EVALUATION OF OPTIMIZATION METHODS FOR CONTROL ALLOCATION

Marc Bodson*
University of Utah, Electrical and Computer Engineering
50 S Central Campus Dr Rm 3280
Salt Lake City UT 84112

Abstract

The paper evaluates the performance and computational requirements of optimization methods for control allocation. Two control allocation problems are formulated: a direct allocation method that preserves the directionality of the moment, and a mixed optimization method that minimizes the error between the desired and the achieved moments as well as the control effort. The constrained optimization problems are transformed into linear programs, so that they may be solved using well-tried linear programming techniques such as the simplex algorithm. The paper discusses a variety of techniques that may be applied for the solution of the control allocation problem in order to accelerate computations. Performance and computational requirements are evaluated using aircraft models with different numbers of actuators and with different properties. In addition to the two optimization methods, three algorithms with low computational requirements are also implemented for comparison: a redistributed pseudo-inverse technique, a quadratic programming algorithm, and a fixed-point method. The major conclusion of the paper is that constrained optimization may be performed with computational requirements that fall within an order of magnitude of those of simpler methods. The performance gains of optimization methods, measured in terms of the error between the desired and achieved moments, are found to be small on the average, but sometimes significant. A variety of issues are discussed in the paper that affect the implementation of the various algorithms in a flight control system.

1. Introduction

In many flight control systems, ganging has been used to associate the three rotational degrees of freedom of an aircraft to its control surfaces. So, a pitching command is transformed into identical left and right elevator deflections, while a rolling command is transformed into opposite left and right aileron deflections. As advanced aircraft with many and unconventional surfaces are being designed, the appropriate manner in which ganging should be implemented is becoming less obvious. Further, there is increased interest in control methods that are capable of reconfiguration after failures. In such cases, it is unlikely that a ganging method designed for an unfailed aircraft would be optimal after a failure. Therefore, there is a need for control allocation algorithms that perform *automatic*

* Marc Bodson is a Professor of Electrical and Computer Engineering and a Senior Member, AIAA.

distribution of the control requirements among a large number of control surfaces, and exploit *all* the maneuvering capabilities of an aircraft given position and rate limits on the actuators.

The simplest control allocation methods are based on the unconstrained least-squares algorithm, and modifications of the solution aimed at accounting for position and rate limits [1], [2]. More complex methods formulate control allocation as a constrained optimization problem [3], [4]. Until recently, it was believed that optimization methods would be too complex, or too time-consuming, to be implemented in a flight control system [1], [5]. However, dramatic increases in computing speed, as well as more efficient algorithms, are rapidly changing that picture.

The objective of this paper is to evaluate optimization-based algorithms for control allocation, and to compare the results to those of simpler methods. The two main questions are: (1) whether these methods provide significant improvements of performance, in terms of extracting all the available control power from the effector suite, and (2) whether their computational requirements place them out of reach of existing computers. The optimization algorithms evaluated in the paper are well-tried methods of linear programming. They are implemented to solve both the direct allocation problem posed by Durham [4], and a more common error minimization objective [6]. A numerical evaluation is performed using a transport aircraft model and a tailless aircraft model. The transport aircraft model is evaluated with and without a ganging matrix that brings the number of actuators from 16 to 8, allowing one to evaluate the cost and benefits of a higher number of actuators in the allocation scheme. The tailless model is of particular interest because it does not satisfy a condition that guarantees uniqueness of the solution in the direct allocation problem.

2. Control Allocation Problem

2.1 Control Allocation for Flight Control

We begin the paper by briefly discussing the motivation for control allocation. Consider the state-space model

$$\begin{aligned} \dot{x} &= Ax + Bu + d \\ y &= Cx \end{aligned} \tag{2.1}$$

where x, d , u, and y are all vectors. For the control of aircraft, the state vector x may include the angle of attack, the pitch rate, the angle of sideslip, the roll rate, and the yaw rate. The output vector y may contain the pitch rate, the roll rate, and the yaw rate. The control input vector u consists of the commanded actuator positions, or in the control surface deflections if the actuator dynamics are neglected. If the control variables are ganged, the number of control variables $p=\dim(u)$ may be as small as 3. Otherwise, the typical range is $p=5 \rightarrow 20$.

Model reference control laws, sometimes referred to as dynamic inversion control laws, rely on a reference model which represents the desired dynamics of the closed-loop system, for example

$$\dot{y}_M = A_M y_M + B_M r_M \tag{2.2}$$

where r_M is a reference input vector (determined by the pilot commands), and y_M represents the desired output of the system. Since the derivative of y is given by

$$\dot{y} = CAx + CBu + Cd \tag{2.3}$$

the objective may be achieved by setting

$$u = (CB)^{-1}\left(-CAx - Cd + A_M y + B_M r_M\right) \tag{2.4}$$

Model matching follows if the matrix *CB* is square and invertible, and if the original system is minimum phase. Adaptive implementations of this control law were discussed in the context of reconfigurable flight control in [7], [8].

If the matrix *CB* is not full rank, model matching may still be possible, but with a different model and a more complex control law. On the other hand, if *CB* is not square but full row rank (has more columns than rows, as in the case of redundant actuators), the same model reference control law can be used if one defines

$$a_d = -CAx - Cd + A_M y + B_M r_M \tag{2.5}$$

and a control input *u* such that

$$(CB)u = a_d \tag{2.6}$$

Obtaining *u* from (2.6) requires that one solve a system of linear equations with more unknowns than equations. This may seem like an easy problem, but the difficulty in practice is that the vector *u* is constrained. The limits generally have the form

$$u_{\min,i} \le u_i \le u_{\max,i} \quad \text{for } i = 1,\ldots, p \tag{2.7}$$

or, $u_{\min} \le u \le u_{\max}$, in vector form. Constraints originate from position and/or rate limits of the actuators. Given the limits, an exact solution may not exist, despite the redundancy. Further, whether an exact solution exists or not, the solution can generally *not* be assumed to be unique. We refer to the problem of finding a vector *u* that is the "best" possible solution of (2.6) within the constraints (2.7) as the *control allocation problem*.

The control allocation problem arises in situations other than the design of model reference control laws, specifically in the context of control laws designed using the concept of pseudo-effectors [2] and reconfigurable control in general [5], [9]. To discuss one approach to reconfigurable control, assume that a nominal control law has been designed so that a control input u_{nom} is produced under the assumption of a nominal matrix CB_{nom}. If a failure occurs, a new control input must be found that accounts for the modified matrix *CB*. In order to preserve the performance of the nominal control law, it is reasonable to set as an objective that

$$(CB)u = (CB)_{nom}\, u_{nom} \tag{2.8}$$

This problem fits into the previous format if we define the vector a_d such that

$$a_d = (CB)_{nom}\, u_{nom} \tag{2.9}$$

2.2 Mathematical Formulations of Control Allocation

In light of this preliminary discussion, we propose four mathematical formulations of the control allocation problem. These formulations take into account the fact that the exact solution of the control allocation problem may not exist, and that the solution may not be unique.

Direct Allocation Problem: given a matrix *CB*, find a real number ρ and a vector u_1 such that $J=\rho$ is maximized, subject to

$$(CB)u_1 = \rho a_d \tag{2.10}$$

and $u_{\min} \le u \le u_{\max}$. If ρ>1, let $u = u_1/\rho$. Otherwise, let $u = u_1$.

