# Resource Balancing Control Allocation

Susan A. Frost[1], *Member, AIAA & IEEE,* and Marc Bodson[2], *Fellow, IEEE, Senior Member, AIAA*

***Abstract*— 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 control effort. The paper discusses the alternative choice of using the $l_1$ norm for minimization of the tracking error and a normalized $l_\infty$ norm, or *sup* norm, for minimization of the control effort. The algorithm computes the norm of the actuator deflections scaled by the actuator limits. Minimization of the control effort then translates into the minimization of the maximum actuator deflection as a percentage of its range of motion. 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 investigated through examples. In particular, the min-max criterion results in a type of resource balancing, where the resources are the control surfaces and the algorithm balances these resources to achieve the desired command. A study of the sensitivity of the algorithms to the data is presented, which shows that the normalized $l_\infty$ algorithm has the lowest sensitivity, although high sensitivities are observed whenever the limits of performance are reached.**

## 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

$$u = Gv \tag{1}$$

where $v$ is the vector of pseudo-effectors, $u$ is the vector of actuator commands, and $G$ is a ganging matrix. Future vehicles may have a large number of actuators, making it less intuitive how the ganging matrix should be defined. Further, limited effectiveness may make it important to optimize their use within position and rate limits. Aircraft with blended wing body configurations [6] 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.

Previous work in control allocation includes the seminal paper of Durham [9], which introduced the concept of direct allocation. Buffington [3] 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 [13] similarly reported the application of $l_1$ optimization, although with fewer details. In [1], 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 [3]. Timing data showed that solutions of the problem could comfortably be performed in real-time, even for a large number 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 [5]. The fixed-point algorithm was extremely simple, but numerical tests showed that convergence could be very slow and strongly depended on the command. An elegant alternative to this algorithm was proposed by Harkegård, using the theory of active sets [12]. The algorithm was 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 [16] and for the $l_2$ norm [17]. 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 [8], [18], the problems posed by nonlinear actuator effectiveness (*e.g.,* [7]), modifications to account for the dynamic response of the actuators (*e.g.,* [15]), and the combination of control allocation with adaptation (*e.g.,* [19-20]).

## II. Optimization Formulations of Control Allocation

### A. *Control allocation problem*

Generally, the control allocation objective consists in finding the control vector $u$ such that

$$CBu = a_d \tag{2}$$

where $a_d$ represents a desired vector and $CB$ is a matrix specifying the effectiveness of the control variables. In [1], this problem formulation is obtained by taking a state-space

Manuscript received September 14, 2009.
[1]S.A. Frost is with NASA Ames Research Center, Moffett Field, CA 94035, USA (phone: 650-604-0828; fax: 650-604-3594; e-mail: susan.a.frost@nasa.gov).
[2]M. Bodson is with the University of Utah, Salt Lake City, UT 84112-9206 USA (phone: 801-581-8590; e-mail: bodson@eng.utah.edu).

representation of an aircraft where the states are the angle of attack, the pitch rate, the angle of sideslip, the roll rate, and the yaw rate, and the outputs are the pitch rate, the roll rate, and the yaw rate. The *i,j*th element of the *CB* matrix, obtained by multiplying the *B* and *C* matrices of the state-space model, specifies how much pitch (for *i=1*), roll (for *i=2*), or yaw (for *i=3*) acceleration is produced by the *j*th control variable. $a_d$ represents desired rotational accelerations.

The difficulty in control allocation is that the vector *u* is constrained. The limits generally take the form

$$u_{\min,i} \le u_i \le u_{\max,i} \qquad \text{for } i = 1,\dots,p \tag{3}$$

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* satisfying the constraints (3) and that either finds an exact solution of (2) or a solution that minimizes the error, as the *control allocation problem*.

Regardless of whether an exact solution exists, it turns out that the optimal solution is often not unique. Therefore, it is typical to look for the solution that requires the least effort, as measured by the magnitude of the control vector. This objective can be expressed as the *mixed optimization* problem:

Given a matrix *CB* and a vector $u_p$, find a vector *u* such that

$$J=\left\|CBu - a_d\right\|+\varepsilon\left\|u - u_p\right\| \tag{4}$$

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 $\varepsilon$. The vector $u_p$ is a preferred value of the control vector (typically zero). If the parameter $\varepsilon$ is small, priority is given to error minimization over control minimization, as is normally desired. Often, the mixed problem may be solved faster, and with better numerical properties, than when the error and control minimization problems are solved sequentially [1].

