Efficient Active Set Algorithms for Solving Constrained Least Squares Problems in Aircraft Control Allocation

Ola Härkegård

Division of Automatic Control
Department of Electrical Engineering
Linköpings universitet, SE-581 83 Linköping, Sweden
WWW: <http://www.control.isy.liu.se>
E-mail: ola@isy.liu.se

May 6, 2002

Report no.: LiTH-ISY-R-2426
Submitted to CDC 2002, Las Vegas, USA

Technical reports from the Control & Communication group in Linköping
are available at <http://www.control.isy.liu.se/publications>.

Efficient Active Set Algorithms for Solving Constrained Least Squares Problems in Aircraft Control Allocation

Ola Härtigård

May 6, 2002

Abstract

In aircraft control, control allocation can be used to distribute the total control effort among the actuators when the number of actuators exceeds the number of controlled variables. The control allocation problem is often posed as a constrained least squares problem to incorporate the actuator position and rate limits. Most proposed methods for real-time implementation, like the redistributed pseudoinverse method, only deliver approximate, and sometimes unreliable solutions. In this paper we investigate the use of active set methods for control allocation. We develop active set algorithms that always find the optimal control distribution, and show by simulation that the timing requirements are in the same range as for the redistributed pseudoinverse method.

1 Introduction

Aircraft control is one important application where the number of actuators exceeds the number of variables to be controlled. Such actuator redundancy is motivated by enhanced maneuverability and tolerance towards actuator failures. How to distribute the control effort among the actuators is known as the control allocation problem.

There are several reasons for designing the control allocation algorithm (how to produce the total control effort) and the feedback law (which control effort to produce) separately. First, not all control design methods have support for handling actuator redundancy. Second, and more importantly, performing control allocation separately makes it straightforward to handle actuator position and rate constraints.

Figure 1: Overview of the modular controller structure.

Two important issues when designing a control allocation scheme are: How should the control allocation problem be posed mathematically? How can the problem be solved numerically in real-time?

Several different formulations, like direct control allocation [7], daisy chaining [5], and constrained linear programming [4, 12] have been proposed, see [2] for a good survey. However, the most frequently encountered approach is based on constrained quadratic programming [10, 6, 16, 3], and this is the formulation that we will use in this paper. Despite its widespread use, most numerical solutions proposed for this formulation of the control allocation problem only deliver approximate solutions of varying accuracy. The underlying, often unspoken, claim is that the general constrained quadratic programming machinery is too complex to be used in real-time aircraft applications.

In this paper we investigate this claim and propose two algorithms, based on active set methods, for finding the optimal solution of the control allocation problem posed as a constrained sequential least squares problem. Simulations show that the proposed algorithms have timing properties similar to the redistributed pseudoinverse [16] and produce solutions with better accuracy.

2 Control Allocation in a Least Squares Framework

Assume that a controller structure like the one in Figure 1 is used. The feedback part of the control system delivers a *virtual* or *generalized* control v of dimension n . Typically, v represents the moments or angular accelerations to be produced in pitch, roll, and yaw ($n = 3$). The true control u , representing the commanded actuator positions, is of dimension $m > n$, where m is the number of independently controlled actuators. For JAS 39 Gripen the main actuators are the canard wings (2), the elevons (4), and

the rudder, which gives $m = 7$. In general, the typical range is $5 \leq m \leq 20$ [2].

Ignoring the actuator dynamics, which is much faster than the remaining aircraft dynamics, the generated virtual control v_g is given by

$$v_g = Bu \quad (1)$$

where B is the control effectiveness matrix, that may well be state dependent. The control vector u is restricted by

$$\underline{u} \leq u \leq \bar{u} \quad (2)$$

where $\underline{u}$ and $\bar{u}$ are lower and upper bounds determined at each sampling instant by the position and rate limits of the actuators.

From a pragmatic point of view the control allocation problem can be stated as follows. Given v , determine u , satisfying (2), such that $v_g = v$. If there are several solutions, pick the best one. If there is no solution, determine u such that v_g approximates v as well as possible.

