Elastic Wave Equation with Exact Riemann Solver

Project Work: Modern Wave Propagation - Discontinuous Galerkin & Julia Lab
Group Cloud: Ashish (03709039); Keerthi (03710201), Irene

I. Introduction : Elastic Wave Equation

$$\frac{\partial u_p}{\partial t} + A_{pq}\frac{\partial u_q}{\partial x} + B_{pq}\frac{\partial u_q}{\partial y} = S_p$$

The propagation of waves in an elastic medium is based on the theory of linear elasticity. Combining the definition of strain caused by deformations (Hooke's law) and the equations of the dynamic relationship between acceleration and stress, the elastic wave equations can be derived (shown in the equation section on right). Considering the 2-D elastic wave equation for an isotropic medium in velocity-stress formulation and admitting external sources (e.g. moments or body forces) leads to a linear hyperbolic system of the form (shown in the equation section). The compact form of the set of the equations is given above. Here we have a state vector with 5 degrees of freedom as $u = (\sigma_{xx}, \sigma_{yy}, \sigma_{xy}, v, w)^T$. The Jacobian matrix $(A_n)_{pq}$ in normal direction $n = (n_x, n_y)^T$ to an element interface is obtained by the linear combination of the Jacobian matrices. The propagation velocity of the elastic waves are determined by the eigenvalues of the Jacobian matrices A_{pq} and B_{pq} and result in $s_1 = c_p$, $s_2 = c_s$, $s_3 = 0$, $s_4 = -c_s$, $s_5 = -c_p$, where c_p and c_s are the speeds of P-Wave and S-Wave respectively. $R^{An}_{p1}, \ldots, R^{An}_{p5}$ denotes the right eigenvectors of the normal Jacobian matrix in direction n and are given by the columns of the matrix of right eigenvectors (shown in the equation section).

$$\frac{\partial}{\partial t}\sigma_{xx} - (\lambda+2\mu)\frac{\partial}{\partial x}v - \lambda\frac{\partial}{\partial y}w = S_1(x,y,t)$$
$$\frac{\partial}{\partial t}\sigma_{yy} - \lambda\frac{\partial}{\partial x}v - (\lambda+2\mu)\frac{\partial}{\partial y}w = S_2(x,y,t)$$
$$\frac{\partial}{\partial t}\sigma_{xy} - \mu\left(\frac{\partial}{\partial x}w + \frac{\partial}{\partial y}v\right) = S_3(x,y,t)$$
$$\rho\frac{\partial}{\partial t}v - \frac{\partial}{\partial x}\sigma_{xx} - \frac{\partial}{\partial y}\sigma_{xy} = \rho S_4(x,y,t)$$
$$\rho\frac{\partial}{\partial t}w - \frac{\partial}{\partial x}\sigma_{xy} - \frac{\partial}{\partial y}\sigma_{yy} = \rho S_5(x,y,t)$$

$$(A_n)_{pq} = n_x A_{pq} + n_y B_{pq}$$
$$= \begin{pmatrix} 0 & 0 & 0 & -n_x(\lambda+2\mu) & -n_y\lambda \\ 0 & 0 & 0 & -n_x\lambda & -n_y(\lambda+2\mu) \\ 0 & 0 & 0 & -n_y\mu & -n_x\mu \\ -\frac{n_x}{\rho} & 0 & -\frac{n_y}{\rho} & 0 & 0 \\ 0 & -\frac{n_y}{\rho} & -\frac{n_x}{\rho} & 0 & 0 \end{pmatrix}$$

$$c_p = \sqrt{\frac{\lambda+2\mu}{\rho}}, \qquad c_s = \sqrt{\frac{\mu}{\rho}}.$$

$$R^{A_n}_{pq} = \begin{pmatrix} \lambda+2\mu n_x^2 & -2\mu n_x n_y & n_y^2 & -2\mu n_x n_y & \lambda+2\mu n_x^2 \\ \lambda+2\mu n_y^2 & 2\mu n_x n_y & n_x^2 & 2\mu n_x n_y & \lambda+2\mu n_y^2 \\ 2\mu n_x n_y & \mu(n_x^2-n_y^2) & -n_x n_y & \mu(n_x^2-n_y^2) & 2\mu n_x n_y \\ n_x c_p & -n_y c_s & 0 & n_y c_s & -n_x c_p \\ n_y c_p & n_x c_s & 0 & -n_x c_s & -n_y c_p \end{pmatrix}$$

Code: Similar to other worksheets, we have witten all the information about the initial conditions, boundary conditions and flux evaluation in a separate equation file called *elastic.jl*. We have tested our code for 4 scenarios which can be found in the same file. Results of which were discussed later in the report.

