EUROGEN 2019, Guimarães, 12-14 September 2019

The cybermatrix protocol: a link between classical aircraft design and formal multidisciplinary optimization

Caslav Ilic, Andrei Merle, Mohammad Abu-Zurayk, Martin Leitner, Matthias Schulze, Andreas Schuster, Michael Petsch, Sebastian Gottfried
A merger of two approaches

Classic aircraft design
- Focus on process automation, many disciplines, data modeling
- No specific focus on high-performance computing (HPC)
- No formal optimality criteria, suboptimal designs by construction

Formal multidisciplinary optimization
- Focus on analysis fidelity, modeling constraints, and adding disciplines
- Explicit consideration of optimality criteria and often high HPC use
- Simplified tools, poorly scalable in number of disciplines/experts

The proposed solution establishes a link between the two approaches
- Developed within the DLR project VicToria
- Optimality criteria explicit, but applied in a heuristic manner
- Parallelism from ground up, both in expert participation and in use of HPC
- Implementation (human) and execution (computer) phases with analogous communication and control in a matrix-like structure → cybermatrix
The design equation

Any design process can be viewed as an approximate optimization process:

\[
\frac{df(p)}{dp} - \frac{dc(p)}{dp}^T q = 0, \quad c(p) = 0
\]

where \( f \) goal (\( \mathbb{R}^1 \)), \( c \) constraints (\( \mathbb{R}^m \)), \( p \) design parameters (\( \mathbb{R}^n \)), \( q \) goal-to-constraint sensitivities (Lagrange multipliers, \( \mathbb{R}^m \)) → approximate first KKT optimality condition

Expanded for three disciplines \( A, B, C \) and global goal function \( F \) (\( \mathbb{R}^1 \)):

\[
\begin{align*}
\frac{\partial F}{\partial f_A} \frac{df_A}{dp_A} + \frac{\partial F}{\partial f_B} \frac{df_B}{dp_A} + \frac{\partial F}{\partial f_C} \frac{df_C}{dp_A} - \frac{dc_A}{dp_A} q_A - \frac{dc_B}{dp_A} q_B - \frac{dc_C}{dp_A} q_C &= 0, \quad c_A = 0 \\
\frac{\partial F}{\partial f_A} \frac{df_A}{dp_B} + \frac{\partial F}{\partial f_B} \frac{df_B}{dp_B} + \frac{\partial F}{\partial f_C} \frac{df_C}{dp_B} - \frac{dc_A}{dp_B} q_A - \frac{dc_B}{dp_B} q_B - \frac{dc_C}{dp_B} q_C &= 0, \quad c_B = 0 \\
\frac{\partial F}{\partial f_A} \frac{df_A}{dp_C} + \frac{\partial F}{\partial f_B} \frac{df_B}{dp_C} + \frac{\partial F}{\partial f_C} \frac{df_C}{dp_C} - \frac{dc_A}{dp_C} q_A - \frac{dc_B}{dp_C} q_B - \frac{dc_C}{dp_C} q_C &= 0, \quad c_C = 0
\end{align*}
\]
Sidenote: An interpretation of Lagrange multiplier

Example: Find \( p, q \) for an aircraft that minimize mission fuel expenditure \((m_f)\) under max. take-off field length \((s_{TO})\) and other constraints

\[
m_f^* = \min_{p,q} m_f
\]

\[
s_{TO} \leq s_{TO}^* \quad (\rightarrow q_{s_{TO}})
\]

etc.

The measure of how much the goal would change per unit constraint change

An information highly sought for by designers

Why are we never reporting it?
The representation protocol

- Since the design equation is usually implied, use a schematic representation.
- Let each row belong to one discipline (all related to its design parameters).

Indicator that also design dependencies (Jacobian-like data) is exchanged, and not only consistency dependencies (state-like data).

Data dependencies: discipline A takes from discipline B.

Backbone-line indicating that the row belongs to one discipline to converge it to zero.

Indicator that the disciplinary design also takes into account global goal.
The communication protocol

- Each disciplinary design process can have any form, only iteration assumed
- Add to it data exchange points and initial data estimators

Different disciplines may have different exchange periods

Selection of rows, steps and exchange periods recover any possible “MDO architecture”
- In practice always a hybrid architecture

practical visualisation: the base period
Sidenote: An iteration with gradient based processes

- Assume all disciplinary processes A, B, C are gradient-based processes, using (different) off-the-shelf gradient-based optimization algorithms
- Step $k$ of A (analogous for B, C) could be a single gradient computation plus the associated line/trust-region search

$$\frac{\partial F}{\partial f_A} \frac{df_A}{dp_A} \bigg|_{p_A^{k+1}, p_{B,C}^k} + \left( \frac{\partial F}{\partial f_B} \frac{df_B}{dp_A} + \frac{\partial F}{\partial f_C} \frac{df_C}{dp_A} - \frac{dc_B}{dp_A} q_B - \frac{dc_C}{dp_A} q_C \right) \bigg|_{p_A^{k+1}, p_{B,C}^k} = 0$$

which an off-the-shelf optimizer can be tricked to perform by modifying the original disciplinary goal function in the step $k$ as

$$\tilde{f}_A \bigg|_{p_A^{k+1}, p_{B,C}^k} = \frac{\partial F}{\partial f_A} \bigg|_{f_A^k} + \left( \frac{\partial F}{\partial f_B} \frac{df_B}{dp_A} + \frac{\partial F}{\partial f_C} \frac{df_C}{dp_A} - \frac{dc_B}{dp_A} q_B - \frac{dc_C}{dp_A} q_C \right) (p_A^{k+1} - p_A^k)$$

linearized “penalty” – how much to “give up” for other disciplines

- The same idea and rationale as e.g. for coupled-adjoint gradient evaluation
- Jacobi/Gauss-Seidel fixed-point block-iteration to couple processes, each using the best method for its internal iteration, with “rhs modification”
A realization on HPC clusters

