Aeroelastic Prediction of the BSCW using an enhanced OpenFoam-based CFD Solver (Work in progress)

Amin Fereidooni
Anant Grewal
Marcel Grzeszczyk

National Research Council Canada

January 3, 2016
Outline

• Introduction

• OpenFOAM Compressible Solver: rhoCentralFoam

• Enhanced OpenFOAM Coupled Solver

• 2D Simulation: Pitching NACA0012 airfoil

• 3D Simulation: BSCW (case 1 & case 2)
Smart Wing with Integrated Multi-functional Surfaces (SWIMS)

- NRC’s participation in AePW-2 supported by Aeronautics for 21st century (Aero21) program through the SWIMS project
- Multi-disciplinary project coordinated by several groups at NRC
- Focuses on the design of an airfoil with a drooped leading edge

Goals:
- A morphing wing with a continuous surface that adapts to various flight conditions for generating maximum lift and minimum drag
- Drooped nose deflection $\delta_F = 15$ degrees
- About 5% increase in the chord length
• Our group is responsible for performing an **aeroelastic analysis** on a wing with a drooped leading edge

• OpenFOAM is going to be the primary CFD simulation tool for the aeroelastic analysis

• C++ based CFD toolbox

• Simple Structural Dynamic Model: **sixDofRigidBodyMotion** solver

• CFD compressible solver: **rhoCentralFoam**
rhoCentralFoam: segregated solver

Euler equation:

\[
\frac{\partial}{\partial t} \int_{\Omega} W d\Omega + \oint_{\partial\Omega} F_c dS = 0
\]

\[
\begin{align*}
W &= \begin{pmatrix} \rho \\ \rho \vec{U} \\ \rho E \end{pmatrix}, \\
F_c &= \begin{pmatrix} \rho \vec{U} \cdot \vec{n} \\ \rho \vec{U} (\vec{U} \cdot \vec{n}) + p \vec{n} \\ (\rho E + p) \vec{U} \cdot \vec{n} \end{pmatrix}
\end{align*}
\]

Discretized equation:

\[
\frac{d}{dt} (W_i V_i) + \sum_{Faces(i)} F_c \Delta S = 0
\]
Kurganov-Tadmor (KT) and Kurganov-Noelle-Petrova (KNP)

- Convective terms

\[
\int_V \nabla \cdot [u\Psi] dV = \int_S dS \cdot [u\Psi] \approx \sum_f S_f u_f \Psi_f = \sum_f \phi_f \Psi_f
\]

\[
\sum_f \phi_f \Psi_f = \sum_f \left[ (a^+ \phi_f^+ - aSf)\Psi_f^+ + (1 - a^+)\phi_f^- + aSf \right] \Psi_f^-
\]

\[
a^+ = \begin{cases} 
\frac{a_p}{a_p - a_m} & \text{KNP} \\
1 & \text{KT} \\
\frac{1}{2} & \text{KNP}
\end{cases}
\]

\[
aSf = \begin{cases} 
\frac{a_m a_p}{a_p - a_m} & \text{KNP} \\
- \frac{1}{2} \max(|a_m|, |a_p|) & \text{KT}
\end{cases}
\]

\[
a_p = \max \left( \phi_f^+ + c_f^+ |S_f|, \phi_f^- + c_f^- |S_f|, 0 \right)
\]

\[
a_m = \min \left( \phi_f^+ - c_f^+ |S_f|, \phi_f^- - c_f^- |S_f|, 0 \right)
\]
Kurganov-Tadmor (KT) and Kurganov-Noelle-Petrova (KNP)

- Gradient terms

\[
\int_V \nabla \Psi \, dV = \int_S dS \, \Psi \approx \sum_f S_f \Psi_f \\
\sum_f S_f \Psi_f = \sum_f \left[ a^+ S_f \Psi_f^+ + (1 - a^+) S_f \Psi_f^- \right]
\]
Coupled CFD Solver Derived from rhoCentralFoam

• Segregated Solver
\[
\frac{d}{dt}(\vec{W}_iV_i) + \sum_{\text{Faces}(i)} \vec{F}_c \Delta S = 0
\]

• Coupled Solver
\[
[\vec{F}_c]^{n+1} = [\vec{F}_c]^n + \left[\frac{\partial \vec{F}_c}{\partial \vec{W}}\right]^n (\vec{W}^{n+1} - \vec{W}^n) + O\left((\Delta \vec{W})^2\right)
\]
\[
\frac{d}{dt}(\vec{W}_iV_i) + \sum_{\text{Faces}(i)} \left[\frac{\partial \vec{F}_c}{\partial \vec{W}}\right]^n \Delta S \Delta \vec{W} = - \sum_{\text{Faces}(i)} [\vec{F}_c]^n \Delta S
\]
Lower-Upper Symmetric Gauss –Seidel (LU-SGS)

