11th World Congress on Structural and Multidisciplinary Optimization
7th - 12th, June 2015, Sydney, Australia

Structural and Aerostructural Design of Aircraft Wings with a Matrix-Free Optimizer

Andrew B. Lambe[1], Joaquim R. R. A. Martins[2]

[1] Institute for Aerospace Studies, University of Toronto, Toronto, Ontario, Canada, andrew.lambe@mail.utoronto.ca
[2] Department of Aerospace Engineering, University of Michigan, Ann Arbor, Michigan, USA, jrram@umich.edu

1. Abstract
In structural optimization subject to failure constraints, computing the gradients of a large number of functions with respect to a large number of design variables may not be computationally practical. Often, the number of constraints in these optimization problems is reduced using constraint aggregation at the expense of a higher mass of the optimal structural design. This work presents results of structural and coupled aerodynamic and structural design optimization of aircraft wings using a novel matrix-free augmented Lagrangian optimizer. By using a matrix-free optimizer, the computation of the full constraint Jacobian at each iteration is replaced by the computation of a small number of Jacobian-vector products. The low cost of the Jacobian-vector products allows optimization problems with thousands of failure constraints to be solved directly without resorting to constraint aggregation. The results indicate that the matrix-free optimizer reduces the computational work of solving the optimization problem by an order of magnitude compared to a traditional sequential quadratic programming optimizer. Furthermore, the use of a matrix-free optimizer makes the solution of large multidisciplinary design problems, in which gradient information must be obtained through iterative methods, computationally tractable.
2. Keywords: Matrix-free optimizer, multidisciplinary design optimization, structural optimization, constraint aggregation

3. Introduction

When solving structural optimization problems or multidisciplinary design optimization (MDO) [9] problems involving a structural analysis, we want to use failure constraints directly in our problem formulation. However, constraining the optimization problem in this way for complex structures like aircraft wings leads to a problem formulation with thousands of constraints. Often, constraint aggregation, such as Kreisselmeier–Steinhauser (KS) aggregation [7, 12], is employed to reduce the number of constraints in the problem yet still obtain a feasible final design. While this approach is effective in reducing the computational cost of the optimization, it compromises the quality of the final structural design. The KS parameter must be chosen to balance accuracy of the feasible design space with the conditioning of the optimization problem itself.

We propose to avoid excessive constraint aggregation by using a gradient-based optimizer that, instead of requiring the gradients explicitly, requires only matrix-vector products with the constraint Jacobian. When a structural or multidisciplinary analysis is solved at each optimizer iteration, the expression for the constraint Jacobian is

$$\left[\frac{\mathrm{d}c}{\mathrm{d}x}\right] = \left[\frac{\partial C}{\partial x}\right] - \left[\frac{\partial C}{\partial y}\right]\left[\frac{\partial R}{\partial y}\right]^{-1}\left[\frac{\partial R}{\partial x}\right] \tag{1}$$

where R are the governing equations of the analysis, y are the state variables, x are the design variables, and C are the design constraints. To form both forward and transpose matrix-vector products with (1), only a single linear system needs to be solved per product, thus avoiding the high cost of forming Jacobian. An optimizer that accesses gradient information through matrix-vector products alone is referred to as a matrix-free optimizer. The research discussed in this paper follows our previous work developing a matrix-free augmented Lagrangian algorithm [1, 8].

In this paper, we present the results of applying our matrix-free optimizer to a pair of aircraft wing design optimization problems. The first problem is a minimum-mass problem subject to failure constraints on the wing at two load conditions. The second problem is an MDO problem in which the take-off gross weight (TOGW) of an aircraft is minimized for a design mission considering both structural and aerodynamic characteristics of the wing.

