OPTIMIZED THRUSTER CONTROL ALGORITHM FOR DRAG-FREE SPACECRAFT

D. Bindel, ZARM – University of Bremen, 28359 Bremen, Germany
S. Theil, DLR, 28359 Bremen, Germany
F. Cirillo, Astrium GmbH, 88039 Friedrichshafen, Germany
P. Gath, Astrium GmbH, 88039 Friedrichshafen, Germany

Overview

Several space missions for fundamental physics research require a disturbance-free environment for their demanding science experiments. The concepts of these missions incorporate a drag-free technology for the spacecraft in order to counteract any external forces and torques. With a disturbance force, originated by the solar radiation pressure and other effects in the range of several Micronewton, micro-propulsion systems are necessary for the spacecraft AOCS. Today, electrical and coldgas engines are available or in development to provide a continuous and controlled thrust in the Micronewton range. Though, the different technologies for the thrust production have different advantages and disadvantages. An ideal thruster control algorithm should be able to deal with these different attributes in an optimal manner.

The content of this paper is the analysis of the currently used methods to control the spacecraft thrusters of a drag-free mission like MICROSCOPE, LISA Pathfinder or LISA and an introduction of a more advanced method. It shows the basic equations and problems of a Thruster Actuation System (TAS) for a simultaneous control of six degrees of freedom.

The focus is on the modification of the standard linear programming algorithm SIMPLEX to meet robustness requirements for the application on a satellite onboard system. This new method takes care of the thrust range limitations (Control Authority) and the dynamics (change of thrust level) of the engines. As a true optimization algorithm, it also provides suitable control strategies for different types of thrusters. In comparison to the standard optimization method, the required memory and computational effort could also be decreased by a special modification, called “Solution Range Placement”.

The algorithm that is developed is finally applied in a closed-loop, nonlinear simulation environment for the LISA mission. In this context, the science mode performance is demonstrated in which two proof masses on board of the LISA spacecraft are isolated from external disturbances along a sensitive axis down to a level of 3×10^{-15} m/s^2/sqrt(Hz).

At the end of the paper, a small overview is given, how numerical instabilities in the original optimization algorithm can be avoided.

1. BASIC DEFINITIONS

1.1. Abbreviations

AOCS - Attitude and Orbit Control System
TAS - Thruster Actuation System
FEEP - Field Emission Electric Propulsion
SRP - Solution Range Placement
LISA - Laser Interferometer Space Antenna

1.2. Symbols

A - Thruster configuration matrix
A^+ - Pseudoinverse of matrix A
F - vector of requested forces and torques
F_+ - positive elements of force/torque command F
F_- - negative elements of force/torque vector F
A_{plus} - pseudoinverse matrix of A for use with F_+
A_{minor} - pseudoinverse matrix of A for use with F_-
T_i - Thrust generated by engine i
T_n - Null space thrust vector
T_p - preliminary thrust vector
$T_{min,i}$ - minimum thrust of an engine
$T_{max,i}$ - maximum thrust of an engine
ΔT_{max} - maximum thrust change
n - number of available thrusters on spacecraft
c_F - command scaling coefficient
c_{Tn} - Null space vector scaling coefficient
c_s - coefficient for static robustness
c_d - coefficient for dynamic robustness
F_{old} - previously generated force/torque vector
F_{diff} - difference between F and F_{old}
W_i - wear indicator for each of the thrusters
$T_{R,i}$ - virtual thrust for SRP calculation
z - objective function for linear optimiser

e - objective function perturbation vector

2. INTRODUCTION

The task of a thruster actuation system is to drive the available thrusters of a spacecraft in such a way, that the demanded forces and torques of the AOCS controller are achieved.

FIGURE 1. TAS principle

The TAS is an element in the AOCS chain of a spacecraft. It translates the force and torque demands of the controller to dedicated thrust commands for the engines. The difficulty of this task is the problem, that the thrusters may not be aligned parallel to the axis of the satellite. As you can see in the left example of Figure 2 it is possible to route the demanded forces and torques to the engines, that are aligned with the corresponding axis.

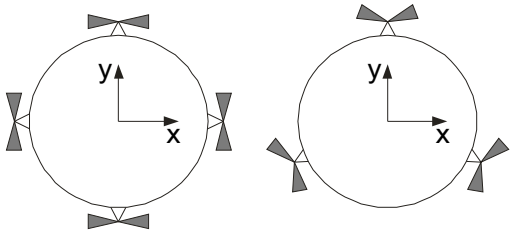

FIGURE 2. Trivial and complex thrusters configuration

Nevertheless, real spacecraft have some restrictions for placing their thrusters like plume contamination, mass-limitations or the position of optics or sun-shields. The position of the engines is most times a complex configuration like the right example of Figure 2. As a result the installed thrusters are acting on several degrees of freedom simultaneously when they are fired. An appropriate algorithm is therefore needed to drive the thrusters with respect to their complex configuration.

