrological tool for characterizing medical ultrasound sources and fields,” The Journal of the Acoustical Society of America, vol. 138, no. 3. Acoustical Society of America (ASA), pp. 1515–1532, Sep. 01, 2015.

O. A. Sapozhnikov, A. E. Ponomarev, and M. A. Smagin, “Transient acoustic holography for reconstructing the particle velocity of the surface of an acoustic transducer,” Acoust. Phys., vol. 52, no. 3, pp. 324–330, 2006.

S. A. Tsysar, D. A. Nikolaev, and O. A. Sapozhnikov, “Broadband Vibrometry of a Two-Dimensional Ultrasound Array Using Transient Acoustic Holography,” Acoustical Physics, vol. 67, no. 3. Pleiades Publishing Ltd, pp. 320–328, May 2021.

D. Maxwell, K. J. Haworth, C. K. Holland, S. A. Hendley, W. Kreider, and K. B. Bader, “Design and Characterization of an Ultrasound Transducer for Combined Histotripsy-Thrombolytic Therapy,” IEEE Trans. Ultrason., Ferroelect., Freq. Control, vol. 69, no. 1, pp. 156–165, Jan. 2022.

H. T. O’Neil, “Theory of Focusing Radiators,” The Journal of the Acoustical Society of America, vol. 21, no. 5. Acoustical Society of America (ASA), pp. 516–526, Sep. 01, 1949.

Z. Kaloev, D. A. Nikolaev, V. A. Khokhlova, S. A. Tsysar, and O. A. Sapozhnikov, “Spatial Correction of an Acoustic Hologram for Reconstructing Surface Vibrations of an Axially Symmetric Ultrasound Transducer,” Acoustical Physics, vol. 68, no. 1. Pleiades Publishing Ltd, pp. 71–82, Feb. 2022.

O. A. Sapozhnikov, Yu. A. Pishchal’nikov, and A. V. Morozov, “Reconstruction of the normal velocity distribution on the surface of an ultrasonic transducer from the acoustic pressure measured on a reference surface,” Acoustical Physics, vol. 49, no. 3. Pleiades Publishing Ltd, pp. 354–360, May 2003.

O. A. Sapozhnikov and M. R. Bailey, “Radiation force of an arbitrary acoustic beam on an elastic sphere in a fluid,” The Journal of the Acoustical Society of America, vol. 133, no. 2. Acoustical Society of America (ASA), pp. 661–676, Jan. 30, 2013.

P. B. Rosnitskiy, B. A. Vysokanov, L. R. Gavrilov, O. A. Sapozhnikov, and V. A. Khokhlova, “Method for designing multielement fully populated random phased arrays for ultrasound surgery applications,” IEEE Trans. Ultrason., Ferroelectr., Freq. Control, vol. 65, no. 4, pp. 630–637, Apr. 2018.