Error Minimization Problem: given a matrix *CB*, find a vector *u* such that

$$J = \|CBu - a_d\| \tag{2.11}$$

is minimized, subject to $u_{\min} \le u \le u_{\max}$.

Control Minimization Problem: given a matrix *CB* and a vector u_p, and given a vector u_1 such that $u_{\min} \le u_1 \le u_{\max}$, find a vector *u* such that

$$J = \|u - u_p\| \tag{2.12}$$

is minimized, subject to

$$(CB)u = (CB)u_1 \tag{2.13}$$

and $u_{\min} \le u \le u_{\max}$.

Mixed Optimization Problem: given a matrix *CB* and a vector u_p, find a vector *u* such that

$$J=\|CBu - a_d\| + \varepsilon\|u - u_p\| \tag{2.14}$$

is minimized, subject to $u_{\min} \le u \le u_{\max}$.

2.3 Discussion of the Direct Allocation Problem

The *direct allocation problem* was proposed by Durham [4]. The objective is to find a control vector *u* that results in the best approximation of the vector a_d, *in the given direction*. The implicit assumption is that directionality is an important characteristic of multivariable control systems, and of flight control in particular.

Numerical algorithms for direct allocation have been proposed. The original algorithm [4] was slow and difficult to implement, but an elegant approach [10] reduced the number of computations considerably. A fast implementation using spherical coordinates and look-up tables was proposed in [11], although it requires a good number of off-line computations. Another fast technique was proposed in [12]. Unfortunately, the algorithm is hard to reproduce, due to lack of details, and it is not guaranteed to converge to the solution. Linear programming algorithms discussed later in this paper avoid most of these problems.

An interesting feature of the direct allocation problem is that its solution is unique under a relatively mild condition on the matrix *CB*. Specifically, the condition is:

Any q columns of the CB matrix are linearly independent (linear independence condition)

where q is the number of rows of CB. Durham considered the problem where $q=3$, which is typical of flight control. In that case, the *3* components of a_d in the model reference control law with q, p, and r as outputs are three desired rotational accelerations (Durham refers to them as moments, which corresponds to a slightly different formulation). The columns of the CB matrix then represent the contributions of the individual actuators towards these desired accelerations. An interpretation of the linear independence condition is that *no three actuators produce coplanar acceleration vectors.*

In many applications, the linear independence condition is satisfied. However, there are cases where it is not. In general, the accelerations produced by pitch thrust vectoring and by two symmetric pitch surfaces (such as elevators) are coplanar. Similarly, a failed actuator yields a zero column in the B matrix, which automatically violates the linear independence condition. It has been argued that the CB matrix could be perturbed by small numbers to enforce the condition, but this fix is likely to induce numerical sensitivity. Nevertheless, the direct allocation concept can be extended to systems that do not satisfy the linear independence condition, and an extension of the method to the coplanar case is discussed in [13].

Another difficulty of the direct allocation formulation is that it requires

$$u_{\min} \le 0 \qquad \text{and} \qquad u_{\max} \ge 0 \tag{2.15}$$

(to be interpreted as vector inequalities), which makes application difficult in the case of rate-limited actuators. A solution to this problem consists in applying the technique to increments of the vector u. However, this solution introduces a wind-up problem, which must be resolved using "restoring" techniques. The direct allocation method has also been criticized for not enabling axis prioritization. However, it presents interesting advantages, including a guarantee of maximum control utilization combined with the existence of fast computational procedures.

2.4 Discussion of the Error Minimization, Control Minimization, and Mixed Optimization Problems

The *error minimization problem* is the most commonly encountered formulation of control allocation. The l_2 norm or Euclidean norm is typically used in (2.11), although the l_1 norm has also been proposed in order to use linear programming techniques. Often, a weighting matrix is inserted in the norm to prioritize the axes.

The *control minimization problem* is a secondary optimization objective to be satisfied if the solution of the primary objective is not unique. The vector u_p represents some preferred position of the actuators (*e.g.*, zero deflections). After a solution yielding minimum error is obtained, the solution of minimum deviation from the preferred position is picked among all equivalent solutions. A weighting matrix may also be incorporated in the norm, to prioritize the actuators.

The control minimization objective has been considered as part of a multi-branch, or two-stage, optimization algorithm [3]. Specifically, the secondary optimization problem was solved (only) when the solution of the primary error minimization problem was $J=0$. Indeed, the solution is typically not unique in that case. One should note, however, that if the linear independence condition is not satisfied, the solution of the primary objective is generally not unique, regardless of the feasibility of the desired vector a_d. Therefore, it is reasonable to perform control minimization no matter what the result of the primary error minimization step is.

The *mixed optimization problem* combines the error and control minimization problems into a single problem, through the use of a small parameter ε. If the parameter ε is small, priority will be given to error minimization over control minimization, as desired. Often, the combined problem may be solved faster than the error and control minimization problems solved sequentially, and with better numerical properties.

3. Three Simple Algorithms for Control Allocation

The formulations of control allocation represent them as constrained nonlinear optimization problems. Because of the limited number of variables and the convexity of the constraint set, the optimization problems are simple, from a modern computational perspective. However, computations must be performed at high rate (around 100 Hz) and in the midst of many other control computations. Further, predictable operation and reliability are required for safety-critical systems, and human intervention is not possible. For those reasons, the most common implementations of control allocation have relied on approximations of the error minimization objective. We discuss three approaches that have been proposed.

3.1 Redistributed Pseudo-Inverse

The *redistributed pseudo-inverse method* of [2], [5] begins by finding the control vector *u* that minimizes

$$J = \|u\|_2^2 \tag{3.1}$$

subject to

$$(CB)u = a_d \tag{3.2}$$

This optimal vector is given through the pseudo-inverse of the *CB* matrix, with

$$u = (CB)^T \left[(CB)(CB)^T \right]^{-1} a_d \tag{3.3}$$

If the control vector satisfies the constraints, the algorithm stops. Otherwise, the components of the control vector that exceed the limits are clipped at their allowable values, and the inverse is re-computed with the components that did not reach their limits. The procedure is repeated until all components have reached their limits, or until the solution of the reduced least-squares problem satisfies the constraints. Computations may be performed by modifying a_d with the contributions of the saturated controls, and zeroing the columns of the *CB* matrix associated with these controls. Because the modified matrix $(CB)(CB)^T$ will typically become singular at some point, it is necessary to replace it by a regularized matrix $(CB)(CB)^T + \varepsilon I$.

The redistributed pseudo-inverse method is very simple, and very effective. However, it does not guarantee full utilization of the actuators' capabilities. For example, consider the desired vector and parameters

$$a_d = \begin{pmatrix} 0 \\ 9 \\ 0 \end{pmatrix}, \quad CB = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{pmatrix} \tag{3.4}$$

with the actuator limits $u_{\max} = -u_{\min} = \begin{pmatrix} 5 & 10 & 2 & 1 \end{pmatrix}$. The solution of the unconstrained least-squares is $u_{LS}^T = \begin{pmatrix} 0 & 6 & -3 & 3 \end{pmatrix}$, which gives the clipped solution $u^T = \begin{pmatrix} 0 & 6 & -2 & 1 \end{pmatrix}$. At the second and final iteration, the control vector $u^T = \begin{pmatrix} 0 & 8 & -2 & 1 \end{pmatrix}$ is obtained. The acceleration vector $a^T = \begin{pmatrix} 0 & 9 & -1 \end{pmatrix}$ is achieved, instead of the desired vector. However, the desired vector a_d may be attained using only the second control variable, specifically

$$u^T = \begin{pmatrix} 0 & 9 & 0 & 0 \end{pmatrix} \quad \Rightarrow \quad a^T = \begin{pmatrix} 0 & 9 & 0 \end{pmatrix} \tag{3.5}$$

What this example shows is that some bad choices may be made early in the iterations of the method, which cannot be recovered from, and which prevent full utilization of the control authority.