2.1 Problem Formulations

As stated in the introduction, we will use the 2-norm, $\|u\|^2 = u^T u$, as a measure of how “good” a solution or an approximation is. This leads to the following **sequential least squares** formulation of the control allocation problem:

$$u_S = \arg \min_u \left\{ \|W(u - u_p)\| : u \in \arg \min_{\underline{u} \leq u \leq \bar{u}} \|W_a(Bu - v)\| \right\} \quad (3)$$

Here, u_p is the preferred control vector and W and W_a are positive definite weighting matrices. Equation (3) should be interpreted as follows: Given the set of feasible controls that minimize $Bu - v$ (weighted by W_a), pick the control that minimizes $u - u_p$ (weighted by W).

An approximate way to reformulate a sequential least squares problem like (3) is to merge the two optimization criteria into one by summation and form a **weighted least squares** problem:

$$u_W = \arg \min_{\underline{u} \leq u \leq \bar{u}} \|W(u - u_p)\|^2 + \gamma \|W_a(Bu - v)\|^2 \quad (4)$$

To emphasize that primarily, $Bu - v$ should be minimized, a large value of the weighting factor γ is used. In [1] it is shown that

$$\lim_{\gamma \rightarrow \infty} u_W(\gamma) = u_S$$

2.2 Numerical Methods

Let us now consider some of the numerical methods proposed in the literature for solving (3) and (4).

2.2.1 Sequential Least Squares

Most existing methods for solving the sequential least squares problem (3) are based on the well known fact that if we disregard the constraints (2), then (3) has a closed form pseudoinverse solution, see, e.g., [1]. In [16, 9] a redistribution scheme is used, in which all actuators that violate their bounds in the pseudoinverse solution are saturated and removed from the optimization. Then, the problem is resolved with the remaining actuators as free variables. The procedure is repeated until a feasible solution has been reached or all variables have saturated. This *redistributed pseudoinverse* (RPI) scheme does not guarantee that the optimal solution is found or even that $Bu = v$ is met when possible, as illustrated in [2]. A similar method, where only one actuator is saturated in each iteration, is proposed in [10].

Another approximate approach, where the $2m$ box constraints (2) are replaced by one ellipsoidal constraint, is presented in [10]. The new problem is shown to be solved efficiently using a bisection method.

Efforts to compute the exact optimal solution can also be found. In [15], an iterative method, which converges to the optimal solution when the feasible region is given by a general convex set, is presented. In [3], exhaustive search is proposed as a way to determine which actuators should saturate. As noted, the computational requirements of this method grows rapidly with the number of actuators.

In [13], a special case of the sequential least squares problem is considered, namely the minimal least squares problem

$$u_M = \arg \min_u \left\{ \|u\| : u \in \arg \min_{\underline{u} \leq u \leq \bar{u}} \|W_a(Bu - v)\| \right\} \quad (5)$$

where the minimal length feasible solution is picked if $\|W_a(Bu - v)\|$ does not have a unique minimizer. The author suggests the use of active set methods for solving the problem. The sequential least squares problem (3) can not always be cast into a minimal least squares problem since in the new variables

$$\tilde{u} = W(u - u_p)$$

the constraints (2) will in general no longer be simple box constraints. However, if W is a diagonal matrix, corresponding to pure scaling of the variables,

the box constraint property is preserved. In these cases, it is favorable to use the minimum least squares approach since it saves us one QR decomposition.

2.2.2 Weighted Least Squares

For the weighted least squares formulation (4), an iterative fixed-point algorithm is proposed in [6]. The algorithm asymptotically converges to the optimal solution.

3 Active Set Algorithms for Control Allocation

Except for the exhaustive search algorithm, all algorithms in the previous section in general only produce approximate solutions, some of which converge to the optimum as the number iterations goes to infinity. In this section we will investigate the use of active set methods for control allocation. Active set methods are used in many of today's solvers for constrained quadratic programming, and can be shown to find the optimal solution in a finite number of iterations [14].

A general active set algorithm is outlined in the next section, in which we also motivate why active set methods seem well suited for control allocation. Two active set algorithms, tailored for solving the sequential least squares problem (3) and the weighted least squares problem (4), are presented in Section 3.2 and Section 3.3, respectively.

