from *SIAM News*, Volume 33, Number 4

Parallel Parameter Estimation in Full Motor Vehicle Dynamics

By Torsten Butz, Oskar von Stryk, Martin Vögel, Thieß-Magnus Wolter, and Cornelius Chucholowski

The numerical simulation of vehicle dynamics plays an important role in the development of new motor vehicles. With numerical simulations, the road performance and handling properties of a new vehicle can be investigated before prototypes are built. In addition to the reduced need for physical prototyping, real-time vehicle dynamics simulations within a hardware-in-the-loop test-bench allow the testing of control units important to driving performance—such as anti-lock braking systems and electronic stability programs—without endangering a test driver or prototype car (see Figure 1).

APPLICATIONS ON ADVANCED ARCHITECTURE COMPUTERS
Greg Astfalk, Editor

The development of intricate electronic vehicle control devices requires an accurate match between the performance of the virtual car and the actual vehicle behavior. A detailed and comprehensive vehicle model is needed to allow for the nonlinear kinematics of wheel and axle, and to depict the drive train, steering mechanism, and tire dynamics. The vehicle model described in this article consists of a suitable multibody system that includes force elements and kinematic connections, in addition to a sophisticated tire model.

General-purpose methods for modeling multi-body systems use the descriptor form for the equations of motion, leading to a large-scale system of differential–algebraic equations (DAEs) with index 3 [13]. For DAE systems, particular numerical techniques must be applied to prevent "drift-off" from the algebraic constraints. We use a tailored model description that yields a system of ordinary differential equations (ODEs).

Since not all the coefficients of the differential equations are readily available, a procedure for calibrating the parameters of the virtual prototype is needed. For use in a test-bench, the desired model parameters need to be adjusted as the testing proceeds. This identification process can be achieved efficiently by means of mathematical optimization algorithms. In practice, the required first-order derivative information can be obtained only from finite-difference approximations. Parallel processing of the objective function evaluations can lead to considerable reductions in the computational time for the optimization. Vehicle dynamics programs are usually developed and implemented by automotive suppliers, and a low-cost parallel computing platform is therefore desirable. The approach presented here makes use of a heterogeneous network of PCs to perform fast and reliable parameter estimation for dynamic vehicle models.

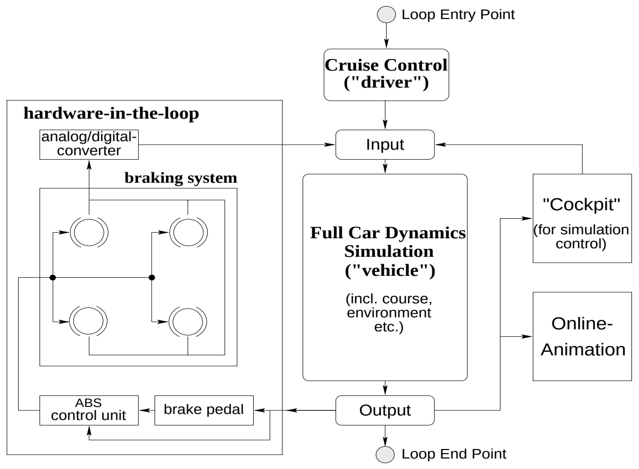

Figure 1. *Hardware-in-the-loop setup for a braking system with an ABS control unit linked to the real-time simulation of full vehicle dynamics.*

Numerical Simulation of Full-car Dynamics

In the work described here we used *vedyna* [1], a vehicle dynamics program developed and distributed commercially by TESIS DYNAware. The *vedyna* simulation results show good agreement with real vehicle behavior. Simulations with time-steps in the millisecond range can be carried out in real time with PC hardware. When implemented on Compaq Alpha multiprocessor boards (as offered by dSPACE from Paderborn, Germany), real-time performance can be achieved for time-steps of less than 0.5 millisecond.