3.2 Quadratic Programming

Quadratic programming generally refers to the numerical solution of the optimization problems with an l_2 norm. Enns [1] proposed several approaches, including approximate methods with low numerical requirements. We consider one of the simplest algorithms here, which begins by finding the control vector *u* that minimizes

$$J = \left\| CBu - a_d \right\|_2^2 \tag{3.6}$$

The solution is again given by the pseudo-inverse, and if the vector satisfies the constraints, the algorithm stops. Otherwise, the method continues in a manner different from the redistributed pseudo-inverse. The optimal solution subject to the equality constraints

$$\left\| u \right\|_2^2 = p \tag{3.7}$$

is computed, where *p* is the dimension of *u*. This step assumes that an affine transformation has been applied to the data of the problem in order to replace the constraint $u_{\min} \le u \le u_{\max}$ by $\left\| u \right\|_\infty = 1$. Then, the l_2 constraint $\left\| u \right\|_2^2 = p$ is applied as an approximation of the box constraint $\left\| u \right\|_\infty = 1$ (the l_2 constraint is such that the sphere contains the box while touching its corners).

The solution to the problem with the equality constraint is

$$u = \left(CB \right)^T \left[\left(CB \right) \left(CB \right)^T + \lambda I \right]^{-1} a_d \tag{3.8}$$

which is reminiscent of the regularized pseudo-inverse. However, λ is not a small constant, but a Lagrange multiplier that must be found by solving the equation

$$\gamma(\lambda) = \left\| \left(CB \right)^T \left[\left(CB \right) \left(CB \right)^T + \lambda I \right]^{-1} a_d \right\|_2^2 = p \tag{3.9}$$

It turns out [1] that the function γ(λ) is monotonically decreasing, and that the solution λ may be found using only a few iterations of a bisection method. The control vector *u* is then clipped to satisfy the constraints.

The advantage of the method is its numerical simplicity. However, the replacement of the l_∞ constraint by an l_2 constraint implies that the solution is not exact. Clipping must be applied to enforce the actual constraint, and may

yield poor results. Also, the origin of the transformed set corresponds to a non-zero vector whenever some surfaces have non-symmetric limits (for example, spoilers). As a result, the control surface deviations produced by the algorithm are usually offset towards undesirable values. While this problem can be addressed by the use of additional steps in the algorithm (secondary optimization), the algorithm then becomes more complex and less attractive.

3.3 Fixed-Point Method

The *fixed-point method* of [14] finds the control vector u that minimizes

$$J = (1-\varepsilon)\left\|(CB)u - a_d\right\|_2^2 + \varepsilon\left\|u\right\|_2^2 \tag{3.10}$$

subject to $u_{\min} \le u \le u_{\max}$. This problem is a special case of the mixed optimization with an l_2 norm and u_p=0. The algorithm proceeds by iterating on the equation

$$u_{k+1} = sat\left[(1-\varepsilon)\eta(CB)^T a_d - (\eta M - I)u_k\right] \tag{3.11}$$

where

$$M = (1-\varepsilon)(CB)^T(CB) + \varepsilon I, \quad \eta = \frac{1}{\left\|M\right\|_2} \tag{3.12}$$

and *sat* is the saturation function that clips the components of the vector u to their allowable values.

The fixed-point algorithm is extremely simple, and much of the computations need to be performed only once, before iterations start. Remarkably, the algorithm also provides an *exact* solution to the optimization problem, and it is guaranteed to converge. Its drawback is that convergence of the algorithm may be very slow and strongly depends on the problem (the number of iterations required may vary by orders of magnitude depending on the desired vector). In addition, the choice of the parameter ε is delicate, as it affects the trade-off between the primary and the secondary optimization objectives, as well as the convergence of the algorithm.

4. Formulation of Control Allocation Problems as Linear Programs

4.1 Linear Programs

Buffington [3] showed that the error minimization and the control minimization problems could be reformulated as linear programs, assuming that the norm used was the l_1 norm. The problems could then be solved *exactly*, using standard linear programming software. A standard linear programming (LP) problem consists in finding a vector x such that

$$J = c^T x \tag{4.1}$$

is minimized, subject to

$$0 \le x \le h, \quad \text{and} \quad Ax = b \tag{4.2}$$

In this equation, vector inequalities are to be interpreted element-by-element. Alternative formulations exist, replacing $0 \le x \le h$ by $x \ge 0$, and $Ax = b$ by $Ax \ge b$. However, these differences are not significant and the present

form will be adequate for our discussion. Note that the matrix *A* and the vector *x* are *not* the same as the state-space variables defined earlier. This slight abuse of notation will enable us to use of standard LP notation.

4.2 Conversion of the Direct Allocation Problem to an LP Problem

The conversion of the control allocation problems to LP problems is not immediately obvious, and several formulations may be derived. It is generally desirable to obtain a formulation with as few rows in the matrix *A* as possible, and we present an approach that requires the smallest number of rows. Direct allocation specifies *q* linear equations

$$a_{d,i} = \rho CB_i u, \quad i = 1,\ldots,q \tag{4.3}$$

where CB_i is the *i*th row of CB, and $a_{d,i}$ is the *i*th element of a_d. Re-ordering the rows of CB and of a_d so that the first element of a_d is the one with the largest magnitude, and eliminating the variable ρ, gives a new system of q linear equations

$$\left(a_{d,i} CB_1 - a_{d,1} CB_i\right) u = 0, \quad i = 2,\ldots q \tag{4.4}$$

which may be written in matrix form as $M \cdot CB \cdot u = 0$, with

$$M = \begin{pmatrix} a_{d,2} & -a_{d,1} & 0 & \ldots & 0 \\ a_{d,3} & 0 & -a_{d,1} & \cdots & 0 \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ a_{d,q} & 0 & 0 & \cdots & -a_{d,1} \end{pmatrix} \tag{4.5}$$

These equality constraints are equivalent to the original conditions if $a_{d,1} \neq 0$. Unless $a_d = 0$ (a trivial case requiring *u*=0), this situation will not be encountered.

Define $x = u - u_{\min}$, so that $0 \le x \le u_{\max} - u_{\min}$ and notice that the optimization criterion may be written as

$$\max_u \left(\rho = \frac{\|CBu\|}{\|a_d\|} \right) = \min_u \left(J = -a_d^T CBu \right) \tag{4.6}$$

given that *CBu* is proportional to a_d. Then, the LP problem may be defined by the following matrices

$$\begin{aligned} &A = M \cdot CB, \quad b = -Au_{\min} \\ &c^T = -a_d^T CB, \quad h = u_{\max} - u_{\min} \end{aligned} \tag{4.7}$$

If a solution is found to the LP problem, the solution to the direct allocation problem may be obtained using

$$\begin{aligned} &u = x + u_{\min}, \quad \rho = a_d^T \, CBu / \|a_d\|^2 \\ &u = u / \rho \ \text{ if } \rho > 1 \end{aligned} \tag{4.8}$$

Note that the direct allocation problem may be solved as an LP problem for an arbitrary number of rows in the matrix *CB*, as opposed to the algorithms proposed earlier [11], [12]. In particular, direct allocation could be performed for forces as well as moments in a flight control application that demanded it.

