Slovak University of Technology in Bratislava
Institute of Information Engineering, Automation, and Mathematics

PROCEEDINGS

of the 18th International Conference on Process Control

Hotel Titris, Tatranská Lomnica, Slovakia, June 14 – 17, 2011

ISBN 978-80-227-3517-9

http://www.kirp.chtf.stuba.sk/pc11

Editors: M. Fikar and M. Kvasnica

Noga, R., Ohtsuka, T.: NMPC for Stiff, Distributed Parameter System: Semi-Automatic Code Generation and Optimality Condition Evaluation, Editors: Fikar, M., Kvasnica, M., In *Proceedings of the 18th International Conference on Process Control*, Tatranská Lomnica, Slovakia, 415–418, 2011.

Full paper online: http://www.kirp.chtf.stuba.sk/pc11/data/abstracts/026.html

NMPC for stiff, distributed parameter system: Semi-Automatic Code Generation and optimality condition evaluation

R. Noga [*,**] **T. Ohtsuka** [***]

[*] *European Organization for Nuclear Research, CERN CH-1211, Genève 23, Switzerland*
[**] *University of Valladolid, c/ Real de Burgos s/n., 47011 Valladolid, Spain*
[***] *Osaka University, 1-3 Machikaneyama, Toyonaka, Osaka 560-8531, Japan (e-mail: mail@rafal-noga.com).*

Abstract: AutoGenU is a Mathematica program to automatically generate simulation programs for Nonlinear Model Predictive Control (NMPC). It analytically evaluates the Jacobians necessary to calculate the optimality condition in the NMPC realized using Continuation/Generalized Minimum Residual (C/GMRES) method. However, in the case of the LHC Superfluid Helium Cryogenic System, which is distributed parameter system, these Jacobians, expressed directly in terms of inputs, states and co-states become complex expressions due to cascading relations between internal variables of the circuit's model. A semi-automatic code generation procedure based on AutoGenU is presented, where intermediate variables are introduced and the chain rule is applied to evaluate the Jacobians, thereby avoiding complex expressions. In addition, the ODE set describing the system state dynamics is stiff, thus the dynamics time integration step is small. The intermediate variables are available at each step and are used to evaluate the optimality condition more precisely at low additional computing cost. The observed computational cost of the semi-automatically generated code is slightly lower than that of automatically generated and the controller performance is similar in both cases. However, the generation of semi-automatic code requires significantly less memory, and is much faster, widening the applicability of code generation for complex systems.

Keywords: Nonlinear Model Predictive Control, Nonlinear Receding Horizon Control, Automatic Code Generation System, Distributed Parameters System, Stiff system

1. INTRODUCTION

AutoGenU is a Mathematica program to automatically generate simulation programs for Nonlinear Model Predictive Control (NMPC) also known as Receding Horizon Control (RHC). Once the state equation, the performance index and some other simulation conditions are specified by a user as an input file in Mathematica® Format, then AutoGenU.nb loads the input file, executes such necessary operations as partial differentiation, and generates a C source file. The generated source file is ready for compilation and execution. The simulation program employs a fast optimization algorithm, Continuation/Generalized Minimum Residual (C/GMRES) (Ohtsuka (2000)). AutoGenU has been applied to generate the NMPC for the Superfluid Helium Cryogenic Circuit (SHCC) at the Large Hadron Collider (LHC) [1]. More precisely, a simulation independent implementation of the NMPC based on the C/GMRES optimization (Ohtsuka (2004)) has been separated from the simulation program. Then, this C code has been used in Matlab® simulations, accessed via MEX functions, (Noga et al. (2010)) and also it has been integrated into the PVSS II® SCADA of the LHC cryogenic system as a prototype implementation of NMPC for the SHCC.

