Hamburger Beiträge
zur Angewandten Mathematik

BNDSCO

A Program for the Numerical Solution of Optimal Control Problems

H.J. Oberle and W. Grimm

Report No. 515 der DFVLR
Deutsche Forschungs- und Versuchsanstalt
für Luft- und Raumfahrt e.V., 1989

Reihe B
Bericht 36
October 2001

Hamburger Beiträge zur Angewandten Mathematik

Reihe A	Preprints
Reihe B	Berichte
Reihe C	Mathematische Modelle und Simulation
Reihe D	Elektrische Netzwerke und Bauelemente
Reihe E	Scientific Computing
Reihe F	Computational Fluid Dynamics and Data Analysis

BNDSCO

A Program for the Numerical Solution of Optimal Control Problems

H.J. Oberle [1]
W. Grimm [2]

DEUTSCHE FORSCHUNGS-UND VERSUCHSANSTALT
FÜR LUFT-UND RAUMFAHRT E.V.

BNDSCO

A Program for the Numerical Solution of Optimal Control Problems

H.J. Oberle [1]
W. Grimm [2]

[1] Institute for Applied Mathematics, University of Hamburg
[2] Institute for Flight Systems Dynamics, German Aerospace Research Establishment DLR, Oberpfaffenhofen

Table of Contents

1. Introduction

The treatment of optimal control problems with state equality and/or inequality constraints leads to boundary value problems (BVP) with switching conditions. Subroutine BNDSCO developed by Oberle [8] on the basis of several earlier BVP solvers due to Bulirsch [4] generates numerical solutions of these problems. BNDSCO is an implementation of the multiple shooting algorithm (see Stoer, Bulirsch [11, 12]) adapted to the special problem mentioned above.

The documentation is structured as follows: Chapter 2 provides an overview over the most common optimal control problems and the necessary conditions associated with them. They are taken from three different sources: The book by Bryson, Ho [3], the paper by Bryson, Denham and Dreyfus [2] and the report of Bock [1]. The nomenclature used in chapter 2 is adapted from Bryson, Ho [3]. The description of BNDSCO itself is given in chapter 3. Section 3.1 describes the mathematical principles and procedures used in BNDSCO; Section 3.2 deals with the practical handling of the program. The description in chapter 3 is similar to the one given in the user manual by Oberle [8]. Chapter 4 explains the methodology to set up the two subroutines which need to be provided for BNDSCO. A numerical example is given in chapter 5. Users who are familiar with optimal control can restrict themselves to chapters 3 and 4. The minimum information necessary to handle BNDSCO is given in chapter 3.1.1, 3.2.1 – 3.2.4 and 4.

This manual is meant only for the version of BNDSCO as indicated in the appendix. Note that the whole source code is written in double precision. For the numerical example given in chapter 5 this version has been tested and run with the SIEMENS–compiler on an IBM 3090 at DLR, Oberpfaffenhofen.

BNDSCO has been applied successfully on many optimal control problems, mainly in the area of flight path optimization. A special feature of BNDSCO compared to other procedures is the high precision of the solutions. Many results generated with BNDSCO can be found in thesis works submitted at the Mathematical Institutes of the Technical University in Munich and the University of Cologne. An easily accessible example is the report of Oberle [9] which in particular describes an optimal control with singular arcs. A lot of optimal control problems have successfully been solved with BNDSCO at the Institute of Flight Systems Dynamics (part of the German Aerospace Research Establishment DLR), Oberpfaffenhofen. Meanwhile, users of BNDSCO and the predecessor routine OPTSOL (see Bulirsch [4]) can be found at research institutes all over the world.

2. The General Optimal Control Problem

2.1 The Unconstrained Optimal Control Problem

2.1.1 Problem Formulation

(P) is a general formulation of the unconstrained optimal control problem:

(P) Find the control function $u(t) \in \mathbb{R}^m$, $0 \le t \le T$, such that the cost function

$$J[u] := \phi(x(T), T)$$

is minimized subject to the following constraints:

a) The state vector $x(t) \in \mathbb{R}^n$ satisfies the state equations:

$$\dot{x}(t) = f(x(t), u(t)) \tag{1}$$

b) Some components of the state x are prescribed at initial time t = 0:

$$x_i(0) = x_{io} \ , \ i \in I \subset \{1,...n\} \tag{2}$$

c) At final time t = T x satisfies the boundary conditions:

$$\psi(x(T),T) = 0 \ , \text{ where } \psi \in \mathbb{R}^k \tag{3}$$

The final time T may be prescribed or free. In the latter case, the free final time T is also subject to optimization.

2.1.2 Two More General Problems and how to Reduce them to Problem 2.1.1

2.1.2.1 Problems of Bolza Type

Let $y \in \mathbb{R}^n$ denote the state vector and $\dot{y}(t) = g(y(t), u(t))$ the state equations. An optimal control problem is said to be of Bolza type if the cost function is of the form

$$J[u] = \Psi(y(T),T) + \int_0^T L(y(t), u(t))\, dt.$$

In the special case $L = 0$ or $\Psi = 0$ we talk about Mayer Problems or Lagrange Problems, respectively. By introducing an additional state variable, say $z(t)$, Bolza Problems can be reduced to Mayer Problems. With the definitions

$$x(t) := \begin{bmatrix} y(t) \\ z(t) \end{bmatrix}, \quad \dot{x}(t) = \begin{bmatrix} g(y(t), u(t)) \\ L(y(t), u(t)) \end{bmatrix} =: f(x(t), u(t)),$$

$$\phi(x(T),T) := \Psi(y(T), T) + z(T)$$

and the initial condition $z(0) = 0$ the standard form (P) is retrieved.

2.1.2.2 Non–Autonomous Systems

The state equations (1) in problem (P) are autonomous, i.e. they do not explicitly depend on t. Non–autonomous systems of the form

$$\frac{d}{d\theta}\, g\,(y\,(\theta)\,,\ v\,(\theta)\,,\ \theta) \quad , \quad \theta_o \le \theta \le \theta_f$$

(θ = independent variable, y = state vector, v = control) can be reduced to the special case of autonomous systems. We consider θ as additional state variable and $t := \theta - \theta_o$ as new independent variable. Defining

$$x(t) := \begin{bmatrix} y(\theta_o + t) \\ \theta_o + t \end{bmatrix} , \quad u(t) := v(\theta_o + t)$$

$$T := \theta_f - \theta_o , \quad f(x(t), u(t)) := \begin{bmatrix} g(y(\theta_o+t), u(t), \theta_o+t) \\ 1 \end{bmatrix}$$

we get

$$\dot{x}(t) = f(x(t), u(t)) , \quad 0 \le t \le T .$$

2.1.3 Necessary Conditions

The theory of optimal control provides necessary conditions for the trajectory $x(t)$ and the control function $u(t)$ associated with it. There are Lagrange multipliers

$$\lambda(t) = (\lambda_{x_1}(t), \ldots, \lambda_{x_n}(t))^T \in \mathbb{R}^n$$

and a (constant) vector $v \in \mathbb{R}^k$ such that the following conditions a) – d) are satisfied along an optimal trajectory $x(t)$, $u(t)$.
Let

$$H(x(t), \lambda(t), u(t)) := \lambda(t)^T f(x(t), u(t)) \tag{4}$$

denote the Hamiltonian.

