Real-time Capable Nonlinear Model Predictive Controller Design for the Upper Stage of a Launch Vehicle

By Yunus Emre ARSLANTAS¹⁾ and Thimo OEHLISCHLÄGEL¹⁾

¹⁾ The Institute of Space Systems, German Aerospace Center (DLR), Bremen, Germany

(Received August 8th, 2015)

In this paper, a real-time capable nonlinear model predictive controller is implemented for the attitude control of an upper stage launch vehicle with liquid propellant. A mass spring model is used as an analogy to simulate the disturbance generated by the sloshing propellant. For the implementation of the nonlinear model predictive controller, an optimal control problem is defined with finite time horizon. The objective function is minimized while satisfying constraints on the control inputs. The resulting optimal control problem is transcribed using single shooting method to parametrize the control inputs using uniform discretization points. The continuous control inputs are obtained by linear interpolation. A dedicated discretization algorithm in FORTRAN is coupled with a solver which used quasi-Newton algorithm to generate solutions fast. Approximation of the Hessian matrix is used to reduce computational requirements. Furthermore, the algorithm can perform parallel computation of the derivatives of the objective function with respect to optimization variables. This results in a real-time capability of generating solutions in the order of hundred milliseconds for each iteration. The algorithm is applied for attitude maneuver and disturbance rejection for the upper stage of a launch vehicle.

Key Words: Upper Stage, Quasi-Newton Method

Nomenclature

x	: State vector
u	: Input vector
t	: Time
X	: State space
U	: Input space
I	: Time interval
J	: Objective function
$\mathcal{U}$	: Admissible input space
Ω	: Terminal region constraint
D	: State cost
E	: Terminal cost
N	: Number of discretization points
B	: Hessian matrix
L^∞	: Lebesgue space
Q	: State performance weight
R	: Input performance weight
Δx	: Perturbation step
α	: Continuous function
∇f	: Gradient of a function
O	: Order
m_s	: Mass of rigid body
m_p	: Mass of propellant
L	: Moment arm
l	: Length of pendulum
J	: Moment of inertia
F	: External force
M	: Moment
z	: Optimization variable
θ	: Pitch angle of rigid body

ψ	: Angle of fluid surface level
w	: Penalty coefficient for terminal cost
$\tilde{f}$	: State vector field
$\tilde{g}_M$	: Moment input map
$\tilde{g}_F$	: Force input map
η	: Denominator in state vector field
BFGS	: Broyden–Fletcher–Goldfarb–Shanno
LV	: Launch vehicle
NLP	: Nonlinear programming
NMPC	: Nonlinear model predictive controller
OCP	: Optimal control problem

Subscripts

0	: Initial
f	: Final
p	: Prediction
c	: Control
i	: Discrete form
l	: Lower bound
u	: Upper bound

Superscripts

m	: Dimension of the input space
n	: Dimension of the state space
∞	: Infinity

1. Introduction

Within the last decade, space science and technology have become an essential part of daily activities. As a

result, there is an increase in demand for launching various payloads for different purposes. One of the challenges in inserting a payload into a desired orbit is the precise attitude control of the upper stage of the LV. Most LVs use liquid propellant for the last stage due to various advantages of throttable engines. Major drawback of these liquid propellant engines in terms of attitude control is the disturbance generated during the maneuver of the vehicle cause by the sloshing effect of the propellant. This paper recognized the sloshing problem and proposes an NMPC for attitude control of an upper stage LV.

Among the available controller design schemes for systems with nonlinear dynamics, NMPC has been implemented in real world problems. For instance, it is used in chemical process control and various aerospace applications.¹⁾²⁾ NMPC has several advantages like its straightforward implementation, capable of handling constraints on inputs as well as states, and its ability to stabilize the system despite model mismatch or external disturbances. One drawback of NMPC is that there is no guarantee to find a solution for the associated optimization problem. The main disadvantage of this method is the demanding necessity for computational resources while for each iteration a related OCP has to be discretized and solved. Therefore, real-time application of NMPC is coupled to the advancements in computing technology. Consequently, NMPC related studies are gaining momentum in line with these advancements. Review of NMPC methods,³⁾ studies about computational aspects of NMPC,⁴⁾ as well as robustness and stability properties,⁵⁾⁶⁾ are available in the literature.

