FUN3D v13.4 Training

Session 19: Multidisciplinary Design

Li Wang

Session Scope

- What this will cover
  - Overview of multidisciplinary optimization of rotors with flexible blades
    - Coupling with rotorcraft Comprehensive Analysis code (DYMORE)
  - Overview of aeroacoustics optimization for rotorcraft noise reduction
    - Coupling with noise propagation and analysis code (ANOPP2)
- What will not be covered
  - DYMORE operation
  - ANOPP2 operation
- What should you already know
  - Basic aerodynamic shape optimization
  - Surface parametrization
  - Time-dependent aerodynamic simulations
  - Overset-dynamic-deforming grid operations
  - Rudimentary rotorcraft aeromechanics

Introduction

- Background
  - Analysis of aircraft systems involves many disciplines: aerodynamics, structural dynamics, aeroacoustics, etc. - rotorcraft aeromechanics is a typical example
  - Gradient-based, multidisciplinary design optimization (MDO) requires sensitivity analysis for all disciplines
  - Adjoint-based sensitivity analysis capabilities for
    - Coupled aero/structure FUN3D/DYMORE system
    - Coupled aero/acoustics FUN3D/ANOPP2 system
- Status
  - FUN3D/DYMORE
    - Loosely and tightly coupled for analysis / only tightly coupled for sensitivity
    - Complex-variable sensitivities for DYMORE, quadratic dependence on time steps, adjoint formulation is ongoing
  - FUN3D/ANOPP2
    - Initial assessment, gaining experience

FUN3D/DYMORE – Overview (1/2)

- DYMORE is an established Comprehensive Analysis code
  - Open-source nonlinear flexible multibody dynamics code
  - Developed and managed by *Prof. Olivier Bauchau* at U. of Maryland
  - Provides static, dynamic, stability, and trim analyses of rotorcraft configurations
  - FE structural dynamics, low-fidelity internal aerodynamics model
- *Best practice* MDO for rotorcraft analysis
  - Loose-coupling trimmed solution initializes tight-coupling analysis
    - Alleviates initial transients effects
    - Eliminates nonphysical structure blade defections
  - Design time interval is set within the FIRST rotor revolution
    - Shortens CFD simulation
    - Comparable cost for DYMORE and FUN3D flow sensitivities
  - Affordable computational cost for MDO of rotorcraft in level-flight conditions
  - Verification for long-time simulations in periodic regime

FUN3D/DYMORE – Overview (2/2)

- FUN3D drives DYMORE sensitivity analysis
  - Most efficient if number of processing cores is equal to (or greater than) total DYMORE DOFs (6 DOFs per airstation)
- FUN3D drives sensitivities to control parameters, e.g., collective and cyclic pitch controls
  - Use DYMORE input decks in FUN3D **`fsi_tight_coupling.input`** file (cover in slide 8)
- Design parameters
  - Shape design parameters (referred to earlier sessions)
  - Pitch control angles
    - Set in **`trimming.data`** and the upper/lower bounds
    - Initialized (radians) by loose-coupling solutions for baseline configuration
    - Automatically updated in DYMORE input decks during optimization

FUN3D/DYMORE – Compilation

- Compiling DYMORE 5
  - Add **-DCOMPLEX_STEP** to **CFLAGS** parameter in DYMORE 5 makefile
  - Provide access to hdf5 module: **LIB=/path/to/hdf5/libhdf5.so**
  - Generated static library with name **libdymore.a**
- Compiling FUN3D
  - Use **--with-hdf5=/path/to/hdf5** and **--with-dymore=/path/to/dymore**
  - FUN3D needs the following libraries in those locations:
    - **libhdf5.so**, *consistent* with hdf5 library used for DYMORE
    - **libdymore.a**, *complex* mode enabled (may be soft link)
  - Use other necessary links for rotorcraft simulations and optimization, e.g.,

  --with-dirtlib=/path/to/dirtlib, **--with-suggar=/path/to/suggar**, and **--with-SNOPT=/path/to/snopt** (if using SNOPT optimizer)
  - After compilation, *config.h* file should have **#define HAVE_DYMORE_HYBRID 1**

FUN3D/DYMORE – Inputs (1/2)

- Follow conventions of body definitions in FUN3D **`moving_body.input`** file
  - Forward path defines bodies with *composite* motions, e.g.,

```
&body_definitions
  output_transform = .true.,           ! output transform matrix (rigid)
  body_name(1)     = 'rotor1_blade1', ! name must be in quotes
  mesh_movement(1) = 'rigid+deform',  ! for elastic rotor
...
```