a) $\lambda(t)$ satisfies the adjoint differential equations:

$$\dot{\lambda}(t) = -H_x^T(x(t), \lambda(t), u(t)) \tag{5}$$

b) $u(t)$ is determined by the Pontryagin Minimum Principle:

$$H(x(t), \lambda(t), u(t)) = \min_{v \in \mathbb{R}^m} H(x(t), \lambda(t), v)$$
$$\forall\, t \in [0,T] \tag{6}$$

This implies

$$H_u = 0 \tag{7}$$

and the Legendre–Clebsch condition demands that H_{uu} be a positive semi–definite matrix.

c) boundary conditions on λ:

$$\lambda_{x_i}(0) = 0 \quad \text{for } i \in K := \{1,\ldots,n\} \setminus I \tag{8}$$

$$\lambda^T(T) = \frac{\partial \phi(x(T),T)}{\partial x(T)} + \nu^T \frac{\partial \psi(x(T),T)}{\partial x(T)} \tag{9}$$

d) In case of free final time T we have

$$H(T) = -\frac{\partial \phi(x(T),T)}{\partial T} - \nu^T \frac{\partial \psi(x(T),T)}{\partial T} \; . \tag{10}$$

Remarks

1) The Hamiltonian H is called regular if the minimum with respect to the control vector is uniquely determined for all $(x,\lambda) \in \mathbb{R}^n \times \mathbb{R}^n$. In particular this implies that H depends nonlinearly on all components of u. If the Hamiltonian is regular the optimal control u(t) is a continuous function of time.

2) H is constant:

$$\begin{aligned} \frac{d}{dt} H(x(t), \lambda(t), u(t)) &= H_x \dot{x} + H_\lambda \dot{\lambda} + H_u \dot{u} \\ &= (-\dot{\lambda})^T \dot{x} + \dot{x}^T \dot{\lambda} + 0 \\ &= 0 \end{aligned}$$

Note that this property only holds for autonomous systems. For these systems it holds for whatever initial conditions are imposed on x and λ.

2.1.4 The Two–Point Boundary Value Problem (TPBVP) for the Optimal Trajectory

2.1.4.1 The TPBVP for Fixed Final Time

The set of boundary conditions (1) – (3) in (P) and the necessary conditions 2.1.3. a) – 2.1.3 d) lead to a TPBVP for the vector function

$$y(t) := \begin{bmatrix} x(t) \\ \lambda(t) \\ \nu \end{bmatrix} \in \mathbb{R}^{2n+k} \quad .$$

Its first time derivative is given by

$$\dot{y}(t) := \begin{bmatrix} f(x(t), u(t)) \\ -H_x^T(x(t), \lambda(t), u(t)) \\ O \end{bmatrix} =: g(y(t)) \quad .$$

The vector function $r: \mathbb{R}^{2n+k} \times \mathbb{R}^{2n+k} \to \mathbb{R}^{2n+k}$ represents the boundary conditions. Here let x_I denote the vector of x–components with indices in I and x_{I0} the vector of prescribed initial values x_{i0}. And finally let λ_K denote the vector of those λ_{x_i} with $i \in K$. Then

$$r(y(0), y(T)) = \begin{bmatrix} x_I(0) - x_{I0} \\ \lambda_K(0) \\ \\ \psi(x(T), T) \\ (\lambda - \phi_x^T - \psi_x^T \nu)_{t=T} \end{bmatrix} = 0 \quad .$$

The extremal $y(t)$ is a solution of the TPBVP

$$\boxed{\begin{array}{c} \dot{y}(t) = g(y(t)), \; 0 \le t \le T \\ \\ r(y(0), y(T)) = 0 \end{array}}$$

Here the control variables u(t) in g(y) are determined by the Minimum Principle 2.1.3.b. In so far u is a function of x and λ:

$$u = u(y)$$

In particular, u is a solution of the equation system

$$H_u(x(t), \lambda(t), u(t)) = 0 \quad .$$

If this equation cannot be solved in closed form a Newton method has to be applied in each integration step to evaluate g(y) (see 4.1.3). Whenever possible the conditions

$$(\lambda - \phi_x^T - \psi_x^T \nu)_{t=T} = 0$$

should be used to eliminate some or all of the ν_i. These ν_i's and the conditions used to eliminate them can be cancelled in the TPBVP. In the most favourable case we end up with n–k equations independent of ν.

2.1.4.2 The TPBVP with Free Final Time T

To incorporate the free final time into the optimization we apply the transformation

$$\tau := \frac{t}{T} \quad , \quad \tau \in [0,1].$$

At the same time T will be considered as an additional state variable. Clearly, the free final time T will be added as an additional component in the vector function y(.) and the independent variable will be τ rather than t.

$$y(\tau) := \begin{bmatrix} x(T \cdot \tau) \\ \lambda(T \cdot \tau) \\ \nu \\ T \end{bmatrix} \in \mathbb{R}^{2n+k+1}$$

By the chain rule we have

$$\frac{dy}{d\tau} = \begin{bmatrix} f(x(T\cdot\tau),\ u(T\cdot\tau)) \\ -H_x^T\,(x(T\cdot\tau),\ \lambda(T\cdot\tau),\ u(T\cdot\tau)) \\ 0 \\ 0 \end{bmatrix} \cdot T \ =: \ g(y(\tau)) \quad .$$

The boundary conditions are

$$r(y(0),y(1)) = \begin{bmatrix} x_I(0) - x_{I0} \\ \lambda_K(0) \\ \psi(x(T),\ T) \\ (\lambda - \phi_x^T - \psi_x^T\,\nu)_{\tau=1} \\ (H + \phi_T + \psi_T^T\,\nu)_{\tau=1} \end{bmatrix} = 0 \quad .$$

The extremal is the solution of the TPBVP

$$\boxed{\begin{array}{c} \frac{dy}{d\tau} = g(y(\tau)),\ 0 \le \tau \le 1 \\ \\ r(y(0),\ y(1)) \ = 0 \end{array}} \qquad (11)$$

2.1.5 The Special Case of Prescribed States at Final Time

We consider the special case where certain x_i, $i \in M \subset \{1,...,n\}$, are prescribed at final time:

$$x_i(T) = x_{iT} \quad \text{for } i \in M \ .$$

Then ψ is simply

$$\psi\,(x(T),T) \ = \ x_M(T) - x_{MT} \quad .$$

Here x_M denotes the vector of states x_i, $i \in M$, x_{MT} denotes the vector of the prescribed final values associated with x_M. In this case the parameter vector ν can be eliminated. The boundary conditions on the adjoints λ_{x_i}, $i \in N := \{1,...,n\} \setminus M$, are

$$\left[\lambda_N - \phi^T_{x_N}\right]_{t=T} = 0 \quad .$$

λ_N and x_N denote the vectors whose components are λ_{x_i} and x_i with $i \in N$, respectively. In case of free final time T, H is determined by

$$(H + \phi_T)_{t=T} = 0 \quad .$$

2.2 The Constrained Optimal Control Problem

In this section we consider problem (P) augmented by a condition of the form

$$S\left[x(t), u(t)\right] \leq 0 \quad \forall\ t \in [0, T] \ .$$

2.2.1 – 2.2.6 deal with the case of a pure state constraint, i.e. S does not explicitly depend on u. In 2.2.7 and 2.2.8 the case $S_u \not\equiv 0$ is added.