A 106.9 m long Standard Cell of the SHCC is composed of eight, main superconducting magnets of the LHC, submerged in a bath of superfluid helium (Brüning et al. (2004)). The magnets are cooled via an over 100 m long Bayonet Heat Exchanger integrated into the magnets (Lebrun et al. (1997)). Our system corresponds to a Sub-Sector of the SHCC that is composed of two Standard Cells that share common helium bath (Gubello et al. (2006)). The system state $x(l,t)$ is the magnet temperatures as a function of time t and is spatially distributed over the Sub-Sector length $0 \leq l \leq 2 \times 106.9$ m. Its dynamics has been modeled as a function of a distributed value of cooling power of two heat exchangers that is a function of helium saturation temperature $a(l,t)$ and mass flow rate $b(l,t)$ in each heat exchanger (Noga (2007), Noga et al. (2010)). After spatial discretization of x, a and b using a Finite Volume approach with $N = 10$ intervals, the dynamics of discretized state

[1] The LHC is the newest particle accelerator and collider at the European Organization for Nuclear Research (CERN)

$$\begin{aligned} \mathrm{d}x_1/\mathrm{d}t &= f_1(x_1, x_2, a_1, b_1) \\ \mathrm{d}x_i/\mathrm{d}t &= f_i(x_{i-1}, x_i, x_{i+1}, a_i, b_i) \quad (i = 2, ..., N-1) \\ \mathrm{d}x_N/\mathrm{d}t &= f_N(x_{N-1}, x_N, a_N, b_N), \end{aligned} \tag{1}$$

with saturation temperature

$$\begin{aligned} &a_1 = a_{I,1} \\ &a_{i+1}(a_i, b_i) \quad (i = 1, ..., N/2-1) \\ &a_{O,1}(a_{N/2}, b_{N/2}) \\ &a_{N/2+1} = a_{I,2} \\ &a_{i+1}(a_i, b_i) \quad (i = N/2+1, ..., N-1) \\ &a_{O,2}(a_N, b_N) \end{aligned} \tag{2}$$

and He II mass flow rate

$$\begin{aligned} &b_1 = b_{I,1} \\ &b_{i+1}(a_i, b_i, x_i) \quad (i = 1, ..., N/2-1) \\ &b_{O,1}(a_{N/2}, b_{N/2}, x_{N/2}) \\ &b_{N/2+1} = b_{I,2} \\ &b_{i+1}(a_i, b_i, x_i) \quad (i = N/2+1, ..., N-1) \\ &b_{O,2}(a_N, b_N, x_N). \end{aligned} \tag{3}$$

Please note the presence of a Two-Point Boundary Value problem, since the boundaries are the He II mass flow rates at heat exchanger inlets that are the model manipulated variables

$$b_{I,k} = u_k \ (k = 1, 2) \tag{4}$$

and the saturation temperature at the outlets $a_{O,k}$. The saturation temperatures are equal in a Sub-Sector $a_{O,k} = a_{SS}$. Due to the spatial discretization scheme chosen, the saturation temperature at the inlets $a_{I,k}$ is solved to satisfy the boundary conditions at the outlets $a_{O,k}$ using the Newton method with n-th iteration

$$(a_{I,k})_{n+1} = (a_{I,k})_n - (\mathrm{d}a_{O,k}/\mathrm{d}a_{I,k})_n^{-1} \, [(a_{O,k})_n - a_{SS}]. \tag{5}$$

In order to evaluate the necessary optimality condition in the NMPC (Ohtsuka (2004)), AutoGenU analytically calculates the Jacobians H_u and H_x of the Hamiltonian

$$H = L(x, u) + \lambda^T \ f(x, u, t) + \mu^T \ C(x, u, t) \tag{6}$$

with respect to vectors of system inputs $u(t)$ and states $x(t)$. Here, t is time, L is a performance index that appears in the cost functional to be minimized during the optimization,

$$\dot{x} = f(x, u), \ C(x, u) = 0 \tag{7}$$

represent the system state dynamics and constraints respectively and $\lambda(t)$ and $\mu(t)$ are the Lagrange multipliers. In case of the SHCC , the Jacobians H_u and H_x expressed directly in terms of inputs and states become very complex and the automatic code generation fails due to excessive operational memory needed. However, the automatic generation of each component of the Jacobians separately has been successful. One method to significantly reduce the complexity of the H_u and H_x expressions is to introduce intermediate variables and use the chain rule to evaluate the Jacobians. A number of intermediate variables corresponding to internal model variables has been chosen to exploit the model structure. Since the choice of the intermediate variables and the implementation of the chain rule are done by hand, the resulting code generation procedure is semi-automatic.

