Structural and Multidisciplinary Optimization manuscript No.
(will be inserted by the editor)

Differential Geometry Tools for Multidisciplinary Design Optimization, Part II: Application to QSD

Craig Bakker · Geoffrey T. Parks

Received: date / Accepted: date

Abstract Having previously developed a differential geometry framework for analyzing and conceptualizing Multidisciplinary Design Optimization (MDO) problems and methods, we now apply that framework to consider the Quasi-Separable Decomposition (QSD) architecture. Based on our theoretical investigations, we predict that QSD will fail to return feasible designs for MDO problems. In the same vein, we analyze the Individual Discipline Feasible (IDF) architecture, predict that IDF will converge to feasible designs, and propose a modified version of QSD which we believe will also output feasible design points. To test these predictions, we run all three architectures on a well-known analytical MDO problem. Our predictions regarding feasibility prove to be accurate: QSD does not return any feasible points, whereas all of the final design points from IDF and the modified QSD are feasible. Now that convergence to feasibility has been established, the next step is to investigate the optimization performance of various QSD modifications.

Keywords Multidisciplinary Design Optimization · Differential Geometry · Design Analysis

1 Introduction

In Part I of this paper, we developed a differential geometry (DG) framework for analyzing and developing Multidisciplinary Design Optimization (MDO) solution methods: we outlined the underlying theory and then translated MDO into that language of that theory. Having done this, we highlighted the analytical work that has already been done in MDO and discussed our past, present and future uses of our framework elsewhere. Finally, we promised to provide an example of how DG can be used to investigate and improve current MDO solution methods. In Part II, we now provide that example through our analysis of the Quasi-Separable Decomposition (QSD) architecture. As mentioned in Part I, Hafkta and Watson (2005) proved solution equivalence between the original and decomposed problems. Using a qualitative analysis of the various manifolds involved in the QSD decomposition, however, we will show that their proof fails in the presence of state equations: given the presence of state equations, QSD will not necessarily return a feasible solution, let alone one which is optimal for the original problem. We will then compare its performance with an architecture which we predict will converge and produce a convergent version of QSD, Modified QSD (MQSD).

2 The Quasi-Separable Decomposition

QSD was first proposed by Haftka and Watson (2005). They began from this quasi-separable constrained optimization formulation

$$\min f_0(\mathbf{z}) + \sum_i f_i\left(\mathbf{x}_{(i)}, \mathbf{z}\right) \tag{1}$$

$$\mathbf{g}_{(0)}(\mathbf{z}) \leq \mathbf{0} \tag{2}$$

$$\mathbf{g}_{(i)}\left(\mathbf{z}, \mathbf{x}_{(i)}\right) \leq \mathbf{0}\ ,\ i = 1, 2, \ldots \tag{3}$$

where $\mathbf{z}$ are the global design variables, $\mathbf{x}_{(i)}$ are the disciplinary design variables for discipline i, f_0 is the global objective function, the f_i's are the disciplinary objective

C. Bakker · G.T. Parks
Department of Engineering
University of Cambridge
Cambridge, United Kingdom
CB2 1PZ E-mail: ckrb2@cam.ac.uk

functions, $\mathbf{g}_{(0)}$ are the global design constraints, and $\mathbf{g}_{(i)}$ are the disciplinary design constraints. There may be explicit bounds on the design variables, but as they are not significant for the decomposition analysis, we omit them for brevity's sake. The decomposed problem has the system-level objective function and constraints

$$\min_{\mathbf{z},\mathbf{b}} f_0\left(\mathbf{z}\right) + \sum_i b^i \tag{4}$$

$$\mathbf{g}_{(0)}\left(\mathbf{z}\right) \leq \mathbf{0} \tag{5}$$

$$\mathbf{s}\left(\mathbf{z},\mathbf{b}\right) \leq \mathbf{0} \tag{6}$$

and the subsystem optimization, for subsystem i, is

$$\min_{\mathbf{x}_{(i)},s^i} s^i \tag{7}$$

$$\mathbf{g}_{(i)}\left(\mathbf{z},\mathbf{x}_{(i)}\right) - s^i \leq \mathbf{0} \tag{8}$$

$$f_i\left(\mathbf{x}_{(i)},\mathbf{z}\right) - b^i - s^i \leq 0 \tag{9}$$

where s^i is the constraint margin for discipline i and b^i is the budget for the objective function of discipline i. In (8), the combination of the vector $\mathbf{g}_{(i)}$ and the scalar s^i is simply a shorthand way of showing that the same quantity s^i is subtracted from each element of $\mathbf{g}_{(i)}$, and the inequality holds element-wise. The explicit design variable constraints are, again, implied but omitted. As Haftka and Watson (2005) observe, it is not necessary to put a constraint margin (i.e. an s^i) on these constraints. In their original formulation, the minimization of the subsystem-level objective function is only with respect to $\mathbf{x}_{(i)}$, not with respect to s^i as well, but it is necessary that the constraint margins vary, to do the optimization, so this discrepancy is not particularly important; in their formulation, Martins and Lambe (2013) have the subsystem optimization with respect to the constraint margins as we have described it here.

In this formulation, there are no state equations, state variables, or equality constraints. Assuming convexity, Haftka and Watson (2005) prove that the (optimal) solutions to the original and decomposed problem correspond to each other. They also note that the constraint margin variables, the s^i's, will generally not be smooth, though they may be piecewise smooth or piecewise analytic. Calculating their values in the system-level optimization in practice, moreover, may require the use of response surfaces, as their values will depend on the results of each subsystem optimization. Irrespective of how those constraint margins are calculated, they are a key part of the architecture – the solution method hinges on satisfying inequality constraints. Later, Haftka and Watson extend QSD and their analysis to include discrete variables and provide some similar analysis (Haftka and Watson 2006).

A full MDO formulation (i.e. one with state variables and state equations) of QSD is given by Martins and Lambe (2013); they also note that, in practice, all practical design problems are quasi-separable – even if a problem were not quasi-separable, it could easily be transformed into a quasi-separable problem through duplicating the necessary variables. The differences between the formulation of Haftka and Watson and that of a full MDO version can be seen by comparing (1)-(9) with (10)-(20). The paper by Martins and Lambe also has different signs or directionality for the constraint margins than does the original by Haftka and Watson: Haftka and Watson use $\leq$, whereas Martins and Lambe use $\geq$, and this correspondingly affects how the constraint margins appear in the formulation elsewhere (e.g. the sign of the constraint margin and thus whether it is minimized or maximized). Here, we will keep the directionality used by Haftka and Watson.

QSD has not been implemented very often, however. In their paper, de Wit and van Keulen (2007) evaluate QSD alongside a number of other MDO architectures on a two-bar truss problem. QSD converged but was one of the more expensive architectures. We also note, as will become meaningful following our analysis, that the optimization problem used there had state variables but no coupled state equations (i.e. the state variables were not coupled to each other through equality constraints); the only constraints on the problem were inequalities.

3 Analyzing, comparing, and fixing QSD

In Section 3 of Part I, we described MDO as consisting of optimization problems on manifolds – i.e. the constrained optimization in $\Re^n$ becomes an unconstrained optimization on a Riemannian manifold; any inequality constraints present define the boundaries of that manifold. We can do the same thing with the various optimization processes in QSD. Although we could use parts of the manifold structure like the metric tensor to aid our investigation into QSD, for the purposes of our analysis here, it will be sufficient to identify the relevant manifolds, discuss how they relate to each other, and describe how the optimizations move along their respective manifolds. Additional analysis with the metric tensor, for example, is informative but prohibitively lengthy, and as it is not necessary for the conclusions we have drawn here, we have not included it. The different options available for calculating the approximation to $\mathbf{s}$ in the system-level problem (Martins and Lambe 2013) would also complicate the details of such analysis – the precise nature of the system-level properties (such as its metric tensor and optimality conditions) would vary depending on how the approximation was performed.