2.2.1 The State Constraint and the Order of the State Constraint

Let us consider the unconstrained optimization problem (P) with conditions (1), (2), (3) as stated in 2.1.1. Now assume an additional constraint is given of the form

$$S(x(t)) \leq 0 \quad \forall\ t \in [0,T] \ . \tag{12}$$

Note that S does not explicitly depend on u. The case where u explicitly appears in S will be discussed in 2.2.7. Without loss of generality it is assumed that S does not explicitly depend on t (otherwise transform the problem as described in 2.1.2.2).

For the treatment of the problem the order q of the constraint (12) is crucial. It is defined as follows: q is the smallest integer i such that $d^i/dt^i\, S(x(t))$ is explicitly dependent on u. Here state derivatives are always eliminated by use of equation (1). Hence, formally the following recursion determines q:

$\underline{i = 0:}$ $\qquad S^{(o)}(x(t)\,,\, u(t)) := S(x(t))$

$\underline{i-1 \to i:}$ $\qquad S^{(i)}(x(t)\,,\, u(t)) := S_x^{(i-1)}(x(t)\,,\, u(t))\;\; f(x(t)\,,\, u(t))$

If $S_u^{(i)} \not\equiv 0$ then stop and set $q := i$.

By assumption $S^{(i)}$ is independent of u for $0 \le i < q$:

$$S^{(i)}(x,u) \;=\; S^{(i)}(x)$$

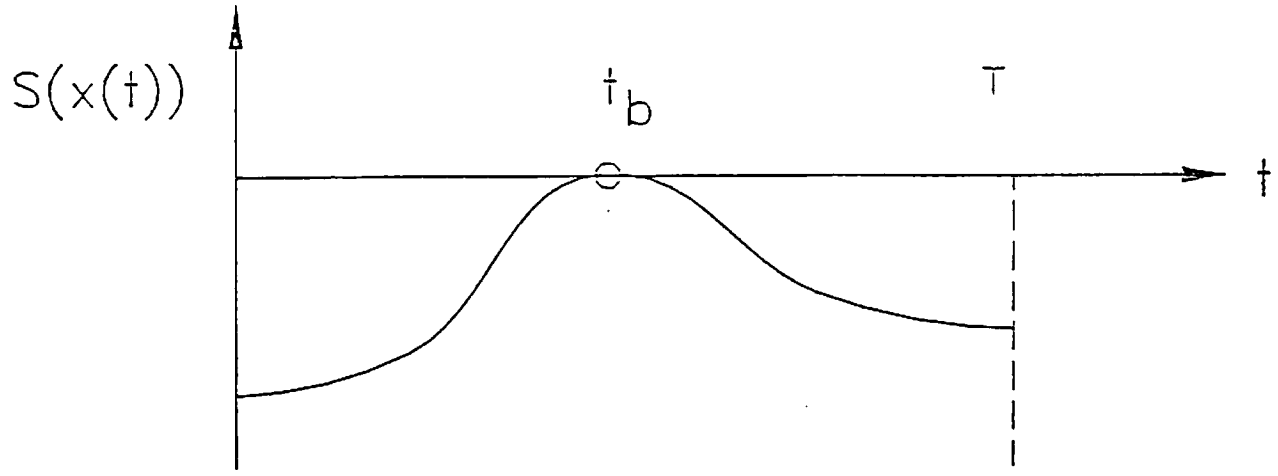
Define

$$N(x(t)) := \begin{bmatrix} S^{(o)}(x(t)) \\ \vdots \\ S^{(q-1)}(x(t)) \end{bmatrix} \in \mathbb{R}^q \quad . \tag{13}$$

2.2.2 Contact Points and Constrained Arcs

State constraint (12) can be active pointwise and/or on whole intervals.

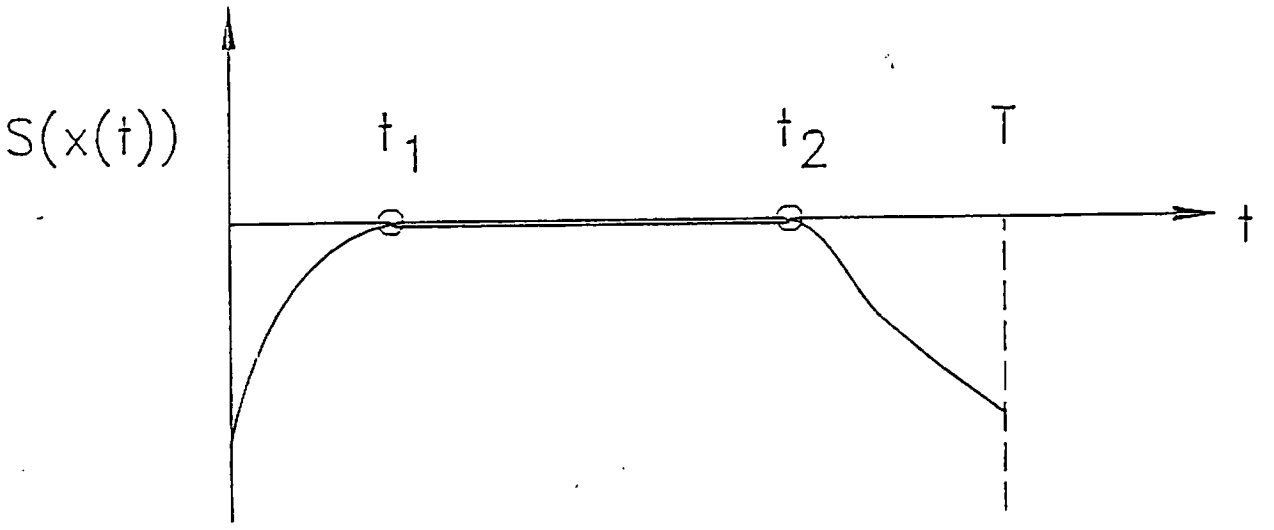
a) An isolated zero $t_b \in [0,T]$ of $S(x(t))$ is called a "contact point". If additionally $S^{(1)}(x(t))$ is 0 at t_b then t_b is called a "touch point". In this case t_b is a local maximum of $S(x(t))$.



b) An interval $[t_1, t_2] \subset [0,T]$, $t_1 < t_2$, is called a constrained arc if

$S(x(t)) = 0$ on $t_1 \leq t \leq t_2$ and

$S(x(t)) < 0$ in some neigbourhoods left of t_1 and right of t_2 , respectively.



$[0, T]$ is partitioned in constrained and unconstrained subarcs by junction points like t_1 and t_2. The sequence of subarcs constituting the extremal is called "switching structure".