There are various controller implementations for the upper stage LV. Most of these applications are based on linear control methods.^{7,8)} There are also adaptive control based methods,⁹⁾ and controllers with nonlinear dynamics.¹⁰⁾ The proposed method uses nonlinear state equations which avoids errors due to linearization. Moreover, the nonlinear dynamical system is stabilized without violating equality or inequality constraints. Therefore the problem of actuator saturation is eliminated.

In this paper, a real-time capable NMPC for the upper stage of a LV is developed using existing algorithms and computational tools. A NMPC is implemented for upper stage LV control problem in FORTRAN for decreasing computational demand. A dedicated script, which includes discretization of associated OCP, numerical integration and differentiation blocks and main NMPC algorithm is coupled with an existing numerical solver. In order to further improve the computation time, the derivative of the objective function is evaluated in parallel with shared-memory architecture using OpenMP.

The structure of the paper is as follows. In Section 2, the NMPC is formulated considering the objective function, equality and inequality constraints for states and control inputs. The methodology for discretizing the infinite dimensional OCP into an NLP is introduced. Details of the solution for the associated NLP using BFGS method are also provided in this section. Section 2 also includes the parallel computation of derivatives of the objective

function with respect to discrete control inputs using finite difference method. The dynamical model of the LV is presented in Section 3. Section 4 includes the numerical results for the simulations of disturbance rejection and attitude control of a large angle maneuver scenarios. Finally, Section 5 concludes the paper with a short summary.

2. Methodology

The NMPC problem is formulated by solving a series of finite-horizon OCPs subject to system dynamics and constraints involving states and controls. Based on the full state information at time t_0 , the controller predicts the future dynamic behavior of the system over a prediction horizon H_p for a time interval $[t_0, t_0 + t_p]$. In other words, it propagates the given control inputs for the prediction horizon H_p . Then, it determines the optimal control inputs which minimize the objective function for the desired performance. The optimal control inputs, obtained by solving the optimization problem, are then applied over a given control horizon H_c within the time interval $[t_0, t_0 + t_c]$ where $t_c \leq t_p$. The principal model of NMPC, representations of H_p and H_c are given in Fig. 1.

Fig. 1. The principle model of NMPC.

2.1. Formulation of NMPC

Consider the nonlinear dynamical system with state space $X \subset \mathbb{R}^n$ and the input space $U \subset \mathbb{R}^m$

$$\dot{x}(t) = f(x(t), u(t)), \quad t \in I = [t_0 + t_f]$$

$$x(t_0) = x_0 \in X \quad (1)$$

The control input is contained in the space of admissible control signals; that is $u(t) \in \mathcal{U} = L^\infty([t_0 + t_f], U)$. For each point in time, the solution or state trajectory $x(\cdot) : \mathbb{R}_+ \rightarrow X$ of Eq. (1) is an element of the state space $x(\cdot) \in X$. The finite horizon open-loop OCP is formulated as

$$\min_{u \in \mathcal{U}} J(x_0, x(t_f), u(t); t_p)$$

$$\text{s.t. } \dot{x}(t) = f(x(t), u(t)), \quad x(t_0) = x_0 \quad (2)$$

$$x(t) \in X \subseteq \mathbb{R}^n, \quad \forall t \in [t_0, t_0 + t_p]$$

$$x(t_0 + t_p) \in \Omega$$

with inequality constraints

$$\begin{aligned} x_l &\leq x(t) \leq x_u \\ u_l &\leq u(t) \leq u_u \end{aligned} \quad (3)$$

where $\dot{x}(t) = f(x(t), u(t))$ is the equality constraint to be satisfied. The objective is to minimize the cost function of the following form

$$J(x_0, x(t_f), u(t); t_p) = \int_{t_0}^{t_0+t_p} (\Delta u^T R \Delta u + \Delta x^T Q \Delta x) dt + E(x_f) \quad (4)$$

where $\Delta u = u - u_{ref}$, $\Delta x = x_f - x_{ref}$. Similarly, x_{ref} and u_{ref} are the desired constant reference values. The cost functional J is defined in terms of the stage cost (the first term in Eq. (4)), which specifies the performance of the controller and terminal cost (the latter term in Eq. (4)).

A quadratic form for stage cost is used. The positive definite matrices Q and R are the weights for the deviations of states and control inputs from desired values. The state information $x(t_0)$ enters the system with initial states. In this paper, it is assumed that full state information is available. The implementation of NMPC could be summarized as follows:

Fig. 2. NMPC flow chart.

