Computational Methods for the Euler Equations

Before discussing the Euler Equations and computational methods for them, let’s look at what we’ve learned so far:

<table>
<thead>
<tr>
<th>Method</th>
<th>Assumptions/Flow type</th>
</tr>
</thead>
<tbody>
<tr>
<td>2-D panel</td>
<td>2-D, Incompressible, Irrotational Inviscid</td>
</tr>
<tr>
<td>Vortex lattice</td>
<td>3-D, Incompressible, Irrotational Inviscid, Small disturbance</td>
</tr>
<tr>
<td>Potential method</td>
<td>3-D, Subsonic compressible, Irrotational, Inviscid, Prandtl-Glauert Small disturbance</td>
</tr>
<tr>
<td>Prandtl-Glauert</td>
<td>3-D, Subsonic compressible, Irrotational, Inviscid, Small disturbance</td>
</tr>
<tr>
<td>Euler CFD</td>
<td>3-D, Compressible (no $M_{∞}$ limit), Rotational, Shocks, Inviscid</td>
</tr>
</tbody>
</table>

The only major effect missing after this week will be viscous-related effects.

2-D Euler Equations in Integral Form

Consider an arbitrary area (i.e. a fixed control volume) through which flows a compressible inviscid flow:

\[ \tilde{n} \equiv \text{outward pointing normal (unit length)} \]
\[ dS \equiv \text{elemental (differential) surface length} \]

\[ \tilde{n}dS = dy\hat{i} - dx\hat{j} \]

Note: Path around surface is taken so that interior of control volume is on left.
Conservation of Mass
\[
\left( \text{rate of change of mass in } C \right) + \left( \text{rate of mass flow out of } C \right) = 0
\]
Mass in \( C \) = \( \int_C p \, dA \)
\[\Rightarrow \text{rate of change} = \frac{d}{dt} \int_C p \, dA \]
of mass in \( C \)

Now, the rate of mass flowing out of \( C \):
Mass flow out of \( C \) = \( \int_C \rho \ddot{u} \cdot \mathbf{n} \, dS \) \( \ddot{u} = \) velocity vector
\[\Rightarrow \frac{d}{dt} \int_C p \, dA + \int_C \rho \ddot{u} \cdot \mathbf{n} \, dS = 0\]

Conservation of x-momentum
Recall that: total rate of change momentum = \( \sum \) forces
For x-momentum this gives:
\[
\left( \text{rate of change of } x \text{-momentum in } C \right) + \left( \text{rate of } x \text{-momflow out of } C \right) = \sum \text{ Forces in } x\text{-direction}
\]
\[
\frac{d}{dt} \int_C \rho u dA + \int_C \rho \ddot{u} \cdot \mathbf{n} dS = \sum \text{ Forces in } x\text{-direction}
\]
Now, looking closer at x-forces, for an inviscid compressible flow we only have pressure (ignoring gravity).
Recall pressure acts normal to the surface
\[\Rightarrow \sum \text{ Force in } x = -\int_C p \ddot{n} \cdot \mathbf{i} \, dS\]
Gives x-direction
Conservation of y-momentum

This follows exactly the same as the x-momentum:

\[
\frac{d}{dt} \iint_{\mathcal{C}} \rho uv dA + \oint_{\partial \mathcal{C}} \rho u \mathbf{\bar{n}} \cdot \mathbf{\bar{v}} dS = - \oint_{\partial \mathcal{C}} p \mathbf{\bar{n}} \cdot \mathbf{\bar{v}} dS
\]

Conservation of Energy

Recalling your thermodynamics:

\[
\left( \text{total rate of change of energy in } C \right) = \left( \text{work done on fluid in } C \right) + \left( \text{heat added to } C \right)
\]

For the Euler equations, we ignore the possibility of heat addition.

\[
\left( \text{total rate of change of energy in } C \right) = \left( \text{rate of change of energy in } C \right) + \left( \text{flowout of } C \right)
\]

The total energy of the fluid is:

\[
\rho E = \rho e + \frac{1}{2} \rho (u^2 + v^2)
\]