Consider an illustration of the reasoning used in our qualitative analysis with manifolds: if an architecture has an optimization process with an associated manifold, M_{opt}, which does not intersect the feasible design manifold, M_{feas}, then that optimization process *cannot* return a feasible point – the process is confined to M_{opt}. However, if M_{opt} and M_{feas} intersect, then the optimization process *may* return a feasible point; the feasibility of the returned point will depend on how the process moves along M_{opt}. Finally, if M_{opt} is a submanifold of M_{feas} (or is equivalent to M_{feas}), then the optimization process *must* return a feasible point.

As discussed in Section 3 of Part I, manifolds can be seen as a set of points in a Euclidean space, but manifolds have more structure than a set of points will, in general. This extra structure is helpful in several ways: it provides a set of coordinates as a reference (which are not necessarily part of a set); it aids in the conception and visualization of the problem to think of the relevant collections of points as manifolds rather than just as sets (i.e. the manifolds are simply like higher-dimensional versions of surfaces, but point sets do not have a similarly concrete conceptual analogy); and the intrinsic and extrinsic manifold perspectives, which do not exist for sets as such, provide two different, complementary ways of considering each manifold. Perhaps most importantly, though, part of the structure of a manifold is dimensionality – a set of points may have a dimension, but that dimensionality is not an integral property of their existence as a set. Dimensionality, however, helps us to see one aspect of the relationship between manifolds (especially manifolds and their submanifolds). For example, if a manifold has a submanifold of a lower dimension, and any point on the manifold is chosen at random, the probability of that point also being contained in the submanifold is effectively zero; more technically, the submanifold is a set of zero content on the manifold.

To aid in our analysis, we have included Table 1 showing variable denotations, meanings, and locations for those relevant quantities not discussed in Part I.

3.1 QSD

Here we will consider the QSD architecture as described by Martins and Lambe (2013) but with our notation and the constraint margin directionality of Haftka and Watson (2005). Using our DG tools and concepts, we will look at the manifolds and submanifolds involved in both the original and decomposed problem and thereby learn something about how this architecture functions. We begin by describing the original problem (for notational simplicity, we assume that each discipline has only one state variable, but this does not affect the analysis) with the formulation

$$\min f_0(\mathbf{z},\mathbf{y}) + \sum_i f_i\left(\mathbf{x}_{(i)},\mathbf{z},y^i\right) \tag{10}$$

$$\mathbf{h}(\mathbf{w},\mathbf{y}) = \mathbf{0} \tag{11}$$

$$\mathbf{g}_{(0)}(\mathbf{z},\mathbf{y}) \leq \mathbf{0} \tag{12}$$

$$\mathbf{g}_{(i)}\left(\mathbf{x}_{(i)},\mathbf{z},y^i\right) \leq \mathbf{0}\ ,\ i = 1,2,\ldots \tag{13}$$

The QSD decomposition is as follows: the system-level problem is

$$\min_{\mathbf{z},\mathbf{y},\mathbf{b}} f_0(\mathbf{z},\mathbf{y}) + \sum_i b^i \tag{14}$$

$$\mathbf{g}_{(0)}(\mathbf{z},\mathbf{y}) \leq \mathbf{0} \tag{15}$$

$$s^i\left(\mathbf{x}_{(i)},\mathbf{z},\psi^i\left(\mathbf{x}_{(i)},\mathbf{z},\tilde{\mathbf{y}}_{(i)}\right),b^i\right) \leq 0\ ,\ i = 1,2,\ldots \tag{16}$$

and the subsystem problem for subsystem i is

$$\min_{\mathbf{x}_{(i)},s^i} s^i \tag{17}$$

$$\mathbf{g}_{(i)}\left(\mathbf{x}_{(i)},\mathbf{z},y^i\right) - s^i \leq \mathbf{0} \tag{18}$$

$$f_i\left(\mathbf{x}_{(i)},\mathbf{z},y^i\right) - b^i - s^i \leq 0 \tag{19}$$

$$y^i = \psi^i\left(\mathbf{x}_{(i)},\mathbf{z},\tilde{\mathbf{y}}_{(i)}\right) \tag{20}$$

As before, in (18), the combination of the vector $\mathbf{g}_{(i)}$ and the scalar s^i is simply a shorthand way of showing that the same quantity s^i is subtracted from each element of $\mathbf{g}_{(i)}$, and the inequality holds element-wise. Using DG in an MDO context can often strain the standard notation conventions for superscripts and subscripts; see Part I or Ivancevic and Ivancevic (2007) for more on those conventions, and see Part I for variable descriptions not listed in Table 1.

The QSD optimization process consists of performing the subsystem optimizations and then doing the system-level optimization (Martins and Lambe 2013). We can now consider the pertinent manifolds for the decomposed problem:

1. $M_{tot} = \Re^{n+3(m-n)}$ with coordinates $(\mathbf{x},\mathbf{z},\mathbf{y},\mathbf{b},\mathbf{s})$.
2. $M_{sys}(\mathbf{x})$ is defined for each value of $\mathbf{x}$ by $\mathbf{s}(\mathbf{z},\mathbf{y},\mathbf{b})$; the local design variables are dropped from the $\mathbf{s}$ equations here to show that they are held constant during the system-level optimization and thus act as fixed parameters and not variables. $M_{sys}(\mathbf{x})$ is an $(n_z + 2(m-n))$-manifold, with ∂M_{sys} defined by (15) and (16), naturally embedded in M_{tot}.
3. Each $M_i\left(\mathbf{z},\tilde{\mathbf{y}}_{(i)},b^i\right)$ is defined by (18)-(20) with all variables except $\mathbf{x}_{(i)}$ and s^i are held constant. To be more precise, (20) defines M_i itself; (18) and (19) define ∂M_i. Each $M_i\left(\mathbf{z},\tilde{\mathbf{y}}_{(i)},b^i\right)$ is an (n_i+1)-manifold naturally embedded in $\Re^{n_i+2}$.

Table 1 Symbols used in Section 3

Symbol	Denotation	Location
f_0	System-level objective function	QSD, MQSD
f_i	Discipline i objective function	QSD, MQSD
$\mathbf{g}_0$	System-level constraints	QSD, MQSD
$\mathbf{g}_{(i)}$	Discipline i constraints	QSD, MQSD
s^i	Discipline i constraint margin	QSD, MQSD
b^i	Discipline i objective function budget	QSD, MQSD
$\boldsymbol{\sigma}$	Duplicate state variables	IDF, MQSD
M_{tot}	The space of all possible variable values	QSD, IDF, MQSD
M_{sys}	System-level optimization manifold	QSD, MQSD
M_i	Subsystem i optimization manifold	QSD, MQSD
n_i	Total design variables in discipline i	QSD, MQSD
M_{IDF}	IDF optimization manifold	IDF
$\phi(\cdot)$	Constraint enforcement function	IDF, MQSD

4. M_{feas} is defined by (11) with its boundary ∂M_{feas} defined by (12) and (13). As such, it is independent of $\mathbf{b}$ and $\mathbf{s}$.

