CONTROL ALLOCATION WITH LOAD BALANCING

Marc Bodson[1]
University of Utah, Salt Lake City, UT 84112

and

Susan A. Frost[2]
NASA Ames Research Center, Moffett Field, CA 94035

Next generation aircraft with a large number of actuators will require advanced control allocation methods to compute the actuator commands needed to follow desired trajectories while respecting system constraints. Previously, algorithms were proposed to minimize the l_1 or l_2 norms of the tracking error and of the actuator deflections. The paper discusses the alternative choice of the l_∞ norm, or *sup* norm. Minimization of the control effort translates into the minimization of the maximum actuator deflection (min-max optimization). The paper shows how the problem can be solved effectively by converting it into a linear program and solving it using a simplex algorithm. Properties of the algorithm are also investigated through examples. In particular, the min-max criterion results in a type of load balancing, where the load is the desired command and the algorithm balances this load among various actuators. The solution using the l_∞ norm also results in better robustness to failures and to lower sensitivity to nonlinearities in illustrative examples.

I. Introduction

Control allocation is the problem of distributing control effort among multiple, redundant actuators. In conventional flight control system design, the issue is resolved through the concept of ganging. Specifically, *pseudo-effectors* v are defined so that

$$u = Gv \tag{0.1}$$

where v is the vector of pseudo-effectors, u is the vector of actuator commands, and G is a ganging matrix. For example, it is typical to define a single elevator command δ_e and a single aileron command δ_a so that

$$u \triangleq \begin{pmatrix} \delta_{el} \\ \delta_{er} \\ \delta_{al} \\ \delta_{ar} \\ \delta_r \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} \delta_e \\ \delta_a \\ \delta_r \end{pmatrix} \triangleq G \cdot v \tag{0.2}$$

where δ_{el}, δ_{er} are the left and right elevator commands, δ_{al}, δ_{ar} are the left and right aileron commands, and δ_r is the rudder command. An elevator command δ_e produces symmetric left and right elevator commands, and an aileron command δ_a produces antisymmetric left and right aileron commands. This ganging typically results in a mostly decoupled response of the aircraft from the elevator, aileron, and rudder commands to the pitch, roll, and yaw responses.

In control design for future vehicles, reasons to look for alternatives to ganging include cases where:

1) the vehicle has a large number of actuators, making it less intuitive how the ganging matrix should be defined;

---

[1] Professor, Electrical & Computer Engineering Department, 50 S Central Campus Dr Rm 3280, and AIAA Senior Member.

[2] Research Engineer, Intelligent Systems Division, M/S 269-1, and AIAA Member.

2) the vehicle has unconventional control effectors, with a significant degree of nonlinearity and of interaction between the effectors, again making it difficult to develop an intuitive solution;
3) the effectiveness of the actuators is limited, making it important to optimize their use within their position and rate limits;
4) the control system is designed to be reconfigurable or adaptive, so that control allocation must be computed in real-time.

Previous work in control allocation includes the seminal paper of Durham[11], which introduced the concept of direct allocation. Although the concept was mathematically formulated, algorithms to solve the problem numerically in the general case were only later developed[12]. In the meantime, Buffington[2] proposed an alternative formulation minimizing the norm of the error between desired and achieved commands. Using the l_1 norm, he showed how the problem could be converted to a linear program and solved *exactly*, using standard linear programming software. Ikeda and Hood[18] similarly reported the application of l_1 optimization, although with fewer details. Nevertheless, it became clear that the solution of optimal control allocation problems was feasible in real-time. In Ref. 22, it was shown that the direct allocation problem could also be solved using linear programming, and that a considerably smaller linear program could be obtained for the l_1 optimization problem, compared to Ref. 3. Timing data showed that solutions of the problem could comfortably be performed in real-time, even for large numbers of actuators, and that the optimal solution improved performance significantly over simpler, approximate methods.

Solutions of the optimal control allocation problems using the l_2 norm were also proposed, with an early solution provided through the *fixed-point method* of Ref. 4. The fixed-point algorithm was extremely simple, and many of the computations needed to be performed only once, before iterations started. Remarkably, the algorithm also provided an *exact* solution to the optimization problem and was guaranteed to converge. Numerical tests, however, showed convergence of the algorithm could be very slow and strongly depended on the problem (the number of iterations required could vary by orders of magnitude, depending on the desired command). An elegant alternative to this algorithm was proposed by Harkegård, using the theory of active sets[17]. The algorithm was very similar to the simplex algorithm used for l_1 optimization, and had the same advantage of completing in finite time and with a small number of iterations.

Interior-point methods were also studied to solve large control allocation problems, both for the l_1 norm[21] and for the l_2 norm[22]. The computational requirements of these methods scaled better with the number of actuators, but the number of actuators had to be quite large (>15) before the advantages become apparent.

Among recent work, one may note several papers considering the application of control allocation to hypersonic vehicles, including some flight tests [9, 10, 23], the problems posed by nonlinear actuator effectiveness (*e.g.,* Ref. 8), modifications to account for the dynamic response of the actuators (*e.g.,* Ref. 20), and the combination of control allocation with adaptation (*e.g.,* Ref. 25, 26). New aircraft, for example those with blended wing body configurations [7] have been identified as presenting control allocation challenges due to novel actuators, distributed actuators with low control authority, interactions between control effectors, and interactions between propulsion and control surfaces. Interestingly, control allocation is also emerging in other applications, including land and marine vehicles [13, 15, 24].

