AC2Dr:
Acoustic Codes in 2-D spherical coordinates

Code Development: Keehoon Kim and Björn Sjögreen

September 25, 2023

This is LLNL distribution LLNL-TM-854687.

1. Introduction

AC2Dr is a 2-D numerical solver for the acoustic wave equation using the finite difference method. The acoustic wave equation is split into two first-order differential equations for pressure and particle velocity, and approximated by the sixth-order accurate central difference in space and time. The equations are solved in the axisymmetric spherical coordinates, and hence, are able to simulate 3-D spherical spreading of wavefields by 2-D. The absorbing boundary, which prevents outgoing waves from reflecting at the computational domain boundary, is realized by the super-grid method based on coordinate stretching. *AC2Dr* supports the message passing interface (MPI) on multicore CPU, improving simulation performance dramatically.

2. Governing Equation

AC2Dr solves the linearized Euler equations for an ideal and perfect gas. It can be applied to acoustic propagation in the atmospheres but may not be applicable to other fluids which are not described by the ideal gas law. We adapt the governing equation of *AC2Dr* in order to be applicable to sea water or other fluids for sound propagation. The linearized Euler equation of *AC2Dr* can be written for small perturbation of density (ρ), pressure (p), and particle velocity ($\mathbf{u}$) as follows [Petersson and Sjögreen, 2018].

$$\frac{\partial \rho}{\partial t} + (\hat{\mathbf{u}} \cdot \nabla)\rho + (\mathbf{u} \cdot \nabla)\hat{\rho} + \hat{\rho}\nabla \cdot \mathbf{u} + \rho \nabla \cdot \hat{\mathbf{u}} = f_p, \tag{1}$$

$$\frac{\partial \mathbf{u}}{\partial t} + (\hat{\mathbf{u}} \cdot \nabla)\mathbf{u} + (\mathbf{u} \cdot \nabla)\hat{\mathbf{u}} + \frac{1}{\hat{\rho}}\nabla p - \frac{\rho}{\hat{\rho}^2}\nabla \hat{p} = \mathbf{f}_{\mathbf{u}}, \tag{2}$$

$$\frac{\partial p}{\partial t} + (\hat{\mathbf{u}} \cdot \nabla)p + (\mathbf{u} \cdot \nabla)\hat{p} + \hat{\rho}\hat{c}^2\nabla \cdot \mathbf{u} + \gamma p \nabla \cdot \hat{\mathbf{u}} = f_p, \tag{3}$$

where $\hat{\rho}$, $\hat{p}$, $\hat{\mathbf{u}}$, and $\hat{c}$ are material density, pressure, moving velocity (e.g., wind), and the speed of sound, respectively. In atmospheric acoustics, the terms $\nabla \cdot \hat{\mathbf{u}}$ and $\nabla \hat{p}$ can be small enough to be ignored, and Equations (1) – (3) can be simplified as follows.

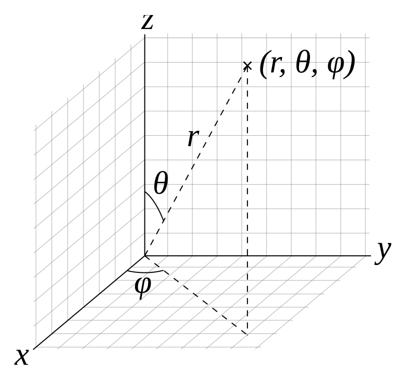

Figure 1: The geometry of spherical coordinate in 3D

$$\frac{\partial \mathbf{u}}{\partial t} + (\hat{\mathbf{u}} \cdot \nabla)\mathbf{u} + (\mathbf{u} \cdot \nabla)\hat{\mathbf{u}} + \frac{1}{\hat{\rho}}\nabla p = \mathbf{f}_{\mathbf{u}}, \tag{4}$$

$$\frac{\partial p}{\partial t} + (\hat{\mathbf{u}} \cdot \nabla)p + \hat{\rho}\hat{c}^2\nabla \cdot \mathbf{u} = f_p. \tag{5}$$

Note that Equations (4) and (5) were also derived by Ostashev et al. [2005] (Equations 17 and 18 in the reference) for sound propagation in arbitrary fluids. Hence, Equations (4) and (5) can be used for any fluid without the limitation of an ideal gas.