The vehicle model in *vedyna* consists of a system of nine rigid bodies comprising the vehicle body, axle suspensions, and wheels; additional partial models depict the characteristics of the drive train, steering mechanism, and tires (see Figure 2). Suitable minimum coordinates and generalized velocities are used to describe the spatial state of the vehicle and its components [12]. The equations of motion, derived from Jourdain's principle, are:

$$M_{BV}\,\dot{z}_{BV} = Q_{BV}\left(y_{BV}, z_{BV}, y_{ST}, z_{ST}, y_{DT}, z_{DT}\right) \tag{1}$$

$$\dot{y}_{BV} = K_{BV}^{-1}\left(y_{BV}\right) z_{BV} \tag{2}$$

$$M_{DT}\,\dot{z}_{DT} = Q_{DT}\left(y_{DT}, z_{DT}\right) \tag{3}$$

$$\dot{y}_{DT} = V_{DT}\, z_{DT} \tag{4}$$

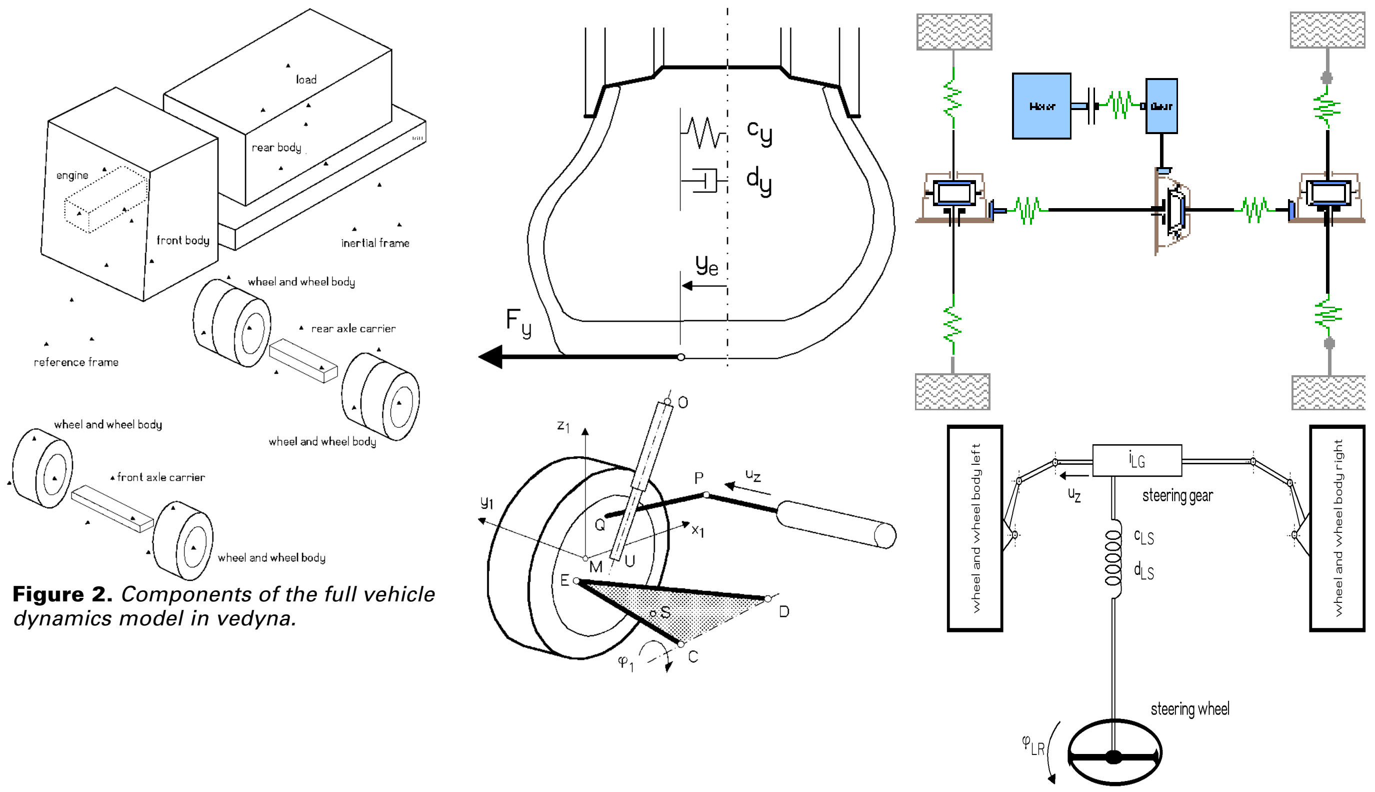

Figure 2. *Components of the full vehicle dynamics model in vedyna.*

$$M_{ST}\ \dot{z}_{ST} = Q_{ST}\left(y_{ST}, z_{ST}\right) \tag{5}$$

$$\dot{y}_{ST} = V_{ST}\ z_{ST} \tag{6}$$

$$D\ \dot{y}_T = F_{stat} - C\ y_T \tag{7}$$

The vehicle dynamics is fully characterized by a system of 24 first-order ODEs, (1) and (2), comprising the vehicle body and axles. Eight ODEs (7) for the spring and damper elements describe the deviations of the tires. The dynamic model of the drive train consists of 19 ODEs, (3) and (4), including four equations governing the angular wheel speeds. Five additional ODEs, (5) and (6), account for the dynamics of the steering system. Couplings between the separate systems occur via the generalized forces and torques Q_{BV}.

Owing to the stiffness of this system of ODEs, numerical integration is carried out via a semi-implicit one-step Euler scheme with a constant time-step. This tailored integration method allows a stable solution for time-steps in the range of several milliseconds. To allow realistic implementations of virtual test-drives in the simulation, we developed models for the driver and the road [4]. Animated results for various vehicle maneuvers, including the full brake maneuver described later in this article, are available at http://www.siam.org/siamnews/05-00/animations.

Numerical Estimation of Vehicle Parameters