4. Matrix-Free Augmented Lagrangian Method

We chose to modify the classical augmented Lagrangian algorithm [2] to create a matrix-free optimizer capable of solving optimization problems with nonlinear equality and inequality constraints. In the augmented Lagrangian algorithm, the nonlinear constraints in the optimization problem

$$\begin{aligned} \text{minimize} \quad & F(x) \\ \text{with respect to} \quad & x \\ \text{subject to} \quad & C(x) \geq 0 \\ & x_L \leq x \leq x_U \end{aligned} \tag{2}$$

are relaxed to yield

$$\begin{aligned} \text{minimize} \quad & \Phi(x,t;\lambda,\rho) = F(x) - \lambda^T(C(x)-t) + \frac{\rho}{2}(C(x)-t)^T(C(x)-t) \\ \text{with respect to} \quad & x,t \\ \text{subject to} \quad & x_L \leq x \leq x_U \\ & t \geq 0, \end{aligned} \tag{3}$$

where Φ is the augmented Lagrangian function, λ is a vector of Lagrange multiplier estimates, ρ is a penalty parameter, and t is a vector of slack variables. Each major iteration of the augmented Lagrangian method consists in solving problem (3) and updating ρ and λ based on the constraint infeasibility at the optimal choice of x and t. We use the updating scheme proposed by Conn et al. [2] to solve problem (3) approximately.

Problem (3) is solved by an L_∞ trust-region approach. The trust region subproblem is given by

$$\begin{aligned} \text{minimize} \quad & Q(p) = \frac{1}{2}p^T Bp + g^T p \\ \text{with respect to} \quad & p \\ \text{such that} \quad & p_L \leq p \leq p_U, \end{aligned} \tag{4}$$

where B is an estimate of the Hessian $\nabla^2\Phi$, g is the gradient $\nabla\Phi$, and p is the search direction in both x and t. The choice of the L_∞ trust region makes handling the bound constraints in problem (3) easy. Subproblem (4) is solved by the algorithm of Moré and Toraldo [10], modified to account for the case where the approximate Hessian B is indefinite. Moré and Toraldo's algorithm needs only matrix-vector products with B to solve (4), so all that is needed to make the algorithm matrix-free is a suitable approximation to $\nabla^2\Phi$.

In our optimization problems of interest, we do not have access to second derivatives of any of the functions. Therefore, we use quasi-Newton methods to approximate $\nabla^2\Phi$ in the trust-region subproblem. To increase the accuracy of our model Hessian B, we use two quasi-Newton approximations within B instead of a single one. We have developed two approaches to approximating $\nabla^2\Phi$ that balance a low cost of implementation with the ability to exploit the structure of $\nabla^2\Phi$. Analytically, the true Hessian of the Lagrangian is given by

$$\nabla^2\Phi = \begin{bmatrix} \nabla^2 F - \sum_{i=1}^m \lambda_i \nabla^2 C_i + \sum_{i=1}^m (\rho(C_i(x)-t_i))\nabla^2 C_i + \rho J^T J & \rho J^T \\ \rho J & \rho I \end{bmatrix}, \tag{5}$$

where $J = [\nabla C(x)]^T$.

In the split-quasi-Newton approach, the augmented Lagrangian is broken up into Lagrangian and infeasibility functions and a separate quasi-Newton method is used to approximate the Hessian of each function. In particular,

$$\begin{aligned} \Phi(x,t;\lambda,\rho) &= \mathscr{L}(x,t;\lambda) + \rho\mathscr{I}(x,t) \\ \mathscr{L}(x,t;\lambda) &= F(x) - \lambda^T(C(x)-t) \\ \mathscr{I}(x,t) &= \frac{1}{2}(C(x)-t)^T(C(x)-t). \end{aligned} \tag{6}$$

We use a symmetric rank-one (SR1) [11] quasi-Newton approximation to $\nabla^2\mathscr{L}$ and a Broyden–Fletcher–Goldfarb–Shanno (BFGS) [11] quasi-Newton approximation to $\nabla^2\mathscr{I}$. These choices were made based on the fact that $\nabla^2\mathscr{I}$ is positive-semidefinite near a local minimum, while $\nabla^2\mathscr{L}$ could be indefinite. Both quasi-Newton approximations are of the limited-memory variety to allow us to solve large problems efficiently.