1. Obtain the measurements x_0 of the system.
2. Calculate the optimal inputs minimizing the cost function in Eq. (4) over prediction horizon H_p .
3. Implement the optimal inputs over the control horizon H_c and get new measurement x_0 .
4. Provide optimal inputs as initial guess for the next iteration.
5. Continue with (2) until the predetermined number of iterations is completed.

Figure 2 shows the flow of the algorithm. In this study, the stopping criterion is determined as the maximum

number of iterations. For each iteration, the states are propagated one second in the given time domain.

Another aspect is the stability and performance of the controller. The prediction horizon H_p plays an important role for the stability of the closed-loop system. On one side, if the H_p is short, the controller tries to steer the system to desired states with a more agile behaviour. If it is even shortened, the controller might not be able to stabilize the system around the given equilibrium point. On the other hand, if H_p is long enough, the size of the problem increases, demanding additional computational resources.

NMPC considers process nonlinearities and constraints explicitly compared to linear feedback controllers. This approach ensures that any constraints and performance considerations are directly taken into account during the solution of the related OCP.

The OCP defined in Eqs. (2) and (3) is infinite dimensional and discretized to obtain a corresponding Nonlinear Programming (NLP). In this paper single shooting method with uniform discretization points in time is used to approximate an associated NLP of the OCP.

2.2. Discretization of the OCP to associated NLP

In order to discretize the OCP defined by Eq. (2), control inputs are parametrized using single shooting method. The control input $u(t)$ is discretized using uniform discretization points in time t_i , resulting in discrete form $u(t_i)$ of control inputs. Values of the control inputs between the discretization points are approximated by interpolating the discrete control inputs $u(t_i)$ using linear interpolation.

Uniform grid points are obtained by discretizing the time interval into equal-spaced discretization points such that

$$\begin{aligned} t_i &\in [t_0 + t_f], & i &= 0, \dots, N \\ x(t_i) &= x_i, \quad x_i \in \mathbb{R}^n, & i &= 0, \dots, N \\ u(t_i) &= u_i, \quad u_i \in \mathbb{R}^m, & i &= 0, \dots, N \end{aligned} \quad (5)$$

holds. Similarly the discrete form of the objective function is

$$J = w(x_f - x_d)^2 + \sum_{i=0}^N \Delta u_i^T R \Delta u_i + \Delta x_i^T Q \Delta x_i, \quad t_i \in [t_0 + t_p], \quad i = 0, \dots, N \quad (6)$$

The trajectories of the states are obtained by integrating the control input by using 4th order Runge-Kutta numerical integration scheme. In order to find the values of the discrete control inputs at the discretization points, resulting NLP needs to be solved by a numerical solver.

2.3. Solution of the NLP

In this paper, the BFGS method is used to solve resulting NLP. The BFGS method is a quasi-Newton method for solving unconstrained nonlinear optimization problems. As in Newton's method, one uses a second order approximation to find the minimum of a function $\alpha(z)$. The Taylor series of $\alpha(z)$ around an iterate is:

$$\alpha(z_k + \Delta z) \approx \alpha(z_k) + \nabla \alpha(z_k)^T \Delta z + \frac{1}{2} \Delta z^T B \Delta z, \quad (7)$$

where $\nabla \alpha$ is the gradient and B an approximation to the Hessian matrix. The gradient of this approximation (with respect to Δz) is

$$\alpha(z_k + \Delta z) \approx \nabla \alpha(z_k) + B \Delta z \quad (8)$$

and setting this gradient to zero (which is the objective of the optimization) provides the Newton step:

$$\Delta z = -B^{-1} \nabla \alpha(z_k) \quad (9)$$

Different from Newton method, the BFGS method uses an approximation of the Hessian matrix of second derivatives instead of exact Hessian matrix.

Limited memory BFGS method (L-BFGS) is a modified version of BFGS method, where L-BFGS stores only a few vectors that represent the approximation implicitly. Memory requirement is linearly dependent on the number of optimization variables, which reduces the necessity of computational power.

The L-BFGS-B algorithm further extends L-BFGS to handle simple box constraints on variables. In other words, constraints of the form $u_l \leq u_i \leq u_u$ where u_l and u_u are constant lower and upper bounds of the respective optimization variable. In this paper, a FORTRAN implementation of the L-BFGS-B method is used.¹¹⁾

Fig. 3. Implementation of NMPC.

Figure 3 shows the block diagram of NMPC and implementation of the dynamic optimizer. The optimization is terminated either when the change in the objective function or infinity norm of the projected gradient becomes sufficiently small.