The system of 56 highly nonlinear ODEs describing the vehicle dynamics, equations (1)–(7), can be summarized as:

$$\dot{x}(t) = g\left(x(t), p, t\right) \tag{8}$$

with suitable initial values

$$x(t_0) = x_0 \tag{9}$$

In addition to the vehicle's state variables $x(t) \in \mathrm{IR}^{n_x}$, which include the minimum coordinates and generalized velocities, the full-vehicle performance is also determined by a set of model parameters $p \in \mathrm{IR}^{n_p}$. All the model parameters are constant for all times t.

For adjustments to the mathematical model, observations of the real vehicle behavior are used to determine suitable estimates for the model parameters. A physical prototype of the vehicle is used to conduct driving tests, during which we record measurement values,

$$\eta_{ij} = x_i(t_j) + \varepsilon_{ij} \quad i \in I_j, \quad j = 1, \ldots, n_t \tag{10}$$

of selected vehicle state variables x_i, $i \in I_j$ at n_t measurement times t_j. Here, $I_j \subseteq \{1,...,n_x\}$ denotes the subset of vehicle states that are available, or desired, at time t_j. The observed quantities are often subject to measurement errors $\varepsilon_{ij} \in \mathrm{IR}$.

Consequently, we estimate the optimal values of the parameters p^* by minimizing the difference between the observed values η_{ij} and the corresponding results obtained by numerical integration. To do so, we solve the nonlinear least-squares problem

$$\begin{aligned}\underset{p \in \mathrm{IR}^{n_p}}{\text{minimize}}\ r(p) &:= \frac{1}{2}\left\| f(p) \right\|_2^2 \\ &:= \frac{1}{2}\sum_{j=1}^{n_t}\sum_{i \in I_j}\left(\eta_{ij} - x_i\left(t_j, p\right)\right)^2\end{aligned} \tag{11}$$

Here, $x(t, p)$ is the numerical solution of the model equations (8) for the parameter set p. Each evaluation of the objective function requires the integration of the initial value problem (8)–(9) for the specific parameter values. The optimal solution often needs to satisfy additional box constraints,

$$l_\mathrm{i} \le p_\mathrm{i} \le u_i \qquad i = 1, \ldots, n_p \tag{12}$$

on the parameter range. These constraints ensure optimization results compatible with the real vehicle properties.