3.1 The Active Set Algorithm

Consider the bounded and equality constrained least squares problem

$$\min_u \|Au - b\| \tag{6a}$$

$$Bu = v \tag{6b}$$

$$Cu \geq U \tag{6c}$$

where (6c) is equivalent to (2) if we define $C = \begin{pmatrix} I \\ -I \end{pmatrix}$ and $U = \begin{pmatrix} u \\ -u \end{pmatrix}$. Active set algorithms solve this problem by solving a sequence of equality constrained problems. In each step some of the inequality constraints are regarded as equality constraints, and form the working set $\mathcal{W}$, while the remaining inequality constraints are disregarded. The working set at the optimum is known as the active set of the solution.

Note that this is much like the RPI method from Section 2.2.1. The difference is that an active set algorithm is more careful regarding which

variables to saturate, and that an active set algorithm has the ability to free a variable that was saturated in a previous iteration.

An active set algorithm for solving (6), adopted from [1], is given below in pseudocode.

Algorithm 3.1 (Active set algorithm)

Let u^0 be a feasible starting point. A point is feasible if it satisfies (6b) and (6c). Let the working set $\mathcal{W}$ contain (a subset of) the active inequality constraints at u^0 .

for $k = 0, 1, 2, \dots, N - 1$

 Given u^k , find the optimal perturbation p , considering the constraints in the working set as equality constraints and disregarding the remaining inequality constraints. Solve

$$\begin{aligned} \min_p \quad & \|A(u^k + p) - b\| \\ & Bp = 0 \\ & p_i = 0, \quad i \in \mathcal{W} \end{aligned}$$

if $u^k + p$ is feasible

 Set $u^{k+1} = u^k + p$ and compute the Lagrange multipliers, $\begin{pmatrix} \mu \\ \lambda \end{pmatrix}$, where μ is associated with (6b) and λ with the active constraints in (6c).

if all $\lambda \geq 0$

u^{k+1} is the optimal solution to (6).

else

 Remove the constraint associated with the most negative λ from the working set.

else

 Determine the maximal step length α such that $u^{k+1} = u^k + \alpha p$ is feasible. Add the primary bounding constraint to the working set.

end

The Lagrange multipliers are determined by

$$A^T(Au - b) = \begin{pmatrix} B^T & C_0^T \end{pmatrix} \begin{pmatrix} \mu \\ \lambda \end{pmatrix} \quad (7)$$

where C_0 contains the rows of C that correspond to constraints in the working set.

So why should an active set algorithm be suited for control allocation?

Active set algorithms are the most efficient when a good estimate of the optimal active set is available. Then the number of changes in the working set, i.e., the number of iterations, will be small. In control allocation, a very good estimate is given by the active set of the solution in the previous sampling instant, since in practice, the optimization problem will not change much between two sampling instants. This means that the optimal solution, and particularly the set of active constraints, will not change much either.

Another appealing property is that in each iteration, a feasible solution u^{k+1} is produced that yields a lower value of the cost function than the previous solution, u^k . Thus, the maximal number of iterations, N , can be set to reflect the maximal computation time available.

Furthermore, the calculations to be performed at each iteration are computationally cheap, consisting mainly of QR decompositions as we will see in Section 4. Also, the active set algorithm is intuitively simple and straightforward to implement.

3.2 Sequential Least Squares

Consider the sequential least squares problem (3). We now present an algorithm for solving this problem in two phases. It is assumed that no three actuators produce coplanar controls. This means that any three columns of B are linearly independent.

Algorithm 3.2 (Active set algorithm for sequential LS)

Phase 1:

1. Let u^0 and $\mathcal{W}$ be the resulting solution and working set from the previous sampling instant.
2. Solve

$$u_1 = \arg \min_u \|W_a(Bu - v)\|$$

$$\underline{u} \leq u \leq \bar{u}$$

using Algorithm 3.1 with the following modification. When the number of free variables exceeds n , the optimal perturbation p is not uniquely determined. In this case, pick the minimum perturbation.

3. If $Bu_1 = v$, move to phase 2, otherwise set $u_S = u_1$ and halt.

Phase 2:

1. Let u^0 and $\mathcal{W}$ be the resulting solution and working set from phase 1.
2. Solve

$$u_S = \arg \min_u \|W(u - u_p)\|$$

$$Bu = v$$

$$\underline{u} \leq u \leq \bar{u}$$

using Algorithm 3.1.