This leads to the basic equation Eq. (1) of the thruster actuation system. The demanded force and torque vector *F* is assumed here to be six-dimensional, so the TAS has to handle the three degrees of freedom for each translation and rotation. The configuration map *A* reflects the position and orientation of all spacecraft thrusters. It has the dimension 6 x *n* for *n* thrusters. By multiplying the thrust of each engine (vector *T*) to the matrix, the forces and torques applied to the spacecraft are calculated.

(1) $$F = AT$$

Unfortunately, the desired thruster command vector *T* can not be achieved by simply inverting the matrix *A*, because it is not quadratic. Furthermore, the problem is underdefined, so there exist a large number of possible solution vectors *T* for the given problem. It is the task of the TAS to find one of them, if it exists.

This is illustrated in Figure 3. For a commanded force and torque vector *F* and a constant configuration map *A* exist more than one solution *T*, assuming that the control authority of the spacecraft is not exceeded. Some of these solutions represent vectors with optimal attributes, like least fuel or power consumption or least engine thrust. They are displayed as points $T_{A,B,C}$ and depend from the given problem.

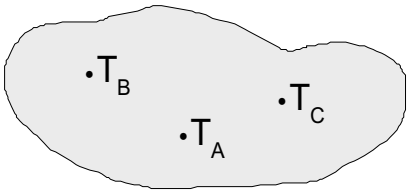

FIGURE 3. Range of possible solutions for Eq. *F=AT*

2.1. Boundary Conditions

The solution range is further decreased, if additional boundary conditions are introduced. This is necessary, to adapt the final solution to the physical behavior of the engines. One example is the thrust range of the engines. They are not only characterised by their overall minimum and maximum thrust (see Eq. (2)), but can also be affected by dynamic effects. Some electrical engines incorporate a controlled gas or fluid feed, that has an influence on the generated thrust. The result is, that the thrusters may not be able, to increase or decrease their thrust output in a short time from minimum to maximum and the other way round. So it may able to vary the thrust only in a certain range, depending on its previous thrust level (see Eq. (3)).

(2) $$T_{\min} \le T \le T_{\max}$$

(3) $$\left|\Delta T_i\right| \le \Delta T_{\max}$$

Another boundary condition can be the overall consumed propellant massflow rate. This can apply to systems with a single main tank and a limited massflow rate for all thrusters. Such a situation is rare, but it exist like the cryogenic missions Gravity Probe B, where the spacecraft produced its propellant by itself by boiling off the liquid helium. Eq. (4) describes this restriction.

(4) $$\left\|T\right\|_1 = \sum_i \left|T_i\right|, \left\|T\right\|_1 \le \dot{m}_{\max}$$

2.2. Optimization Criteria

As explained in the previous section, the range of possible solutions contain many different combinations for the thrust vector *T*. By defining some criteria, an optimization method could find the best suitable vector from the band of all possible solutions.

To ensure a good handling of the real thrusters, the optimization should reflect a physical property of the engines. In this case, a reduction of the fuel consumption can be an optimization aspect. Depending on the utilized thruster-technology, there are two different approaches.

Coldgas systems and a lot of electrical engines are using a single propellant reservoir. Also a lower thrust will lead to

a lower fuel consumption. So by minimizing the sum of all thruster commands in the vector *T*, the massflow will also be minimized. In Eq. (5) this is displayed as a minimum of the 1-Norm of the vector *T*.

$$\min_T \|T\|_1 = \min_T \sum_i |T_i| \tag{5}$$

Other types of engines like the FEEP have an own tank for every thruster. Here it is better, to reduce the thrust of each thruster, to preserve propellant. The Eq. (6) shows, how the Infinity-Norm of the vector *T* is minimized to find that optimal solution.

$$\begin{aligned} \|T\|_\infty &= \max(|T_i|), \; so\, that \\ \min_T(\|T\|_\infty) &= \min_T(\max(|T_i|)) \end{aligned} \tag{6}$$

The same procedure can be used, to apply a degradation-optimization. Primarily the electrical engines are affected with the wear of their components. It could be effective, to track down the engine degradation and to use this information in the optimization methods. The analysis of the thrusters commands over the lifetime can give an assumption about the cumulative wear W_i of a thruster. It tells, how much that particular engine have been degraded or worn out. In the second step, this information can be used, to improve the infinity norm optimization (see Eq. (7)). The system reduces the applied thrust for especially that thrusters, that have been used intensely in the past time.

Long term tests with FEEP [1] engines have shown, that the degradation is not a linear function of the applied thrust. So a nonlinear function *Useage(T)* for the actual usage of the thruster should be established for calculating the wear.

$$\begin{aligned} W_i &= \int_0 Useage(T_i)dt \\ \min_T(\|T(W)\|_\infty) &= \min_T(\max(|W_i \cdot T_i|)) \end{aligned} \tag{7}$$

The optimization with respect to degradation is an interesting feature, that may help to enlarge the lifetime of FEEP and colloid thrusters. But grid ion thrusters are also affected by erosion due to intensive useage.

