8th International IFAC Symposium on
Dynamics and Control of Process Systems
Preprints Vol.1, June 6-8, 2007, Cancún, Mexico

FAST COMPUTATION OF THE HESSIAN OF THE LAGRANGIAN IN SHOOTING ALGORITHMS FOR DYNAMIC OPTIMIZATION

Ralf Hannemann, **Wolfgang Marquardt** [1]

Lehrstuhl für Prozesstechnik, RWTH Aachen University,
Turmstr. 46, 52064 Aachen, Germany

Abstract: One approach to solve optimal control problems by direct methods is the so called sequential approach or single shooting. Only the control variables are discretized resulting in a NLP which can be solved with SQP or interior point methods. This paper presents a new methodology to efficiently provide the Hessian of the Lagrangian of that resulting NLP. The algorithm is based on the second-order adjoint method and introduces the novel concept of composite adjoints to reduce the computational effort of a Hessian evaluation. Though, this contribution is for sake of simplicity restricting to single shooting, the same methodology can also be easily applied to multiple shooting.

Keywords: optimal control, single shooting, multiple shooting, Hessian, Lagrangian, adjoints

1. INTRODUCTION

Optimal control problems arise in many engineering applications. Originally dealing with problems in aerospace nowadays optimal control and especially nonlinear model predictive control (Rawlings, 2000), where a large number of related optimal control problems have to be solved on a moving horizon, is a current research issue of the process industry.

Direct methods have been proven to efficiently solve large scale optimal control and nonlinear model predictive control problems. They rely on a discretization of the optimal control problem and apply nonlinear programming techniques to solve the resulting finite dimensional nonlinear program (NLP). Depending on the kind of discretization and the formulation of the NLP one distinguishes between the *full discretization approach*, *multiple shooting* and *single shooting*. A comprehensive overview of the different solution techniques is given in the book of Grötschel *et al.* (2001). For the sake of notational simplicity this paper focuses on single shooting though the results are also applicable for the multiple shooting approach.

In single shooting typically medium size dense NLPs have to be solved. State of the art nonlinear programming solvers employ either interior point methods or sequential quadratic programming (SQP). In both cases either the exact or an approximate Hessian of the Lagrangian is required by the nonlinear programming solver. The contribution of this paper is a new methodology for the fast computation of this Hessian.

Hitherto most single and multiple shooting approaches employ coarse approximations of the Hessian by means of quasi-Newton updates since the computation of the Hessian has been to expensive so far. Vassiliadis *et al.* (1999) employ second-

[1] Corresponding author: marquardt@lpt.rwth-aachen.de

order variational equations for Hessian computation yielding a decrease in NLP iterations but an increase in the computational time.

The novel algorithm presented in this contribution overcomes the latter drawback. It is based on second-order adjoint equations. Haug and Ehle (1982) employ second-order adjoint equations for the sensitivity analysis of mechanical systems. Özyurt and Barton (2005) investigate the combination of directional second-order adjoint equations with automatic differentiation techniques. The latter approaches are very efficient when dealing with unconstrained optimal control problems by means of single or multiple shooting. The novelty of this contribution is the introduction of the concept of composite adjoints (cf. Section 7) to make the advantages of second-order adjoint sensitivity analysis available also for path constrained optimal control problems.

2. PROBLEM STATEMENT

We consider a class of Mayer-type optimal control problems ($\mathcal{OCP}$):

$$\begin{aligned}
\min_{u(t),p} \quad & \Phi(x(t_f)) && (1)\\
\text{s. t.} \quad & \dot{x}(t) = f(x(t),u(t))\,, && (2)\\
& x(t_0) = x_0(p_0) \in \mathbb{R}^{n_x}\,, && (3)\\
& g(x(t),u(t)) \leq 0 \in \mathbb{R}^{n_g}\,, && (4)\\
& h(x(t_f)) \leq 0 \in \mathbb{R}^{n_h}\,, && (5)\\
& t \in [t_0, t_f]\,, && (6)
\end{aligned}$$

where $x(t) \in \mathbb{R}^{n_x}$, $u(t) \in \mathbb{R}^{n_u}$, $p_0 \in \mathbb{R}^{P_0}$, $f(x(t),u(t)) \in \mathbb{R}^{n_x}$. The final time t_f is fixed or free. In the case of a free final time, $\mathcal{OCP}$ can be transformed in an equivalent optimal control problem with fixed final time. Without loss of generality we will assume a fixed final time from therefore.