In the standard direct allocation problem where the number of rows of *CB* is 3, the number of rows of the matrix *A* is only 2, which makes the size of the LP problem very small. Linear programming theory implies that optimal solutions to the problem will be such that all variables except 2 will be at their limits (before the final division by ρ). This observation was made earlier by Durham using different arguments. Linear programming theory also indicates that even if the linear independence assumption is not satisfied, at least one of the optimal solutions will be such that all variables except 2 are at their limits.

4.3 Conversion of the Mixed Optimization Problem to an LP Problem

The error minimization and control minimization problems can also be converted to LP problems, as shown in [3]. This conversion requires that the norm used in the optimization criterion is the l_1 norm. Interestingly, various LP problems may be obtained for the same control allocation problem, and formulations smaller than those of [3] may be obtained. Also, from multiple investigations, we have concluded that the mixed optimization problem yielded the most useful formulation, and one that could be solved most effectively. Therefore, we only discuss this problem.

First, we define the function $s(x)$

$$\begin{aligned} s(x) &= x \quad \text{if } x > 0 \\ &= 0 \quad \text{otherwise} \end{aligned} \tag{4.9}$$

This function is to be interpreted element-by-element in the vector case. We also assume that the preferred vector satisfies $u_{\min} \le u_p \le u_{\max}$. This condition may be eliminated easily, once the technique is understood. Define

$$u^+ = s(u - u_p), \quad u^- = -s(u_p - u) \tag{4.10}$$

so that

$$\begin{aligned} &u = u^+ - u^- + u_p \\ &0 \le u^+ \le u_{\max} - u_p,\ 0 \le u^- \le u_p - u_{\min} \end{aligned} \tag{4.11}$$

Similarly, define

$$e = CBu - a_d, \quad e^+ = s(e), \quad e^- = -s(-e) \tag{4.12}$$

so that

$$e = e^+ - e^-, \quad 0 \le e^+ \le e_{\max}, \quad 0 \le e^- \le e_{\max} \tag{4.13}$$

where $e_{\max}$ is some upper bound on the achievable error, *e.g.*, $e_{\max} = \left\| CBu_p - a_d \right\|_1$.

With these definitions, the optimization problem involves a system of n linear equations

$$e^+ - e^- - CBu^+ + CBu^- = CBu_p - a_d \tag{4.14}$$

and the cost criterion

$$J = \sum_{i=1}^{q} e_i^+ + \sum_{i=1}^{q} e_i^- + \varepsilon \sum_{i=1}^{p} u_i^+ + \varepsilon \sum_{i=1}^{p} u_i^- \tag{4.15}$$

Therefore, defining the vector $x^T = \begin{pmatrix} e^+ & e^- & u^+ & u^- \end{pmatrix}$, the LP problem is specified by

$$\begin{aligned} A &= \begin{pmatrix} I & -I & -CB & CB \end{pmatrix}, \quad b = CBu_p - a_d \\ c^T &= \begin{pmatrix} 1 & \dots & 1 & \varepsilon & \dots & \varepsilon \end{pmatrix} \\ h^T &= \begin{pmatrix} e_{\max} & e_{\max} & u_{\max} - u_p & u_p - u_{\min} \end{pmatrix} \end{aligned} \tag{4.16}$$

Note that the *A* matrix of the LP problem has as many rows as the *CB* matrix (one more row than for direct allocation). Arbitrary problems can be solved, but for the standard 3-moment case, the dimension is only 3. The problem is therefore very manageable, and only slightly more complex than the direct allocation problem. As for direct allocation, linear programming theory implies certain properties of the solution. All the components of the optimal vector *x* except 3 will be at either the upper limit or the lower limit. In terms of the control vector, this property implies that all but three control variables will be either at the upper limit, or at the lower limit, or at the preferred position. If the vector a_d cannot be achieved in any direction, all the control variables will be at one of the limits or at the preferred positions. The desirability of the property may be debated: on the one hand , it is desirable to see the algorithm use only effective surfaces [2], and on the other hand, it is desirable to see all surfaces move together to achieve the desired moment [15]. It has been argued that the property is detrimental to on-line identification, since some surfaces may not be used at all. However, we will see in the numerical section of this paper that the conclusion is only valid if the commanded accelerations are small.

5. Implementation using the Simplex Algorithm

5.1 Simplex Algorithm

A well-established method for solving LP problems is the *simplex algorithm* [16]. The algorithm is guaranteed to find an optimal solution in a finite period of time, it is easy to code, and it works well in practice. In the case of the direct allocation problem, the simplex algorithm applied to the associated LP problem constitutes an alternative to the implementation of [12], with the benefit that it guarantees a solution in a finite period of time. The simplex algorithm can also be applied to the mixed optimization problem. Speed of execution can be maximized by taking advantage of the particular aspects of the LP problem.

An important definition in the context of the simplex algorithm is that of a basic feasible solution. A vector *x* is called a *feasible solution* of the LP problem if $0 \le x \le h$ and $Ax = b$. A vector *x* is called a *basic feasible solution* if it is a feasible solution and $n - m$ of its variables are at their upper or lower limits, where *m* is the number of rows of *A* and *n* is the number of columns of *x* (*n* is also the number of elements of *x*). It can be shown that if there is a unique optimal solution, it is a basic feasible solution. Further, if there exists a set of optimal solutions, at least one of the solutions is a basic feasible solution.

The idea of the simplex algorithm is to start from a basic feasible solution and, at every step, to find a new basic feasible solution with a lower cost. From one step to the next, only one of the variables that is at a limit is swapped with one that is not. When it is not possible to decrease the cost, the algorithm stops. It can be proved that the resulting solution is optimal. Because the possible number of elements in the set of basic feasible solutions is finite, the number of iterations is bounded by a known quantity. Specifically, the number of iterations is at most

$\frac{n!}{m!(n-m)!}$. For a direct allocation problem in a 3-dimensional space with *p* control variables, *n=p* and *m=2*. For a mixed optimization problem in a 3-dimensional space with *p* control variables, *n=2p+6* and *m=3*. Practically, however, convergence occurs well before the maximum number of iterations is reached.

Various implementations of the simplex algorithm are possible. However, it may be noted that the control allocation problems under consideration are such that $n \gg m$. Therefore, the problems are well suited for the so-called *revised simplex method* [16]. The number of computations in that method depend only moderately on the number of columns *m*. Also, most variables of the vector *x* naturally have both upper and lower bounds (which is the reason why we defined the LP problem with both bounds, although the problem with only lower bounds is more commonly encountered). If an LP code only handles lower bounds of the type $x \geq 0$, a constraint $0 \leq x_1 \leq h$ can be handled by adding a so-called *slack variable*, with the constraints $x_1 \geq 0$, $x_2 \geq 0$, and $x_1 + x_2 = h$. However, this approach increases the number of equality constraints by a particularly large number for the problems under consideration. For direct allocation, *m* would become *p+2* instead of *2*, and *n* would become *2p* instead of *p*. The variable *p* is the number of actuators, and therefore $p \gg 2$. Instead of using slack variables, an implementation of the algorithm with upper and lower bounds avoids this problem [17]. It appears that the revised simplex algorithm with upper and lower bounds may be close to the one used in [19], although too few details are available in the paper to be sure.