We now consider all of these (parameterized) manifolds embedded in M_{tot} and make several observations:

1. At a given point of intersection, the M_i's are all mutually orthogonal, but M_{sys} is not orthogonal to the M_i's.
2. M_{sys} intersects M_{feas}, but neither is a submanifold of the other. The intersection $M_{sys}(\mathbf{x}) \cap M_{feas}$ is defined by $\mathbf{y} = \mathbf{y}(\mathbf{z})$ and $\mathbf{s} = \mathbf{s}(\mathbf{z}, \mathbf{y}(\mathbf{z}), \mathbf{b})$, where $\mathbf{y}(\mathbf{z})$ is determined by the solution of (11) with $\mathbf{x}$ held constant. Submanifolds of M_{feas} with constant $\mathbf{x}$ values are submanifolds of M_{sys}.
3. The M_i's may intersect M_{feas}, but neither is a submanifold of the other.

The main observation of this analysis is that, in the multidisciplinary case (i.e. with state equations), apart from the use of optimizer-enforced feasibility or a system-level multidisciplinary analysis – neither of which show up in the formulation given by Martins and Lambe (2013) – the architecture has no way to force the optimization either to stay on M_{feas} or to return to it in the event that the optimization leaves it.

In their paper, Haftka and Watson (2005) prove that solutions to the decomposed problem are equivalent to solutions to the original problem (both feasible and optimal solutions). Their formulation, however, had no state equations. If the qualitative analysis done above were re-done without state equations, it would show the system manifold to be a submanifold of the feasible design space, and thus the optimization would never leave the feasible design space; correspondingly, each subsystem manifold would also be a submanifold of the feasible design space, and this would always be the case. However, in the multidisciplinary formulation, the system optimization may move off of the feasible design manifold because the system manifold is not a submanifold of the feasible design space; although they intersect, there is no mechanism for preventing the optimization from leaving that intersection or returning the optimization to that intersection once it has left.

The subsystem optimizations would then take their starting $\mathbf{y}$ values from a point not on the feasible design manifold, and the M_i's defined by the disciplinary analyses and that starting point would be not be guaranteed to intersect the feasible design manifold anywhere. In other words, if the subsystem optimizations do not *start* from a feasible point, it may not be possible for them to *finish* at a feasible point, and the system-level optimization is under no compulsion (or even inclination) to return a feasible point. Moreover, even if the subsystem optimizations start from a feasible point, they, too, will not generally return a feasible design point – and for the same reasons that the system-level optimization fails to return feasible points.

Inspecting the optimality conditions for both the decomposed and original problems would explicitly show how the condition of multidisciplinary feasibility is necessary in order to make the optimality conditions equivalent. That analysis, however, would require more space than we have remaining, and it would only support our conclusions rather than lead to new ones in this regard.

3.2 IDF

Elsewhere, we considered some other MDO architectures within our framework (Bakker et al. 2012), and it may be instructive to reiterate some of our analysis there in comparing one of those architectures with QSD at this point. Martins and Lambe (2013) classify QSD as a distributed form of the Individual Discipline Feasible (IDF) architecture (Cramer et al. 1994), so we will consider IDF. The original MDO problem of minimizing $f(\mathbf{w},\mathbf{y})$ subject to $\mathbf{h}(\mathbf{w},\mathbf{y})=\mathbf{0}$ and $\mathbf{g}(\mathbf{w},\mathbf{y})\leq\mathbf{0}$ is transformed into

$$\min_{\mathbf{w},\boldsymbol{\sigma}} f(\mathbf{w},\mathbf{y})+\phi(\mathbf{y}-\boldsymbol{\sigma}) \tag{21}$$

$$\mathbf{y}=\boldsymbol{\psi}(\mathbf{w},\boldsymbol{\sigma}) \tag{22}$$

$$\mathbf{g}(\mathbf{w},\mathbf{y})\leq\mathbf{0} \tag{23}$$

where the $\phi(\mathbf{y}-\boldsymbol{\sigma})$ in the objective function indicates that the argument of ϕ is an equality constraint which is handled by the optimizer – possibly through a penalty function – and thus satisfied by a convergent final solution (though not necessarily at every iteration). M_{feas} is a submanifold of the architecture's manifold, M_{IDF}, along which the optimizer moves. M_{IDF} is defined by (22), ∂M_{IDF} is defined by (23), M_{feas} is the submanifold resulting from the additional constraint $\mathbf{y}=\boldsymbol{\sigma}$, and M_{tot} is just $\Re^{2m-n}$; $M_{feas}\subset M_{IDF}\subset M_{tot}$. The optimizer, not a multidisciplinary analysis, is thus responsible for eventually forcing the optimization to the feasible submanifold of M_{IDF}, and the presence of $\phi(\mathbf{y}-\boldsymbol{\sigma})$ ensures that it does so.

3.3 MQSD

There are several different ways in which QSD could be modified. In keeping with the description of QSD as a distributed form of IDF, we chose to model our approach after IDF: we used duplicate state variables and optimizer-driven multidisciplinary feasibility in the system-level problem. The system-level problem is thus

$$\min_{\mathbf{z},\boldsymbol{\sigma},\mathbf{b}} f_0(\mathbf{z},\mathbf{y})+\phi(\mathbf{y}-\boldsymbol{\sigma})+\sum_i b^i \tag{24}$$

$$\mathbf{g}_{(0)}(\mathbf{z},\mathbf{y})\leq\mathbf{0} \tag{25}$$

$$s^i\left(\mathbf{x}_{(i)},\mathbf{z},\psi^i\left(\mathbf{x}_{(i)},\mathbf{z},\tilde{\boldsymbol{\sigma}}_{(i)}\right),b^i\right)\leq 0\ ,\ i=1,2,\ldots \tag{26}$$

$$\mathbf{y}=\boldsymbol{\psi}(\mathbf{w},\boldsymbol{\sigma}) \tag{27}$$

and the subsystem problem for subsystem i is

$$\min_{\mathbf{x}_{(i)},s^i} s^i \tag{28}$$

$$\mathbf{g}_{(i)}\left(\mathbf{x}_{(i)},\mathbf{z},y^i\right)-s^i\leq\mathbf{0} \tag{29}$$

$$f_i\left(\mathbf{x}_{(i)},\mathbf{z},y^i\right)-b^i-s^i\leq 0 \tag{30}$$

$$y^i=\psi^i\left(\mathbf{x}_{(i)},\mathbf{z},\tilde{\boldsymbol{\sigma}}_{(i)}\right) \tag{31}$$

We now analyze MQSD in a way paralleling our analysis of QSD. We begin with the relevant manifolds:

1. $M_{tot}=\Re^{n+4(m-n)}$ with coordinates $(\mathbf{x},\mathbf{z},\mathbf{y},\mathbf{b},\mathbf{s},\boldsymbol{\sigma})$.
2. $M_{sys}(\mathbf{x})$ is defined for each value of $\mathbf{x}$ by $\mathbf{s}(\mathbf{z},\boldsymbol{\sigma},\mathbf{b})$ and (27); the local design variables are dropped from the $\mathbf{s}$ equations here to show that they are held constant during the system-level optimization and thus act as fixed parameters and not variables. $M_{sys}(\mathbf{x})$ is an $(n_z+2(m-n))$-manifold naturally embedded in M_{tot}. The boundary, ∂M_{sys}, is defined by (25) and (26).
3. Each $M_i\left(\mathbf{z},\tilde{\boldsymbol{\sigma}}_{(i)},b^i\right)$ is defined by (31), and all variables except $\mathbf{x}_{(i)}$ and s^i are held constant; each M_i is an (n_i+1)-manifold naturally embedded in $\Re^{n_i+2}$.
4. M_{feas} is defined as it was for the original QSD formulation; $\mathbf{h}(\mathbf{w},\mathbf{y})=\mathbf{0}$ is equivalent to the combination of (27) and $\mathbf{y}=\boldsymbol{\sigma}$. M_{feas} is still independent of $\mathbf{b}$ and $\mathbf{s}$ when embedded in M_{tot}, as before.