  - Adjoint path defines *component* motions for one body and builds "*parent-child*" relationship, e.g.,

```
&body_definitions
  output_transform = .true.,           ! optional for output
  body_name(1)     = 'rotor1_blade1', ! name must be in quotes
  mesh_movement(1) = 'rigid',          ! parent component
  mesh_id(1)       =  1                ! actual moving-body index
  ...
  body_name(2)     = 'rotor1_blade1', ! name must be in quotes
  mesh_movement(2) = 'deform',         ! child component
  parent_name(2)   = 'rotor1_blade1'  ! name of the parent
  mesh_id(2)       =  1                ! actual moving-body index
...
```

FUN3D/DYMORE – Inputs (2/2)

- Add to **`fsi_tight_coupling.input`** file DYMORE 5 (main) input decks to enable computation of trim derivatives

```
./dymore5_uh60/uh60_4bltight.dym                ! Main DYMORE input
1.0
1
./dymore5_uh60/uh60_4blrd.dym                   ! enabling kinematics dv.
./dymore5_uh60/uh60_4bltighttrimDV1.dym         ! (theta0  i 1.0e-50)
./dymore5_uh60/uh60_4bltighttrimDV2.dym         ! (theta1c i 1.0e-50)
./dymore5_uh60/uh60_4bltighttrimDV3.dym         ! (theta1s i 1.0e-50)
29  35  41                                      ! line numbers for theta0,
                                                ! theta1c, theta1s entries
```

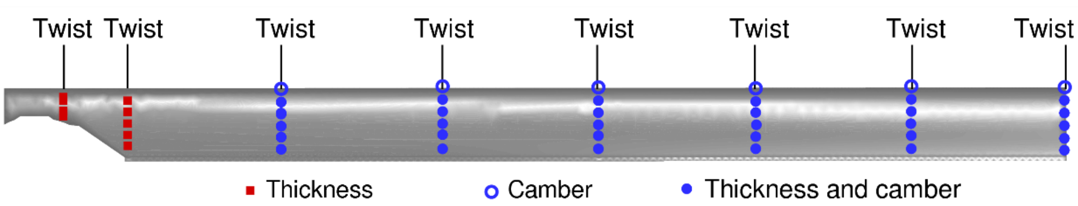
- Trim constraints defined in **`rubber.data`** file, including rotor thrust, rolling and pitching moments (design-function names are, **`rtr_thrust`**, **`cmx`**, and **`cmy`**, respectively)
- Surface parameterization and shape design parameters are described (referred to earlier sessions)

FUN3D/DYMORE – Design Example (1/2)

HART-II rotorcraft in descent flight, Adv. Ratio = 0.15, AOA = 4.5°

Gradient-based optimization: minimize rotor power, subject to thrust and rolling and pitching moments constraints

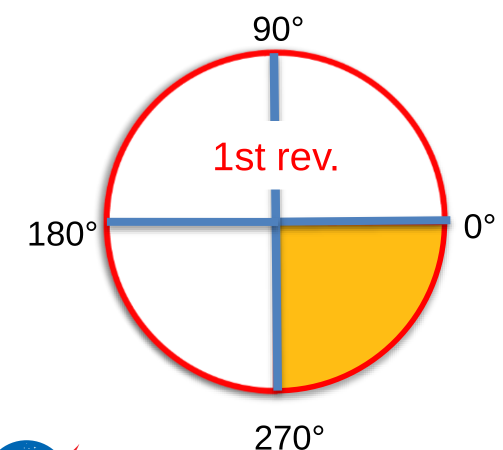
Shape and trim design parameters



Design parameters: 8 twist, 36 camber, 35 thickness, (thickness at blade tip deactivated), and 3 trim control angles

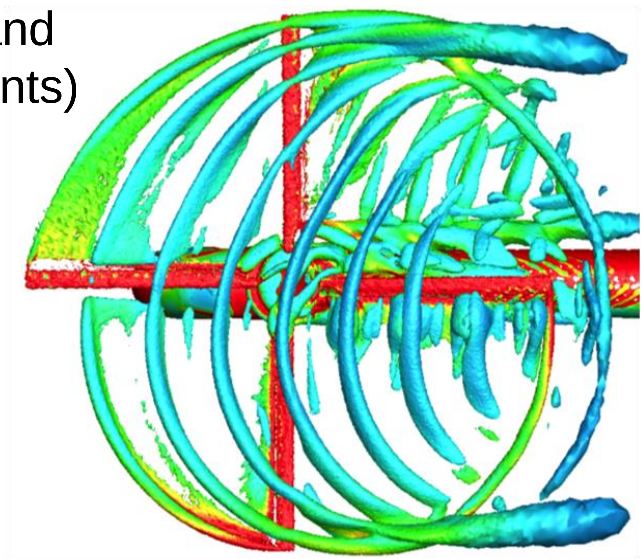
Overset mesh: 7 million nodes

Four design outputs: objective function (rotor power) and three constraints (rotor thrust, rolling and pitching moments)