2.2.3 Necessary Conditions along an Optimal Trajectory

Assume the state constraint (12) is of order q, i.e. $S(x(t)) \equiv 0$ on $t \in [t_1, t_2]$ is equivalent to $S^{(o)}(x(t)) = 0,\ldots, S^{(q-1)}(x(t)) = 0$, $S^{(q)}(x(t), u(t)) \equiv 0$ on $t \in [t_1, t_2]$. Let the Hamiltonian be defined as before:

$$H(x, \lambda, u) := \lambda^T f(x, u) \tag{14}$$

and let the Lagrange multipliers be the solutions of the adjoint equations

$$\dot{\lambda} = -H_x^T (x, \lambda, u) - \mu\ S_x^{(q)T}(x, u) \quad . \tag{15}$$

Here μ is a multiplier to link condition $S^{(q)}(x, u) \equiv 0$ to the Hamiltonian. The Minimum Principle is restricted to control vectors u satisfying $S^{(q)}(x, u) = 0$:

$$u(t) = \arg\min_{\substack{v \in \mathbb{R}^m \\ S^{(q)}(x(t),v) = 0}} H\ (x(t), \lambda(t), v) \tag{16}$$

A solution of (16) satisfies the first order conditions

$$H_u \left[x(t), \lambda(t), u(t)\right] + \mu\ S_u^{(q)} \left[x(t), u(t)\right] = 0 \tag{17}$$

$$S^{(q)}\left[x(t), u(t)\right] = 0$$

and the second order condition

$$\delta u^T \left[H_{uu} \left[x(t), \lambda(t), u(t)\right] + \mu\ S_{uu}^{(q)} \left[x(t), \lambda(t), u(t)\right]\right] \delta u \geq 0$$
$$\forall\ \delta u \in \mathbb{R}^m \text{ satisfying } S_u^{(q)}\ (x(t), u(t))\ \delta u = 0\ . \tag{18}$$

In general these equations determine u(t) and $\mu(t)$ uniquely for given states x(t) and costates $\lambda(t)$, and the trajectory can be integrated.

2.2.4 Necessary Conditions for Unconstrained Arcs and Touch Points

At switching point t_1, the beginning of the constrained arc, we have

$$N(x(t_1)) = 0 . \tag{19}$$

The multiplier vector λ (see Equ. (13) for the definition of N) is continuous at t_2, but discontinuous at t_1 :

$$\lambda^T(t_1^+) = \lambda^T(t_1^-) - \sigma^T N_x(x(t_1)) \tag{20}$$

where

$$\sigma_i \geq 0 \quad , \quad i = 0 ,..., q-1 \tag{21}$$

are additional constant multipliers. At both switching points, t_1 and t_2, the Hamiltonian is continuous:

$$H(t_i^+) = H(t_i^-) \ , \ i = 1,2 . \tag{22}$$

The multiplier $\mu(t)$ is q times continuously differentiable on (t_1,t_2), and with $\mu_i(t) := (-d/dt)^i \mu(t)$ we have

$$\mu_i(t) \geq 0 \quad \text{on } (t_1 , t_2) \ , \ i = 0,..., q , \tag{23}$$

$$\mu_i(t_1^+) = \sigma_{q-1-i} \ , \ \mu_i(t_2^-) = 0 , \tag{24}$$

for $q \geq 2$ and $i = 0, \ldots, q-2$

and

$$\mu_{q-1}(t_1^+) \geq \sigma_o \ , \ \mu_{q-1}(t_2^-) \geq 0 \ . \tag{25}$$

If $q = 1$ "=" instead of "≥" is necessary in (25).

A touch point t_b is determined by the conditions

$$S^{(i)}(x(t_b) \ , \ u(t_b)) \ = \ 0 \ , \ i = 0, \ i \ , \tag{26}$$

$$H(t_b^+) \ = \ H(t_b^-) \ , \tag{27}$$

$$\lambda^T(t_b^+) \ = \ \lambda^T(t_b^-) - l_o \ S_x(x(t_b)) \ \text{ with } l_o \ \geq \ 0 \ . \tag{28}$$

Condition (22) can be examined without explicitly taking into account the jump conditions (20). We have

$$H(t_1^+) - H(t_1^-) \ = \ \lambda(t_1^+)^T \ f\left[x(t_1) \ , \ u(t_1^+)\right] \ - \ \lambda(t_1^-)^T \ f\left[x(t_1) \ , \ u(t_1^-)\right]$$

$$= \left[\lambda(t_1^-)^T \ - \ \sigma^T N_x\left[x(t_1)\right]\right] \ f\left[x(t_1) \ , \ u(t_1^+)\right] \ - \ \lambda(t_1^-)^T \ f\left[x(t_1) \ , \ u(t_1^-)\right]$$

$$= \lambda(t_1^-)^T \ \left[f\left[x(t_1) \ , \ u(t_1^+)\right] \ - f\left[x(t_1) \ , \ u(t_1^-)\right]\right] \ .$$

This holds true as along the optimal trajectory

$$N_x\left[x(t_1)\right] \ f\left[x(t_1) \ , \ u(t_1^+)\right] \ = \begin{bmatrix} S^{(1)}|_{t=t_1^+} \\ \vdots \\ S^{(q)}|_{t=t_1^+} \end{bmatrix} \ = \ 0 \ .$$

Here $u(t_1^-)$ is the unconstrained control (solution of equ. (6) with $\lambda(t_1^-)$) and $u(t_1^+)$ is the constrained control (solution of equ. (16) with $\lambda(t_1^+)$) at time t_1. If H is regular, then u(t) is continuous. In this case, condition (22) is equivalent to $u(t_1^+) = u(t_1^-)$. The same remarks hold for t_2.

2.2.5 The Existence of Constrained Arcs and Touch Points in Dependence of the Order of the Constraint

Not for all positive integers q both constrained arcs and touch points can occur. E.g. constraints of order q = 1 do not become active in form of touch points. On the other hand, constrained arcs are impossible for q = 3, 5, 7,.... A summary of these results is given in the following table.

q	touch points possible	constrained arcs possible
1	no	yes
2,4,6,...	yes	yes
3,5,7,...	yes	no

2.2.6 The MPBVP for the Case of a Constrained Arc

By the necessary conditions stated in 2.2.3 and 2.2.4 the extremal of a constrained optimal control problem is characterized as the solution of a multipoint boundary value problem (MPBVP).

Generally the construction of the MPBVP depends on the switching structure associated with the problem under consideration. As an example, the MPBVP will be given here for the case of a single constrained arc. The construction in case of other switching structures is similar. The dependent variables are states x, adjoints λ, parameters ν and σ, and the free final time T. To allow integration within fixed bounds, the time interval [0,T]

will be normalized to $[0,1]$ as follows:

$$y(\tau) := \begin{bmatrix} x(T\tau) \\ \lambda(T\tau) \\ \nu \\ \sigma \\ T \end{bmatrix} \quad , 0 \le \tau \le 1$$

$$\frac{dy}{d\tau} = \begin{bmatrix} f\left[x(T\tau),\ u(T\tau)\right] \\ -H_x^T\left[x(T\tau),\ \lambda(T\tau),\ u(T\tau)\right] - \mu\, S_x^{(q)T}\left[x(T\tau),\ u(T\tau)\right] \\ 0 \\ 0 \\ 0 \end{bmatrix} \cdot T =: g\left[y(\tau)\right]$$

$$r\left[y(0),y(1)\right] = \begin{bmatrix} x_I(0) - x_{I0} \\ \lambda_K(0) \\ \psi(x(T),T) \\ (\lambda - \phi_x^T - \psi_x^T \nu)_{\tau=1} \\ (H + \phi_T + \psi_T^T \nu)_{\tau=1} \end{bmatrix} = 0 .$$

