Real-time control allocation using zonotopes

Max Demenkov

\(^1\)Institute of Control Sciences, Russian Academy of Sciences
Moscow, Russia

Part of this research is done in collaboration with De Montfort University (UK), QinetiQ Ltd (UK) and Central Aerohydrodynamic Inst. (TsAGI, Russia)
What is control allocation?

- Future mechanical systems are designed to be over-actuated (i.e. they have more actuators than needed for control)
- This is done to add additional control capabilities and guarantee reliability in case of actuator fault
- In order to simplify control and/or avoid control redesign and tuning when faults occur, the main controller produces the desired vector of moments, or angular rates, considered as virtual actuators
- Then a control allocation algorithm commands physical actuators to provide the desired moments or angular rates
The origins of control allocation problem


- Later, the problem attracted attention in general control journals and meetings, where it is mostly considered together with the system and actuator dynamics.

- We consider here only stand-alone problem of control allocator design for 2D/3D case, the same way as it was posed within aerospace community—without consideration of dynamics.
An example: X-33, prototype of unmanned reentry vehicle

Vertical rudders
- Rudders: 60° outboard and 30° inboard deflection
- Electromechanical actuators
- Function: yaw control and pitch trim bias

Elevons
- Inboard and outboard elevons: ±25°
- Electromechanical actuators
- Function: pitch control and roll control at all speeds

Body flaps
- Electromechanical actuators
- Pneumatic load assist device
- Flaps: −15°, 26°
- Function: pitch control at all speeds, yaw control and entry

Attainable angular rates set

The eight control surfaces of X-33 vehicle have control power capable of providing redundant pitch, roll, and yaw restoring moments such that if one surface fails, the potential exists for an alternate control scheme that will maintain control of the vehicle.
An example: Control Moment Gyroscopes (CMG)

CMG is useful for attitude control of a space station or satellite: MIR, International Space Station, imaging satellites, etc…

Control allocation solution designed at CNES (Toulouse, France) and implemented on board of high resolution European satellite “Pleyades” (Thieuw and Marcille, 2007):

\[
h = F(\delta), \quad J(\delta) - \text{Jacobian matrix}
\]

\[
\dot{h} = J(\delta) \dot{\delta} \quad \Rightarrow \quad \dot{\delta} = J^T (JJ^T)^{-1} h
\]
Moore-Penrose Pseudo Inverse: the simplest control allocation solution

\[ y = Bu, \quad u \in R^m, \quad y \in R^n, \quad m \gg n \Rightarrow u = B^T (BB^T)^{-1} y \]

- Minimum normed vector; the solution that requires minimum energy
- In case \( B \) is a Jacobian matrix - excessively large controls near a singular state, where rank of the matrix decreases
- Maximum possible set \( Y \) of attainable moments/rates cannot be realized due to interval constraints on \( u \):

\[
\begin{align*}
&u_{\text{min}} \leq u \leq u_{\text{max}} \\
&Y \supseteq \{ y : u_{\text{min}} \leq B^T (BB^T)^{-1} y \leq u_{\text{max}} \} 
\end{align*}
\]
Linear constraint satisfaction problem

\[ y = Bu, \ U = \{u, \ u_{\text{min}} \leq u \leq u_{\text{max}}\}, \ 0 \in U \]

\[ y_{\text{des}} \rightarrow u_0 \in U, \ \text{fixed time + on-line reconfigurable} \]

Can be easily solved by classical LP or multi-parametric LP, but:

1) safety certification problems in some cases (like in civil aviation) - for a large number of runs, you’ll find several cases when LP cannot find a solution in reasonable time

2) Complexity of on-board software implementation – not many aerospace engineers are also experts in optimization

3) Multi-parametric closed-form solution is good from the viewpoint of certification and quite simple in implementation, but lacks in reconfiguration capability (changing \( B \) requires off-line recomputation of the solution tables)
Geometry of the problem

\[ Y = BU, \quad U = \{u, \ u_{\text{min}} \leq u \leq u_{\text{max}}, \ 0 \in U \} \]

Y is a zonotope – an image of a cube U under affine projection B

Zonotopes have been previously used in:

- Affine arithmetics within interval analysis community
- Reachable set estimation (e.g. Antoine Girard, etc…) and state estimation within control community, including people here at SWIM!