Single shooting is one approach to solve $\mathcal{OCP}$. The basic idea is to substitute the control vector $u(t)$ by an approximation $u(t,p)$ employing parameters p_{ij} and basis functions $\phi_{ij}(t)$:

$$u_i(t,p) := \sum_{j=1}^{P_i} p_{ij}\,\phi_{ij}(t), \quad p_{ij} \in \mathbb{R}, \quad i = 1,\ldots,n_u\,. \tag{7}$$

Typically, the $\phi_{ij}(t)$ are constant, linear or cubic B-splines.

$$p = (p_{01},\ldots,p_{0P_0},p_{11},\ldots,p_{n_uP_{n_u}})^T \in \mathbb{R}^{n_p},$$

is the vector of all degrees of freedom.

The path constraints (4) are relaxed by defining a grid $t_0 < t_1 < t_2 < \cdots < t_N = t_f$ and writing:

$$g(x(t_k),u(t_k,p)) \leq 0,\ k = 1,\ldots N.$$

The infinite dimensional optimal control problem is approximated by a finite dimensional nonlinear program ($\mathcal{NLP}$):

$$\begin{aligned}
\min_{p} \quad & \Phi(x(t_f)) && (8)\\
\text{s. t.} \quad & \dot{x}(t) = f(x(t),u(t,p))\,, && (9)\\
& x(t_0) = x_0(p_0) \in \mathbb{R}^{n_x}\,, && (10)\\
& g(x(t_i),u(t_i,p)) \leq 0, \quad i = 1,\ldots,N && (11)\\
& h(x(t_f)) \leq 0\,, && (12)\\
& t \in [t_0, t_f]\,, && (13)
\end{aligned}$$

which can be considered as a discretization of $\mathcal{OCP}$. Note that equation (9) is solved by an underlying integration, which justifies the terminology. The state $x(t)$ is implicitly dependent from p. Nevertheless, for notational convenience, we will refer to the states as $x(t)$ instead of $x(t,p)$. For the same reason we will write $u(t)$ instead of $u(t,p)$ and x_0 instead of $x_0(p_0)$. The appreciated reader should keep in mind that $x(t)$, $u(t)$ and x_0 depend on p.

The Lagrangian of $\mathcal{NLP}$ can be stated as

$$\begin{aligned}
\mathcal{L}(p,\mu,\nu) =& \Phi(x(t_f)) + \left(\sum_{k=1}^{N} \mu_k^T\, g(x(t_k),u(t_k))\right)\\
& + \nu^T\, h(x(t_f)) \qquad (14)
\end{aligned}$$

with Lagrange multipliers $\mu_i \in \mathbb{R}^{n_g}, i = 1,\ldots,N$ and $\nu \in \mathbb{R}^{n_h}$.

3. REQUIREMENTS OF GRADIENT BASED OPTIMIZATION ALGORITHMS

For the solution of generally constrained nonlinear programs SQP and interior point methods have proven to be mostly efficient. Both methods require the gradients of the objective function and the constraint functions with respect to the unknowns p. In case of $\mathcal{NLP}$, these are

$$\begin{aligned}
& \frac{d\Phi(x(t_f))}{dp}\,, \quad \frac{dh(x(t_f))}{dp}\,,\\
& \frac{dg(x(t_k),u(t_k))}{dp},\ k = 1,\ldots,N\,. \qquad (15)
\end{aligned}$$

Furthermore, these methods require an approximation of the Hessian of the Lagrangian with respect to p:

$$\mathcal{L}_{pp}(p,\mu,\nu) = \frac{d^2\mathcal{L}(p,\mu,\nu)}{dp^2}\,. \tag{16}$$

Some optimization algorithms show an improved robustness and less iterations, if the exact Hessian instead of an approximation is employed, as shown by Vassiliadis *et al.* (1999). Vassiliadis et al. use second-order forward sensitivity equations to obtain the Hessian and therefore have the drawback

of a large computational effort. The number of iterations decreases, but the overall computational time increases.