An iterative algorithm is used, starting from a feasible initial guess p^0 and gradually improving the parameter values,

$$p^{k+1} = p^k + \alpha_k d^k \quad k = 0, 1, 2, \ldots \tag{13}$$

which are intended to converge to an optimal solution. Here, $d^k \in \mathrm{IR}^{n_p}$ and $\alpha_k > 0$ denote a suitable descent direction and a positive step-size, respectively.

The solution of the present parameter estimation problem—the box-constrained nonlinear least-squares problem (11)–(12)—is achieved with numerical algorithms that apply to both unconstrained and constrained optimization. For the unconstrained problem, we use the Gauss–Newton method, implemented in the NLSCON code [5, 11], and the Levenberg–Marquardt algorithm, in the LMDER code [10]. Each of these techniques determines a sequence of improved parameter estimates by solving an approximate linear least-squares problem at every iteration.

For the box-constrained optimization, we use the sequential quadratic programming code NLSSOL [7, 8], which exploits the specific least-squares structure of the objective function. Moreover, the optimization is carried out with the implicit filtering code IFFCO [9], a projected quasi-Newton method designed for the solution of noisy minimization problems. For reasons of efficiency, we have implemented algorithms that rely only on the first-order derivatives of the objective function with respect to the model parameters. Additional speedup is achieved with the parallel approach, as described in the following section.

Parallel Optimization

The parameter estimation problem is solved in a framework that couples the optimization algorithms with the vehicle dynamics simulation performed by *vedyna* [3].

Because of the complexity of the vehicle model, the required first-order derivatives of the objective function $r(p)$ are generated by finite differences. For the optimization with NLSCON, LMDER, and NLSSOL, the partial derivatives $\partial_i f(p)$ with respect to the ith parameter estimate are approximated by the one-sided differences

$$\left(\partial_i f(p)\right)_{\pm h} = \frac{f\left(p \pm he_i\right) - f(p)}{\pm h} \tag{14}$$

depending on the feasibility of $p + he_i$ or $p - he_i$. Here, $e_i \in \mathrm{IR}^{n_p}$ denotes the ith canonical unit vector, and $h > 0$ is a suitable finite-difference increment. The implicit filtering code IFFCO makes use of central-difference approximations,

$$\left(\partial_i f(p)\right)_{2h} = \frac{f\left(p + he_i\right) - f\left(p - he_i\right)}{2h} \tag{15}$$

whenever possible. Therefore, computation of the objective function gradient $\nabla r(p)$ generally requires n_p to 2 n_p additional evaluations of the objective function, depending on whether one-sided or central differences are used. The numerical accuracy of the objective function value evaluations is limited by peculiarities in the integration scheme for the equations of motion and by simplifications in the vehicle model of *vedyna*. Consequently, special care must be taken in the choice of the finite-difference increments [8]. It is worth mentioning that the tailored modeling and simulation approach prohibits the application of automatic differentiation or internal numerical differentiation techniques [2, 3].

Most of the time required for the identification process is spent on numerical integration of the model equations (8) and (9) for the varying sets of parameters. Considerable savings in computational time can be achieved if the objective function evaluations are performed in parallel on a cluster of computers. The necessary communication between processes on different machines across the network is handled by remote procedure calls (RPCs). We use the ONC RPC library from Sun Microsystems, ported to Microsoft Windows [6].

