

# Industrial Strength Optimization

**Mark Abramson**, USAF Institute of Technology

**Charles Audet, Gilles Couture**,  
École Polytechnique. de Montréal

**John Dennis**, Rice University

**Andrew Booker, Evin Cramer**,

**Paul Frank, Joerg Gablonski**, Boeing Phantom Works

# Industrial Strength Optimization

**Mark Abramson**, USAF Institute of Technology

**Charles Audet, Gilles Couture**,  
École Polytechnique. de Montréal

**John Dennis**, Rice University

**Andrew Booker, Evin Cramer**,

**Paul Frank, Joerg Gablonski**, Boeing Phantom Works

Thanks to: **AFOSR, Boeing, LANL, SANDIA, ExxonMobil, NSF**

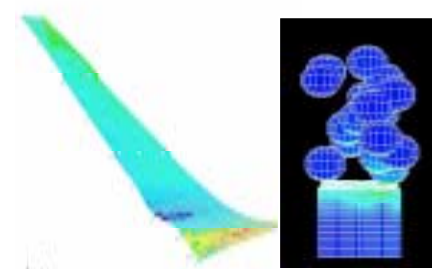
# Design Explorer Applications



Helicopter Rotor Design



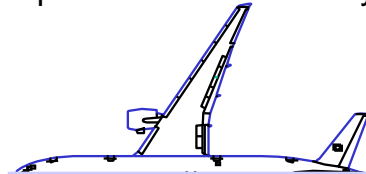
Space Station Power System



Shot peen forming of wing skins



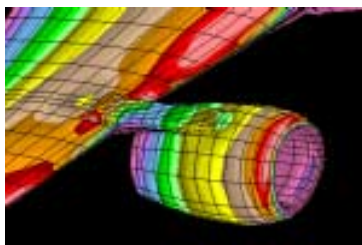
Aerospike Nozzle



Multidisciplinary wing planform design



3-D Fighter Aerodynamics



Engine Nozzle Performance



777 Engine Duct Seals



Machining, riveting, and drilling database

# The target optimization problem

The design problem is:

$$\begin{aligned} & \text{minimize} && f_p(x) \\ & \text{subject to} && x \in X \cap \{x \in \mathfrak{R}^n : C_p(x) \leq 0\}, \end{aligned}$$

# The target optimization problem

The design problem is:

$$\begin{aligned} & \text{minimize} && f_p(x) \\ & \text{subject to} && x \in X \cap \{x \in \mathfrak{R}^n : C_p(x) \leq 0\}, \end{aligned}$$

There are parameters and variables:

- ◆ Contextual parameters  $p$ , supposedly fixed, and

# The target optimization problem

The design problem is:

$$\begin{aligned} & \text{minimize} && f_p(x) \\ & \text{subject to} && x \in X \cap \{x \in \mathfrak{R}^n : C_p(x) \leq 0\}, \end{aligned}$$

There are parameters and variables:

- ◆ Contextual parameters  $p$ , supposedly fixed, and
- ◆ Optimization or design or control variables  $x$

# Properties of the target problem

- ◆  $p$  is fixed in the simulation, but it is subject to uncertainty because of incomplete knowledge (e.g. material properties) or because the product is used differently (e.g., the altimeter is off by 5%)

# Properties of the target problem

- ◆  $p$  is fixed in the simulation, but it is subject to uncertainty because of incomplete knowledge (e.g. material properties) or because the product is used differently (e.g., the altimeter is off by 5%)
- ◆  $x \in X$  must be satisfied when the underlying simulations for  $f_p, C_p$  are called.  $X$  is simple, say a polygon

# Properties of the target problem

- ◆  $p$  is fixed in the simulation, but it is subject to uncertainty because of incomplete knowledge (e.g. material properties) or because the product is used differently (e.g., the altimeter is off by 5%)
- ◆  $x \in X$  must be satisfied when the underlying simulations for  $f_p, C_p$  are called.  $X$  is simple, say a polygon
- ◆  $C_p(x) \leq 0$  only has to hold at the solution

# Properties of variables and constraints

- ◆ Function evaluations typically involve numerically linking legacy PDE solvers.

# Properties of variables and constraints

- ◆ Function evaluations typically involve numerically linking legacy PDE solvers. Evaluations may fail unexpectedly

# Properties of variables and constraints

- ◆ Function evaluations typically involve numerically linking legacy PDE solvers. Evaluations may fail unexpectedly
- ◆ There are few correct digits.

# Properties of variables and constraints

- ◆ Function evaluations typically involve numerically linking legacy PDE solvers. Evaluations may fail unexpectedly
- ◆ There are few correct digits. Accurate approximation of derivatives is problematic

# Properties of variables and constraints

- ◆ Function evaluations typically involve numerically linking legacy PDE solvers. Evaluations may fail unexpectedly
- ◆ There are few correct digits. Accurate approximation of derivatives is problematic
- ◆ Some variables are **categorical**, i.e., the simulations only run with certain discrete choices, e.g., choice and sequencing of manufacturing processes - Lecture 3.