If we only consider an axisymmetric case around the z axis in the spherical coordinate (Figure 1), derivatives with respect to φ ($\partial/\partial\varphi$) are zero for all independent and dependent variables. Particle motions ($\mathbf{u}$) and background flow velocity ($\hat{\mathbf{u}}$) can be defined in 2-D as

$$\mathbf{u} = (u_r, u_\theta, u_\varphi = 0) = u_r\vec{\mathbf{r}} + u_\theta\vec{\theta}, \tag{6}$$

$$\hat{\mathbf{u}} = (\hat{u}_r, \hat{u}_\theta, \hat{u}_\varphi = 0) = \hat{u}_r\vec{\mathbf{r}} + \hat{u}_\theta\vec{\theta}, \tag{7}$$

where $\vec{\mathbf{r}}$ and $\vec{\theta}$ are the unit vectors in the r and θ directions. Note that u_r and u_θ denote velocity components in the corresponding directions.

Vector operators in Equation (4) and (5) can be expressed in the axisymmetric spherical coordinates as

$$(\hat{\mathbf{u}}\cdot\nabla)\mathbf{u} = \left(\hat{u}_r\frac{\partial u_r}{\partial r} + \frac{\hat{u}_\theta}{r}\frac{\partial u_r}{\partial\theta} - \frac{\hat{u}_\theta u_\theta}{r}\right)\vec{\mathbf{r}} + \left(\hat{u}_r\frac{\partial u_\theta}{\partial r} + \frac{\hat{u}_\theta}{r}\frac{\partial u_\theta}{\partial\theta} + \frac{\hat{u}_\theta u_r}{r}\right)\vec{\theta}, \tag{8}$$

$$(\mathbf{u}\cdot\nabla)\hat{\mathbf{u}} = \left(u_r\frac{\partial \hat{u}_r}{\partial r} + \frac{u_\theta}{r}\frac{\partial \hat{u}_r}{\partial\theta} - \frac{u_\theta \hat{u}_\theta}{r}\right)\vec{\mathbf{r}} + \left(u_r\frac{\partial \hat{u}_\theta}{\partial r} + \frac{u_\theta}{r}\frac{\partial \hat{u}_\theta}{\partial\theta} + \frac{u_\theta \hat{u}_r}{r}\right)\vec{\theta}, \tag{9}$$

$$\nabla p = \frac{\partial p}{\partial r}\vec{\mathbf{r}} + \frac{1}{r}\frac{\partial p}{\partial\theta}\vec{\theta}, \tag{10}$$

$$\hat{\mathbf{u}}\cdot\nabla p = \hat{u}_r\frac{\partial p}{\partial r} + \frac{\hat{u}_\theta}{r}\frac{\partial p}{\partial\theta}, \tag{11}$$

$$\nabla\cdot\mathbf{u} = \frac{\partial u_r}{\partial r} + \frac{2}{r}u_r + \frac{1}{r}\frac{\partial u_\theta}{\partial\theta} + \frac{\cos\theta}{r\sin\theta}u_\theta. \tag{12}$$

Equation (4) and (5) can be rewritten with respect to $\mathbf{q} = (p, u_r, u_\theta)^T$ as

$$\mathbf{q}_t + A\partial_r\mathbf{q} + B\partial_\theta\mathbf{q} + C\mathbf{q} = \mathbf{f}, \tag{13}$$

where

$$A = \begin{pmatrix} \hat{u}_r & \hat{\rho}\hat{c}^2 & 0 \\ \frac{1}{\hat{\rho}} & \hat{u}_r & 0 \\ 0 & 0 & \hat{u}_r \end{pmatrix},\ B = \frac{1}{r}\begin{pmatrix} \hat{u}_\theta & 0 & \hat{\rho}\hat{c}^2 \\ 0 & \hat{u}_\theta & 0 \\ \frac{1}{\hat{\rho}} & 0 & \hat{u}_\theta \end{pmatrix},\ C = \begin{pmatrix} 0 & \frac{2\hat{\rho}\hat{c}^2}{r} & \frac{\hat{\rho}\hat{c}^2\cos\theta}{r\sin\theta} \\ 0 & \frac{\partial\hat{u}_r}{\partial r} & \frac{1}{r}\frac{\partial\hat{u}_r}{\partial\theta} - \frac{2}{r}\hat{u}_\theta \\ 0 & \frac{\hat{u}_\theta}{r} + \frac{\partial\hat{u}_\theta}{\partial r} & \frac{1}{r}\frac{\partial\hat{u}_\theta}{\partial\theta} + \frac{\hat{u}_r}{r} \end{pmatrix}. \tag{14}$$

3. Getting Started

3.1. Building

AC2Dr is written in *C* language and can be compiled by the GNU Compiler Collection (GCC). A makefile is provided in the main source directory and can be run by Make:

```
make
```

3.2. Running

AC2Dr codes are parallelized based on MPI. The code can be run on multicore CPUs:

```
mpirun -N [numer of processors] ac2dr_mpi [configuration file]
```

4. Configuration

The modeling parameters for *AC2Dr* are set up by a configuration file. The following is an example of parameters in a configuration file.