For our parallel approach, the actual optimization is performed by the client process. When the objective function derivatives are needed, the remote server routines are called asynchronously, i.e., the additional objective function evaluations for the arguments $p \pm he_i$, $i = 1, \ldots, n_p$, are carried out simultaneously by all available server processes. For the central-difference computations, one of these evaluations is performed by the client program since the objective value for the current iterate p is not required. The UDP transport protocol is used for the exchange of data between client and server processes, because only function arguments of moderate size need to be communicated.

Optimization Results

Our program was used to estimate the brake friction coefficients for a commercial passenger car in *vedyna* [3]. With only a limited number of estimated vehicle parameters, we used appropriate parameter sets, validated by TESIS DYNAware for the same vehicle type, for the remaining coefficients of the vehicle model.

The test data consisted of measurement values recorded during a full brake maneuver. The vehicle started at rest and accelerated to the experimentally observed speed of approximately 70 km/hour over a period of 22 seconds before braking. The recorded state variables for the vehicle consisted of the brake pressures at the four wheels (see Figure 3) and the corresponding rotational wheel speeds. All quantities were measured at intervals of 6×10^{-3} seconds at $21.6 \leq t \leq 24.6$. The maneuvering data for the vehicle were provided by an automotive supplier.

The estimated brake friction coefficients are the ratios for generated braking moments transmitted to the four wheels. Because only disc brake models are used in *vedyna*, regardless of the actual brake mechanism, these coefficients reflect adjustments to the vehicle model rather than estimates of true physical properties.

We obtained optimal parameter estimates by performing the optimization on a heterogeneous Windows NT 4.0 and Windows 98 network at TESIS DYNAware and TESIS WAMware in München. Suitable initial guesses,

$$p^0 = (0.33, 0.33, 0.34, 0.34)^T \tag{16}$$

were chosen from the default parameter values for the vehicle model. The initial parameter estimates (16) refer to the friction coefficients for the left and right wheels, front and rear, respectively. The associated least-squares residual, indicating the quality of the parameter values, is $r(p^0) = 53289.3$.

The optimization produced a minimum residual $r(p^*) = 2603.2$ for the parameter values

$$p^* = (0.1954, 0.1772, 0.1672, 0.1669)^T \tag{17}$$

which were computed by the implicit filtering code IFFCO. The simulation results are compared with experimental measurements of the rotational wheel speeds in Figure 4. The erratic deviations seen in Figure 4(b) must be attributed to measurement errors. When these deviations are neglected, good agreement is achieved between the model and the experimental measurements. The deviations between the estimates (17) for the left and right front wheels can be attributed to measurement errors in the input brake pressures and to differences in wear for the respective braking devices.

We performed a statistical analysis to assess the quality of the parameter estimates. By linearizing the least-squares function $f(p)$ about the optimal solution, we obtained the objective function's Jacobian $J = \partial f(p^*)/\partial p$ and the approximate variance–covariance matrix $\sigma^2(J^TJ)^{-1}$. Here, σ is an estimate of the standard deviation in the measurement errors ε_{ij}. We also computed approximate 95%-confidence intervals: [0.1942,0.1967], [0.1762, 0.1782], [0.1664,0.1680], and [0.1662, 0.1677], respectively. Despite their limited reliability in the nonlinear least-squares case, the small interval widths indicate that the computed values of the desired friction coefficients are acceptable.

For comparison, the numerical optimization was also carried out sequentially, i.e., the objective function evaluations required for computation of the finite differences were also performed by the client process. In this case a Dell 400-MHz PC was used. The processor time required for evaluation of the objective function was, on average, 6.1 seconds.