II. Optimization formulations of control allocation

A. Control allocation in model reference control

We introduce control allocation in the context of model reference control (a form of dynamic inversion). However, solutions may be used in a variety of control design methods. To state the problem mathematically, we consider the state-space model

$$\begin{aligned} \dot{x}_A &= A_A x_A + Bu + d \\ y_A &= Cx_A \end{aligned} \tag{0.3}$$

where $x_A \in \mathrm{R}^n$, $d \in \mathrm{R}^n$, $u \in \mathrm{R}^p$, $y_A \in \mathrm{R}^q$. For the control of aircraft, the states are given by the vector x_A and may include the angle of attack, the pitch rate, the angle of sideslip, the roll rate, and the yaw rate ($n=5$). The output vector y_A may contain the pitch rate, the roll rate, and the yaw rate ($q=3$). The control input vector u consists of the commanded actuator positions. In a conventional aircraft, these commands are the deflections of the two elevators, the two ailerons, and the rudder ($p=5$). The disturbance vector d represents the forces and moments that the control surfaces must cancel in order to trim the aircraft (*i.e.,* to create an equilibrium of the dynamical system).

For the purpose of example, consider a simple model reference control law. The method relies on a reference model that represents the desired dynamics of the closed-loop system

$$\dot{y}_M = A_M y_M + B_M r_M \tag{0.4}$$

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

$$\dot{y}_A = CA_A x_A + CBu + Cd \tag{0.5}$$

the objective may be achieved by setting

$$CBu = -CA_A x_A - Cd + A_M y_A + B_M r_M \triangleq a_d \tag{0.6}$$

where a_d represents the desired vector to be matched by CBu. If y is a vector composed of the rotational rates (as is typically the case), a_d represents desired rotational accelerations.

Obtaining u from a_d requires that one solve a system of linear equations with more unknowns than equations. Solving such a system is easy, but the difficulty in control allocation is that the vector u is constrained. The limits generally have the form

$$u_{\min,i} \le u_i \le u_{\max,i} \qquad \text{for } i = 1,...,p \tag{0.7}$$

or, $u_{\min} \le u \le u_{\max}$, in vector form. There may be additional constraints due to the maximum rate of deflection of the actuators. We refer to the problem of finding a vector u that is the "best" possible solution of (0.6) within the constraints (0.7) as the *control allocation problem.*

Given the constraints, the control allocation problem may be such that:

- many solutions exist,
- only one solution exists,
- no exact solution exists.

One is naturally drawn to finding solutions that minimize the error CBu-a_d. Indeed, providing all the control authority available may make the difference between a maneuver being achievable or not, and between an unusual condition being recoverable from or not. However, the question also arises as to which solution is the most desirable when many solutions exist. Therefore, control allocation typically consists both in *error minimization* and *control optimization.*

B. Formulations of control allocation

The fundamental control allocation problem can be formulated as the following error minimization objective.

Error minimization: given a matrix CB, find a vector u such that

$$J = \left\| CBu - a_d \right\| \tag{0.8}$$

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

The problem is solved exactly if $J=0$. However, regardless of whether an exact solution exists, the following control minimization problem may be considered as well.

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

$$J = \left\| u - u_p \right\| \tag{0.9}$$

is minimized, subject to

$$(CB)u = (CB)u_1 \tag{0.10}$$

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

The control minimization problem is a secondary optimization objective to be satisfied if the solution of the primary objective, given by u_1, 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 with minimum deviation from the preferred position is picked among all equivalent solutions. For both problems, weighting of the elements of the vectors may be inserted in the norms, either to prioritize the axes or to prioritize the actuators.

The norm used in the optimization criteria is a design choice that has more consequences than might be expected. The l_1 norm of a vector x is the sum of the absolute values of the elements of the vector

$$\|x\|_1 = \sum_{i=1}^{n} |x_i| \tag{0.11}$$

while the l_2 norm is the usual Euclidean norm

$$\|x\|_2 = \sqrt{\sum_{i=1}^{n} |x_i|^2} \tag{0.12}$$

Algorithms have been proposed for both norms and the results of the optimization problems are sometimes quite different.

A possible implementation of optimization for control allocation consists in the sequential minimization of the error vector and of the control vector. Specifically, the error is minimized first, and then the control vector is minimized among all equivalent solutions. In Ref. 3, the control minimization problem was solved only when the solution of the primary error minimization problem was $J=0$. However, it should be noted that, unless the matrix CB satisfies specific conditions (any $q \times q$ submatrix of CB must be nonsingular), the solution is not necessarily unique, even if the desired vector a_d is not feasible. Given this fact, mixed optimization makes sense, and has several advantages over sequential optimization.

Mixed optimization: Given a matrix CB and a vector u_p, find a vector u such that

$$J = \|CBu - a_d\| + \varepsilon \|u - u_p\| \tag{0.13}$$

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

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 is given to error minimization over control minimization, as is normally desired. Often, the combined problem may be solved faster, and with better numerical properties, than when the error and control minimization problems are solved sequentially[1].

C. Optimization using the l_1 norm

In this section, we review how the mixed l_1 optimization problem can be converted to a linear program of small size, following the presentation of Ref. 1. Further derivations later in the paper will build on this background.

A standard linear programming problem consists of finding a vector x such that

$$J = c^T x \tag{0.14}$$

is minimized, subject to

$$0 \le x \le h, \quad \text{and} \quad Ax = b \tag{0.15}$$

In (0.15), 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 is preferable for the control allocation problem.

For the conversion of the mixed optimization problem, define the function $s(x)$

$$\begin{aligned} s(x) &= x \quad \text{if } x > 0 \\ &= 0 \quad \text{otherwise} \end{aligned} \tag{0.16}$$

This function is to be interpreted element-by-element in the vector case. We assume that the preferred vector satisfies $u_{\min} \le u_p \le u_{\max}$. This condition may be eliminated without much difficulty, once the technique is understood. Define

$$u^+ = s(u - u_p), \quad u^- = -s(u_p - u) \tag{0.17}$$

so that

$$\begin{aligned} &u = u^+ - u^- + u_p \\ &0 \le u^+ \le u_{\max} - u_p, \; 0 \le u^- \le u_p - u_{\min} \end{aligned} \tag{0.18}$$

Similarly, define

$$e = CBu - a_d, \quad e^+ = s(e), \quad e^- = -s(-e) \tag{0.19}$$

so that

$$e = e^+ - e^-, \quad 0 \le e^+ \le e_{\max}, \quad 0 \le e^- \le e_{\max} \tag{0.20}$$