The set of ODEs describing the SHCC state dynamics is stiff. In the C/GMRES version for systems with stiff dynamics, the residuum of the optimality condition is calculated using integrals of the Jacobian H_u over the prediction horizon grid (Noga et al. (2010)). The grid corresponds to intervals with constant control signal. The automatically generated H_u is expressed directly in terms of system inputs and states and thus has high computational cost, thus the integrals are calculated using simple quadratures such as one-point rectangular or two-points trapezoidal, where H_u is evaluated exclusively at the horizon grid. However, once the intermediate variables are introduced and then available at each state integration step, which in case of stiff system is much shorter than the grid interval, the H_u may be evaluated at each step at low additional computational cost, resulting in a more precise calculation of the optimality condition.

This paper presents the semi-automatic procedure of NMPC code generation for the SHCC based on AutoGenU. This section introduced AutoGenU and the motivation for the use of intermediate variables in case of the SHCC. Next, the choice of intermediate variables and the implementation of the chain rule are described in detail. Then the evaluation of the optimality condition using the intermediate variables is presented. Finally the performance of the semi-automatically generated NMPC code is compared against that generated automatically. The low memory required by the semi-automatic generation process is highlighted.

2. SEMI-AUTOMATIC CODE GENERATION

During automatic code generation using symbolic mathematics, the cascading relations between i-th and $(i-1)$-th variable a and b in Eqs. (2) and (3) enable propagation of complex expressions. This is especially visible in case of the Hamiltonian, Eq. (6), that involves the system dynamics f, Eq. (1), and, trough the index L, helium mass flow rates at the heat exchanger outlets calculated from Eq. (3). Its Jacobians H_u and H_x expressed directly in terms of inputs and states become very complex and the automatic code generation fails due to excessive operational memory needed. However, automatic generation of C code for each component of the Jacobians separately requires less memory and has been successful.

A method to avoid the propagation that significantly reduces the complexity of the expressions for H_u and H_x is to use a_i and b_i as intermediate variables. The components of H_u and H_x, which are the Jacobians f_x, f_u, C_x, C_u, L_x and L_u, are calculated using the chain rule

$$\frac{\mathrm{d}f_i}{\mathrm{d}x_j} = \frac{\partial f_i}{\partial x_j} + \frac{\partial f_i}{\partial a_i}\frac{\mathrm{d}a_i}{\mathrm{d}x_j} + \frac{\partial f_i}{\partial b_i}\frac{\mathrm{d}b_i}{\mathrm{d}x_j} \tag{8}$$

$$\frac{\mathrm{d}f_i}{\mathrm{d}u_k} = \frac{\partial f_i}{\partial a_i}\frac{\mathrm{d}a_i}{\mathrm{d}u_k} + \frac{\partial f_i}{\partial b_i}\frac{\mathrm{d}b_i}{\mathrm{d}u_k}, \tag{9}$$

recalling that $b_1 = u_1$ and $b_{N/2+1} = u_2$, see Eq. (4). The Jacobians of intermediate variables with respect to the states and inputs are calculated as follows. At the discretization points along the heat exchanger $i = 1, ..., N/2-1, N/2+1, ..., N-1$