Note: \( e = c_v T \) where \( c_v \) = specific heat at constant volume

Static temperature

So,

\[
\left( \text{total rate of change of energy in } C \right) = \frac{d}{dt} \iint_{\mathcal{C}} \rho E dA + \oint_{\partial \mathcal{C}} \rho E \mathbf{\bar{v}} \cdot \mathbf{\bar{n}} dS
\]

The work done on the fluid is through pressure forces and is equal to the pressure forces multiplied by (i.e. acting in) the velocity direction:

\[
\text{(work)} = \oint_{\partial \mathcal{C}} (p \mathbf{\bar{n}}) \cdot \mathbf{\bar{v}} dS
\]

Pressure force
Summary of 2-D Euler Equations

\[
\frac{d}{dt} \int dA + \oint \rho \mathbf{u} \cdot \mathbf{n} \, dS = 0
\]

\[
\frac{d}{dt} \int dA + \oint \rho \mathbf{u} \cdot \mathbf{n} \, dS = -\oint \rho \mathbf{n} \cdot \mathbf{u} \, dS
\]

\[
\frac{d}{dt} \int dA + \oint \rho \mathbf{v} \cdot \mathbf{n} \, dS = -\oint \rho \mathbf{n} \cdot \mathbf{v} \, dS
\]

\[
\frac{d}{dt} \int dA + \oint \rho E \mathbf{u} \cdot \mathbf{n} \, dS = -\oint \rho \mathbf{n} \cdot \mathbf{E} \, dS
\]

These are often written very compactly as:

\[
\frac{d}{dt} \int dA + \oint (\mathbf{F} \mathbf{u} + \mathbf{G} \mathbf{v}) \cdot \mathbf{n} \, dS = 0
\]

Conservative state vector

\[ U = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho E \end{bmatrix} \]

Flux vector for x-direction

\[ F = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ \rho u v \\ \rho u H \end{bmatrix} \]

Flux vector for y-direction

\[ G = \begin{bmatrix} \rho v \\ \rho v u \\ \rho v^2 + p \\ \rho v H \end{bmatrix} \]

Total enthalpy

\[ H \equiv \text{total enthalpy} \equiv E + \frac{p}{\rho} \]

Ideal gas: \[ p = \rho RT = (\gamma - 1) \left[ \rho E - \frac{1}{2} \rho (u^2 + v^2) \right] \]

A Finite Volume Scheme for the 2-D Euler Eqns.

Here’s the basic idea:

(1) Divide up (i.e. discretize) the domain into simple geometric shapes (triangles and quads)
Looking at this small region:

Cell 0 is surrounded by cells 1, 2, & 3.

i.e. cell 0 has 3 neighbors: cell 1, 2, & 3.

Nearest neighbors
(2) Decide how to place the unknowns in the grid.
(a) Cell-centered: cell-average values of the conservative state vector are stored for each cell.
(b) Node-based: point values of the conservative state vector are stored at each node.
The debate still rages about which of these options is best. We will look at cell-centered schemes because these are easiest (although not necessarily the best). Also, they are very widely used in the aerospace industry.

(3) Approximate the 2-D integral Euler equation on the grid to determine the chosen unknowns.
Mach Number - Original Mesh
Refined Mesh - 7506 Nodes
Let’s look in detail at step (3):

Cell-average unknowns:

\[
U_0 = \begin{bmatrix}
\rho_0 \\
\rho u_0 \\
\rho v_0 \\
\rho E_0
\end{bmatrix}, \quad U_1 = \begin{bmatrix}
\rho_1 \\
\rho u_1 \\
\rho v_1 \\
\rho E_1
\end{bmatrix}, \quad U_2 = \ldots, \quad U_3 = \ldots
\]

Specifically, we define \( U_0 \) as:

\[
U_0 \equiv \frac{1}{A_0} \int_{C_0} \! UdA
\]

where \( C_0 \equiv \text{cell 0}, \quad A_0 \equiv \text{area of cell 0} \)