5.2 Refinements to the Algorithm

Several refinements can be used to accelerate execution. First of all, note that the algorithm must be initialized with a basic feasible solution. Typically, this is achieved during an initialization phase which, itself, requires the use of the simplex algorithm. A vector x_S of slack variables is introduced so that $Ax + x_s = b$, and the algorithm is used to eliminate x_S by setting $J = \|x_S\|$. For the direct allocation problem, this step requires the preliminary solution of an LP problem with *n=p+2* and *m=2* variables.

Interestingly, the algorithm for mixed optimization can be initialized directly, by-passing the need for a two-step procedure. Specifically, the algorithm may be initialized with $u=u_p$, so that

$$\begin{aligned} u^+ &= 0, \quad e^+ = s(CBu_p - a_d) \\ u^- &= 0, \quad e^- = -s(-CBu_p + a_d) \end{aligned} \tag{5.1}$$

Note that the resulting vector *x* has all but *n* of its elements at zero (at least), and solves *Ax=b*. Therefore, it is a basic feasible solution. Because of this special feature, we have found that the mixed optimization problem could be solved in approximately the same time as the direct allocation problem, even though the dimension of the LP problem is larger.

Another useful technique to speed-up computations is the Sherman-Morrison-Woodbury formula [20]. The revised simplex method requires the inversion of the square matrix obtained by taking the columns of the *A* matrix that correspond to the non-basic variables (the variables that are not at the limits). However, only one column of the matrix changes from one iteration to the next. Therefore, the inverse of the matrix can be computed recursively

using the inverse at the previous iteration, requiring much fewer computations than the full matrix inverse. This algorithm is particularly attractive if the matrix has a known inverse at the initial step. Fortunately, modulo some sign changes, the initial matrix is simply the identity matrix for the mixed optimization problem.

Another issue to be dealt with is the fact that the finite-time convergence of the simplex algorithm is based on the assumption that the cost is monotonically decreasing. If such is not the case, the algorithm may cycle among a set of basic feasible solutions having identical costs. Cycling is a singular condition that requires two real variables to be exactly equal. Although one would expect this situation to be extremely rare, it was rapidly encountered in our experiments of control allocation. The reason was that the aircraft models had identical control derivatives for symmetric surfaces, and that a large number of cases were computed in a small period of time. Fortunately, anticycling procedures have been studied that help to deal with this problem (see, in particular, [18] and [21]). Some work is required to adapt the methods to the revised simplex method with upper and lower bounds.

6. Numerical Results

6.1 General Conditions of the Numerical Evaluation

The methods of control allocation were evaluated numerically on several examples. Three aircraft models were used. The first model is a C-17 model with 8 actuators, where four of the actuators represent ganged surfaces. The control effectors are left elevators, right elevators, a left aileron, a right aileron, a lower rudder, an upper rudder, left spoilers, and right spoilers. The second model is the same C-17 model, but with no ganging, resulting in 2 left elevators, 2 right elevators, a left aileron, a right aileron, a lower rudder, an upper rudder, 4 left spoilers, and 4 right spoilers. The third model is the model of a tailless aircraft found in [6]. It is based on Lockheed's ICE (*innovative control effectors*) model, and has 11 actuators: left elevon, right elevon, pitch flaps, left all-moving tip, right all-moving tip, pitch thrust vectoring, yaw thrust vectoring, left spoiler slots, right spoiler slots, left outboard leading-edge flaps, and right outboard leading-edge flaps.

The C-17 model satisfies the linear independence condition (*i.e.*, it does not have coplanar controls). The tailless model does not: pitch thrust vectoring and pitch flaps produce pure pitching accelerations, and therefore the vectors are linearly dependent. Each of these control variables is also linearly dependent on the left and right elevons. As a result, the control variables needed to produce an acceleration vector are not unique, whether the vector is feasible or not. Given a control allocation solution, for example, it is easy to obtain a different one where the effect of a deflection of pitch flaps is canceled by pitch thrust vectoring.

Although rate limits may easily be incorporated in the mixed l_1 optimization method, the analysis of this paper only considers position limits, in order to make the comparisons more tractable and meaningful. The results assess the ability of the algorithms to reach the set of attainable accelerations. The interactions between the algorithms and a closed-loop flight control system are not addressed (see [22] for some results in that direction).

The parameters used in the algorithms were as follows. In the redistributed pseudo-inverse, $\varepsilon = 10^{-4}$. In the quadratic programming method, a tolerance of 10^{-6} was used for the stopping rule of the bisection method in the γ equation. For the fixed-point method, the number of iterations was set at 50, with $\varepsilon = 10^{-3}$. Note that the ε parameter must be large enough to get good convergence properties, and small enough that the control minimization

objective remains secondary to the error minimization objective. In the mixed optimization objective, ε does not have an impact on convergence, although it should not be too small with respect to the numerical precision of the machine. A value of 10^{-6} was chosen.

The results were obtained on a Pentium III machine running at 500 MHz. Most results were obtained using implementations of the algorithms as m-files in Matlab 5.3. Timing results were obtained using the tic/toc commands, but are given for comparison purposes only. To obtain timing information for the simplex algorithm, the mixed optimization was coded in Fortran, yielding much faster execution times.

6.2 C-17 Model

Table 1 shows the results for the C-17 model with 8 actuators. The five methods discussed in the paper are listed: the redistributed pseudo-inverse technique, the quadratic programming algorithm, the fixed-point method, the direct allocation algorithm based on the simplex, and the mixed optimization algorithm using the simplex. The table gives the results obtained for 1000 randomly distributed vectors. The components of the vectors were evenly distributed in the range [±40, ±20, ±5] deg/s^2 (the vectors are associated with rotational accelerations in the pitch/roll/yaw axes). A procedure similar to the one described in [10], [11] was used to plot the set of achievable accelerations, and the ranges were selected by visual inspection of the set, so that all the vectors in the set would be feasible.

The columns of Table 1 give the values of the average error $\|CBu - a_d\|_2$ over 1000 points, the maximum error $\|CBu - a_d\|_2$, the average execution time, and the average norm of the control $\|u\|_2$. Since u_p was set at zero, the last column is a measure of the secondary performance objective. Note that the norm that was used for evaluation was the l_2 norm, which favors the first three methods (the mixed optimization method minimizes the l_1 norm).

One may first notice that the average and maximum errors are zero for the direct allocation and mixed optimization methods, indicating that the random vectors were indeed attainable. The average errors for the other methods are small, but nonzero. Part of the error for the fixed-point method is due to the trade-off between the error minimization and the control minimization. Interestingly, the mixed optimization method using the simplex algorithm (which is based on the l_1 norm) does *not* yield such a trade-off here. Note that while the direct allocation method is guaranteed to find an exact solution, if one exists, the same is not necessarily true for the mixed optimization approach. Nevertheless, the error $\|CBu - a_d\|_2$ is typically zero when the desired vector is feasible, and the size of the error can be shown to be at most of the order of ε. A control surface will not be moved by 1 degree if the benefit in terms of the error is less than ε deg/s^2. Conversely, the error will be made exactly zero if the required control deflection is not excessive. As a result, for problems with minimally effective surfaces, ε small, and feasible vectors, the error is zero with the mixed optimization method.