4.1. path command

```
path input=[path to input files] output=[path to output files]
```

The *path* command includes paths to `[input files]` and `[output files]`. *AC2Dr* will try to find any input files (e.g., background atmosphere profiles) in the input directory and write output files (e.g., wavefield images, waveforms) in the output directory. See the example files.

4.2. grid command

The *grid* command in the configuration file defines the finite-difference grid for simulations. The available options and syntax for *grid* are as follows.

```
grid elevMax=[top elevation in meters]
     angleMax=[rightmost angle in degrees between 0 and 180]
     h=[vertical spacing in meters]
     radius=[radius of sphere in kilometers]
```

The grid of *AC2Dr* is defined in the polar coordinate. The grid always starts from 0 meter in elevation and 0 degree (not a radian) in polar angle. The *elevMax* and *angleMax* determine the upper and rightmost boundary of the grid. The grid spacing is specifid by h, and *radius* defines a radius of the surface at zero elevation.

4.3. time command

This command defines the simulation duration of t in seconds and the Courant-Friedrichs-Lewy condition (*cfl*) for stable temporal integration. Generally, any values less than *cfl* = *1.0* are acceptable. If *cfl* is not provided, it is set to be *1.0* as default value.

```
time t=[total simulation time in second]
     cfl=[CFL number]
```

4.4. mspeed command

Sound propagation in the atmosphere or other fluid is affected by the condition of background materials. *AC2Dr* accepts the sound speed, density, and mean flow of background materials to specify the material properties for sound propagation. The *mspeed* commands specifies the speed of sound for the background materials and accepts a single scalar value for homgeneous material or 1-d/2-d profiles for heterogeneous materials. Only one option among *value*, *profile*, or *2dfile* should be selected exclusively. For 1-d profile or 2d-section of sound speed, *mspeed* needs an input file with the specific format described in the section of Input and Output Files.

```
mspeed value=[speed of sound in m/s]
mspeed profile=[filename for a 1-d sound speed profile]
mspeed 2dfile=[filename for 2-d section of sound speed]
```

4.5. mdensity command

mdensity specifies the density of background materials. The same options used in *mspeed* are available for the density.

```
mdensity value=[material density in kg/m^3]
mdensity profile=[1-d density profile]
mdensity 2dfile=[2-d density section]
```

4.6. wind command

wind defines the horizontal mean flow of background materials. The governing equations of *AC2Dr* only include horizontal flows parallel to the surface.

```
wind value=[mean horizonta flow]
     profile=[1-d wind profile (horizontal)]
     2dfile=[2-d wind section (horizontal)]
```

4.7. asource command

The *asource* command defines the source of acoustic waves. The elevation and angle of source position is specified by *elev* and *angle*. The peak amplitude and corner frequency of the source are set by *p0* and *freq*. The source time function is defined as *Gaussian function* by *type*. Currently it supports only Gaussian source time function but will accept other functions in the future.

```
asource elev=[source elevation in meters]
        angle=[source angle in degrees]
        p0=[source amplitude in Pa]
        freq=[source frequency in Hz]
        type=Gaussian
```

4.8. rec command

The *rec* command records simulated waveform outputs at specified positions. The synthetic receiver name and its position are determined by the *name*, *elev*, and *angle* options. The *mode* determines the computation variables to be recorded. p, v, and w represent pressure, vertical motion, and horizontal motion. Either one or all of them can be specified for one receiver. The *format* option is for the output file format of the waveforms. The details of binary format can be found in the section of Input and Output Files.

```
rec name=[reciver name]
    elev=[elevation]
    angle=[angle]
    mode=[p|v|w]
    format=binary
```

4.9. image command

The *image* command saves the entire 2-D section for a computed variable. The option *timeInterval* and *mode* set a time interval and variable for the section. $mode = p$, v, and w save pressure, vertical motion, and horizontal motion, respectively. *format* sets the output file format described in the Input and Output Files section.

```
image timeInterval=[time interval]
      mode=[p|v|w]
      file=[filename]
      format=binary
```