# Uncertainty in design variables

Instead of specified design  $x_*$ , the delivered product might be the design specified by a nearby  $x$

# Uncertainty in design variables

Instead of specified design  $x_*$ , the delivered product might be the design specified by a nearby  $x$

Very common in engineering design. Good data may be tough to get

# Uncertainty in design variables

Instead of specified design  $x_*$ , the delivered product might be the design specified by a nearby  $x$

Very common in engineering design. Good data may be tough to get

Robustify against such “variability” by giving up some optimality at a given point in return for optimality over nearby designs

# Uncertainty in context variables

Products are designed for optimal performance under specified operating conditions, loads, etc

## Uncertainty in context variables

Products are designed for optimal performance under specified operating conditions, loads, etc

We would be willing to give up some performance at the exact conditions  $p$  in return for optimal performance over all likely nearby conditions  $q$  - or we can constrain probability of failure for the nominal  $p$

# Uncertainty in context variables

Products are designed for optimal performance under specified operating conditions, loads, etc

We would be willing to give up some performance at the exact conditions  $p$  in return for optimal performance over all likely nearby conditions  $q$  - or we can constrain probability of failure for the nominal  $p$

Seems to be “chance constrained” formulation - nothing new

## Hubris or Sanguinity?

Both types of uncertainty can be modelled for optimization by incorporation into  $f, C$

## Hubris or Sanguinity?

Both types of uncertainty can be modelled for optimization by incorporation into  $f, C$

$$\begin{array}{ll} \text{minimize} & \hat{f}(x) \\ \text{subject to} & x \in \hat{X} \cap \{x \in \mathbb{R}^n : \hat{C}(x) \leq 0\}, \end{array}$$

where  $\hat{f}(x), \hat{c}_j(x)$  incorporate uncertainty modelled by the user's scheme of choice

## Hubris or Sanguinity?

Both types of uncertainty can be modelled for optimization by incorporation into  $f, C$

$$\begin{array}{ll} \text{minimize} & \hat{f}(x) \\ \text{subject to} & x \in \hat{X} \cap \{x \in \mathbb{R}^n : \hat{C}(x) \leq 0\}, \end{array}$$

where  $\hat{f}(x), \hat{c}_j(x)$  incorporate uncertainty modelled by the user's scheme of choice

Still belongs to target class, Bring 'em on

## Hubris or Sanguinity?

Both types of uncertainty can be modelled for optimization by incorporation into  $f, C$

$$\begin{array}{ll} \text{minimize} & \hat{f}(x) \\ \text{subject to} & x \in \hat{X} \cap \{x \in \mathbb{R}^n : \hat{C}(x) \leq 0\}, \end{array}$$

where  $\hat{f}(x), \hat{c}_j(x)$  incorporate uncertainty modelled by the user's scheme of choice

Still belongs to target class, Bring 'em on for now, but seek tighter coupling

## A confidence builder

Meckesheimer, Booker, and Torng report very good performance of our approach on some structural engineering problems in dealing with probability of failure by constraints or as the objective

“Reliability based design optimization using Design Explorer” available from [Andrew.J.Booker@boeing.com](mailto:Andrew.J.Booker@boeing.com)

## A confidence builder

Meckesheimer, Booker, and Torng report very good performance of our approach on some structural engineering problems in dealing with probability of failure by constraints or as the objective

“Reliability based design optimization using Design Explorer” available from [Andrew.J.Booker@boeing.com](mailto:Andrew.J.Booker@boeing.com)

From this point on, assume formulation incorporates uncertainty

## More problem structure

Typically,  $f_p(x)$  is actually  $\tilde{f}_p(x, u(x))$ , where  $u(x)$  are ancillary variables gotten from some simulation with input  $x$

## More problem structure

Typically,  $f_p(x)$  is actually  $\tilde{f}_p(x, u(x))$ , where  $u(x)$  are ancillary variables gotten from some simulation with input  $x$

I.e., given  $x$ , solve  $F_p(x, u) = 0$  for  $u(x)$ . Usually a discretized PDE or several such legacy solvers. This is a major reason why  $f_p(x)$  fails to evaluate - the scheme to couple the solvers fails to compute  $u(x)$