Again, we consider all of these (parameterized) manifolds embedded in M_{tot}.

1. The M_i's are all mutually orthogonal, but M_{sys} is not orthogonal to the M_i's.
2. M_{sys} intersects M_{feas}, but neither is a submanifold of the other. The intersection $M_{sys}(\mathbf{x})\cap M_{feas}$ is defined by $\mathbf{y}=\boldsymbol{\psi}(\mathbf{w},\boldsymbol{\sigma})$ with $\mathbf{x}$ held constant, $\mathbf{y}=\boldsymbol{\sigma}$, and $\mathbf{s}=\mathbf{s}(\mathbf{z},\mathbf{y}(\mathbf{z}),\mathbf{b})$. Submanifolds of M_{feas} with constant $\mathbf{x}$ are submanifolds of $M_{sys}(\mathbf{x})$.
3. The M_i's may intersect M_{feas}, but those intersecting M_i's are not submanifolds of M_{feas}.

The modification to QSD has not changed the architecture manifolds significantly; the subsystem optimizations, for example, may still move the design point away from M_{feas}. However, the system-level optimizer now tries to return to M_{feas} through the use of penalty functions or some other optimizer method for enforcing the constraint $\mathbf{y}=\boldsymbol{\sigma}$. Like IDF, within a given system-level optimization, those constraints will only be satisfied at an optimum point – feasibility will only be enforced progressively – but the optimizer should eventually ensure that they are satisfied. In fact, at the end of each system-level optimization, the architecture should return a multidisciplinary feasible design; since each architecture iteration ends with a system-level optimization (Martins and Lambe 2013), each iteration of the architecture should return a feasible point. Again, checking the optimality conditions could help to confirm this, but given our qualitative analysis, we now wish to proceed to testing this experimentally.

4 Numerical testing

Our analysis predicts that QSD diverges when the problem it is applied to has state equations (which serve to couple the state variables from different disciplines). We now want to test that prediction. In order to do this, we will use an analytical test problem from Sellar et al. (1996); this problem has been used in other papers (Perez et al. 2004; Tedford and Martins 2006), though not with QSD. We predict that when this problem is optimized with the QSD architecture, it will diverge from the feasible design manifold (as measured by the norm of the state equation residual $\mathbf{h} = \mathbf{y} - \boldsymbol{\psi}(\mathbf{w}, \mathbf{y})$). Conversely, we predict that IDF and MQSD will return feasible design points.

To test these predictions, we ran 500 optimizations from random points on M_{tot} (thus typically infeasible points) and 500 optimizations from random points on M_{feas}, solving for the $\mathbf{y}$-coordinates so that the initial point was feasible. We generated our points using the *sobolset* function, solved our state equations with *fsolve*, and performed our optimizations with *fmincon*, all in MATLAB® (The MathWorks Inc. R2010a). To clarify the point generation procedure by way of an example, for QSD, the infeasible points were generated by applying *sobolset* to $(\mathbf{x}, \mathbf{z}, \mathbf{y}, \mathbf{b})$ within their bounds, and the feasible points instead applied *sobolset* to $(\mathbf{x}, \mathbf{z}, \mathbf{b})$. For the Sellar problem, at least, it did not matter what $\mathbf{s}$ was initialized to because the QSD architecture begins from the subsystem problem; $\mathbf{s}$ was then calculated directly from the other variables as seen in (50) and (51).

The possible ranges for the design variables were already given in the original problem formulation, and for generating our initial points, we produced $\mathbf{b}$'s and $\boldsymbol{\sigma}$'s with coordinate values between 0 and 10. We considered the optimization to have converged once there was less than a 0.001 relative change in the *original objective function*; this was a more stringent criterion than that of the same relative change in the system-level objective function, and it allowed us to apply the same criterion to all three architectures.

4.1 Sellar analytical problem

The Sellar problem is

$$\min f = \left(x^1\right)^2 + z^2 + y^1 + e^{-y^2} \tag{32}$$

$$y^1 = \left(z^1\right)^2 + x^1 + z^2 - 0.2y^2 \tag{33}$$

$$y^2 = \sqrt{y^1} + z^1 + z^2 \tag{34}$$

$$1 - \frac{y^1}{3.16} \leq 0 \tag{35}$$

$$\frac{y^2}{24} - 1 \leq 0 \tag{36}$$

$$-10 \leq z^1 \leq 10, 0 \leq z^2 \leq 10, 0 \leq x^1 \leq 10 \tag{37}$$

Note that we have renamed the variables to correspond to our local/global design variable and subscript/superscript conventions (see Part I).

4.2 QSD reformulation

The QSD for this problem has the following system-level objective function and constraints

$$\min_{\mathbf{z},\mathbf{y},\mathbf{b}} f_0 = z^2 + b^1 + b^2 \tag{38}$$

$$-10 \leq z^1 \leq 10, 0 \leq z^2 \leq 10 \tag{39}$$

$$s^1, s^2 \leq 0 \tag{40}$$

The subsystem 1 optimization is

$$\min_{x^1, s^1} s^1 \tag{41}$$

$$\left(x^1\right)^2 + y^1 - b^1 - s^1 \leq 0 \tag{42}$$

$$1 - \frac{y^1}{3.16} - s^1 \leq 0 \tag{43}$$

$$y^1 = \left(z^1\right)^2 + x^1 + z^2 - 0.2y^2 \tag{44}$$

$$0 \leq x^1 \leq 10 \tag{45}$$

and the subsystem 2 optimization is

$$\min_{s^2} s^2 \tag{46}$$

$$e^{-y^2} - b^2 - s^2 \leq 0 \tag{47}$$

$$\frac{y^2}{24} - 1 - s^2 \leq 0 \tag{48}$$

$$y^2 = \sqrt{y^1} + z^1 + z^2 \tag{49}$$

Some of the expressions with state variables could have been divided up differently between the subsystem and system optimizations, but we chose to try and put as much as possible in the subsystem-level problems; our final results, qualitatively speaking, do not depend on this, however. Note also that because of the simplicity of the problem and the analytical relations between variables, we can evaluate the s^i's analytically rather than using metamodels. Combining the subsystem inequality constraints with the ψ^i equations produces

$$s^1 \geq \begin{cases} \left(x^1\right)^2 + \left(z^1\right)^2 + x^1 + z^2 - 0.2y^2 - b^1 \\ 1 - \frac{\left(z^1\right)^2 + x^1 + z^2 - 0.2y^2}{3.16} \end{cases} \tag{50}$$

$$s^2 \geq \begin{cases} e^{\left(-\sqrt{y^1} - z^1 - z^2\right)} - b^2 \\ \frac{\sqrt{y^1} + z^1 + z^2}{24} - 1 \end{cases} \tag{51}$$

Each s^i is thus simply the larger of the two values on its respective right-hand side.

4.3 IDF reformulation

The IDF reformulation of the Sellar problem is

$$\min_{\mathbf{z},\boldsymbol{\sigma}} f = \left(x^1\right)^2 + z^2 + y^1 + e^{-y^2} \tag{52}$$

$$y^1 = \left(z^1\right)^2 + x^1 + z^2 - 0.2\sigma^2 \tag{53}$$

$$y^2 = \sqrt{\sigma^1} + z^1 + z^2 \tag{54}$$

$$\mathbf{y} - \boldsymbol{\sigma} = \mathbf{0} \tag{55}$$

$$1 - \frac{y^1}{3.16} \le 0 \tag{56}$$

$$\frac{y^2}{24} - 1 \le 0 \tag{57}$$

$$-10 \le z^1 \le 10, 0 \le z^2 \le 10, 0 \le x^1 \le 10 \tag{58}$$

The constraints in (55)-(58) were handled by the optimizer; the disciplinary analyses were directly integrated into the objective function and constraints.