$$\begin{bmatrix} \frac{\mathrm{d}a_{i+1}}{\mathrm{d}x_j} \\ \frac{\mathrm{d}b_{i+1}}{\mathrm{d}x_j} \end{bmatrix} = \begin{bmatrix} \frac{\partial a_{i+1}}{\partial a_i} & \frac{\partial a_{i+1}}{\partial b_i} \\ \frac{\partial b_{i+1}}{\partial a_i} & \frac{\partial b_{i+1}}{\partial b_i} \end{bmatrix} \begin{bmatrix} \frac{\mathrm{d}a_i}{\mathrm{d}x_j} \\ \frac{\mathrm{d}b_i}{\mathrm{d}x_j} \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{\partial b_{i+1}}{\partial x_i} \ \delta_{i,j} \end{bmatrix} \tag{10}$$

$$\begin{bmatrix} \frac{\mathrm{d}a_{i+1}}{\mathrm{d}u_k} \\ \frac{\mathrm{d}b_{i+1}}{\mathrm{d}u_k} \end{bmatrix} = \begin{bmatrix} \frac{\partial a_{i+1}}{\partial a_i} & \frac{\partial a_{i+1}}{\partial b_i} \\ \frac{\partial b_{i+1}}{\partial a_i} & \frac{\partial b_{i+1}}{\partial b_i} \end{bmatrix} \begin{bmatrix} \frac{\mathrm{d}a_i}{\mathrm{d}u_k} \\ \frac{\mathrm{d}b_i}{\mathrm{d}u_k} \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{\partial b_{i+1}}{\partial u_k} \end{bmatrix} \quad (11)$$

with Kronecker delta $\delta_{i,j}$. At the outlets of the k-th heat exchanger ($i = N/2, N$)

$$\begin{bmatrix} \frac{\mathrm{d}a_{O,k}}{\mathrm{d}x_j} \\ \frac{\mathrm{d}b_{O,k}}{\mathrm{d}x_j} \end{bmatrix} = \begin{bmatrix} \frac{\partial a_{O,k}}{\partial a_i} & \frac{\partial a_{O,k}}{\partial b_i} \\ \frac{\partial b_{O,k}}{\partial a_i} & \frac{\partial b_{O,k}}{\partial b_i} \end{bmatrix} \begin{bmatrix} \frac{\mathrm{d}a_i}{\mathrm{d}x_j} \\ \frac{\mathrm{d}b_i}{\mathrm{d}x_j} \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{\partial b_{O,k}}{\partial x_i} \end{bmatrix} \quad (12)$$

$$\begin{bmatrix} \frac{\mathrm{d}a_{O,k}}{\mathrm{d}u_k} \\ \frac{\mathrm{d}b_{O,k}}{\mathrm{d}u_k} \end{bmatrix} = \begin{bmatrix} \frac{\partial a_{O,k}}{\partial a_i} & \frac{\partial a_{O,k}}{\partial b_i} \\ \frac{\partial b_{O,k}}{\partial a_i} & \frac{\partial b_{O,k}}{\partial b_i} \end{bmatrix} \begin{bmatrix} \frac{\mathrm{d}a_i}{\mathrm{d}u_k} \\ \frac{\mathrm{d}b_i}{\mathrm{d}u_k} \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{\partial b_{O,k}}{\partial u_k} \end{bmatrix} \quad (13)$$

At the inlets of k-th heat exchanger, the manipulated inputs are the boundary conditions, see Eqs. (3) and (4), thus

$$\begin{aligned} \mathrm{d}b_i/\mathrm{d}x_j &= 0 \ (i = 1, N/2+1) \\ \mathrm{d}b_i/\mathrm{d}u_k &= 1 \ (i = 1, k = 1 \text{ and } i = N/2+1, k = 2) \\ \mathrm{d}b_i/\mathrm{d}u_k &= 0 \ (i = 1, k = 2 \text{ and } i = N/2+1, k = 1), \end{aligned} \quad (14)$$

however the saturation pressure in Eq. (2) is fixed at the outlet, thus

$$\begin{aligned} \mathrm{d}a_{O,k}/\mathrm{d}x_j &= 0 \\ \mathrm{d}a_{O,k}/\mathrm{d}u_k &= 0 \end{aligned} \quad (15)$$

and the partial derivatives at the inlet is calculated as

$$\begin{aligned} \mathrm{d}a_i/\mathrm{d}x_j &= -(\partial a_{O,k}/\partial a_i)^{-1}\ \partial a_{O,k}/\partial x_j \\ \mathrm{d}a_i/\mathrm{d}u_k &= -(\partial a_{O,k}/\partial a_i)^{-1}\ \partial a_{O,k}/\partial u_k. \end{aligned} \quad (16)$$