The switching points t_1 , t_2 are transformed into

$$\tau_i := \frac{t_i}{T} \quad , \quad i = 1, 2 .$$

According to 2.2.4 the switching conditions at τ_1 are

$$0 = S_1\left[y(\tau_1)\right] := \begin{bmatrix} N\left[x(T\tau_1)\right] \\ H\big|_{\tau=\tau_1^+} - H\big|_{\tau=\tau_1^-} \end{bmatrix} \in \mathbb{R}^{q+1} .$$

Furthermore the jump conditions are

$$0 = h\left[y(\tau_1)\right] := \lambda^T(T\,\tau_1^+) - \lambda^T(T\,\tau_1^-) + \sigma^T N_x\left[x(T\,\tau_1)\right] \quad .$$

At switching point τ_2 we have

$$0 = S_2\left[y(\tau_2)\right] := H|_{\tau = \tau_2^+} - H|_{\tau = \tau_2^-} \quad .$$

Together we get the following MPBVP:

$\frac{dy}{d\tau} = g(y) \quad , \quad \tau \in [0,\ 1]$	
boundary conditions	$r(y(0)\ ,\ y(1)) = 0$
switching conditions	$S_1(y(\tau_1)) = S_2(y(\tau_2)) = 0$
jump conditions	$h(y(\tau_1)) = 0$

These are $2n + k + q + 3$ conditions for $2n + k + q + 3$ unknowns, namely, y, τ_1, τ_2. Control u and multiplier μ are determined by

$$\begin{cases} \left.\begin{aligned} \lambda^T f_u(x,u) &= 0 \\ \mu &= 0 \end{aligned}\right\} \begin{array}{l}\text{along unconstrained arcs} \\ 0 \le \tau < \tau_1 \ , \ \tau_2 < \tau \le 1\end{array} \\ \left.\begin{aligned} \lambda^T f_u(x,u) + \mu\, S_u^{(q)}(x,u) &= 0 \\ S^{(q)}(x,u) &= 0 \end{aligned}\right\} \begin{array}{l}\text{along the constrained} \\ \text{arc } \tau_1 \le \tau \le \tau_2\end{array} \end{cases}$$

Remarks

(1) If the final time T is fixed and prescribed in advance then the last differential equation and the last boundary condition is dropped. Also the transformation $\tau = t/T$ is no longer necessary then.

(2) The dimension of y should be reduced as much as possible by eliminating components of parameter vector ν (see remark in 2.1.4). If all final states are prescribed explicitly then ν is of dimension zero (see 2.1.5).

(3) By virtue of equation (24) condition $H|_{\tau = \tau_2^+} = H|_{\tau = \tau_2^-}$ can be replaced by

$$\mu\,(\tau = \tau_2^-) \;=\; 0 \;\text{ if } q \;\geq\; 2\,.$$

(4) In case of a touch point t_b we only have the conditions

$$S(x(t_b)) \;=\; 0\;,\; H|_{t = t_b^+} \;=\; H|_{t = t_b^-}\;.$$

The condition $S^{(1)}(x(t_b)) \;=\; 0$ will be satisfied automatically if the assumed switching structure is correct. In contrary to the case of a constrained arc it is always necessary to perform the jumps in the multipliers λ (equ. (28)) before evaluating $H|_{t = t_b^+}$.

(5) By integrating the trajectory backwards from t = T to t = 0 the meaning of t_1 and t_2 as beginning and end of the constrained arc is interchanged. Applying the necessary conditions as stated in 2.2.3, 2.2.4 to this trajectory may yield additional information. It is possible that a trajectory satisfies all necessary conditions when integrating forward while violating an optimality condition when the direction of integration is reversed. An example can be found in Hiltmann [7].

2.2.7 Constraint Explicitly Dependent on Controls

The constraint

$$S\left[x(t), u(t)\right] \leq 0 \quad \text{with} \quad S_u \not\equiv 0 \tag{29}$$

can be regarded as special case of a state constraint namely of order 0. Accordingly all conditions stated in 2.2.3 hold with $q = 0$ and $S^0 = S$. Generally, active constraints of the form (29) do not yield touch points. Differences with respect to the case $q \geq 1$ arise in the treatment of constrained arcs. The Lagrange multipliers are continuous at the switching points t_1, t_2. Hence there are no parameters σ. The only switching condition at t_i is the continuity of the Hamiltonian:

$$H(t_i^+) = H(t_i^-)$$

2.2.8 Control Constraints of the Form $u_1(t) \geq u_{1max}$

As a special case of 2.2.7 let us consider the control constraint

$$S(x(t), u(t)) = u_1(t) - u_{1max} \leq 0 \tag{30}$$

Here u_1 and u_{1max} denote the first component of u and a constant upper bound on u_1, respectively. As a result of the simple structure of constraint (30) the multiplier μ can be eliminated. The Hamiltonian is as before:

$$H(x, \lambda, u) := \lambda^T f(x, u)$$

Again control u is determined from the Minimum Principle

$$u(t) = \arg \min_{\substack{v \in \mathbb{R}^m \\ v_1 \leq u_{1max}}} H(x(t), \lambda(t), v) \tag{31}$$

where the multipliers $\lambda(t)$ are solutions of the adjoint differential equations

$$\dot{\lambda} = -H_x^T \quad .$$

At $t = 0$ and $t = T$ the transversality conditions (8), (9) and (10) as stated in 2.1.3 are valid. The structure of the optimal control $u(t)$ determined by the Minimum Principle (31) depends on whether or not (31) has a unique solution for all (x,λ) ("regularity of the Hamiltonian").

2.2.8.1 Regular Hamiltonian

A regular Hamiltonian always depends nonlinearly on u. The optimal control is continuous. Along an unconstrained arc the optimal control (denoted by u^{free}) is in the interior of the admissible domain $u_1 \leq u_{1max}$ and hence is solution of

$$0 = H_u (x, \lambda, u^{free}) \quad .$$

Along a constrained arc the optimal control furnishes the constrained minimum of H:

$$u^{constrained} = \arg \min_{\substack{v \in \mathbb{R}^m \\ v_1 = u_{1max}}} H \ (x(t), \lambda(t), v)$$

In practice the unconstrained minimum u^{free} of the Hamiltonian represents the optimal control as long as $u^{free} \leq u_{1max}$.