II. Source Term and Fractional-Step method

Till now we have considered homogeneous conservation laws of the form $q_t + f(q)_x = 0$ in all four worksheets. But there are many situations in which source terms also appear in the equations for example the Elastic Wave equation stated in a section I. So we wish to solve the system of the form $q_t + f(q)_x = \psi(q)$. To solve such a system we have used a fractional-step or operator-splitting method. Given problem is splitted into two separate problems, problem A: $q_t + f(q)_x = 0$ which is hyperbolic PDE, and can be solved using high-resolution methods (without changing our existing code). And Problem B: $q_t = \psi(q)$, which is an ODE, and can be solved using simple time stepping methods (we have used Forward Euler in our code).

Code: To solve problem B, forward Euler code is written in file *TerraDG.jl* after a function call to *evaluate_rhs()*. So in a code, all 5 DOFs (for all cells in a grid) will get modified by solving problem A first (in evaluate_rhs() function using DG with Exact Riemann solver) and then solving problem B independently in *main()* function. Calculation of the source term has been written in the function *evaluate_source()* in *elastic.jl* file.

III. Exact Riemann Solver

We replaced the Russanov solver with the exact Riemann solver, which can be computed since we are dealing with a linear equation. We have to find the flux value at boundaries in the x and y-direction. This means that at the boundaries our problem reduces to a one dimensional one. We diagonalize the matrices A and B from the equation given in the introduction and use their eigenvectors to find the solution in the x and y-direction respectively. We write the flux at each side of the boundary as a linear combination of these eigenvectors. We did this with a Mathematica notebook which is included in the gitlab repository (also as a pdf). Note that outputs 15-16 & 22-23 show the coefficients (ɣ, β, μ and η) of these linear combinations as a function of the flux at both sides of the boundaries.

Then our solutions are:

x-direction: $$F(\mathbf{u}) = \sum_{i=1}^{j} \gamma_i r_i^A + \sum_{i=j+1}^{5} \beta_i r_i^A$$ and y-direction: $$F(\mathbf{u}) = \sum_{i=1}^{j} \mu_i r_i^B + \sum_{i=j+1}^{5} \eta_i r_i^B$$

where r^A and r^B denote the eigenvectors of A and B respectively, and j is the maximum index such that $-s_j t < 0$. In our case j = 2, as $-s_2 t = -c_s t < 0$ satisfies this inequality but $-s_3 t = 0$ doesn't. Outputs 17 & 24 in the Mathematica notebook show the solutions as a function of the flux at both sides of the boundaries. Note that in the code we implement this as a function of the flux in the current cell and the neighbouring cell. Therefore, we need to use the sign of the outward normal to the boundary when necessary.

Code: The user can choose for the configuration file which solver they want to use by calling either "solver: 'russanov'" or "solver:'exact'". The function evaluate_face_integral() then takes this information as an input and runs the functions rusanov() or exactRiemann().