In phase 1, step 2, when the choice of p is not unique, another possibility would be to pick the perturbation that minimizes the secondary cost function, $\|W(u^k + p - u_p)\|$. However, this requires an extra QR decomposition to be computed and simulations show that this choice yields worse overall timing results.

3.3 Weighted Least Squares

The weighted least squares problem (4) has a more straightforward solution.

Algorithm 3.3 (Active set algorithm for weighted LS)

1. Let u^0 and $\mathcal{W}$ be the resulting solution and working set from the previous sampling instant.
2. Rewrite the cost function as

$$\|W(u - u_p)\|^2 + \gamma \|W_a(Bu - v)\|^2 = \left\| \underbrace{\begin{pmatrix} \gamma W_a B \\ W \end{pmatrix}}_A u - \underbrace{\begin{pmatrix} \gamma W_a v \\ W u_p \end{pmatrix}}_b \right\|^2$$

and solve

$$u_W = \arg \min_u \|Au - b\|$$

$$\underline{u} \leq u \leq \bar{u}$$

using Algorithm 3.1.

The constraint rows are ordered first in A and b to avoid numerical problems, see [1, pp. 192-193].

When we have rate constraints affecting $\underline{u}$ and $\bar{u}$, u^0 and $\mathcal{W}$ may not be “compatible” since the fixed components in u^0 are set to the limits specified by $\underline{u}$ and $\bar{u}$ from the previous sampling instant. In this case we update the values of the fixed variables in u^0 to reflect the new limits.

4 Computing the Solution

In each step of the active set algorithm we need to compute the optimal perturbation p . Also, if $u^k + p$ is feasible, we need to compute the Lagrange multipliers, λ , to determine whether or not we have reached the optimum.

To compute p we will need to solve least squares problems of the form

$$\min_p \|Ap - b\|$$

A numerically stable and efficient way of solving least squares problems is to use QR decomposition [11]. If A has full column rank then it can be decomposed into

$$A = QR = \begin{pmatrix} Q_1 & Q_2 \end{pmatrix} \begin{pmatrix} R_1 \\ 0 \end{pmatrix}$$

where Q is unitary ($Q^T Q = I$) and R_1 is upper triangular and non-singular. Since premultiplying with Q^T does not affect the norm we get

$$\|Ap - b\|^2 = \|Rp - Q^T b\|^2 = \|R_1 p - Q_1^T b\|^2 + \|Q_2^T b\|^2$$

The minimum norm is obtained for

$$p = R_1^{-1} Q_1^T b$$

Note that since R_1 is upper triangular we do not need to form R_1^{-1} explicitly but p can be solved from $R_1 p = Q_1^T b$ by backsubstitution. For convenience, introduce the short hand notation

$$p = A \backslash b$$

for the least squares solution, using MATLAB style.

Let us now investigate the numerical computations involved in the two algorithms in some detail.

4.1 Sequential Least Squares

Phase 1: Optimization problem:

$$\begin{aligned} \min_u \|Au - b\| \\ Cu \geq U \end{aligned}$$

where $A = W_a B$ and $b = W_a v$.

Solution update

$$\begin{aligned} \min_p \|A(u^k + p) - b\| \\ p_i = 0, \quad i \in \mathcal{W} \end{aligned}$$

For convenience, assume that p is partitioned as $p = \begin{pmatrix} p_f \\ 0 \end{pmatrix}$, where p_f are the free variables ($\dim p_f = m_f$), and let $A = \begin{pmatrix} A_f & A_0 \end{pmatrix}$. This yields

$$\|A(u^k + p) - b\| = \|A_f p_f - d\|$$

where $d = b - Au^k$. For $m_f \leq n$, the unique minimizer given by

$$p_f = A_f \backslash d \tag{8}$$

For $m_f > n$, a parameterization of the minimizing solutions can be obtained from the QR decomposition of A_f^T .