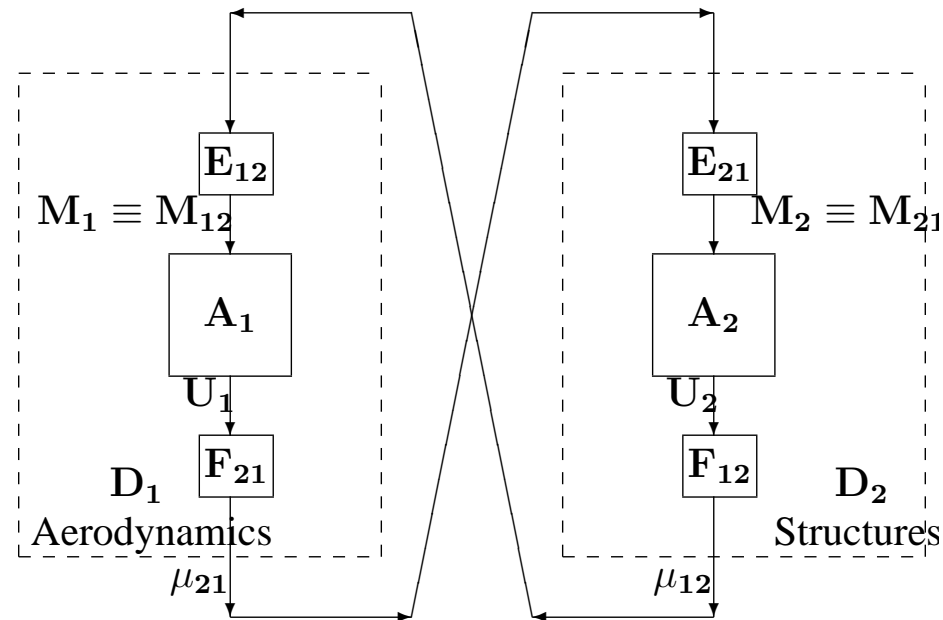
## More problem structure

Typically,  $f_p(x)$  is actually  $\tilde{f}_p(x, u(x))$ , where  $u(x)$  are ancillary variables gotten from some simulation with input  $x$

I.e., given  $x$ , solve  $F_p(x, u) = 0$  for  $u(x)$ . Usually a discretized PDE or several such legacy solvers. This is a major reason why  $f_p(x)$  fails to evaluate - the scheme to couple the solvers fails to compute  $u(x)$

These are called MDO problems, and they are hard...

# MDA = Multidisciplinary analysis



- ◆ Disciplines  $D_1, D_2$  with analysis codes  $A_1, A_2$
- ◆  $\mu_{ij}$  is compressed info from  $D_j$  needed by  $D_i$

# Mathematical formulation of MDA

Write  $F(x, u) = 0$ ,

# Mathematical formulation of MDA

Write  $F(x, u) = 0$ , but solve  $F(x, M_1, M_2) = 0$  for  $M$ , where

$$F_1(x, M_1, M_2) = E_{21} \circ F_{21} \circ A_1(M_1) - M_2,$$

$$F_2(x, M_1, M_2) = E_{12} \circ F_{12} \circ A_2(M_2) - M_1$$

# Mathematical formulation of MDA

Write  $F(x, u) = 0$ , but solve  $F(x, M_1, M_2) = 0$  for  $M$ , where

$$F_1(x, M_1, M_2) = E_{21} \circ F_{21} \circ A_1(M_1) - M_2,$$

$$F_2(x, M_1, M_2) = E_{12} \circ F_{12} \circ A_2(M_2) - M_1$$

- ◆  $A_1$  finite difference and  $A_2$  finite element maybe

# Mathematical formulation of MDA

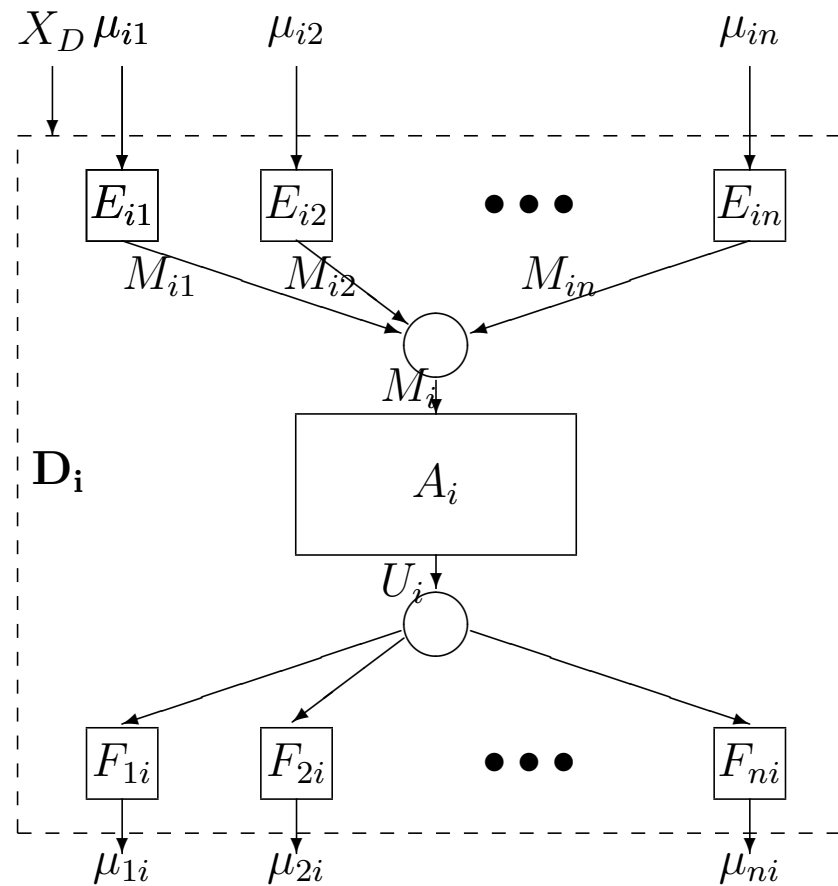
Write  $F(x, u) = 0$ , but solve  $F(x, M_1, M_2) = 0$  for  $M$ , where