The norm used in the optimization objective 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, while the $l_2$ norm is the usual Euclidean norm. Algorithms have been proposed for both norms and the results of the optimization problems are sometimes quite different.

## *B. 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 [1]. Further derivations later in the paper will build on this background and use the notation.

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

$$J = c^T x \tag{5}$$

is minimized, subject to

$$0 \le x \le h, \quad \text{and} \quad Ax = b \tag{6}$$

In (6), 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 $s(x)$

$$\begin{aligned} s(x) &= x \quad \text{if } x > 0 \\ &= 0 \quad \text{otherwise} \end{aligned} \tag{7}$$

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

$$u^+ = s(u - u_p), \quad u^- = -s(u_p - u) \tag{8}$$

so that

$$\begin{aligned} &u = u^+ - u^- + u_p \\ &0 \le u^+ \le u_{\max} - u_p,\ 0 \le u^- \le u_p - u_{\min} \end{aligned} \tag{9}$$

Similarly, define

$$e = CBu - a_d, \quad e^+ = s(e), \quad e^- = -s(-e) \tag{10}$$

so that

$$e = e^+ - e^-, \quad 0 \le e^+ \le e_{\max}, \quad 0 \le e^- \le e_{\max} \tag{11}$$

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

$$e^+ - e^- - CBu^+ + CBu^- = CBu_p - a_d \tag{12}$$

and the cost criterion

$$J = \sum_{i=1}^{q} e_i^+ + \sum_{i=1}^{q} e_i^- + \varepsilon \sum_{i=1}^{p} u_i^+ + \varepsilon \sum_{i=1}^{p} u_i^- \tag{13}$$

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

$$\begin{aligned} &A = \begin{pmatrix} I & -I & -CB & CB \end{pmatrix}, \quad b = CBu_p - a_d \\ &c^T = \begin{pmatrix} 1 & \dots & 1 & \varepsilon & \dots & \varepsilon \end{pmatrix} \\ &h^T = \begin{pmatrix} e_{\max} & e_{\max} & u_{\max} - u_p & u_p - u_{\min} \end{pmatrix} \end{aligned} \tag{14}$$

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 (if anticycling procedures are used [1]), 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. In particular, an initialization phase can be avoided if the simplex algorithm is directly started with a so-called *basic feasible solution*. 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

$$\begin{aligned} &u^+ = 0, \quad e^+ = s(CBu_p - a_d) \\ &u^- = 0, \quad e^- = -s(-CBu_p + a_d) \end{aligned} \tag{15}$$

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.

## III. Control allocation with Resource balancing

### A. *Properties of $l_1$ optimization problems and resource 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 are 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 (or less) are 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 are 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 $l_\infty$ norm

$$\|u\|_\infty = \max_i |u_i| \tag{16}$$

The $l_\infty$ 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, the $l_\infty$ norm in

$$J = \|u - u_p\|_\infty \tag{17}$$

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 [14] and computer networks [11]. Typically, this criterion arises when attempting to balance the loads among multiple resources, such as processors or communication nodes.

In control allocation, use of the $l_\infty$ 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 $l_\infty$ norm can be converted to linear programs that are similar to the $l_1$ linear programs, and solved using the same algorithms [2].

### B. *Mixed $l_1$-$l_\infty$ optimization*

We consider the optimization of the criterion

$$J = \|CBu - a_d\|_1 + \varepsilon \|u - u_p\|_\infty \tag{18}$$

In other words, the $l_1$ norm is used for the error minimization and the $l_\infty$ norm is used for control minimization, with both criteria mixed in a single, mixed optimization criterion. In [2], it was shown that a small modification of the approach used for mixed $l_1$ optimization yielded the desired linear program. It was also found that the problem could be solved with the $l_\infty$ norm used for error minimization, but with no clear benefit and an additional computational cost.