On the other hand, the power consumption of electrical engine can also be an issue of optimization, or at least a limiting factor of the power supply or the propulsion system. Here the vector norm 2 can be used, to express the consumed power, because it also involves the square value of the thrust (see Eq. (8)). Although, an optimization can also been done, without calculating the square-root.

$$\min_T(\|T\|_2) = \min_T(\sqrt{\sum_i T_i^2}) \tag{8}$$

It should be noted, that all optimized solutions for the different vector norms of *T* are located inside the solution range. The Figure 3 depicts that fact.

3. ALGORITHMS

There are different approaches, to find the desired thrust solution vector *T*. The state of the art methods are using a pseudoinverse matrix. A more complex attempt are linear programming methods. Both have their advantages and disadvantages.

3.1. Pseudoinverse Method

The basic idea behind that algorithm is a so called pseudoinverse matrix A^+ that meets the requirement of Eq. (9).

$$A \cdot A^+ = I \tag{9}$$

As described in the introduction, there does not exist a single inverse of the configuration matrix *A*. The A^+ must be calculated by a different way. Jeng-Heng Chen [2] introduced the Moore-Penrose (Eq. (10)) inverse to solve this problem.

$$A^+ = A^T(AA^T)^{-1} \tag{10}$$

The calculated pseudoinverse matrix has the features of Eq. (9), but a direct solution for *T* is impossible without the additional nullspace vector T_n. Eq. (11) shows the main feature of this vector, because it is a thrust solution, that does not apply any resulting force or torque to the spacecraft.

One of the main problems of the pseudoinvers approach is the fact, that a simple multiplication of the matrix A^+ and the vector *F* will lead to negative values in the solution vector *T*. That is impossible to realize, because the engines can produce only positive thrust. By adding a nullspace vetor T_n of an appropriate size (Eq. (12)), the commanded thrust will be shiftet up, while the resulting forces and torques stay unaffected.

$$AT_n = 0, \quad so\, that \tag{11}$$

$$\begin{aligned} T &= A^+F + T_n, \quad with \\ T_n &\geq 0 \; and \; T_i \geq T_{\min,i} \end{aligned} \tag{12}$$

In terms of fuel consumption or optimization, the nullspace vector T_n is very important. Like the standard problem Eq. (1) there exist a band of different solutions for the problem in Eq. (11) and with this a T_{n_min} and T_{n_max}. As a first step, the final T_n can be computed from the average of these two vectors (Eq. (13)) and used as a fixed bias in Eq. (12). It should be noted, that solving the Eq. (11) to retrieve the nullspace vectors is not done on board the spacecraft, but offline by using optimization algorithms (least squares method).

$$T_n = (T_{n\_\min} + T_{n\_\max})/2 \tag{13}$$

Here the pseudoinverse methods faces one of its major disadvantages. It has no mechanism, to ensure a solution, where every thruster fits into its own thrust boundaries (Eq. (3)), but it can barely try to find a thrust vector T that meets the requirements of Eq. (2}).

The pseudoinverse approach can not offer the variety of the complete six-dimensional solution space like in Figure 3, that solves the standard problem. It can only generate one fixed point solution, or a line at least, if the nullspace vector is seen as a variable. So T_n is one of the few ways, to influence the calculated pseudoinverse solution.

Beside the fixed bias as in Eq. (13) there is also a method, to scale the nullspace vector T_n, so that all elements of the thrust command vector T are above or equal to the global minimum thrust T_{min} of the engines.

(14) $$T_p = A^+ F$$

(15) $$R_i = (T_{\min} - (T_p)_i)/(T_n)_i$$

(16) $$T_{n\_dyn} = \max(R_i) T_n$$

(17) $$T = T_p + T_{n\_dyn}$$

One way to do that, is a dynamic nullspace vector bias, that uses a preliminary thrust solution T_p (Eq. (14)) and a ratio R_i (Eq. (15)) for every engine to retrieve the maxim of R_i. The dynamic nullspace vector T_{n_dyn} is then calculated with this ratio and the fixed nullspace vector T_n (Eq. (16)). Afterwards the final thrust command vector T is composed from the preliminary solution T_p and the dynamic T_{n_dyn} (Eq. (17)).

Another approach to eliminate low thrust commands is an iterative process, that Hai Ping Jin [3] introduced in 1996.

(18) $$F_c(0) = F$$

(19) $$T_p = A^+ F_c(k) + T_n$$

(20) $$T = (T_p + |T_p|)/2$$

(21) $$F_c(k+1) = F_c(k) + (F - AT) K_{iter}$$

It is based on the calculation of a preliminary thrust vector T_p (Eq. (19)), and an intermediate force/torque vector F_c (Eq. (21)). The idea behind that system is, that Eq. (20) will change the thrust vector T unless, it has all positive values. The resulting force/torque error is applied with a gain factor K_{iter} and fed back into the loop.