An alternative to the mixed optimization algorithm is the two-stage algorithm of [3], [24], where the first stage minimizes the error and the second stage minimizes the control deflection. Both problems may be solved as linear programs. With this approach, any realizable vector will be exactly achieved. However, the computational load is significantly higher, and the potential benefits in terms of the error are insignificant. Nevertheless, it has been found

helpful to use the intermediate results of this approach in order to reduce reference command levels when acceleration limits were reached.

The column with the maximum errors shows a large maximum error for the redistributed pseudo-inverse. In other experiments, even larger errors were found. For example, the following pair was found

$$\begin{aligned} a_d &= \begin{pmatrix} 23.7 & 12.7 & -4.9 \end{pmatrix} \\ \Rightarrow a &= \begin{pmatrix} 0.19 & 18.6 & -3.9 \end{pmatrix} \end{aligned} \tag{6.1}$$

yielding an error norm of 24.2 deg/s^2. Yet, the desired vector is exactly feasible! This example, as the one in section 3.1, is such that the algorithm makes a bad choice in one of the iterations, and never recovers. Analyzing the example, it was found that the control vector applied a differential elevator deflection, while realization of the pitching moment required full symmetric deflection. It was also found that such large errors occurred in only small subsets of the acceleration space. Indeed, the average error is very small. In practice, it is likely that such large but infrequent errors would not degrade significantly the closed-loop performance, although they could be the source of glitches in the control variables such as those observed in [2].

The timing data should be interpreted with caution, since it was obtained with an interpreted language. However, it is useful for comparison of the methods. The timing information shows that the computational requirements of both optimization methods are within an order of magnitude of the simpler methods. Requirements for direct allocation and mixed optimization are also comparable. Although direct allocation involves a smaller LP problem, the mixed optimization by-passes the initialization step.

The last column of Table 1 shows the control deflections required to achieve the desired moments. Larger numbers are found for quadratic programming and for direct allocation. Regarding quadratic programming, the single-pass algorithm considered here does not minimize the norm of the control vector, and non-symmetric control surfaces such as spoilers offset the solution towards a non-zero vector. A secondary optimization could be performed, but would take away from the simplicity of the method. Regarding direct allocation, the larger control deflection are an inherent feature of the method, which does not attempt to minimize the control deflections.

Table 2 is similar to Table 1, except that a larger set of desired accelerations is used, with values in the range [±80,±40,±10] deg/s^2. This set contains infeasible accelerations, so that all errors in the table are nonzero. The improvements provided by optimization are noticeable in the table (10% to 20% on the average). Recall that the norm used for comparison is the l_2 norm, so that the comparison favors the first three methods. Timing shows slightly larger numbers than for the feasible case, but overall, the trends are similar to the previous case.

Table 3 and Table 4 are similar to Table 1 and Table 2, respectively. The C-17 model was replaced by the original model without ganging of the actuators. Although the number of actuators doubled, the timing information indicates a computational increase that is quite less than a factor of two. Other trends in the data are generally similar to the previous case. Interestingly, the numbers for the average error are significantly lower in Table 4 than in Table 2. For example, the average error provided by the mixed optimization is 5.98 deg/s^2 instead of 7.78 deg/s^2, a reduction of almost 25%. This increase in performance is solely due to the "unganging" of the spoilers and elevators.

6.3 Tailless Model

Table 5 and Table 6 are similar to Table 1 and Table 2, respectively, but with the C-17 model replaced by the tailless model. The *C* matrix was the same as before, so that the vector a_d remained a pitch/roll/yaw acceleration vector. The ranges of desired vectors were [±150, ±200, ±15] deg/s^2 for the feasible case, and [±300,±400,±30] deg/s^2 for the infeasible case. The redistributed pseudo-inverse method does not exhibit the large errors as it did for the C-17 model, but this may be true only because the ranges for the feasible case were smaller in relation to the boundary of the feasible set. On the average, a 20% improvement or more is gained using mixed optimization.

Note that the tailless model is such that the linear independence condition is not satisfied. Although the original direct allocation method assumed such condition, the optimization objective may be defined without the condition. The geometry of the set of attainable vectors is different, and the solution is not unique on the boundary [13]. Nevertheless, the simplex algorithm is capable of finding (one of) the solutions in that case.

6.4 Realistic Timing Data

Table 7 addresses more realistically the issue of computational feasibility and, in particular, the two questions: (1) what is the computational requirement of the mixed optimization algorithm in a compiled language, and (2) how much larger is the worst-case requirement than the average requirement. Here, the C-17 model with 8 actuators and the tailless model with 11 actuators are used, with the feasible set as well as the infeasible set. Only the mixed optimization algorithm was implemented, because of its particular interest in light of the previous results. Evaluation was performed with 1000 random vectors. However, for this analysis, each vector was computed 10000 times to obtain reliable timing data.

Generally, the errors reported in Table 7 are comparable to the ones obtained earlier. They are not exactly identical because of some small differences in implementation and because of different random numbers. Also, the Fortran code used a single precision format. Single precision was used to demonstrate that the method was not sensitive to small numerical errors. Indeed, while small errors are found in the feasible case, they are comparable to the machine precision. Double precision was also tried with very little or no increase in computing time, and resulted in negligible error numbers for the feasible sets.

Timing data shows execution times much smaller than in the interpreted Matlab environment. The times are well within the capabilities of existing computers and future flight control systems (confirming the conclusions of [19]). The worst-case timing requirement is about 2 times the average value. Trends in the timing requirements for feasible *vs.* infeasible sets and C-17 *vs.* tailless models are comparable to those observed in Matlab. One may have confidence that the methods would exhibit comparable performance with other aircraft models, and could be implemented easily with much greater number of actuators.

6.5 Identifiability Considerations

Table 8 reflects a different issue than those addressed earlier. It considers the impact that a control allocation algorithm has on the information available to identify the dynamics of an aircraft in real-time (*e.g.,* for reconfigurable control). The analysis is superficial, but the results are interesting enough to report. It has been remarked that the actuator commands produced by most control laws do not excite the system dynamics sufficiently

for the estimation of the stability and control derivatives. For example, any control law that applies identical commands to two control surfaces makes it impossible to determine the control derivatives of the separate surfaces. In general, the signals applied to the control surfaces must be linearly independent functions of time in order for identification to be possible. As a result, a technique such as ganging or pseudo-inverse does not provide adequate information for identification. In such cases, one says that the system is not *sufficiently excited*.

Given the discrete-time history of a control vector *u(k)*, an indicator of the level of excitation provided (for the determination of the control derivatives) is the autocorrelation matrix

$$R_u = \sum_{k=1}^{N} u(k)u(k)^T \tag{6.2}$$

If any components of the control vector are linearly dependent, the matrix R_u will be singular. In general, a good measure of excitation is then the condition number of the matrix, which we denote CN(R_u). A small condition number is indicative of a large degree of linear independence between the signals, and therefore, a better conditioned identification problem.