4.4 MQSD reformulation

The system-level optimization of MQSD is

$$\min_{\mathbf{z},\mathbf{b},\boldsymbol{\sigma}} f_0 = z^2 + b^1 + b^2 \tag{59}$$

$$y^1 = \left(z^1\right)^2 + x^1 + z^2 - 0.2\sigma^2 \tag{60}$$

$$y^2 = \sqrt{\sigma^1} + z^1 + z^2 \tag{61}$$

$$\mathbf{y} - \boldsymbol{\sigma} = \mathbf{0} \tag{62}$$

$$-10 \le z^1 \le 10, 0 \le z^2 \le 10 \tag{63}$$

$$s^1, s^2 \le 0 \tag{64}$$

As with IDF, (62) was handled by the optimizer, and the disciplinary analyses were directly integrated into the relevant equations. The subsystem 1 objective function and constraints are

$$\min_{x^1, s^1} s^1 \tag{65}$$

$$\left(x^1\right)^2 + y^1 - b^1 - s^1 \le 0 \tag{66}$$

$$1 - \frac{y^1}{3.16} - s^1 \le 0 \tag{67}$$

$$y^1 = \left(z^1\right)^2 + x^1 + z^2 - 0.2\sigma^2 \tag{68}$$

$$0 \le x^1 \le 10 \tag{69}$$

and the subsystem 2 optimization is

$$\min_{s^2} s^2 \tag{70}$$

$$e^{-y^2} - b^2 - s^2 \le 0 \tag{71}$$

$$\frac{y^2}{24} - 1 - s^2 \le 0 \tag{72}$$

$$y^2 = \sqrt{\sigma^1} + z^1 + z^2 \tag{73}$$

Combining the subsystem inequality constraints with the disciplinary analyses, we get

$$s^1 \ge \begin{cases} \left(x^1\right)^2 + \left(z^1\right)^2 + x^1 + z^2 - 0.2\sigma^2 - b^1 \\ 1 - \frac{\left(z^1\right)^2 + x^1 + z^2 - 0.2\sigma^2}{3.16} \end{cases} \tag{74}$$

$$s^2 \ge \begin{cases} e^{\left(-\sqrt{\sigma^1} - z^1 - z^2\right)} - b^2 \\ \frac{\sqrt{\sigma^1} + z^1 + z^2}{24} - 1 \end{cases} \tag{75}$$

4.5 QSD results

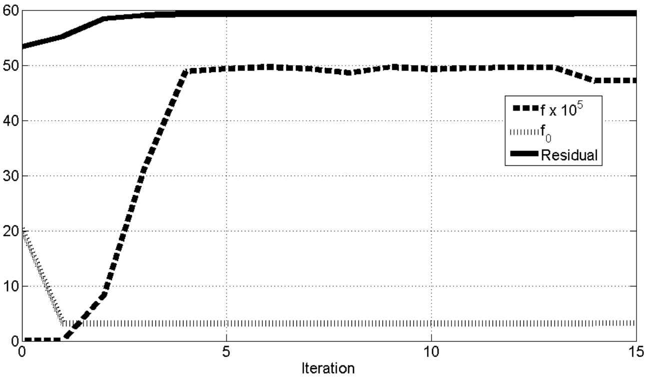

Fig. 1 Objective function, system-level objective, and state equation residual for a single optimization run (QSD)

The history of a single optimization run (Fig. 1), initialized from a random infeasible point in the design space, shows the objective function diverging very strongly as the residual fails to go to zero; the objective function ends up hovering around 5 × 10^6. This was probably due to the e^{-y^2} term: in the decomposed system-level optimization, y^2 is no longer connected to its analysis and thus is not restrained in the optimization process. Handling the explicit constraint on y^2 in the system-level optimization instead of the subsystem optimization would limit this kind of behaviour, but such a reassignment would not eliminate the divergence. Because the state variables were not consistent with each other, reducing f_0 did not help in reducing f – it actually had the opposite effect, in this case. Our

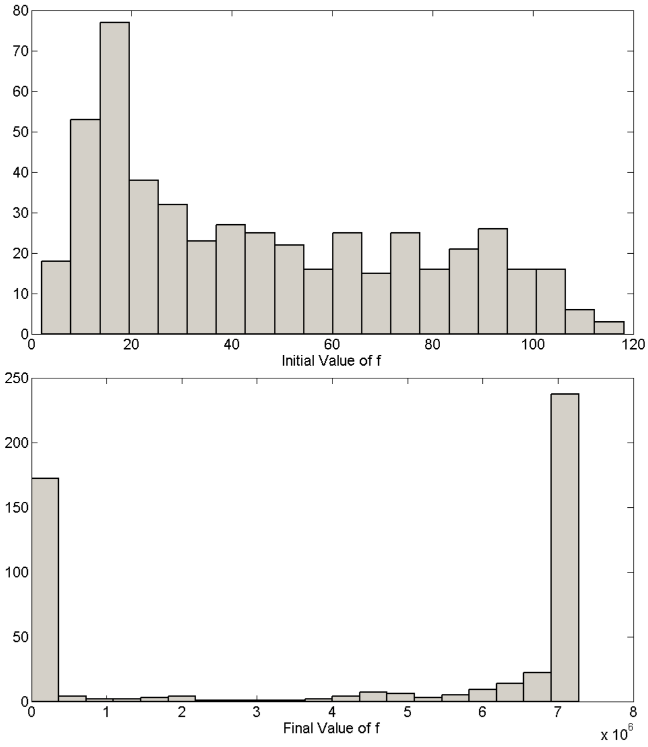

Fig. 2 Histograms comparing initial and final f values (infeasible start, QSD)

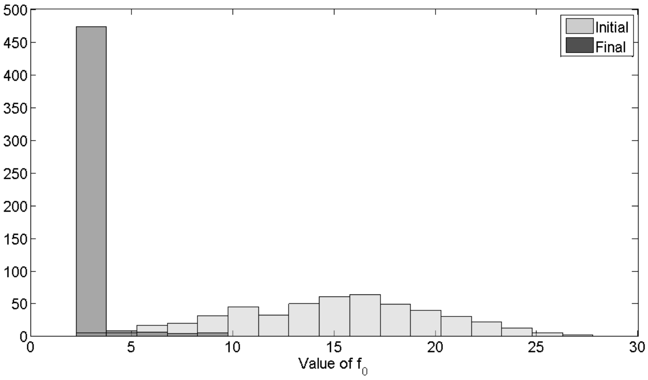

Fig. 3 Histograms comparing initial and final f_0 values (infeasible start, QSD)

analysis concerned feasibility, so that behaviour was not part of our prediction; that being said, it would not be surprising, in general, to see the optimization fail if the design trajectory diverges from feasibility. That divergence is exactly what we see here: the residual initially increased slightly and then remained fairly constant as the design point did not change much after that point.

