1 Introduction

Multidisciplinary design optimization (MDO): a field of engineering that uses numerical optimization to perform the design of systems that involve a number of disciplines or subsystems.

- the best design of a multidisciplinary system can only be found when the interactions between the system's disciplines are fully considered.
- Considering these interactions in the design process cannot be done in an arbitrary way
- requires a sound mathematical formulation.

By solving the MDO problem early in the design process and taking advantage of advanced computational analysis tools, *designers can simultaneously improve the design and reduce the time and cost of the design cycle.*

The origins of MDO can be traced back to Schmit [131, 132, 133] and Haftka [55, 56, 58], who extended their experience in structural optimization to include other disciplines.

One of the first applications of MDO was aircraft **wing design**, where aerodynamics, structures, and controls are three *strongly coupled* disciplines [51, 52, 100, 99].

Since then, the application of MDO has been extended:

- complete aircraft [88, 102], rotorcraft [45, 49], and spacecraft [23, 27].
- bridges [9] and buildings [32, 47];
- railway cars [64, 42], automobiles [109, 85], and ships [123, 70];
- and even microscopes [124].

Important considerations when implementing MDO:

- How to organize the disciplinary analysis models?
- What optimization software/methods do we choose?
- Do we use approximation (surrogate) models?

MDO architecture: Combination of problem formulation and organizational strategy. The MDO architecture defines both how the different models are coupled and how the overall optimization problem is solved.

There are several terms in literature used to describe what we mean by "architecture" [22, 87, 138, 30]:

- "method" [88, 144, 158, 128, 1],
- "methodology" [77, 114, 112],
- "problem formulation" [35, 4, 5],
- "strategy" [161, 59],
- "procedure" [145, 81] and
- "algorithm" [135, 143, 137, 41]

Our preference is for the term “architecture”, because the relationship between problem formulation and solution algorithm is not one-to-one. For example, replacing a particular disciplinary simulation with a surrogate model or reordering the disciplinary simulations do not affect the problem formulation but strongly affect the solution algorithm.

Choosing the most appropriate architecture for the problem can significantly reduce the solution time. These time savings come from

- the methods chosen to solve each discipline;
- the optimization algorithm driving the process;
- the coupling scheme used in the architecture, and;
- the degree to which operations are carried out in parallel.

The latter consideration becomes especially important as the design becomes more detailed and the number of variables and/or constraints increases.

The purpose of this chapter:

- survey the available MDO architectures
- present architectures in a unified notation to facilitate understanding and comparison
- introduce the Extended Design Structure Matrix (XDSM), a diagram to visualize the algorithm of a given MDO architecture

2 Unified Description of MDO Architectures

2.1 Terminology and Mathematical Notation

design variable: a variable in the MDO problem that is always under the explicit control of an optimizer.

local design variable: design variables relevant to a single discipline only
$\rightarrow$ denoted by x_i for discipline i

shared design variable: design variables used by several disciplines
$\rightarrow$ denoted by x_0

Full set of design variables:

$$x = \left[x_0^T, x_1^T, \ldots, x_N^T\right]^T$$

Think of some examples of local and shared design variables in the context of aerostructural optimization

discipline analysis: a simulation that models the behavior of one aspect of a multidisciplinary system
$\rightarrow$ represented by equations $\mathcal{R}_i = 0$,

state variables: set of variables determined by solving a discipline analysis.
$\rightarrow$ denoted by $\bar{y}_i$

Give some examples of aerodynamic discipline analyses. What are the corresponding state variables?

coupling variables: variables determined by one discipline and that influence another discipline

response variables: coupling variables from a specific discipline
$\rightarrow$ denoted by y_i for discipline i
target variables: values of the response variables that we need to match
$\rightarrow$ denoted by y_i^t for discipline i
consistency constraints: constraints that ensure the response variables match the values of the target variables
$\rightarrow$ for discipline i we have $c_i^c = y_i^t - y_i$

Choose a multidisciplinary problem. Identify the coupling variables.

Table 1: Mathematical notation for MDO problem formulations

Symbol	Definition
x	Vector of design variables
y^t	Vector of coupling variable targets (inputs to a discipline analysis)
y	Vector of coupling variable responses (outputs from a discipline analysis)
$\bar{y}$	Vector of state variables (variables used inside only one discipline analysis)
f	Objective function
c	Vector of design constraints
c^c	Vector of consistency constraints
$\mathcal{R}$	Governing equations of a discipline analysis in residual form
N	Number of disciplines
$n_{()}$	Length of given variable vector
$m_{()}$	Length of given constraint vector
$()_0$	Functions or variables that are shared by more than one discipline
$()_i$	Functions or variables that apply only to discipline i
$()^*$	Functions or variables at their optimal value
$\tilde{()}$	Approximation of a given function or vector of functions

2.2 Architecture Diagrams — The Extended Design Structure Matrix

An Extended Design Structure Matrix, or XDSM [89], is a convenient and compact way to describe the sequence of operations in an MDO architecture. The XDSM was developed to simultaneously communicate data dependency and process flow between computational components of the architecture on a single diagram

The XDSM is based on Design Structure Matrix [146, 25] and follows its basic rules:

- architecture components are placed on main diagonal of the “matrix”
- inputs to a component are placed in the same column
- outputs to a component are placed in the same row
- External inputs and outputs may also be defined and are placed on the outer edges of the diagram

- Thick gray lines are used to show the data flow between components

- A numbering system is used to show the order in which the components are executed (The algorithm starts at component zero and proceeds in numerical order)

- consecutive components in the algorithm are connected by a thin black line

- Loops are denoted using the notation $j \to k$ for $k < j$ so that the algorithm must return to step k until a looping condition is satisfied before proceeding

Example 1: Gauss–Seidel MDA

Algorithm 1 Block Gauss–Seidel multidisciplinary analysis algorithm

```
Input: Design variables $x$
Output: Coupling variables, $y$
  0: Initiate MDA iteration loop
  repeat
    1: Evaluate Analysis 1 and update $y_1$
    2: Evaluate Analysis 2 and update $y_2$
    3: Evaluate Analysis 3 and update $y_3$
  until $4 \to 1$: MDA has converged
```

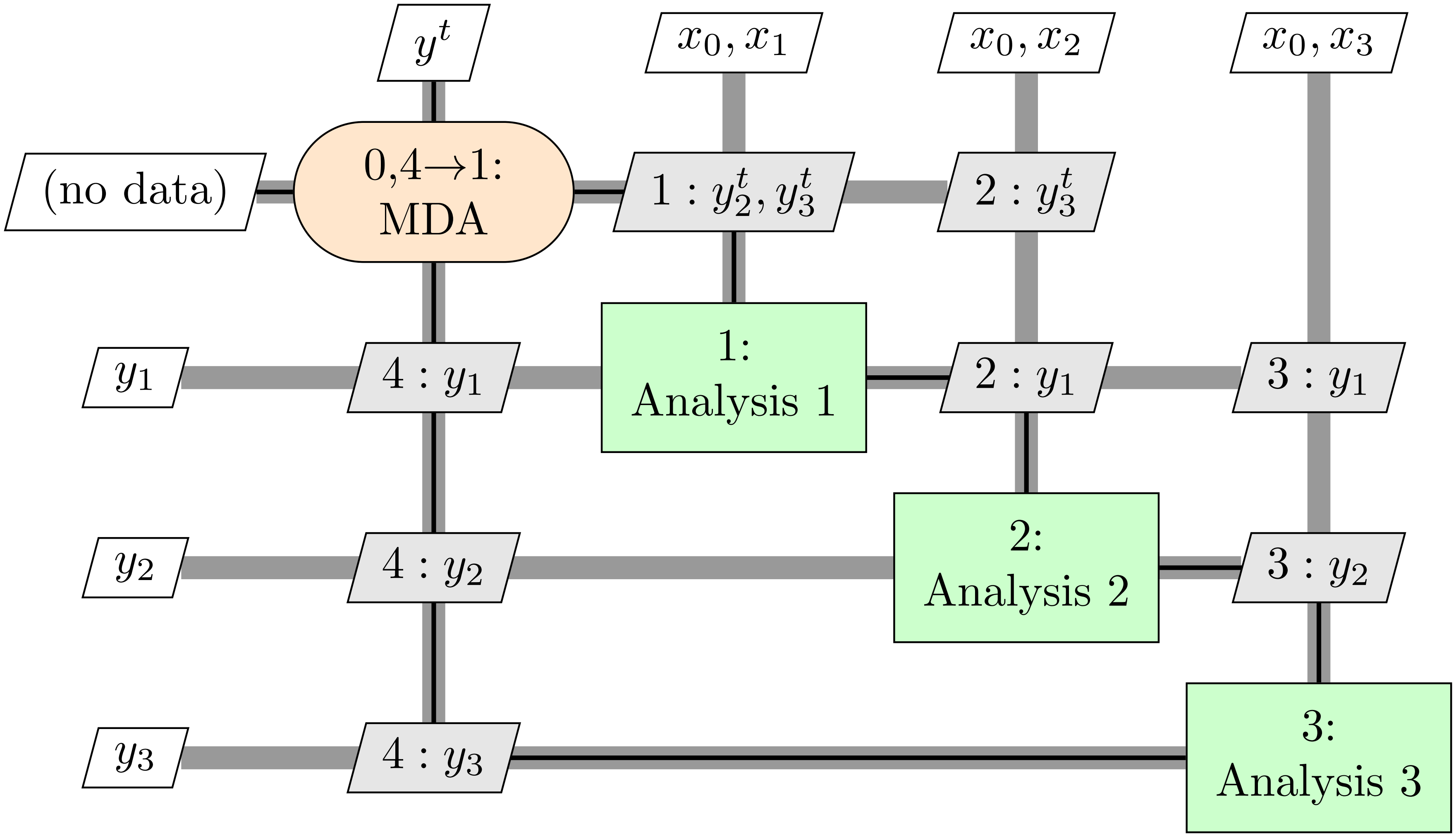


Figure 1: A block Gauss–Seidel multidisciplinary analysis (MDA) process to solve a three-discipline coupled system.

Example 2: Gradient-based Optimization

- objective, constraints, and gradient evaluations are the (3) individual components

Example 2: Gradient-based Optimization

- objective, constraints, and gradient evaluations are the (3) individual components

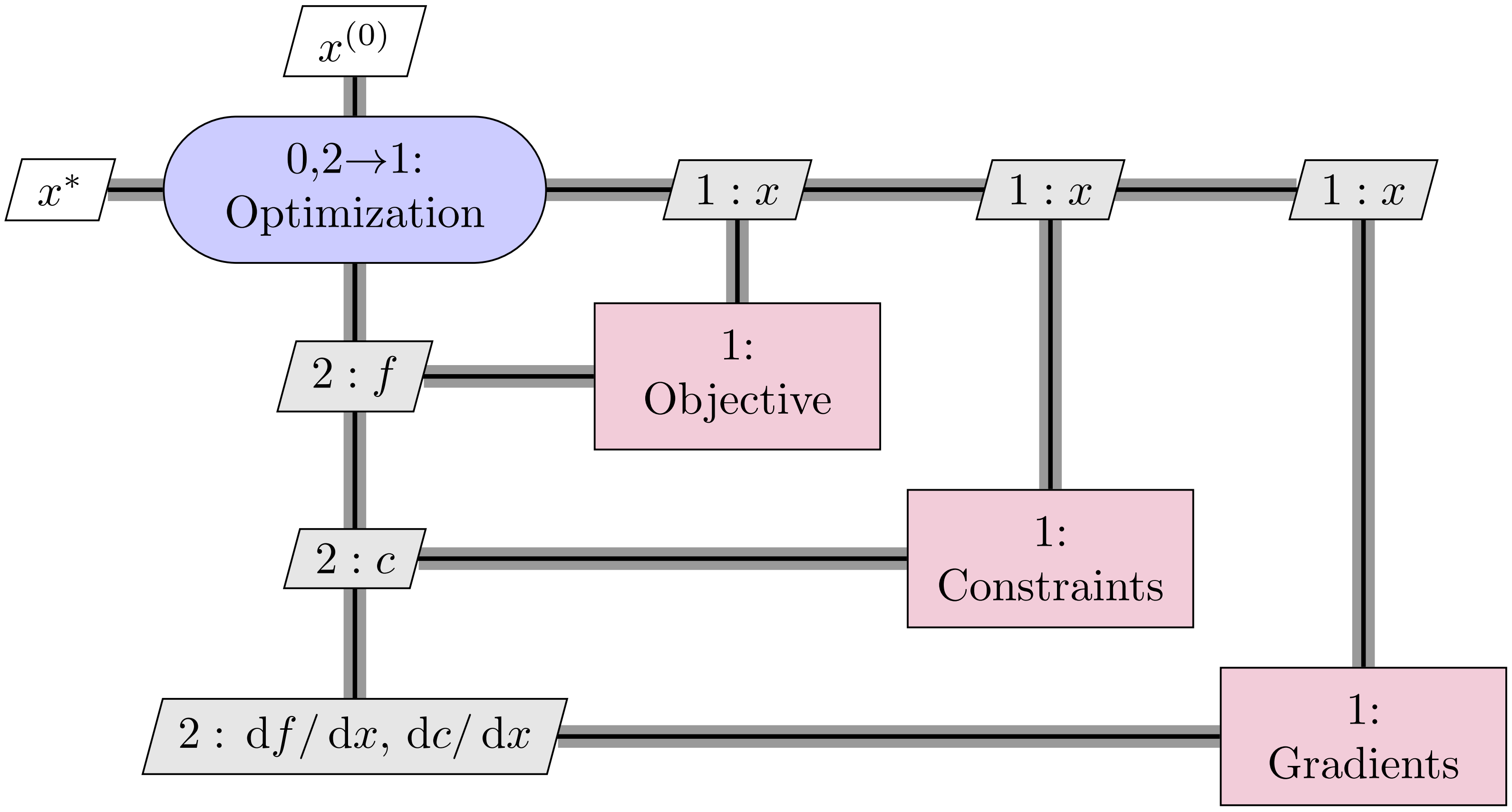


Figure 2: A gradient-based optimization procedure.

2.3 The All-At-Once (AAO) Problem Statement

Consider the general "all-at-once" (AAO) MDO problem statement:

$$\begin{array}{rll} \text{minimize} & f_0(x,y) + \sum_{i=1}^{N} f_i(x_0,x_i,y_i) & \\ \text{with respect to} & x, y^t, y, \bar{y} & \\ \text{subject to} & c_0(x,y) \geq 0 & \\ & c_i(x_0,x_i,y_i) \geq 0 & \text{for } i = 1,\dots,N \\ & c_i^c = y_i^t - y_i = 0 & \text{for } i = 1,\dots,N \\ & \mathcal{R}_i\left(x_0,x_i,y_{j\neq i}^t,\bar{y}_i,y_i\right) = 0 & \text{for } i = 1,\dots,N. \end{array}$$

- Typically omit the local objective functions f_i unless necessary
- Some authors refer to AAO as simultaneous analysis and design
- AAO is rarely solved in practice: we can eliminate the c_i^c.

“All-at-once” (AAO) problem

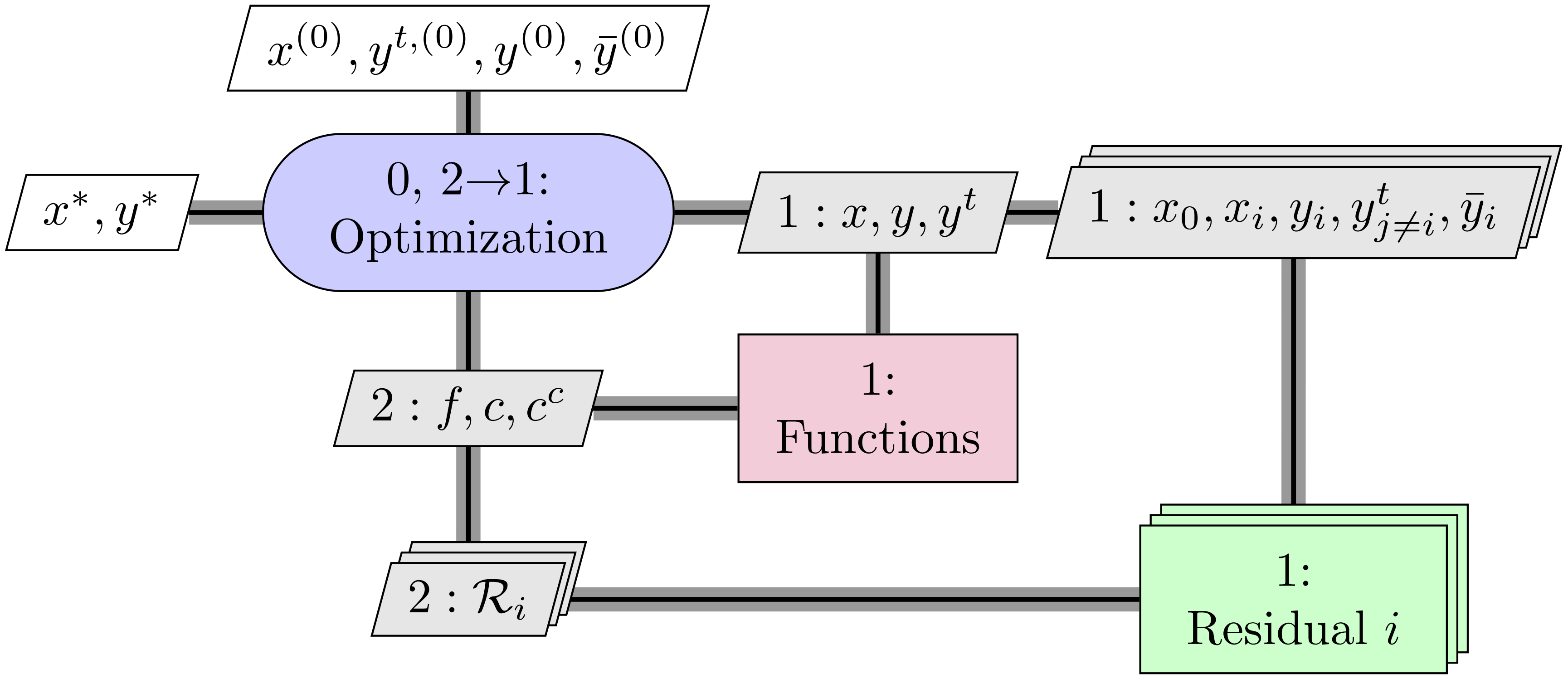


Figure 3: XDSM for solving the AAO problem.

3 Monolithic Architectures

AAO provides a unifying starting point for other architectures; depending on which constraints are eliminated, we can derive different monolithic architectures.

Monolithic Architectures: an architecture that the solves the MDO problem as a single optimization problem.

We will consider three monolithic architectures:

1. Simultaneous Analysis and Design (SAND);
2. Individual Discipline Feasible (IDF), and;
3. Multidisciplinary Feasible (MDF).

The monolithic architectures are distinguished by how they achieve *multidisciplilnary feasibility*.

