Acoustic

Wide-anlge mode parabolic equation (WAMPE) solver for Helmholtz equation.

1 Getting started

1.1 Requirements

The following is required to build Acoustic from scratch

- C++ compiler with C++17 support (gcc is recommended)
- CMake
- ALGLIB
- BOOST C++ Libraries
- nlohmann::json
- fftw3
- Included as git submodules
  - Acoustics-at-home
  - DORK
  - delaunay

1.2 Building

You can build ACOUSTIC with CMake as follows

```
$ cd build/cmake
$ mkdir ACOUSTIC && cd ACOUSTIC
$ cmake -DCMAKE_BUILD_TYPE=Release ..
```

2 Usage

Solver utilizes command line interface as follows

```
$ ACOUSTIC [task] [[option1] [option2] ...]
```

2.1 Tasks

Currently the following tasks are supported

- `solution (default)` Compute WAMPE solution
- `impulse` Compute acoustic impulse at receivers
- `modes` Compute wavenumbers and modal functions
- `rays` Compute acoustic rays
- `init` Compute initial conditions

2.2 Options

Currently the following options are supported

2.2.1 General options

- `-h [ --help ]` Print help message
- `-v [ --verbosity ] arg` Set verbosity level to `arg`. The following values are supported (bigger values include all previous)
  - `0` nothing (default value)
  - `1` show execution time
  - `2` show progress bar
  - `3` print configuration information
- `-r [ --report ] k` Only affects `solution` task. If verbosity level > 0 report every `k` computed rows. Prints nothing if set to `0` (default)
- `-c [ --config ] arg` Specifies path to configuration file. Default is `config.json`

2.2.2 Output options

- `-o [ --output ] filename` Specifies path to output file. Default is `output.txt`
- `-s [ --step ] k` Output every `k`-th computed row. Default is `100`
- `--binary` Switches to binary output

2.2.3 Computational options

- `-w [ --workers ] n` Sets the number of threads for computation. Only affects `solution` and `impulse` tasks. Be default uses one thread
- `-b [ --buff ] arg` Sets buffer size for multithreaded computations. Default is `100`

3 Configuration file

The configuration file is stored in JSON format with any of the following fields

3.1 General format

Real valued data is specified as one floating point value. Complex data is stored as two consecutive floating point values

3.2 Binary data types

- `uint32` Unsigned 32-bit integer
- `double` IEEE 754 double value
- `complex` Two `double values`

3.3 In-file data

Data stored in a file can be specified as follows

3.3.1 Text file

```
{
    "field": [
        "text_file",
        "filename"
    ]
}
```

3.3.2 Binary file

```
{
    "field": [
        "binary_file",
        "filename"
    ]
}
```

3.4 Table data

3.4.1 Text data

The first row and column are coordinates values. The first coordinate follows columns, the second — rows. Values should be (but are not required to) separated with spaces

```
0    y0 ...  yM
x0  v00 ... v0M
... ... ... ...
xN  vN0 ... vNM
```

3.4.2 Binary data

The binary table data is stored as follows

Type	Length	Description
`uint32`	`1`	Unsigned number `N`
`uint32`	`1`	Unsigned number `M`
`double`	`N`	First coordinate values
`double`	`M`	Second coordinate values
`double`	`NM`	Data values

3.4.3 In-place data

Table data can be specified directly in a configuration file. `x`, `y` are first and second coordinate names respectively

```
{
    "field": [
        "values",
        {
            "x": [],
            "y": [],
            "values": [ [], [], ]
        }
    ]
}
```

3.5 Floating-point fields

- `"mode_subset"` Value in range `[-1, 1]`, used to truncate computed modes
- `"x0", "x1"` Domain border over `x` coordinate. `x0` is only used for ray starters otherwise is `0`
- `"y0", "y1"` Domain borders over `y` coordinate
- `"y_s", "z_s"` `y` and `z` coordinates of the source
- `"tolerance"` For impulse computation values less than `tolerance * max(spectre)` are skipped
- `"a0", "a1"` Min and max radian angles used for ray starters
- `"l0", "l1"` Min and max natural parameters used for ray starter

3.6 Integer fields

- `"max_mode"` Maximal number number of modes to use (-1 uses as many as there are)
- `"n_modes"` Use this many modes (`"max_mode"` still takes effect)
- `"nx", "ny"` Number of points over `x` and `y` coordinates. Only affects `solution` and `impulse` tasks
- `"ppm"` Number of points for modes computation

- `"mnx"`, `"mny"` Number of points over `x` and `y` coordinates for modes computation
- `"ordRich"` Order of Richardson extrapolation
- `"n_layers"` Number of water layers
- `"past_n"` History length for transparent boundary conditions
- `"border_width"` Width of smoothed areas over left and rights domain borders. Should be less than `ny / 2`
- `"na"` Number of angular point for ray starters
- `"nl"` Number of natural parameter points for ray starter

3.7 Boolean fields

- `"complex_modes"` Uses complex-valued modes (accounts for attenuation)
- `"const_modes"` Modes are assumed to be `x`-independent
- `"additive_depth"` Add bottom layer depths instead of setting it

3.8 Array fields

All following fields are real-valued

- `"betas"` Attenuation coefficients for **all** layers (water and bottom)
- `"bottom_layers"` Depths of bottom layers
- `"bottom_rhos"` Density of bottom layers
- `"bottom_c1s"`, `"bottom_c2s"` Sound speed at the top and bottom of each bottom layer
- `"k0"`, `"phi_s"` Wavenumbers and modal functions of the source. Both fields must be present to take effect

3.9 Bathymetry

`"bathymentry"` specifies bottom depth of the domain and is given as Table data. The co-ordinates names are `"x"` and `"y"`

3.10 Hydrolody

`"hydrology"` specifies sound speed in water over `"x"` and `"z"` coordinates as Table data. Missing values can be specified as `-1`

3.11 Modes

`"modes"` is used to explicitly pass wavenumbers and modal functions to be used during computation.