Here, we show that a further modification yields the solution of a new problem where the actuator deflections are weighted in the computation of the $l_\infty$ norm as per unit values, where a unit is the maximum deflection of the actuator. The solution is not a trivial rescaling of the control variables, because the weighting may be different for positive and negative deflections.

Introduce an additional variable $u^*$, which is intended to become the $l_\infty$ norm of the normalized $u$-$u_p$ (less than 1). Next, vectors $\delta u^+$ and $\delta u^-$ are introduced such that

$$\begin{aligned} \delta u^+ &= u^* - u^+ \\ \delta u^- &= u^* - u^- \end{aligned} \tag{19}$$

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

$$J = \sum_{i=1}^{q} e_i^+ + \sum_{i=1}^{q} e_i^- + \varepsilon u^* \tag{20}$$

and the optimization vector

$$x^T = \begin{pmatrix} e^+ & e^- & u^+ & u^- & \delta u^+ & \delta u^- & u^* \end{pmatrix} \tag{21}$$

The linear program to be solved is

$$\begin{gathered} A = \begin{pmatrix} I_{qxq} & -I_{qxq} & -CB & CB & 0_{qxp} & 0_{qxp} & 0 \\ 0_{pxq} & 0_{pxq} & I_{pxp} & 0_{pxp} & N^+_{pxp} & 0_{pxp} & -1_{px1} \\ 0_{pxq} & 0_{pxq} & 0_{pxp} & I_{pxp} & 0_{pxp} & N^-_{pxp} & -1_{px1} \end{pmatrix} \\ N^+_{pxp} = diag\left\{\frac{1}{u_{\max,i} - u_{p,i}}\right\}, N^-_{pxp} = diag\left\{\frac{1}{u_{p,i} - u_{\min,i}}\right\} \\ b = \begin{pmatrix} CBu_p - a_d \\ 0_{px1} \\ 0_{px1} \end{pmatrix} \\ c^T = \begin{pmatrix} 1_{1xq} & 1_{1xq} & 0_{1xp} & 0_{1xp} & 0_{1xp} & 0_{1xp} & \varepsilon \end{pmatrix} \\ h^T = \begin{pmatrix} e_{\max} & e_{\max} & u_{\max} - u_p & u_p - u_{\min} & \cdots \\ \cdots & u_{\max} - u_p & u_p - u_{\min} & u^*_{\max} \end{pmatrix} \end{gathered} \tag{22}$$

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 $u^*_{\max} = 1$. As noted earlier, the algorithm enables a normalization of a positive deflection by the positive limit and a possibly different normalization of a negative deflection by the negative limit.

Since the 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 $\delta u^+$ and $\delta u^-$. The major drawback is that the number of rows has grown considerably. 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. Studies of the mixed $l_1$-$l_\infty$ control allocation problem without normalization demonstrated some advantages of this algorithm over the mixed $l_1$ algorithm, including robustness to actuator failures and lower sensitivity to nonlinearities in the actuator effectiveness [2]. These advantages carry over to the normalized algorithm, in addition to some further reduction in sensitivity, as will be discussed in the next section.

## IV. Sensitivity of Control Allocation Algorithms

### A. Benefits of low sensitivity and metrics

Computational studies have shown that the solution of control allocation methods can be sensitive to the value of the desired acceleration vector. A high sensitivity may yield to actuator rate saturation in the case of rapid changes of command, especially if the control allocation solutions are implemented in a look-up table, or if the rate limits are not inserted in the problem formulation. Reducing the sensitivity lowers the risk that rate limits will be encountered.

In order to quantify the sensitivity of the algorithms to the data, a sensitivity metric was computed to assess how much the solution changes in a neighborhood of a given input vector $a_d$. The sensitivity metric compares the computed effector deflections for a given desired acceleration vector and the computed deflections for the same desired acceleration vector with a small constant vector added to it. The sensitivity for an acceleration vector $a_d$ is defined to be

$$\text{sens}(a_d) = \frac{\|u_{cmd} - u_{delta}\|_2}{\|\Delta\|_2} \tag{23}$$

where $u_{cmd}$ is the vector of control surface deflections computed by the algorithm for $a_d$ , $u_{delta}$ is the vector of control surface deflections for $a_d + \Delta$, and $\Delta$ is a small acceleration vector. The total sensitivity of a set of acceleration vectors, S, is given by

$$\text{sens}_{tot}(S) = \frac{1}{\text{size}(S)} \sum_{a_d \in S} \frac{\|u_{cmd} - u_{delta}\|_2}{\|\Delta\|_2} \tag{24}$$

where size($S$) is the number of elements in the set $S$.