$$F_1(x, M_1, M_2) = E_{21} \circ F_{21} \circ A_1(M_1) - M_2,$$

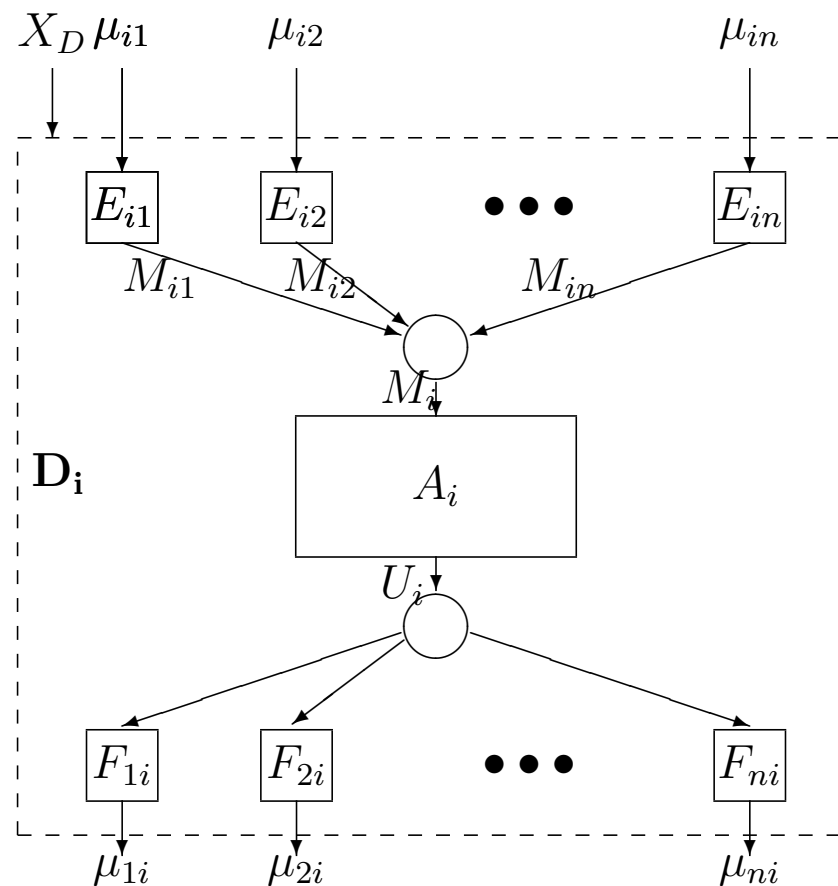
$$F_2(x, M_1, M_2) = E_{12} \circ F_{12} \circ A_2(M_2) - M_1$$

- ◆  $A_1$  finite difference and  $A_2$  finite element maybe
- ◆  $E, F$  are interface routines that do such bookkeeping as converting pressures to loads and deflections to shape