3.1 Simultaneous Analysis and Design (SAND)

Idea: use a single copy of the coupling variables that is shared between disciplines to eliminate the consistency constraints. This simplification yields the SAND architecture [57]:

$$
\begin{aligned}
\text{minimize} \quad & f_0\left(x, y\right) \\
\text{with respect to} \quad & x, y, \bar{y} \\
\text{subject to} \quad & c_0\left(x, y\right) \geq 0 \\
& c_i\left(x_0, x_i, y_i\right) \geq 0 \qquad \text{for } i = 1, \ldots, N \\
& \mathcal{R}_i\left(x_0, x_i, y, \bar{y}_i\right) = 0 \ \text{for } i = 1, \ldots, N.
\end{aligned}
$$

Simultaneous Analysis and Design

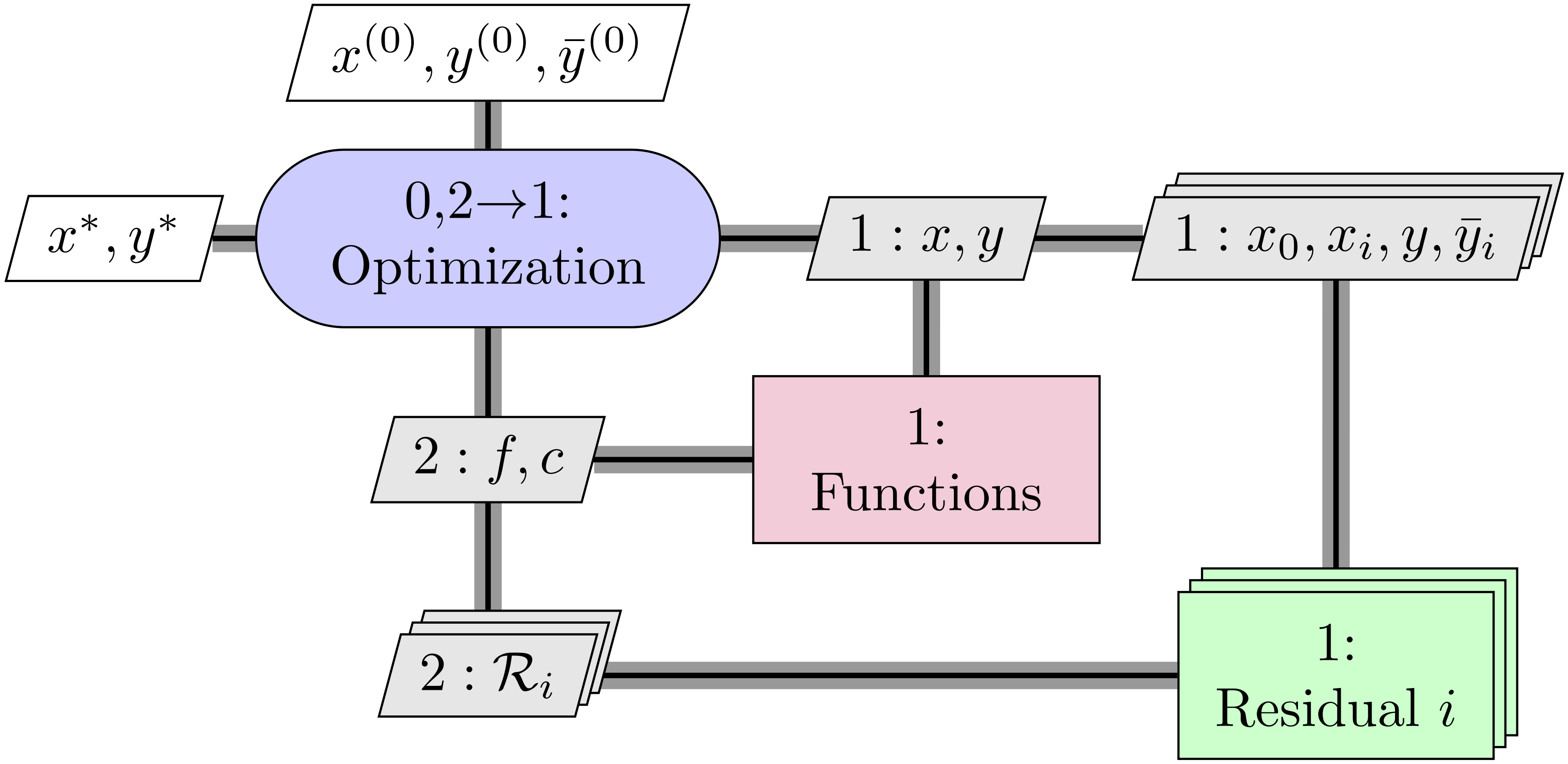


Figure 4: Diagram for the SAND architecture.

Several features of the SAND architecture are noteworthy:

- The optimizer is responsible for simultaneously analyzing and designing the system

- At each iteration, the analyses do not need to be solved exactly: $\mathcal{R}_i \neq 0$
  $\rightarrow$ inexact solution to analyses: potential to solve optimization quickly
  $\rightarrow$ 5-10 times a single MDA

- Can also be used to solve single discipline optimization problem

- Full-space PDE-constrained optimization is an example

Question: Can you think of some disadvantages of SAND?

Disadvantages of the SAND approach:

1. Large problem size: all state variables must be available to the optimizer
   $\rightarrow$ existing optimization software cannot handle typical CFD problem

2. Globalization is an issue: converging problem when initial design/state is far from final solution
   $\rightarrow$ globalizing analysis codes is a formidable challenge on its own

3. Residual values and (likely) their derivatives must be available to optimization software
   $\rightarrow$ Many discipline analysis codes operate like "black-boxes"

4. Quasi-Newton methods may be slow, because of large problem size
   $\rightarrow$ need to consider some form of Newton's method; possible but complicated

3.2 Individual Discipline Feasible (IDF)

Idea: Solve the discipline analyses exactly for given x and y^t

- eliminates the disciplinary analysis constraints $\mathcal{R}_i(x_0, x_i, y_i, y^t_{j \neq i}, \bar{y}_i) = 0$ from the optimization problem

- invokes the implicit function theorem on the $\mathcal{R}_i$, hence

$$\begin{aligned} \bar{y}_i &= \bar{y}_i(x, y^t) \\ y_i &= y_i(x, y^t) \end{aligned}$$

- IDF also called distributed analysis optimization [4] and optimizer-based decomposition [87]

The problem statement for the IDF architecture is

$$\begin{aligned}
\text{minimize} \quad & f_0\left(x, y\left(x, y^t\right)\right) \\
\text{with respect to} \quad & x, y^t \\
\text{subject to} \quad & c_0\left(x, y\left(x, y^t\right)\right) \geq 0 \\
& c_i\left(x_0, x_i, y_i\left(x_0, x_i, y^t_{j\neq i}\right)\right) \geq 0 \;\; \text{for } i = 1, \ldots, N \\
& c^c_i = y^t_i - y_i\left(x_0, x_i, y^t_{j\neq i}\right) = 0 \quad \text{for } i = 1, \ldots, N.
\end{aligned}$$

Individual Discipline Feasible

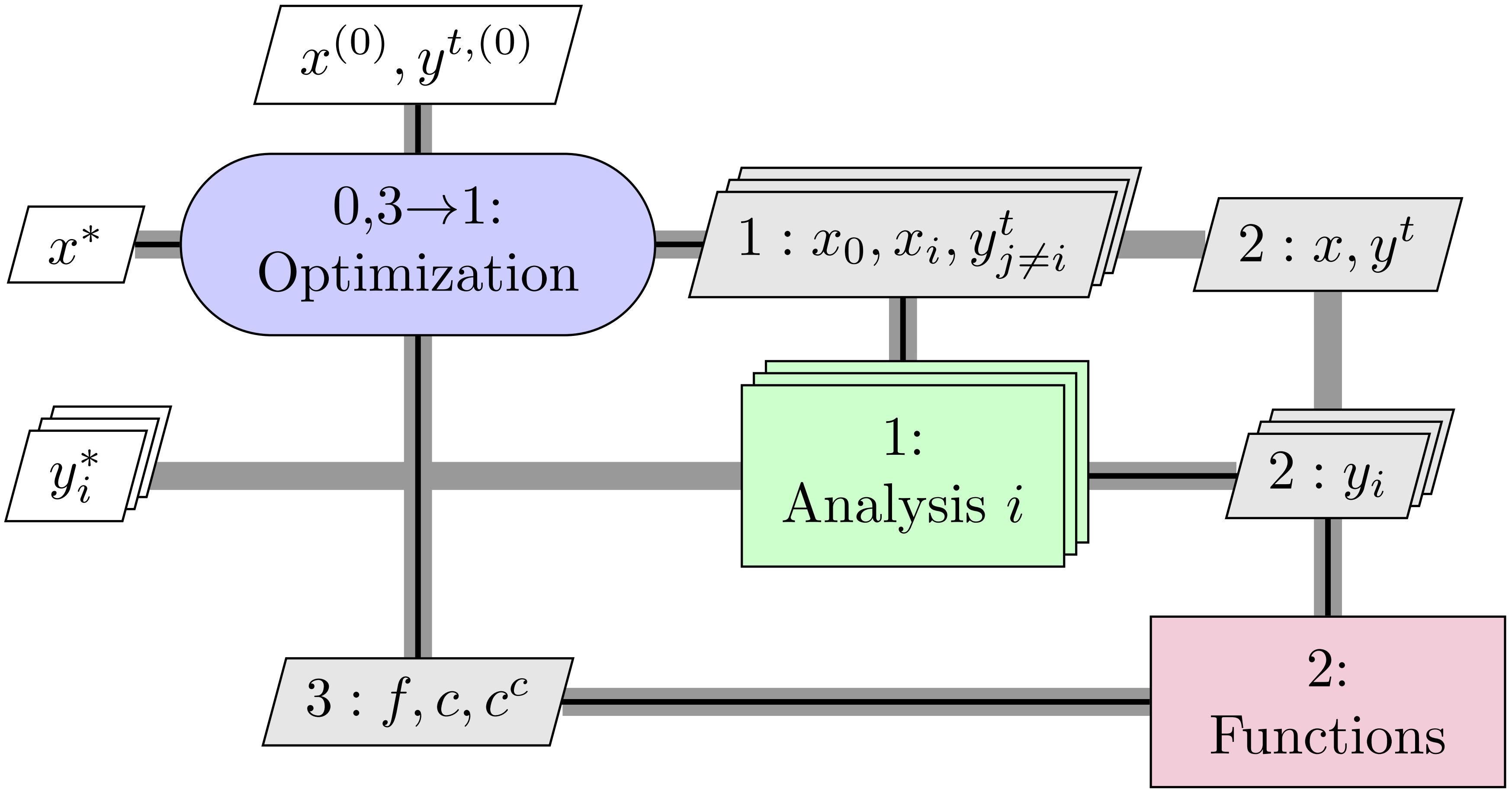


Figure 5: Diagram of the IDF architecture.

Some advantages of IDF include the following:

- all state variables and discipline analysis equations are removed from optimization
- discipline analyses can be performed in parallel
- analysis codes are already highly specialized and efficient at solving their respective equations
  $\rightarrow$ for example, globalization is simplified

The increased flexibility of IDF comes at a price. List some of the drawbacks of this architecture?

Disadvantages of IDF:

1. if optimization terminates before satisfying the KKT conditions (for example), the intermediate solution is not multidisciplinary feasible
   $\rightarrow$ in some sense, this is no better than SAND

2. number of coupling variables may still be large
   $\rightarrow$ typically 10^3 – 10^5 for aerostructural problems

3. gradient computation, if necessary, is expensive
   $\rightarrow$ Adjoint-based gradients are critical for efficiency, but requires intrusion into source code

3.3 Multidisciplinary Feasible (MDF)

Idea: Solve the multidisciplinary analysis exactly for given x.

- eliminate both the disciplinary analyses and consistency constraints from optimization problem
- also called Fully Integrated Optimization [4] and Nested Analysis and Design [10]

MDF problem statement:

$$\begin{aligned} \text{minimize} \quad & f_0\left(x, y\left(x, y\right)\right) \\ \text{with respect to} \quad & x \\ \text{subject to} \quad & c_0\left(x, y\left(x, y\right)\right) \geq 0 \\ & c_i\left(x_0, x_i, y_i\left(x_0, x_i, y_{j \neq i}\right)\right) \geq 0 \;\; \text{for}\, i = 1, \ldots, N. \end{aligned}$$

Multidisciplinary Feasible

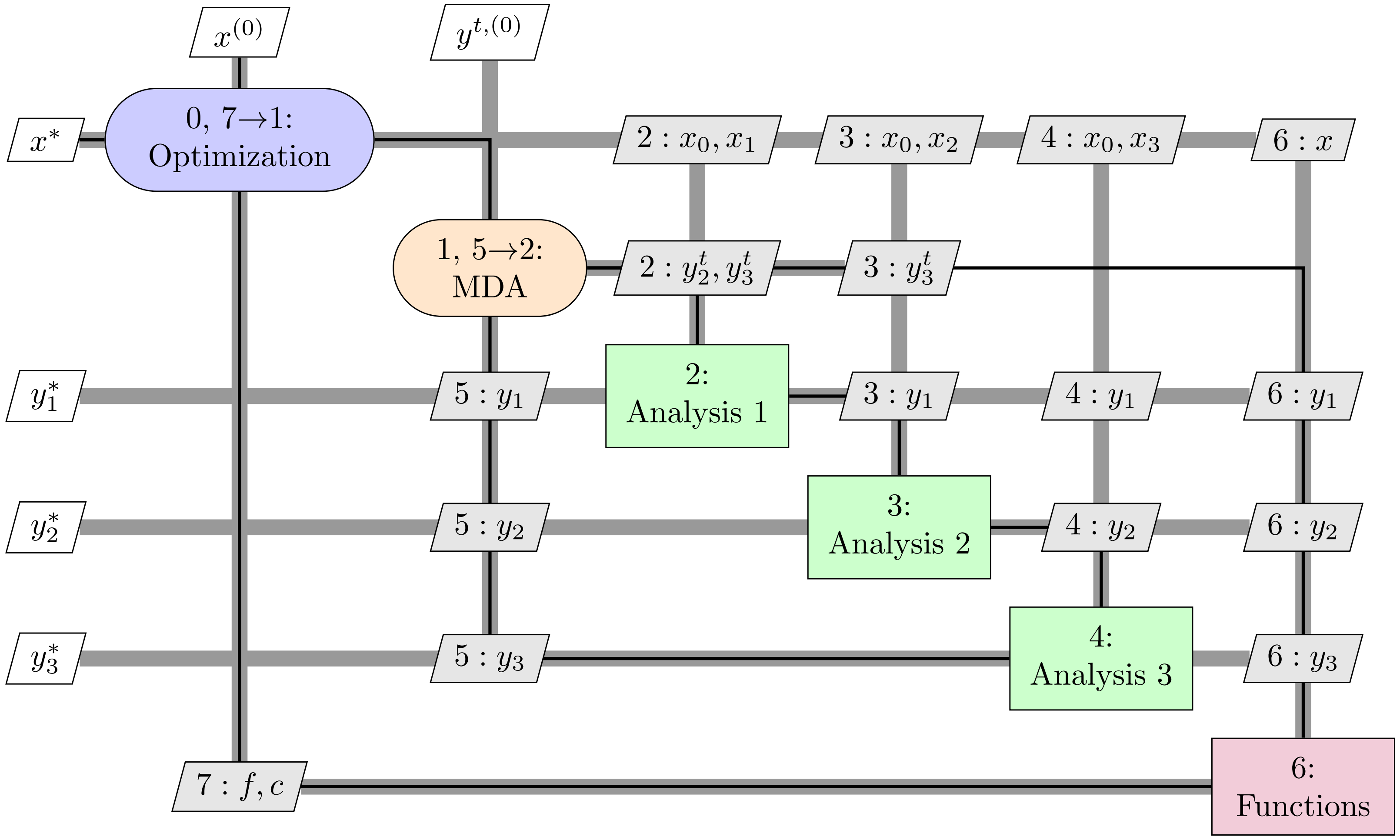


Figure 6: Diagram for the MDF architecture with a Gauss–Seidel multidisciplinary analysis.

Advantageous features of MDF:

- optimization algorithm is responsible for the design variables, objective, and design constraints only
  $\rightarrow$ optimization problem is as small as possible
- design is always feasible in a multidisciplinary sense

The MDA can be solved in various ways, but efficiency of optimization is tied to this choice:

- simple, popular choice is a fixed-point iteration like Gauss-Seidel
- Newton-Krylov approach is much more efficient, but requires intrusion

What about disadvantages of MDF?

Drawbacks of using MDF:

1. architecture requires full MDA for each (optimization) iteration
   MDA is, on its own, a challenging task
   $\rightarrow$ may not be an issue if only one discipline dominates CPU time

2. gradient computation is complex
   $\rightarrow$ for efficiency, need a coupled adjoint for whole MDA

3. For premature iteration state variables are feasible, but design may not be
   $\rightarrow$ dependent on choice of optimization algorithm

4 Distributed Architectures

Consider the following problem:

$$\begin{aligned}
\text{minimize} \quad & \sum_{i=1}^{N} f_i(x_i) \\
\text{with respect to} \quad & x_1, \ldots, x_N \\
\text{subject to} \quad & c_0(x_1, \ldots, x_N) \le 0 \\
& c_1(x_1) \le 0, \ldots, c_N(x_N) \le 0.
\end{aligned}$$

- there are no shared design variables, x_0
- the objective function is separable: it can be expressed as a sum of functions, each of which depends only on corresponding local design variables
- constraints depend on more than one set of design variables

Question: if c_0 did not exist, how could we solve this problem?

The previous problem is referred to as a *complicating constraints problem* [33].

Another possibility:

$$\begin{aligned} \text{minimize} \quad & \sum_{i=1}^{N} f_i(x_0, x_i) \\ \text{with respect to} \quad & x_0, x_1, \ldots, x_N \\ \text{subject to} \quad & c_1(x_0, x_1) \le 0, \ldots, c_N(x_0, x_N) \le 0. \end{aligned}$$

This is referred to as a problem with *complicating variables* [33].

- Decomposition would be straightforward if there were no shared design variables, x_0, and we could solve N optimization problems independently and in parallel.

Distributed Architectures: an architecture that the solves the MDO problem using a set of optimization problems or subproblems.

The primary motivation for decomposing the MDO problem comes from the structure of the engineering design environment.

- typical industrial practice involves breaking up the design of a large system and distributing aspects of that design to specific engineering groups.
- These groups may be geographically distributed and may only communicate infrequently.
- These groups typically like to retain control of their own design procedures and make use of in-house expertise; they object to a central design authority [87].

Decomposition through distributed architectures allow individual design groups to work in isolation, controlling their own sets of design variables, while periodically updating information from other groups to improve their aspect of the overall design. This approach to solving the problem conforms more closely with current industrial design practice than the approach of the monolithic architectures.

4.1 Classification

Previous classifications of MDO architectures:

- based on observations of which constraints were available to the optimizer to control [10, 3].
- Alexandrov and Lewis [3] used the term “closed” to denote when a set of constraints cannot be satisfied by explicit action of the optimizer, and “open” otherwise.
  For example, the MDF architecture is closed with respect to both analysis and consistency constraints, because their satisfaction is determined through the process of converging the MDA. Similarly, IDF is closed analysis but open consistency since the consistency constraints can be satisfied by the optimizer adjusting the coupling targets and design variables.
- Tosserams et al. [154] expanded on this classification scheme by discussing whether or not distributed architectures used open or closed local design constraints in the system subproblem.

Closure of the constraints is an important consideration when selecting an architecture because most robust optimization software will permit the exploration of infeasible regions of the design space. Such exploration may result in faster solutions via fewer optimizer iterations, but this must be weighed against the increased optimization problem size and the risk of terminating the optimization at an infeasible point.

New classification for distributed MDO architectures: a distributed MDO architecture can be classified based on their monolithic analogues: either MDF, IDF, or SAND.

This stems from the different approaches to handling the state and coupling variables in the monolithic architectures.

- similar to the previous classifications in that an equality constraint must be removed from the optimization problem — i.e., closed — for every variable removed from the problem statement.

- However, using a classification based on the monolithic architectures makes it much easier to see the connections between distributed architectures, even when these architectures are developed in isolation from each other.

In many cases, the problem formulations in the distributed architecture can be derived directly from that of the monolithic architecture by adding certain elements to the problem, by making certain assumptions, and by applying a specific decomposition scheme. This classification can also be viewed as a framework in which we can develop new distributed architectures, since the starting point for a distributed architecture is *always* a monolithic architecture.

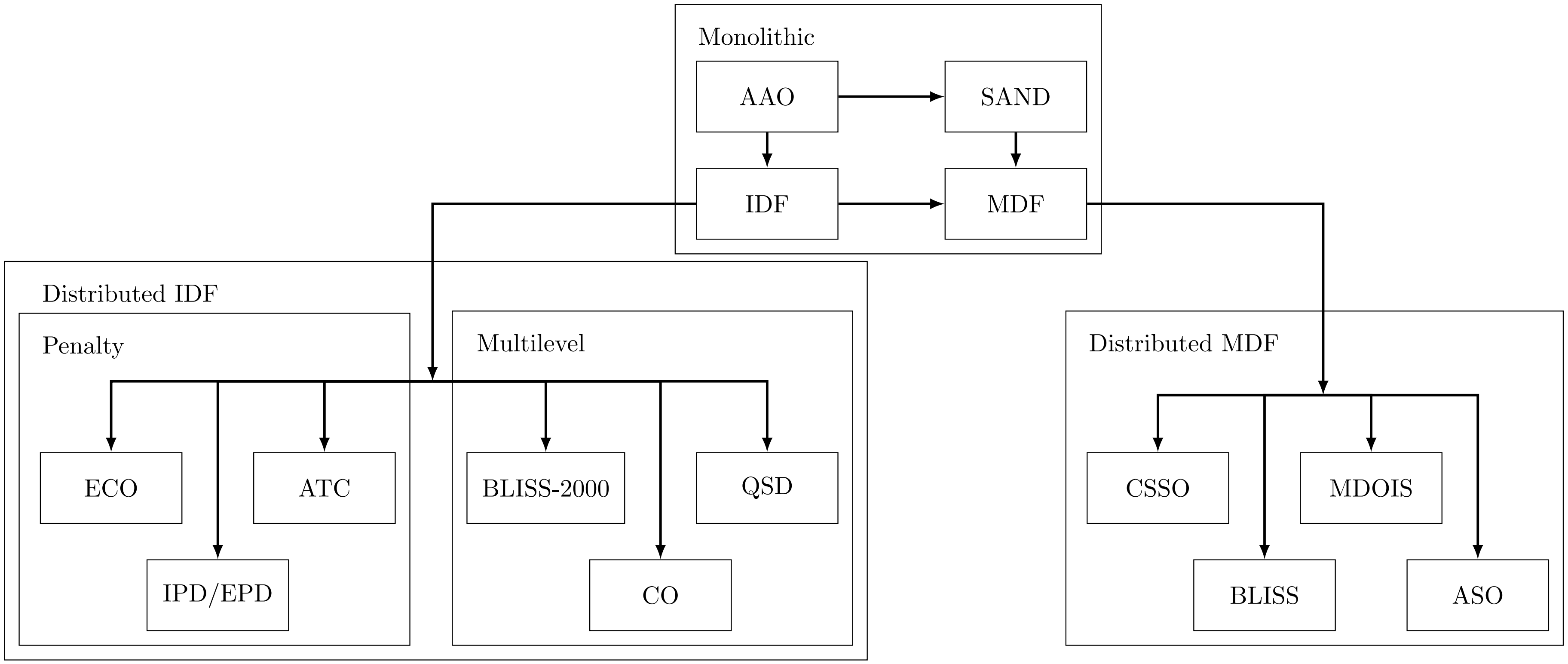


Figure 7: Classification of the MDO architectures.