$$A_f^T = QR = \begin{pmatrix} Q_1 & Q_2 \end{pmatrix} \begin{pmatrix} R_1 \\ 0 \end{pmatrix} = Q_1 R_1$$

along with the change of variables

$$p_f = Qq = \begin{pmatrix} Q_1 & Q_2 \end{pmatrix} \begin{pmatrix} q_1 \\ q_2 \end{pmatrix} = Q_1 q_1 + Q_2 q_2$$

gives us

$$\|A_f p_f - d\| = \|R_1^T Q_1^T (Q_1 q_1 + Q_2 q_2) - d\| = \|R_1^T q_1 - d\| = 0$$

for

$$q_1 = R_1^{-T} d$$

This means that the set of minimizing solutions can be written as

$$p_f = Q_1 R_1^{-T} d + Q_2 q_2$$

where q_2 is a free variable. Since the columns of Q_1 and Q_2 are orthonormal, $\|p_f\|$, and thereby also $\|p\|$, is minimized for

$$q_2 = 0$$

As noted in Section 3.2, another alternative is to use this freedom to minimize $\|W(u - u_p)\|$:

$$\begin{aligned} \|W(u^k + p - u_p)\| &= \left\| \begin{pmatrix} W_f & W_0 \end{pmatrix} \begin{pmatrix} p_f \\ 0 \end{pmatrix} - W(u_p - u^k) \right\| \\ &= \|W_f(Q_1 R_1^{-T} d + Q_2 q_2) - W(u_p - u^k)\| = \min \end{aligned}$$

yields

$$q_2 = (W_f Q_2) \setminus (W(u_p - u^k) - W_f Q_1 R_1^{-T} d)$$

Lagrange multipliers Comparing with (7) we get

$$A^T(Au - b) = C_0^T \lambda$$

Since $C_0 C_0^T = I$, multiplying by C_0 from the left yields

$$\lambda = C_0 A^T(Au - b) \tag{9}$$

Phase 2: Optimization problem:

$$\begin{aligned} \min_u & \|Au - b\| \\ & Bu = v \\ & Cu \geq U \end{aligned}$$

where $A = W$ and $b = Wu_p$.

Solution update

$$\begin{aligned} \min_p & \|A(u^k + p) - b\| & \min_p & \|Ap - d\| \\ & Bp = 0 & \iff & \\ & p_i = 0, \ i \in \mathcal{W} & & Ep = \begin{pmatrix} B \\ C_0 \end{pmatrix} p = 0 \end{aligned}$$

where $d = b - Au^k$. Using the QR decomposition

$$E^T = QR = \begin{pmatrix} Q_1 & Q_2 \end{pmatrix} \begin{pmatrix} R_1 \\ 0 \end{pmatrix} = Q_1 R_1 \tag{10}$$

and introducing

$$p = Qq = \begin{pmatrix} Q_1 & Q_2 \end{pmatrix} \begin{pmatrix} q_1 \\ q_2 \end{pmatrix} = Q_1 q_1 + Q_2 q_2$$

gives us

$$Ep = R_1^T Q_1^T (Q_1 q_1 + Q_2 q_2) = R_1^T q_1 = 0 \quad \Longleftrightarrow \quad q_1 = 0$$

since R_1 is non-singular. Inserting $p = Q_2 q_2$ into the optimization criterion yields

$$\|Ap - d\| = \|AQ_2 q_2 - d\| = \min$$

for

$$q_2 = (AQ_2)^{\setminus} d$$

Essentially, this is where our algorithm differs from the minimal least squares algorithm proposed in [13]. With $A = I$, the optimal solution is simply given by $q_2 = Q_2^T d$.

Lagrange multipliers Comparing with (7) we get

$$A^T(Au - b) = E^T \Lambda = Q_1 R_1 \Lambda$$

Premultiplying both sides with $R_1^{-1} Q_1^T$ yields the solution

$$\Lambda = \begin{pmatrix} \mu \\ \lambda \end{pmatrix} = R_1^{-1} Q_1^T A^T (Au - b)$$

from which the relevant Lagrange variables λ can be extracted.

4.2 Weighted Least Squares

In Algorithm 3.3 the optimization problem is given by

$$\begin{aligned} \min_u & \|Au - b\| \\ & Cu \geq U \end{aligned}$$

This problem coincides with the phase 1 problem of sequential least squares. Here, since A has full column rank, the unique optimal perturbation in each iteration is determined by (8) and the Lagrange multipliers by (9).