# A generic discipline in an MDO problem

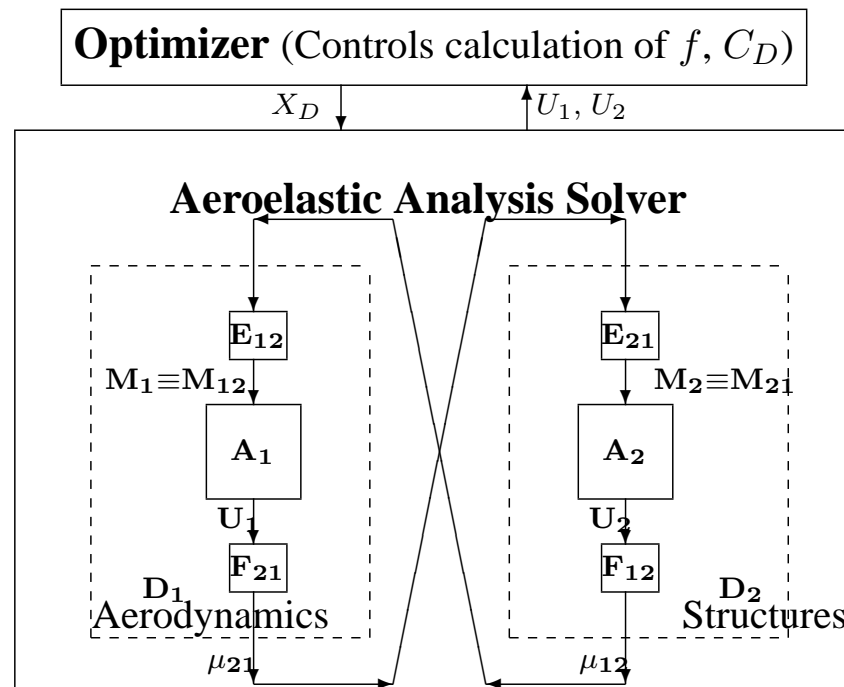


# A generic discipline in an MDO problem

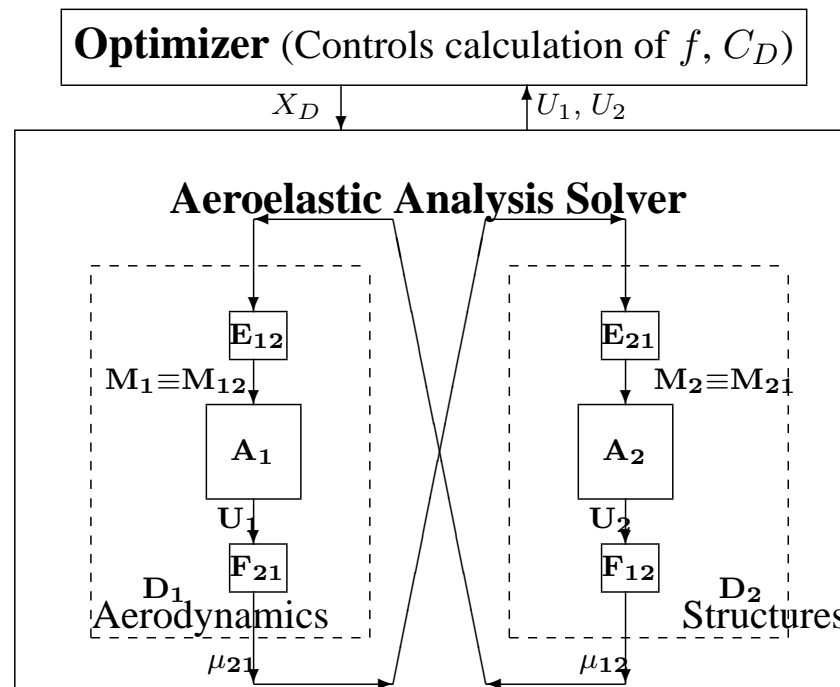


$E$ ,  $F$  routines are a major headache to write

# The MDF or blackbox formulation

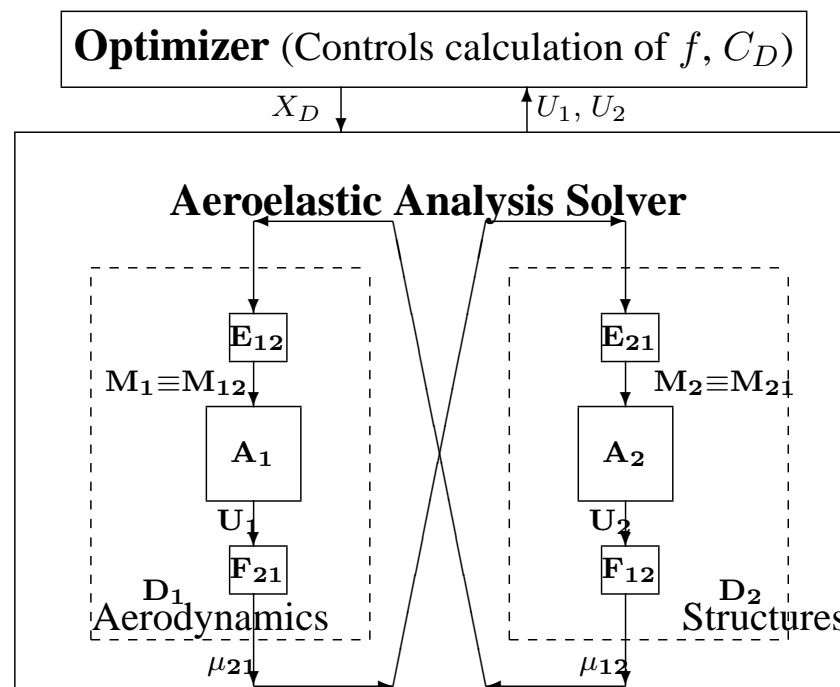


# The MDF or blackbox formulation



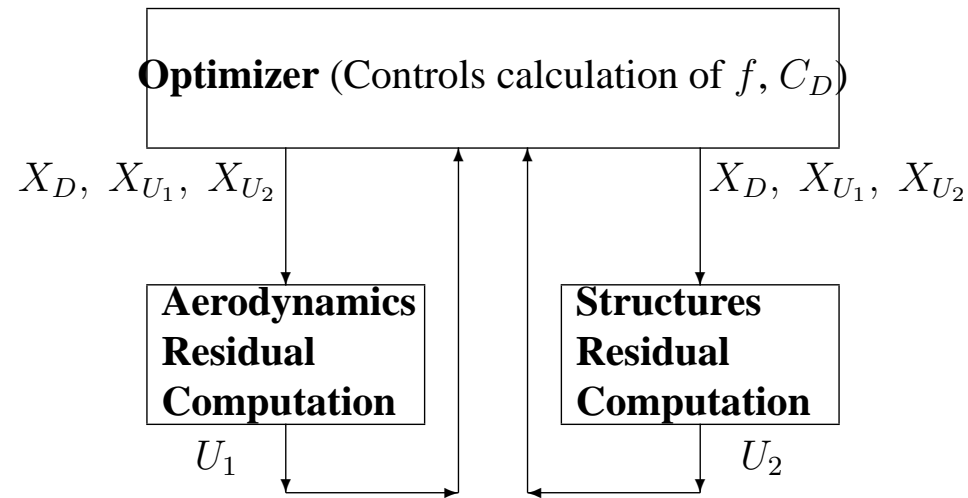
- ◆ Requires a full MDA for every evaluation