Some notes on the classification diagram:

- Known relationships between the architectures are shown by arrows.
- we have only included the “core”architectures in our diagram. Details on the available variations for each distributed architecture are presented in the relevant sections.
- Note that none of the distributed architectures developed to date have been considered analogues of SAND.
- Our classification scheme does not distinguish between the different solution techniques for the distributed optimization problems. For example, we have not focused on the order in which the distributed problems are solved. Coordination schemes are partially addressed in the Distributed IDF group, where we have classified the architectures as either “penalty” or “multilevel”, based on whether penalty functions or a problem hierarchy is used in the coordination. This grouping follows from the work of de Wit and van Keulen [38].

The following sections introduce the distributed architectures for MDO.

- We prefer to use the term “distributed” as opposed to “hierarchical” or “multilevel” because these architectures do not necessarily create a hierarchy of problems to solve.

- Furthermore, neither the systems being designed nor the design team organization need to be hierarchical in nature for these architectures to be applicable.

- Our focus here is to provide a unified description of these architectures and explain some advantages and disadvantages of each. Along the way, we will point out variations and applications of each architecture that can be found in the literature.

- We also aim to review the state-of-the-art in architectures, since the most recent detailed architecture survey in the literature dates back from more than a decade ago [145]. More recent surveys, such as that of Agte et al. [1], discuss MDO more generally without detailing the architectures themselves.

4.2 Concurrent Subspace Optimization (CSSO)

- CSSO is one of the oldest distributed architectures for large-scale MDO problems.
- The original formulation [139, 18] decomposes the system problem into independent subproblems with disjoint sets of variables.
- Global sensitivity information is calculated at each iteration to give each subproblem a linear approximation to a multidisciplinary analysis, improving the convergence behavior.
- At the system level, a coordination problem is solved to recompute the "responsibility", "tradeoff", and "switch" coefficients assigned to each discipline to provide information on design variable preferences for nonlocal constraint satisfaction. Using these coefficients gives each discipline a certain degree of autonomy within the system as a whole.

The version we consider here, due to Sellar et al. [135], uses metamodel representations of each disciplinary analysis to efficiently model multidisciplinary interactions. Using our unified notation, the CSSO system subproblem is given by

$$\begin{aligned} \text{minimize} \quad & f_0\left(x, \tilde{y}\left(x, \tilde{y}\right)\right) \\ \text{with respect to} \quad & x \\ \text{subject to} \quad & c_0\left(x, \tilde{y}\left(x, \tilde{y}\right)\right) \geq 0 \\ & c_i\left(x_0, x_i, \tilde{y}_i\left(x_0, x_i, \tilde{y}_{j \neq i}\right)\right) \geq 0 \,\text{for}\, i = 1, \ldots, N \end{aligned} \tag{1}$$

and the discipline i subproblem is given by

$$\begin{aligned} \text{minimize} \quad & f_0\left(x, y_i\left(x_i, \tilde{y}_{j \neq i}\right), \tilde{y}_{j \neq i}\right) \\ \text{with respect to} \quad & x_0, x_i \\ \text{subject to} \quad & c_0\left(x, \tilde{y}\left(x, \tilde{y}\right)\right) \geq 0 \\ & c_i\left(x_0, x_i, y_i\left(x_0, x_i, \tilde{y}_{j \neq i}\right)\right) \geq 0 \\ & c_j\left(x_0, \tilde{y}_j\left(x_0, \tilde{y}\right)\right) \geq 0 \quad \text{for}\, j = 1, \ldots, N \, j \neq i. \end{aligned} \tag{2}$$

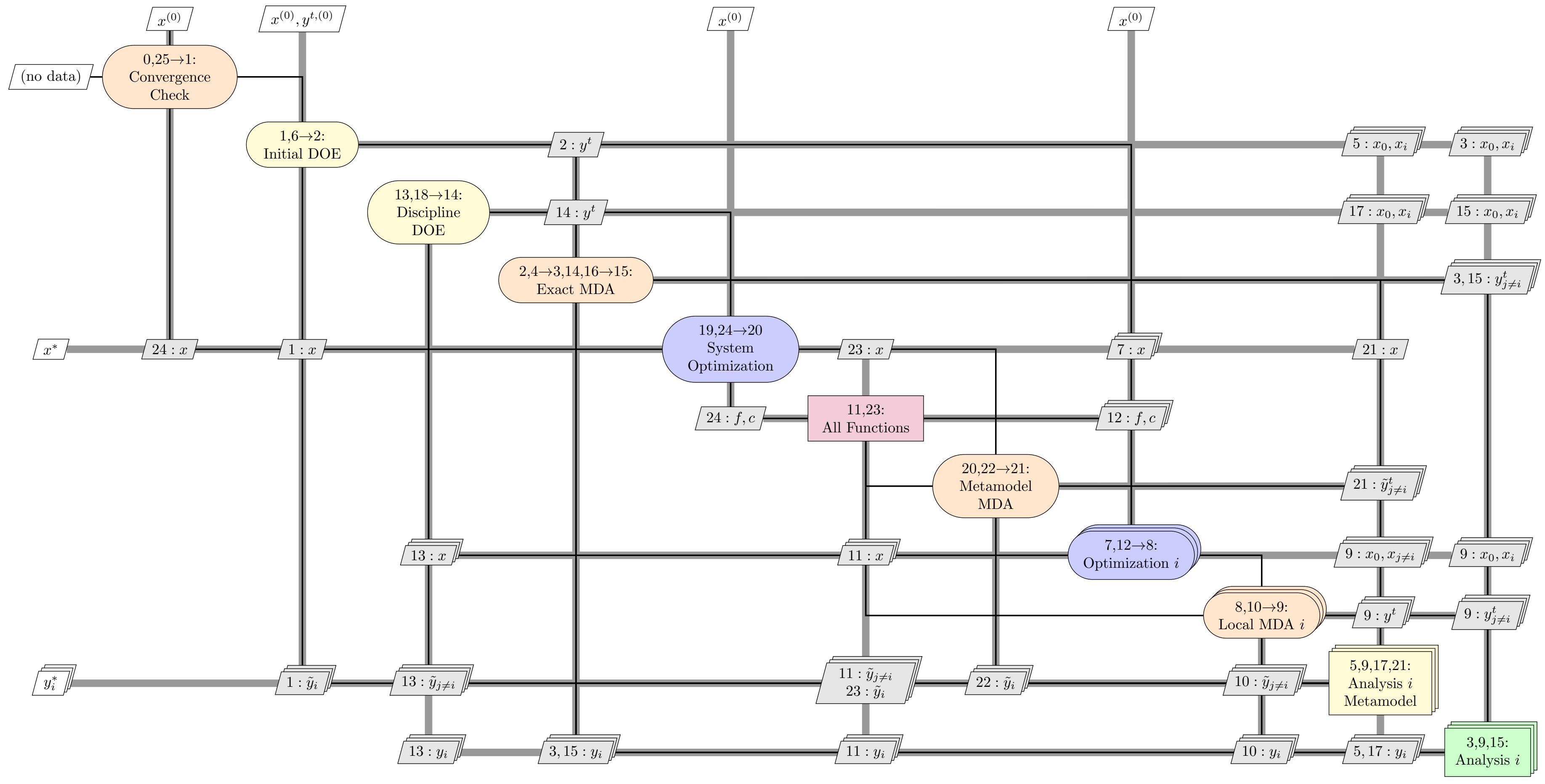


Figure 8: Diagram for the CSSO architecture.

Algorithm 2 CSSO

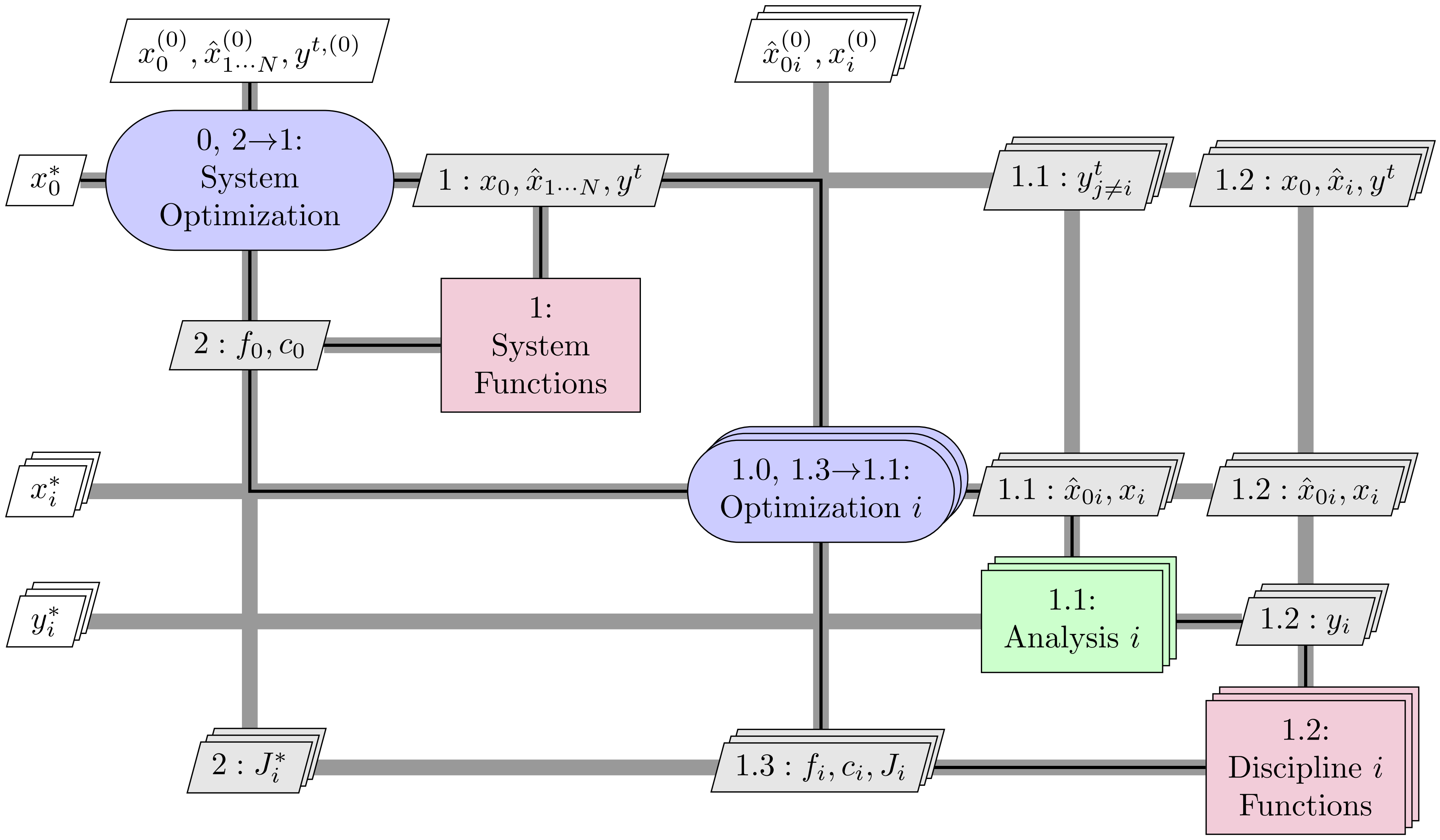
```
Input: Initial design variables $x$
Output: Optimal variables $x^*$, objective function $f^*$, and constraint values $c^*$
  0: Initiate main CSSO iteration
  repeat
      1: Initiate a design of experiments (DOE) to generate design points
      for Each DOE point do
          2: Initiate an MDA that uses exact disciplinary information
          repeat
              3: Evaluate discipline analyses
              4: Update coupling variables $y$
          until $4 \to 3$: MDA has converged
          5: Update the disciplinary metamodels with the latest design
      end for $6 \to 2$
      7: Initiate independent disciplinary optimizations (in parallel)
      for Each discipline $i$ do
          repeat
              8: Initiate an MDA with exact coupling variables for discipline $i$ and approximate coupling variables for the other disciplines
              repeat
                  9: Evaluate discipline $i$ outputs $y_i$, and metamodels for the other disciplines, $\tilde{y}_{j \neq i}$
              until $10 \to 9$: MDA has converged
              11: Compute objective $f_0$ and constraint functions $c$ using current data
          until $12 \to 8$: Disciplinary optimization $i$ has converged
      end for
      13: Initiate a DOE that uses the subproblem solutions as sample points
      for Each subproblem solution $i$ do
          14: Initiate an MDA that uses exact disciplinary information
          repeat
              15: Evaluate discipline analyses.
          until $16 \to 15$ MDA has converged
          17: Update the disciplinary metamodels with the newest design
      end for $18 \to 14$
      19: Initiate system-level optimization
      repeat
          20: Initiate an MDA that uses only metamodel information
          repeat
              21: Evaluate disciplinary metamodels
          until $22 \to 21$: MDA has converged
```

A potential pitfall of CSSO architecture is the necessity of including all design variables in the system subproblem. For industrial-scale design problems, this may not always be possible or practical.

There have been some benchmarks comparing CSSO with other MDO architectures. Perez et al., [121] Yi et al. [162], and Tedford and Martins [150] all show CSSO requiring many more analysis calls than other architectures to converge to an optimal design. The results of de Wit and van Keulen [37] showed that CSSO was unable to reach the optimal solution of even a simple minimum-weight two-bar truss problem. Thus, CSSO seems to be largely ineffective when compared with newer MDO architectures.

4.3 Collaborative Optimization (CO)

- In CO, the disciplinary optimization problems are formulated to be independent of each other by using target values of the coupling and shared design variables [20, 21].
- These target values are then shared with all disciplines during every iteration of the solution procedure.
- The complete independence of disciplinary subproblems combined with the simplicity of the data-sharing protocol makes this architecture attractive for problems with a small amount of shared data.



$x_0^{(0)}, \hat{x}_{1\ldots N}^{(0)}, y^{t,(0)}$
$\hat{x}_{0i}^{(0)}, x_i^{(0)}$
x_0^*
0, 2→1: System Optimization
$1 : x_0, \hat{x}_{1\ldots N}, y^t$
$1.1 : y^t_{j\neq i}$
$1.2 : x_0, \hat{x}_i, y^t$
$2 : f_0, c_0$
1: System Functions
x_i^*
1.0, 1.3→1.1: Optimization i
$1.1 : \hat{x}_{0i}, x_i$
$1.2 : \hat{x}_{0i}, x_i$
y_i^*
1.1: Analysis i
$1.2 : y_i$
$2 : J_i^*$
$1.3 : f_i, c_i, J_i$
1.2: Discipline i Functions

Figure 9: Diagram for the CO architecture.

Braun [20] formulated two versions of the CO architecture: CO$_1$ and CO$_2$. CO$_2$ is the most frequently used of these two original formulations so it will be the focus of our discussion. The CO$_2$ system subproblem is given by:

$$
\begin{aligned}
\text{minimize} \quad & f_0\left(x_0, \hat{x}_1, \ldots, \hat{x}_N, y^t\right) \\
\text{with respect to} \quad & x_0, \hat{x}_1, \ldots, \hat{x}_N, y^t \\
\text{subject to} \quad & c_0\left(x_0, \hat{x}_1, \ldots, \hat{x}_N, y^t\right) \geq 0 \\
& J_i^* = ||\hat{x}_{0i} - x_0||_2^2 + ||\hat{x}_i - x_i||_2^2 + \\
& \qquad ||y_i^t - y_i\left(\hat{x}_{0i}, x_i, y_{j\neq i}^t\right)||_2^2 = 0 \;\; \text{for}\, i = 1, \ldots, N
\end{aligned}
\tag{3}
$$

where $\hat{x}_{0i}$ are copies of the global design variables passed to discipline i and $\hat{x}_i$ are copies of the local design variables passed to the system subproblem.

- Note that copies of the local design variables are only made if those variables directly influence the objective.

- In CO$_1$, the quadratic equality constraints are replaced with linear equality constraints for each target-response pair. In either case, post-optimality sensitivity analysis, i.e. computing derivatives with respect to an optimized function, is required to evaluate the derivatives of the consistency constraints J_i^*.

The discipline i subproblem in both CO$_1$ and CO$_2$ is

$$\begin{aligned} \text{minimize} \quad & J_i\left(\hat{x}_{0i}, x_i, y_i\left(\hat{x}_{0i}, x_i, y^t_{j\neq i}\right)\right) \\ \text{with respect to} \quad & \hat{x}_{0i}, x_i \\ \text{subject to} \quad & c_i\left(\hat{x}_{0i}, x_i, y_i\left(\hat{x}_{0i}, x_i, y^t_{j\neq i}\right)\right) \geq 0. \end{aligned} \tag{4}$$

- The system-level problem is responsible for minimizing the design objective

- The discipline level problems minimize system inconsistency.

- Braun [20] showed that the CO problem statement is mathematically equivalent to the original AAO MDO problem.

Algorithm 3 Collaborative optimization

```
Input: Initial design variables $x$
Output: Optimal variables $x^*$, objective function $f^*$, and constraint values $c^*$
  0: Initiate system optimization iteration
  repeat
    1: Compute system subproblem objectives and constraints
    for Each discipline $i$ (in parallel) do
      1.0: Initiate disciplinary subproblem optimization
      repeat
        1.1: Evaluate disciplinary analysis
        1.2: Compute disciplinary subproblem objective and constraints
        1.3: Compute new disciplinary subproblem design point and $J_i$
      until $1.3 \to 1.1$: Optimization $i$ has converged
    end for
    2: Compute a new system subproblem design point
  until $2 \to 1$: System optimization has converged
```

In spite of the organizational advantage of having fully separate disciplinary subproblems, CO has major weaknesses in the mathematical formulation that lead to poor performance in practice [4, 40].

- In particular, the system problem in CO$_1$ has more equality constraints than variables, so if the system cannot be made fully consistent, the system subproblem is infeasible. This can also happen in CO$_2$, but it is not the most problematic issue.

- The most significant difficulty with CO$_2$ is that the constraint gradients of the system problem at an optimal solution are all zero vectors. This represents a breakdown in the constraint qualification of the Karush–Kuhn–Tucker optimality conditions, which slows down convergence for most gradient-based optimization software [4]. In the worst case, the CO$_2$ formulation may not converge at all.

These difficulties with the original formulations of CO have inspired several researchers to develop modifications to improve the behavior of the architecture.

In a few cases, problems have been solved with CO and a gradient-free optimizer, such as a genetic algorithm [164], or a gradient-based optimizer that does not use the Lagrange multipliers in the termination condition [96] to handle the troublesome constraints. While such approaches do avoid the obvious problems with CO, they bring other issues.

- Gradient-free optimizers are computationally expensive and can become the bottleneck within the CO architecture.

- Gradient-based optimizers that do not terminate based on Lagrange multiplier values, such as feasible direction methods, often fail in nonconvex feasible regions. As pointed out by DeMiguel [40], the CO system subproblem is set-constrained, i.e., nonconvex, because of the need to satisfy optimality in the disciplinary subproblems.

