Problem

The numerical solution of optimal control problems by indirect methods (such as multiple shooting or collocation) usually requires a considerable amount of analytic calculation to establish a numerically tractable system. The reason for that is that certain steps in the analytical preparation of the calculation, which are simple in principle, may be very elaborate and can lead to rather complex expressions. Implementation of these into numerical code by hand is tiresome and error-prone.

In spite of the rather high standards of numerical algorithms for such problems (BOCK[1, 2, 3], BULIRSCH [4], DEUFLHARD/FIEDLER/KUNKEL [6]) the analytic calculations—often classified as simple in principle, but rather tedious in realistic examples—are nowadays mostly still done by hand—and thus prone to calculation errors. We present the package OCCAL [9] (mnemotechnically for Optimal Control CALculator), a system capable of automating this analytic processing to a reasonable extent by means of a modern symbolic manipulation language, intending to solve as many of the arising problems as possible.

Our combined symbolic-numeric system symbolically analyzes the given problem, using the Computer Algebra System REDUCE. This approach additionally permits to generate optimized and correct numerical subprograms for use with existing numerical solvers for boundary value problems. These are employed in the second step to obtain numerical solutions.

Let us introduce in short the notation we are going to use throughout this paper. We start from a vector of state variables $y(t)$, $y: [a, b] \rightarrow \mathbb{R}^n$ and a vector of control variables $u(t)$, $u: [a, b] \rightarrow \mathbb{R}^k$ which are subject to a system of ordinary differential equations:

$$y' = f(t, y, u)$$

with boundary conditions

$$r(y(a), y(b)) = 0, \quad r: \mathbb{R}^{2n} \rightarrow \mathbb{R}^n.$$
At the moment we restrict ourselves to problems where the functional to be mini-
mimized is an integral of a function depending on the state and control variables.
We intend to enlarge this class in the future. The problem is the minimization of
the integral
\[
I[u] := \int_a^b f_0(y, u, t) \, dt.
\]
Introducing the \( n \) adjoint variables \( \lambda_i(t) \) we obtain the Hamiltonian
\[
H(t, y, \lambda, u) := f_0(t, y, u) + \sum_{i=1}^n \lambda_i f_i(t, y, u)
\]
that is to be minimized according to the minimum principle by Pontryagin. This
leads to the canonical equations
\[
\begin{align*}
y'_i &= H_{\lambda_i} = f_i(t, y, u), \\
\lambda'_i &= -H_{y_i}, \\
&= -\frac{\partial f_0}{\partial y_i}(t, y, u) - \sum_{j=1}^n \lambda_i \frac{\partial f_j}{\partial y_i}(t, y, u).
\end{align*}
\]
Together with the boundary conditions above, we now have a boundary value
problem.

Overview of symbolic tasks

In this section we will present the essential features of the currently implemented
system. For a start, we think that it is very important for the user of such a system
to state his or her problem in a simple way. To relieve the user from the burden
to write the input in a form that is directly understandable by the underlying
computer algebra system REDUCE we have defined an easily understandable input
format which is automatically translated into a sequence of REDUCE commands.
This preprocessing step performs already a number of consistency checks, such as
testing whether there is a differential equation for every dynamic variable, and so
on. A first version of this preprocessor was implemented in perl, due to the wide-
spread availability and excellent string manipulation facilities of this tool. A second
version is being implemented in REDUCE’s system programming environment
RLISP.

The underlying computer algebra system REDUCE performs the symbolic
step of the system: the problem at hand is analyzed and analytical computation is
tried as far as possible. The generation of the adjoint equations is rather simple,
since it involves only differentiation. Nevertheless it may lead to rather complicated