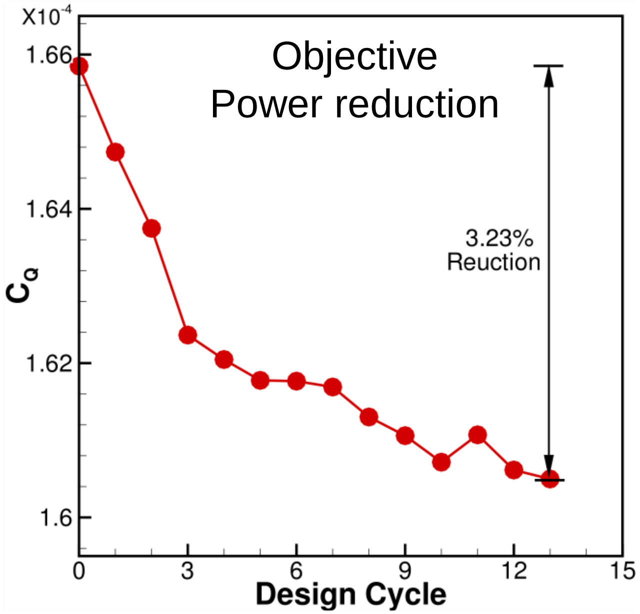


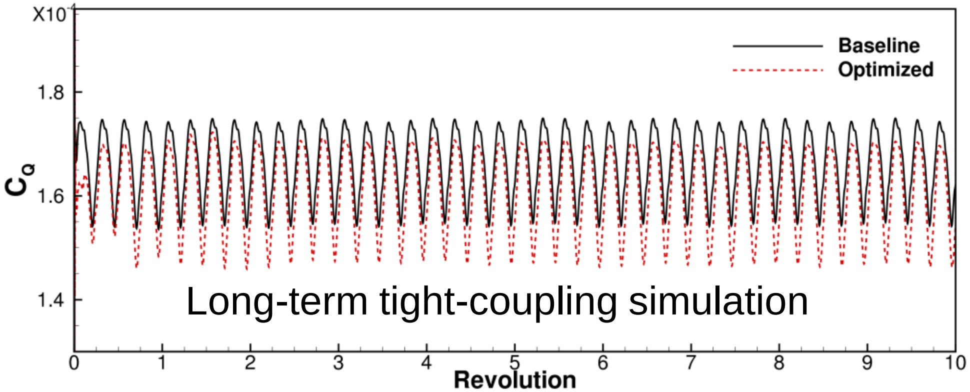
Design outputs defined within 1st rotor revolution

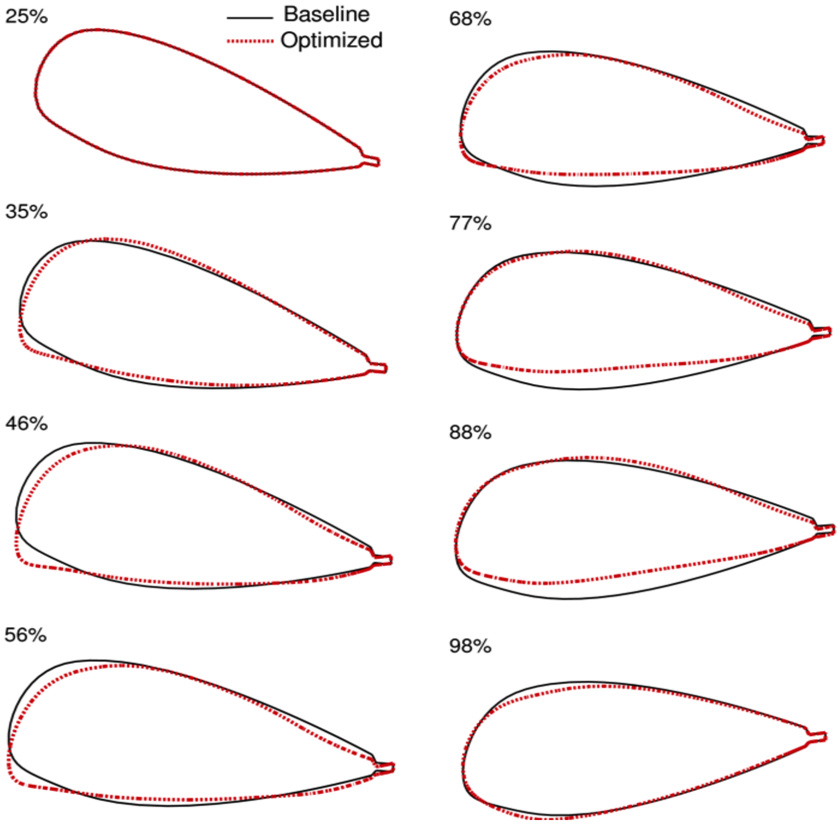
One design cycle takes 7 wall-clock hours on ~2000 processing cores