The approach taken by DeMiguel and Murray [40] to fix the problems with CO is to relax the troublesome constraints using an L_1 exact penalty function with a fixed penalty parameter value and add elastic variables to preserve the smoothness of the problem. This revised approach is called Modified Collaborative Optimization (MCO).

- This approach satisfies the requirement of mathematical rigor, as algorithms using the penalty function formulation are known to converge to an optimal solution under mild assumptions [44, 115].

- However, the test results of Brown and Olds [24] show strange behavior in a practical design problem. In light of their findings, the authors rejected MCO from further testing.

Another idea, proposed by Sobieski and Kroo [138], uses surrogate models, also known as metamodels, to approximate the post-optimality behavior of the disciplinary subproblems in the system subproblem.

- This both eliminates the post-optimality sensitivity calculation and improves the treatment of the consistency constraints.
- While the approach does seem to be effective for the problems they solve, to our knowledge, it has not been adopted by any other researchers to date.

The simplest and most effective known fix for the difficulties of CO involves relaxing the system subproblem equality constraints to inequalities with a relaxation tolerance, which was originally proposed by Braun et al. [21].

- This approach was also successful in other test problems [120, 103], where the choice of tolerance is a small fixed number, usually 10^{-6}.
- The effectiveness of this approach stems from the fact that a positive inconsistency value causes the gradient of the constraint to be nonzero if the constraint is active, eliminating the constraint qualification issue.
- Nonzero inconsistency is not an issue in a practical design setting provided the inconsistency is small enough such that other errors in the computational model dominate at the final solution.

Li et al. [93] build on this approach by adaptively choosing the tolerance during the solution procedure so that the system-level problem remains feasible at each iteration. This approach appears to work when applied to the test problems in [4], but has yet to be verified on larger test problems.

Despite the numerical issues, CO has been widely implemented on a number of MDO problems. Most of applications are in the design of aerospace systems. Examples include

1. the design of launch vehicles [23],
2. rocket engines [27],
3. satellite constellations [26],
4. flight trajectories [22, 91],
5. flight control systems [122],
6. preliminary design of complete aircraft [88, 102], and
7. aircraft family design [7].

Outside aerospace engineering, CO has been applied to problems involving automobile engines [109], bridge design [9], railway cars [42], and even the design of a scanning optical microscope [124].

Adaptations of the CO architecture have also been developed for multiobjective, robust, and multifidelity MDO problems. Multiobjective formulations of CO were first described by Tappeta and Renaud [149]. McAllister et al. [110] present a multiobjective approach using linear physical programming. Available robust design formulations incorporate the decision-based models of Gu et al. [53] and McAllister and Simpson [109], the implicit uncertainty propagation method of Gu et al. [54], and the fuzzy computing models of Huang et al. [68]. Multiple model fidelities were integrated into CO for an aircraft design problem by Zadeh and Toropov [163].

The most recent version of CO — Enhanced Collaborative Optimization (ECO) — was developed by Roth and Kroo [129, 128]. Figure 10 shows the XDSM corresponding to this architecture. The problem formulation of ECO, while still being derived from the same basic problem as the original CO architecture, is radically different and therefore deserves additional attention. In a sense, the roles of the system and discipline optimization have been reversed in ECO when compared to CO. In ECO the system subproblem minimizes system infeasibility, while the disciplinary subproblems minimize the system objective. The system subproblem is

$$\begin{aligned} \text{minimize} \quad & J_0 = \sum_{i=1}^{N} ||\hat{x}_{0i} - x_0||_2^2 + ||y_i^t - y_i\left(x_0, x_i, y_{j\neq i}^t\right)||_2^2 \\ \text{with respect to} \quad & x_0, y^t. \end{aligned} \tag{5}$$

Note that this subproblem is unconstrained. Also, unlike CO, post-optimality sensitivities are not required by the system subproblem because the disciplinary responses are treated as parameters. The system subproblem chooses the shared design variables by averaging all disciplinary preferences.

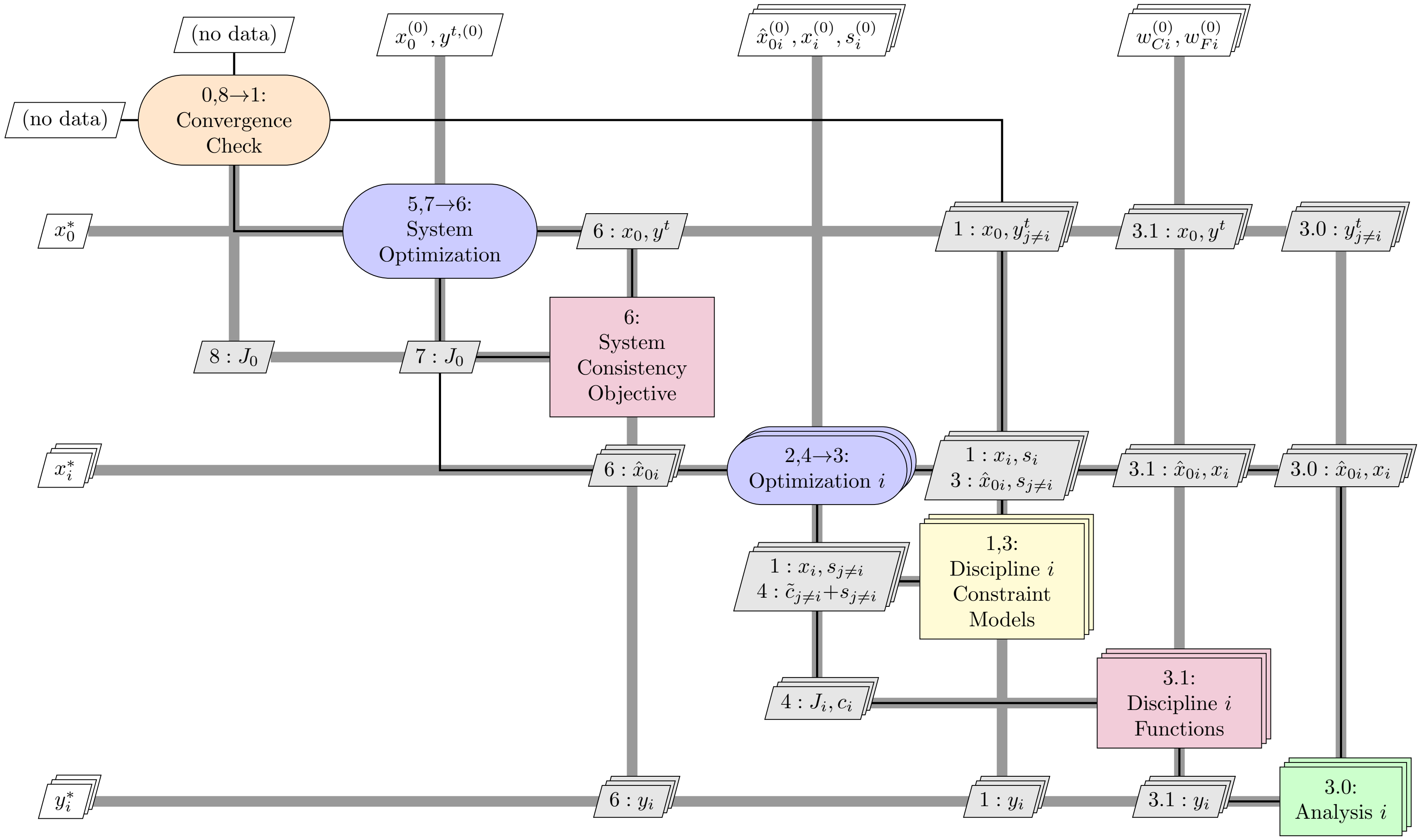

(no data)
$x_0^{(0)}, y^{t,(0)}$
$\hat{x}_{0i}^{(0)}, x_i^{(0)}, s_i^{(0)}$
$w_{Ci}^{(0)}, w_{Fi}^{(0)}$
(no data)
0,8→1: Convergence Check
x_0^*
5,7→6: System Optimization
$6 : x_0, y^t$
$1 : x_0, y^t_{j \neq i}$
$3.1 : x_0, y^t$
$3.0 : y^t_{j \neq i}$
$8 : J_0$
$7 : J_0$
6: System Consistency Objective
x_i^*
$6 : \hat{x}_{0i}$
2,4→3: Optimization i
$1 : x_i, s_i$
$3 : \hat{x}_{0i}, s_{j \neq i}$
$3.1 : \hat{x}_{0i}, x_i$
$3.0 : \hat{x}_{0i}, x_i$
$1 : x_i, s_{j \neq i}$
$4 : \tilde{c}_{j \neq i} + s_{j \neq i}$
1,3: Discipline i Constraint Models
$4 : J_i, c_i$
3.1: Discipline i Functions
y_i^*
$6 : y_i$
$1 : y_i$
$3.1 : y_i$
3.0: Analysis i

Figure 10: XDSM for the ECO architecture

The i^{th} disciplinary subproblem in ECO is

$$\begin{array}{rll}
\text{minimize} & J_i = \tilde{f}_0\left(\hat{x}_{0i}, y_i\left(\hat{x}_{0i}, x_i, y^t_{j\neq i}\right)\right) + & \\
& w_{Ci}\left(||\hat{x}_{0i} - x_0||_2^2 + ||y_i^t - y_i\left(\hat{x}_{0i}, x_i, y^t_{j\neq i}\right)||_2^2\right) + & \\
& w_{Fi} \displaystyle\sum_{j=1, j\neq i}^{N} \sum_{k=1}^{n_s} s_{jk} & \\
\text{with respect to} & \hat{x}_{0i}, x_i, s_{j\neq i} & \\
\text{subject to} & c_i\left(\hat{x}_{0i}, x_i, y_i\left(\hat{x}_{0i}, x_i, y^t_{j\neq i}\right)\right) \geq 0 & \\
& \tilde{c}_{j\neq i}\left(\hat{x}_{0i}\right) - s_{j\neq i} \geq 0 & j = 1, \ldots, N \\
& s_{j\neq i} \geq 0 & j = 1, \ldots, N,
\end{array} \tag{6}$$

where w_{Ci} and w_{Fi} are penalty weights for the consistency and nonlocal design constraints, and s is a local set of elastic variables for the constraint models. The w_{Fi} penalty weights are chosen to be larger than the largest Lagrange multiplier, while the w_{Ci} weights are chosen to guide the optimization toward a consistent solution.

The main new idea introduced in ECO is to include linear models of nonlocal constraints, represented by $\tilde{c}_{j \neq i}$, and a quadratic model of the system objective function in each disciplinary subproblem, represented by $\tilde{f}_0$. This is meant to increase each discipline's "awareness" of their influence on other disciplines and the global objective as a whole. The constraint models for each discipline are constructed by first solving the optimization problem that minimizes the constraint violation with respect to local elastic and design variables.

$$
\begin{aligned}
\text{minimize} \quad & \sum_{k=1}^{n_s} s_{ik} \\
\text{with respect to} \quad & x_i, s_i \\
\text{subject to} \quad & c_i\left(x_0, x_i, y_i\left(x_0, x_i, y^t_{j \neq i}\right)\right) + s_i \geq 0 \\
& s_i \geq 0.
\end{aligned}
\tag{7}
$$

Algorithm 4 Enhanced collaborative optimization

```
Input: Initial design variables $x$
Output: Optimal variables $x^*$, objective function $f^*$, and constraint values $c^*$
  0: Initiate ECO iteration
  repeat
    for Each discipline $i$ do
      1: Create linear constraint model
      2: Initiate disciplinary subproblem optimization
      repeat
        3: Interrogate nonlocal constraint models with local copies of shared
        variables
        3.0: Evaluate disciplinary analysis
        3.1: Compute disciplinary subproblem objective and constraints
        4: Compute new disciplinary subproblem design point and $J_i$
      until $4 \rightarrow 3$: Disciplinary optimization subproblem has converged
    end for
    5: Initiate system optimization
    repeat
      6: Compute $J_0$
      7: Compute updated values of $x_0$ and $y^t$.
    until $7 \rightarrow 6$: System optimization has converged
  until $8 \rightarrow 1$: The $J_0$ is below specified tolerance
```

Based on the Roth's results [129, 128], ECO is effective in reducing the number of discipline analyses compared to CO. The trade-off is in the additional time required to build and update the models for each discipline, weighed against the simplified solution to the decomposed optimization problems. The results also show that ECO compares favorably with the Analytical Target Cascading architecture (which we will describe in Section 4.5).

While ECO seems to be effective, CO tends to be an inefficient architecture for solving MDO problems.

- Without any of the fixes discussed in this section, the architecture always requires a disproportionately large number of function and discipline evaluations [79, 83, 37, 162], assuming it converges at all.

- When the system-level equality constraints are relaxed, the results from CO are more competitive with other distributed architectures [121, 103, 150] but still compare poorly with the results of monolithic architectures.

4.4 Bilevel Integrated System Synthesis (BLISS)

The BLISS architecture [143], like CSSO, is a method for decomposing the MDF problem along disciplinary lines.

- Unlike CSSO, however, BLISS assigns local design variables to disciplinary subproblems and shared design variables to the system subproblem.

- The basic approach of the architecture is to form a path in the design space using a series of linear approximations to the original design problem, with user-defined bounds on the design variable steps, to prevent the design point from moving so far away that the approximations are too inaccurate.

This is an idea similar to that of trust-region methods [34]. These approximations are constructed at each iteration using global sensitivity information.

The BLISS system level subproblem is formulated as

$$
\begin{aligned}
\text{minimize} \quad & (f_0^*)_0 + \left(\frac{\mathrm{d}f_0^*}{\mathrm{d}x_0}\right)\Delta x_0 \\
\text{with respect to} \quad & \Delta x_0 \\
\text{subject to} \quad & (c_0^*)_0 + \left(\frac{\mathrm{d}c_0^*}{\mathrm{d}x_0}\right)\Delta x_0 \geq 0 \\
& (c_i^*)_0 + \left(\frac{\mathrm{d}c_i^*}{\mathrm{d}x_0}\right)\Delta x_0 \geq 0 \;\; \text{for}\, i = 1, \ldots, N \\
& \Delta x_{0L} \leq \Delta x_0 \leq \Delta x_{0U}.
\end{aligned} \tag{8}
$$

The BLISS discipline i subproblem is given by

$$\begin{aligned}
\text{minimize} \quad & (f_0)_0 + \left(\frac{\mathrm{d}f_0}{\mathrm{d}x_i}\right)\Delta x_i \\
\text{with respect to} \quad & \Delta x_i \\
\text{subject to} \quad & (c_0)_0 + \left(\frac{\mathrm{d}c_0}{\mathrm{d}x_i}\right)\Delta x_i \geq 0 \\
& (c_i)_0 + \left(\frac{\mathrm{d}c_i}{\mathrm{d}x_i}\right)\Delta x_i \geq 0 \\
& \Delta x_{iL} \leq \Delta x_i \leq \Delta x_{iU}.
\end{aligned} \tag{9}$$

Note the extra set of constraints in both system and discipline subproblems denoting the design variables bounds.

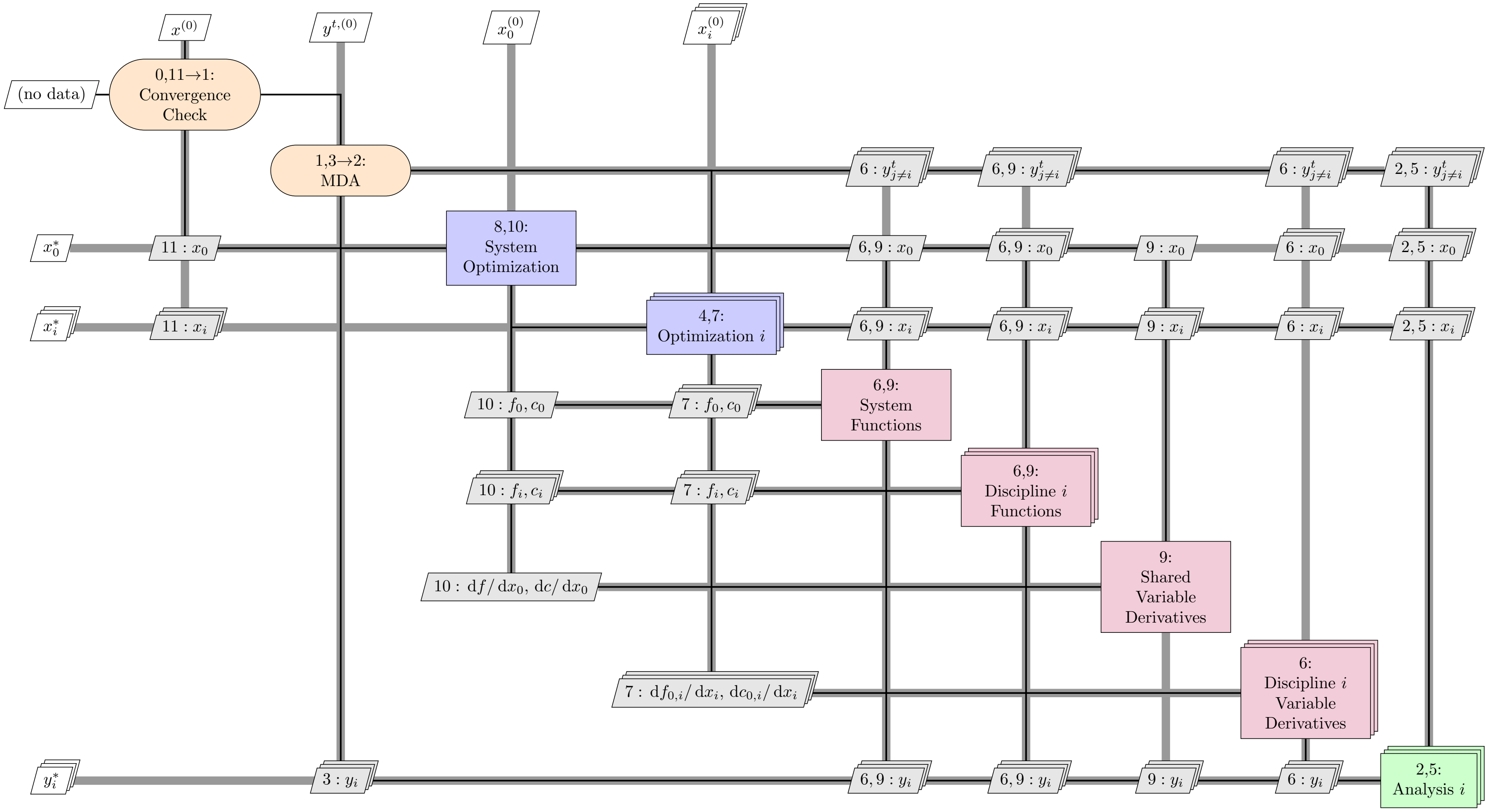


Figure 11: Diagram for the BLISS architecture

Algorithm 5 BLISS

```
Input: Initial design variables $x$
Output: Optimal variables $x^*$, objective function $f^*$, and constraint values $c^*$
  0: Initiate system optimization
  repeat
    1: Initiate MDA
    repeat
      2: Evaluate discipline analyses
      3: Update coupling variables
    until $3 \to 2$: MDA has converged
    4: Initiate parallel discipline optimizations
    for Each discipline $i$ do
      5: Evaluate discipline analysis
      6: Compute objective and constraint function values and derivatives with
      respect to local design variables
      7: Compute the optimal solutions for the disciplinary subproblem
    end for
    8: Initiate system optimization
    9: Compute objective and constraint function values and derivatives with
    respect to shared design variables using post-optimality sensitivity analysis
    10: Compute optimal solution to system subproblem
  until $11 \to 1$: System optimization has converged
```

In order to prevent violation of the disciplinary constraints by changes in the shared design variables, post-optimality sensitivity information is required to solve the system subproblem.

For this step, Sobieski [143] presents two methods:

BLISS/A: using a generalized version of the Global Sensitivity Equations [140], and

BLISS/B: using the "pricing" interpretation of local Lagrange multipliers.

Other variations use response surface approximations to compute post-optimality sensitivity data [81, 73].

Some notes on the BLISS algorithm:

- due to the linear nature of the optimization problems under consideration, repeated interrogation of the objective and constraint functions is not necessary once gradient information is available.

- However, this reliance on linear approximations is not without difficulties. If the underlying problem is highly nonlinear, the algorithm may converge slowly. The presence of user-defined variable bounds may help the convergence if these bounds are properly chosen, such as through a trust region framework.

- Detailed knowledge of the design space can also help, but this increases the overhead cost of implementation.

Two other adaptations of the original BLISS architecture are known in the literature.

1. The first is Ahn and Kwon's proBLISS [2], an architecture for reliability-based MDO. Their results show that the architecture is competitive with reliability-based adaptations of MDF and IDF.