Now, we apply conservation eqns:

\[
\frac{d}{dt} \int_{C_0} \! UdA + \oint_{\partial C_0} (F_i + G_j) \cdot \hat{n} dS = 0
\]

The time-derivative term can be simplified a little:

\[
\frac{d}{dt} \int_{C_0} \! UdA = A_0 \frac{dU_0}{dt}
\]

The surface flux integral can also be simplified a little:

\[
\oint_{\partial C_0} (F_i + G_j) \cdot \hat{n} dS = \int_a^b (F_i + G_j) \cdot \hat{n} dS + \int_b^c (F_i + G_j) \cdot \hat{n} dS + \int_c^d (F_i + G_j) \cdot \hat{n} dS
\]
Combining these expressions:

\[
A_0 \frac{dU_0}{dt} + \int_a^b (F\bar{i} + G\bar{j}) \cdot \vec{n}ds + \int_b^c (F\bar{i} + G\bar{j}) \cdot \vec{n}ds + \int_c^a (F\bar{i} + G\bar{j}) \cdot \vec{n}ds = 0
\]

No approximations so far!

Now, we make some approximations. Let’s look at the surface integral from \(a \to b\):

\[
\int_a^b (F\bar{i} + G\bar{j}) \cdot \vec{n}ds
\]

The normal can easily be calculated since the face is a straight line between nodes a & b. Recall, the unknowns are stored at all centers. So, what would be a logical approximation for:

\[
\int_a^b \left( F\bar{i} + G\bar{j} \right) \cdot \vec{n}_{ab} dS = ???
\]

Option #1=
Option #2=

Note: Option #1 \( \neq \) option #2 in general.

There is very little difference in practice between these options. Let’s stick with:

\[
\mathcal{I}_{ab} \equiv \int_a^b \left( F\bar{i} + G\bar{j} \right) \cdot \vec{n}_{ab} dS = \left[ \frac{1}{2} \left( F_0 + F_1 \right) i + \frac{1}{2} \left( G_0 + G_1 \right) j \right] \cdot \vec{n}_{ab} \Delta S_{ab}
\]

\[
\mathcal{I}_{bc} \equiv \int_b^c \left( F\bar{i} + G\bar{j} \right) \cdot \vec{n}_{bc} dS = \left[ \frac{1}{2} \left( F_0 + F_2 \right) i + \frac{1}{2} \left( G_0 + G_2 \right) j \right] \cdot \vec{n}_{bc} \Delta S_{bc}
\]

\[
\mathcal{I}_{ca} \equiv \int_c^a \left( F\bar{i} + G\bar{j} \right) \cdot \vec{n}_{ca} dS = \left[ \frac{1}{2} \left( F_0 + F_3 \right) i + \frac{1}{2} \left( G_0 + G_3 \right) j \right] \cdot \vec{n}_{ca} \Delta S_{ca}
\]

Where

\[
F_0 \equiv F(U_0) \quad G_0 \equiv G(U_0)
F_1 \equiv F(U_1) \quad G_1 \equiv G(U_1)
F_2 \equiv F(U_2) \quad G_2 \equiv G(U_2)
F_3 \equiv F(U_3) \quad G_3 \equiv G(U_3)
\]
Finally, we have to approximate $A_0 \frac{dU_0}{dt}$ somehow. The simplest approach is forward Euler:

$$A_0 \frac{dU_0}{dt} + \mathcal{I}_{ab} + \mathcal{I}_{bc} + \mathcal{I}_{ca} = 0$$

And $\mathcal{I}_{ab}$, etc. are defined as:

$$\mathcal{I}_{ab} = \left[ \frac{1}{2} (F_0^n + F_1^n) i + \frac{1}{2} (G_0^n + G_1^n) j \right] \cdot \bar{n}_{ab} \Delta S_{ab}$$

$$F_0^n \equiv F(U_0^n) \quad \text{etc.}$$

$$F_1^n \equiv F(U_1^n)$$

For steady solution, basic procedure is to make a guess of $U$ at $t = 0$ and then iterate until the solution no longer changes. This is called time marching.

**Question**

What assumptions have we made in developing our 2-D Euler Equation Finite Volume Method?