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

$$e^+ - e^- - CBu^+ + CBu^- = CBu_p - a_d \tag{0.21}$$

and the cost criterion

$$J = \sum_{i=1}^{q} e_i^+ + \sum_{i=1}^{q} e_i^- + \varepsilon \sum_{i=1}^{p} u_i^+ + \varepsilon \sum_{i=1}^{p} u_i^- \tag{0.22}$$

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

$$\begin{aligned} A &= \begin{pmatrix} I & -I & -CB & CB \end{pmatrix}, \quad b = CBu_p - a_d \\ c^T &= \begin{pmatrix} 1 & \dots & 1 & \varepsilon & \dots & \varepsilon \end{pmatrix} \\ h^T &= \begin{pmatrix} e_{\max} & e_{\max} & u_{\max} - u_p & u_p - u_{\min} \end{pmatrix} \end{aligned} \tag{0.23}$$

Note that the vector c implements an equal weighting of the elements within the vector a_d and within u. However, the elements of the vector c can be changed to account for various objectives. For example, Ref. 3 showed how various choices could be made to minimize drag, wing loading, radar signature, or the use of thrust vectoring. Interestingly, different weights can even be applied for positive and negative values.

Note that the A matrix of the linear programming problem has as many rows as the CB matrix. For the standard case with a 3-dimensional vector a_d, the number of rows is only 3. This size is very small in linear programming, so the problem can be solved in a few iterations using, for example, the *simplex algorithm*. 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. Speed of algorithm execution can be minimized by taking advantage of particular aspects of the control allocation problem. Because the number of columns in the A matrix is typically much greater than the number of rows, the problem is well suited for the so-called *revised simplex method*. The number of computations in this method depends only moderately on the number of columns. Because most variables of the vector x naturally have both upper and lower bounds, it is also advantageous to implement a simplex algorithm with both bounds, as opposed to the more common method with lower bounds which requires a large number of so-called *slack variables*.

Speed of execution may also be improved significantly by initializing the simplex algorithm with a so-called *basic feasible solution*. Without this special feature, an initialization phase has to be added, requiring the use of another implementation of the simplex algorithm. A basic feasible solution is a vector x that solves $Ax=b$ and is such that all elements of x are at their limits except q elements, where q is the number of rows of A. The mixed optimization algorithm can be initialized with $u=u_p$ as a feasible solution, so that

$$\begin{aligned} u^+ &= 0, \quad e^+ = s(CBu_p - a_d) \\ u^- &= 0, \quad e^- = -s(-CBu_p + a_d) \end{aligned} \tag{0.24}$$

In general, q elements of e^+ and e^- will be equal to zero, leaving only q elements (or fewer) different from zero. These elements are the basic variables of the initial feasible solution. Other useful techniques for the implementation of the revised simplex algorithm include the Sherman-Morrison-Woodbury formula to reduce the size of the matrices to be inverted and anticycling procedures to avoid infinite loops[1].

D. Implementation of control allocation solutions

The implementation of optimal control allocation methods in real-time has become feasible. It is likely that obstacles will lie solely with validation and certification issues [7]. Generally, active sets and simplex methods require a finite number of steps for convergence, but the theoretical maximum is significantly greater than the typical number required. Further, the theoretical maximum assumes perfect computations. In the near term, the most viable implementation of optimal control allocation may be in the form of table look-up[14]. Control allocation may also be implemented as an add-on module that is engaged when failures are detected.

An issue for implementation of control allocation algorithms is the rate limits of the actuators. These can be accommodated by reducing the limits u_{min}, u_{max} by the amount imposed by the rate limits. The preference vector may also be set to be the current actuator position. In this paper, we do not consider rate limits, focusing instead on the properties of control allocation solutions in general terms.

III. Control allocation with load balancing

A. Properties of l_1 optimization problems and load balancing

Linear programming theory implies certain properties of the solution of the mixed l_1 optimization problem. Specifically, if the matrix CB has 3 rows, all the elements of the optimal vector x except 3 will be either at their upper limit or at their lower limit. In terms of the control vector, this property implies that all but three control variables will be either at the upper limit, 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 this property may be debated: on the one hand, it makes sense if the algorithm does not use ineffective surfaces. On the other hand, it is desirable to see all surfaces move together to achieve the desired moment. A more balanced distribution of the required effort to the control surfaces reduces the chances of encountering the control surface rate limits.

In this section, we consider control minimization using the l_∞ norm

$$\|u\|_\infty = \max_i |u_i| \tag{0.25}$$

The l_∞ norm of a vector is the maximum of the absolute values of the elements of the vector. It is also called the *sup* norm. For control optimization, use of the l_∞ norm in

$$J = \|u - u_p\|_\infty \tag{0.26}$$

leads to an optimization criterion referred to as a *min-max* criterion, since the objective is to minimize the maximum value (in absolute terms) of the elements of the vector. This criterion has been used in a variety of networking control problems, including communication networks[19] and computer networks[16]. Typically, this criterion arises when attempting to balance the loads among multiple resources, such as processors or communication nodes. The l_∞ or min-max criterion has also been used in control design[5], although not as frequently as the l_2 criterion.

In control allocation, use of the l_∞ norm implies that one attempts to minimize the deflection of the actuators in the min-max sense. It does not matter how many actuators move: the maximum deflection should just be as small as possible. The solution that is obtained reflects this choice, by providing a more balanced distribution of the deflections than with the l_1 norm. Interestingly, the control allocation problems using the l_∞ norm can be converted to linear programs that are similar to the l_1 linear programs, and solved using the same algorithms.

B. Mixed l_1-l_∞ optimization

We first consider the optimization of the criterion

$$J = \|CBu - a_d\|_1 + \varepsilon \|u - u_p\|_\infty \tag{0.27}$$

In other words, the l_1 norm is used for the error minimization and the l_∞ norm is used for control minimization, with both criteria mixed in a single, mixed optimization criterion. A small modification of the approach used for mixed l_1 optimization yields the desired linear program.