5. Input and Output Files

5.1. Sound speed, density, and wind data

5.1.1. 1-D vertical profile

AC2Dr accepts a 1-D vertical profile of sound speed, density, and horizontal winds in binary format. The profile needs to include two columns including elevation (m) and either sound speed (m/s), density (kg/m^3), or wind (m/s). Each value needs to be stored as a double precision float type (8 bytes). For example, if the input file includes N rows of elevation from H_1 to H_N and data from D_1 to D_N. They must be stored in the following order:

[H_1 (8 bytes)][D_1 (8 bytes)][H_2][D_2]...[H_N][D_N].

The interval of elevations in the input file does not need to be the same as the grid spacing (h) in the finite difference mesh. *AC2Dr* internally performs a linear interpolation and find values that fit the finite difference grid.

5.1.1. 2-D vertical section

The 2-D veritical section data for sound speed, density, and horizontal winds need to be gridded. The data points are supposed to be defined at certain elevations (H_i) and angle (θ_i). The interval of elevations (ΔH) and angles ($\Delta\theta$) must be uniform in the input data. The format of the binary file is as follows.

Order	Type	Bytes	Item
1	int	4	m (the number of data points for polar angle)
2	int	4	n (the number of data points for elevation)
3	double	8	$\Delta\theta$ (the interval of the angles)
4	double	8	ΔH (the interval of the elevations)
4+1	double	8	data at $(H_1\ ,\ \theta_1)$
4+2	double	8	data at $(H_2\ ,\ \theta_1)$
4+3	double	8	data at $(H_3\ ,\ \theta_1)$
.	.	.	.
.	.	.	.
4+n	double	8	data at (H_n, θ_1)
4+(n+1)	double	8	data at (H_1, θ_2)
4+(n+2)	double	8	data at (H_2, θ_2)
.	.	.	.
.	.	.	.
4+(n×m)	double	8	data at (H_n, θ_m)

5.2. *Waveform output*

The waveform output is a two-column binary file. The order and type of data are as follows.

Order	Type	Bytes	Item
1	double	8	Δt (time interval)
2+0	double	8	data (pressure or particle velocity) at $0s$
2+1	double	8	data at $1 \times \Delta ts$
2+2	double	8	data at $2 \times \Delta ts$
.	.	.	.
.	.	.	.
.	.	.	.
2+N	double	8	data at $N \times \Delta ts$

5.3. *Image output*

The image file is the output of the *image* command in the configuration file. This is a binary file having the same structure as for the 2-D vertical section data.

Order	Type	Bytes	Item
1	int	4	m (the number of data points for polar angle)
2	int	4	n (the number of data points for elevation)
3	double	8	$\Delta\theta$ (the interval of the angles)
4	double	8	ΔH (the interval of the elevations)
4+1	double	8	data at $(H_1\ ,\ \theta_1)$
4+2	double	8	data at $(H_2\ ,\ \theta_1)$
4+3	double	8	data at $(H_3\ ,\ \theta_1)$
.	.	.	.
.	.	.	.
4+n	double	8	data at (H_n, θ_1)
4+(n+1)	double	8	data at (H_1, θ_2)
4+(n+2)	double	8	data at (H_2, θ_2)
.	.	.	.
.	.	.	.
4+(n×m)	double	8	data at (H_n, θ_m)

References

Calabrese, G. and Neilsen, D. (2004). Spherical excision for moving black holes and summation by parts for axisymmetric systems. *Phys. Rev. D*, 69:044020.

Gundlach, C., Martín-García, J. M., and Garfinkle, D. (2013). Summation by parts methods for spherical harmonic decompositions of the wave equation in any dimensions. *Classical and Quantum Gravity*, 30(14):145003.

Ostashev, V., Wilson, D., Liu, L., Aldridge, D., Symons, N., and Marlin, D. (2005). Equations for finite-difference, time-domain simulation of sound propagation in moving inhomogeneous media and numerical implementation. *The Journal of the Acoustical Society of America*, 117:503.

Petersson, N. A. and Sjögreen, B. (2018). High order accurate finite difference modeling of seismo-acoustic wave propagation in a moving atmosphere and a heterogeneous earth model coupled across a realistic topography. *Journal of Scientific Computing*, 74(1):290–323.