In the parallel approach, the client process was executed on the same machine. The four objective function evaluations for the forward-difference computations

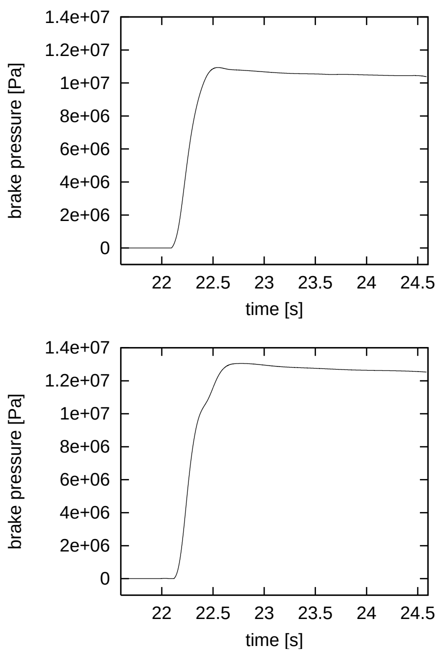

Figure 3. *Input brake pressures at the left front (a, top) and rear (b, bottom) wheels.*

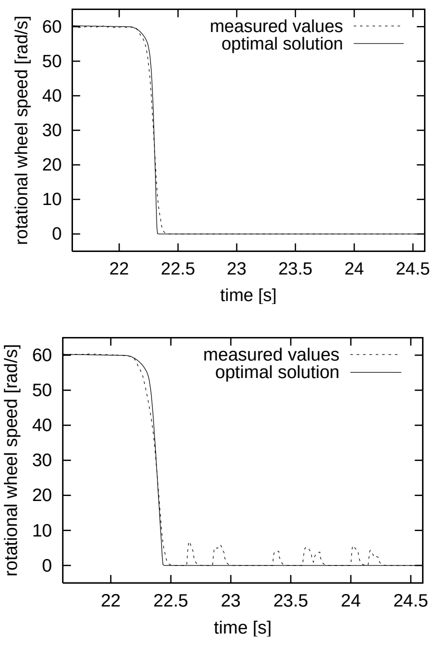

Figure 4. *Rotational speeds of the left front (a, top) and rear (b, bottom) wheels.*

in NLSCON, LMDER, and NLSSOL were performed on two Sie-mens 450-MHz PCs and two Dell 333-MHz notebooks. The processor times for the objective function evaluations were 5.5 seconds and 6.5 seconds, respectively. The three additional parallel evaluations for the central-difference computations in IFFCO were done on a Dell 300-MHz notebook and two Siemens PCs with 300-MHz and 266-MHz CPUs. The processor times were 8.2 seconds, 7.3 seconds, and 8.8 seconds, respectively. The time for communication between the client and server processes is negligible compared with the computational time required for the optimization process and the objective function evaluations.

Algorithm	$r(p^*)$	n_{seq}	t_{seq}	n_{par}	t_{par}	t_{seq}/t_{par}
IFFCO	2603.2	204	1252.1	64	445.6	2.81
LMDER	2613.5	55	338.5	19	122.7	2.76
NLSCON	2612.7	68	416.9	24	156.7	2.66
NLSSOL	2607.9	115	704.4	23	152.3	4.63

Table 1. *Comparison of computational results for the sequential and parallel parameter estimation schemes.*

Table 1 compares the different optimization codes [3], giving the least-squares residual $r(p^*)$ for the respective solutions, and the computational times t_{seq} and t_{par} for the sequential and parallel executions of the optimization. The ratios of these CPU times, i.e., the parallel speedup achieved, are shown in the last column. Also listed are the numbers of objective function evaluations performed during the entire optimization (n_{seq}) and by the client process in the parallel framework (n_{par}).

For this problem, all the algorithms produced reasonably small residuals. Similarly, the parameter estimates are bounded in the confidence intervals. Parallel execution of the finite-difference computations reduced the computational time for all algorithms by more than a factor of two. The speedup for the sequential quadratic programming code NLSSOL was nearly optimal. The time for the implicit filtering code was reduced by a factor of approximately three.