Introduce an additional variable u^*, which is intended to become the l_∞ norm of u-u_p. Next, vectors of slack variables δu^+ and δu^- are introduced such that

$$\begin{aligned} \delta u^+ &= u^* - u^+ \\ \delta u^- &= u^* - u^- \end{aligned} \tag{0.28}$$

Using the same notation as for the l_1 optimization, a linear program can be defined with the cost criterion

$$J = \sum_{i=1}^{q} e_i^+ + \sum_{i=1}^{q} e_i^- + \varepsilon u^* \tag{0.29}$$

and the optimization vector

$$x^T = \begin{pmatrix} e^+ & e^- & u^+ & u^- & \delta u^+ & \delta u^- & u^* \end{pmatrix} \tag{0.30}$$

The linear program to be solved is

$$A = \begin{pmatrix} I_{q\times q} & -I_{q\times q} & -CB & CB & 0_{q\times p} & 0_{q\times p} & 0 \\ 0_{p\times q} & 0_{p\times q} & I_{p\times p} & 0_{p\times p} & I_{p\times p} & 0_{p\times p} & -1_{p\times 1} \\ 0_{p\times q} & 0_{p\times q} & 0_{p\times p} & I_{p\times p} & 0_{p\times p} & I_{p\times p} & -1_{p\times 1} \end{pmatrix}$$

$$b = \begin{pmatrix} CBu_p - a_d \\ 0_{p\times 1} \\ 0_{p\times 1} \end{pmatrix} \tag{0.31}$$

$$c^T = \begin{pmatrix} 1_{1\times q} & 1_{1\times q} & 0_{1\times p} & 0_{1\times p} & 0_{1\times p} & 0_{1\times p} & \varepsilon \end{pmatrix}$$

$$h^T = \begin{pmatrix} e_{\max} & e_{\max} & u_{\max} - u_p & u_p - u_{\min} & u_{\max} - u_p & u_p - u_{\min} & u^*_{\max} \end{pmatrix}$$

where $I_{a\times b}$ is the identity matrix of dimension a×b, $0_{a\times b}$ is a matrix of dimension a×b filled with zeros, $1_{a\times b}$ is a matrix of dimension a×b filled with ones, and

$$u^*_{\max} = \max\left(\left\| u_{\max} - u_p \right\|_\infty, \left\| u_p - u_{\min} \right\|_\infty \right) \tag{0.32}$$

Since the mixed l_1 -l_∞ control allocation problem can be converted to a linear program, standard algorithms can be applied to solve it efficiently. Initialization with a basic feasible solution can be performed as for the l_1 optimization, by adding to the original basic variables the new variables δu^+ and δu^-. The major drawback is that the number of rows has grown considerably in the process. From q rows (typically 3), the number has grown to $q+2p$ (where p is the number of actuators). Nevertheless, such problems can still be solved very quickly on standard computing hardware.

C. Mixed l_∞ optimization

The mixed l_∞ control allocation problem is defined by the criterion

$$J = \left\| CBu - a_d \right\|_\infty + \varepsilon \left\| u - u_p \right\|_\infty \tag{0.33}$$

This problem can be converted to a linear program using similar techniques as the mixed l_1 -l_∞ control allocation problem. Specifically, define an additional variable e^*, which is intended to become the l_∞ norm of e. Next, introduce vectors of slack variables δe^+ and δe^- such that

$$\begin{aligned} \delta e^+ &= e^* - e^+ \\ \delta e^- &= e^* - e^- \end{aligned} \tag{0.34}$$

A linear program can be defined with the cost criterion

$$J = e^* + \varepsilon u^* \tag{0.35}$$

and the optimization vector

$$x^T = \begin{pmatrix} e^+ & e^- & u^+ & u^- & \delta u^+ & \delta u^- & u^* & \delta e^+ & \delta u^- & e^* \end{pmatrix} \tag{0.36}$$

The linear program to be solved is

$$A = \begin{pmatrix} I_{q\times q} & -I_{q\times q} & -CB & CB & 0_{q\times p} & 0_{q\times p} & 0_{q\times 1} & 0_{q\times q} & 0_{q\times q} & 0_{q\times 1} \\ 0_{p\times q} & 0_{p\times q} & I_{p\times p} & 0_{p\times p} & I_{p\times p} & 0_{p\times p} & -1_{p\times 1} & 0_{p\times q} & 0_{p\times q} & 0_{p\times 1} \\ 0_{p\times q} & 0_{p\times q} & 0_{p\times p} & I_{p\times p} & 0_{p\times p} & I_{p\times p} & -1_{p\times 1} & 0_{p\times q} & 0_{p\times q} & 0_{p\times 1} \\ I_{q\times q} & 0_{q\times q} & 0_{q\times p} & 0_{q\times p} & 0_{q\times p} & 0_{q\times p} & 0_{q\times 1} & I_{q\times q} & 0_{q\times q} & -1_{q\times 1} \\ 0_{q\times q} & I_{q\times q} & 0_{q\times p} & 0_{q\times p} & 0_{q\times p} & 0_{q\times p} & 0_{q\times 1} & 0_{q\times q} & I_{q\times q} & -1_{q\times 1} \end{pmatrix}$$

$$b = \begin{pmatrix} CBu_p - a_d \\ 0_{p\times 1} \\ 0_{p\times 1} \\ 0_{q\times 1} \\ 0_{q\times 1} \end{pmatrix} \tag{0.37}$$

$$c^T = \begin{pmatrix} 0_{1\times q} & 0_{1\times q} & 0_{1\times p} & 0_{1\times p} & 0_{1\times p} & 0_{1\times p} & \varepsilon & 0_{1\times q} & 0_{1\times q} & 1 \end{pmatrix}$$

$$h^T = \begin{pmatrix} e_{\max} & e_{\max} & u_{\max} - u_p & u_p - u_{\min} & u_{\max} - u_p & u_p - u_{\min} & u^*_{\max} & e_{\max} & e_{\max} & e_{\max} \end{pmatrix}$$

An issue with this new formulation is that the computations grow even more, due to the further increase in the number of rows. It is possible that a smarter implementation will produce a more efficient algorithm. For example, Ref. 6 has an algorithm that is said to be more efficient than the simplex algorithm. However, constraints are not included.