2. The second is LeGresley and Alonso's BLISS/POD [92], an architecture that integrates a reduced-order modeling technique called Proper Orthogonal Decomposition [13] to reduce the cost of the multidisciplinary analysis and sensitivity analysis steps. Their results show a significant improvement in the performance of BLISS, to the point where it is almost competitive with MDF.

As an enhancement of the original BLISS, a radically different formulation called **BLISS-2000** was developed by Sobieski et al. [144].

- BLISS-2000 does not require a multidisciplinary analysis to restore feasibility of the design, so we have separated it from other BLISS variants in the classification tree shown earlier.

- like other IDF-derived architectures, BLISS-2000 uses coupling variable targets to enforce consistency at the optimum. Information exchange between system and discipline subproblems is completed through surrogate models of the disciplinary optima.

The BLISS-2000 system subproblem is given by

$$\begin{aligned}
\text{minimize} \quad & f_0\left(x, \tilde{y}\left(x, y^t\right)\right) \\
\text{with respect to} \quad & x_0, y^t, w \\
\text{subject to} \quad & c_0\left(x, \tilde{y}\left(x, y^t, w\right)\right) \geq 0 \\
& y_i^t - \tilde{y}_i\left(x_0, x_i, y_{j\neq i}^t, w_i\right) = 0 \ \text{for}\, i = 1, \ldots, N.
\end{aligned} \tag{10}$$

The BLISS-2000 discipline i subproblem is

$$\begin{aligned}
\text{minimize} \quad & w_i^T y_i \\
\text{with respect to} \quad & x_i \\
\text{subject to} \quad & c_i\left(x_0, x_i, y_i\left(x_0, x_i, y_{j\neq i}^t\right)\right) \geq 0.
\end{aligned} \tag{11}$$

A unique aspect of this architecture is the use of a vector of weighting coefficients, w_i, attached to the disciplinary states. These weighting coefficients give the user a measure of control over state variable preferences. Generally speaking, the coefficients should be chosen based on the structure of the global objective to allow disciplinary subproblems to find an optimum more quickly. How much the choice of coefficients affects convergence has yet to be determined.

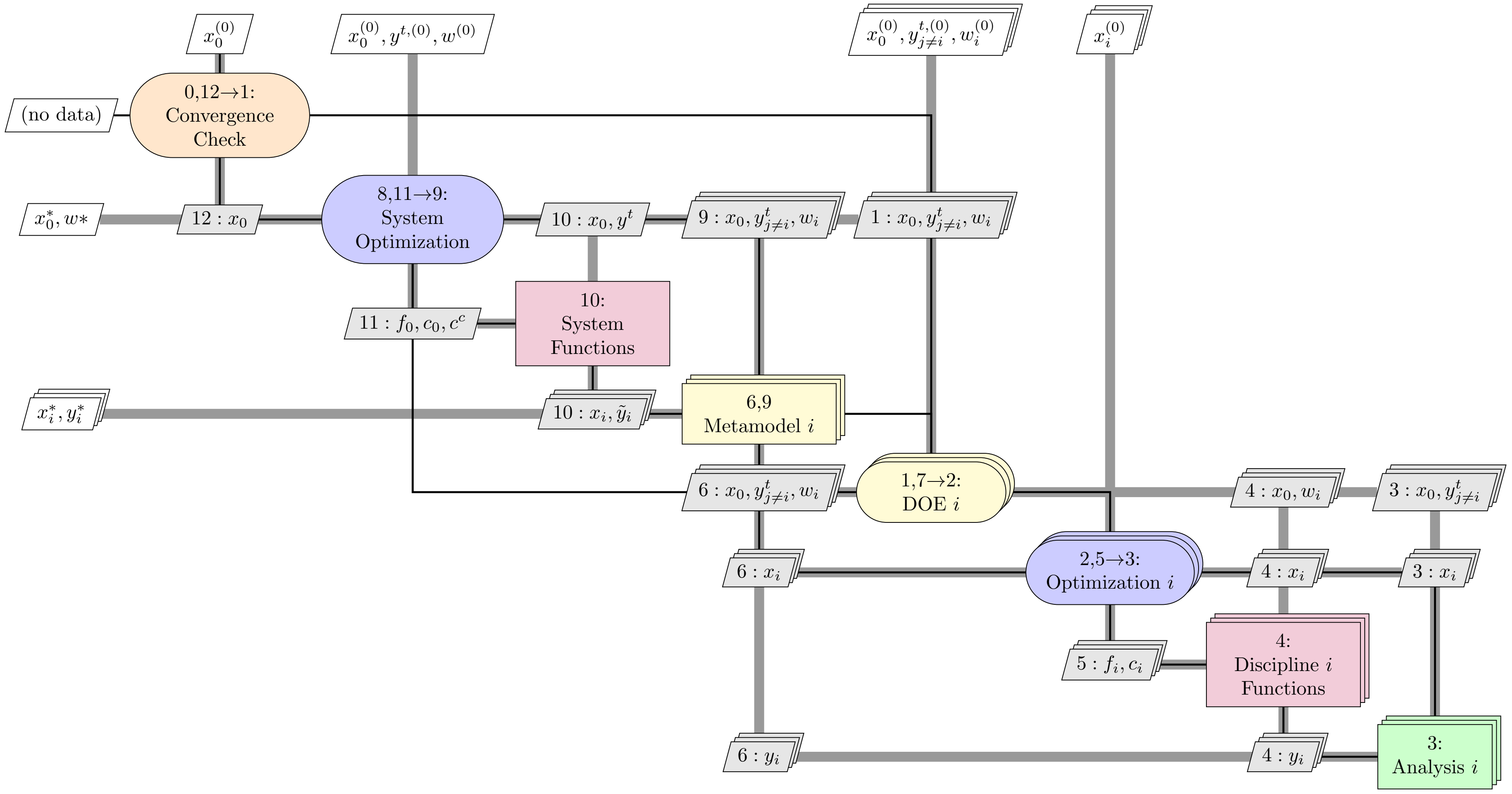


Figure 12: Diagram for the BLISS-2000 architecture

Algorithm 6 BLISS-2000

```
Input: Initial design variables $x$
Output: Optimal variables $x^*$, objective function $f^*$, and constraint values $c^*$
  0: Initiate system optimization
  repeat
    for Each discipline $i$ do
      1: Initiate a DOE
      for Each DOE point do
        2: Initiate discipline subproblem optimization
        repeat
          3: Evaluate discipline analysis
          4: Compute discipline objective $f_i$ and constraint functions $c_i$
          5: Update the local design variables $x_i$
        until $5 \to 3$: Disciplinary optimization subproblem has converged
        6: Update metamodel of optimized disciplinary subproblem with new solution
      end for $7 \to 1$
    end for
    8: Initiate system subproblem optimization
    repeat
      9: Interrogate metamodels with current values of system variables
      10: Compute system objective and constraint function values
      11: Compute a new system design point
    until $11 \to 9$ System subproblem has converged
  until $12 \to 1$: System optimization has converged
```

A unique aspect of this architecture is the use of a vector of weighting coefficients, w_i, attached to the disciplinary states. These weighting coefficients give the user a measure of control over state variable preferences.

- the coefficients should be chosen based on the structure of the global objective to allow disciplinary subproblems to find an optimum more quickly

- How much the choice of coefficients affects convergence has yet to be determined.

BLISS-2000 possesses several advantages over the original BLISS architecture.

1. the solution procedure is much easier to understand.

2. the decomposed problem formulation of BLISS-2000 is equivalent to the AAO problem we want to solve [144].

3. by using metamodels for each discipline, rather than for the whole system, the calculations for BLISS-2000 can be run in parallel with minimal communication between disciplines.

4. BLISS-2000 seems to be more flexible than its predecessor.

Recently, Sobieski detailed an extension of BLISS-2000 to handle multilevel, system-of-systems problems [142]. In spite of these advantages, it appears that BLISS-2000 has not been used nearly as frequently as the original BLISS formulation.

4.5 Analytical Target Cascading (ATC)

The ATC architecture was not initially developed as an MDO architecture, but as a method to propagate system targets — i.e., requirements or desirable properties — through a hierarchical system to achieve a feasible system design satisfying these targets [74, 77].

- If the system targets were unattainable, the ATC architecture would return a design point minimizing the inattainability.

- Effectively, the ATC architecture is no different from an MDO architecture with a system objective of minimizing the squared difference between a set of system targets and model responses.

- By simply changing the objective function, we can solve general MDO problems using ATC.

The ATC problem formulation that we present here is due to Tosserams et al. [153]:

$$\begin{aligned}
\text{minimize} \quad & f_0\left(x, y^t\right) + \sum_{i=1}^{N} \Phi_i\left(\hat{x}_{0i} - x_0, y_i^t - y_i\left(x_0, x_i, y^t\right)\right) + \\
& \Phi_0\left(c_0\left(x, y^t\right)\right) \\
\text{with respect to} \quad & x_0, y^t,
\end{aligned} \tag{12}$$

where Φ_0 is a penalty relaxation of the global design constraints and Φ_i is a penalty relaxation of the discipline i consistency constraints. The i^{th} discipline subproblem is:

$$\begin{aligned}
\text{minimize} \quad & f_0\left(\hat{x}_{0i}, x_i, y_i\left(\hat{x}_{0i}, x_i, y^t_{j\neq i}\right), y^t_{j\neq i}\right) + f_i\left(\hat{x}_{0i}, x_i, y_i\left(\hat{x}_{0i}, x_i, y^t_{j\neq i}\right)\right) + \\
& \Phi_i\left(y_i^t - y_i\left(\hat{x}_{0i}, x_i, y^t_{j\neq i}\right), \hat{x}_{0i} - x_0\right) + \\
& \Phi_0\left(c_0\left(\hat{x}_{0i}, x_i, y_i\left(\hat{x}_{0i}, x_i, y^t_{j\neq i}\right), y^t_{j\neq i}\right)\right) \\
\text{with respect to} \quad & \hat{x}_{0i}, x_i \\
\text{subject to} \quad & c_i\left(\hat{x}_{0i}, x_i, y_i\left(\hat{x}_{0i}, x_i, y^t_{j\neq i}\right)\right) \geq 0.
\end{aligned} \tag{13}$$

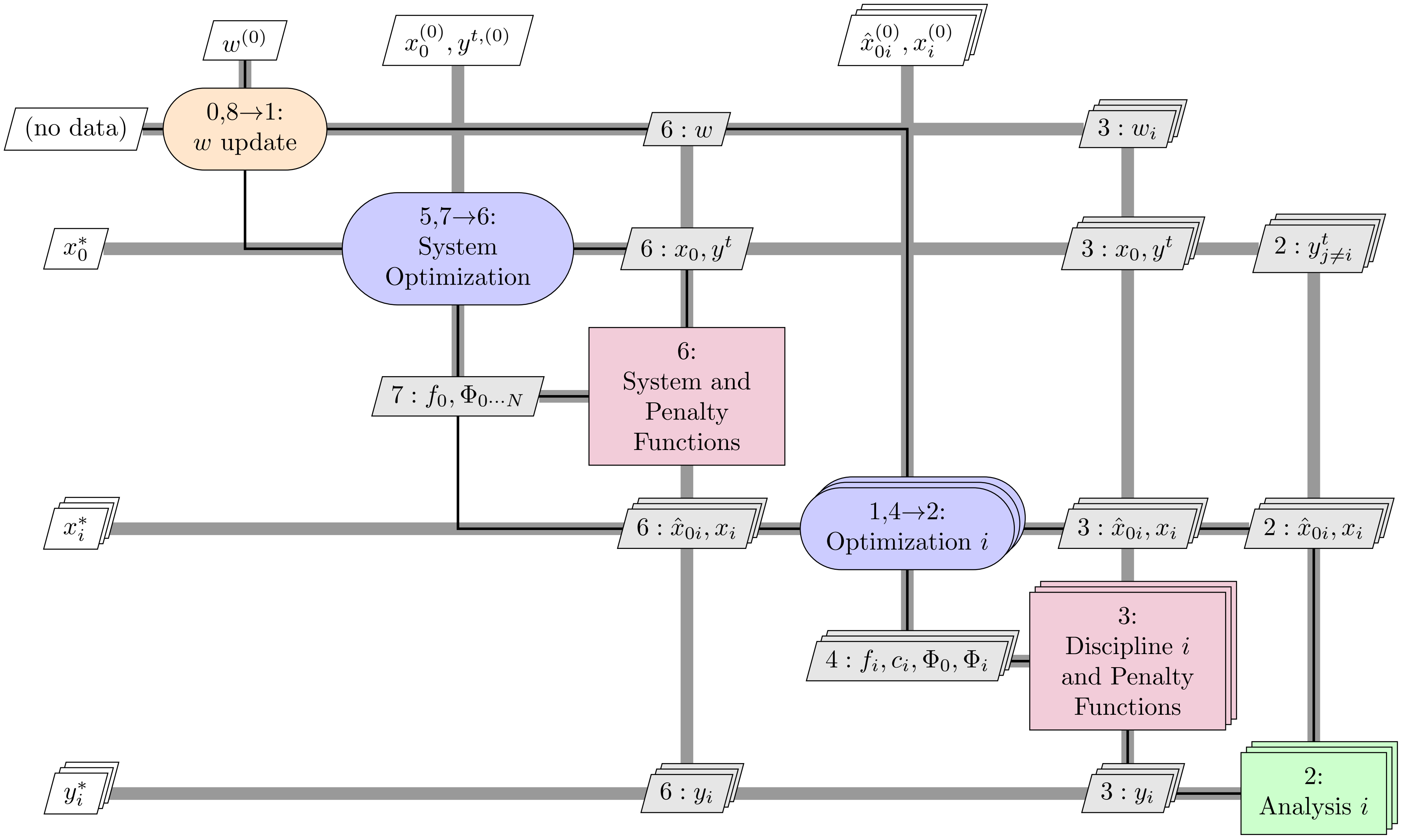


Figure 13: Diagram for the ATC architecture

Algorithm 7 ATC

```
Input: Initial design variables $x$
Output: Optimal variables $x^*$, objective function $f^*$, and constraint values $c^*$
  0: Initiate main ATC iteration
  repeat
    for Each discipline $i$ do
      1: Initiate discipline optimizer
      repeat
        2: Evaluate disciplinary analysis
        3: Compute discipline objective and constraint functions and penalty
        function values
        4: Update discipline design variables
      until $4 \rightarrow 2$: Discipline optimization has converged
    end for
    5: Initiate system optimizer
    repeat
      6: Compute system objective, constraints, and all penalty functions
      7: Update system design variables and coupling targets.
    until $7 \rightarrow 6$: System optimization has converged
    8: Update penalty weights
  until $8 \rightarrow 1$: Penalty weights are large enough
```

Note that ATC can be applied to a multilevel hierarchy of systems just as well as a discipline-based non-hierarchic system.

- In the multilevel case, the penalty functions are applied to all constraints that combine local information with information from the levels immediately above or below the current one.

- Also note that post-optimality sensitivity data is not needed in any of the subproblems as nonlocal data are always treated as fixed values in the current subproblem.

The most common penalty functions in ATC are quadratic penalty functions.

- The proper selection of the penalty weights is important for both final inconsistency in the discipline models and convergence of the algorithm.
- Michalek and Papalambros [112] present an effective weight update method that is especially useful when unattainable targets have been set in a traditional ATC process.
- Michelena et al. [114] present several coordination algorithms using ATC with quadratic penalty functions and demonstrate the convergence for all of them.

Note that, as with penalty methods for general optimization problems, (see, e.g., Nocedal and Wright [115, chap. 17]) the solution of the MDO problem must be computed to reasonable accuracy before the penalty weights are updated. However, because we are now dealing with a distributed set of subproblems, the whole hierarchy of subproblems must be solved for a given set of weights. This is due to the nonseparable nature of the quadratic penalty function.

Several other penalty function choices and associated coordination approaches have also been devised for ATC.

1. Kim et al. [75] outline a version of ATC that uses Lagrangian relaxation and a sub-gradient method to update the multiplier estimates.

2. Tosserams et al. [151] use augmented Lagrangian relaxation with Bertsekas' method of multipliers [14] and alternating direction method of multipliers [15] to update the penalty weights. They also group this variant of ATC into a larger class of coordination algorithms known as Augmented Lagrangian Coordination [153].

3. Li et al. [94] apply the diagonal quadratic approximation approach of Ruszcynski [130] to the augmented Lagrangian to eliminate subproblem coupling through the quadratic terms and further parallelize the architecture.

4. Han and Papalambros [62] propose a version of ATC based on sequential linear programming [11, 28], where inconsistency is penalized using infinity norms. They later presented a convergence proof of this approach in a short note [61].

For each of the above penalty function choices, ATC was able to produce the same design solutions as the monolithic architectures.

Despite having been developed relatively recently, the ATC architecture has been widely used. By far, ATC has been most frequently applied to design problems in the field for which it was developed, the automotive industry [84, 76, 78, 19, 85, 147, 29, 63].

- However, the ATC approach has also proven to be useful in aircraft design [8, 7, 158] and building design [32].

- ATC has also found applications outside of strict engineering design problems, including manufacturing decisions [95], supply chain management [67], and marketing decisions in product design [111].

- Huang et al. [66] have developed an ATC-specific web portal to solve optimization problems via the ATC architecture. Etman et al. [43] discuss the automatic implementation of coordination procedures, using ATC as an example architecture.

- There are ATC formulations that can handle integer variables [113] and probabilistic design problems [86, 98].

- Another important adaptation of ATC applies to problems with block-separable linking constraints [155]. In this class of problems, c_0 consists of constraints which are sums of functions depending on only shared variables and the local variables of one discipline.

The performance of ATC compared with other architectures is not well known because only one result is available.

- In de Wit and Van Keulen's architecture comparison [37], ATC is competitive with all other benchmarked distributed architectures, including standard versions of CSSO, CO, and BLISS.

- However, ATC *and the other distributed architectures* are not competitive with a monolithic architecture in terms of the number of function and discipline evaluations.

More commonly, different versions of ATC are benchmarked against each other.

- Tosserams et al. [151] compared the augmented Lagrangian penalty approach with the quadratic penalty approach and found much improved results with the alternating direction method of multipliers.

- Surprisingly, de Wit and van Keulen [37] found the augmented Lagrangian version performed worse than the quadratic penalty method for their test problem.

- Han and Papalambros [62] compared their sequential linear programming version of ATC to several other approaches and found a significant reduction in the number of function evaluations. However, they note that the coordination overhead is large compared to other ATC versions and still needs to be addressed.

4.6 Exact and Inexact Penalty Decomposition (EPD and IPD)

If there are no system-wide constraints or objectives, i.e., if neither f_0 and c_0 exist, the Exact or Inexact Penalty Decompositions (EPD or IPD) [39, 41] may be employed. Both formulations rely on solving the disciplinary subproblem

$$\begin{aligned} \text{minimize} \quad & f_i\left(\hat{x}_{0i}, x_i, y_i\left(\hat{x}_{0i}, x_i, y^t_{j\neq i}\right)\right) + \Phi_i\left(\hat{x}_{0i} - x_0, y^t_i - y_i\left(\hat{x}_{0i}, x_i, y^t_{j\neq i}\right)\right) \\ \text{with respect to} \quad & \hat{x}_{0i}, x_i \\ \text{subject to} \quad & c_i\left(\hat{x}_{0i}, x_i, y_i\left(x_{0i}, x_i, y^t_{j\neq i}\right)\right) \geq 0. \end{aligned} \tag{14}$$

Here, Φ_i denotes the penalty function associated with the inconsistency between the i^{th} disciplinary information and the system information. In EPD, Φ_i is an L_1 penalty function with additional variables and constraints added to ensure smoothness. In IPD, Φ_i is a quadratic penalty function with appropriate penalty weights. The notation $\hat{x}_{0i}$ denotes a local copy of the shared design variables in discipline i, while x_0 denotes the system copy.