Here the Jacobians $\partial a_{O,k}/\partial x_j$ and $\partial a_{O,k}/\partial u_k$ are found as $\mathrm{d}a_{O,k}/\mathrm{d}x_j$ and $\mathrm{d}a_{O,k}/\mathrm{d}u_k$ using $\mathrm{d}a_1/\mathrm{d}x_j = 0$ and $\mathrm{d}a_{N/2+1}/\mathrm{d}u_k = 0$, see Eqs. (10) and (11). The partial derivatives $\partial a_{O,k}/\partial a_i$ are calculated in an iterative manner similar to Eqs. (10)–(13).

All other partial derivatives in Eqs. (8)–(13), are generated automatically as in AutoGenU. However, as demonstrated, the choice of the intermediate variables and the implementation of the chain rule are done by hand, thus the resulting code generation procedure is semi-automatic. Regarding other components of H_u and H_x, the cost function L is a sum of terms among which some are independent of the intermediate variables and the corresponding parts of the gradients L_x, L_u can be generated automatically. However, the helium mass flow rate at the heat exchanger outlets is minimized, thus enters the performance index and corresponding parts of the gradients have been generated similarly to and using some sub-expressions of the f_x, f_u. In the system, the Jacobians of the constraints, C_x and C_u, do not involve the intermediate variables and can be generated automatically.

The semi-automatic code generation procedure is much faster and requires much less operational memory than the automatic used in AutoGenU. The observed generation time was seconds vs. 30 min. and the maximum memory used to store all the data for the Mathematica session was 7.9 MB vs. 1034 MB. This makes it possible to apply this type of the semi-automatic code generation process to more complicated systems.

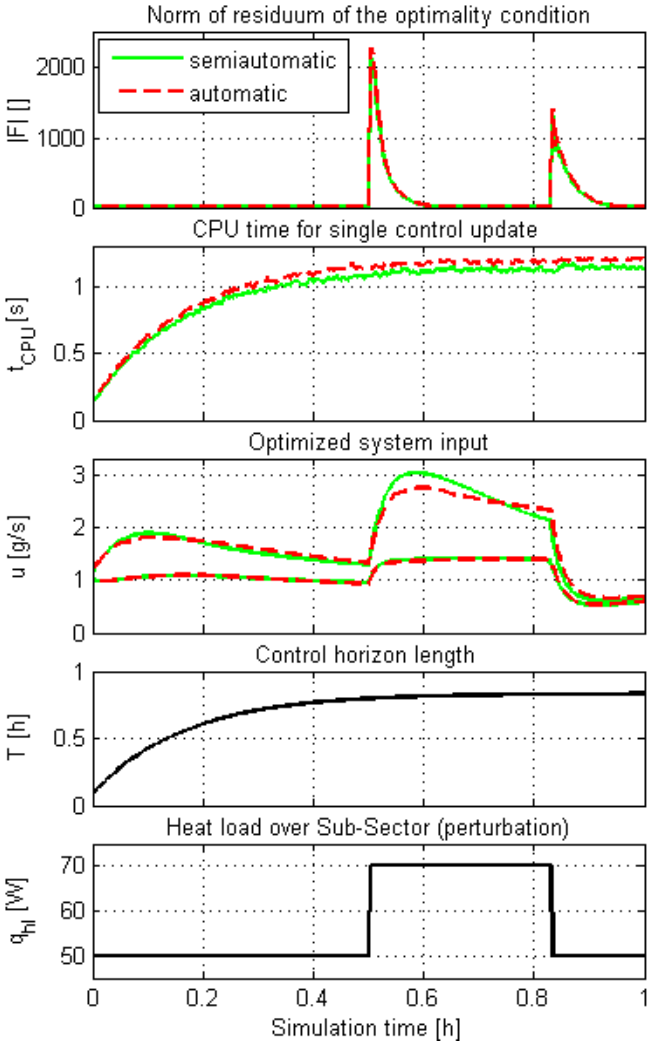

Fig. 1. Comparison of the performance of the automatically and semi-automatically generated controllers.