FUN3D/DYMORE – Design Example (2/2)







Blade cross-section geometry (3:1 scale)

After 13 design cycles: *rotor power* reduced by 3.23%; all *constraints* satisfied
Long-term simulations: *improved performance* preserved; *trim conditions* maintained

FUN3D/ANOPP2 – Overview (1/2)

- ANOPP2 - Aircraft NOise Prediction Program toolkit developed at NASA
  - Noise propagation and prediction methods
  - Various acoustic noise metrics at a set of observer locations
    - User-interface codes specify the acoustic function and observer locations
    - FUN3D only uses Ffowcs Williams and Hawkings (FWH) acoustics models in ANOPP2
  - Sensitivity analysis
- FUN3D/ANOPP2 analysis
  - FUN3D solves the flow equations and outputs flow data on disk
    - Dimensionalized quantities: surface pressure and geometry data, physical times, etc. - use **`&fwh_acoustic_data`** namelist in **`fun3d.nml`** file as per slide 14
  - ANOPP2 reads data from disk, computes acoustic metrics
- FUN3D/ANOPP2 sensitivity analysis
  - ANOPP2 computes acoustic sensitivities wrt pressure and surface grid, sends them to FUN3D to form RHS for adjoint equations
  - FUN3D solves adjoint equations, computes sensitivities to design parameters

FUN3D/ANOPP2 – General Info (2/2)

- Configuring FUN3D/ANOPP2
  - Use **`--with-anopp2=/path/to/anopp2`** and **`--with-anopp2-user=/path/to/anopp2_user`**
  - FUN3D needs the following libraries (may be soft links):

    `libANOPP2.so, libAFFI.so` and **`libAFAI.so`**

    and interface codes (may be soft links):

    `ANOPP2.api.f90, AFFI.api.f90`, and **`AFAI.api.f90`**
  - Set the environment variable **`LD_LIBRARY_PATH`**, e.g.,

```
setenv LD_LIBRARY_PATH ${LD_LIBRARY_PATH}:/path/to/anopp2
setenv LD_LIBRARY_PATH ${LD_LIBRARY_PATH}:/path/to/anopp2_user
```

  - ANOPP2 user-interface codes specify directory for CFD output data. Soft link is more convenient than copying data from flow-run directory.

FUN3D/ANOPP2 – Inputs (1/2)

- Control of ANOPP2 acoustic function
  - Set design function as **anopp2** in **rubber.data** file

```
Cost function (1) or constraint (2)
  1
If constraint, lower and upper bounds
  0.0  0.0
Number of components for function   1
  1
Physical timestep interval where function is defined
  361  1080                         ! time-step interval for funct. eval.
Composite function weight, target, and power
  1.0  0.0  1.0
Components of function 1: boundary id (0=all)/name/value/weight/target/power
  0 anopp2  1.0   1.0   0.0   2.0
Current value of function   1
  1.0
Current derivatives of function wrt global design variables
  0.0
...
```

  - Set control angles (in deg.) in **trimming.data** file
  - Specify **trim_control(ibody)='design'** in **&body_definitions** namelist in FUN3D **moving_body.input** file

FUN3D/ANOPP2 – Inputs (2/2)

- Control of flow data (binary) output
  - use **`&fwh_acoustic_data`** namelist in **`fun3d.nml`** file

```
&fwh_acoustic_data
 anopp2_data_format     = .true.
 an2_length_factor      = 1.000          ! dimensionalization
 an2_c_ref              = 340.297
 an2_rho_ref            = 1.225
 an2_write_normals      = .false.
 an2_double_precision   = .true.
 an2_start_iter         = 361            ! starting timestep
 an2_stop_iter          = 1080           ! last timestep for output
 fwh_data_freq          = 1
 append_to_prior_data   = .false.
 n_fwh_bndry            = -1
 fwh_bndry_list         = '1,3,5,7'      ! boundaries for output
 geom_time_variation(1) = 'aperiodic_all'
 data_time_variation(1) = 'aperiodic_all'
/
```