3.11.1 In-file

Modal data can be specified as In-file data in either text or binary format

- `N` — the number of points over `x`,
- `M` — the number of points over `y`,
- `K` — the number of modes,
- `k` — wavenumber value,
- `p` — modal function value

`x`-independent

For `x`-independent modes the following format is used

```
M K
y0 ... yM
k00 ... k0M
... ... ...
kK0 ... kKM
p00 ... p0M
... ... ...
pK0 ... pKM
```

`x`-dependent

For `x`-dependent modes the following format is used

```
N M K
x0 ... xN
y0 ... yN
k000 ... k00M
k100 ... k10M
.... ... ....
kN0K ... kNMK
p000 ... p00M
```

```
.... ... ....
pN0K ... pNMK
```

3.11.2 In-place

Modal data can also be specified as In-place data. For `x`-independent modes `"y"` can be omitted, `"k"` and `"phi"` are two-dimensional

```
{
    "modes": [
        "values",
        {
            "x": [],
            "y": [],
            "k": [ [ [], [], ], [], ],
            "phi": [ [ [], [], ], [], ]
        }
    ]
}
```

3.12 Receivers data

Receivers can be specified as In-file data or In-place data using `"receivers"` key as an array of tuples of three real values: `x`, `y` and `z` coordinates of the receiver. The first value of binary data must be `uint32` — the number of receivers, text data must only contain coordinates.

3.13 Initial values

Initial data is specified using the `"init"` key. Currently `"green"`, `"gauss"`, `"ray_simple"` and `"ray"` values are supported

3.13.1 Green

The standard Greene starter

$$\mathcal{A}_j\left(0,y\right)=\frac{\varphi_j\left(z_s\right)}{2\sqrt{\pi}}\left(1.4467-0.8402k_{j,0}^2\left(y-y_s\right)^2\right)e^{-\frac{k_{j,0}^2\left(y-y_s\right)}{1.5256}}$$

3.13.2 Gauss

The standard Gauss starter

$$\mathcal{A}_j\left(0,y\right)=\frac{\varphi_j\left(z_s\right)}{2\sqrt{\pi}}e^{-k_{j,0}^2\left(y-y_s\right)}$$

3.13.3 Ray starters

Computes initial data using ray theory at `"x0"`. For `"ray_simple"` homogeneous medium is assumed and only source modes are used. For `"ray"` actual modes are used either dependent on `x` or not based on respective configuration parameter

3.14 Tapering

`"tapering"` is used to smooth ray starters edges.

```
{
    "tapering": {
        "type": {
            "value": 0,
            "left": 0,
            "right": 0
        }
    }
}
```

The `"type"` can either be `"percentage"` or `"angled"`, and either `"value"` or both `"left"` and `"right"` must be present meaning percentage or angle range to be smoothed respectively

3.15 Coefficients

`"coefficients` for square root operator approximation can be specified as one of the following

```
{
    "coefficients": [
        "pade"
    ]
}
```

```
{
    "coefficients": [
        "abs",
        {
            "a": 0,
            "b": 0,
            "c": 0
        }
    ]
}
```

`"pade"` being $a = 1, b = 0.75, c = 0.25$

3.16 Default configuration

By default the following configuration is used and new values are either replaced or added

```
{
    "mode_subset": -1,
    "max_mode": -1,
    "n_modes": 0,
    "ppm": 2,
    "ordRich": 3,
    "source_function": 25,
    "z_s": 100,
    "y_s": 0,
    "receivers": [
        "values",
        [
            [0, 0, 30]
        ]
    ],
    "n_layers": 1,
    "bottom_layers": [500],
    "bottom_c1s": [1700],
    "bottom_c2s": [1700],
    "bottom_rhos": [1.5],
    "betas": [0, 0.5],
    "complex_modes": true,
    "const_modes": true,
    "additive_depth": false,
    "past_n": 0,
    "border_width": 10,
    "bathymetry": [
        "values",
        {
            "x": [0, 1],
            "y": [0, 1],
            "values": [
                [200, 200],
                [200, 200]
            ]
        }
    ],
```

```
    "hydrology": [
        "values",
        {
            "x": [0, 1],
            "z": [0, 1],
            "values": [
                [1500, 1500],
                [1500, 1500]
            ]
        }
    ],
    "x0": 0,
    "x1": 15000,
    "nx": 15001,
    "y0": -4000,
    "y1": 4000,
    "ny": 8001,
    "coefficients": [
        "pade"
    ],
    "a0": -0.7854,
    "a1":  0.7854,
    "na": 90,
    "l0": 0,
    "l1": 4000,
    "nl": 4001,
    "init": "green",
    "tapering": {
        "angled": {
            "value": 0.1
        }
    },
    "tolerance": 0.02
}
```