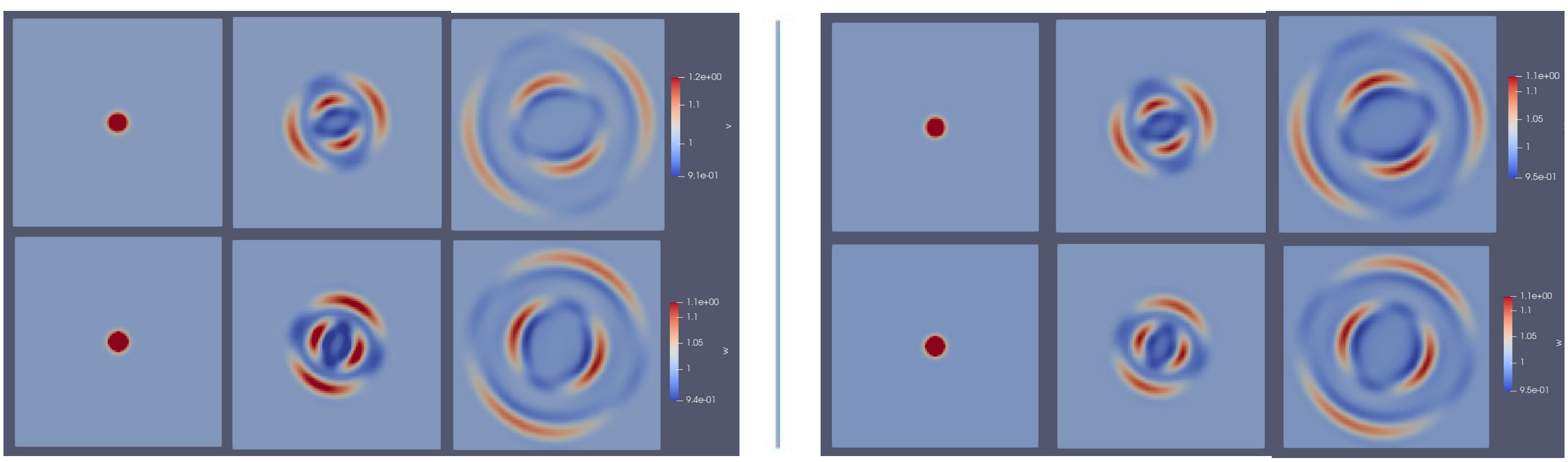
IV. Scenarios and Results

1. Gaussian Point initialisation with approximate Riemann solver with Rusanov flux:

A simple scenario was implemented to test our exact Riemann solver implementation and compare results to approximate Riemann solver with Rusanov flux. The gaussian point scenario has the following initial conditions:

$$\sigma^{11}(x,y,0) = \sigma^{22}(x,y,0) = \sigma^{12}(x,y,0) = 0.0$$
$$u(x,y,0) = v(x,y,0) = 10^{-3}\mathcal{G}\left(\begin{pmatrix} x-\frac{1}{2} \\ y-\frac{1}{2} \end{pmatrix}, 1\right)$$

The visualization of velocities in x and y-direction obtained from Rusanov flux can be seen in figure below (on the left hand side).



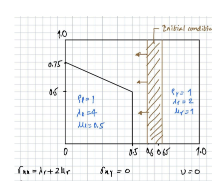
The velocities at timestep=0 are zero everywhere in the domain except at the center. As the timesteps increase, we observed that P-waves propagate faster than S-waves. This is exactly what was expected as Lame parameters were set accordingly. The speeds of the waves depend on the Lame parameters and material parameters.

2. Gaussian Point initialisation with an Exact Riemann solver:

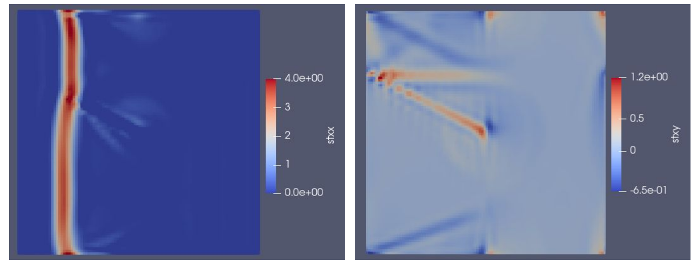
Initial conditions for this problem are chosen in a similar way as above problem, visualization of the results can be seen in the figure above (on the right hand side). The results of both the solvers look quite similar with negligible difference. In conclusion, the exact riemann solver is working quite well except that it took about 20% more time to run than the riemann solver with rusanov flux.

3. Stiff Inclusion problem:

In this problem, the domain is partitioned and both partitions are given different material parameter values such that one is stiffer than the other. Then the initial conditions are chosen as follows: The initial conditions between 0.6 and 0.65 move from right to left and we observe how the wave propagates once it hits the interface of the other material. The results obtained are as follows,



The left image is a visualization of the normal stress component in x-direction. The P-wave moves from right to left and as soon as it hits the interface, the transmitted P-wave moves more faster (speed of P-wave changes from 2 to 2.2) and there is a partial reflection. In the elastic case there is also both a transmitted and reflected S-wave along the ramp portion of the interface. These are faintly visible here, since S-waves at an oblique angle have a nonzero component σ_{xx}. The image on the right is the shear stress component propagating in the xy-direction. S-waves are clearly visible in this image.



