Adjoints for Scientists and Engineers
Spring 2018
Prof. Hicken • JEC 2036 • email: *hickej2@rpi.edu*

Assignment #4 **Due Date: May 1, 2018**

We use this last assignment to get experience implementing and using the adjoint in the context of unsteady problems.

1 Initial Boundary Value Problem and Functional

1.1 The 1D Euler Equations

In this assignment we consider the one-dimensional (1D) Euler equations, which can be obtained by dropping the area dependence from the quasi-1D Euler equations and adding a time derivative. Specifically, the PDE is given by

$$R(q) \equiv \frac{\partial q}{\partial t} + \frac{\partial F(q)}{\partial x} - G(x,t) = 0, \qquad \forall\, x \in [0,1], t \in [0,2], \tag{1}$$

where the flux and source are now

$$F(q) = \begin{pmatrix}\rho u, & (\rho u^2 + p), & u(e+p)\end{pmatrix}^T \qquad \text{and} \qquad G(x,t) = \left(0, \quad s(t)\exp\left(-\tfrac{(x-x_s)^2}{\sigma_s^2}\right), \quad 0\right)^T.$$

The momentum source, $G(x,t)$, has a squared-exponential shape that is centered at $x_s = 0.5$ with a "width" $\sigma_s = 0.05$. The magnitude of the source varies in time and is controlled by $s(t)$; the discretized version of $s(t)$ will provide the design variables for this problem, but its baseline value is $s(t) = 0$.

The initial condition for the flow is a momentum perturbation on a uniform supersonic flow from left to right. In non-dimensional terms we have

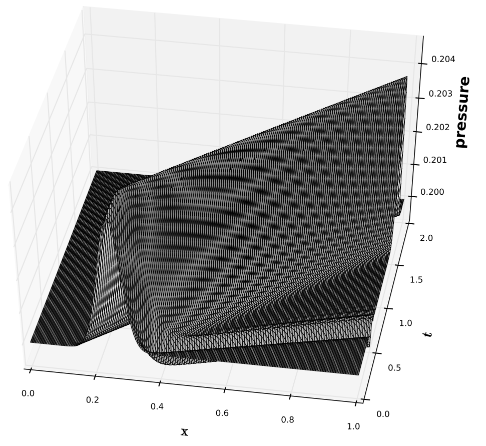
$$q(x,0) = \begin{pmatrix}1, & 1, & 1\end{pmatrix}^T + \begin{pmatrix}0, \widetilde{\rho u}(x), 0\end{pmatrix}^T,$$

where

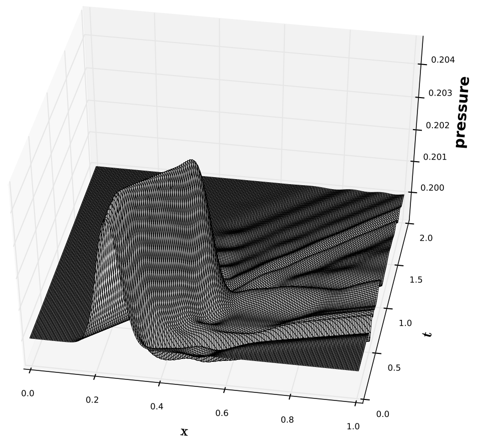
$$\widetilde{\rho u}(x) = -\frac{1}{100}\exp\left(-\frac{(x-x_{\text{IC}})^2}{\sigma_{\text{IC}}^2}\right).$$

The center of the momentum perturbation is $x_{\text{IC}} = 0.25$ and $\sigma_{\text{IC}} = 0.05$. This initial condition creates a disturbance composed of both acoustic waves. As the solution evolves, the faster acoustic wave overtakes the slower one, resulting in two solitons moving from left to right. This is illustrated in Figure 1(a).

We apply characteristic boundary conditions at the inflow, $x = 0$, such that the $q(0,t) = (1,1,1)^T$. We will assume the flow is supersonic for all sources; thus, no boundary condition is necessary at the outflow $x = 1$.



(a) no momentum source



(b) optimized momentum source

Figure 1: Pressure evolution, visualized in space-time, for the initial (left) and optimized (right) momentum source in the acoustic control problem.

1.2 Functional

The functional for this assignment is a "pressure sensor", which consists of the space-time integral of a weighted (squared) pressure difference:

$$J(q) = \int_0^2 \int_0^1 \kappa(x) \frac{1}{2} (p(x,t) - p_{\text{targ}})^2 \, dx \, dt$$

where $\kappa(x)$ is a weighting kernel defined by

$$\kappa(x) = \exp\left(-\frac{(x - x_\kappa)^2}{\sigma_\kappa^2} \right),$$

with $x_\kappa = 0.85$ and $\sigma_\kappa = 0.05$. The target pressure is given by the quiescent background pressure, $p_{\text{targ}} = 0.2$. Qualitatively, the objective function is minimized when the difference between the pressure and target pressure is small in the vicinity of x_κ for all $t \in [0, 2]$. For completeness, the pressure evolution for an optimal source $s(t)$ is shown in Figure 1(b).

2 Discretization

The spatial discretization is the same as that used in Assignments 2 and 3, i.e. a nodal discontinuous Galerkin method [1], with the area set to unity. Let $R_h : \mathbb{R}^s \times \mathbb{R} \to \mathbb{R}^s$ denote the discretization of the spatial terms, which depends on the state q_h and source magnitude s.

Two time-marching schemes are implemented and available for you to use: the (explicit) 4th-order Runge-Kutta scheme and the (implicit) 2nd-order Crank-Nicholson scheme. These are briefly described below.