- After completion of flow solve, **`[project]_00#_anopp2.bin`** files should be in the flow-run directory; # denotes specific boundary

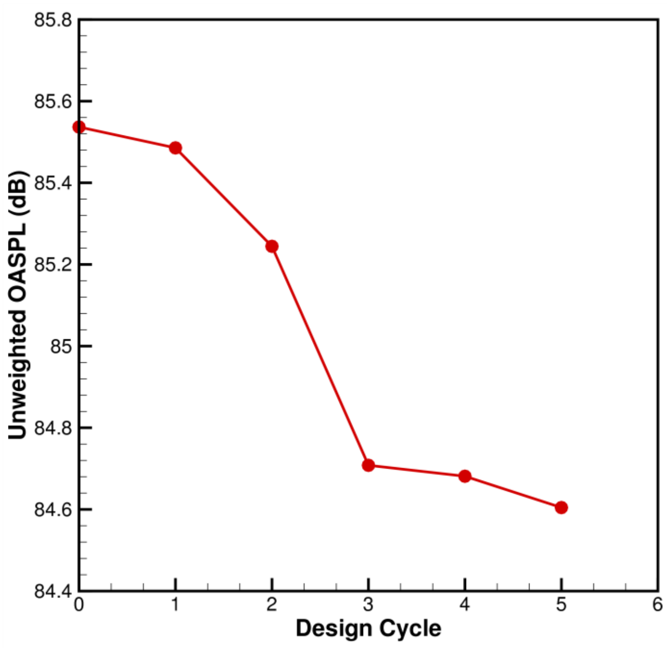
FUN3D/ANOPP2 – Design Example

HART-II rotorcraft in forward flight, tip M = 0.6387, Adv. Ratio = 0.1566, AOA = 0

In-plane observer, 10R from rotation center

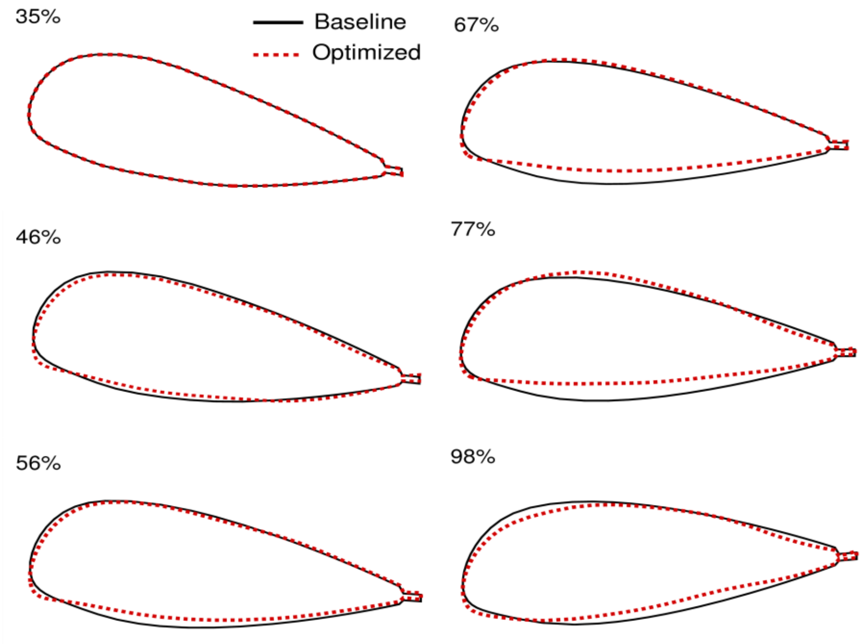
Gradient-based optimization: minimize unweighted overall sound pressure level (in dB) at observer, subject to thrust and rolling and pitching moments constraints

After 5 design cycles :1 dB noise reduction; all constraints satisfied



Acoustic objective function

Design parameters:
8 twist
36 camber
35 thickness
as per slide 9



Blade cross-section geometries

Session Summary

- MDO for FUN3D/DYMORE and FUN3D/ANOPP2 systems
- Key inputs and controls for setting up a rotorcraft design optimization based on the MDO systems
- Design optimization demonstrations showing basic capabilities
  - Constrained Aero/Structure design for rotor power reduction
  - Constrained Aero/Acoustics design for low-noise rotorcraft design
- We would like to help you if you are interested in MDO using FUN3D