In this method the nullspace vector T_n stays unaffected and the intermediate force/torque vector F_c is changed, until the thrust vector T contains no more negative values, and the difference of commanded and calculated forces and torques $(F - AT)$ is below a certain threshold value. To speed up the convergence, the gain factor K_{iter} is used with values that are typically around 2 or 3.

One of the latest developments on the topic of the pseudoinvers thruster actuation systems was the introduction of the least squares thruster dispatching (LSQ dispatching) in 2004 by Denis Fertin [4] and Shu-Fan Wu. It is an approach, to avoid negative thruster commands from the first step of calculation by using two different and complementary pseudoinverse matrices A_{plus} and A_{minor}. The method also splits the commanded force/torque vector into its positive and negative components (Eq. (22)). The negative elements of F_+ and the positive elements of F_- are replaced by zeros.

(22) $$F = F_+ + F_- \; with \; F_+ \geq 0 \; and \; F_- \leq 0$$

The two complementary pseudoinverse matrives are constructed in a way, that they meet the requirements of Eqs. (23) and (24) and consist only of elements greater or equal to zero.

(23) $$A \cdot A_{plus} = I$$

(24) $$A \cdot A_{\min or} = -I$$

By redefining the original pseudoinverse equation to the new system (Eq. (25)) it is guaranteed, that the vector T is always positive. That is founded in the fact, that $A_{minor} > 0$ and $F_- < 0$, so the second term of the new equation is always negative and thus the result of the whole equation is always positive.

(25) $$T = A_{plus} F_+ - A_{\min or} F_-$$

Unfortunately, this least squares thruster dispatching can only guarantee thrust commands above or equal zero. For a solution, that meets at least the requirements of minimum thrust as described in Eq. (2) there is again a nullspace vector application required.

In 2004 Alexander Schleicher [5] introduced a scaling method (Eq. (26)), to bring the smallest element in the thrust command vector T to exactly the minimum thrust T_{min}.

(26) $$T = c_F (A_{plus} F_+ - A_{\min or} F_-) + c_{Tn} T_n$$

Here it can be seen, that there are two scale factors c_F and c_{Tn} to influence the nullspace vector and the commanded force/torque vector. The requirements that have to be met, are displayed in Eqs. (27) and (28). The factor c_F should be as close as possible to 1.0, to avoid large differences between the commanded and actuated force and torque vector F. Furthermore c_{Tn} should be as close as possible to zero in order to avoid excessive thrust commands and fuel consumption.

(27) $$\min(T) = T_{\min}$$

(28) $$\max(T) \leq T_{\max}$$

The major advantage of this algorithm is the ability, to scale down the demanded force/torque vector, if the control authority of the propulsion system is exceeded.

The biggest advantage of all the pseudoinverse algorithms is the simplicity of their construction. The solution is found basically by multiplying matrices and vectors. Beside the

iterative approach of Jin, this involves no iterative loops. The big disadvantage is, that the possible six-dimensional space of solutions is reduced to only a small band of vectors. Thus, there is no real optimization of the final solution possible. But anyway, the scaling methods are quite effective in reducing the fuel consumption. There is still the problem, that the algorithm might violate the thrust requirement of Eq. (3). For this reason, the AOCS controller has to be tuned to a point, where it commands no big variations of the force/torque vector F to the TAS.

3.2. Linear Programming

As depicted in Figure 3, the range of possible solutions for the original problem $F=AT$ is very large. Due to the linearity of the system of equations it is obvious that a linear programming method could be used, to find a proper solution T.

In the analysis of this topic, the Simplex algorithm was chosen as a linear optimizer to solve the set of equations and restraints. The advantage is, that such a method is able, to apply some true optimizations. So there will be first an introduction to the problem definition for this numeric system.

The optimization for the vector norm 1 (Eq. (4)) is the most simple method, because the objective function z of the optimizer is simply the sum of all elements in the vector T (Eq. (29)). One of the features of this specific simplex implementation is the fact, that its output is by definition greater equal zero.

But anyway it is possible, to apply restrictions to the thrust range for each thruster (Eq. (31)) separately to meet the requirements for thrust variations and the maximum thrust range in one step. Overall, this method requires six equalities in the form of Eq. (30) to fullfill the basic task, and two times the number of thrusters as inequalities, to incorporate the thrust range restrictions.

$$\min_T(z) = \min_T \sum_i T_i \tag{29}$$

$$subject\ to\ AT = F \tag{30}$$

$$and\ T_{\min,i} \le T_i \le T_{\max,i} \tag{31}$$

The search for a solution, that represents an optimal infinity norm (Eq. (5)) of the vector T is more complex. The objective function in Eq. (32) is reduced to z, so the algorithm will try to minimize it. Due to the additional inequalities in Eq. (33) every thruster will be kept below this value z. By this way, the result is a thrust vector T with a minimized maximum element.

$$\min z \tag{32}$$

$$subject\ to\ T_i \le z \tag{33}$$

$$and\ AT = F \tag{34}$$

$$and\ T_{\min,i} \le T_i \le T_{\max,i} \tag{35}$$

The implemented linear optimizer Simplex has also some special features that have to be considered. The major difficulty is the correct setup of the linear program with its equalities and inequalities. The setup must also be able to handle the characteristics of the optimizer. The numeric optimizer is not able to handle an infeasible problem. That situation will occur, if the commanded force/torque vector F exceeds the control authority. On the other hand, the algorithm has the ability, to find the global optimal solution in a finite amount of iterations, if such a solution exists. This is a major difference to other optimisers, that are iterating, until the solution fulfils a certain abort-accuracy.