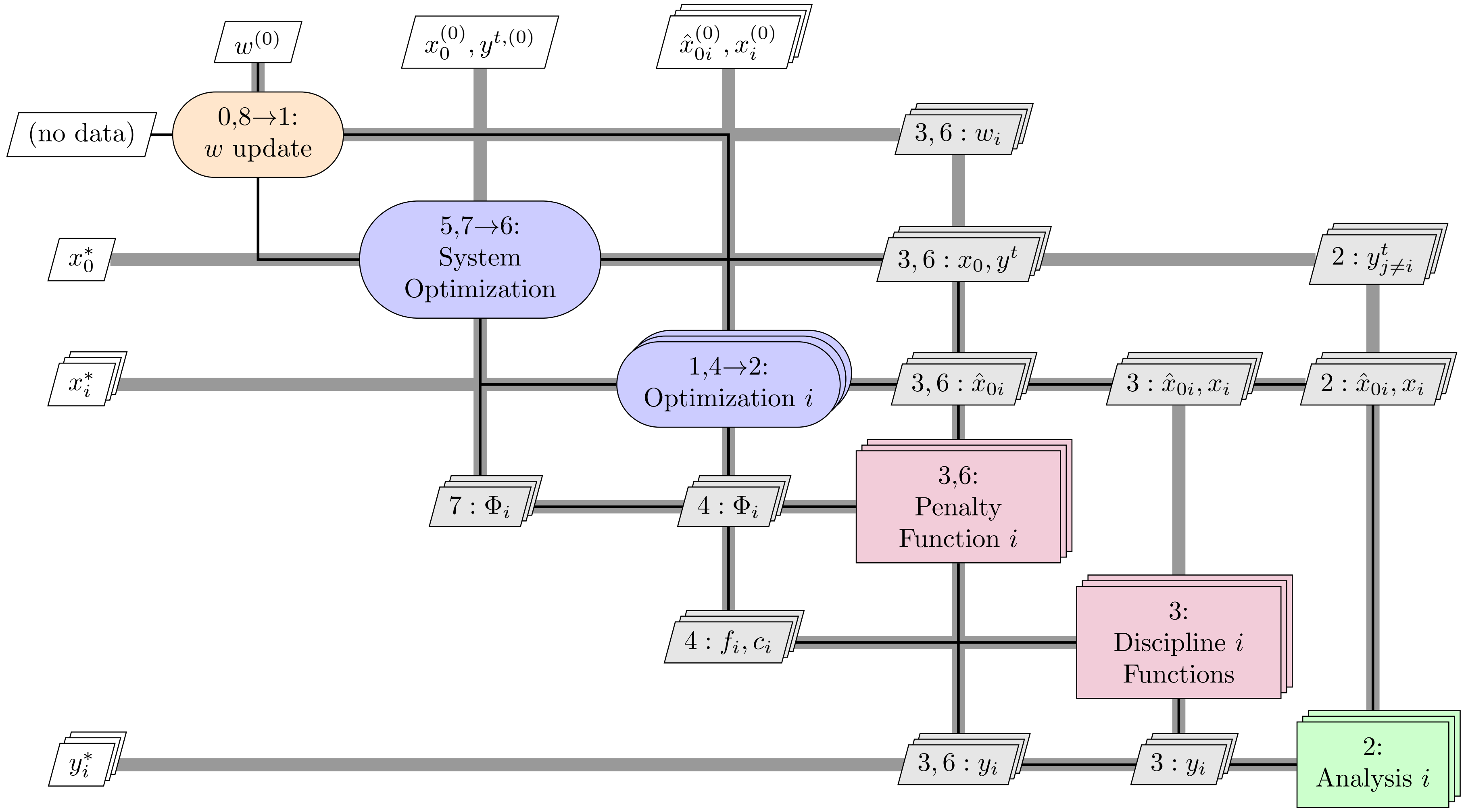


Figure 14: Diagram for the penalty decomposition architectures EPD and IPD

At the system level, the subproblem is an unconstrained minimization with respect to the target variables. The objective function is the sum of the optimized disciplinary penalty terms, denoted as Φ_i^*.

$$\begin{aligned} \text{minimize} \quad & \sum_{i=1}^{N} \Phi_i^* \left(x_0, y^t\right) = \sum_{i=1}^{N} \Phi_i \left(\hat{x}_{0i} - x_0, y_i^t - y_i \left(\hat{x}_{0i}, x_i, y_{j\neq i}^t\right)\right) \\ \text{with respect to} \quad & x_0, y^t \end{aligned} \tag{15}$$

The penalty weights are updated upon solution of the system problem. Figure 14 shows the XDSM for this architecture, where w represents the penalty weights. The sequence of operations in this architecture is detailed in Algorithm 8.

Algorithm 8 EPD and IPD

```
Input: Initial design variables $x$
Output: Optimal variables $x^*$, objective function $f^*$, and constraint values $c^*$
  0: Initiate main iteration
  repeat
    for Each discipline $i$ do
      repeat
        1: Initiate discipline optimizer
        2: Evaluate discipline analysis
        3: Compute discipline objective and constraint functions, and penalty function values
        4: Update discipline design variables
      until $4 \to 2$: Discipline optimization has converged
    end for
    5: Initiate system optimizer
    repeat
      6: Compute all penalty functions
      7: Update system design variables and coupling targets
    until $7 \to 6$: System optimization has converged
    8: Update penalty weights.
  until $8 \to 1$: Penalty weights are large enough
```

Both EPD and IPD have mathematically provable convergence under the linear independence constraint qualification and with mild assumptions on the update strategy for the penalty weights [41].

- In particular, the penalty weight in IPD must monotonically increase until the inconsistency is sufficiently small, similar to other quadratic penalty methods [115].

- For EPD, the penalty weight must be larger than the largest Lagrange multiplier, following established theory of the L_1 penalty function [115], while the barrier parameter must monotonically decrease like in an interior point method [160].

- If other penalty functions are employed, the parameter values are selected and updated according to the corresponding mathematical theory.

Under these conditions, the solution obtained under EPD and IPD will also be a solution to the desired AAO MDO problem.

Only once in the literature has either penalty decomposition architecture been tested against any others.

- The results of Tosserams et al. [152] suggest that performance depends on the choice of penalty function employed.
- A comparison between IPD with a quadratic penalty function and IPD with an augmented Lagrangian penalty function showed that the latter significantly outperformed the former in terms of both time and number of function evaluations on several test problems.

4.7 MDO of Independent Subspaces (MDOIS)

If the problem contains no system-wide constraints or objectives, i.e., if neither f_0 and c_0 exist, and the problem does not include shared design variables, i.e., if x_0 does not exist, then the MDO of independent subspaces (MDOIS) architecture [137] applies. In this case, the discipline subproblems are fully separable (aside from the coupled state variables) and given by

$$\begin{aligned} \text{minimize} \quad & f_i\left(x_i, y_i\left(x_i, y^t_{j \neq i}\right)\right) \\ \text{with respect to} \quad & x_i \\ \text{subject to} \quad & c_i\left(x_i, y_i\left(x_i, y^t_{j \neq i}\right)\right) \geq 0. \end{aligned} \tag{16}$$

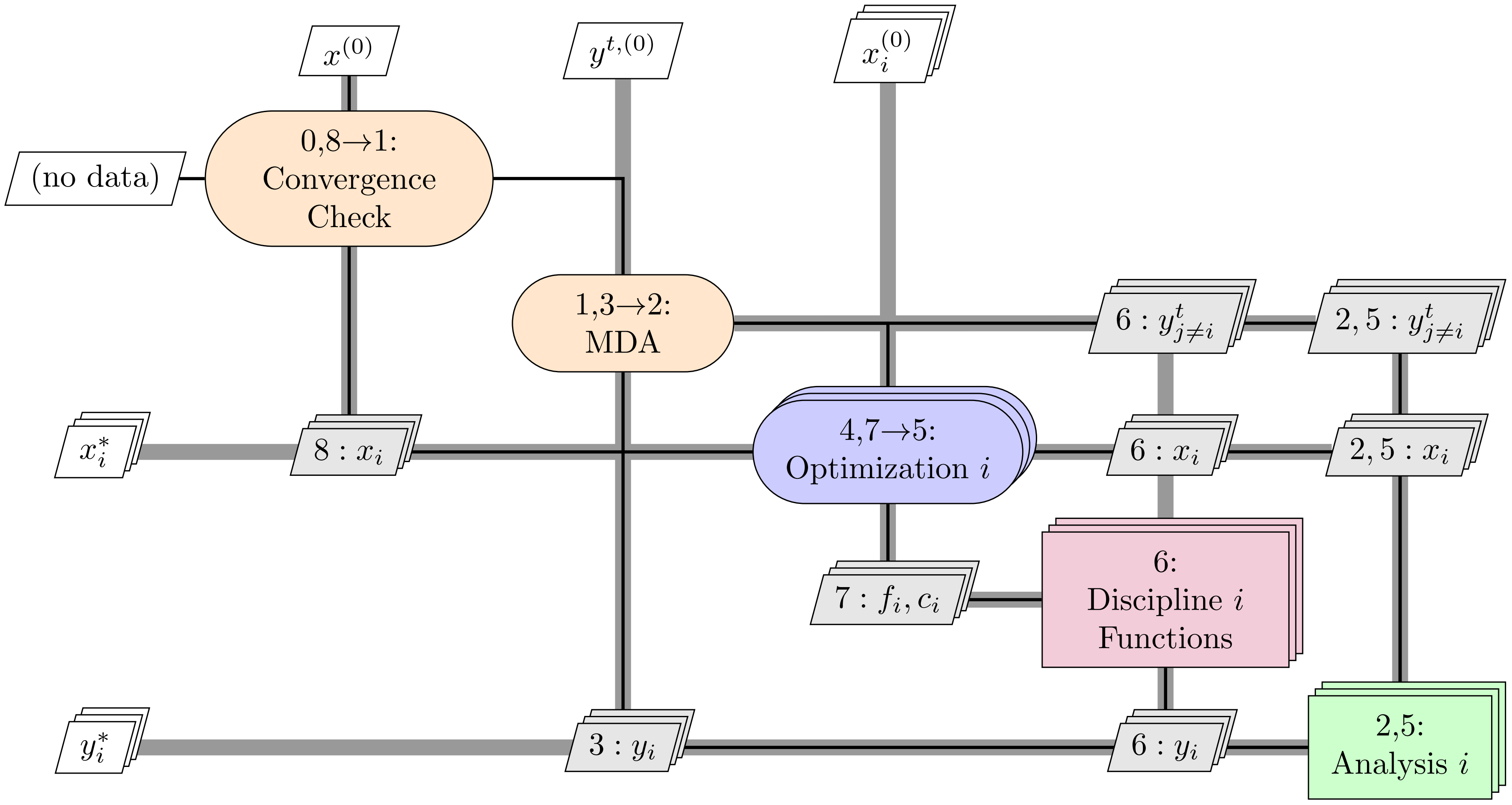


Figure 15: Diagram for the MDOIS architecture

Algorithm 9 MDOIS

```
Input: Initial design variables $x$
Output: Optimal variables $x^*$, objective function $f^*$, and constraint values $c^*$
  0: Initiate main iteration
  repeat
    repeat
      1: Initiate MDA
      2: Evaluate discipline analyses
      3: Update coupling variables
    until $3 \to 2$: MDA has converged
    for Each discipline $i$ do
      4: Initiate disciplinary optimization
      repeat
        5: Evaluate discipline analysis
        6: Compute discipline objectives and constraints
        7: Compute a new discipline design point
      until $7 \to 5$: Discipline optimization has converged
    end for
  until $8 \to 1$ Main iteration has converged
```

Some notes on MDOIS:

- The targets are just local copies of system state information.
- Upon solution of the disciplinary problems, which can access the output of individual disciplinary analysis codes, a full multidisciplinary analysis is completed to update all target values. Thus, rather than a system subproblem used by other architectures, the MDA is used to guide the disciplinary subproblems to a design solution.
- Shin and Park [137] show that under the given problem assumptions an optimal design is found using this architecture.

Benchmarking results are available comparing MDOIS to some of the older architectures. These results are given by Yi et al. [162].

- In many cases, MDOIS requires fewer analysis calls than MDF while still being able to reach the optimal solution.
- However, MDOIS still does not converge as fast as IDF and the restrictive

definition of the problem means that the architecture is not nearly as flexible as MDF.

- A practical problem that can be solved using MDOIS is the belt-integrated seat problem of Shin et al. [136]. However, the results using MDOIS have not been compared to results obtained using other architectures.

4.8 Quasiseparable Decomposition (QSD)

Haftka and Watson [59] developed the QSD architecture to solve *quasiseparable* optimization problems.

- In a quasiseparable problem, the system objective and constraint functions are assumed to be dependent only on global variables (i.e., the shared design and coupling variables).
- This type of problem may be thought of as identical to the complicating variables problems.

We have not come across any practical design problems that satisfy this property. However, if required by the problem, we can easily transform the general (AAO) MDO problem into a quasiseparable problem.

- This is accomplished by duplicating the relevant local variables, and forcing the global objective to depend on the target copies of local variables.
- The resulting quasiseparable problem and decomposition is mathematically equivalent to the original problem.

The system subproblem is given by

$$\begin{aligned}
\text{minimize} \quad & f_0\left(x_0, y^t\right) + \sum_{i=1}^{N} b_i \\
\text{with respect to} \quad & x_0, y^t, b \\
\text{subject to} \quad & c_0\left(x_0, y^t\right) \geq 0 \\
& s_i^*\left(x_0, x_i, y_i\left(x_0, x_i, y^t_{j\neq i}\right), b_i\right) \geq 0 \;\; \text{for}\, i = 1, \ldots, N.
\end{aligned} \tag{17}$$

where s_i is the constraint margin for discipline i and b_i is the “budget” assigned to each disciplinary objective. The discipline i subproblem becomes

$$\begin{aligned}
\text{minimize} \quad & -s_i \\
\text{with respect to} \quad & x_i, s_i \\
\text{subject to} \quad & c_i\left(x_0, x_i, y_i\left(x_0, x_i, y^t_{j\neq i}\right)\right) - s_i \geq 0 \\
& f_i\left(x_0, x_i, y_i\left(x_0, x_i, y^t_{j\neq i}\right)\right) - b_i - s_i \geq 0 \\
& y^t_i - y_i\left(x_0, x_i, y^t_{j\neq i}\right) = 0
\end{aligned} \tag{18}$$

where k is an element of the constraint vector c_i.

Algorithm 10 QSD

```
Input: Initial design variables $x$
Output: Optimal variables $x^*$, objective function $f^*$, and constraint values $c^*$
  0: Initiate system optimization
  repeat
    1: Compute system objectives and constraints
    for Each discipline $i$ do
      1.0: Initiate discipline optimization
      repeat
        1.1: Evaluate discipline analysis
        1.2: Compute discipline objective and constraints
        1.3: Update discipline design point
      until $1.3 \to 1.1$: Discipline optimization has converged
    end for
    2: Compute a new system design point
  until $2 \to 1$: System problem has converged
```

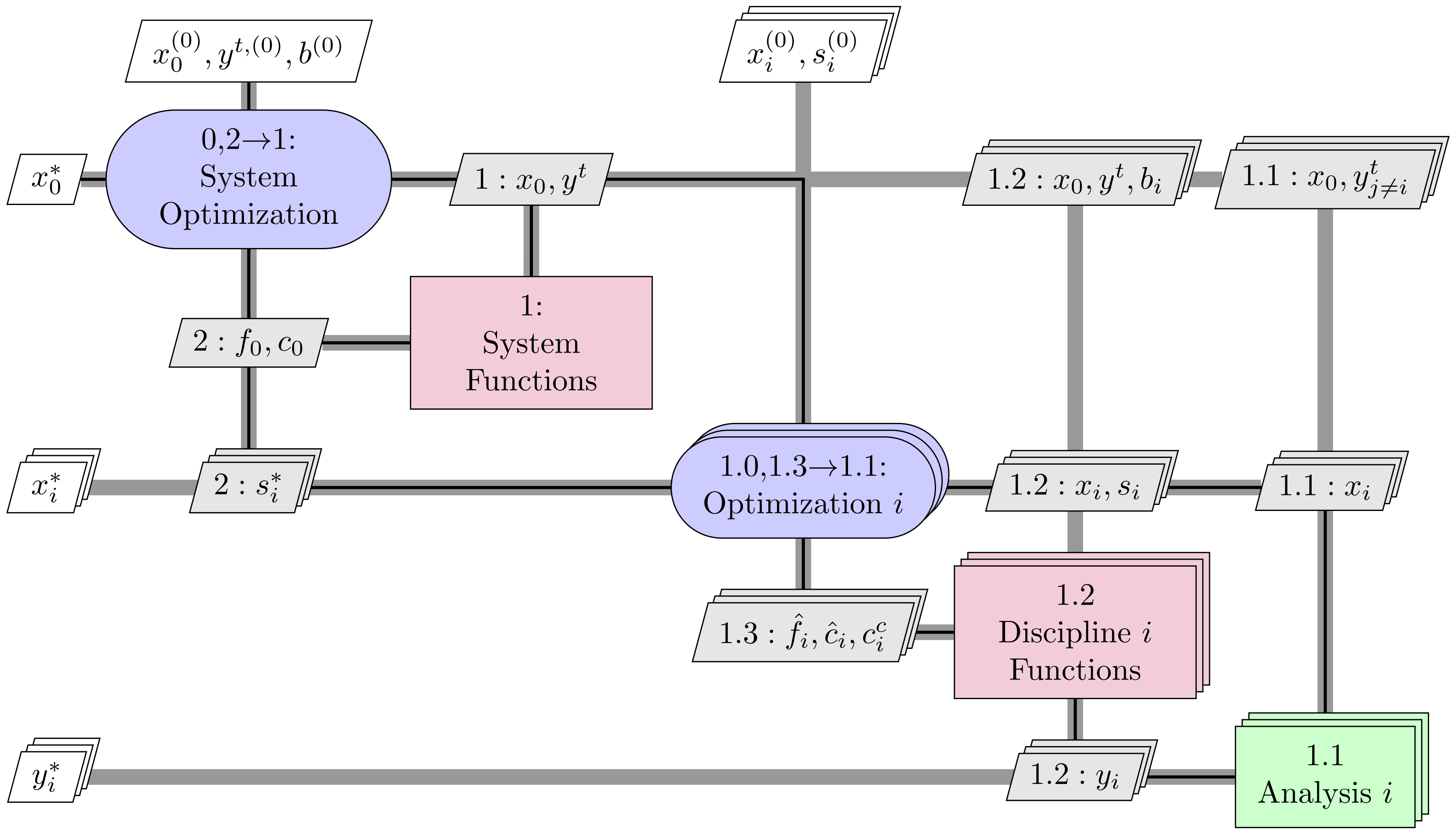

$x_0^{(0)}, y^{t,(0)}, b^{(0)}$
$x_i^{(0)}, s_i^{(0)}$
x_0^*
0,2→1: System Optimization
$1 : x_0, y^t$
$1.2 : x_0, y^t, b_i$
$1.1 : x_0, y^t_{j \neq i}$
$2 : f_0, c_0$
1: System Functions
x_i^*
$2 : s_i^*$
1.0,1.3→1.1: Optimization i
$1.2 : x_i, s_i$
$1.1 : x_i$
$1.3 : \hat{f}_i, \hat{c}_i, c_i^c$
1.2 Discipline i Functions
y_i^*
$1.2 : y_i$
1.1 Analysis i

Figure 16: Diagram for the QSD architecture

Some notes on QSD:

- Due to the use of target copies, we classify this architecture as distributed IDF.
- This is a bilevel architecture where the solutions of disciplinary subproblems are constraints in the system subproblem. Therefore, post-optimality sensitivities or surrogate model approximations of optimized disciplinary subproblems are required to solve the system subproblem.
- Haftka and Watson have extended the theory behind QSD to solve problems with a combination of discrete and continuous variables [60].
- Liu et al. [97] successfully applied QSD with surrogate models to a structural optimization problem. However, they made no comparison of the performance to other architectures, not even QSD without the surrogates.
- A version of QSD without surrogate models was benchmarked by de Wit and van Keulen [37]. Unfortunately, this architecture was the worst of all the architectures tested in terms of disciplinary evaluations.

A version of QSD using surrogate models should yield improved performance, due to the smoothness introduced by the model, but this version has not been benchmarked to our knowledge.

4.9 Asymmetric Subspace Optimization (ASO)

The ASO architecture [31] is a new distributed-MDF architecture.

- It was motivated by the case of high-fidelity aerostructural optimization, where the aerodynamic analysis typically requires an order of magnitude more time to complete than the structural analysis [105].
- To reduce the number of expensive aerodynamic analyses, the structural analysis is coupled with a structural optimization inside the MDA.
- This idea can be readily generalized to any problem where there is a wide discrepancy between discipline analysis times.