The focus of this paper is to present a novel method to compute the exact Hessian (16) to yield a decrease of the number of iterations and simultaneously a decrease of the overall computational time.

4. FORWARD SENSITIVITY INTEGRATION

The gradients (15) can be computed by the chain rule, e.g. for $1 \leq k \leq N$:

$$\frac{dg(x(t_k),u(t_k))}{dp} = \frac{\partial g(x(t_k),u(t_k))}{\partial x}\frac{dx(t_k)}{dp} + \frac{\partial g(x(t_k),u(t_k))}{\partial u}\frac{du(t_k)}{dp}.$$

dx/dp can be calculated by forward sensitivity integration which is the method of choice in case of a large number of function gradients to be evaluated and a moderate number of parameters. The sensitivity equations are obtained by differentiating (9),(10) of $\mathcal{NLP}$ with respect to $p_i, i = 1,\ldots,n_p$:

$$\frac{d\dot{x}}{dp_i} = \frac{\partial f}{\partial x}\frac{dx}{dp_i} + \frac{\partial f}{\partial u}\frac{du}{dp_i}, \tag{17}$$

$$\frac{dx}{dp_i}(t_0) = \frac{dx_0(p)}{dp_i}. \tag{18}$$

5. FIRST-ORDER ADJOINT EQUATIONS

An alternative way to compute the wanted gradient is the so called adjoint sensitivity analysis. We demonstrate the application of the adjoint sensitivity analysis with a scalar functional $r(x,u)$ in order to determine the gradient

$$\frac{dr(x(t_k),u(t_k))}{dp_i}, \quad i = 1,\ldots,n_p,$$

for some $k \leq N$.

Instead of using the chain rule, the gradients of our example are computed directly introducing the so called adjoint variables $\lambda(t) \in \mathbb{R}^{n_x}$ (Cao *et al.*, 2002). The adjoint variables are computed by the integration of the first-order adjoint equations:

$$\dot{\lambda}(t) = -\lambda(t)^T f_x(x(t),u(t)), \tag{19}$$

$$\lambda(t_k)^T = r_x(x(t_k),u(t_k)), \tag{20}$$

$$t \in [t_0,t_k]. \tag{21}$$

The gradients are computed by the formula

$$\frac{dr(x(t_k),u(t_k))}{dp_i} = r_u(x(t_k),u(t_k))\frac{du}{dp_i}(t_k) + \int_{t_0}^{t_k} \lambda^T f_u \frac{du}{dp_i}\,dt + \lambda(t_0)^T\frac{dx_0}{dp_i}. \tag{22}$$

Suppose that we want to compute all gradients in (15) by means of first-order adjoint equations. Then, the total number of adjoint systems to solve is (15) is $N\cdot n_g + n_h + 1$, which can be computationally expensive, especially if N is large. Hence, for path-constrained optimal control problems, sensitivity equations are the method of choice to obtain first-order gradients.

6. SECOND-ORDER ADJOINT EQUATIONS

To compute second order sensitivities with respect to the parameters p_i and p_j by second-order adjoint equations, the first-order adjoint equations (19) – (22) are differentiated with respect to p_j (Özyurt and Barton, 2005):

$$\frac{d\dot{\lambda}}{dp_j}^T = -\frac{d\lambda}{dp_j}^T f_x - \lambda^T\left(\frac{d}{dp_j}f_x\right) \tag{23}$$

$$\frac{d\lambda}{dp_j}(t_k) = \left(r_{xx}\frac{dx}{dp_j} + r_{xu}\frac{du}{dp_j}\right)\Big|_{t=t_k}, \tag{24}$$