### B. Results

The simplex algorithms described above were used to implement the $l_1$ optimization, the $l_1$-$l_\infty$ optimization, and the $l_1$-$l_\infty$ optimization with normalization. Computations were performed on an aircraft model based on Lockheed Martin's ICE (*innovative control effectors*) tailless aircraft model found in [3]. 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 associated with an output vector composed of pitch rate, roll rate, and yaw rate is given in [3] as

$$CB = \begin{pmatrix} -2.5114 & -2.5115 & -1.9042 & -0.9494 & -0.9494 & \cdots \\ 3.7830 & -3.7830 & 0 & 1.8255 & -1.8255 & \cdots \\ 0.0453 & -0.0453 & 0 & -0.2081 & 0.2081 & \cdots \\ & -1.1329 & 0 & 1.5046 & 1.5046 & -0.0003 & -0.0004 \\ & 0 & 0.0790 & -2.0956 & 2.0957 & -0.3067 & 0.3067 \\ & 0 & -0.8038 & -0.0283 & 0.0283 & 0.0937 & -0.0937 \end{pmatrix} \tag{25}$$

for a flight condition at Mach 0.4 and 15,000 ft altitude. The position limits are given in [4] as

$$u_{\max} = \begin{pmatrix} 30 & 30 & 30 & 60 & 60 & 10 & 10 & 10 & 10 & 40 & 40 \end{pmatrix}' \tag{26}$$

$$u_{\min} = \begin{pmatrix} -30 & -30 & -30 & 0 & 0 & -10 & -10 & 0 & 0 & 0 & 0 \end{pmatrix}' \tag{27}$$

The limits of the spoiler slot deflectors were lowered from 60 degrees to 10 degrees in [4] 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. In the computations, $\varepsilon = 10^{-3}$ and $u_p \equiv [0]_{11x1}$.

Figs. 1-2 compare the results of computations for the ICE model with pure roll accelerations given as the desired accelerations (*i.e.,* the pitch and yaw acceleration components were identically zero). The simulation had 110 input points with roll accelerations from 0.0 to 1.1 times the maximum attainable roll acceleration of 351.9 deg/s$^2$. For the sensitivity analysis, $\Delta$ had a roll acceleration equal to 2.0 and pitch and yaw accelerations identically zero.

Fig. 1 shows the sensitivities of the algorithms at each point. An interesting observation in Fig. 1 is that all the algorithms show increased sensitivity when the commanded accelerations approach the achievable acceleration limits. This "terminal" sensitivity is a problematic feature for all algorithms used for control allocation [10].

Fig. 2 shows the commanded control surface deflections for the ICE model using the three algorithms. Comparison of the plots in Fig. 2 reveals that the allocation of the control surfaces was more distributed in solutions using the $l_\infty$ norm. The $l_\infty$ norm results in a type of resource balancing, where the resource is the desired command and the algorithm balances this resource among various actuators. Examination of Figs. 1-2 shows the sensitivity of the algorithms increases once one of the control surfaces becomes saturated.

An acceleration error for each algorithm was computed as the $l_2$ norm of the difference between the accelerations resulting from the commanded deflections and the desired accelerations. The acceleration error is given by

$$\text{accelError}(a_d) = \|CBu - a_d\|_2 \qquad (28)$$

where $a_d$ is the vector of desired accelerations and $u$ is the vector of commanded deflections determined for $a_d$ by the algorithm. For all three algorithms, the acceleration errors were essentially zero and were within the numerical noise floor when the desired accelerations were achievable. This observation means that the solutions are the same as would be obtained with a two-step optimization procedure, where error is minimized first and control is minimized within this solution as a secondary objective.

The sensitivity of the algorithms was analyzed for points randomly sampled from one of two sets: a cube containing the attainable moment set (AMS) or the boundary of the AMS (we use the standard terminology of [9], although our set is an achievable acceleration set). Computations using these two sets allow examination of the overall algorithm sensitivity and the terminal sensitivity at the AMS boundary.