3. C/GMRES OPTIMALITY CONDITION

In NMPC an open loop optimal control problem is solved over the future prediction time t' horizon taken from the current time t to $t+T$ (Ohtsuka (2004)):

$$\text{minimize } \bar{J} = \phi + \int_t^{t+T} L + \lambda^T\ (f - \dot{x}) + \mu^T\ C\ \mathrm{d}t', \quad (17)$$

with the predicted state trajectory starting at current state: $x'(0,t) = x(t)$. Based on the optimized, predicted future input trajectory $u'(t,t')$, the feedback control is realized by applying only its initial part $u(t) = u'(0,t)$ and continuously repeating the optimization using current measurements and receding the time horizon as the time passes.

In case of stiff dynamics, the state/costate integration step must be very short and in order to separate its length from the control horizon discretization grid t'_i, $i = 1..N_{t'}$, $u'(t',t)$ and $\mu(t',t)$ are parameterized using $N_{t'}$ discrete inputs $u^*_i(t)$ and Lagrange multipliers $\mu^*_i(t)$ (Noga et al. (2010))

$$u'(t',t) = \sum_{i=0}^{N_{t'}-1} \sigma_i(t')\ u_i^*(t) \tag{18}$$

$$\mu(t',t) = \sum_{i=0}^{N_{t'}-1} \sigma_i(t')\ \mu_i^*(t)\,, \tag{19}$$

with basis window functions:

$$\sigma_i(t') = \begin{cases} 1 & \text{if } t'_i \le t' < t'_{i+1} \\ 0 & \text{otherwise.} \end{cases} \tag{20}$$

The necessary condition for an extremum of $\bar{J}$, Eq. (17), are: the constraints (7), the costate dynamics

$$\mathrm{d}\lambda/\mathrm{d}t' = -H_x^T(x',u',\lambda,\mu) \tag{21}$$

$$\lambda(t+T,t) = \phi_x^T(x'(t+T,t)), \tag{22}$$

with the Hamiltonian H as in Eq. (6), and a nonlinear equation (Bryson and Ho (1975); Ohtsuka (2004); Noga et al. (2010)),

$$F(U(t),x(t),t) = 0, \tag{23}$$

$$F := [H_{u,0}\ C_0^T\ \cdots\ H_{u,N_{t'}-1}\ C_{N_{t'}-1}^T]^T, \tag{24}$$

$$U(t) := [u_0^{*T}\ \mu_0^{*T}\ \cdots\ u_{N_{t'}-1}^{*T}\ \mu_{N_{t'}-1}^{*T}]^T, \tag{25}$$

with the integrals

$$H_{u,i} := \int_{t'_i}^{t'_{i+1}} H_u\ \mathrm{d}t',\ C_i := \int_{t'_i}^{t'_{i+1}} C\ \mathrm{d}t'. \tag{26}$$

For a given sequence of $u_i^*(t)$ and $\mu_i^*(t)$, $\mathrm{d}x'/\mathrm{d}t'$ is integrated over the finite horizon $t < t' < t+T$, starting from $x(t)$. Then $\mathrm{d}\lambda/\mathrm{d}t'$ is integrated backwards from $t+T$ back to t. Finally, $H_{u,i}$ and C_i are evaluated and assembled into the residuum of the necessary optimality condition F.