$$t \in [t_0,t_k]. \tag{25}$$

The second-order derivatives are computed by the formula

$$\begin{aligned}\frac{d^2r(x(t_k),u(t_k))}{dp_i dp_j} =& \\ &\left(\frac{du}{dp_i}^T r_{ux}\frac{dx}{dp_j} + \frac{du}{dp_i}^T g_{uu}\frac{du}{dp_j}\right)\Big|_{t=t_k} \\ &+ \int_{t_0}^{t_k}\left[\frac{d\lambda}{dp_j}^T f_u\frac{dx}{dp_i} + \lambda^T f_u\frac{d^2u}{dp_i dp_j}\right. \\ &\left. + \lambda^T\left(\frac{d}{dp_j}f_u\right)\frac{du}{dp_i}\right]dt \\ &+ \left(\frac{d\lambda}{dp_j}^T\frac{dx_0}{dp_i} + \lambda^T\frac{d^2x_0}{dp_i dp_j}\right)\Big|_{t=t_0}.\end{aligned} \tag{26}$$

We denote by

$$\lambda_p := \left(\frac{d\lambda}{dp_1},\ldots,\frac{d\lambda}{dp_{n_p}}\right)^T$$

the vector of all second-order adjoint variables. Since we are only interested in the second-order derivative of the Lagrangian, on the first sight, second-order adjoint sensitivity analysis seems to be the method of choice. Unfortunately, the ODE-embedded functionals that constitute the Lagrangian are evaluated at N different points in time. Following the ordinary second-order adjoint approach, N different second-order adjoint systems have to be solved which is not very efficient.

7. COMPOSITE ADJOINTS

In this section, we introduce the concept of so called composite adjoints and show, how the Hes-

sian can be computed by solving only one second-order adjoint system.

7.1 A homogenous system

We note that (19) together with (23) for $j = 1, \ldots, n_p$, form a linear time-variant homogenous DAE system. Introducing

$$\zeta(t) := \begin{pmatrix} \lambda(t) \\ \lambda_p(t) \end{pmatrix} \in \mathbb{R}^{n_x \cdot (n_p+1)}, \tag{27}$$

and defining the matrix function $A(t)$ in accordance with eqns. (19) and (23), the composite system has the form:

$$\dot{\zeta}(t) = A(t)\,\zeta(t). \tag{28}$$

It is important to note, that $A(t)$ is independent of $r(x,u)$. The dependence of the adjoint variable vector $\zeta(t)$ on $r(x,u))$ is covered by the final conditions:

$$\zeta(t_k) = \zeta_f(t_k; r) = \begin{pmatrix} r_x(x(t_k),u(t_k))^T \\ \left(r_{xx}\frac{dx}{dp_1} + r_{xu}\frac{du}{dp_1}\right)\Big|_{t=t_k} \\ \vdots \\ \left(r_{xx}\frac{dx}{dp_{n_p}} + r_{xu}\frac{du}{dp_{n_p}}\right)\Big|_{t=t_k} \end{pmatrix}, \tag{29}$$

with ζ_f defined in accordance with eqns. (20) and (24). Let ζ_1 and ζ_2 be two solutions of the system (28). Then, for $\alpha, \beta \in \mathbb{R}$, $\zeta = \alpha\,\zeta_1 + \beta\,\zeta_2$ is also a solution, since the system is homogenous. Furthermore, if we define $J(t; r(\cdot)), K(t), L$ in accordance with eqn. (25), then, for $i, j = 1, \ldots, n_p$, the second-order sensitivities are obtained from

$$\left.\frac{d^2 r}{dp^2}\right|_{t=t_k} = J(t_k; r) + \int_{t_0}^{t_k} K(t)\,\zeta(t)\,dt + L\,\zeta(t_0)\,, \tag{30}$$

where $J(t; r(\cdot))$ is the only function which depends on $r(\cdot)$.

Assume we are interested in a linear combination of the second-order derivatives

$$\sum_{k=1}^{N} \alpha_k \frac{d^2 r_k\,(x(t_k), u(t_k))}{dp^2} \tag{31}$$

of the different functionals $r_k(x(t_k), u(t_k)), k = 1, \ldots, N$, evaluated at N different points in time $t_1 < \cdots < t_N$. This is the case, if we are interested in the Hessian of the Lagrangian (14). In a straightforward manner, one would formulate N adjoint systems two get the single derivatives and then add the results:

$$\begin{aligned} &\zeta_k(t_k) = \zeta_{k,f}(t_k; r_k), \\ &\dot{\zeta}_k(t) = A(t)\,\zeta_k(t), \\ &t \in [t_0, t_k], \\ &\frac{d^2 r_k}{dp^2} = J(t_k; r_k) + \int_{t_0}^{t_k} K(t)\,\zeta_k(t)\,dt + L\,\zeta_k(t_0). \end{aligned}$$

By doing this, we solve N second-order adjoints systems. In the following, we show how the linear combination (31) can be obtained by solving one composite adjoint system.