Fig. 3(a) shows the mean total sensitivity for the first case. Ten sets of 500 desired accelerations were randomly sampled from a cube with its boundary outside the attainable moment set (see table I for the cube limits). The sensitivity calculation used random $\Delta$ vectors of length 3.0. Each marker in Fig. 3 represents the mean total sensitivity of an algorithm for a specified value of $\varepsilon$. The mixed $l_1$-$l_\infty$ optimization with normalization was less sensitive than the mixed $l_1$-$l_\infty$ optimization algorithm, which was less sensitive than the mixed $l_1$ algorithm. For both the $l_1$ algorithm and the $l_1$-$l_\infty$ normalized algorithm, a small trade-off was observed between the sensitivity of the algorithm and the value of $\varepsilon$. For the mixed $l_1$-$l_\infty$ optimization algorithms, the sensitivities are relatively similar for the values of $\varepsilon$ used in the computations.

A total acceleration error was computed to assess the acceleration errors for the algorithms over a set of points. The total acceleration error for a set, $S$, is given by

$$\text{accelError}_{tot}(S) = \frac{1}{\text{size}(S)} \sum_{a_d \in S} \|CBu - a_d\|_2 \qquad (29)$$

where $a_d$ is the vector of desired accelerations, $u$ is the vector of deflections determined for $a_d$ and size($S$) is the number of elements in the set $S$. All three algorithms had total acceleration errors of 25.09 deg/s$^2$ for points sampled from the cube around the AMS. The total errors were large because each set had unattainable points, resulting in large errors that dominated the smaller errors.

Fig. 3(b) shows the total sensitivity for points randomly sampled from the AMS boundary. These results tell a similar story to the results of Fig. 3(a), except the sensitivity magnitudes are larger for points sampled from the AMS boundary. Mean total acceleration errors are given in table II for the mixed $l_1$ algorithm. The acceleration errors increased with $\varepsilon$, due to the increased priority given to control minimization. These results suggest a trade-off between performance (error) and cost (control) for the $l_1$ algorithm on the AMS boundary. Zero acceleration errors were observed for both mixed $l_1$-$l_\infty$ optimization algorithms, suggesting no trade-off between error and control for these algorithms.

## V. Conclusions

We presented a new algorithm for control allocation that makes use of mixed $l_1$-$l_\infty$ optimization with normalization. This algorithm allows the control surface deflections to be more evenly allocated among the available control surface resources. The sensitivity of the mixed $l_1$ optimization, the mixed $l_1$-$l_\infty$ optimization, and the mixed $l_1$-$l_\infty$ optimization with normalization was examined. The mixed $l_1$-$l_\infty$ optimization with normalization had the lowest sensitivity overall, although high sensitivities were observed for all algorithms near the boundary of the AMS.

## 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] M. Bodson & S. A. Frost, "Control Allocation with Load Balancing", *AIAA Guidance, Navigation, & Control Conference,* Chicago, IL, August 2009.

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

[4] 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.

[5] 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.

[6] 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.

[7] 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.

[8] 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.

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

[10] S. A. Frost & M. Bodson, "Sensitivity Analysis of Linear and Quadratic Programming Algorithms for Control Allocation," *AIAA Infotech@Aerospace Conference,* Seattle, WA, April 2009.

[11] 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.

[12] 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.

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

[14] 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.

[15] 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.

[16] J. Petersen & M. Bodson, "Interior-Point Algorithms for Control Allocation," *Journal of Guidance, Control, and Dynamics*, vol. 28, no. 3, 2005, pp. 471-480.

[17] 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.
[18] 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,” *Journal of Guidance, Control, and Dynamics*, vol. 27, no. 6, 2004, pp. 975-988.
[19] J. Tjonnas & T.A. Johansen, “Adaptive Control Allocation,” *Automatica*, vol. 44, 2008, pp. 2754-2765.
[20] 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.