Conclusions

To calibrate the vehicle model of a commercial vehicle dynamics program, we developed a parameter estimation tool that relies on observations obtained from test drives. The associated nonlinear least-squares problem can be solved by means of mathematical optimization algorithms, most of which make use of gradient information. Because of the complexity of the vehicle dynamics program, only the objective function derivatives can be approximated with finite differences. Use of this approach, however, results in significant savings in computational time when the additionally required objective evaluations are performed in parallel. The low-cost parallel computing platform used, consisting of a heterogeneous cluster of PCs, is well suited for the needs of the automotive industry and suppliers applying vehicle dynamics simulations.

References

[1] *vedyna User's Guide*, TESIS DYNAware, München, Germany, 1997.

[2] H.G. Bock, *Recent advances in parameter identification techniques for ODE*, in P. Deuflhard and E. Hairer (eds.), *Numerical Treatment of Inverse Problems in Differential and Integral Equations, Progress in Scientific Computing*, 2, Birkhäuser, Boston, 1983, 95–121.

[3] T. Butz, *Parameter Identification in Vehicle Dynamics*, Diploma Thesis, Zentrum Mathematik, Tech-nische Universität München, 1999.

[4] C. Chucholowski, M. Vögel, O. von Stryk, and T.-M. Wolter, *Real-time simulation and online control for virtual test drives of cars*, in H.-J. Bungartz, F. Durst, and C. Zenger (eds.), *High Performance Scientific and Engineering Computing, Lecture Notes in Computational Science and Engineering*, 8, Springer-Verlag, Berlin, 1999, 157–166.

[5] P. Deuflhard and V. Apostolescu, *An underrelaxed Gauss–Newton method for equality-constrained nonlinear least squares problems*, in J. Stoer (ed.), *Optimization Techniques, Lecture Notes in Control and Information Sciences*, 7, Springer-Verlag, Berlin, 1978, 22–32.

[6] M. Gergeleit, ONC RPC for Windows NT Homepage, http://www.dcs.qmw.ac.uk/~williams/nisgina-current/src/rpc110/oncrpc.htm, 1996.

[7] P.E. Gill, W. Murray, M.A. Saunders, and M.H. Wright, *User's guide for NPSOL (Version 5.0): A Fortran package for nonlinear programming*, Numerical Analysis Report 98–2, Department of Mathematics, University of California, San Diego, 1998.

[8] P.E. Gill, W. Murray, and M.H. Wright, *Practical Optimization*, Academic Press, London and New York, 1981.

[9] P. Gilmore and C.T. Kelley, *An implicit filtering algorithm for optimization of functions with many local minima*, SIAM J. Optimization, 5 (2), 1995, 269–285.

[10] J.J. Moré, *The Levenberg–Marquardt algorithm: Implementation and theory*, in A. Dold and B. Eckmann (eds.), *Numerical Analysis, Lecture Notes in Mathematics*, 630, Springer-Verlag, 1978, 105–116.

[11] U. Nowak and L. Weimann, *A family of Newton codes for systems of highly nonlinear equations*, Technical Report TR 91–10, ZIB, Berlin, 1991.

[12] G. Rill, *Simulation von Kraft-fahrzeugen*, Vieweg, Braunschweig, Germany, 1994.

[13] W. Schiehlen, *Multibody Systems Handbook*, Springer-Verlag, Berlin, 1990.

Torsten Butz (butz@ma.tum.de) and Martin Vögel (voegel@ma.tum.de) are research scientists at the Chair M2 for Numerical Analysis of the Technische Universität München, Germany. Oskar von Stryk (stryk@ma.tum.de) is an assistant professor in the Department of Mathematics at the Technische Universität München. Thieß-Magnus Wolter (t.wolter@tesis.de) is a project engineer at TESIS DYNAware, München. Cornelius Chucholowski (c.chucholowski@tesis.de) is senior scientist and CEO of TESIS DYNAware, München. The project has been supported by FORTWIHR, the Bavarian Consortium for High Performance Scientific Computing.