Attainable Moment/Rate Sets

Can be computed in the same way as controllable/reachable regions for linear discrete systems: convex hull, Fourier-Motzkin elimination or other off-line methods can be used.

\[ Y = \sum_{i=1}^{m} b_i[u_{i\min}, u_{i\max}] \]

MATLAB GUI for attainable moment sets analysis – developed at De Montfort University for QinetiQ
Real-time construction of zonotope in the half-plane form

Suppose that we want to maximize a normal to a facet over a zonotope:

\[ d_i^T Bu \rightarrow \max, \]
\[ u_{\min} \leq u \leq u_{\max} \]

\[ d_i^T B \] is just a linear function \( \Rightarrow \) control takes values at the corners of \( U \)

\[ d_i^T Bu \rightarrow \max \Rightarrow d_i^T Bu = \sum_{j=1}^{m} d_i^T b_j \begin{cases} 
    u_{i\min}, & \text{if } d_i^T b_j < 0 \\
    u_{i\max}, & \text{if } d_i^T b_j > 0 \\
    u_{i\min} \text{ or } u_{i\max}, & \text{if } d_i^T b_j = 0 
\end{cases} \]

2D(3D) case: to allow more than 1(2) vertices to be a solution of our maximization problem, we need \( d_i \) to be orthogonal to (more than) one column of \( B \) matrix \( \Rightarrow \) Any facet normal is a cross product of two columns of \( B \) in 3D case and orthogonal to a column in 2D case!
Real-time construction of zonotope in the half-plane form

\[ B = [b_1, \ldots, b_m] \]

3D case: \( d_k = b_i \times b_j = \begin{bmatrix} b_i(2)b_j(3) - b_i(3)b_j(2) \\ b_i(3)b_j(1) - b_i(1)b_j(3) \\ b_i(1)b_j(2) - b_i(2)b_j(1) \end{bmatrix}, \ k = 1, M, \ M = \frac{m!}{2(m-2)!} \)

2D case: \( d_k = \begin{bmatrix} b_k(2) \\ -b_k(1) \end{bmatrix}, \ k = 1, m. \)

\[ U = \{u: |u| \leq u_{\text{max}}\} \Rightarrow \max_{u \in U} (d_k^T Bu) = \sum_{i=1}^{m} d_k^T b_i \text{sign}(d_k^T b_i) u_{\text{max}} \]

System of linear inequalities:

\[ Y = \{y: d_k^T y \leq \max_{u \in U} (d_k^T Bu), \ k = 1, \ldots, M\} \]

Normalization:

\[ d_k = d_k \cdot \max_{u \in U} (d_k^T Bu) \Rightarrow Y = \{y: d_k^T y \leq 1, \ k = 1, \ldots, M\} \]
Generalized zonotope construction in the form of linear inequalities
(post-conference slide)

How to build half-plane representation of a zonotope in $n$-dimensional case?
Basically, in the same way, but $d_i$ is orthogonal now to $n-1$ columns of $B$…

The algorithm has been proposed (independently) in:


Each facet of a zonotope is constructed totally separately from the others, leading to easy parallelization on multi-core and GPU processors. Note that with increasing of dimensionality, the number of zonotope facets is growing very rapidly…

Thanks to SWIM participants Vu Tuan Hieu Le and Xin Chen!
Two ways of determining control vector

Problem: \( y \rightarrow u \in U \)

• **Convex cone method**

  See Demenkov M. (2011). Reconfigurable direct control allocation for overactuated systems. In *Proceedings of 18th IFAC World Congress, Milan, Italy.*

• **Bisection method**


  There are, of course, much more possible ways…. but, here I’m talking about my own ones…
Real-time determination of control: convex cone method

1) To determine in which cone we are (i.e. in which $y$ lies), multiply $y$ to a matrix of facet normals and then take maximum. The cone for which the maximum is achieved, is the one where $y$ lies:

$$y_i = Bu_i$$

$$k = \arg \max_{i=1,M} d_i^T y$$

$$y \notin Y \Rightarrow y = y_i / d_k^T y$$

The procedure was proposed in the context of regulator design using piecewise-linear Lyapunov functions in Blanchini (Automatica, Vol. 31, 1995) - the regulator is designed off-line and is linear in every cone.

An algorithm for splitting $Y$ into convex cones is described e.g. in:

Real-time determination of control: convex cone method

1) To determine in which cone we are (i.e. in which $y$ lies), multiply $y$ to a matrix of facet normals and then take maximum. The cone for which the maximum is achieved, is the one where $y$ lies:

$$k = \arg \max_i d_i^T y, \ i = 1, N$$

2) Find all vertices $y_i = Bu_i$ attributed to $k$-th cone (can be derived from optimization equations considered previously)

2) Find all possible combinations of such vertices. For each combination, solve a system of linear equalities to determine weights $\lambda$

3) Find $u$ as the linear combination of vertices:

$$\lambda_1 y_1 + \lambda_2 y_2 (+\lambda_3 y_3) = y \ \Rightarrow \ u = \lambda_1 u_1 + \lambda_2 u_2 (+\lambda_3 u_3)$$
**Drawbacks of convex cone method**

- Each zonotope cone consists of several subcones which can intersect each other.