The system subproblem in ASO is

$$\begin{aligned} \text{minimize} \quad & f_0\left(x, y\left(x, y\right)\right) + \sum_k f_k\left(x_0, x_k, y_k\left(x_0, x_k, y_{j\neq k}\right)\right) \\ \text{with respect to} \quad & x_0, x_k \\ \text{subject to} \quad & c_0\left(x, y\left(x, y\right)\right) \geq 0 \\ & c_k\left(x_0, x_k, y_k\left(x_0, x_k, y_{j\neq k}\right)\right) \geq 0 \qquad \text{for all } k, \end{aligned} \tag{19}$$

where subscript k denotes disciplinary information that remains outside of the MDA. The disciplinary problem for discipline i, which is resolved inside the MDA, is

$$\begin{aligned} \text{minimize} \quad & f_0\left(x, y\left(x, y\right)\right) + f_i\left(x_0, x_i, y_i\left(x_0, x_i, y_{j\neq i}\right)\right) \\ \text{with respect to} \quad & x_i \\ \text{subject to} \quad & c_i\left(x_0, x_i, y_i\left(x_0, x_i, y_{j\neq i}\right)\right) \geq 0. \end{aligned} \tag{20}$$

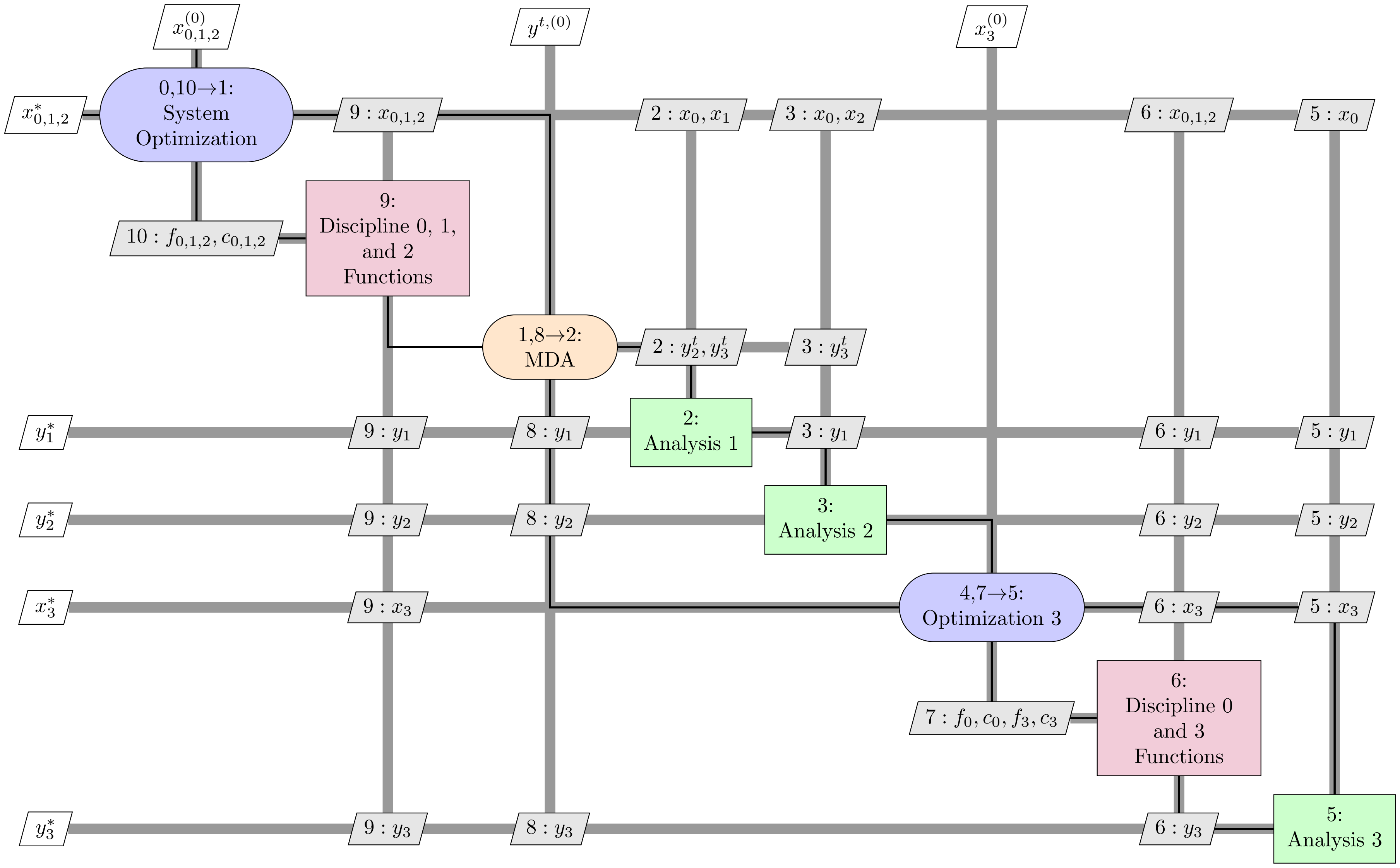


Figure 17: Diagram for the ASO architecture

Algorithm 11 ASO

```
Input: Initial design variables $x$
Output: Optimal variables $x^*$, objective function $f^*$, and constraint values $c^*$
  0: Initiate system optimization
  repeat
    1: Initiate MDA
    repeat
      2: Evaluate Analysis 1
      3: Evaluate Analysis 2
      4: Initiate optimization of Discipline 3
      repeat
        5: Evaluate Analysis 3
        6: Compute discipline 3 objectives and constraints
        7: Update local design variables
      until $7 \to 5$: Discipline 3 optimization has converged
      8: Update coupling variables
    until $8 \to 2$ MDA has converged
    9: Compute objective and constraint function values for all disciplines 1 and 2
    10: Update design variables
  until $10 \to 1$: System optimization has converged
```

Notes on ASO:

- The optimality of the final solution is preserved by using the coupled post-optimality sensitivity (CPOS) equations, developed by Chittick and Martins [31], to calculate gradients at the system level.
- CPOS represents the extension of the coupled sensitivity equations [141, 106] to include the optimality conditions.
- ASO was later implemented using a coupled-adjoint approach as well [30].
- Kennedy et al. [72] present alternative strategies for computing the disciplinary subproblem optima and the post-optimality sensitivity analysis.
- As with other bilevel MDO architectures, the post-optimality analysis is necessary to ensure convergence to an optimal design of the original monolithic problem.

Results of ASO show a substantial reduction in the number of calls to the aerodynamics analysis, and even a slight reduction in the number of calls to the structural analysis [31].

- However, the total time for the optimization routine is only competitive with MDF if the aerodynamic analysis is substantially more costly than the structural analysis.
- If the two analyses take roughly equal time, MDF is still much faster.
- Furthermore, the use of CPOS increases the complexity of the sensitivity analysis compared to a normal coupled adjoint, which adds to the overall solution time. As a result, this architecture may only appeal to practitioners solving MDO problems with widely varying computational cost in the discipline analysis.

References

[1] J. Agte, O. de Weck, J. Sobieszczanski-Sobieski, P. Arendsen, A. Morris, and M. Spieck. MDO: Assessment and Direction for Advancement — an Opinion of One International Group. *Structural and Multidisciplinary Optimization*, 40:17–33, 2010.

[2] J. Ahn and J. H. Kwon. An Efficient Strategy for Reliability-Based Multidisciplinary Design Optimization using BLISS. *Structural and Multidisciplinary Optimization*, 31:363–372, 2006.

[3] N. M. Alexandrov and R. M. Lewis. Comparative Properties of Collaborative Optimization and Other Approaches to MDO. In *1st ASMO UK/ISSMO Conference on Engineering Design Optimization*, 1999.

[4] N. M. Alexandrov and R. M. Lewis. Analytical and Computational Aspects of Collaborative Optimization for Multidisciplinary Design. *AIAA Journal*, 40(2):301–309, 2002.

[5] N. M. Alexandrov and R. M. Lewis. Reconfigurability in MDO Problem

Synthesis, Part 1. In *10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, number September, Albany, NY, 2004.

[6] J. T. Allison, M. Kokkolaras, and P. Y. Papalambros. On Selecting Single-Level Formulations for Complex System Design Optimization. *Journal of Mechanical Design*, 129(September):898–906, Sept. 2007.

[7] J. T. Allison, B. Roth, M. Kokkolaras, I. M. Kroo, and P. Y. Papalambros. Aircraft Family Design Using Decomposition-Based Methods. In *11th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, Sept. 2006.

[8] J. T. Allison, D. Walsh, M. Kokkolaras, P. Y. Papalambros, and M. Cartmell. Analytical Target Cascading in Aircraft Design. In *44th AIAA Aerospace Sciences Meeting*, 2006.

[9] R. Balling and M. R. Rawlings. Collaborative Optimization with Disciplinary Conceptual Design. *Structural and Multidisciplinary Optimization*, 20(3):232–241, Nov. 2000.

[10] R. J. Balling and J. Sobieszczanski-Sobieski. Optimization of Coupled

Systems: A Critical Overview of Approaches. *AIAA Journal*, 34(1):6–17, 1996.

[11] M. S. Bazaara, H. D. Sherali, and C. M. Shetty. *Nonlinear Programming: Theory and Algorithms*. John Wiley & Sons, 2006.

[12] J. F. Benders. Partitioning Procedures for Solving Mixed Variables Programming Problems. *Numerische Mathematik*, 4:238–252, 1962.

[13] G. Berkooz, P. Holmes, and J. L. Lumley. The Proper Orthogonal Decomposition in the Analysis of Turbulent Flows. *Annual Review of Fluid Mechanics*, 25:539–575, Jan. 1993.

[14] D. P. Bertsekas. *Constrained Optimization and Lagrange Multiplier Methods*. Athena Scientific, 1996.

[15] D. P. Bertsekas and J. N. Tsitsiklis. *Parallel and Distributed Computation: Numerical Methods*. Athena Scientific, 1997.

[16] L. T. Biegler, O. Ghattas, M. Heinkenschloss, and B. van Bloemen

Waanders, editors. *Large-Scale PDE-Constrained Optimization*. Springer-Verlag, 2003.

[17] C. Bloebaum. Coupling strength-based system reduction for complex engineering design. *Structural Optimization*, 10:113–121, 1995.

[18] C. L. Bloebaum, P. Hajela, and J. Sobieszczanski-Sobieski. Non-Hierarchic System Decomposition in Structural Optimization. *Engineering Optimization*, 19(3):171–186, 1992.

[19] V. Y. Blouin, G. M. Fadel, I. U. Haque, J. R. Wagner, and H. B. Samuels. Continuously Variable Transmission Design for Optimum Vehicle Performance by Analytical Target Cascading. *International Journal of Heavy Vehicle Systems*, 11:327–348, 2004.

[20] R. D. Braun. *Collaborative Optimization: An Architecture for Large-Scale Distributed Design*. PhD thesis, Stanford University, Stanford, CA 94305, 1996.

[21] R. D. Braun, P. Gage, I. M. Kroo, and I. P. Sobieski. Implementation and Performance Issues in Collaborative Optimization. In *6th AIAA, NASA,*

and ISSMO Symposium on Multidisciplinary Analysis and Optimization, 1996.

[22] R. D. Braun and I. M. Kroo. Development and Application of the Collaborative Optimization Architecture in a Multidisciplinary Design Environment. In N. Alexandrov and M. Y. Hussaini, editors, *Multidisciplinary Design Optimization: State-of-the-Art*, pages 98–116. SIAM, 1997.

[23] R. D. Braun, A. A. Moore, and I. M. Kroo. Collaborative Approach to Launch Vehicle Design. *Journal of Spacecraft and Rockets*, 34(4):478–486, July 1997.

[24] N. F. Brown and J. R. Olds. Evaluation of Multidisciplinary Optimization Techniques Applied to a Reusable Launch Vehicle. *Journal of Spacecraft and Rockets*, 43(6):1289–1300, 2006.

[25] T. R. Browning. Applying the Design Structure Matrix to System Decomposition and Integration Problems: A Review and New Directions. *IEEE Transactions on Engineering Management*, 48(3):292–306, 2001.

[26] I. A. Budianto and J. R. Olds. Design and Deployment of a Satellite Constellation Using Collaborative Optimization. *Journal of Spacecraft and Rockets*, 41(6):956–963, Nov. 2004.

[27] G. Cai, J. Fang, Y. Zheng, X. Tong, J. Chen, and J. Wang. Optimization of System Parameters for Liquid Rocket Engines with Gas-Generator Cycles. *Journal of Propulsion and Power*, 26(1):113–119, 2010.

[28] T.-Y. Chen. Calculation of the Move Limits for the Sequential Linear Programming Method. *International Journal for Numerical Methods in Engineering*, 36(15):2661–2679, Aug. 1993.

[29] Y. Chen, X. Chen, and Y. Lin. The Application of Analytical Target Cascading in Parallel Hybrid Electric Vehicle. In *IEEE Vehicle Power and Propulsion Conference*, pages 1602–1607, 2009.

[30] I. R. Chittick and J. R. R. A. Martins. Aero-Structural Optimization Using Adjoint Coupled Post-Optimality Sensitivities. *Structural and Multidisciplinary Optimization*, 36:59–70, 2008.

[31] I. R. Chittick and J. R. R. A. Martins. An Asymmetric Suboptimization

Approach to Aerostructural Optimization. *Optimization and Engineering*, 10(1):133–152, March 2009.

[32] R. Choudhary, A. Malkawi, and P. Y. Papalambros. Analytic Target Cascading in Simulation-based Building Design. *Automation in Construction*, 14(4):551–568, Aug. 2005.

[33] A. J. Conejo, F. J. Nogales, and F. J. Prieto. A Decomposition Procedure Based on Approximate Newton Directions. *Mathematical Programming*, 93:495–515, 2002.

[34] A. R. Conn, N. I. M. Gould, and P. L. Toint. *Trust Region Methods*. SIAM, Philadelphia, PA, 2000.

[35] E. J. Cramer, J. E. Dennis Jr, P. D. Frank, R. M. Lewis, and G. R. Shubin. Problem Formulation for Multidisciplinary Optimization. *SIAM Journal on Optimization*, 4(4):754–776, 1994.

[36] G. B. Dantzig and P. Wolfe. Decomposition Principle for Linear Programs. *Operations Research*, 8:101–111, 1960.

[37] A. J. de Wit and F. van Keulen. Numerical Comparison of Multilevel Optimization Techniques. In *Proceedings of the 48th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, Apr. 2007.

[38] A. J. de Wit and F. van Keulen. Overview of Methods for Multi-Level and/or Multi-Disciplinary Optimization. In *51st AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, number April, Orlando, FL, Apr. 2010.

[39] A.-V. DeMiguel. *Two Decomposition Algorithms for Nonconvex Optimization Problems with Global Variables*. PhD thesis, Stanford University, 2001.

[40] A.-V. DeMiguel and W. Murray. An Analysis of Collaborative Optimization Methods. In *8th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis & Optimization*, Long Beach, CA, 2000.

[41] V. DeMiguel and W. Murray. A Local Convergence Analysis of Bilevel Decomposition Algorithms. *Optimization and Engineering*, 7(2):99–133, June 2006.

[42] R. Enblom. Two-Level Numerical Optimization of Ride Comfort in Railway Vehicles. *Journal of Rail and Rapid Transit*, 220(1):1–11, Mar. 2006.

[43] L. F. P. Etman, M. Kokkolaras, A. T. Hofkamp, P. Y. Papalambros, and J. E. Rooda. Coordination Specification in Distributed Optimal Design of Multilevel Systems Using the χ Language. *Structural and Multidisciplinary Optimization*, 29:198–212, 2005.

[44] A. V. Fiacco and G. P. McCormick. *Nonlinear Programming: Sequential Unconstrained Minimization Techniques*. SIAM, 1990.

[45] R. Ganguli. Survey of Recent Developments in Rotorcraft Design Optimization. *Journal of Aircraft*, 41(3):493–510, 2004.

[46] C. Geethaikrishnan, P. M. Mujumdar, K. Sudhakar, and V. Adimurthy. A Hybrid MDO Architecture for Launch Vehicle Conceptual Design. In *51st AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, number April, Orlando, FL, Apr. 2010.

[47] P. Geyer. Component-oriented Decomposition for Multidisciplinary Design

Optimization in Building Design. *Advanced Engineering Informatics*, 23(1):12–31, 2009.

[48] P. E. Gill, W. Murray, and M. A. Saunders. SNOPT: An SQP Algorithm for Large-Scale Constrained Optimization. *SIAM Review*, 47(1):99–131, 2005.

[49] B. Glaz, P. P. Friedmann, and L. Liu. Helicopter Vibration Reduction throughout the Entire Flight Envelope Using Surrogate-Based Optimization. *Journal of the American Helicopter Society*, 54:12007—-1–15, 2009.

[50] J. Gray, K. T. Moore, and B. A. Naylor. OpenMDAO: An Open Source Framework for Multidisciplinary Analysis and Optimization. In *13th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, Fort Worth, TX, Sept. 2010.

[51] B. Grossman, Z. Gurdal, G. J. Strauch, W. M. Eppard, and R. T. Haftka. Integrated Aerodynamic/Structural Design of a Sailplane Wing. *Journal of Aircraft*, 25(9):855–860, 1988.

[52] B. Grossman, R.T.Haftka, P.-J. Kao, D.M.Polen, and M.Rais-Rohani. Integrated Aerodynamic-Structural Design of a Transport Wing. *Journal of Aircraft*, 27(12):1050–1056, 1990.

[53] X. Gu, J. E. Renaud, L. M. Ashe, S. M. Batill, A. S. Budhiraja, and L. J. Krajewski. Decision-Based Collaborative Optimization. *Journal of Mechanical Design*, 124(1):1–13, Mar. 2002.

[54] X. S. Gu, J. E. Renaud, and C. L. Penninger. Implicit Uncertainty Propagation for Robust Collaborative Optimization. *Journal of Mechanical Design*, 128(4):1001–1013, July 2006.

[55] R. T. Haftka. Automated Procedure for Design of Wing Structures to Satisfy Strength and Flutter Requirements. Technical Report TN D-7264, NASA Langley Research Center, Hampton, VA, 1973.

[56] R. T. Haftka. Optimization of Flexible Wing Structures Subject to Strength and Induced Drag Constraints. *AIAA Journal*, 14(8):1106–1977, 1977.

[57] R. T. Haftka. Simultaneous Analysis and Design. *AIAA Journal*, 23(7):1099–1103, July 1985.

[58] R. T. Haftka and C. P. Shore. Approximate Methods for Combined Thermal/Structural Design. Technical Report TP-1428, NASA, June 1979.

[59] R. T. Haftka and L. T. Watson. Multidisciplinary Design Optimization with Quasiseparable Subsystems. *Optimization and Engineering*, 6:9–20, 2005.

[60] R. T. Haftka and L. T. Watson. Decomposition Theory for Multidisciplinary Design Optimization Problems with Mixed Integer Quasiseparable Subsystems. *Optimization and Engineering*, 7(2):135–149, June 2006.

[61] J. Han and P. Y. Papalambros. A Note on the Convergence of Analytical Target Cascading with Infinite Norms. *Journal of Mechanical Design*, 132(3):034502—-1–6, Mar. 2010.

[62] J. Han and P. Y. Papalambros. A Sequential Linear Programming

Coordination Algorithm for Analytical Target Cascading. *Journal of Mechanical Design*, 132(3):021003—-1–8, Mar. 2010.

[63] J. Han and P. Y. Papalambros. Optimal Design of Hybrid Electric Fuel Cell Vehicles Under Uncertainty and Enterprise Considerations. *Journal Of Fuel Cell Science And Technology*, 7(2):021020—-1–9, Apr. 2010.

[64] Y. He and J. Mcphee. Multidisciplinary Optimization of Multibody Systems with Application to the Design of Rail Vehicles. *Multibody System Dynamics*, 14(2):111–135, 2005.

[65] C.-H. Huang, J. Galuski, and C. L. Bloebaum. Multi-Objective Pareto Concurrent Subspace Optimization for Multidisciplinary Design. *AIAA Journal*, 45(8):1894–1906, Aug. 2007.

[66] G. Q. Huang, T. Qu, D. W. L. Cheung, and L. Liang. Extensible Multi-Agent System for Optimal Design of Complex Systems using Analytical Target Cascading. *International Journal of Advanced Manufacturing Technology*, 30:917–926, 2006.

[67] G. Q. Huang and T. . A. Qu. Extending Analytical Target Cascading for

Optimal Configuration of Supply Chains with Alternative Autonomous Suppliers. *International Journal of Production Economics*, 115:39–54, 2008.

[68] H.-Z. Huang, Y. Tao, and Y. Liu. Multidisciplinary Collaborative Optimization using Fuzzy Satisfaction Degree and Fuzzy Sufficiency Degree Model. *Soft Computing*, 12:995–1005, 2008.

[69] K. F. Hulme and C. L. Bloebaum. A Simulation-Based Comparison of Multidisciplinary Design Optimization Solution Strategies using CASCADE. *Structural and Multidisciplinary Optimization*, 19:17–35, 2000.