2.4. Numerical derivatives

In order to obtain the solution of NLP, derivatives of the objective function with respect to optimization variables are needed. These derivatives are obtained using finite difference method by using the Taylor series expansion of the objective function.

$$\alpha(z + \Delta z) = \alpha(z) + \Delta z \dot{\alpha}(z) + O(\Delta z^2). \quad (10)$$

Solving for $\dot{\alpha}(z)$ gives

$$\dot{\alpha}(z) = \frac{\alpha(z + \Delta z) - \alpha(z)}{\Delta z} + O(\Delta z). \quad (11)$$

The perturbation value $\Delta z \ll 1$ is problem dependent and chosen by tuning. Obtaining the derivatives of the objective function is the most time consuming phase during solution of the associated NLP.

In order to speed-up the computations, the derivatives are evaluated independently using shared memory parallelization with OpenMP.¹²⁾ In shared memory parallelization, a set of processors have access to the same common memory. A thread is an execution context, which all the information a CPU needs to execute a stream of instructions. When a thread executing a serial region encounters a parallel region, it creates a group of slave threads, and becomes the master thread of the team. In this way, a master thread generates a specified number of slave threads and the system divides a task among them. These threads then run at the same time, with the runtime environment allocating threads to different cores of the CPU. The division of the tasks is illustrated in Fig. 4, where each slave thread executes tasks independently.

In the proposed method, the master thread executes serial region of the algorithm. Only the computation of the numerical derivatives is defined as parallel region. Slave threads are created to evaluate the derivative of the objective function with respect to optimization variables in parallel. Each slave thread has access to the common shared memory and executes the instructions independent of each other decreasing the computation time substantially.

Fig. 4. Division of tasks for computation of derivative of the objective function in parallel.

3. The Nonlinear Dynamical Model of the Upper Stage

Liquid sloshing is the uncontrolled movement of the propellant inside a tank or container. During the maneuver of the upper stage of a LV, this movement acts as a disturbance to the system. There are various passive and active methods to mitigate the effects of the sloshing. For the active methods, the objective is to design feedback controllers so that the controlled LV achieves desired attitude while rejecting the disturbance caused by sloshing effect as shown in Fig. 5.

The nonlinear model of the upper stage LV is taken

from Reference 13). A mass-damper-spring system is used to model the nonlinear dynamics of the upper stage LV with the sloshing effect.

Fig. 5. Upper-stage LV with sloshing effect of propellant.

The nonlinear dynamical model consists of four states $x = (\theta, \dot{\theta}, \psi, \dot{\psi})^T$, where θ is the pitch angle of the upper stage LV and ψ is the relative angle of the fluid surface with respect to horizontal plane. The control vector includes two control inputs $u = (F, M)^T$, force and torque couple acting on the body as illustrated in Fig. 6. The control inputs are defined in box constraints, i.e.,

$$\begin{aligned} F(t) &\in (0, 400] \text{ N} \\ M(t) &\in (-400, 400] \text{ Nm} \end{aligned} \quad (12)$$

The thrust is defined in half-open interval to ensure that during the maneuver, the propellant stays in liquid state in microgravity environment. The state and control vectors satisfy the relation with $\dot{x}(t) = \tilde{f}(x) + [\tilde{g}_F(x) \ \tilde{g}_M(x)]u$ with

$$\tilde{f} = \begin{pmatrix} \dot{\theta} \\ \frac{m_f m_s \sin(2\psi - 2\theta) L^2 \dot{\theta}^2 + 2l m_f m_s \sin(\psi - \theta) L \dot{\psi}^2}{2\eta} \\ \dot{\psi} \\ \frac{2JL\dot{\theta}^2 m_f \sin(\psi - \theta) + 2JL\dot{\theta}^2 m_s \sin(\psi - \theta) + \frac{2\eta l}{2\eta l}}{2\eta l} \\ \frac{2L^3 \dot{\theta}^2 m_f m_s \sin(\psi - \theta) + L^2 \dot{\psi}^2 l m_f m_s \sin(2\psi - 2\theta)}{2\eta l} \end{pmatrix} \quad (13)$$

$$\tilde{g}_F = \begin{pmatrix} 0 \\ \frac{L m_f \sin(2\psi - 2\theta)}{2\eta} \\ 0 \\ -\frac{\sin(\psi - \theta)(J m_f + m_s(m_f L^2 + J))}{l m_s \eta} \end{pmatrix} \quad (14)$$

$$\tilde{g}_M = \begin{pmatrix} 0 \\ \frac{m_f + m_s}{\eta} \\ 0 \\ \frac{2L \cos(\psi - \theta)(m_f + m_s)}{l(2J m_f + 2J m_s + L^2 m_f m_s - L^2 m_f m_s \cos(2\psi - 2\theta))} \end{pmatrix} \quad (15)$$

where $\eta = m_f m_s L^2 \sin(\psi - \theta)^2 + J m_f + J m_s$.