• Approximation

$$(D + L + U)x = b$$

$$(D + L + U) \approx (D + L)D^{-1}(D + U) = (D + U + L + LD^{-1}U)$$

• Two-step process:

  Forward Sweep: $$Dx^1 = b - Lx^1$$
  Backward Sweep: $$Dx = b - Lx^1 - Ux$$
Dual Time Stepping

\[
\frac{d}{dt}(\mathbf{W}_i V_i) + \sum_{\text{Faces}(i)} \left[ \frac{\partial F_c}{\partial \mathbf{W}} \right]^n \Delta S \Delta \mathbf{W} = - \sum_{\text{Faces}(i)} [\tilde{F}_c]^n \Delta S
\]

- **First Order**

\[
\left( \frac{V_i^{n+1}}{\Delta t} + \frac{V_i^{n+1}}{\Delta t^*} + \sum_{\text{Faces}(i)} \left[ \frac{\partial \tilde{F}_c}{\partial \mathbf{W}} \right]^n \Delta S \right) \Delta \mathbf{W}_i^k
\]

\[
= \frac{V_i^{n+1}}{\Delta t} \mathbf{W}_i^k + \frac{V_i^n}{\Delta t} \mathbf{W}_i^n - \sum_{\text{Faces}(i)} [\tilde{F}_c]^n \Delta S
\]

- **Second Order**

\[
\left( \frac{3 V_i^{n+1}}{2 \Delta t} + \frac{V_i^{n+1}}{\Delta t^*} + \sum_{\text{Faces}(i)} \left[ \frac{\partial \tilde{F}_c}{\partial \mathbf{W}} \right]^n \Delta S \right) \Delta \mathbf{W}_i^k
\]

\[
= - \frac{3 V_i^{n+1}}{2 \Delta t} \mathbf{W}_i^k + 2 \frac{V_i^n}{\Delta t} \mathbf{W}_i^n - \frac{1}{2} \frac{V_i^{n-1}}{\Delta t} \mathbf{W}_i^{n-1} - \sum_{\text{Faces}(i)} [\tilde{F}_c]^n \Delta S
\]
2D Simulation: Pitching NACA0012 Airfoil

- Pitching angle $\alpha = 3.16^\circ + 4.59^\circ \sin(2\pi \times 5 \ t)$
- Reduced frequency $k = \frac{\omega c}{2U_\infty} = 0.081$
- Mach number $M = 0.6$
3D Simulation: BSCW (Case 1)

- Steady State
- All the results: Coarse Mesh, Euler solution
- Pressure Coefficient at the 60% Spanwise Location
- Max courant number: 10 and 100
3D Simulation: BSCW (Case 1)

- Steady State
- Pressure Coefficient at the 95% Spanwise Location

Lower Surface

Upper Surface
3D Simulation: BSCW (Case 1)

- Forced Case
- FRF magnitude at the 60% Spanwise Location
- Time steps: $\Delta T = 5 \times 10^{-4}$, number of dual time steps: 200
3D Simulation: BSCW (Case 1)

- Forced Case
- FRF Phase at the 60% Spanwise Location
3D Simulation: BSCW (Case 1)

- Forced Case
- FRF magnitude at the 95% Spanwise Location

![Graph showing FRF magnitude comparison between Lower and Upper Surfaces](image-url)
3D Simulation: BSCW (Case 1)

- Forced Case
- FRF Phase at the 95% Spanwise Location

![Graphs showing FRF Phase for Lower and Upper Surfaces.](image)
3D Simulation: BSCW (Case 2)

- Steady State
- Coarse Mesh
- Pressure Coefficient at the 60% Spanwise Location
- Max courant number: 10 and 100

Lower Surface

Upper Surface
3D Simulation: BSCW (Case 2)

- Steady State
- Coarse Mesh
- Pressure Coefficient at the 95% Spanwise Location
- Max courant number: 10 and 100
3D Simulation: BSCW (Case 2)

- Flutter case
- Coarse Mesh
- How did we start the dynamic case?

\[ \Delta T = 5 \times 10^{-3} \]
3D Simulation: BSCW (Case 2)

- Flutter case
- Coarse Mesh
Future Work

• Complete Case 2 flutter analysis for inviscid case
• Further code development to consider viscous flux Jacobian
• Re-run cases 1, 2 with viscous forces
• Attempt URANS simulations for case 3
THANK YOU