- It is not clear how to pick up solution if it belongs to several subcones – for example one can take point in $U$ which is nearest to the previously computed.

- “Interior points” – it is possible to delete them and “order” corner points of a facet, but in this case algorithm depends on tolerances!

- Redundant inequalities- they can be removed, but in this case, as well as with interior points, additional processing is needed.

No possibility of optimization!
Bisection algorithm

Instead of dividing the zonotope into cones, divide the control hypercube into smaller ones and determine whether the given vector $y$ belongs to their images or not.

\[ Y = \{ y : d_i^T y \leq 1, \ i = 1, \ldots, M \} \]

Checking system of linear inequalities

\[ \Delta u = \max_i (u_{i\text{max}} - u_{i\text{min}}) \]

\[ \varepsilon \geq \frac{\Delta u}{2^N} \Rightarrow N \geq \log_2 \left( \frac{\Delta u}{\varepsilon} \right) \]

Tolerance $\varepsilon$ defines time of computation

Exact description of the cube image – no overestimation problem!
Bisection algorithm: additional criterion

Suppose images of both boxes contain $y$ – which one we have to discard?

$Y = \{ y : d_i^T y \leq 1, i = 1,..,M \}$

Checking if previously computed control vector is inside
An example: back to CMG Control Scheme

Torque Commands

STEERING ALGORITHMS

Rate Command to each CMG

Realized Torque

\[ h = F(\delta), \quad J(\delta) - \text{Jacobian matrix} \]

\[ \dot{h} = J(\delta) \dot{\delta}, \quad |\dot{\delta}| \leq \dot{\delta}_{\text{max}} \]
What if Jacobian rank < 3 ?

In case of degeneracy, cross product of any two Jacobian columns gives us normal to a plane.

The desired vector of momentum rates can be projected onto that plane and the problem can be solved in that plane in 2D.
Numerical (2D) example

\[ J(\delta) = \begin{bmatrix}
-\cos(\beta)\cos(\delta_1) & \sin(\delta_2) & \cos(\beta)\cos(\delta_3) & -\sin(\delta_4) \\
-\sin(\delta_1) & -\cos(\beta)\cos(\delta_2) & \sin(\delta_3) & \cos(\beta)\cos(\delta_4) \\
\sin(\beta)\cos(\delta_1) & \sin(\beta)\cos(\delta_2) & \sin(\beta)\cos(\delta_3) & \sin(\beta)\cos(\delta_4)
\end{bmatrix} \]

\[ \delta = [90^\circ \ 0 \ -90^\circ \ 0] \Rightarrow J(\delta) = \begin{bmatrix}
0 & 0 & 0 & 0 \\
-1 & -0.6 & -1 & 0.6 \\
0 & 0.8 & 0 & 0.8
\end{bmatrix} \]

Pyramid type CMG cluster with skew angle \( \beta = 53.13 \)

Projection onto 2D plane:

\[ \dot{\delta} \]

\[ y = [1 \ 0]^T, \ B = \begin{bmatrix}
-1 & -0.6 & -1 & 0.6 \\
0 & 0.8 & 0 & 0.8
\end{bmatrix} \]

Numerical (2D) example: bisection method

By solid lines we depict the attainable set $Y$ for the box that has been chosen by the algorithm at this iteration, while dashed lines represent the one for the box that has been deleted.

The cross inside the circle represents the commanded vector $y$.

The solution after 32 bisections:

$$\begin{bmatrix} -0.5078 \\ -0.0078 \\ -0.5078 \\ -0.0078 \end{bmatrix} \leq \dot{y} \leq \begin{bmatrix} -0.5 \\ 0 \\ -0.5 \\ 0 \end{bmatrix}$$
Current research: optimization

- Divide $U$ into many boxes

- Use some convex function and interval analysis to discard boxes

$U = U_1 \cup U_2 \cup U_3 \cup U_4 \cup U_5 \cup U_6 \cup U_7 \cup U_8 \cup U_9 \cup U_{10} \cup U_{11} \cup U_{12}$

$F(u) = c_2$

$F(u) = c_1$

$c_2 < c_1$
Conclusion

• It is possible to construct 2D/3D zonotopic sets in real time, in the form of linear inequalities and compute control in real-time from the given matrix B and control constraints (Fault Detection and Isolation is required)

• Bisection-based control allocation algorithm is discussed

• It has a guarantee of obtaining the solution for every given desired vector, control effectiveness matrix and the set of control constraints in a finite and known in advance number of steps

• The proposed method possess the property of utilizing the whole attainable set and has relatively low algorithmic complexity

Further research is connected with adding optimization capabilities and higher-dimensional extension
Incomplete list of references in control allocation


Incomplete list of references in control allocation


Incomplete list of references in control allocation