Our evaluation of the mixed l_∞ criterion in specific examples has not shown significant differences with the mixed l_1-l_∞ optimization criterion. One reason is that any difference can only be noticed for non-feasible acceleration vectors (an achievable vector yields a zero error no matter what norm is used). Another reason is that the error vector has a small dimension compared to the control vector, and differences between the norms only become apparent when the dimensions of the vectors are large.

IV. Numerical Results

A. Low dimension example

We first consider the aircraft model used by Durham[11], which comes from NASA Dryden's "Controls Design Challenge", for a flight condition at Mach 0.5, 10,000 ft altitude. The *CB* matrix and actuator limits are given by

$$\mathrm{CB} = \begin{pmatrix} 7.35 & 7.55 & -1.35 \\ 0.856 & 5.13 & -13.7 \end{pmatrix} \tag{0.38}$$

$$\begin{aligned} u_{\max} &= [20,20,30]' \\ u_{\min} &= -u_{\max} \end{aligned} \tag{0.39}$$

The original *CB* matrix was multiplied here by 10^{-4} for numerical reasons, and the outputs associated with *CB* are moments, instead of accelerations, but these differences are insignificant for this example. The rows of the *CB* matrix are associated with roll and yaw moments, and the commands are the ailerons, the differential horizontal tail, and the rudder.

Fig. 1 shows the results of three control allocation algorithms for a roll command. The x-axis gives the roll command, as a percentage of the maximum achievable pure roll command. The top plot shows the aileron command, and the bottom plot shows the differential horizontal tail command. The plots show the results of mixed l_1 optimization using the algorithm of Ref. 1 (solid), of mixed l_2 optimization using the algorithm of Ref. 17 (dashed), and of mixed l_1-l_∞ optimization using the algorithm of Ref. 1 modified as indicated in this paper (dotted). For mixed l_2 optimization, the criterion that is optimized is

$$J = \left\| CBu - a_d \right\|_2^2 + \varepsilon^2 \left\| u \right\|_2^2 \tag{0.40}$$

subject to $u_{\min} \le u \le u_{\max}$. Note that the norms are squared, as well as the control weighting parameter ε. The parameter ε was set to 10^{-3} for mixed l_1 optimization and mixed l_1-l_∞ optimization, and 10^{-6} for mixed l_2 optimization. A smaller parameter was set in the l_2 optimization to avoid solutions with acceleration errors.

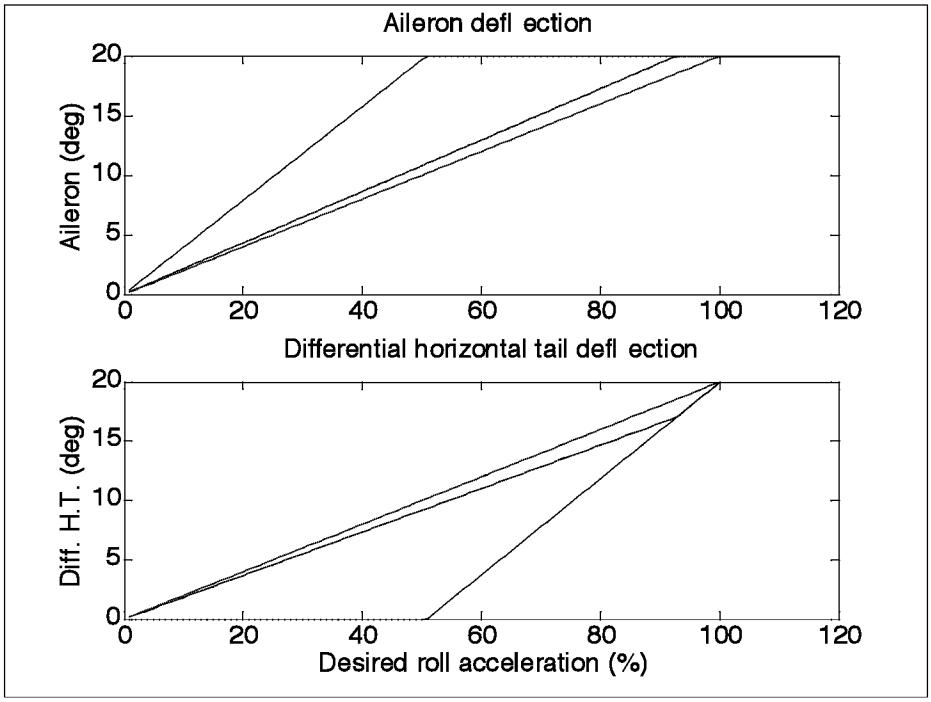

Figure 1. Aileron and differential horizontal tail commands as functions of roll acceleration (solid: l_1 optimization, dashed: l_2 optimization, dotted: l_1-l_∞ optimization).

The top plot in Fig. 1 shows that the mixed l_1 optimization uses the most effective surface (the ailerons) to achieve the desired result for commands up to 50%. Once the limit is reached, the second most effective surface (the differential tail) is used. Note that the differential horizontal tail is actually slightly more effective at producing a roll moment. However, the significant yaw moment it produces makes it less effective than the aileron in the sense specified by the optimization criterion. The mixed l_2 optimization and mixed l_1-l_∞ optimization produce similar results, using both actuators from the beginning of the plot to the end. However, the mixed l_1-l_∞ optimization produces equal deflections of the two actuators. In this case, optimal load balancing results in *load equalization* (this is not true in general for control allocation).