The above equations represent the affine nonlinear dynamics of the system in state space. The parameters associated with the nonlinear dynamics are given in Table 1.

Fig. 6. Analogy of mass-spring system for modeling the sloshing effect of propellant.

Table 1. Nominal values for parameters.

m_s	m_f	L	l	J
10170 kg	3470 kg	1 m	3 m	105000 kgm ²

4. Numerical Results

The closed loop system is simulated for two different scenarios. In the first scenario, the initial angle of the fluid acts as disturbance to the system and the controller stabilizes the plant with the induced disturbance resulting from the sloshing effect of the fluid. In the second scenario, the upper stage LV performs a large angle maneuver while keeping the sloshing effect to the minimum. The computations are performed using an Intel i7 Quad Core CPU operating at 2.80 GHz and 8 GB of RAM.

The sloshing amount is defined as the quantitative measure for sloshing and defines the relative difference of the pitch angle and the level of the fluid surface. The objective is to obtain the desired states with optimal control inputs while reducing the sloshing amount. During the simulations, the prediction horizon H_p is selected as 60s and control horizon H_c as 1s. For each iteration of NMPC simulates 1s of the closed loop system and the simulation stops after the specified number of iterations are completed.

Figure 7 shows the trajectories of the states and control inputs for the simulation of first scenario with the presence of initial disturbance with $x_0 = (60^\circ, 0, 0, 0)$. The NMPC is able to attenuate the sloshing amount, while steering the system to the desired values with $x_f = (0, 0, 0, 0)$.

Fig. 7. State trajectories of upper-stage LV with initial disturbance.

Similarly, Fig. 8 shows the trajectories of the states and control inputs for the second scenario with a large angle maneuver. Initially, both the surface level of fluid and the pitch angle of the upper stage LV are equal with $x_0 = (0, 0, 0, 0)$. The NMPC tries to stabilize the system while the upper stage LV performs a large angle maneuver to $x_f = (90^\circ, 0, 90^\circ, 0)$. Previous comments also hold for attitude control of the upper stage LV in terms of obeying constraints and behaviour of the control inputs. In this scenario, the magnitude of the sloshing amount is bounded at around 20° .

Table 2. Comparison of computation time for disturbance rejection scenario per iteration.

	1 st Scenario	2 nd Scenario
Sequential Computing	414	583
Parallel Computing	132	188

Table 2 shows the computation time per iteration of NMPC for both cases. It is obvious that if the derivatives of the objective function with respect to discrete control inputs are computed in parallel, the computation time per iteration drops significantly. In this case one single CPU is used and the workload is distributed to 4 cores. The speed-up is 3.13 for the first scenario with the Quad Core CPU. In the second case, the speed-up is likewise with the previous case with 3.10. The main reason for why the super linear speed-up is not achieved, is due to the fact that only computation of derivatives was performed in parallel. In the second scenario, the computation time for each iteration is more compared to the first scenario with initial disturbance on the fluid level. It can be deduced that, for the second scenario computing the optimal control inputs is more demanding in terms of computational resources.

Fig. 8. State trajectories of upper-stage LV with for large angle maneuver scenario.

Robustness of the NMPC is shown with Monte Carlo simulations. The pitch angle is varied with $0 \leq \theta_0 \leq 20^\circ$ and rate of fluid level is varied with $-2^\circ/s \leq \dot{\psi}_0 \leq 2^\circ/s$ using uniform distribution. 250 simulations are carried out with Monte Carlo analysis for 400 seconds for each specific case.

As shown in Fig. 9, all the test cases converge to the desired value while obeying the constraints on the control inputs. Table 3 tabulates the numerical values for the Monte Carlo analysis. In all cases the states converge to the desired final values. During the simulations the constraints associated with control inputs were not violated in any case. Moreover, except a few cases, the control inputs approach to zero. As expected, as the current states converge to the desired values, less control authority is needed.