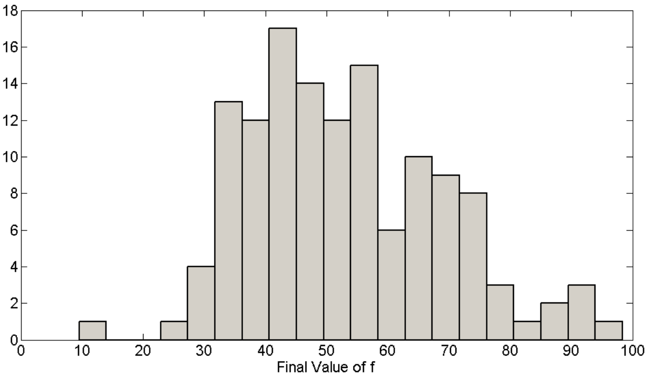

Fig. 4 Histogram showing final f values below 100 (infeasible start, QSD)

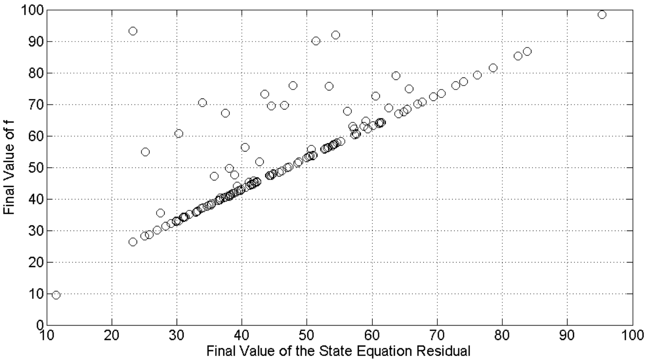

Fig. 5 Plot showing final f values vs. final state equation residual values for $f \leq 100$ (infeasible start, QSD)

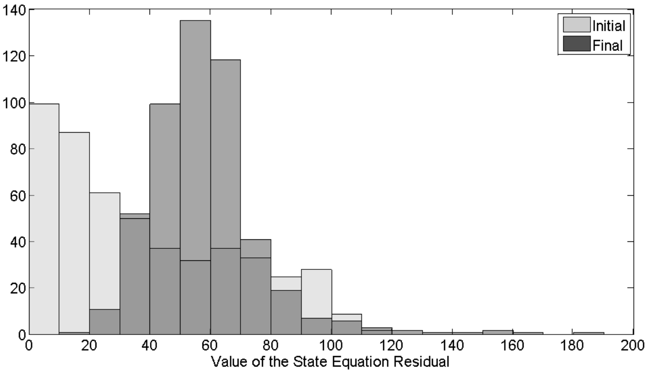

Fig. 6 Histograms comparing initial and final state equation residual values (infeasible start, QSD)

The histograms of the initially infeasible optimization runs in Figs. 2 and 3 show that although the optimizer was clearly working to minimize f_0, typically converging to a value between 3 and 3.5, the original objective function was not being minimized so successfully: only 55 of the 500 runs reduced the value of f, and most of the runs diverged to f values around 10^6.

Even the 132 runs which had final f values below 100, shown in Fig. 4, failed to include any good results: the lowest value reached was 9.52. For those final f values below 100, though – runs which could be considered only *moderately* divergent – Fig. 5 appears to show a

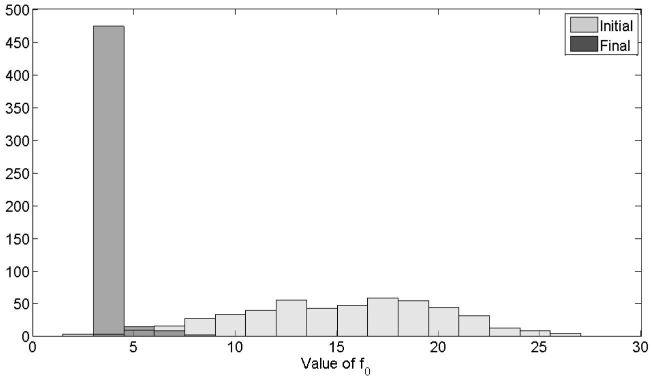

Fig. 7 Histograms comparing initial and final f_0 values (feasible start, QSD)

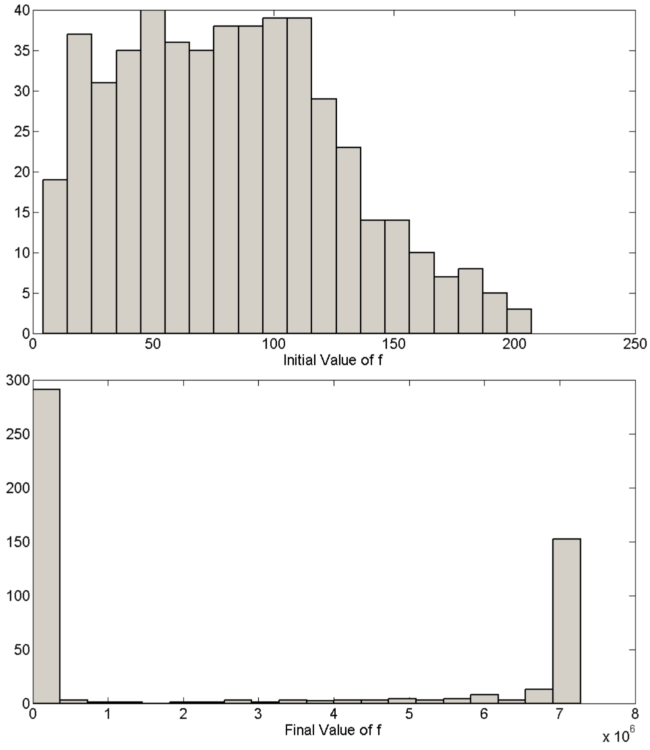

Fig. 8 Histograms comparing initial and final f values (feasible start, QSD)

very strong correlation between residual and objective function value. Residual values did change during the course of the optimization, as can been seen in Fig. 6, but only 134 runs resulted in reduced residual values, and none of those residuals went to zero. Moreover, the average residual value increased from 38.71 to 58.49. Interestingly, in line with the trend seen in Fig. 5, the optimization run with the lowest final residual value (11.48) also had the lowest final objective function value.

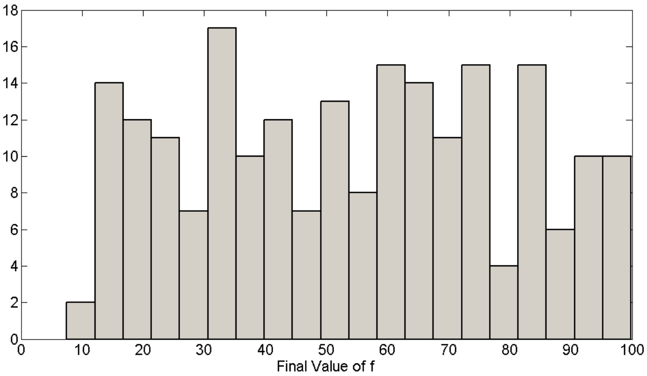

Fig. 9 Histogram showing final f values below 100 (feasible start, QSD)

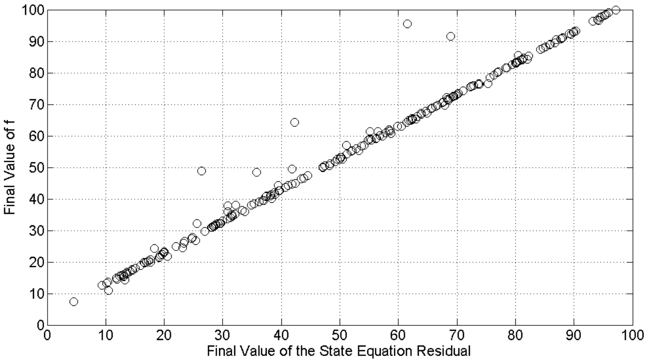

Fig. 10 Plot showing final f values vs. final state equation residual values for $f \leq 100$ (feasible start, QSD)