As pointed out in Ref. 14, the fact that mixed l_1 optimization only uses the most effective actuator may cause problems if the main actuator for a given axis fails. Fig. 2 shows the acceleration that is achieved in the case of an aileron failure (top) and differential horizontal tail failure (bottom). Since only the ailerons are used in the case of a small command, the aileron failure results in a loss of response when using the mixed l_1 optimization. In fact, a small *negative* response is observed, due to negative moment produced by the rudder (which is applied to compensate for the yaw moment that would arise if the ailerons were effective). For larger commands, the system responds when the differential horizontal tail kicks in.

Conversely, a failure of the differential tail does not degrade the performance of the system with mixed l_1 optimization for small roll commands, because the actuator is not used. Degradation is found when the commands become large. The mixed l_2 optimization and the mixed l_1-l_∞ optimization degrade more gracefully, reducing the response for either failure, but maintaining a monotonically growing response throughout the range. The mixed l_1-l_∞ optimization actually maintains a linear response, although with reduced magnitude, throughout the range of operation and for both failures..

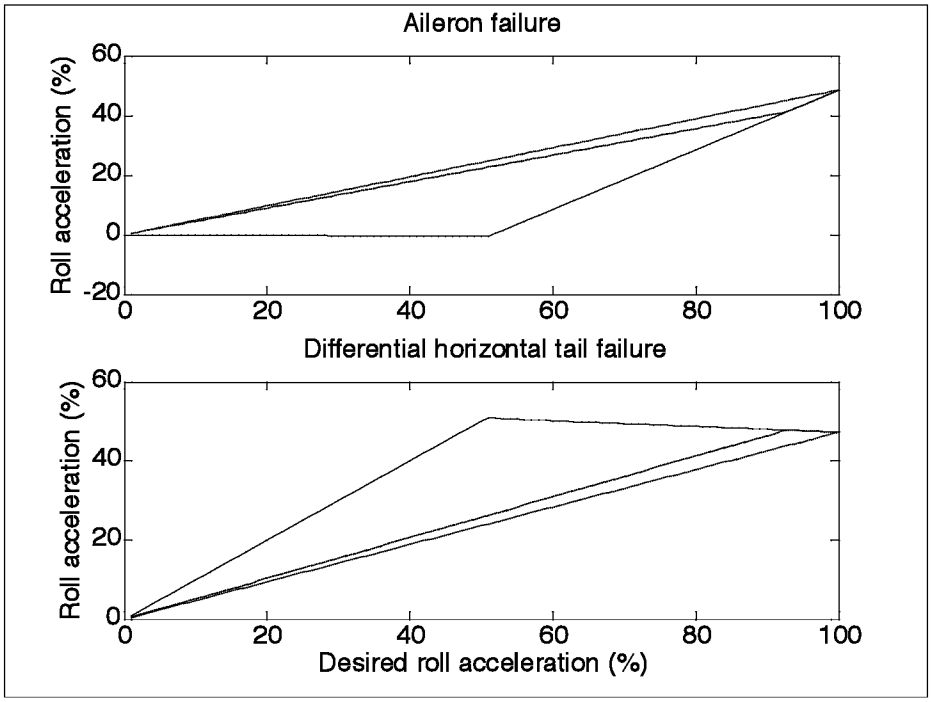

Figure 2. Roll acceleration obtained after failure of aileron (top) and differential tail (bottom) (solid: l_1 optimization, dashed: l_2 optimization, dotted: l_1-l_∞ optimization).

B. Tailless aircraft example

The second example is based on Lockheed's ICE (*innovative control effectors*) tailless aircraft model found in Ref. 2. It 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 *CB* matrix corresponding to an output vector composed of pitch rate, roll rate, and yaw rate is given in Ref. 2 as

$$CB = \begin{pmatrix} -2.5114 & -2.5115 & -1.9042 & -0.9494 & -0.9494 & \dots \\ 3.7830 & -3.7830 & 0 & 1.8255 & -1.8255 & \dots \\ 0.0453 & -0.0453 & 0 & -0.2081 & 0.2081 & \dots \\ -1.1329 & 0 & 1.5046 & 1.5046 & -0.0003 & -0.0004 \\ 0 & 0.0790 & -2.0956 & 2.0957 & -0.3067 & 0.3067 \\ 0 & -0.8038 & -0.0283 & 0.0283 & 0.0937 & -0.0937 \end{pmatrix} \tag{0.41}$$

for a flight condition at Mach 0.4 and 15,000 ft altitude. The position limits are given in Ref. 3 as

$$u_{\max} = (30 \;\; 30 \;\; 30 \;\; 60 \;\; 60 \;\; 10 \;\; 10 \;\; 10 \;\; 10 \;\; 40 \;\; 40)' \tag{0.42}$$

$$u_{\min} = (-30 \;\; -30 \;\; -30 \;\; 0 \;\; 0 \;\; -10 \;\; -10 \;\; 0 \;\; 0 \;\; 0 \;\; 0)' \tag{0.43}$$

Note that the limits of the spoiler slot deflectors were lowered from 60 degrees to 10 degrees in Ref. 3 to reduce nonlinear interactions between the spoiler slot deflectors and the elevons. The same limits were used here, although the nonlinear effects were not part of the evaluation.

Fig. 3 shows the l_1, l_2, and l_∞ norms of the control vector, as functions of the percentage of maximum yaw acceleration. As may be expected, the l_1 norm is minimized for the mixed l_1 optimization, the l_2 norm is minimized for the mixed l_2 optimization, and the l_∞ norm is minimized for the mixed l_1- l_∞ optimization. This is generally true, but not always, due to the mixed nature of the optimization criteria. In Fig. 3, one finds that the growth of the l_∞ norm of the control vector is considerably delayed with the mixed l_1- l_∞ optimization compared to the other methods, even though the l_1 and l_2 norms are quite comparable. The extra room that is provided away from the

limits may be useful, for example, to add additional excitation for the purpose of real-time parameter identification (*e.g.,* the null space injection discussed in Ref. 3).