In the approximate-Jacobian approach, we directly approximate the Jacobian itself using a quasi-Newton method and truncate the term $\sum_{i=1}^m (\rho(C_i(x)-t_i))\nabla^2 C_i$ in (5). This truncation is justified by the fact that $C(x)-t=0$ at an optimal solution, so the term becomes negligible as optimality is approached. Again, we use a limited-memory SR1 approximation to $\nabla^2\mathscr{L}$ for the Lagrangian Hessian, while the Jacobian is approximated by a full-memory adjoint Broyden method [13]. The adjoint Broyden update is given by

$$A^{k+1} = A^k + \frac{\sigma^k\sigma^{k,T}}{\sigma^{k,T}\sigma^k}\left(J^{k+1} - A^k\right), \tag{7}$$

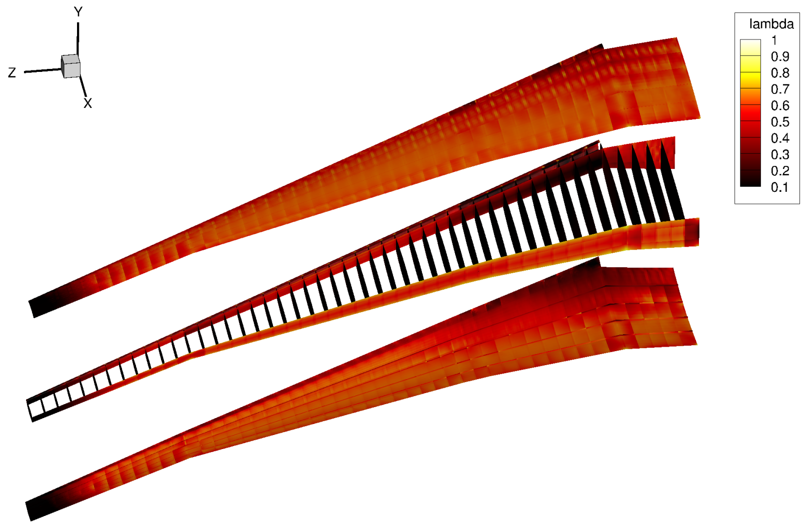

Figure 1: Exploded view of structure layout used in the test problems. The contours represent stress as a fraction of the yield stress for the optimal wing structure using a KS parameter value of 100.

where A^k is the approximate Jacobian at iteration k and

$$\sigma^k = (J^{k+1} - A^k)s^k. \tag{8}$$

In contrast to quasi-Newton methods for square matrices, there are no limited-memory variants of the adjoint Broyden method with a convergence guarantee, so we use the full memory version. To improve the computational performance of this method on large problems in a parallel computing environment, message passing interface (MPI) standard instructions are used to distribute the matrix approximation over multiple processors and form matrix-vector products with A.

5. Analysis Software

Our matrix-free optimizer, AUGLAG, is benchmarked against the SQP optimizer SNOPT [3]. SNOPT also uses a limited-memory quasi-Newton approximation to $\nabla^2\mathscr{L}$ to solve a given optimization problem. However, SNOPT requires the full constraint Jacobian to be computed at each iteration, while AUGLAG needs only Jacobian-vector products. While we expect to see a difference in the results due to the difference in optimization algorithms, our aim is to show that AUGLAG is still competitive with SNOPT due to the low cost of the trust-region iterations in AUGLAG.

The wing structure used in our test problems is analyzed using the Tookit for the Analysis of Composite Structures (TACS) [4], a finite-element analysis code. The wing aerodynamics were analyzed using the three-dimensional panel code TriPan [5]. All analysis and optimization codes were accessed through the MACH framework (MDO of aircraft configurations at high fidelity) [6]. The MACH framework includes modules for aerostructural analysis and geometry warping.

Prior to this project, both the TACS and TriPan codes and the MACH framework possessed modules for efficiently computing derivatives using the adjoint method. However, in order to enable the matrix-free approach to optimization, modules needed to be added to compute forward and transpose matrix-vector products with the partial derivative matrices shown in (1). We expect other researchers interested in using a matrix-free optimizer would need to undertake similar modifications to their solvers. However, if the direct and adjoint methods are already available, the implementation is relatively straightforward.