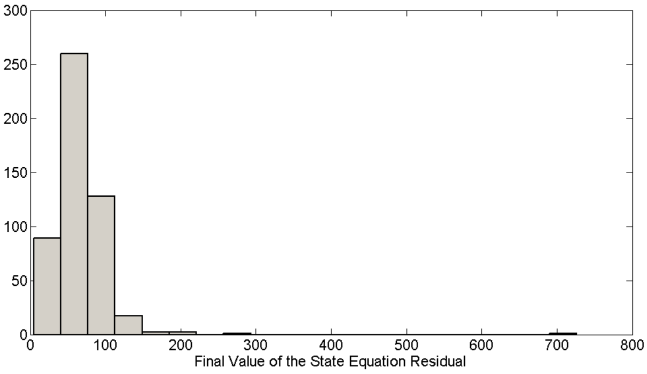

Fig. 11 Histogram of final state equation residual values (feasible start, QSD)

The histograms in Figs. 7-11 for the initially feasible optimization points paint a similar picture to their infeasible counterparts: the histograms for f_0 and f are basically the same, and the final residual is in fact, on average, higher than the infeasible starts (66.54 for feasible starts as compared to 58.49 for infeasible starts).

That being said, there are more objective function values below 100 (213 as compared to 132), and the minimum value reached is lower (7.49 as compared to 9.52). Also, as with the infeasible start, the optimization run with the minimum final residual value (4.54) had the lowest final objective function value; again, there was a strong correlation between final residual values and final objective function values for runs which were only moderately divergent, as shown in Fig. 10.

4.6 IDF results

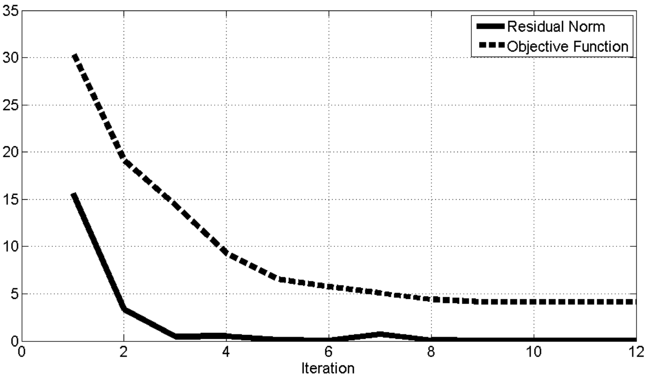

Fig. 12 Objective function and state equation residual for a single optimization run (IDF)

Figure 12 shows essentially what we would expect from an initially infeasible IDF optimization run: the state equation residual and objective function both reduce over time and converge together to a final feasible, optimal design. The decrease in the residual is not entirely monotonic, but it is close.

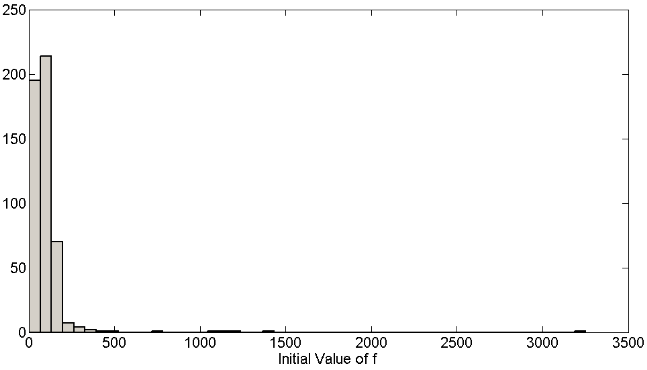

Fig. 13 Histogram of initial f values (infeasible start, IDF)

The initial points for the IDF runs were not necessarily the same as those for the QSD problems because of the different number of variables involved, but the same algorithm (*sobolset*) was used to generate them. As the histograms in Figs. 13 and 14 show (the initial f values for the feasible starts are same as for QSD, shown in Fig. 8), the IDF optimizations all converged to one of two objective function values; the infeasible starts had a much wider range of initial f values, but more of those values got to the lower objective function value. Histograms of the state equation residuals are not shown, but the initial residuals for the feasible start points and the final residuals for feasible and infeasible start points were on the order of 10^{-6} – the final design points obtained were feasible, in other words. We could have gone for tighter constraint tolerances in the optimizer, but this seemed sufficient.

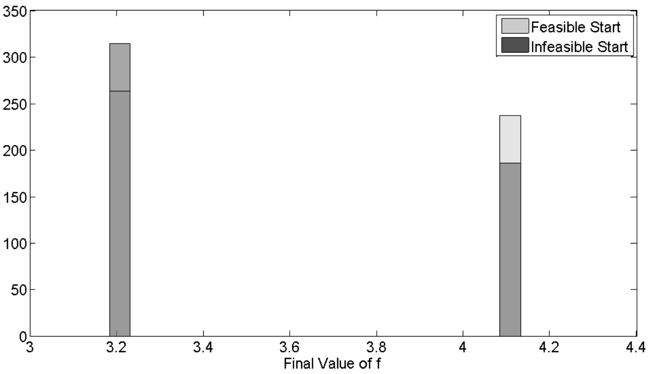

Fig. 14 Histograms comparing final f values for feasible and infeasible starts (IDF)

4.7 MQSD results

MQSD usually converged in two or three architecture iterations. Each architecture iteration was composed of the subsystem optimizations followed by a system-level optimization, so showing a plot like Fig. 1 would not be very informative. Feasibility would be obtained by the end of the first architecture-level iteration and maintained until convergence an iteration or two later.

The plots in Fig. 15 are arranged so that, for each architecture-level iteration, the behaviour of the subsystem optimizations is shown for the first half of the iteration and the system-level optimization is shown in the second half; the subsystem optimizations followed by the system-level optimization constitutes one architecture-level iteration. For example, in the first architecture-level iteration, the progress of the subsystem optimizations is shown between 0 and 0.5, and the progress of the ensuing system-level optimization is shown between 0.5 and 1. The subsystem and system-level optimizations each required a certain number of their own iterations, but that is not really our concern here – we simply want to give a qualitative picture of the architecture's behaviour.

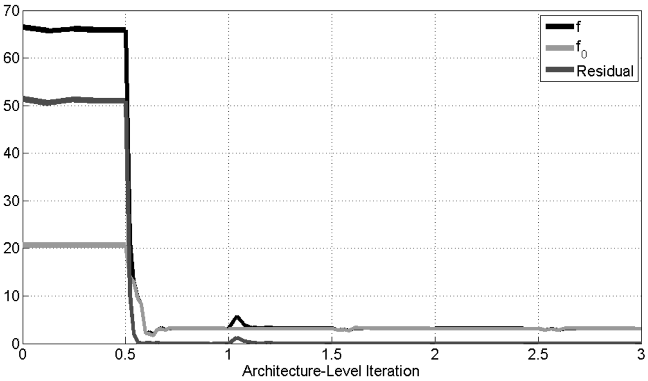

Fig. 15 Objective function, system-level objective, and state equation residual for a single optimization run (MQSD)

Figure 15 shows that most of the optimization work is being done in the first architecture iteration (by the system-level optimization). The following two iterations are only tweaking the result of the first iteration. As predicted, the system-level optimization also produces a feasible design at the end of each architecture-level iteration. We also see that, as the design point becomes feasible, $f_0 \to f$, as is necessary for the architecture to work properly: f_0 is constant during the subsystem optimizations while f varies, but the two are brought back to the same value during the system-level optimization.