Since H_u expressed directly in terms of system inputs and states has high computational cost, the integrals $H_{u,i}$ are calculated using simple quadratures such as one-point rectangular or two-points trapezoidal, where H_u is evaluated exclusively at the horizon grid. However, once the intermediate variables are introduced and are then available at each state integration step, the H_u may also be evaluated at each step at low additional computational cost, resulting in a more precise calculation of the optimality condition. Also the derivative $\mathrm{d}a_{O,k}/\mathrm{d}a_{I,k}$ employed in the Newton iteration, Eq. (5), is calculated using the intermediate variables.

The performance of the C/GMRES controller generated using automatic and semi-automatic procedure has been simulated, see Fig. 1. In the simulation, the prediction horizon length increases at the beginning to reach $\max(T) = 50$ min and is discretized with $N_{t'} = 10$ intervals. The time step for system dynamics integration is 62 times shorter than the horizon grid interval, thus the integrals $H_{u,i}$ and C_i are evaluated using one point and 62 points in automatic and semi-automatic code, respectively. The observed computational cost of the semi-automatically generated code is slightly less than that of automatically generated and the controller performance is similar in both cases.

4. CONCLUSIONS

AutoGenU automatically generates the C code of the Jacobians H_x and H_u, required to evaluate the necessary optimality condition in NMPC. For the case of the SHCC, which is a distributed parameter system, the Jacobians expressed using symbolic mathematics directly in terms of system inputs, states and co-states are complex due to the propagation of expressions along the cascading structure of the discretized system model. Based on AutoGenU, a semi-automatic NMPC code generation procedure has been developed for the SHCC, where intermediate variables are used to avoid this propagation, thereby reducing the complexity of the Jacobians. In the case of a stiff system, the time step of the system state dynamics integration is small and the intermediate values, which are available at each step, are used to evaluate the optimality condition more precisely at low additional computational cost. The observed computational cost of control update calculated using the semi-automatic code is slightly less than that generated automatically and the controller performance is similar in both cases. However, this semi-automatic code generation process requires significantly less memory, which makes it possible to apply the code generation process with analytically calculated Jacobians to more complicated systems.

ACKNOWLEDGMENTS

Part of this work was done during the stay of the first author at Osaka University for the "FrontierLab@OsakaU" programme. This work was supported by Spanish Government (project DPI2009-12805), University of Valladolid, CERN, Osaka University, Japan Student Services Organization and Grant-in-Aid for Scientific Research (C) No. 21560465. This support is very gratefully acknowledged.

REFERENCES

Brüning, O., Collier, P., Lebrun, P., Myers, S., Ostojic, R., Poole, J., and Proudlock, P. (2004). *LHC Design Report*. CERN.

Bryson, A.E. and Ho, Y. (1975). *Applied optimal control : optimization, estimation, and control*. Hemisphere Pub. Corp.

Gubello, G., Serio, L., and Soubiran, M. (2006). The circuits of the LHC Cryogenic System. Engineering Specification.

Lebrun, P., Serio, L., Tavian, L., and Van Weelderen, R. (1997). Cooling Strings of Superconducting Devices below 2 K: the Helium II Bayonet Heat Exchanger. *Adv. Cryog. Eng., A*, 43A, 419 426.

Noga, R. (2007). *Modeling and control of the String2 LHC Prototype at CERN*. Master's thesis, Gdansk University of Technology, University of Karlsruhe, Grenoble Institute of Technology.

Noga, R., Ohtsuka, T., de Prada, C., Blanco, E., and Casas, J. (2010). Nonlinear Model Predictive Control for the Superfluid Helium Cryogenic Circuit of the Large Hadron Collider. In *Proceedings of the 2010 IEEE International Conference on Control Applications*, 1654 – 1659.

Ohtsuka, T. (2000). AutoGenU: Readme.txt. URL `http://www-sc.sys.es.osaka-u.ac.jp/~ohtsuka/`.

Ohtsuka, T. (2004). A continuation/GMRES method for fast computation of nonlinear receding horizon control. *Automatica*, 40(4), 563 – 574.