6. Structural Optimization Results

The first problem we study is the minimization of a wing box mass subject to failure constraints. The outer wing geometry is based on the Boeing 777-200ER civil transport. Figure 1 shows the structural layout used. The wing is 30.5 m from root to tip, and the assumed aircraft mass is 298 000 kg. The wing is analyzed at two load

Table 1: Average run times for specific computations in the wing structure optimization problem

Computation	**Wall Time, 32 proc.**
Objective and 2832 constraints	2.31 s
Jacobian of 2832 constraints	188.50 s
Objective gradient only	0.02 s
Jacobian-vector product with 2832 constraints	0.40 s
Transpose Jacobian-vector product with 2832 constraints	0.33 s

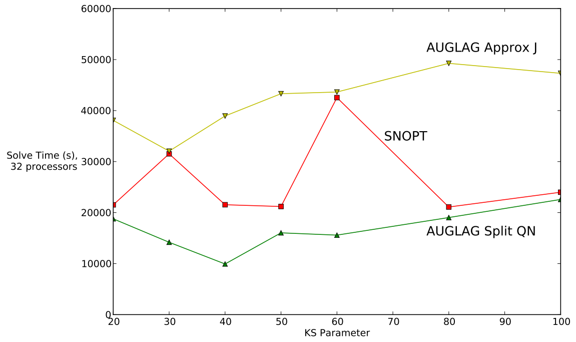

Figure 2: Run time to solve the wing structure optimization problem for a range of KS parameter values

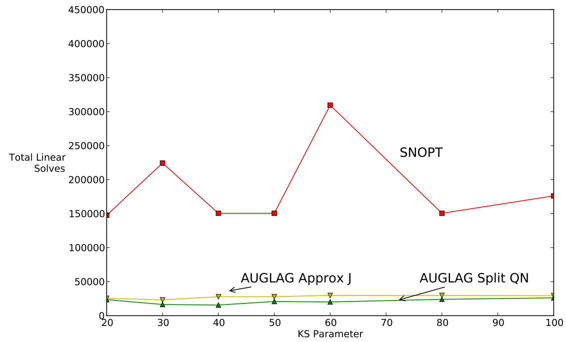

Figure 3: Number of linear solve operations to solve the wing structure optimization problem for a range of KS parameter values

cases: a 2.5g pull-up maneuver and a 1g push-over maneuver. The wing model itself contains nearly 46 000 finite elements and 250 000 degrees of freedom. Individual thickness design variables and yield stress failure constraints are assigned to consistent patches of elements. Note that KS aggregation is still used in the failure constraints of this problem, but only at the level of the element patches. The optimization problem has 1416 variables and 2832 constraints.

Figures 2 and 3 show the cost of the optimization using SNOPT and the two versions of AUGLAG over a range of KS parameters in terms of both run time and the number of linear solve operations. The latter metric treats one matrix-vector product as equivalent in cost to forming a single row or column of the constraint Jacobian. Because the linear solve operation is the most expensive in forming the Jacobian (1), the number of linear solves acts as a computational cost estimate. Figure 3 shows that AUGLAG is far more efficient than SNOPT at optimizing the wing design, in terms of the number of linear solves, for a range of KS parameter values. Reductions in the number of linear solves can be up to an order of magnitude. However, Figure 2 shows that AUGLAG is only more efficient in terms of run time when using the split-quasi-Newton Hessian approximation.

The variation in the results with the increasing KS parameter value shown in Figures 2 and 3 raises the question of whether or not this is a random phenomenon based on the starting point. The starting point used was a constant

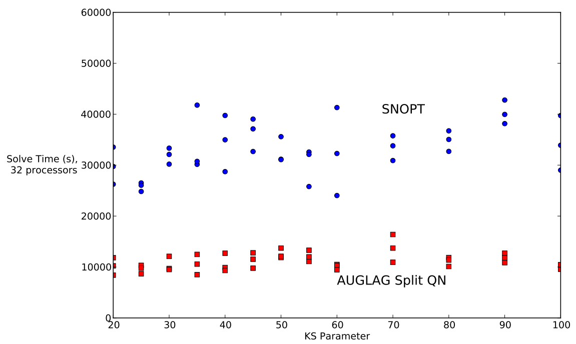

Figure 4: Run time to solve the wing structure optimization problem from a random starting point