Fig. 4 shows how the reduction in the magnitude of the control vector with the mixed l_1- l_∞ optimization may yield better performance, by avoiding end-of-range nonlinearities in the actuator effectiveness. Specifically, the figures assume that the actual acceleration vector produced is of the form

$$a = CB\left(u - \sigma u^3\right) \tag{0.44}$$

In other words, a small cubic nonlinearity reduces the actuators effectiveness uniformly as the deflections increase. The function u^3 is to be interpreted element by element. The constant σ was set so that the effectiveness of a given element was reduced by 5% when the magnitude of the element of u was 60 degrees. This is a fairly small amount of nonlinearity, especially since most surfaces are limited well before 60 degrees.

The nonlinearity only yields to a reduction of yaw acceleration of the order of 3% at the end of the range. However, note that a significant cross-coupling error occurs in the roll axis. In all plots, one finds that the mixed l_1-l_∞ optimization criterion yields the lowest error, by virtue of the smaller actuator commands that result from its solution.

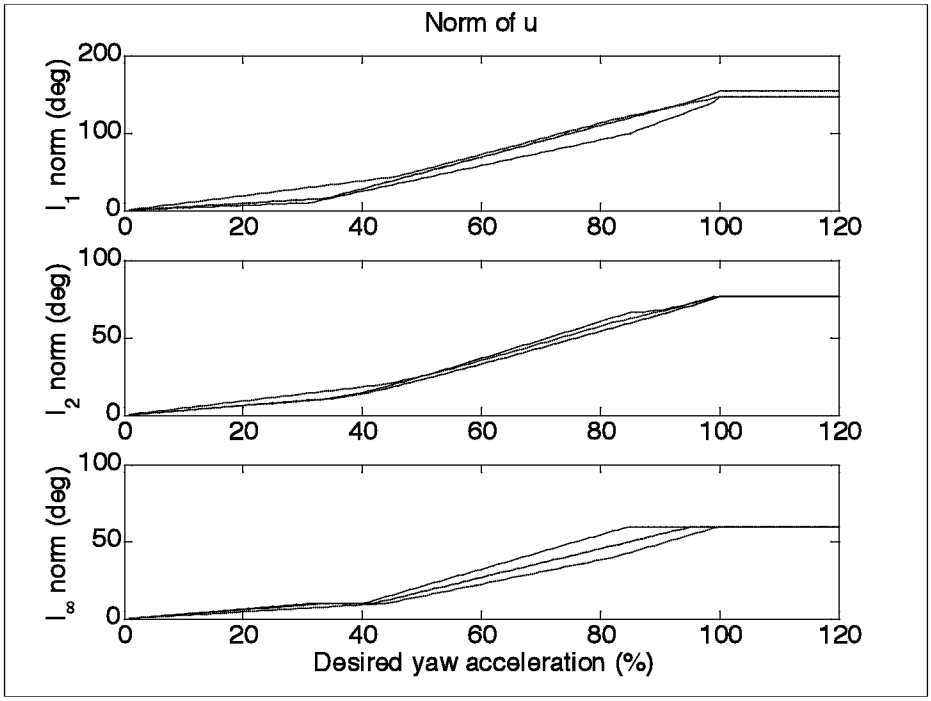

Figure 3. l_1, l_2, and l_∞ norms of the control vector, as functions of the percentage of maximum yaw acceleration (solid: l_1 optimization, dashed: l_2 optimization, dotted: l_1-l_∞ optimization).

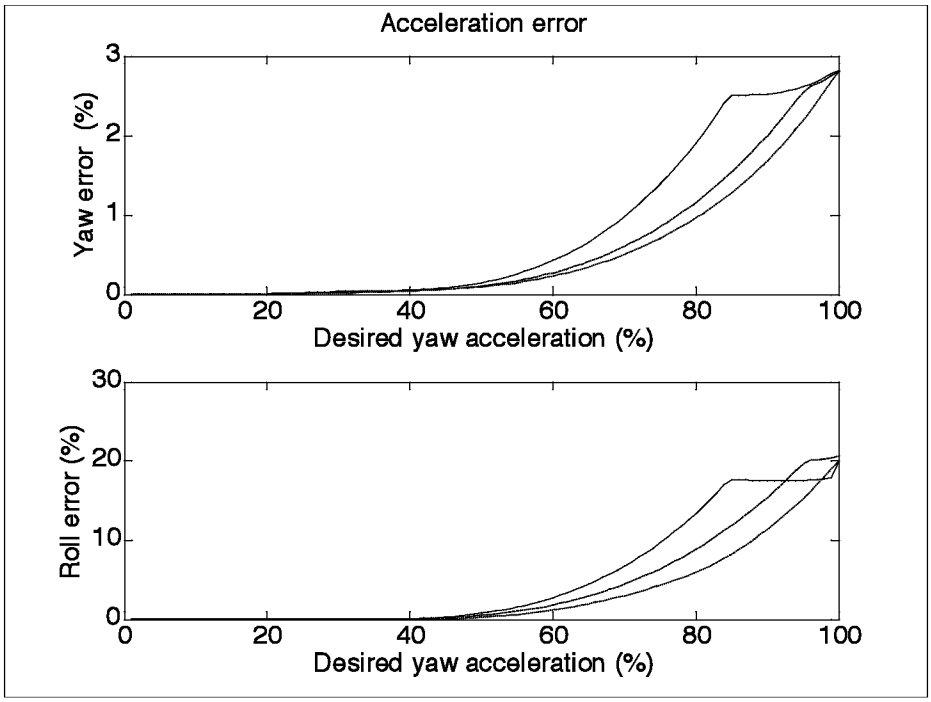

Figure 4. Yaw and roll errors due to nonlinear actuator response as functions of the percentage of maximum yaw acceleration (solid: l_1 optimization, dashed: l_2 optimization, dotted: l_1-l_∞ optimization).