2.2.8.2 Linear Control

The linear control is the most common case of a nonregular Hamiltonian. Consider the following structure of H:

$$H(x, \lambda, u) = h_o(x, \lambda, u_2,..., u_m) + u_1 \cdot h_1 (x, \lambda, u_2,..., u_m) \quad . \tag{32}$$

To guarantee a solution of the Minimum Principle (31), (32) only makes sense with u_1 restricted to a compact interval:

$$u_{1min} \leq u_1 \leq u_{1max}$$

the Minimum Principle takes the form

$$u(t) = \arg \min_{\substack{v \in \mathbb{R}^m \\ u_{1min} \leq v_1 \leq u_{1max}}} H(x(t), \lambda(t), v) \tag{33}$$

If $h_1 \neq 0$ holds for the solution of (33), the optimal value of u_1 must be one of the bounds:

$$u_1 = \begin{cases} u_{1min}, & \text{if } h_1 > 0 \\ u_{1max}, & \text{if } h_1 < 0 \end{cases}$$

Extremals associated with a Hamiltonian like (32) are often completely made up of constrained arcs for u_1. Such a sequence of subarcs alternating between u_{1min} and u_{1max} is called "bang–bang–control". The switching points are determined as zeros of h_1 which is called "switching function" in this context.
Moreover, subarcs with $h_1 \equiv 0$ can occur. Along such an interval u_1 is called "singular control" and generally takes values in the interior of $[u_{1min}, u_{1max}]$. Note that $H_{u_1} = h_1$ does not explicitly depend on u_1. A representation of u_1 as a function of x and λ requires at least twice differentiation of $h_1 \equiv 0$. For further information see e.g. Oberle [9]. This report also includes the solution of a singular control problem with BNDSCO.

3. The Multiple Shooting Method and its Implementation in Program BNDSCO

3.1 General Description of the Method

3.1.1 The Boundary Value Problem with Switching Conditons

The TPBVP 2.1.4 and the MPBVP 2.2.6 belong to the general class of boundary value problems with switching conditions. This is the most general kind of BVP that can be solved with BNDSCO. Find:

a) a piecewise smooth function $y(x) \in \mathbb{R}^n$, $x_1 \le x \le x_f$, and

b) switching points ξ_i , i = 1,..., s, where $x_1 =: \xi_0 < \xi_1 < ... < \xi_s < \xi_{s+1} := x_f$ with the following properties:

1) $y'(x) = f_k(x, y(x))$, $\xi_k \le x \le \xi_{k+1}$, $0 \le k \le s$.

2) $y(\xi_k^+) = h_k(\xi_k, y(\xi_k^-))$ for $k = 1,..., s$.

3) $r_i(y(x_1), y(x_f)) = 0$ for $1 \le i \le n_1$.

4) $r_i(\xi_{k_i}, y(\xi_{k_i}^-)) = 0$ for $i = n_1 + 1,..., n + s$.

Condition 1) represents a piecewise defined system of ODEs on y. In the case of an underlying optimal control problem, the f_k represent the state equations with different types of optimal control.

Equation 2) describes a jump condition at switching point ξ_k . In the framework of an optimal control problem h_k could be the jump condition (20) at the beginning of a state constrained subarc.

Equations 3) are the usual two point boundary conditions.

Equations 4) stand for switching conditions at point ξ_{k_i} . The index k_i indicates that one, more than one, or possibly no switching condition can be associated with a single switching point.

Important: The switching structure, that means the number of switching points and the sequence of different right–hand sides f_k in 1) has to be prescribed by the user.

3.1.2 The Multiple Shooting Method for Solving Boundary Value Problems with Switching Conditions

Starting point for the Multiple Shooting Method is the choice of a fixed subdivision of the time interval

$$x_1 < x_2 <...< x_m := x_f$$

where $x_k \neq \xi_i$ for $k = 1,..., m$, $i = 1,..., s$, and the guessing of initial data:

$$Y_j^{(o)} \quad : \text{guess for } y(x_j) \qquad j = 1,..., m-1$$
$$\xi_k^{(o)} \quad : \text{guess for } \xi_k \qquad k = 1,..., s$$

These data will be iteratively changed into a solution of 1) – 4) in 3.1.1. In the following description of a single iteration the superscript for iteration count will be omitted. Define

$$z(x) := \begin{bmatrix} y(x) \\ \xi \end{bmatrix}, \qquad \xi := \left[\xi_1 ,..., \xi_s\right]^T ,$$

$$Z_j := \begin{bmatrix} Y_j \\ \xi \end{bmatrix} \quad \text{for} \quad j := 1,..., m-1 .$$

Step 1:

Find the numerical solution of the initial value problems

$$z'(x) = \begin{bmatrix} f(x, y(x)) \\ 0 \end{bmatrix}, \qquad x_j \le x \le x_{j+1} , \; z(x_j) = Z_j$$

for $j = 1,..., m-1$. Along the process also the jump conditions 2) have to be executed at switching points $\xi_k \in [x_j , x_{j+1}]$. Note also that the right–hand side f changes with k.

Let

$$z(x; x_j, Z_j) = \begin{bmatrix} y(x; x_j, Z_j) \\ \xi \end{bmatrix}$$

denote the solution in the interval $\left[x_j, x_{j+1}\right]$.

Step 2:

a) Compute the jumps

$$F_j(Z_1, \ldots, Z_{m-1}) := z(x_{j+1}; x_j, Z_j) - Z_{j+1} \in \mathbb{R}^{n+s}$$

for $1 \le j \le m-2$.

b) Evaluate the boundary and switching conditions

$$F_{m-1}(Z_1, \ldots, Z_{m-1}) := \begin{bmatrix} R(Z_1, Z_{m-1}) \\ S(Z_1, \ldots, Z_{m-1}) \end{bmatrix} \in \mathbb{R}^{n+s}$$

where

$$R(Z_1, Z_{m-1}) := \left[r_i(Y_1, y(x_m; x_{m-1}, Z_{m-1}))\right]_{i=1,\ldots,n_1} \in \mathbb{R}^{n_1} ,$$

$$S(Z_1, \ldots, Z_{m-1}) := \left[r_i(\xi_{k_i}, y(\xi_{k_i}^-; x_{s_i}, Z_{s_i}))\right]_{i=n_1+1,\ldots,n+s} \in \mathbb{R}^{n+s-n_1} .$$

The index s_i is defined by

$$x_{s_i} < \xi_{k_i} < x_{s_i+1} .$$

Also let $F(Z_1, \ldots, Z_{m-1}) = (F_1^T, \ldots, F_{m-1}^T)^T \in \mathbb{R}^{(m-1)(n+s)}$. A trajectory $y(x)$, and the switching points associated with it are solution of problem 3.1.1 if and only if vector $Z := (Z_1, \ldots, Z_{m-1})$ is a zero of F. Program BNDSCO is basically an implementation of

the Modified Newton Method to determine this zero. Steps 1), 2) serve for evaluating function F. Steps 3), 4), 5) determine the Newton correction.

Step 3:

Numerical approximation of the Jacobian matrix M of F (Z) via numerical differentiation or via an appropriate Broyden update of M as obtained in a previous iteration. For details see 3.1.3 and 3.1.4.

Step 4:

Compute Newton correction ΔZ from $M \cdot \Delta Z = -F(Z)$. For details see 3.1.5 .

Step 5:

Determine an appropriate relaxation factor $\lambda \in [0, 1]$ and compute

$$Z^{(new)} = Z + \lambda \cdot \Delta Z \ .$$

For details see 3.1.6.