Acronym	Description
SLS	Sequential least squares (Algorithm 3.2)
MLS	Minimal least squares [13]
WLS	Weighted least squares (Algorithm 3.3 with $\gamma = 1000$)
WLS2	WLS with a maximum of $N = 2$ iterations
RPI	Redistributed pseudoinverse [16]
FXP	Fixed-point iteration algorithm [6] ($\gamma = 1000$, 50 iterations)
LSQLIN	<code>lsqlin</code> from MATLAB's Optimization Toolbox

Table 1: Numerical solvers used for evaluation.

5 Simulations

Let us now evaluate the two proposed algorithms and compare them with some of the existing alternative solvers discussed in Section 2.2. We will discuss the quality of the solutions produced, their timing properties, and finally comment on some implementation issues.

5.1 Algorithms

The algorithms to be evaluated are the sequential and weighted least squares algorithms from Section 3, the minimal least squares algorithm from [13], the redistributed pseudoinverse algorithm from [16] (see Section 2.2.1), the fixed-point iteration algorithm from [6], and the standard least squares solver `lsqlin` from MATLAB's Optimization Toolbox. `lsqlin` is used as a reference to confirm that the sequential and minimal least squares algorithms indeed find the optimal solutions, but is only applicable when the demanded virtual control is feasible. Table 1 lists the algorithms along with their acronyms.

In the weighted least squares problem formulation (4), used in the WLS and FXP algorithms, $\gamma = 1000$ is used (corresponding to $\varepsilon = 0.001$ in [6]). In FXP, the number of iterations is set to 50 and the solution from the previous sample is used as an initial guess to hot start the algorithm.

5.2 Aircraft Simulation Data

Aircraft data, consisting of the control effectiveness matrix B (3×8) and the position and rate limits, are taken from [8, sec. Comparisons/Control Configuration] to which we refer for details. The virtual control trajectory, $v(t)$, is shown in Figure 2, and corresponds to the helical path with radius

Figure 2: Virtual control trajectory to be produced.

0.06 used in [8]. Its components are the commanded aerodynamic moment coefficients C_l , C_m , and C_n . By varying t_f , the final time of the trajectory, different helical rates [8] are achieved. When the rate is too high, some parts of the trajectory become infeasible due to the actuator rate constraints.

5.3 Simulation Results

Three different cases were simulated:

Case 1: Feasible trajectory ($t_f = 25$ s, helical rate = 0.045) with W and W_a set to identity matrices and u_p set to the zero vector. See Table 2 and Figure 4 for simulation results.

Case 2: Same trajectory as in case 1 but with

$$W = \text{diag} \begin{pmatrix} 10 & 10 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \quad (11)$$

This corresponds to penalizing the use of the horizontal tails for control. See Table 3 and Figure 5.

Case 3: Partially infeasible trajectory ($t_f = 4$ s, helical rate = 0.28) with the same parameter setting as in case 1. See Table 4 and Figure 6.

The simulations were performed in MATLAB 6.1 running on a 800 MHz Pentium III computer. The `tic` and `toc` commands were used to compute the timing properties which were averaged over 1000 runs. The mean and max errors given correspond to the mean and max values of $\|Bu(t) - v(t)\|$ over time. Furthermore, the average number of saturated controls in the solutions are given and also the average number of iterations performed by each algorithm.

5.4 Comments

Solution Quality

All algorithms produce solutions which satisfy both actuator position and rate constraints, see Figure 4-6.

By construction, SLS and MLS both generate the exact solution to the control allocation problem formulated as the sequential least squares problem (3). WLS only solves an approximation of the original problem, see (4), but with the weight γ set to 1000 as in the simulations, WLS comes very close to recovering the true optimal solution.

WLS2 uses at most two iterations, corresponding to at most two changes per sample in the set of active actuator constraints. For the feasible trajectory, WLS2 almost exactly recovers the optimal solution, while for the infeasible trajectory, the solution quality is somewhat degraded. The reason for WLS2 being so successful can be seen from Figure 3. In most sampling instants, in particular for the feasible trajectory, WLS (without any restriction on the number of iterations) finds the optimum in only one or two iterations. In those instants where WLS needs three or more iterations, WLS2 only finds a suboptimal solution, but can be thought of as retrieving the correct active set a few sampling instants later.