7.2 Definition of composite adjoints

The main idea and the novelty of this contribution is to compute a composite adjoint vector $\zeta(t) \in \mathbb{R}^{n_x \cdot (n_p+1)}$ defined by

$$\zeta(t) := \sum_{k=l}^{N} \alpha_k \zeta_k(t)\,,\ t_{l-1} < t \le t_l,\ l = 1, \ldots, N\,.$$

Obviously $\zeta(t)$ satisfies eqn. (28) for $t \neq t_k, k = 1, \ldots, N$. We demonstrate how $\zeta(t)$ can be computed. The trajectory $\zeta(t)$ is computed in N steps. In the first step, the interval $(t_{N-1}, t_N]$ is treated by an integration backwards in time:

$$\begin{aligned} &\zeta(t_N) = \alpha_N\,\zeta_{N,f}(t_N; r_N)\,, && (32)\\ &\dot{\zeta}(t) = A(t)\,\zeta(t)\,, && (33)\\ &t \in (t_{N-1}, t_N]. && (34) \end{aligned}$$

In the k-th step, $k = 2, \ldots, N$, the solution on the interval $(t_{N-k}, t_{N-k+1}]$ is computed by a backwards integration as well:

$$\begin{aligned} &\zeta(t_{N-k+1}) = \alpha_{N-k+1}\,\zeta_{N-k+1,f}(t_{N-k+1}, r_{N-k+1}) \\ &\qquad + \zeta(t_{N-k+1}+), && (35)\\ &\dot{\zeta}(t) = A(t)\,\zeta(t), && (36)\\ &t \in (t_{N-k}, t_{N-k+1}]. && (37) \end{aligned}$$

The key point is, that in (35), the final value of the adjoint system belonging to the $(N-k)$-th ODE-embedded functional is added to the value of the composite adjoint vector. Since

$$\int_{t_0}^{t_N} K(t)\,\zeta(t)\,dt = \sum_{k=1}^{N} \alpha_k \int_{t_0}^{t_k} K(t)\,\zeta_k(t)\,dt \tag{38}$$

and

$$L\,\zeta(t_0) = \sum_{k=1}^{N} \alpha_k\,L\,\zeta_k(t_0)$$

hold, the linear combination of the second-order derivatives is obtained by

$$\sum_{k=1}^{N} \alpha_k\,J(t_k; r_k) + \int_{t_0}^{t_N} K(t)\,\zeta(t)\,dt + L\,\zeta(t_0)\,. \tag{39}$$

8. COMPUTATION OF THE HESSIAN OF THE LAGRANGIAN

In the preceding sections we founded the prerequisites for the fast computation of the Hessian of the Lagrangian (14). Setting

$$r_N := \Phi + \mu_N^T g + \nu^T h,\ r_i := \mu_i^T g,\ i = 1, \ldots, N-1, \text{ and } \alpha_i := 1, i = 1, \ldots, N\,, \tag{40}$$

we have exactly the situation as sketched in section 7. Hence, the Hessian of the Lagrangian can be computed by one integration of one second-order adjoint system.

9. NUMERICAL CASE STUDY

We demonstrate the application of the proposed methodology by the optimal control of the van der Pol oscillator.

9.1 van der Pol oscillator

The problem statement is taken from (Augustin and Maurer, 2001):

$$\begin{aligned} \min_{u(t)} \quad & x_3(t_f) \\ \text{s. t.} \quad & \dot{x}_1 = x_2\,, \\ & \dot{x}_2 = (1-x_1)^2 x_2 - x_1 + u\,, \\ & \dot{x}_3 = x_1^2 + x_2^2 + u^2\,, \\ & x(0) = x_0\,, \\ & \gamma - x_2(t) \le 0\,, \\ & t \in [0, t_f]\,. \end{aligned}$$

The parameter settings are $t_f = 5, \gamma = -0.4$ and $x_0 = (1, 0, 0)^T$.