Because the vectors a_d used for evaluation in the examples studied before were randomly generated, their time histories constituted linear independent signals. Different control allocation algorithms, however, obtained different control vectors based on these identical requirements. Table 8 shows the condition numbers for the different algorithms, and the same sets of 1000 random vectors. An additional "small" set of accelerations in the range [±8, ±4, ±1] deg/s^2 was added to the earlier feasible and infeasible sets. The C-17 model with 8 actuators was used.

As pointed out earlier, least-squares methods can be expected to give poor results, because they approximate the fixed ganging matrix of the pseudo-inverse when limits are not encountered. The l_1 optimization method can also be expected to give poor results, because it tends to use only the most effective surfaces. The control derivatives of any unused surface cannot be determined. Recent reconfigurable control experiments using these methods [23], [24], [25] have indeed showed the need for separate excitation mechanisms, which typically used random noise injection.

Table 8 confirms these observations, but adds some interesting information. Mixed optimization performs poorly for small moment vectors only. As soon as the vectors are large enough, condition numbers become very small. In that case, all surfaces are used and the special properties of mixed l_1 optimization actually become an advantage.

Interestingly, direct allocation gives good and consistent results in all three cases. Indeed, the transformation from the required moments to the surface deviations is highly nonlinear, even from small moments (the method does *not* reduce to a generalized inverse around the origin). The fact that the control vectors are scaled from the vectors needed to achieve the maximum moment in the desired direction may be the reason why the condition numbers are so uniform across the range of magnitude. The drawback, of course, is that surface deflections are much larger than necessary for feasible moments. Further, the condition numbers are still somewhat large for real-time adaptation, and a separate means of excitation may be needed to achieve satisfactory results in a practical system.

7. Conclusions

Constrained optimization methods can realistically be considered for real-time control allocation in flight control systems. Both direct allocation and mixed error/control optimization problems may be solved using algorithms from linear programming. Computation times are much less than a millisecond, and within an order of magnitude of simpler methods. Gains in performance are small (10% to 20%), but noticeable. The reassurance that any feasible requirement will be met is valuable, especially in light of the fact that large errors are occasionally observed with simpler methods.

On the downside, linear programming code such as the simplex algorithm is relatively complex. Also, although the ratio of worst-case to average time requirement is small, the variable number of iterations is a disadvantage in flight control applications. Cycling is an issue that may be avoided with anticycling procedures, but remains an issue of concern in the presence of numerical errors. Of course, other methods have their drawbacks too. Large errors are found occasionally with the redistributed pseudo-inverse, large control signals are obtained with the single-pass quadratic programming and with direct allocation, and precision and/or number of iterations vary considerably with the fixed-point method. Perhaps the best results will be obtained in practice by combining a simple method and an optimization-based method with a fixed number of iterations, choosing the best solution provided by the two algorithms. Concerns about reliability could certainly be alleviated this way.

Other optimization algorithms may be considered instead of the simplex algorithm. Interior-point methods are promising, but perhaps for reasons other than those that have made them popular in different applications. The uniform convergence properties of the algorithms towards the optimum solution make them well suited to an environment that requires a fixed bound on the number of iterations. The ability to evaluate the distance from the optimum solution (in primal-dual methods) is also valuable. However, although the algorithms require less computations than the simplex algorithm for very large problems, the picture may actually be reversed for the "small" problems of control allocation. This is a subject of current and future research.

As opposed to simpler methods of control allocation, constrained optimization methods open doors to the more complicated problems of the future. Even simple linear programming algorithms enable the incorporation of constraints between the actuators (*e.g.*, limited differential deflection of tail surfaces, bound on the sum of reaction jets in spacecraft). More advanced techniques may also extend the range of application to problems where the effectiveness of the actuators is highly nonlinear in the control deflections.

8. References

[1] D. Enns, "Control Allocation Approaches," *Proc. of the AIAA Guidance, Control, and Dynamics Conference*, Boston, MA, pp. 98-108, 1998.

[2] J. Virnig & D. Bodden, "Multivariable Control Allocation and Control Law Conditioning when Control Effectors Limit," *Proc. of the AIAA Guidance, Navigation, and Control Conference*, Scottsdale, AZ, pp. 572-582, 1994.

[3] J. Buffington, "Modular Control Law Design for the Innovative Control Effectors (ICE) Tailless Fighter Aircraft Configuration 101-3," Report AFRL-VA-WP-TR-1999-3057, Air Force Research Laboratory, Wright-Patterson AFB OH 45433-7542, 1999.

[4] W. Durham, "Constrained Control Allocation," *AIAA J. of Guidance, Control, and Dynamics*, vol. 16, no. 4, pp. 717-725, 1993.

[5] R. Eberhardt & D. Ward, "Indirect Adaptive Flight Control of a Tailless Fighter Aircraft," *Proc. of the AIAA Guidance, Control, and Dynamics Conference*, Portland, OR, pp. 466-476, 1999.

[6] J.Buffington, "Tailless Aircraft Control Allocation," *Proc. of the AIAA Guidance, Navigation, and Control Conference*, New Orleans, LA, pp. 737-747, 1997.

[7] M. Bodson & J. Groszkiewicz, "Multivariable Adaptive Algorithms for Reconfigurable Flight Control," *IEEE Trans. on Control Systems Technology*, vol. 5, no. 2, pp. 217-229, 1997.

[8] M. Bodson & W. Pohlchuck, "Command Limiting in Reconfigurable Flight Control," *AIAA J. of Guidance, Control, and Dynamics*, vol. 21, no. 4, pp. 639-646, 1998.

[9] J. Brinker & K. Wise, "Reconfigurable Systems for Tailless Fighter Aircrafts -- Restore," Report AFRL-VA-WP-TR-1999-3067, Air Force Research Laboratory, Wright-Patterson AFB OH 45433-7542, 1999.

[10] W. Durham, "Attainable Moments for the Constrained Control Allocation Problem," *AIAA J. Guidance, Control, and Dynamics*, vol. 17, no. 6, pp. 1371-1373, 1994.

[11] J. Petersen & M. Bodson, "Fast Control Allocation Using Spherical Coordinates," *Proc. of the AIAA Guidance, Navigation, and Control Conference*, Portland, OR, pp. 1321-1330, 1999.

[12] W. Durham, "Computationally Efficient Control Allocation," *Journal of Guidance, Control, and Dynamics*, vol. 24, no. 3, pp. 519-524, 2001.

[13] J. Petersen & M. Bodson, "Control Allocation for Systems with Coplanar Controls," *Proc. of the AIAA Guidance, Navigation, and Control Conference*, Denver, CO, August 2000.

[14] J. Burken, P. Lu, Z. Wu, & C. Bahm "Two Reconfigurable Flight-Control Design Methods: Robust Servomechanism and Control Allocation," *Journal of Guidance, Control, and Dynamics*, vol. 24, no. 3, pp. 482-493, 2001.

[15] D. Cameron & N. Princen, "Control Allocation Challenges and Requirements for the Blended Wing Body," *Proc. of the AIAA Guidance, Control, and Dynamics Conference*, Denver, CO, 2000.

[16] D. Luenberger, *Introduction to Linear and Nonlinear Programming*, Addison-Wesley, Reading, MA, 1984, pp. 60.