2.1 Runge-Kutta

Let $q_h^{(n)}$ and $s^{(n)}$ denote the state solution and source at time step n (which may be the initial condition). Then the Runge-Kutta scheme that determines $q_h^{(n+1)}$ is defined by the following four stages:

- Euler predictor: $\hat{q}_h = q_h^{(n)} + \frac{\Delta t}{2}\mathsf{M}^{-1}R_h(q_h^{(n)}, s^{(n)})$
- Euler Corrector: $\tilde{q}_h = q_h^{(n)} + \frac{\Delta t}{2}\mathsf{M}^{-1}R_h(\hat{q}_h, s^{(n+1/2)})$
- Leapfrog Predictor: $\bar{q}_h = q_h^{(n)} + \Delta t\mathsf{M}^{-1}R_h(\tilde{q}_h, s^{(n+1/2)})$
- Milne Corrector:

$$q_h^{(n+1)} = q_h + \frac{\Delta t}{6}\mathsf{M}^{-1}\Big[R_h(q_h^{(n)}, s^{(n)}) + \\ 2\Big(R_h(\hat{q}_h, s^{(n+1/2)}) + R_h(\tilde{q}_h, s^{(n+1/2)})\Big) + R_h(\bar{q}_h, s^{(n+1)})\Big]$$

In the above stages, M^{-1} denotes the inverse matrix matrix, which is diagonal for this discretization. Furthermore, the source at the half time step is given by $s^{(n+1/2)} = \frac{1}{2}(s^{(n)} + s^{(n+1)})$; because of this linear approximation of the source, our implementation is only 2nd-order accurate for non-zero sources.

The provided code returns $q_h^{(n+1)}$ and not the intermediate states, $\hat{q}_h$, $\tilde{q}_h$, and $\bar{q}_h$; however, you will want to account for these intermediate states if you base your adjoint scheme on the above Runge-Kutta method.

2.2 Crank-Nicholson

In addition to the explicit Runge-Kutta scheme described above, a second-order implicit Crank-Nicholson scheme has also been implemented. For this implicit scheme, the state $q_h^{(n+1)}$ satisfies the following equation:

$$\hat{R}^{(n+1)}(q_h^{(n+1)}; q_h^{(n)}, s^{(n)}, s^{(n+1)}) = \\ \frac{1}{\Delta t}\mathsf{M}(q_h^{(n+1)} - q_h^{(n)}) + \frac{1}{2}\Big[R_h(q_h^{(n+1)}, s^{(n+1)}) + R_h(q_h^{(n)}, s^{(n)})\Big] = 0.$$

This nonlinear equation is solved using Newton's method.

3 Provided Code

Please download the supplementary zip file for this assignment, which has code that has been updated from Assignment 3. Here is a list of some of the functions you should be aware of.

- `unsteadySolveRK4!` advances the solution from time $t = 0$ to $t = T$ using the 4th-order Runge-Kutta method described above. The solution at each time step (but not each stage) can be saved to a file in a binary format if desired.

- `stepRK4!` performs one step of the 4th-order Runge-Kutta method; **warning**, there are two versions of this, one for problems with varying nozzle areas and one for a source term. You want the one that includes a source term.
- `unsteadySolveCN!` advances the solution from time $t = 0$ to $t = T$ using the second-order Crank-Nicholson method described above. As with the Runge-Kutta method, you can save the solution to file.
- `stepCN!` performs one step of Crank-Nicholson; this is where Newton's method is used to solve the nonlinear system.
- `calcTimeIntegratedObjective!` computes the functional J defined above.

The file `assign4_example.jl` is provided to show you how to run the unsteady code.

4 Questions

1. Derive the adjoint IBVP for the Euler equations and the functional $J(q)$.
2. Choose either the 4th-order Runge-Kutta scheme or the Crank-Nicholson scheme and derive the corresponding discrete adjoint. What I am looking for here are the details that will help you implement the adjoint time-marching scheme.
3. Solve for the adjoint variables corresponding when $s^{(n)} = 0, \quad \forall n$.
   (a) Provide verification that your adjoint is implemented correctly.
   (b) Plot the three adjoints, ψ_ρ, $\psi_{\rho u}$, and ψ_e, versus x at following times: $t = 0$, $t = 0.25$, and $t = 0.5$.
4. Use your adjoint solution to evaluate the gradient $DJ/Ds^{(n)}$ for all time steps n. Verify your gradient using a (directional) finite-difference.

5 Report and Code Expectations

Your report should be typeset in LaTeX. As before, the report should be fewer than 10 pages if you use a 12 point font. This is a "soft" page limit, but please do not be overly verbose.

The following should be included in the report.

- An abstract that summarizes your results.
- Provide a separate section or subsection in which you answer each question in the above list of questions.

You do not need to include an introduction, since your audience (me!) knows the background. You can include the results of additional studies that you performed out of interest.

The report will include several figures as requested in the questions. Please use an adequate font size on the figures! It should be legible if printed.

Please provide the modified code for your project as a separate zip file. You do not need to include in the report itself. All code should have adequate commenting.

Collaboration

You are permitted and encouraged to discuss the assignment with each other, provided each of you writes your own code and report. A good policy to follow in order to avoid academic misconduct is to not take notes or exchange files with one another; i.e. exchange information verbally and you should be fine.

References

[1] Hesthaven, J. S. and Warburton, T., *Nodal discontinuous Galerkin methods: algorithms, analysis, and applications*, Springer-Verlag, New York, 2008.