3.2.1. Robustness

So one of the most important components for a correct linear programm setup is an enhancement to pass only feasible problems to the Simplex method. This is necessary, to guarantee a valid and robust thruster command vector T in the realtime environment onboard the spacecraft for every AOCS loop.

Two different failure cases have been identified, where a solution of the commanded vector F is not feasible with the given thrust restrictions of the engines.

The first situation is an exceeding of the control authority. The propulsion system is not able, to provide a force or torque vector of such a big length, as it is demanded from the AOCS controller. One possible solution is to scale down the commanded vector F until it is again feasible to be solved. A static robustness coefficient c_s is used to do so.

In Eq. (36) the extension of the original TAS problem is described. For c_s being zero, the equation represents the original commanded force/torque vector F. For any value between 0 and 1 of c_s, the vector F is scaled down until it is 0. By assuming, that the propulsion system is able to provide a state, where no forces and torques are acting on the spacecraft, this coefficient will lead to a linear program, feasible to solve within the current control authority.

$$\begin{aligned} AT &= F - c_s F \\ AT + c_s F &= F \\ with \quad & 0 \le c_s \le 1 \end{aligned} \tag{36}$$

Due to the physical properties of the proposed electrical engines, the dynamic behaviour must also be considered. Assuming a continuous working state of the propulsion system, it may not be able to reach a nullvector (zero forces and torques). A restriction of the feasible thrust variation (Eq. (3)) may prevent this. So a further extension of the main TAS equation is necessary.

For this purpose, the dynamic robustness coefficient c_d is introduced. It uses the difference vector F_{diff} (Eq. (37)) between the commanded F and the previously actuated F_{old}, to make the linear program robust against oversized command variations. In Eq. (38) this is shown as an equality constraint for the TAS problem. In Eq. (39) the combined static and dynamic coefficients are inserted into the linear program.

(37) $$F_{diff} = F - F_{old}$$
$$AT = F - c_d F_{diff}$$

(38) $$AT + c_d F_{diff} = F$$
$$with \quad 0 \le c_d \le 1$$

(39) $$AT + c_s F + c_d F_{diff} = F$$

To use the proposed coefficients in a linear program for the optimization algorithm, additional inequalities for the range of c_s and c_d must be introduced and the objective function must be updated. For the norm 1 optimizer this is done in Eq. (40) and for the infinity norm optimizer in Eq. (41). The value of the scaling weights σ and Φ should be at least three orders of magnitude higher than the maximum thrust of the engines. Simulations have shown, that this penalty values have almost no influence to the calculated thrust command solution. Comparing results from non-enhanced linear programs and robust variations indicate differences in the range of ten orders of magnitude lower than the commanded and actuated vector F.

(40) $$\min_T(\sum T_i + \sigma c_s + \phi c_d)$$

(41) $$\min_T(z + \sigma c_s + \phi c_d)$$

(42) $$\sigma = \phi \approx 1000 \cdot T_{\max}$$

3.2.2. Computational effort

One of the issues, that has to be tackled when developing a TAS for a real time environment at an spacecraft computer is the amount of numeric resources. An iterative optimisation algorithm can be a risk for the timing of the task schedules. By using the simplex implementation for the linear programming, at least the problem of the iteration limitations is fixed.

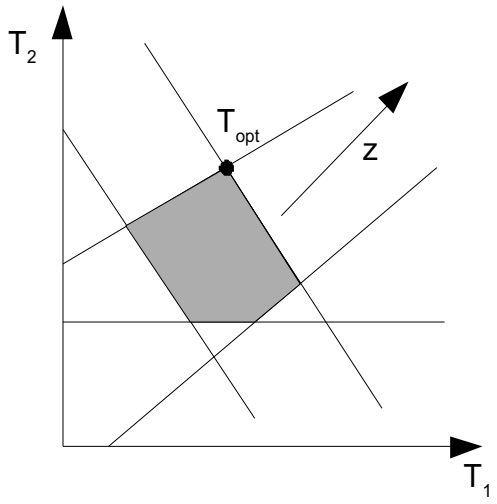

FIGURE 4. Simplex solution principle

In Figure 4 is the principle of a two-dimensional linear problem displayed. There exists a gray field of possible solutions, that is limited by a set of linear inequalities. The vector z shows the gradient of the objective function. It is obvious for this kind of problem, that there exists only one corner or edge, that is the global optimum T_{opt}. So the simplex algorithm works in a way, that it inspects the corner points of the linear bound solution space, to find the optimum. That is the reason, for the theoretical limited number of iterations, that the optimiser needs, to get the global optimum.