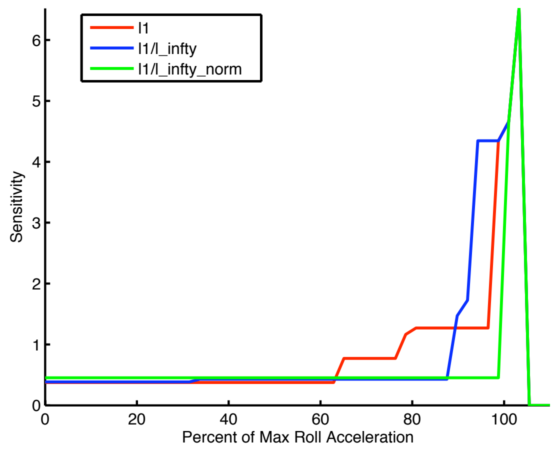

**Fig. 1.** Sensitivities of $l_1$, $l_1$-$l_\infty$, and $l_1$-$l_\infty$ normalized optimization algorithms for pure roll acceleration.

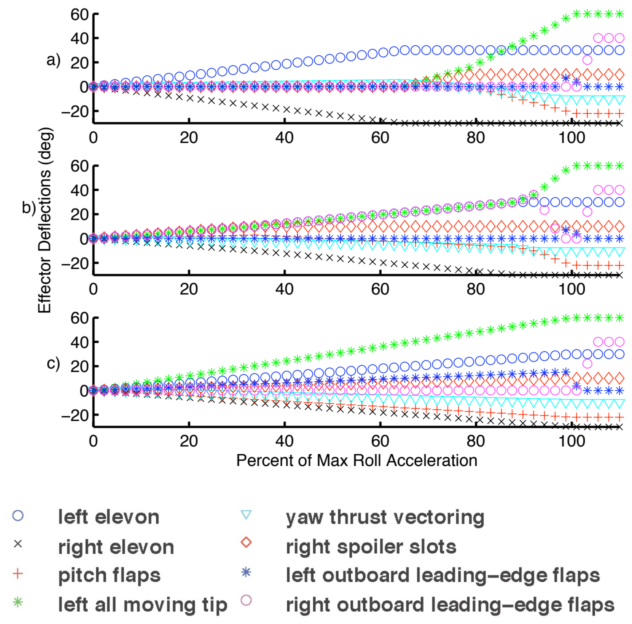

**Fig. 2.** Control surface deflections for pure roll acceleration for (a) $l_1$, (b) $l_1$-$l_\infty$, and (c) $l_1$-$l_\infty$ normalized optimization algorithms.

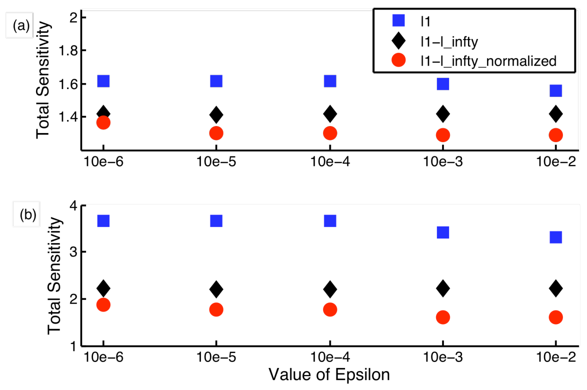

**Fig. 3.** Mean total sensitivity for $l_1$ optimization, $l_1$-$l_\infty$ optimization, and $l_1$-$l_\infty$ normalized optimization for (a) accelerations from cube around AMS and (b) accelerations from AMS boundary.

**Table I.** Acceleration limits (deg/s$^2$) for cube around AMS.

| Limits | Pitch | Roll | Yaw |
|---|---|---|---|
| Negative (deg/s$^2$) | -333.098 | -351.941 | -25.772 |
| Positive (deg/s$^2$) | 249.234 | 351.941 | 25.772 |

**Table II.** Total acceleration errors (deg/s$^2$) $l_1$ algorithm with input from AMS boundary.

| Value of epsilon | Accel. Error (deg/s$^2$) |
|---|---|
| 1e-6 | 6.8e-9 |
| 1e-5 | 1.2e-8 |
| 1e-4 | 1.2e-8 |
| 1e-3 | 4.2e-4 |
| 1e-2 | 7.2e-3 |