9.2 Discretization

The control variable is approximated by a piecewise constant function on a equidistant grid $0 = \tau_0 < ... < \tau_{n_p} = t_f$:

$$u(t; p) = p_i,\ t \in (\tau_{i-1}, \tau_i], \tau_i = \frac{i \cdot t_f}{n_p}\,.$$

The path constraint $\gamma - x_2(t) \le 0$ is relaxed by pointwise constraints on a equidistant grid $0 = t_0 < ... < t_{\rho\, n_p} = t_f$ with a positive integer ρ:

$$\gamma - x_2(t_i) \le 0,\ t_i = \frac{i \cdot t_f}{\rho\, n_p},\ i = 0, \ldots, \rho\, n_p\,.$$

9.3 Implementation details

The optimizer IPOPT (Wächter and Biegler, 2006) is chosen to perform the optimization, because it is capable to employ either BFGS updates of the Hessian or the exact Hessian supplied by the user. BFGS updates are a technique to construct a coarse approximation of the Hessian by first-order gradient information (Fletcher, 1970). The latest pre-3.0 Fortran version of IPOPT is employed. As integrator the explicit Runge-Kutta code DOP853 (Hairer and Wanner, 1993) is employed for the state and first-order sensitivity integration. The error tolerance for the integration is set to $Tol = 10^{-10}$, both for the absolute and the relative error.

For the integration of the second-order adjoint system two alternative integration schemes were utilized. On the one hand, again DOP853 with $Tol = 10^{-10}$ is employed. On the other hand a simple explicit Runge-Kutta scheme of order 2 without error control is employed by just taking one Runge-Kutta step from t_i to t_{i-1} yielding an inaccurate Hessian for coarse grids if $\rho \cdot n_p$ is small. The reason for this approach is that second-order sensitivities usually don't need to be as accurate as first-order sensitivities and the computation of the Hessian can be sped up significantly by using a low-order integration scheme. For the Hessian evaluation by means of second order adjoint equations, the states and first-order sensitivities are stored at the points $t_i,\ i = 0, \ldots, \rho\, n_p$. Instead of using an interpolation for the backwards integration of the first- and second-order adjoint system, the differential equations for the states and first-order sensitivities are solved simultaneously. The quadrature problem in (26) is reformulated as ODE and solved simultaneously as well. A starting vector $p^T = (0.7, \ldots, 0.7)$ is chosen.

The optimizer IPOPT evaluates the constraint Jacobian prior to a call of the Hessian. Consequently in the implementation the trajectories of the states and sensitivities are stored during the Jacobian evaluation and need not be recalculated during the Hessian evaluation.

All computations are performed on a Windows XP PC with a 3.0 GHz Intel Pentium D processor and a memory of 1024 MB using a single core. The numerical test examples are coded in Fortran 95 and compiled with the Intel Fortran compiler (version 9.1) utilizing the "Maximize Speed"-optimization option.

9.4 Results

The main purpose of this numerical case study is not a performance comparison but a demonstration of the general applicability of the proposed algorithm.

In Table 1 the number of iterations of IPOPT is presented: BFGS refers to IPOPT using BFGS-updates, RUNGE refers to the inaccurate Hessians computed by the simple two-order Runge-Kutta scheme and DOP853 refers to the Hessian computed by DOP853. The number of iterations of IPOPT (Nit) has the following relations to the number of Jacobian evaluations ($Njac$) and Hessian evaluations ($Nhess$):

$$Njac = Nit + 2, \qquad Nhess = Nit + 1\,.$$

Hence a comparison of the number of iterations with the average computational times for a Ja-

n_p	ρ	BFGS	RUNGE	DOP853
50	1	24	13	12
50	2	27	15	15
200	1	23	15	15
200	2	27	18	18
500	1	26	18	18
500	2	29	19	19