[70] R. Kalavalapally, R. Penmetsa, and R. Grandhi. Multidisciplinary optimization of a lightweight torpedo structure subjected to an underwater explosion. *Finite Elements in Analysis and Design*, 43(2):103–111, December 2006.

[71] G. J. Kennedy and J. R. R. A. Martins. Parallel Solution Methods for Aerostructural Analysis and Design Optimization. In *Proceedings of the*

13th AIAA/ISSMO Multidisciplinary Analysis Optimization Conference, Forth Worth, TX, September 2010. AIAA 2010-9308.

[72] G. J. Kennedy, J. R. R. A. Martins, and J. S. Hansen. Aerostructural Optimization of Aircraft Structures Using Asymmetric Subspace Optimization. In *12th AIAA/ISSMO Multidisciplinary Analysis and Optimizaton Conference*, Victoria, BC, Canada, 2008. AIAA.

[73] H. Kim, S. Ragon, G. Soremekun, B. Malone, and J. Sobieszczanski-Sobieski. Flexible Approximation Model Approach for Bi-Level Integrated System Synthesis. In *10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, pages 1–11, Aug. 2004.

[74] H. M. Kim. *Target Cascading in Optimal System Design*. Phd thesis, University of Michigan, 2001.

[75] H. M. Kim, W. Chen, and M. M. Wiecek. Lagrangian Coordination for Enhancing the Convergence of Analytical Target Cascading. *AIAA Journal*, 44(10):2197–2207, Oct. 2006.

[76] H. M. Kim, M. Kokkolaras, L. S. Louca, G. J. Delagrammatikas, N. F. Michelena, Z. S. Filipi, P. Y. Papalambros, J. L. Stein, and D. N. Assanis. Target cascading in vehicle redesign: a class VI truck study. *International Journal of Vehicle Design*, 29(3):199–225, 2002.

[77] H. M. Kim, N. F. Michelena, P. Y. Papalambros, and T. Jiang. Target Cascading in Optimal System Design. *Journal of Mechanical Design*, 125(3):474–480, Sept. 2003.

[78] H. M. Kim, D. G. Rideout, P. Y. Papalambros, and J. L. Stein. Analytical Target Cascading in Automotive Vehicle Design. *Journal of Mechanical Design*, 125(3):481–490, Sept. 2003.

[79] S. Kodiyalam. Evaluation of Methods for Multidisciplinary Design Optimization ({MDO}), Part 1. \emph{{NASA} Report} CR-2000-210313, NASA, 1998.

[80] S. Kodiyalam, M. Kremenetsky, and S. Posey. Balanced HPC Infrastructure for CFD and Associated Multi-Discipline Simulations of Engineering Systems. *Journal of Aerospace Sciences and Technologies*, 61(3):434–443, 2009.

[81] S. Kodiyalam and J. Sobieszczanski-Sobieski. Bilevel Integrated System Synthesis with Response Surfaces. *AIAA Journal*, 38(8):1479–1485, Aug. 2000.

[82] S. Kodiyalam and J. Sobieszczanski-Sobieski. Multidisciplinary Design Optimization - Some Formal Methods, Framework Requirements, and Application to Vehicle Design. *International Journal of Vehicle Design*, 25:3–22, 2001.

[83] S. Kodiyalam and C. Yuan. Evaluation of Methods for Multidisciplinary Design Optimization ({MDO}), Part 2. \emph{{NASA} Report} CR-2000-210313, NASA, Nov. 2000.

[84] M. Kokkolaras, R. Fellini, H. M. Kim, N. F. Michelena, and P. Y. Papalambros. Extension of the Target Cascading Formulation to the Design of Product Families. *Structural and Multidisciplinary Optimization*, 24:293–301, 2002.

[85] M. Kokkolaras, L. S. Louca, G. J. Delagrammatikas, N. F. Michelena, Z. S. Filipi, P. Y. Papalambros, J. L. Stein, and D. N. Assanis. Simulation-based Optimal Design of Heavy Trucks by Model-based

Decomposition: an Extensive Analytical Target Cascading Case Study. *International Journal of Heavy Vehicle Systems*, 11:403–433, 2004.

[86] M. Kokkolaras, Z. P. Mourelatos, and P. Y. Papalambros. Design Optimization of Hierarchically Decomposed Multilevel Systems Under Uncertainty. *Journal of Mechanical Design*, 128(2):503–508, Mar. 2006.

[87] I. M. Kroo. {MDO} for Large-Scale Design. In N. Alexandrov and M. Y. Hussaini, editors, *Multidisciplinary Design Optimization: State-of-the-Art*, pages 22–44. SIAM, 1997.

[88] I. M. Kroo, S. Altus, R. Braun, P. Gage, and I. Sobieski. Multidisciplinary Optimization Methods for Aircraft Preliminary Design. In *5th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, 1994.

[89] A. B. Lambe and J. R. R. A. Martins. Extensions to the Design Structure Matrix for the Description of Multidisciplinary Design, Analysis, and Optimization Processes. *Structural and Multidisciplinary Optimization*, 2012. (In press).

[90] L. S. Lasdon. *Optimization Theory for Large Systems*. The Macmillan Company, 1970.

[91] L. A. Ledsinger and J. R. Olds. Optimized Solutions for Kistler K-1 Branching Trajectories Using Multidisciplinary Design Optimization Techniques. *Journal of Spacecraft and Rockets*, 39(3):420–429, May 2002.

[92] P. A. Legresley and J. J. Alonso. Improving the Performance of Design Decomposition Methods with POD. In *Proceedings of the 10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, Aug. 2004.

[93] X. Li, W. Li, and C. Liu. Geometric Analysis of Collaborative Optimization. *Structural and Multidisciplinary Optimization*, 35:301–313, 2008.

[94] Y. Li, Z. Lu, and J. J. Michalek. Diagonal Quadratic Approximation for Parallelization of Analytical Target Cascading. *Journal of Mechanical Design*, 130(5):051402—-1–11, May 2008.

[95] Z. Li, M. Kokkolaras, P. Y. Papalambros, and S. J. Hu. Product and Process Tolerance Allocation in Multistation Compliant Assembly Using Analytical Target Cascading. *Journal of Mechanical Design*, 130(9):091701—-1–9, Sept. 2008.

[96] J. G. Lin. Analysis and Enhancement of Collaborative Optimization for Multidisciplinary Design. *AIAA Journal*, 42(2):348–360, Feb. 2004.

[97] B. Liu, R. T. Haftka, and L. T. Watson. Global-Local Structural Optimization Using Response Surfaces of Local Optimization Margins. *Structural and Multidisciplinary Optimization*, 27(5):352–359, July 2004.

[98] H. Liu, W. Chen, M. Kokkolaras, P. Y. Papalambros, and H. M. Kim. Probabilistic Analytical Target Cascading: A Moment Matching Formulation for Multilevel Optimization Under Uncertainty. *Journal of Mechanical Design*, 128(4):991–1000, July 2006.

[99] E. Livne. Integrated Aeroservoelastic Optimization: Status and Direction. *Journal of Aircraft*, 36(1):122–145, 1999.

[100] E. Livne, L. Schmit, and P. Friedmann. Towards Integrated

Multidisciplinary Synthesis of Actively Controlled Fiber Composite Wings. *Journal of Aircraft*, 27(12):979–992, December 1990.

[101] C. A. Mader, J. R. R. A. Martins, J. J. Alonso, and E. van der Weide. ADjoint: An Approach for the Rapid Development of Discrete Adjoint Solvers. *AIAA Journal*, 46(4):863–873, April 2008.

[102] V. M. Manning. *Large-Scale Design of Supersonic Aircraft via Collaborative Optimization*. PhD thesis, Stanford University, 1999.

[103] C. Marriage. *Automatic Implementation of Multidisciplinary Design Optimization Architectures Using πMDO*. Master's thesis, University of Toronto, 2008.

[104] C. J. Marriage and J. R. R. A. Martins. Reconfigurable Semi-Analytic Sensitivity Methods and MDO Architectures within the πMDO Framework. In *Proceedings of the 12th AIAA/ISSMO Multidisciplinary Analysis and Optimizaton Conference*, Victoria, BC, Canada, Sept. 2008.

[105] J. R. R. A. Martins, J. J. Alonso, and J. J. Reuther. High-Fidelity

Aerostructural Design Optimization of a Supersonic Business Jet. *Journal of Aircraft*, 41(3):523–530, 2004.

[106] J. R. R. A. Martins, J. J. Alonso, and J. J. Reuther. A Coupled-Adjoint Sensitivity Analysis Method for High-Fidelity Aero-Structural Design. *Optimization and Engineering*, 6(1):33–62, March 2005.

[107] J. R. R. A. Martins, C. Marriage, and N. Tedford. pyMDO: An Object-Oriented Framework for Multidisciplinary Design Optimization. *ACM Transactions on Mathematical Software*, 36(4):20:1–25, 2009.

[108] J. R. R. A. Martins, P. Sturdza, and J. J. Alonso. The Complex-Step Derivative Approximation. *ACM Transactions on Mathematical Software*, 29(3):245–262, 2003.

[109] C. D. McAllister and T. W. Simpson. Multidisciplinary Robust Design Optimization of an Internal Combustion Engine. *Journal of Mechanical Design*, 125(1):124–130, Mar. 2003.

[110] C. D. McAllister, T. W. Simpson, K. Hacker, K. Lewis, and A. Messac. Integrating Linear Physical Programming within Collaborative

Optimization for Multiobjective Multidisciplinary Design Optimization. *Structural and Multidisciplinary Optimization*, 29(3):178–189, Mar. 2005.

[111] J. J. Michalek, F. M. Feinberg, and P. Y. Papalambros. Linking Marketing and Engineering Product Design Decisions via Analytical Target Cascading. *Journal of Product Innovation Management*, 22(1):42–62, 2005.

[112] J. J. Michalek and P. Y. Papalambros. An Efficient Weighting Update Method to Achieve Acceptable Consistency Deviation in Analytical Target Cascading. *Journal of Mechanical Design*, 127(March):206–214, Mar. 2005.

[113] J. J. Michalek and P. Y. Papalambros. BB-ATC: Analytical Target Cascading using Branch and Bound for Mixed Integer Nonlinear Programming. In *Proceedings of the ASME Design Engineering Technical Conference*, 2006.

[114] N. F. Michelena, H. Park, and P. Y. Papalambros. Convergence Properties of Analytical Target Cascading. *AIAA Journal*, 41(5):897–905, May 2003.

[115] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer-Verlag, 2nd edition, 2006.

[116] S. L. Padula, N. Alexandrov, and L. L. Green. MDO Test Suite at NASA Langley Research Center. In *Proceedings of the 6th AIAA/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, Bellevue,˜WA, 1996.

[117] S. L. Padula and R. E. Gillian. Multidisciplinary Environments: A History of Engineering Framework Development. In *11th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, Porthsmouth, VA, Sept. 2006.

[118] S. Parashar and C. L. Bloebaum. Multi-objective genetic algorithm concurrent subspace optimization (MOGACSSO) for Multidisciplinary design. In *11th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, May 2006.

[119] S. Parashar and C. L. Bloebaum. Robust Multi-Objective Genetic Algorithm Concurrent Subspace Optimization (R-MOGACSSO) for

Multidisciplinary Design. In *11th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, Sept. 2006.

[120] R. E. Perez. *A Multidisciplinary Optimization Framework for Flight Dynamics and Control Integration in Aircraft Design*. PhD thesis, University of Toronto, 2007.

[121] R. E. Perez, H. H. T. Liu, and K. Behdinan. Evaluation of Multidisciplinary Optimization Approaches for Aircraft Conceptual Design. In *10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, number September, Albany, NY, Aug. 2004.

[122] R. E. Perez, H. H. T. Liu, and K. Behdinan. Multidisciplinary Optimization Framework for Control-Configuration Integration in Aircraft Conceptual Design. *Journal of Aircraft*, 43(6):1937–1948, Nov. 2006.

[123] D. Peri and E. F. Campana. Multidisciplinary Design Optimization of a Naval Surface Combatant. *Journal of Ship Research*, 47(1):1–12, 2003.

[124] B. Potsaid, Y. Bellouard, and J. T.-Y. Wen. A Multidisciplinary Design and Optimization Methodology for the Adaptive Scanning Optical

Microscope (ASOM). *Proceedings of the SPIE*, 6289(May 2010):62890L1–12, 2006.

[125] J. E. Renaud and G. A. Gabriele. Improved Coordination in Nonhierarchic System Optimization. *AIAA Journal*, 31(12):2367–2373, Dec. 1993.

[126] J. E. Renaud and G. A. Gabriele. Approximation in Non-Hierarchic System Optimization. *AIAA Journal*, 32(1):198–205, 1994.

[127] T. D. Robinson, K. E. Willcox, M. S. Eldred, and R. Haimes. Multifidelity Optimization for Variable Complexity Design. In *11th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, 2006.

[128] B. Roth and I. Kroo. Enhanced Collaborative Optimization: Application to an Analytic Test Problem and Aircraft Design. In *12th AIAA/ISSMO Multidisciplinary Analysis and Optimizaton Conference*, number September, Victoria, British Columbia, Canada, Sept. 2008.

[129] B. D. Roth. *Aircraft Family Design using Enhanced Collaborative Optimization*. phdthesis, Stanford University, 2008.

[130] A. Ruszczynski. On Convergence of an Augmented Lagrangian Decomposition Method for Sparse Convex Optimization. *Mathematics of Operations Research*, 20(3):634–656, 1995.

[131] L. A. Schmit. Structural Design by Systematic Synthesis. In *2nd Conference on Electronic Computation*, pages 105–132, New York, NY, 1960. ASCE.

[132] L. A. Schmit and W. A. Thornton. Synthesis of an Airfoil at Supersonic Mach Number. Technical Report CR 144, NASA, Jan. 1965.

[133] L. A. Schmit, Jr. Structural synthesis — precursor and catalyst. Recent experiences in multidisciplinary analysis and optimization. Technical Report CP-2337, NASA, 1984.

[134] L. A. Schmit Jr. and R. K. Ramanathan. Multilevel approach to minimum weight design including buckling constraints. *AIAA Journal*, 16(2):97–104, Feb. 1978.

[135] R. S. Sellar, S. M. Batill, and J. E. Renaud. Response Surface Based,

Concurrent Subspace Optimization for Multidisciplinary System Design. In *34th AIAA Aerospace Sciences and Meeting Exhibit*, 1996.

[136] M. K. Shin, B. S. Kang, and G. J. Park. Application of the Multidisciplinary Design Optimization Algorithm to the Design of a Belt-Integrated Seat while Considering Crashworthiness. *Journal of Automobile Engineering*, 219(11):1281–1292, Nov. 2005.

[137] M.-K. Shin and G.-J. Park. Multidisciplinary Design Optimization based on Independent Subspaces. *International Journal for Numerical Methods in Engineering*, 64:599–617, 2005.

[138] I. P. Sobieski and I. M. Kroo. Collaborative Optimization Using Response Surface Estimation. *AIAA Journal*, 38(10):1931–1938, 2000.

[139] J. Sobieszczanski-Sobieski. Optimization by Decomposition: a Step from Hierarchic to Non-Hierarchic Systems. Technical Report September, NASA Langley Research Center, Hampton, VA, 1988.

[140] J. Sobieszczanski-Sobieski. Sensitivity of Complex, Internally Coupled Systems. *AIAA Journal*, 28(1):153–160, Apr. 1990.

[141] J. Sobieszczanski-Sobieski. Sensitivity of Complex, Internally Coupled Systems. *AIAA Journal*, 28(1):153–160, 1990.

[142] J. Sobieszczanski-Sobieski. Integrated System-of-Systems Synthesis. *AIAA Journal*, 46(5):1072–1080, May 2008.

[143] J. Sobieszczanski-Sobieski, J. S. Agte, and R. R. Sandusky Jr. Bilevel Integrated System Synthesis. *AIAA Journal*, 38(1):164–172, Jan. 2000.

[144] J. Sobieszczanski-Sobieski, T. D. Altus, M. Phillips, and R. R. Sandusky Jr. Bilevel Integrated System Synthesis for Concurrent and Distributed Processing. *AIAA Journal*, 41(10):1996–2003, Oct. 2003.

[145] J. Sobieszczanski-Sobieski and R. T. Haftka. Multidisciplinary Aerospace Design Optimization: Survey of Recent Developments. *Structural Optimization*, 14(1):1–23, 1997.

[146] D. V. Steward. The Design Structure Matrix: A Method for Managing the Design of Complex Systems. *IEEE Transactions on Engineering Management*, 28:71–74, 1981.

[147] J. Tajima, F. Momiyama, and N. Yuhara. A New Solution for Two-Bag Air Suspension System with Leaf Spring for Heavy-Duty Vehicle. *Vehicle System Dynamics*, 44(2):107–138, Feb. 2006.

[148] R. V. Tappeta, S. Nagendra, and J. E. Renaud. A Multidisciplinary Design Optimization Approach for High Temperature Aircraft Engine Components. *Structural Optimization*, 18(2-3):134–145, Oct. 1999.

[149] R. V. Tappeta and J. E. Renaud. Multiobjective Collaborative Optimization. *Journal of Mechanical Design*, 119:403–411, 1997.

[150] N. P. Tedford and J. R. R. A. Martins. Benchmarking Multidisciplinary Design Optimization Algorithms. *Optimization and Engineering*, 11:159–183, 2010.

[151] S. Tosserams, L. F. P. Etman, P. Y. Papalambros, and J. E. Rooda. An Augmented {Lagrangian} Relaxation for Analytical Target Cascading using the Alternating Direction Method of Multipliers. *Structural and Multidisciplinary Optimization*, 31(3):176–189, Mar. 2006.

[152] S. Tosserams, L. F. P. Etman, and J. E. Rooda. An Augmented

{Lagrangian} Decomposition Method for Quasiseparable Problems in {MDO}. *Structural and Multidisciplinary Optimization*, 34:211–227, 2007.

[153] S. Tosserams, L. F. P. Etman, and J. E. Rooda. Augmented {Lagrangian} Coordination for Distributed Optimal Design in {MDO}. *International Journal for Numerical Methods in Engineering*, 73:1885–1910, 2008.

[154] S. Tosserams, L. F. P. Etman, and J. E. Rooda. A Classification of Methods for Distributed System Optimization based on Formulation Structure. *Structural and Multidisciplinary Optimization*, 39(5):503–517, 2009.

[155] S. Tosserams, L. F. P. Etman, and J. E. Rooda. Block-Separable Linking Constraints in Augmented Lagrangian Coordination in {MDO}. *Structural and Multidisciplinary Optimization*, 37(5):521–527, 2009.

[156] S. Tosserams, L. F. P. Etman, and J. E. Rooda. A Micro-accelerometer {MDO} Benchmark Problem. *Structural and Multidisciplinary Optimization*, 41(2):255–275, 2010.

[157] S. Tosserams, L. F. P. Etman, and J. E. Rooda. Multi–Modality in Augmented Lagrangian Coordination for Distributed Optimal Design. *Structural and Multidisciplinary Optimization*, 40:329–352, 2010.

[158] S. Tosserams, M. Kokkolaras, L. F. P. Etman, and J. E. Rooda. A Nonhierarchical Formulation of Analytical Target Cascading. *Journal of Mechanical Design*, 132(5):051002—-1–13, May 2010.

[159] G. N. Vanderplaats. An Efficient Feasible Directions Algorithm for Design Synthesis. *AIAA Journal*, 22(11):1633–1640, Nov. 1984.

[160] S. J. Wright. *Primal-Dual Interior Point Methods*. SIAM, 1997.

[161] B. A. Wujek, J. E. Renaud, S. M. Batill, and J. B. Brockman. Concurrent Subspace Optimization Using Design Variable Sharing in a Distributed Computing Environment. *Concurrent Engineering: Research and Applications*, 4(4):361–377, 1996.

[162] S. I. Yi, J. K. Shin, and G. J. Park. Comparison of MDO Methods with Mathematical Examples. *Structural and Multidisciplinary Optimization*, 39:391–402, 2008.

[163] P. M. Zadeh and V. V. Toropov. Multi-Fidelity Multidisciplinary Design Optimization based on Collaborative Optimization Framework. In *9th AIAA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, number September, Atlanta, GA, Sept. 2002.

[164] P. M. Zadeh, V. V. Toropov, and A. S. Wood. Metamodel-based Collaborative Optimization Framework. *Structural and Multidisciplinary Optimization*, 38:103–115, 2009.

[165] K.-S. Zhang, Z.-H. Han, W.-J. Li, and W.-P. Song. Bilevel Adaptive Weighted Sum Method for Multidisciplinary Multi-Objective Optimization. *AIAA Journal*, 46(10):2611–2622, Oct. 2008.