[17] *ibid.*, pp. 53.

[18] *ibid.*, pp. 78 & 83.

[19] Y. Ikeda & M. Hood, "An Application of L1 Optimization to Control Allocation," *Proc. of the AIAA Guidance, Control, and Dynamics Conference*, Denver, CO, 2000.

[20] J. Nocedal & S.J. Wright, *Numerical Optimization*, Springer, New York, 1999, pp. 605.

[21] C. Kim, *Introduction to Linear Programming*, Holt, Rinehart & Winston, New York, 1971, pp. 446.

[22] A. Page & M. Steinberg, "A Closed-Loop Comparison of Control Allocation Methods," *Proc. of the AIAA Guidance, Control, and Dynamics Conference*, Denver, CO, 2000.

[23] J. Buffington, P. Chandler, & M. Pachter, "Integration of On-line System Identification and Optimization-based Control Allocation," *Proc. of the AIAA Guidance, Control, and Dynamics Conference*, Boston, MA, pp. 1746-1756, 1998.

[24] D. Doman, A. Ngo, D. Leggett, M. Saliers, & M. Pachter, "Development of a Hybrid Direct-Indirect Adaptive Control System for the X-33," *Proc. of the AIAA Guidance, Control, and Dynamics Conference*, Denver, CO, 2000.

[25] M. Elgersma, D. Enns, S. Shald, & P. Voulgaris, "Parameter Identification for Systems with Redundant Actuators," *Proc. of the AIAA Guidance, Control, and Dynamics Conference*, Boston, MA, pp. 109-117, 1998.

9. Acknowledgements

The author would like to thank Dr. Jim Buffington and Dr. David Doman for several useful discussions on topics related to this paper. This effort was sponsored by the Air Force Office of Scientific Research, Air Force Materiel Command, USAF, under grant F49620-98-1-0013.

List of Table Captions

Table 1: Control allocation results. C-17 aircraft model with 8 actuators. Feasible set.

Table 2: Control allocation results. C-17 aircraft model with 8 actuators. Infeasible set.

Table 3: Control allocation results. C-17 aircraft model with 16 actuators. Feasible set.

Table 4: Control allocation results. C-17 aircraft model with 16 actuators. Infeasible set.

Table 5: Control allocation results. Tailless aircraft model with 11 actuators. Feasible set.

Table 6: Control allocation results. Tailless aircraft model with 11 actuators. Infeasible set.

Table 7: Control allocation results with mixed l_1 optimization. C-17 aircraft model with 8 actuators and tailless model. Feasible and infeasible sets. FORTRAN implementation.

Table 8: Identifiability results. C-17 aircraft model with 8 actuators. Infeasible set.

Method	Average error ($^o/s^2$)	Maximum error ($^o/s^2$)	Average time (ms)	Average control (^{o})
Redistributed pseudo-inverse	0.24	14.5	3.13	16.4
Quadratic programming	0.05	2.49	2.31	30.1
Fixed-point method	0.16	2.68	5.71	15.0
Direct allocation (simplex)	0.00	0.00	15.2	28.8
Mixed l_1 optimization	0.00	0.00	15.3	17.7

Table 1: Control allocation results. C-17 aircraft model with 8 actuators. Feasible set.

Method	Average error ($^o/s^2$)	Maximum error ($^o/s^2$)	Average time (ms)	Average control (^{o})
Redistributed pseudo-inverse	10.1	48.0	4.12	41.7
Quadratic programming	10.5	39.5	4.50	40.2
Fixed-point method	8.97	34.6	6.37	34.0
Direct allocation (simplex)	9.00	41.4	17.0	48.0
Mixed l_1 optimization	7.79	35.7	20.2	44.7

Table 2: Control allocation results. C-17 aircraft model with 8 actuators. Infeasible set.

Method	Average error ($^o/s^2$)	Maximum error ($^o/s^2$)	Average time (ms)	Average control (^{o})
Redistributed pseudo-inverse	0.09	14.0	4.12	21.4
Quadratic programming	0.18	3.28	2.58	49.3
Fixed-point method	0.12	1.89	7.25	20.2
Direct allocation (simplex)	0.00	0.00	18.7	39.9
Mixed l_1 optimization	0.00	0.00	17.9	26.2

Table 3: Control allocation results. C-17 aircraft model with 16 actuators. Feasible set.

Method	Average error ($^o/s^2$)	Maximum error ($^o/s^2$)	Average time (ms)	Average control (^{o})
Redistributed pseudo-inverse	8.20	47.1	5.82	52.2
Quadratic programming	9.46	40.2	3.63	58.0
Fixed-point method	7.17	34.4	7.91	43.6
Direct allocation (simplex)	6.76	40.1	21.2	66.8
Mixed l_1 optimization	5.98	35.7	25.5	61.5

Table 4: Control allocation results. C-17 aircraft model with 16 actuators. Infeasible set.

Method	Average error ($^o/s^2$)	Maximum error ($^o/s^2$)	Average time (ms)	Average control (^{o})
Redistributed pseudo-inverse	0.00	0.02	3.79	27.7
Quadratic programming	2.07	65.5	2.47	61.0
Fixed-point method	3.26	9.19	6.70	25.8
Direct allocation (simplex)	0.00	0.00	18.1	43.1
Mixed l_1 optimization	0.00	0.00	17.4	31.9

Table 5: Control allocation results. Tailless aircraft model with 11 actuators. Feasible set.

Method	Average error ($^o/s^2$)	Maximum error ($^o/s^2$)	Average time (ms)	Average control (^{o})
Redistributed pseudo-inverse	15.3	197.	4.51	68.7
Quadratic programming	44.4	234.	2.91	79.0
Fixed-point method	17.4	158.	6.76	54.3
Direct allocation (simplex)	15.2	196.	18.2	81.4
Mixed l_1 optimization	11.5	180.	20.0	72.9

Table 6: Control allocation results. Tailless aircraft model with 11 actuators. Infeasible set.

Aircraft model	Average error ($^o/s^2$)	Maximum error ($^o/s^2$)	Average time (μs)	Maximum time (μs)
C-17 (8) Feasible set	$1.8\ 10^{-6}$	$3.9\ 10^{-4}$	84	159
C-17 (8) Infeasible set	7.6	36.1	122	220
Tailless Feasible set	$1.1\ 10^{-5}$	$2.4\ 10^{-4}$	91	193
Tailless Infeasible set	8.8	189	132	225

Table 7: Control allocation results with mixed l_1 optimization. C-17 aircraft model with 8 actuators and tailless model. Feasible and infeasible sets. FORTRAN implementation.

Method	CN(R_u) Small set	CN(R_u) Feasible set	CN(R_u) Infeasible set
Redistributed pseudo-inverse	$2.1\ 10^{16}$	$2.3\ 10^{3}$	194.
Quadratic programming	$1.5\ 10^{18}$	$3.3\ 10^{16}$	$2.9\ 10^{4}$
Fixed-point method	$3.9\ 10^{16}$	$2.6\ 10^{4}$	$2.0\ 10^{3}$
Direct allocation (simplex)	453.	425.	392.
Mixed l_1 optimization	$1.1\ 10^{30}$	98.	324.

Table 8: Identifiability results. C-17 aircraft model
with 8 actuators. Infeasible set.