Fig. 9. Monte Carlo Analysis for the NMPC.

Table 3. Monte Carlo analysis results.

	Mean	Standard Deviation
θ (°)	0.0164	0.1447
$\dot{\theta}$ (°/s)	-0.0002	0.0017
ψ (°)	0.0029	0.1239
$\dot{\psi}$ (°/s)	0.0001	0.0032
F (N)	6.5562	8.4413
M (Nm)	-0.0036	0.1832

5. Conclusion

In this paper an algorithm for real-time capable NMPC is introduced. The NMPC is established with a given nonlinear dynamical model of the upper stage LV, equality and inequality constraints leading to an OCP. The infinite dimensional OCP is discretized with single shooting method and control inputs are parametrized with equidistant discretization points in time. All values between the discretization points are obtained by using linear interpolation.

During the two simulated scenarios, the NMPC stabilizes the upper stage LV with different initial and terminal conditions. The derivatives of the objective function with respect to optimization variables are evaluated in parallel to decrease computation time per each NMPC iteration. Computation time per each NMPC cycle is shown for these two scenarios. The solutions are obtained for discrete control inputs and state trajectories are evaluated by numerical integration of these control inputs. It was seen that for all the simulation results, the control inputs are bounded with the given upper and lower limits.

Monte Carlo analysis shows the robustness of the solution, ensuring stability with different initial conditions. Further developments include the use of nonuniform grid points in time to parametrize the control inputs. The selection of prediction horizon H_p on stability and computation time could also be further analyzed.

Acknowledgments

This research is supported by DLR (German Aerospace Center) and DAAD (German Academic Exchange Service) Research Fellowship Program.

References

- 1) Qin, S. J. and Badgwell, T. A.: A Survey of Industrial Model Predictive Control Technology, *Control Engineering Practice*, **11** (2003), pp. 733-764.
- 2) Qin, S. J. and Badgwell, T. A.: An Overview of Nonlinear Model Predictive Control Applications, *Nonlinear Model Predictive Control*, **26** (2000), pp. 369-392.
- 3) Allgöwer, F., Badgwell, T. A., Qin, S. J., Rawlings, J. and Wright, S.: Nonlinear Predictive Control and Moving Horizon Estimation - An Introductory Overview, *Advances in Control*, Springer, London, 1999, pp. 391-449.
- 4) Grüne, L. and Pannek, J.: *Nonlinear Model Predictive Controller (Theory and Algorithms)*, Springer-Verlag, 2011.
- 5) Bornard, G. and Alamir, M.: Stability of a Truncated Infinite Constrained Receding Horizon Scheme: The General Discrete Nonlinear Case, *Automatica*, **31** (1995), pp. 1353-1356.
- 6) Blauwkamp, R. and Basar, T.: A Receding-Horizon Approach to Robust Output Feedback Control for Nonlinear Systems, 38th IEEE Conference on Decision and Control, San Diego, USA, 1999.
- 7) Wie, B.: *Space Vehicle Dynamics and Control*, AIAA Education Series, 1998.
- 8) Bryson, A. E.: *Control of Spacecraft and Aircraft*, Princeton University Press, 1994.
- 9) Adler, J. M., Lee, M. S. and Saugen, J. D.: Adaptive Control of Propellant SLOSH for Launch Vehicles, *Proceedings of the Sensors and Sensor Integration*, Apr. 1991.
- 10) Reyhanoglu, M. and Hervas, J. R.: Nonlinear Control of Space Vehicles with Multi-Mass Fuel SLOSH Dynamics, Proceedings of 5th International Conference on Recent Advances in Space Technologies, Jun. 2011.
- 11) Zhu, C., Byrd, R. H., Lu, P. and Nocedal, J.: Algorithm 778: L-BFGS-B: Fortran Subroutines for Large-Scale Bound-Constrained Optimization, *ACM Trans. Math. Softw.*, **23** (1997), pp. 550-560.
- 12) Hermanns, M.: Parallel Programming in Fortran 95 Using OpenMP, available at http://www.openmp.org/presentations/miguel/F95_OpenMPv1_v2.pdf (cited 30 July 2015).
- 13) Kheirikhah, A.: Passivitätsbasierte Regelung einer Wiedorzündbaren Kryogenen Oberstufe zur Unterdrückung des Treibstoffschwappens Während Ausgesuchter Manöver, Master's Thesis, Leibniz Universität Hannover, 2013 (in German).