# The MDF or blackbox formulation

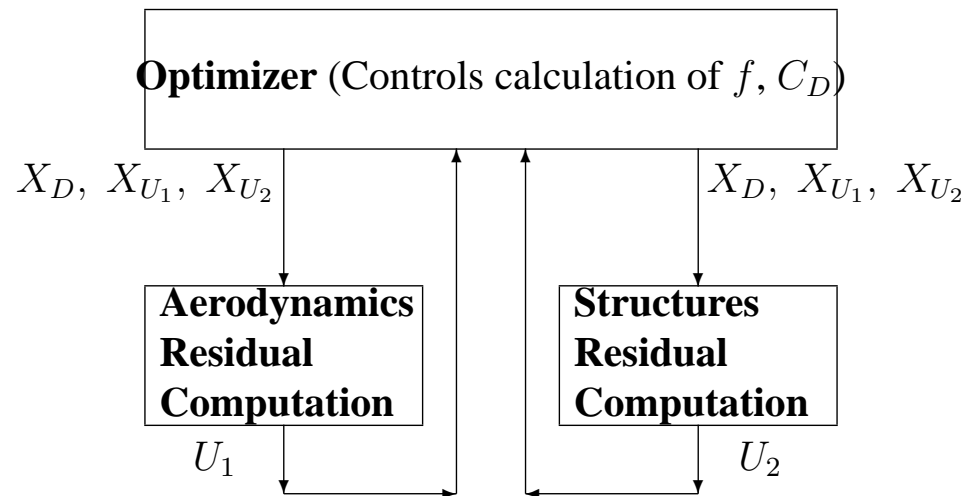


- ◆ Requires a full MDA for every evaluation
- ◆ Derivatives must be through solvers - requires  $\frac{du}{dx}$

# The All-At-Once or SAND formulation

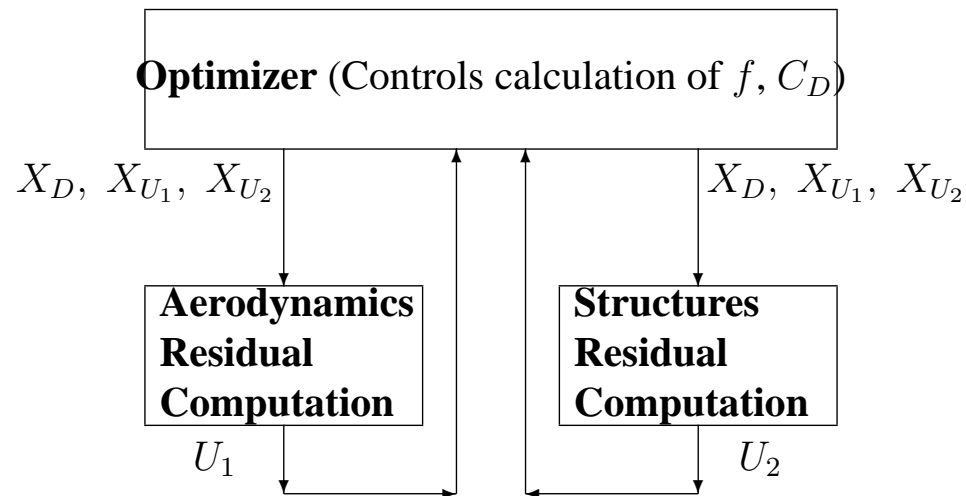


# The All-At-Once or SAND formulation



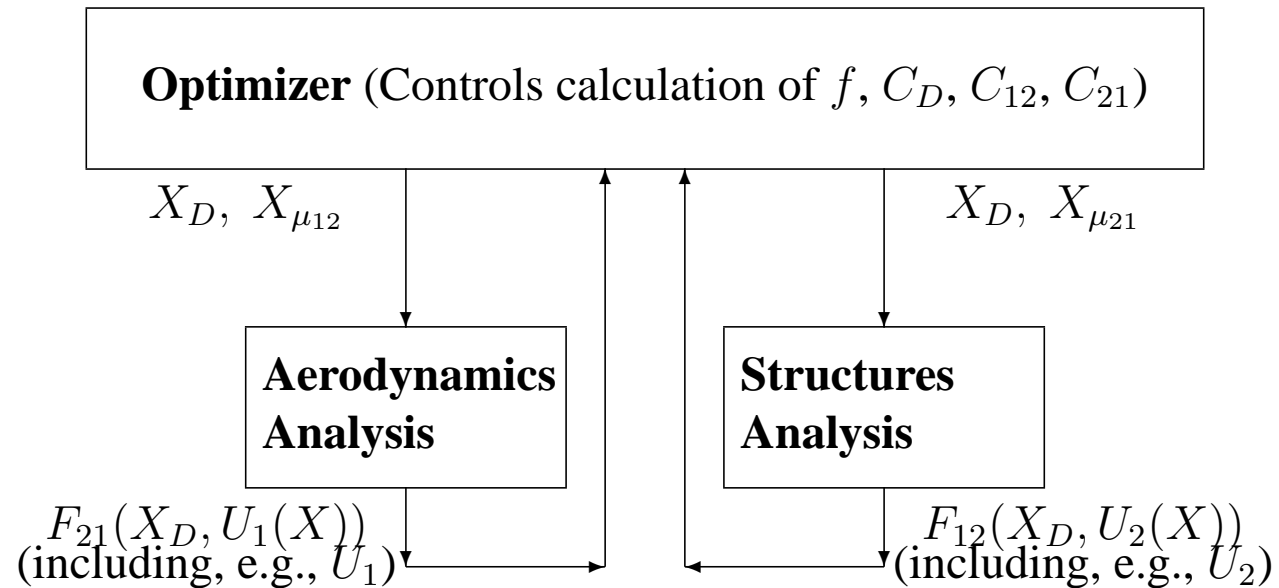
- ◆  $(x, u) = (x_d, x_u)$  all are optimization variables, and  $F_p(x_d, x_u) = 0$  is enforced only in the limit