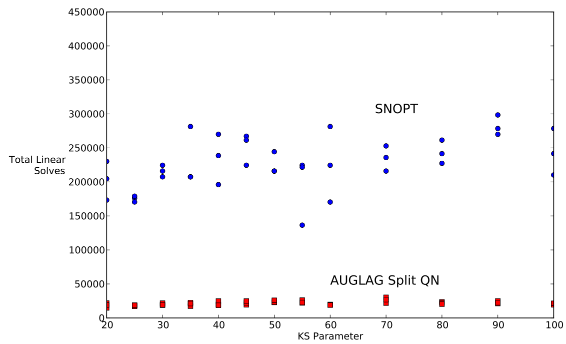

Figure 5: Number of linear solve operations to solve the wing structure optimization problem from a random starting point

Table 2: Average run times for specific computations in the aerostructural optimization problem

Computation	**Wall Time, 32 proc.**
Objective and 4251 constraints	16.82 s
Jacobian of 4251 constraints	26 926.00 s
Objective gradient only	3.86 s
Jacobian-vector product with 4251 constraints	14.36 s
Transpose Jacobian product with 4251 constraints	8.15 s

Table 3: Computational resources to solve the aerostructural optimization problem

Optimizer	**Number of Linear Solves**	**Wall Time, 32 proc.**
SNOPT (50 iteration estimate)	217107	382.5 hr
AUGLAG Split QN (average)	38481	38.0 hr

thickness in all elements. Figures 4 and 5 show results for SNOPT and the split-quasi-Newton version of AUGLAG using sets of randomly-generated thickness values as the starting points. The data in Figures 4 and 5 fall into a particular range for each optimizer, suggesting that the oscillations observed in Figures 2 and 3 are indeed a random occurrence.

The difference in the results between Figures 2 and 3 comes from the fact that the relative cost of computing the Jacobian in this problem is low compared to evaluating the constraints. Table 1 shows that, using 32 processors in parallel, obtaining the Jacobian requires only 90 times as much time as computing the objective and all the failure constraints. If all linear solves were near-equal effort, we would expect a factor closer to a thousand. However, TACS is designed to compute gradients very efficiently once the functions have been evaluated. As a result, the large number of iterations taken by AUGLAG hinders the performance more than the matrix-free interface helps. The results are further tempered by the fact that both versions of AUGLAG could only converge reliably to an optimality tolerance of 3×10^{-4} while SNOPT was able to converge to a tolerance of 10^{-5}. We believe this is caused by the inability of the augmented Lagrangian algorithm to update the Lagrange multipliers as frequently. Despite the different tolerances, AUGLAG computes optimal mass values that are within 1-3% of the mass predicted by SNOPT. Because of this similarity in solutions, we can say that AUGLAG provides a fast estimate of an optimal solution for problems with many design variables and many constraints.

7. Aerostructural Optimization Results

The wing design optimization problem is now expanded to include aerodynamic analysis and aerostructural coupling. The objective of the new problem is to minimize the TOGW of the aircraft for a 7725-nautical-mile design mission. TOGW includes the weight of the wing structure, the weight of the fuel burned, and fixed weights representing the payload and the rest of the aircraft structure. A third load case and another 1416 failure constraints are added to model the wing at cruise. An angle-of-attack design variable is introduced for each load case along with corresponding constraints to match the aircraft weight to the lift generated by the wing. Finally, five twist variables are introduced to allow the optimizer to twist the jig shape of the wing to alleviate high loads. The final design problem contains 1424 design variables and 4251 constraints.

The critical feature of this problem is the high cost of computing the Jacobian. Because the problem is multidisciplinary, the linear system that is solved in forming the Jacobian or Jacobian-vector products can only be solved iteratively, so the cost of the Jacobian is far higher than the cost of the objective and constraint functions. Our tests (see Table 2) suggest that computing the full Jacobian would take about 7.5 hours using 32 processors. Under this time constraint, an optimizer like SNOPT would only be able to complete five iterations in a typical two-day high-performance computing job. Given that the structural optimization problem shown in Section 6 required approximately 50 iterations to converge, this run time is estimated to be 16 days, which is too long.