- A cybermatrix HPC process integration framework in development
  - Starts disciplinary processes, assigns resources, monitors progress
  - Triggers data exchanges and determines global convergence
- Disciplinary experts do not work with the framework directly
  - No need to learn yet another integration framework
  - Only provide **input collector** scripts to copy data from other disciplines
  - The whole MDO process implementation: a directory of input collectors
- Maintainable by standard software engineering tools and practices
  - Set of input collectors under source version control
  - Integration framework is an **interpreter** of the set of collectors and some meta-data (data exchange periods, etc)
- Currently data exchange performed over parallel on-disk file system
  - Parallel in-memory or area-network file system possible in principle
  - No changes to disciplinary processes in any case
On-machine appearance

```
/home_cluster/someuser/
toolbox/
mdoproc-0.5/
sertlib/
mdoproc
mdoproc.senv
...
inpcoll/
  exec-dspA/
    toolspec
      from-input
      from-dspB
      from-dspC
  exec-dspB/
    toolspec
      from-input
      from-dspA
      from-dspC
  exec-dspC/
    toolspec
      from-input
      from-dspA
      from-dspB
example/
fullac1/
input/
...
```

setup and call the HPC integration framework

under source version control

input collectors (data fetching scripts)
and tool cspecs
(links to standalone disciplinary procs.
and definition of data exchange intervals)

working examples, unit-like tests
Example: MDO of a long-range transport aircraft

- Large twin-engine wide-body long-range transport aircraft
- Wing-body-tail-pylon-flow through nacelle
- 250 t max. take-off mass class

Global goal function: minimize fuel consumption

Involved disciplinary processes:
- Overall aircraft wing planform design (oad)
- Aerodynamic design of wing airfoils (aero)
- Structural member sizing of wing and tail (struct)
- Determination and evaluation of design loads (loads)
Example: Problem setup

**oad**
- Rolling trade study (tuned trust reg. SQP)
- CAD+ROM wing shape, < 10 design parameters
- Minimize mission fuel
- Step: one QP approx. and trust reg. step

**struct**
- Fully-stressed design
- Global FEM, 42,000 els
- Model region thicknesses, 364 design parameters
- Minimize mass for limit strength, buckling per LC
- Step: one full sizing

**aero**
- Adjoint aeroelastic optimization
- RANS flow, mesh 5,900,000 pts
- CAD+ROM airfoil shapes, 126 design parameters
- Minimize drag at trimmed flight
- Step: one gradient and line search

**loads**
- Dynamic gust, turbulence
- Dynamic FEM, 1,060 DoF
- Panel aero, 1,160 boxes
- 1,200 LCs / 2 MCs
- No goal/cons./design par.
- Step: one full evaluation

Diagram:
- Total mass
- Total drag
- Airfoil shapes
- Wing planform
- Design loads
- Global FEM, CoG
- Dynamic FEM, MCs
Example: Optimization results

- Still robustness problems on planform variations, so planform fixed → oad only for evaluation.

- Total run time: **98 hours** on **192 cores**
  - Base period duration: 13.9 h avg
  - Drag reduction (-7.2%) more significant than mass increase (1.6% wing, 0.16% total), resulting in mission fuel reduction (-6.9%)
    - Wing sections slightly retwisted and reshaped to reduce shock waves
    - Somewhat less favorable spanwise load distribution results in higher design loads
    - Variation in number of design load cases not large, but not negligible

- What is the baseline for comparison?
  - Time 0 on wall-time axis has no meaning
  - Intention-dependent: here result of an optimization with fixed aero design param.
Sidenote: Jacobi, Gauss-Seidel, mixed iteration

- Performed with a coarser CFD mesh

What does this reveal about the problem?
Sidenote: Which process "must" run before which?

- Just a difference in time to convergence
- Though if multiple optima, could fall into a different one
Conclusions and outlook

 firma of a cybermatrix-based MDO process demonstrated
  - Aero-structural approximate overall aircraft optimization with
    configuration-dependent variable number of design load cases
  - Maneuver and gust loads process following certification regulations
  - CAD-based shape parametrization through reduced order modeling

Improvement to the core process
  - More robustness in local design on planform variations
  - More flight points and powered engine for aerodynamic design
  - Control laws and high-fidelity corrections for loads
  - More design dependencies (Jacobian-like information)

Beyond the core process
  - Higher fidelity structural modeling (separate wing/fuselage disciplines)
  - Tighter geometry and mass synthesis (aircraft synthesis discipline)
  - Configuration-dependent engine conceptual design (engine discipline)
  - Flutter analysis (to eliminate planforms exhibiting inherent flutter)
Thank you for your attention!

...plans of penguins and people...