4. Elastic wave equation with source term:
For this we have considered the solution of our problem as $u_p(x,y,t)$ and then we have calculated the spatial and temporal derivatives and imposed the initial conditions (as shown in the right side) in the system of elastic equations stated in the section I, to get the source term S_p. We then calculated the source term inside the function *evaluate_source()* and the problem then solved using the fractional-step method explained in section II. We have solved this problem using an approximate riemann solver with rusanov flux and exact riemann solver. From the error tables (as shown on the right hand side) we can see, there is a reduction in the L∞ norms. So we have obtained **more accurate results in exact riemann solver at the expense of more computational time.**
Following figure shows the visualisation of the results. We can observe that all 5 DOFs show exact results as per assumed in the analytical solution $u_p(x,y,t)$. All the waves in the five variables oscillate for 3, 2, 1, 10 and 20 periods, respectively. Especially due to the rather high temporal frequencies of the last two variables, we expect the accuracy of the time discretization to be of great importance.

$$u_p(x,y,t) = u_p^0 \cdot \sin\left(\frac{2\pi}{\lambda_p^x}x\right) \cdot \sin\left(\frac{2\pi}{\lambda_p^y}y\right) \cdot \sin\left(\frac{2\pi}{T_p}t\right)$$

$$u_p^0 = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \end{pmatrix}^T,$$

$$\lambda_p^x = \lambda_p^y = \begin{pmatrix} 33\frac{1}{3} & 50 & 100 & 50 & 33\frac{1}{3} \end{pmatrix}^T$$

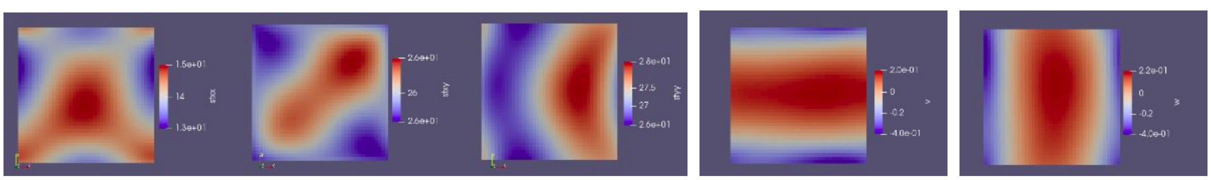
$$T_p = \begin{pmatrix} 33\frac{1}{3} & 50 & 100 & 10 & 5 \end{pmatrix}^T.$$

Errors for each variable.

Var	L1	L2	L∞
stxx	1.407685e+01	1.408351e+01	1.543152e+01
styy	2.730351e+01	2.971519e+01	4.820603e+01
stxy	2.603334e+01	2.603601e+01	2.671552e+01
v	1.435690e-01	1.675804e-01	4.165937e-01
w	1.443762e-01	1.690188e-01	4.117900e-01

Errors for each variable.

Var	L1	L2	L∞
stxx	1.407685e+01	1.407895e+01	1.452254e+01
styy	2.730351e+01	2.730871e+01	2.823116e+01
stxy	2.603334e+01	2.603430e+01	2.646944e+01
v	1.428693e-01	1.652503e-01	3.980394e-01
w	1.435389e-01	1.663518e-01	3.983682e-01



V. References

[1] Finite Volume Methods for Hyperbolic Problems, LeVeque
[2] An arbitrary high-order discontinuous Galerkin method for elastic waves on unstructured meshes – I. The two-dimensional isotropic case with external source terms(https://onlinelibrary.wiley.com/doi/10.1111/j.1365-246X.2006.03051.x)
[3] Finite Volume and Linear Elastic Wave, Margret Westerkamp, Computational Science and Engineering, TU München,Seminar Course - Fundamentals of Wave Simulation - Solving Hyperbolic Systems of PDEs

VI. Contribution

1. Ashish Darekar : Literature review and Implementation of Elastic wave equations (section I); literature review and Implementation of source term and fractional-step method (section II); literature review of the exact riemann solver for 1D and attempt of mathematical part for 2D; Implementation of 4th scenario elastic wave equation with source term.

2. Keerthi Gaddameedi : Literature review, Implementation of Gaussian point scenario and Stiff Inclusion problem.

3. Irene: Literature review, mathematical derivation and implementation of exact Riemann solver for 2D

Note: Project code and related documents are available on GIT https://gitlab.lrz.de/ge25duq/terradg.jl