Using the split-quasi-Newton version of AUGLAG and a convergence tolerance of 3×10^{-4}, we are able to compute optimal wing designs within the two-day limit. Table 3 shows the run time and number of linear solves required by AUGLAG, averaged over seven KS parameter values, compared with our estimate of the resources required for SNOPT to solve the same problem. In terms of both the number of linear solves and run time, AUGLAG is up to an order of magnitude faster than the estimated results of SNOPT.

8. Conclusions

We have presented results for the optimization of a wing structure using a new matrix-free optimizer. We deliberately avoided aggressive constraint aggregation and formulated problems with both thousands of variables and thousands of constraints. Our matrix-free optimizer is capable of solving these large optimization problems much more quickly than a traditional optimizer. Depending on the relative cost of the Jacobian evaluation, the total computational effort can be reduced by up to an order of magnitude. We expect that further gains are possible if more advanced optimization algorithms are adapted to be matrix-free.

9. Acknowledgements

Computations were performed on the GPC supercomputer at the SciNet HPC Consortium. SciNet is funded by: the Canada Foundation for Innovation under the auspices of Compute Canada; the Government of Ontario; Ontario Research Fund - Research Excellence; and the University of Toronto.

10. References

[1] S. Arreckx, A. Lambe, J. R. R. A. Martins, and D. Orban. A matrix-free augmented Lagrangian algorithm with application to large-scale structural design optimization. *Optimization and Engineering*, 2015. (Accepted subject to revisions).

[2] A. R. Conn, N. I. M. Gould, and P. L. Toint. A Globally Convergent Augmented Lagrangian Algorithm for Optimization with General Constraints and Simple Bounds. *SIAM Journal on Numerical Analysis*, 28(2): 545–572, 1991.

[3] P. E. Gill, W. Murray, and M. A. Saunders. SNOPT: An SQP Algorithm for Large-Scale Constrained Optimization. *SIAM Journal on Optimization*, 12(4):979–1006, 2002.

[4] G. J. Kennedy and J. R. R. A. Martins. A parallel finite-element framework for large-scale gradient-based design optimization of high-performance structures. *Finite Elements in Analysis and Design*, 87:56–73, Sept. 2014. ISSN 0168874X. doi: 10.1016/j.finel.2014.04.011. URL `http://linkinghub.elsevier.com/retrieve/pii/S0168874X14000730`.

[5] G. J. Kennedy and J. R. R. A. Martins. A parallel aerostructural optimization framework for aircraft design studies. *Structural and Multidisciplinary Optimization*, 50(6):1079–1101, December 2014. doi: 10.1007/s00158-014-1108-9.

[6] G. K. W. Kenway, G. J. Kennedy, and J. R. R. A. Martins. Scalable parallel approach for high-fidelity steady-state aeroelastic analysis and derivative computations. *AIAA Journal*, 52(5):935–951, May 2014. doi: 10.2514/1.J052255.

[7] G. Kreisselmeier and R. Steinhauser. Systematic Control Design by Optimizing a Vector Performance Indicator. In *Symposium on Computer-Aided Design of Control Systems*, pages 113–117, Zurich, Switzerland, 1979. IFAC.

[8] A. B. Lambe and J. R. R. A. Martins. A matrix-free approach to large-scale structural optimization. In *Proceedings of the 10th World Congress on Structural and Multidisciplinary Optimization*, Orlando, FL, May 2013.

[9] J. R. R. A. Martins and A. B. Lambe. Multidisciplinary design optimization: A survey of architectures. *AIAA Journal*, 51(9):2049–2075, September 2013. doi: 10.2514/1.J051895.

[10] J. J. Moré and G. Toraldo. On the Solution of Large Quadratic Programming Problems with Bound Constraints. *SIAM Journal on Optimization*, 1(1):93–113, 1991.

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

[12] N. M. K. Poon and J. R. R. A. Martins. An adaptive approach to constraint aggregation using adjoint sensitivity analysis. *Structural and Multidisciplinary Optimization*, 34:61–73, 2007. doi: 10.1007/s00158-006-0061-7.

[13] S. Schlenkrich, A. Griewank, and A. Walther. On the local convergence of adjoint Broyden methods. *Mathematical Programming*, 121:221–247, 2010. doi: 10.1007/s10107-008-0232-y.