# The All-At-Once or SAND formulation

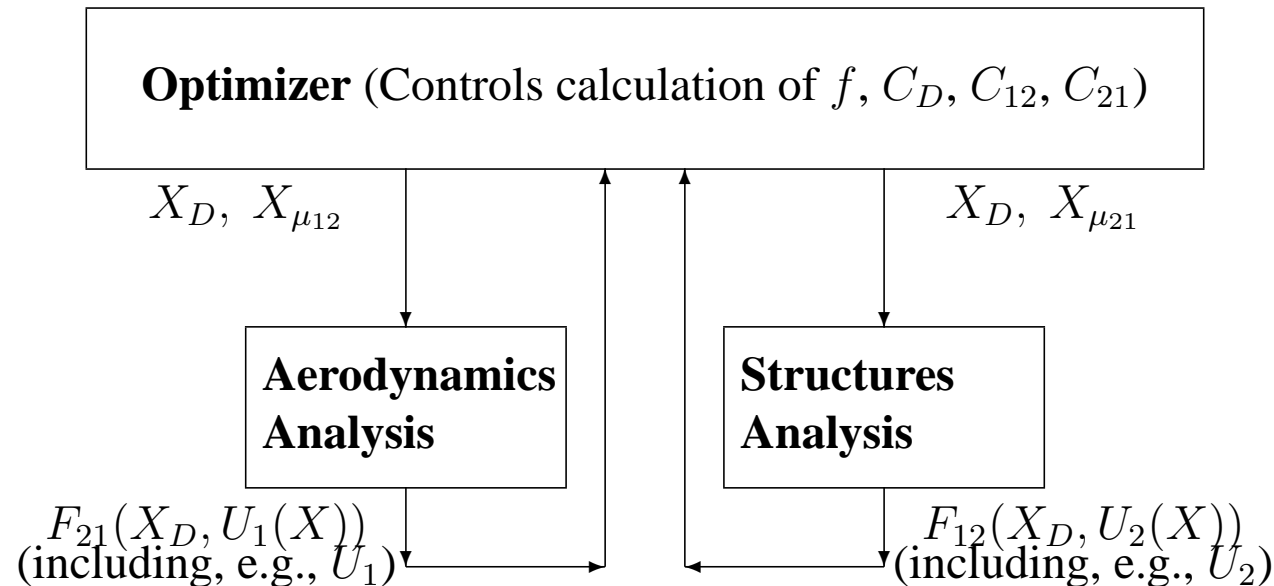


- ◆  $(x, u) = (x_d, x_u)$  all are optimization variables, and  $F_p(x_d, x_u) = 0$  is enforced only in the limit
- ◆ Can't use legacy solvers, but derivatives are simpler -  $\frac{du}{dx}$  avoided