Another interesting question is the way how the linear problem is prepared, before it is fed to the optimiser. Similar to the modification for the robustness in section 3.2.1. a modification was developed, to increase the usability for a real time application.

The Solution Range Placement utilises the attribute of the simplex implementation, that it can only calculate results, that are greater or equal to zero. So all lower boundaries for the thrusters are shifted toward zero (Eq. (47)).

(43) $$T_{\min,i} \le T_i \le T_{\max,i}$$

(44) $$T_i = T_{R,i} + T_{offset,i}$$

(45) $$with\ T_{offset,i} = T_{\min,i} \ \ follows$$

(46) $$T_{\min,i} \le T_{R,i} + T_{\min,i} \le T_{\max,i}$$

(47) $$0 \le T_{R,i} \le T_{\max,i} - T_{\min,i}$$

The known linear programming problem in Eq. (39) was altered accordingly in Eq. (50).

(48) $$F = AT = A(T_R + T_{offset})$$

(49) $$F = AT_R + AT_{\min}$$

(50) $$AT_R + c_s F + c_d F_{diff} = F - AT_{\min}$$

After retrieving optimal values for the variables T_R, c_s and c_d the required solution vector T is calculated by Eq. (44).

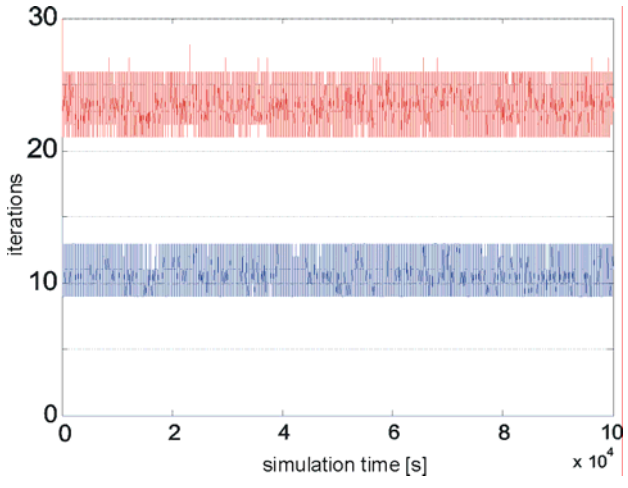

FIGURE 5. Number of iterations with and without SRP

The number of iterations can be reduced to roughly half of the original value, due to the Solution Range Placement (see Figure 5). The resulting thrust vector T is identical in both cases.

3.2.3. Wear optimisation

In the section 2.2 the effect of wear or degradation was introduced and described in Eq. (7). The linear optimisation offers a very smooth way to implement this feature to the objective function.

$$\min_{T}\left(\sum W_i T_{R,i} + \sigma c_s + \phi c_d\right) \tag{51}$$

Whereas the weightings of the vector W can be fixed values, that are uploaded by the satellite command center, and are based on lifetime expectations and analysis of the engines.

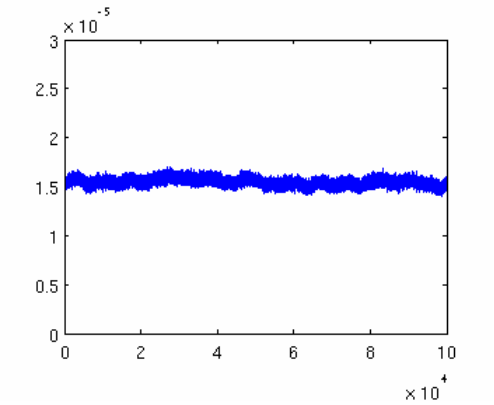

FIGURE 6. Example of LISA thruster with Wear factor 1.0

In a simulation for a typical LISA science mode, the wear factor for one thrusters was changed from 1.0 to 2.0, to simulate a degraded engine. The results are display in Figure 6 and 7.

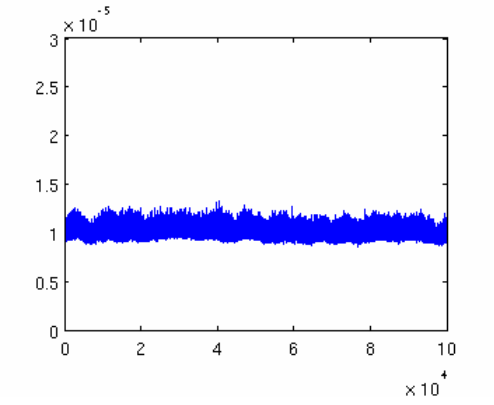

FIGURE 7. Example of LISA thruster with Wear factor 2.0

The optimisation algorithms shifted automatically shifted workload to this thrusters and so this degraded engine may extend its lifetime a little bit. Of course other thrusters of the spacecraft have to take over the missing thrust from this engine. But anyway, this feature can be interesting to the end of a deep space mission, when the hardware is on its limits and should be handled with the most care. Due to Eq. (51), the Wear vector is constant during one calculation and acts as a weighting factor in the objective function. So the wear optimisation requires no additional computational effort.