The RPI performance seems difficult to predict. In case 1, the RPI solution satisfies $Bu = v$ at each sampling instant although some control surface time histories differ slightly from the optimal ones produced by SLS and MLS. However, in case 2, the RPI solution is rather degraded. The same goes for case 3 where especially the pitching coefficient is poorly reproduced. The general flaw with RPI is the heuristic rule it is based on which claims that it is optimal to saturate all control surfaces which violate their bounds in some iteration. Note that RPI yields the highest average number of actuator saturations in all cases.

In all of the simulated cases, FXP generates poor, although continuous,

Figure 3: Histograms showing the number of iterations performed by WLS for the three simulated cases. In all cases, it holds that in less than half of the sampling instants more than two iterations are required to find the optimal solution.

solutions to the control allocation problem. The FXP algorithm relies on gradient search methods to update the solution. Gradient search methods are the most successful when the level curves of the cost function are circular so that the gradient points towards the minimum. Here, however, the level curves are stretched out to ellipsoids due to the scaling of the two objectives in (4). This also explains the degradation in quality in case 2 where further weighting is introduced. Neither lowering γ nor increasing the number of iterations significantly increases the FXP solution quality (doubling the number of iterations yields a mean error decrease by 15-25 % in the simulated cases).

Timing Properties

Overall, the timing results for the different methods (except LSQLIN) are within the same order of magnitude. Thus, if RPI is considered a viable alternative for real-time applications, then so should SLS, MLS, and definitely WLS be. For comparison, the typical sampling frequency in modern aircraft is 50-100 Hz which corresponds to a sampling time of 10-20 ms.

WLS2 and FXP both have a maximal computation time which by construction is independent of the trajectory type. This is appealing since it makes the control allocation task easy to schedule in a real-time implementation.

Implementation

FXP is the most straightforward algorithm to implement, consisting mainly of matrix multiplications and saturations.

RPI is also easy to implement and relies heavily on computing pseudoinverse solutions to underdetermined least squares problems, which in the implementation is done using QR decomposition.

SLS, MLS, and WLS are active set based methods and as such they all require book keeping and maintenance of the working set of active constraints. Solving constrained least squares problems is also central to active set methods and in the implementation this is done using QR decomposition.

All solvers have been implemented as MATLAB M-files and can be downloaded from

<http://www.control.isy.liu.se/~ola/>

6 Conclusions

The main conclusion that can be drawn from our investigations is that classical active set methods seem well suited for real-time aircraft control allocation. The computational complexity is similar to that of the redistributed pseudoinverse method and the solutions produced are in general of the same or better quality.

7 Acknowledgment

The author would like to express his gratitude to Emeritus Professor Åke Björck at the Department of Mathematics, Linköpings universitet, for remembering and suggesting the 1984 paper by Lötstedt [13].

Algo- rithm	Mean time (ms)	Max time (ms)	Mean error	Max error	Satur- ations	Iter- ations
SLS	1.12	2.44	0	0	1.8	2.5
MLS	1.01	1.83	0	0	1.8	2.5
WLS	0.56	1.17	1.64e-5	3.12e-5	1.8	1.5
WLS2	0.55	0.77	1.63e-5	3.12e-5	1.8	1.5
RPI	0.91	1.50	0	0	2.0	2.4
FXP	1.97	1.99	1.56e-2	2.86e-2	0.5	50.0
LSQLIN	15.04	18.57	0	0	1.8	–

Table 2: Control allocation results for case 1.

Algo- rithm	Mean time (ms)	Max time (ms)	Mean error	Max error	Satur- ations	Iter- ations
SLS	1.30	3.41	0	0	2.9	2.9
MLS	1.12	2.75	0	0	2.9	2.8
WLS	0.60	1.84	1.34e-4	6.35e-4	2.9	1.8
WLS2	0.56	0.82	1.06e-3	2.13e-2	3.1	1.5
RPI	1.09	1.84	1.11e-2	9.03e-2	4.3	3.0
FXP	1.98	1.98	4.00e-2	7.12e-2	1.0	50.0
LSQLIN	16.84	22.00	0	0	2.8	–

Table 3: Control allocation results for case 2.