3.1.3 The Structure of the Multiple Shooting Matrix

As block matrix of format (m−1, m−1) M is given by

$$M = \begin{bmatrix} G_1 & -I & & \\ & G_2 & -I & \\ & & \ddots & \ddots \\ & & G_{m-2} & -I \\ A_1 & \cdots & A_{m-2} & A_{m-1} \end{bmatrix} .$$

Here G_i and A_i are square matrices of dimension n+s. Explicitly

$$G_j := \frac{\partial}{\partial Z_j} z(x_{j+1} ; x_j , Z_j) \quad \text{for } j = 1,..., m-2 \quad ,$$

$$A_1 := \begin{bmatrix} \frac{\partial}{\partial Z_1} R(Z_1 , Z_{m-1}) \\ \frac{\partial}{\partial Z_1} S(Z_1 ,\ldots , Z_{m-1}) \end{bmatrix} ,$$

$$A_j := \begin{bmatrix} 0 \\ \frac{\partial}{\partial Z_j} S(Z_1 ,..., Z_{m-1}) \end{bmatrix} \quad \text{for } j = 2 ,..., m-2 \quad ,$$

$$A_{m-1} := \begin{bmatrix} \frac{\partial}{\partial Z_{m-1}} R(Z_1 , Z_{m-1}) \\ \frac{\partial}{\partial Z_{m-1}} S(Z_1 ,\ldots , Z_{m-1}) \end{bmatrix} .$$

3.1.4 Computation of the Multiple Shooting Matrix

Before each Newton iteration the program assigns scaling factors $w_j = (w_{1j} ,..., w_{n+s,j})^T \in \mathbb{R}^{n+s}$. Basically

$$w_j = \frac{1}{2} (|Z_j| + |Z_j^{old}|) \quad \text{for } 1 \le j \le m-1$$

where $|\cdot|$ is understood componentwise, and "old" denotes the second last iteration.

$$W_j := \text{diag}(w_{1j} ,..., w_{n+s,j}) \in \mathbb{R}^{(n+s,\ n+s)} \quad .$$

3.1.4.1 Numerical Differentiation

The differential quotients in 3.1.3 are replaced by difference quotients with an increment δZ_{ij} for the i–th component of the vector Z_j. This increment is proportional to ETA which is an input parameter in the parameter list of BNDSCO. Except for some special cases δZ_{ij} is computed as follows:

a) $i \le n$: Z_{ij} is a y–component .
$\delta Z_{ij} := \text{sign}\,(Z_{ij}) \cdot \eta \cdot w_{ij}$.

b) $i > n$: Z_{ij} is a switching point .
$\delta Z_{ij} := \eta\, Z_{ij}$.

The computational effort for the calculation of difference quotients is considerable as each function evaluation $z\,(x\,;\,x_j\,,\,Z_j + \delta\, Z_{ij}\, e_i\,)$ requires solving an additional initial value problem ($e_i \in \mathbb{R}^{n+s}$ is the i–th canonical unit vector).

3.1.4.2 Broyden Approximation

During the k–th iteration this method replaces the submatrix $A_j^{(k)}$ (or $G_j^{(k)}$) of the multiple shooting matrix by the sum of $A_j^{(k-1)}$ (or $G_j^{(k-1)}$) and an appropriate rank–1 matrix. A general description of this update formula can be found in Stoer [10], pp. 239–242. The Broyden Approximation will be applied only during the "convergence phase" (see 3.1.6).

3.1.5 Solving the Linear System $M \cdot \Delta Z = -F(Z)$

First the system will be rescaled in a numerically reasonable way. The multiple shooting matrix M will be replaced by

$$M^* = M \cdot \begin{bmatrix} W_1 & & 0 \\ & \ddots & \\ 0 & & W_{m-1} \end{bmatrix}$$

(for the definition of W_i see 3.1.4).

Instead of the original correction step $(\Delta Z_1, \ldots, \Delta Z_{m-1})$ this yields the scaled correction step

$$\Delta Z_w := (W_1^{-1} \Delta Z_1, \ldots, W_{m-1}^{-1} \Delta Z_{m-1}) .$$

By a sequence of Householder transformations M^* is brought to upper triangular form and we get the decomposition

$$M^* = Q R ,$$

where R is a right upper triangular matrix and Q is a unitary matrix (Q is the product of all Householder transformations mentioned above). With this decomposition the computation of $\Delta Z_w = - M^{*-1} F(Z) = - R^{-1} Q^T F(Z)$ basically reduces to a backward substitution. A general description of this procedure can be found in Stoer [10], pp. 164–170. For the remainder let r_{ii} be the diagonal elements of R. Following Deuflhard [5] we have

$$\rho := \max_i |r_{ii}| \le \text{lub}_2 M^* := \max_{x \neq 0} \frac{\|M^* x\|_2}{\| x \|_2} ,$$

$$\sigma := \frac{\max |r_{ii}|}{\min |r_{ii}|} \le \text{cond}_2 M^* := (\text{lub}_2 M^*) (\text{lub}_2 M^{*-1}) .$$

ρ and σ serve as estimates on the magnitude of the norm and condition number of M^*, respectively. In case $1/\sigma$ is less than the machine precision M^* will be regarded as singular and the calculation stops.

3.1.6 The λ–Strategy

The procedure to determine the relaxation factor $\lambda \in [0, 1]$ is based on a work by Deuflhard [6]. The variables crucial for the choice of λ_k in the k–th iteration are $\Delta Z_w^{(k)}$, and the number

$$\overline{\Delta Z_w^{(k)}} := -(M^{*(k-1)})^{-1} F(Z^{(k)})$$

obtained from a "simplified" Newton step.
With

$$\mu_k := \frac{\|\Delta Z_w^{(k)}\|_2}{\|\Delta Z_w^{(k)} - \overline{\Delta Z_w^{(k)}}\|_2} \lambda_{k-1}$$

we set

$$\lambda_k := \begin{cases} 1 & \text{if } \mu_k \geq 0.7 \\ \mu_k & \text{if } \mu_k < 0.7 \end{cases}$$

In a first attempt $Z^{(k+1)} = Z^{(k)} + \lambda_k \Delta Z^{(k)}$, $\overline{\Delta Z_w^{(k+1)}} = -(M^{*(k)})^{-1} F(Z^{(k+1)})$ and $F(Z^{(k+1)})$ will be computed. Then three test functions $T_i(F(Z), \Delta Z_w)$, i = 1, 2, 3, determine whether relaxation factor λ_k is accepted ("monotonicity tests"): If

$$T_i\left[F(Z^{(k+1)}), \overline{\Delta Z_w^{(k+1)}}\right] \leq T_i\left[F(Z^{(k)}), \Delta Z_w^{(k)}\right]$$

for some $i \in \{1, 2, 3\}$ then λ_k is accepted. Otherwise λ_k is decreased until one of these inequalities is satisfied.
The Broyden approximation of the multiple shooting method is applied only if all three monotonicity tests are positive. Explicitly the three test functions ("level functions") are

Level 1: "relative error"

$$T_1\left[F(Z), \Delta Z_w\right] := \|F_{m-1}(Z)\|_2^2 + \sum_{j=1}^{m-2} \|W_j^{-1} F_j(Z)\|_2^2 \quad .$$

So the jumps at nodes are weighted while the deviations in boundary and switching conditions are not.