4. COMPARISON

From the huge number of available algorithms, four candidates have been selected, to compare its performance in a closed-loop simulation for the LISA mission.

The state-of-the-art technology is demonstrated by pseudoinverse algorithms. A deviate of the Moore-Penrose and one of the LSQ Dispatching technology have been used for this task.

The linear programming was performed with a Norm 1 and a Norm Infinity optimiser, based on a Solution Range Placement implementation.

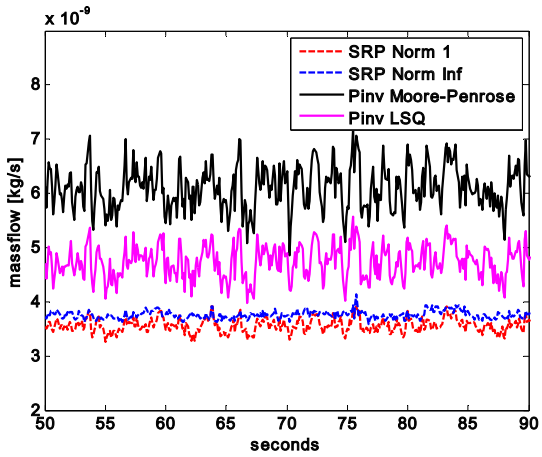

FIGURE 8. Massflows for LISA science mode

The result of the comparison can be seen in Figure 8. An interesting fact is, that the implementations of linear programming deliver almost the same result, although they are using different objective functions. A numeric overview about the results is shown in Tab. 1.

Algorithm	Overall Impulse	Max. Single Impulse
Pinv MP	12.7	2.2
Pinv LSQ	15.4	2.6
SRP Norm 1	10.1	2.4
SRP Norm Infinity	10.7	1.7

TAB 1. Avg. impulse for LISA science mode in [Ns/day]

The linear programming performance in terms of fuel consumption, or required impulse in this case, depends heavily on the thruster configuration and the mission scenario. So test cases with other missions may differ in terms of massflow ratio between psudoinverse methods and linear programming.

The results of the maximum single impulse in Tab. 1 is an index for the distribution of the workload to every thruster. The lower the maximum single impulse is, the lower is the usage of every thruster. The norm infinity optimisation tries to distribute the thrust as evenly as possible to the available thrusters. As a result, the engines are worn out more equally. In a long term mission, that may have a positive impact on the thrusters lifetime.

5. NUMERIC INSTABILITIES

One of the difficulties of the linear programming algorithm Simplex is the problem of numeric instabilities. The optimiser is able to find the global solution of a linear problem, if this solution exists. By the definition of a linear problem, the optimal solution can either be a point or an edge.

Figure 9 shows such an example of a two-dimensional, ill-conditioned problem. The boundary conditions of the thrusters form a square area, but the equality constraint *AT=F* is a single line. If this line is perpendicular to the gradient of the objective function, then all points of this line bear the same objective value. So the whole line is the optimal solution. Small variations of the force/torque command *F* and numeric instabilities lead to an optimiser result, that is either on one or the other end-point of the line.

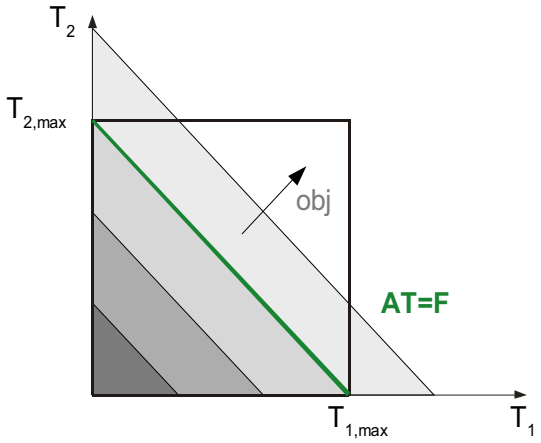

FIGURE 9. Example of an ill-conditioned linear problem

Successive calculations of almost the same force/torque commands can lead to a so called thrust jitter. The solution jumps between two alternatives, which have almost the same optimality. This behaviour is displayed in Figure 10.

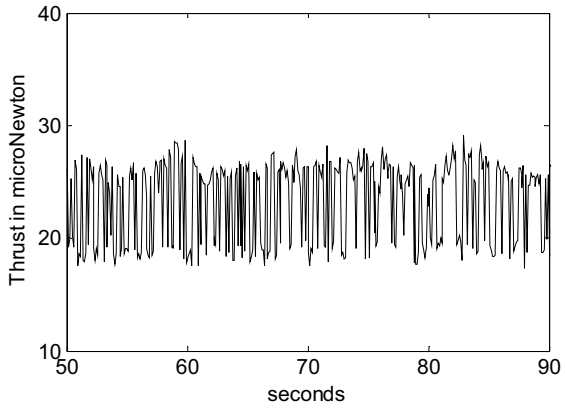

FIGURE 10. Jitter of a thruster, LISA science mode

Such instabilities in the thrusters commands can be dangerous for the engines, as their thrust level is constantly changing.