Algo- rithm	Mean time (ms)	Max time (ms)	Mean error	Max error	Satur- ations	Iter- ations
SLS	0.80	3.28	4.51e-3	1.25e-2	6.1	2.5
MLS	0.89	2.52	4.51e-3	1.25e-2	6.1	2.5
WLS	0.68	2.11	4.43e-3	1.23e-2	6.1	2.5
WLS2	0.51	0.79	1.04e-2	2.68e-2	6.4	1.6
RPI	0.94	1.46	3.00e-2	1.09e-1	7.6	2.6
FXP	1.98	2.01	1.34e-2	2.11e-2	1.0	50.0

Table 4: Control allocation results for case 3.

Figure 4: Simulation results for case 1. Left: Commanded (gray) vs. generated (black) virtual controls. Middle and right: Left horizontal tail (u_1) position and rate. The dashed lines mark the actuator constraints.

Figure 5: Simulation results for case 2. Left: Commanded (gray) vs. generated (black) virtual controls. Middle and right: Left horizontal tail (u_1) position and rate. The dashed lines mark the actuator constraints.

Figure 6: Simulation results for case 3. Left: Commanded (gray) vs. generated (black) virtual controls. Middle and right: Left horizontal tail (u_1) position and rate. The dashed lines mark the actuator constraints.

References

- [1] Åke Björck. *Numerical Methods for Least Squares Problems*. SIAM, 1996.
- [2] Marc Bodson. Evaluation of optimization methods for control allocation. In *AIAA Guidance, Navigation, and Control Conference and Exhibit*, Montreal, Canada, August 2001.
- [3] Marc Bodson and William A. Pohlchuk. Command limiting in reconfigurable flight control. *Journal of Guidance, Control, and Dynamics*, 21(4):639–646, July–August 1998.
- [4] J. Buffington, P. Chandler, and M. Pachter. On-line system identification for aircraft with distributed control effectors. *International Journal of Robust and Nonlinear Control*, 9(14):1033–1049, 1999.
- [5] James M. Buffington and Dale F. Enns. Lyapunov stability analysis of daisy chain control allocation. *Journal of Guidance, Control, and Dynamics*, 19(6):1226–1230, November–December 1996.
- [6] John J. Burken, Ping Lu, Zhenglu Wu, and Cathy Bahm. Two reconfigurable flight-control design methods: Robust servomechanism and control allocation. *Journal of Guidance, Control, and Dynamics*, 24(3):482–493, May–June 2001.
- [7] Wayne C. Durham. Constrained control allocation: Three moment problem. *Journal of Guidance, Control, and Dynamics*, 17(2):330–336, March–April 1994.
- [8] Wayne C. Durham and Kenneth A. Bordignon. Multiple control effector rate limiting. *Journal of Guidance, Control, and Dynamics*, 19(1):30–37, January–February 1996.
- [9] Rowena L. Eberhardt and David G. Ward. Indirect adaptive flight control of a tailless fighter aircraft. In *AIAA Guidance, Navigation, and Control Conference and Exhibit*, pages 466–476, Portland, OR, 1999.
- [10] Dale Enns. Control allocation approaches. In *AIAA Guidance, Navigation, and Control Conference and Exhibit*, pages 98–108, Boston, MA, 1998.
- [11] Gene H. Golub and Charles F. Van Loan. *Matrix Computations*. John Hopkins University Press, third edition, 1996.

- [12] Yutaka Ikeda and Mark Hood. An application of L1 optimization to control allocation. In *AIAA Guidance, Navigation, and Control Conference and Exhibit*, Denver, CO, August 2000.
- [13] Per Lötstedt. Solving the minimum least squares problem subject to bounds on the variables. *BIT*, 24:206–224, 1984.
- [14] Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer, 1999.
- [15] Ashutosh Sabharwal and Lee C. Potter. Convexly constrained linear inverse problems: Iterative least-squares and regularization. *IEEE Transactions on Signal Processing*, 46(9):2345–2352, September 1998.
- [16] John C. Virnig and David S. Bodden. Multivariable control allocation and control law conditioning when control effectors limit. In *AIAA Guidance, Navigation, and Control Conference and Exhibit*, Scottsdale, AZ, August 1994.