Table 1. Number of iterations

n_p	ρ	SENS	RUNGE	DOP853
50	1	0.005	< 0.001	0.013
50	2	0.008	0.001	0.015
200	1	0.043	0.012	0.120
200	2	0.08	0.02	0.23
500	1	0.25	0.08	0.73
500	2	0.50	0.16	1.48

Table 2. Computational times in seconds

n_p	ρ	SENS	RUNGE	DOP853
50	1	0.15	0.10	0.25
50	2	0.35	0.20	0.45
200	1	1.1	1.0	3.1
200	2	2.5	2.2	6.2
500	1	7.2	6.7	19.1
500	2	16.7	14.0	40.8

Table 3. Total time spent in function evaluations in seconds

cobian or Hessian evaluation gives information about the overall computational effort. The average computational times for one Jacobian evaluation (SENS), one low-order inaccurate Hessian evaluation (RUNGE) and one accurate Hessian evaluation (DOP853) are given in Table 2. The total time spent in function evaluations is given in Table 3. The results show that the supplement of the "exact" Hessian always reduces the number of iterations in comparison with BFGS updates. Even if the Hessian is very inaccurate as in the first test case, where only 10 low-order Runge-Kutta steps were performed to compute the Hessian, the number of iterations decreases. Since the low-order computation of the Hessian is even faster than the Jacobian evaluation the overall solution of the optimization problem is sped up in comparison to the BFGS updates.

Employing a highly accurate Hessian by the high-order code DOP853 still reduces the number of iterations but because of the higher computational effort for a Hessian evaluation, the overall computation time increases. Though the computational effort for the highly accurate Hessian is still acceptable. Employing alternative methods like finite differences or second-order sensitivity equations would certainly lead to poorer computational times.

10. CONCLUSIONS AND OUTLOOK

A new methodology to efficiently provide the Hessian of the Lagrangian of in single shooting has been proposed. The algorithm employs the novel concept of composite adjoints to reduce the computational effort of a Hessian evaluation. An ad hoc-implementation of the algorithm based on explicit Runge-Kutta schemes has been successfully applied to a numerical case study showing the potential of this algorithm. Although not an issue of this contribution that algorithm can easily be adapted for multiple shooting.

Future research focuses on a more reliable and more efficient implementation of the algorithm by utilizing integration schemes for stiff and differential-algebraic systems. Furthermore the exploitation of exact Hessians for on-line applications will be investigated.

REFERENCES

Augustin, Dirk and Helmut Maurer (2001). Computational sensitivity analysis for state constrained control problems. *Annals of Operations Research* **101**, 75–99.

Cao, Y., S. Li, L. Petzold and R. Serban (2002). Adjoint sensitivity analysis for differential-algebraic equations: The adjoint dae system and its numerical solution. *SIAM J. Sci. Comput.* **24**(3), 1076–1089.

Fletcher, R. (1970). A new approach to variable metric algorithms. *The Computer Journal* **13**, 317–322.

Grötschel, M., S. O. Krumke and J. Rambau (2001). *Online Optimization of Large Scale Systems*. Springer, Berlin.

Hairer, E. and G. Wanner (1993). *Solving Ordinary Differential Equations I – Nonstiff Problems*. Springer, Berlin.

Haug, E. J. and P. E. Ehle (1982). Second-order design sensitivity analysis of mechanical system dynamics. *Internat. J. Numer. Methods Engrg.* **18**, 1699–1717.

Özyurt, Derya B. and Paul I. Barton (2005). Cheap second order directional derivatives of stiff ode embedded functionals. *SIAM J. Sci. Comput.* **26**(5), 1725–1743.

Rawlings, J. B. (2000). Tutorial overviwe of model predictive control. *IEEE Control Systems Magazine*.

Vassiliadis, V. S., E. B. Canto and J. R. Banga (1999). Second-order sensitivities of general dynamic systems with applications to optimal control problems. *Chemical Engineering Science* **54**(17), 3723–3955.

Wächter, A. and L. T. Biegler (2006). On the implementation of a primal-dual interior point filter line search algorithm for large-scale nonlinear programming. *Mathematical Programming* **106**(1), 25–57.