# The IDF or DAO formulation

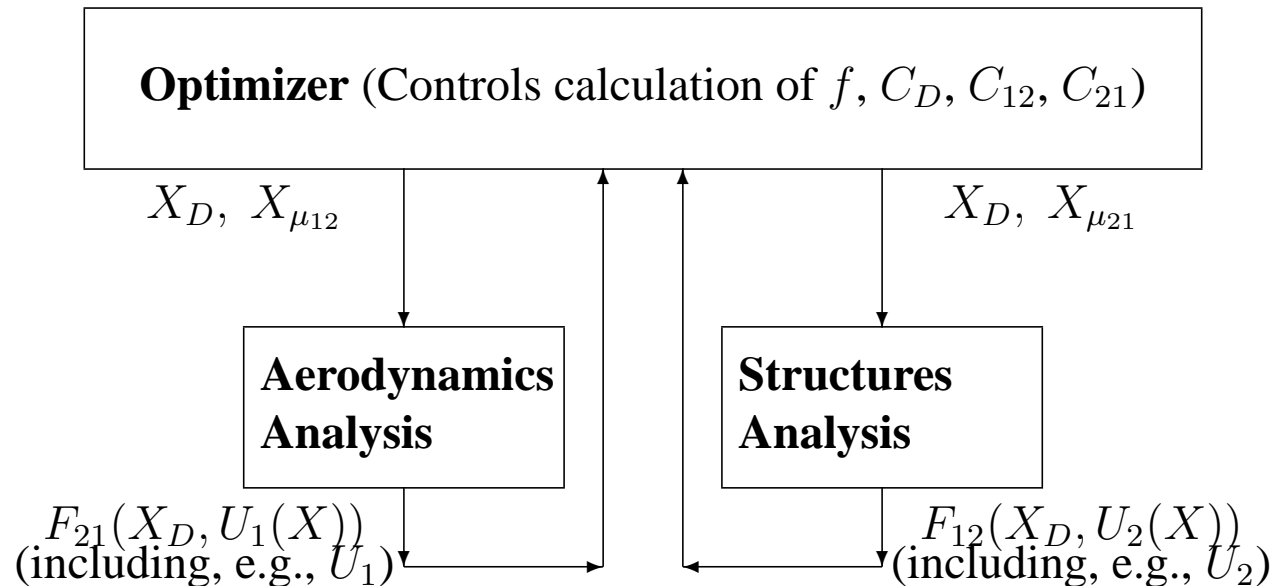


# The IDF or DAO formulation



- ◆ MDA only in the limit - many fewer variables than AAO

# The IDF or DAO formulation



- ◆ MDA only in the limit - many fewer variables than AAO
- ◆ Uses legacy solvers, but requires  $\frac{du}{dx}$

# Summary

- ◆ Blackbox/MDF is least efficient, but most stable, and most often used in practice.

# Summary

- ◆ Blackbox/MDF is least efficient, but most stable, and most often used in practice. Derivatives can sometimes be gotten by adjoint approaches, but not if there are many constraints

# Summary

- ◆ Blackbox/MDF is least efficient, but most stable, and most often used in practice. Derivatives can sometimes be gotten by adjoint approaches, but not if there are many constraints
- ◆ AAO is most efficient when it works, but incompatibility with legacy codes is a huge barrier to practical use in most fields

# Summary

- ◆ Blackbox/MDF is least efficient, but most stable, and most often used in practice. Derivatives can sometimes be gotten by adjoint approaches, but not if there are many constraints
- ◆ AAO is most efficient when it works, but incompatibility with legacy codes is a huge barrier to practical use in most fields Very common in chemical process control

# Summary

- ◆ Blackbox/MDF is least efficient, but most stable, and most often used in practice. Derivatives can sometimes be gotten by adjoint approaches, but not if there are many constraints
- ◆ AAO is most efficient when it works, but incompatibility with legacy codes is a huge barrier to practical use in most fields Very common in chemical process control
- ◆ IDF/DAO seems a good compromise - generalizes domain decomposition

# Summary

- ◆ Blackbox/MDF is least efficient, but most stable, and most often used in practice. Derivatives can sometimes be gotten by adjoint approaches, but not if there are many constraints
- ◆ AAO is most efficient when it works, but incompatibility with legacy codes is a huge barrier to practical use in most fields Very common in chemical process control
- ◆ IDF/DAO seems a good compromise - generalizes domain decomposition

# Summary

- ◆ Many designers mistrust a method like AAO or IDF that doesn't have all feasible iterates

# Summary

- ◆ Many designers mistrust a method like AAO or IDF that doesn't have all feasible iterates
- ◆ Blackbox or MDF will be used in Lec4 numerical results for the SMF, which is derivative-free

# Summary

- ◆ Many designers mistrust a method like AAO or IDF that doesn't have all feasible iterates
- ◆ Blackbox or MDF will be used in Lec4 numerical results for the SMF, which is derivative-free
- ◆ A heuristic called “collaborative optimization” mimics the “by hand” traditional approach

# Summary

- ◆ Many designers mistrust a method like AAO or IDF that doesn't have all feasible iterates
- ◆ Blackbox or MDF will be used in Lec4 numerical results for the SMF, which is derivative-free
- ◆ A heuristic called “collaborative optimization” mimics the “by hand” traditional approach
- ◆ DAKOTA project at SANDIA attacks same problem class by more heuristic approaches

## What next

- ◆ In Lec2, Charles will show you the filtered pattern search (FPS) method for handling general nonlinear programming without derivatives

## What next

- ◆ In Lec2, Charles will show you the filtered pattern search (FPS) method for handling general nonlinear programming without derivatives
- ◆ In Lec3, Mark will show you how to extend the FPS algorithm to handle categorical variables and give an interesting application

## What next

- ◆ In Lec2, Charles will show you the filtered pattern search (FPS) method for handling general nonlinear programming without derivatives
- ◆ In Lec3, Mark will show you how to extend the FPS algorithm to handle categorical variables and give an interesting application
- ◆ In Lec4, I will come back and show how to use surrogates for the expensive evaluations with FPS and give several numerical tests

## What next

- ◆ In Lec2, Charles will show you the filtered pattern search (FPS) method for handling general nonlinear programming without derivatives
- ◆ In Lec3, Mark will show you how to extend the FPS algorithm to handle categorical variables and give an interesting application
- ◆ In Lec4, I will come back and show how to use surrogates for the expensive evaluations with FPS and give several numerical tests

- ◆ Wednesday's IMA tutorial will lay the groundwork for the Workshop on the AAO or SAND approach

# Stairway to the Stars