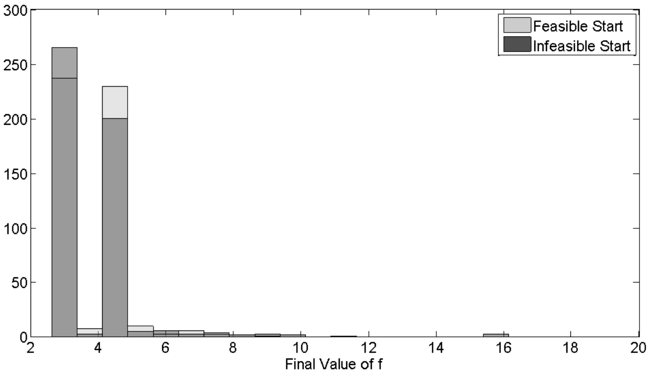

Fig. 16 Histograms comparing final f values for feasible and infeasible starts (MQSD)

For MQSD, the initial f_0 values for feasible and infeasible starts, the initial f values for feasible starts, and the initial residual for infeasible starts were all the same as for the original QSD formulation (shown in Section 4.5); the initial f values for the infeasible starts were the same as for IDF (Fig. 13). The only difference is that the optimizer crashed for four of the runs starting from infeasible start points: the optimizer was using finite differences to get derivative information, and when $y^1 \approx 0$, these four points ended up with imaginary components to the derivatives (through the $\sqrt{y^1}$ term). We could have dealt with this by introducing the constraint $y^1 \geq 0$, but this was not explicitly part of the original problem, and using it actually resulted in the optimizer producing different results for a number of unaffected runs. We wanted the architecture tests to be as alike as possible, so we simply removed the four offending start points from our sample. Given the sample size we used, this seemed a simple and unproblematic way of dealing with this problem. The optimizations themselves overwhelmingly went to the same two points that IDF did. There were a few runs which did not converge to these points (slightly more for the feasible starts than the infeasible starts), but the relevant peaks are still easily identifiable in Fig. 16. Except for one outlier near 10^{-5} from a feasible start, all of the final residuals were on the order of 10^{-6}. Again, the architecture was returning feasible design points. The final values for f_0 were the same as those for f, shown in Fig. 16, which we should expect if the architecture is working properly – it means that minimizing f_0 is actually minimizing f.

4.8 Discussion

As we predicted, QSD failed to return feasible designs. In fact, the final design points were both exceedingly non-optimal and infeasible. Interestingly, there was a strong correlation between successes in optimization and in feasibility if the optimization did not diverge too wildly, but this could be problem-specific. It was necessary to choose somewhat arbitrary starting points for $\mathbf{b}$ and $\boldsymbol{\sigma}$, as they were not specified in the original problem description. Different ranges for those variables might have altered the exact results obtained but would not have changed the overall convergence behaviour – the analysis is independent of the initial values for these variables (i.e. the infeasible results are due to the nature of QSD, not any particular starting point).

IDF, on the other hand, converged very well – both in terms of optimality and in terms of feasibility – and MQSD demonstrated similar performance. The optimization in MQSD was not quite as effective as IDF, but it still worked in the overwhelming majority of cases. This optimization convergence could possibly be improved by making the optimization criteria more stringent. It is also possible, though, that QSD creates extra minima: Haftka and Watson (2005) claim that QSD can actually remove spurious local optima, but they were also dealing with a QSD formulation that lacked state equations. Our analysis was focused on feasibility, not optimality, but checking the optimality conditions for QSD (or MQSD) could help to resolve this.

Comparing iterations with the original QSD formulation would be rather pointless, seeing as that architecture failed to converge, but it might be useful to consider other ways of modifying QSD to ensure convergence and then comparing those different ways to see which modifications are more efficient or effective.

The focus of this paper, however, was the prediction that QSD would not return feasible designs but that IDF and MQSD *would* return feasible designs. This prediction proved to be accurate for the MDO problem we tested it on. Given the generality of the analysis, we would expect the same results in all other nonlinearly coupled MDO problems: both QSD and IDF are relaxations of the original problem, but IDF is a relaxation which works, and QSD is a relaxation which does not. Relaxation can be very effective – the issue does not lie in relaxing the original optimization problem – but such a technique must still be able to solve that original optimization problem.

The analysis we did to come to these conclusions was relatively simple and straightforward: all we did was identify the relevant manifolds and submanifolds for the architecture and the original problem, determine the relationships between those manifolds and submanifolds, and then describe how the architecture moved along those manifolds during the course of optimization. Once the manifold/submanifold concept was grasped, with the support of our framework, there were no long proofs or further mathematical gymnastics.

5 Conclusions and recommendations

Using the concepts and tools provided by our DG framework for MDO, we predicted that QSD would fail to return a feasible design in optimization problems with state equations – i.e. a typical MDO problem. Importantly, the framework enabled us to explain *why* QSD would fail and thus enabled us to propose changes to QSD which would render it convergent; we were also able to analyze an architecture which we successfully predicted would converge to feasible designs.

Having done this, we tested all three architectures on a well-known analytical MDO problem. As predicted, QSD failed to converge, but both IDF and MQSD converged. MQSD was not quite as successful in the optimization as IDF was, but both architectures returned feasible designs, and that was the focus of our analysis. Now that the question of feasibility has been resolved, the optimizing behaviour of MQSD variations could be investigated analytically, by examining their optimality conditions, and experimentally, by comparing their performance in numerical simulations.

Acknowledgments

This research is supported by the Natural Sciences and Engineering Research Council of Canada (NSERC) and the Cambridge Commonwealth Trust and Cambridge Overseas Trust (CCT/COT).

References

Bakker C, Parks GT, Jarrett JP (2012) Geometric perspectives on MDO and MDO architectures. In: 12th Aviation Technology, Integration and Operations (ATIO) Conference and 14th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, AIAA, Indianapolis, Indiana

Cramer E, Dennis JE Jr, Frank PD, Lewis RM, Shubin GR (1994) Problem formulation for multidisciplinary optimization problems. SIAM Journal on Optimization 4:754–776

Haftka RT, Watson LT (2005) Multidisciplinary design optimization with quasiseparable subsystems. Optimization and Engineering 6:9–20

Haftka RT, Watson LT (2006) Decomposition theory for multidisciplinary design optimization problems with mixed integer quasiseparable subsystems. Optimization and Engineering 7:135–149

Ivancevic VG, Ivancevic TT (2007) Applied Differential Geometry: A Modern Introduction. World Scientific Publishing Co. Pte. Ltd., Singapore

Martins JRRA, Lambe AB (2013) Multidisciplinary design optimization: Survey of architectures. AIAA Journal 51:2049–2075

Perez RE, Liu HHT, Behdinan K (2004) Evaluation of multidisciplinary optimization approaches for aircraft conceptual design. In: 10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, AIAA, Albany, New York

Sellar RS, Batill SM, Renaud JE (1996) Response surface based concurrent subspace optimization for multidisciplinary system design. In: 34th Aerospace Sciences Meeting and Exhibit, AIAA, Reno, Nevada

Tedford NP, Martins JRRA (2006) On the common structure of MDO problems: A comparison of architectures. In: 11th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, AIAA, Portsmouth, Virginia

The MathWorks Inc (R2010a) Matlab® manual

de Wit AJ, van Keulen F (2007) Numerical comparison of multi-level optimization techniques. In: 48th AIAA/ASME/ASCE/ASC Structures, Structural Dynamics, and Materials Conference, AIAA, Honolulu, Hawaii