A solution for this problem is the introduction of additional perturbations to the objective function.

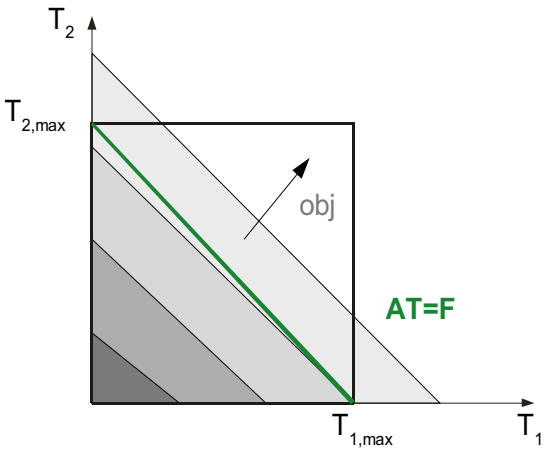

FIGURE 11. Disturbed objective function of previous problem

In Figure 11 this principle is shown graphically. The objective function is displayed as bars of different gray values. The linear equality condition *AT=F* is no longer perpendicular to the gradient of the objective function, and thus the upper left end of the line is identified as the optimal solution.

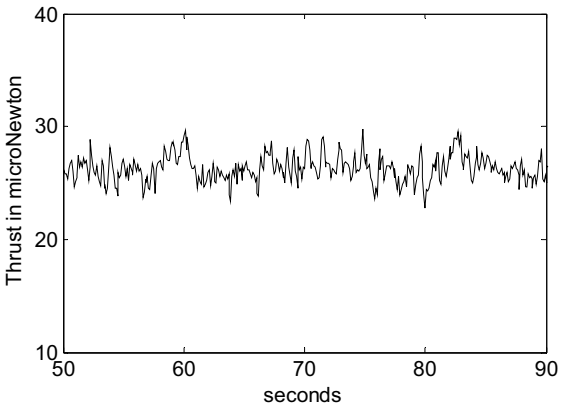

FIGURE 12. Jitter reduction for LISA thruster

The result of that small variation is displayed in Figure 12. In comparison to Figure 10, the sequence of commands for the same thruster is much more smooth.

$$e_{base} = 1.001$$

(52) $$e_i = e_{base}^i$$

(53) $$\min_T \left(\sum e_i T_i + \sigma c_s + \varphi c_d \right)$$

The mathematical expression of the disturbance is given in Eq. (52). The basic idea is, to modify the weightings for

every element in the objective function, that is connected to a thruster (see Eq. (53)). The perturbation vector *e* is an exponential function. Tests with the LISA thruster configuration and a synthetic sample for the science mode force/torque profile have shown, that a basic disturbance of 1.001 is sufficient to suppress any thrust jitter. Even for the highest thruster number 12, the variation of the objective function is below 2 percent. The basic equality condition of Eq (1) stays unaffected.

Although, the problem of thrust jitter and numerically ill-conditioned linear problems exists, it does show up only at a combination of different factors. The thruster configuration matrix *A* and the force/torque command vector *F* are the two main sources. It is not yet fully described, in which way these two elements interact with each other, to cause an ill-conditioned problem.

Future analysis of the complex six-dimensional problem may help to understand this relation. A result of such an analysis can be the automatic calculation of an appropriate and constant perturbation vector *e*.

ACKNOWLEDGEMENT

This work was supported by the DLR (German Aerospace Center) in the frame of a research project for optimal thruster control of micro-propulsion engines on science spacecraft (2004-2007). This project was a co-operative work between the ZARM (Center of Applied Space Technology and Microgravity) in Bremen and EADS Astrium GmbH in Friedrichshafen.

REFERENCES

[1] A. Genovese, M. Tajmar, N. Buldrini, M. Scheerer and W. Steiger, *"Indium FEEP Multiemitter Development and Test Results"*, 40th AIAA/ASME/SAE/ASEE Joint Propulsion Conference and Exhibit, July 2004, AIAA-2004-3620

[2] Jeng-Heng Vhen, *"Helium Thruster Propulsion System for Precise Attitude Control and Drag Compensation of the Gravity Probe-B Satellite"*, PhD thesis, Department of Aeronautics and Astronautics, Stanford University, December 1983

[3] HaiPing Jin, *"Translation and Attitude Control for the QUICK-STEP Satellite"*, PhD thesis, Department of Aeronautics and Astronautics, Stanford University, December 1996

[4] Denis Fertin and Wu Shufan, *"Analysis of design of Lisa Pathfinder drag-free controllers"*, Memo TEC-ECN/DF/S2-21/10/2004, ESA, December 2004

[5] Alexander Schleicher, *"Thruster actuation algorithm design and analysis"*, Technical report S2-ZAR-TN-2001, ZARM, July 2004