V. Conclusions

The paper first reviewed how the mixed l_1 optimization criterion could be converted into a linear program and solved using a simplex algorithm. Then, it was shown that the formulation could be modified to solve mixed l_1-l_∞ (l_1 for error, l_∞ for control) and mixed l_∞ (l_∞ for error and control) optimization problems. It was argued that the l_∞ norm leads to better "load balancing", as defined in networks. A numerical example indeed showed a better balance in the use of the actuators. Advantages of such a feature were shown to include a greater resilience to actuator failures and to nonlinear effectiveness for large actuator deflections. The goal of the paper was not to prove that one method is better than another: rather, it was to increase the number of choices available to the engineer, as well as the understanding of how the choices relate to important properties of the solutions.

References

[1] M. Bodson, "Evaluation of Optimization Methods for Control Allocation," *Journal of Guidance, Control, and Dynamics*, vol. 25, no. 4, 2002, pp. 703-711.

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

[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]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, 2001, pp. 482-493.

[5]J.A. Cadzow, "Algorithm for the Minimum-Effort Problem," *IEEE Trans. on Automatic Control*, vol. 16, no. 1, 1971, pp. 60-63.

[6]J.A. Cadzow, "An Efficient Algorithmic Procedure for Obtaining a Minimum l_∞-Norm Solution to a System of Consistent Linear Equations," *SIAM J. Num. Anal.*, vol. 11, no. 6, 1974, pp. 1151-1165.

[7]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.
[8]J.B. Davidson, F.J. Lallman, & W.T. Bundick, "Real-Time Adaptive Control Allocation applied to a High Performance Aircraft," *5th SIAM Conference on Control & Its Applications*, 2001.
[9]D. B. Doman & A. D. Ngo, "Dynamic Inversion-Based Adaptive/Reconfigurable Control of the X-33 on Ascent," *Journal of Guidance, Control, and Dynamics*, vol. 25, no. 2, 2002, pp. 275-284.
[10]D.B. Doman, B.J. Gamble, & A.D. Ngo, "Quantized Control Allocation of Reaction Control Jets and Aerodynamic Control Surfaces, *Journal of Guidance, Control, and Dynamics*, vol. 32, no. 1, 2009, pp. 13-24.
[11]W. Durham, "Constrained Control Allocation," *Journal of Guidance, Control, and Dynamics*, vol. 16, no. 4, 1993, pp. 717-725.
[12]W. Durham, "Computationally Efficient Control Allocation," *Journal of Guidance, Control, and Dynamics*, vol. 24, no. 3, 2001, pp. 519-524.
[13]T.I. Fossen & T.A. Johansen, "A Survey of Control Allocation Methods for Ships and Underwater Vehicles," *14th Mediterranean Conference on Control and Automation*, 2006 pp. 1-6.
[14]K. Gundy-Burlet, K. Krishnakumar, G. Limes, & D. Bryant, "Control Reallocation Strategies for Damage Adaptation in Transport Class Aircraft," AIAA 2003-5642, *Proc. of the AIAA Guidance, Navigation, and Control Conference and Exhibit*, Austin, Texas, 2003.
[15]A. Hac, D. Doman, & M. Oppenheimer, "Unified Control of Brake-and Steer-by-wire Systems Using Optimal Control Allocation Methods," SAE Paper no. 2006-01-0924, 2006.
[16]C.-C. Han, K.G. Shin, & S.K. Yun, "On Load Balancing in Multicomputer/Distributed Systems Equipped with Circuit or Cut-Through Switching Capability," *IEEE Trans. on Computers*, vol. 49, no. 9, 2000, pp. 947-957.
[17]O. Harkegård, "Efficient Active Set Algorithms for Solving Constrained Least Squares Problems in Aircraft Control Allocation," *IEEE Conference on Decision and Control*, 2002, pp. 1295-1300.
[18]Y. Ikeda & M. Hood, "An Application of L1 Optimization to Control Allocation," *Proc. of the AIAA Guidance, Control, and Dynamics Conference*, Denver, CO, 2000.
[19]C.P. Low, "An Efficient Algorithm for the Minimum Cost Min-Max Load Terminal Assignment Problem," *IEEE Communication Letters*, vol. 9, no. 11, 2005, pp. 1012-1014.
[20]Y. Luo, A. Serrani, S. Yurkovich, D. B. Doman, & M. W. Oppenheimer, "Model-Predictive Dynamic Control Allocation Scheme for Reentry Vehicles," *Journal of Guidance, Control, and Dynamics*, vol.30, no.1, 2007, pp. 100-113.
[21]J. Petersen & M. Bodson, "Interior-Point Algorithms for Control Allocation," *Journal of Guidance, Control, and Dynamics*, vol. 28, no. 3, 2005, pp. 471-480.
[22]J. Petersen & M. Bodson, "Constrained Quadratic Programming Techniques for Control Allocation," *IEEE Trans. on Control Systems Technology*, vol. 14, no. 1, 2006, pp. 91-98.
[23]J. D. Schierman, D. G. Ward, J. R. Hull, N. Gandhi, M. W. Oppenheimer, & D. B. Doman, "Integrated Adaptive Guidance and Control for Re-Entry Vehicles with Flight-Test Results," J*ournal of Guidance, Control, and Dynamics*, vol. 27, no. 6, 2004, pp. 975-988.
[24]B. Schofield & T. Hagglund, "Optimal Control Allocation in Vehicle Dynamics Control for Rollover Mitigation," *American Control Conference*, Seattle, WA, 2008, pp. 3231 – 3236.
[25]J. Tjonnas & T.A. Johansen, "Adaptive Control Allocation," *Automatica*, vol. 44, 2008, pp. 2754-2765.
[26]E. R Van Oort, L. Sonneveldt, Q. P. Chu, & J. A. Mulder, "A Comparison of Adaptive Nonlinear Control Designs for an Over-actuated Fighter Aircraft Model," *AIAA Guidance, Navigation, and Control Conference and Exhibit*, Paper AIAA 2008-6786, Honolulu, HI, 2008.