Level 2: "mixed test function"

$$T_2\left[F(Z), \Delta Z_w\right] := \|W_1^{-1} \Delta Z_1\|_2^2 + \sum_{j=1}^{m-2} \|W_j^{-1} F_j(Z)\|_2^2 \quad .$$

Level 3: "weighted corrections"

$$T_3\left[F(Z), \Delta Z_w\right] := \|\Delta Z_w\|_2^2 \quad .$$

Interesting for the user but without influence on the procedure is the "absolute error"

$$T_0\left[F(Z), \Delta Z_w\right] := \|F(Z)\|_2^2 \quad .$$

3.1.7 The Stop Criterion

Vector Z is considered as a sufficiently good approximation of a zero of F if all components of the weighted correction ΔZ_w are smaller in absolute value than a given upper bound "eps" :

$$\|\Delta Z_w\|_\infty \leq \text{eps} \quad .$$

eps determines the prescribed relative precision of the solution and is an input parameter of BNDSCO (parameter EPS).

3.2 The Use of BNDSCO

3.2.1 The Parameter List of BNDSCO

The parameter list of BNDSCO is

(F, R, METHOD, X, XS, Y, WORK, IWORK, JS, N, M, MS, KS, TOL, ITMAX, KP, MMAX, MSMAX, MMS, NMS, NDW, NDIW, NFILE, PAR, IPAR)

The first three entries are user supplied subprograms:

F: subroutine to evaluate the right–hand side of the system of ODEs as given in 1) in chapter 3.1.1 (see also section 4.1)

R: subroutine to evaluate the boundary and switching conditions 3), 4) in 3.1.1 and to perform the jump conditions 2) in 3.1.1 (see also section 4.2)

METHOD: subroutine to numerically solve initial value problems for ODEs (see also 3.2.3)

The remaining variables are scalars or arrays. Their type is always consistent with the declaration IMPLICIT DOUBLE PRECISION (A–H,O–Z). The explanations below refer to the symbols in chapter 3.1.1. The following variables must be initialized by the calling program before the BNDSCO–call:

X: double precision array of dimension MMS. X contains the nodes x_i in ascending order (see 3.1.1). In particular, X(1) is the initial point and X(M) the final point.

XS: double precision array of dimension MSMAX. XS(I) is the initial guess for the switching point ξ_i (see 3.1.1).

Y: double precision array of dimension (NMS, MMS). Y(i,j) is the initial guess for $y_i(x_j)$, the i–th y–component at the j–th node (see 3.1.1).

N: integer variable; dimension of the ODE–systems f_k in 3.1.1, item 1).

M: integer variable; number of nodes.

MS: integer variable; number of switching points.

KS: For KS = 1 in the printout of the final result the switching points along with the associated y–components will be inserted in order with the nodes. If KS = 0 this is not the case.

TOL: double precision variable; desired relative precision of the solution (parameter eps in 3.1.7). Moreover, TOL affects the error tolerance of the integrator (parameter TOL in subroutine METHOD): It is set to the maximum of TOL * 1.E–2 and EPMIN (see 3.2.2).

ITMAX: integer variable; maximum number of iterations. A reasonable value is ITMAX = 20. ITMAX = 0: Step 1 in 3.1.2 is performed once. ITMAX = –1: Step 1 in 3.1.2 is performed once but in single shooting mode.

KP: integer variable; print parameter (see also 3.2.4).

MMAX: integer variable; dimension of X as declared in the calling program; must be greater or equal M.

MSMAX: integer variable; first dimension of XS and second dimension of JS as declared in the calling program. MSMAX must be greater or equal MS.

MMS: integer variable; second dimension of Y and first dimension of JS as declared in the calling program. MMS must be greater or equal M + MS.

NMS: integer variable; first dimension of Y as declared in the calling program. NMS must be greater or equal N + MS.

NDW: integer variable; dimension of WORK as declared in the calling program. NDW must be greater or equal NMS * (8 + MMAX * (10 + 6 * NMS)).

NDIW: integer variable; dimension of IWORK as declared in the calling program. NDIW must be greater or equal 5 * NMS.

NFILE: integer variable; logical file number for output.

PAR: double precision array whose dimension is fixed by the user. PAR is not changed by BNDSCO and its subprograms. PAR is channelled to the user supplied subprograms F and R. Thus the user is enabled to use application dependent model parameters in F and R without defining common blocks.

IPAR: integer array; the same as PAR only for integer variables.

The following arrays are only used as working space by BNDSCO:

WORK: double precision array of dimension NDW

IWORK: integer array of dimension NDIW

JS: integer array of dimension (MMS, MSMAX)

The input values of X, XS, Y, M and KP are overwritten by BNDSCO. The meaning of the variables on output is as follows:

X: X is unchanged if KS = 0. If KS = 1 the last values of the switching points are inserted into the X–array.

XS: last values of the switching points.

Y: Y(i,j) is the last iterate for $y_i(x_j)$, the i–th y–component at node x_j.

KP: error flag (see also 3.2.6).

M: unchanged if KS = 0; = M + MS if KS = 1

The last values of the XS– and Y–components are the solution data if convergence was achieved. In case of failure XS and Y contain the last iterates.

3.2.2 Local Method Parameters in BNDSCO

BNDSCO contains some local constants affecting the multiple shooting algorithm. They are initialized in a DATA–statement at the beginning of the BNDSCO–routine. The statement should be modified if the values do not seem to be appropriate for the user's machine or application. The names and meanings of the local constants are given below. All of them are double precision constants except KBROY which is of type integer.

EPMACH: EPMACH should contain the actual value for the relative machine precision. EPMACH prevents too small values for the desired accuracy of the solution: If TOL (see 3.2.1) is less than EPMACH * 1.E+4 then TOL is set to this value.

EPMIN: EPMIN is the lower bound for the error tolerance of the integrator.

ETA: parameter η to compute the increments on numerical differentiation (see 3.1.4.1). Recommendation: ETA = SQRT(MAX(EPMIN,TOL*1.E–2)). The number under the square root is the internal error tolerance of the integrator. ETA should always be greater than this number.

FCMIN: FCMIN is the minimum permitted value for the relaxation parameter λ (see 3.1.6). Simultaneously, FCMIN is the initial value for λ. Usual value: FCMIN = 1.E–2.

COND: The iteration matrix is regarded to be singular if the estimate for the condition number (σ in 3.1.5) is greater than COND. In this case BNDSCO returns immediately.

KBROY: KBROY = 1: The iteration matrix is replaced by a Broyden update (see 3.1.4.2) of the previous one as long as all monotonicity tests (see 3.1.6) are satisfied. KBROY = 0: no Broyden update at all.

3.2.3 The Integration Routine

The integration routine provided under the formal parameter METHOD has to have the following parameter list:

(N, F, X, Y, XEND, TOL,HMAX, H, J, L, JS, MMS, PAR, IPAR)

In the following the meaning of these parameters is explained.

N:	number of ordinary differential equations
F:	subroutine to evaluate the right–hand side of the system of ODEs. The parameter list of F is described in chapter 4.1.1.
X:	independent variable
Y(N):	vector containing the dependent variables
XEND:	end of the integration interval
TOL:	upper bound for the local discretization error
HMAX:	maximum step size
H:	initial step size

J, L, JS, MMS, PAR, IPAR are part of the parameter list since they are input paramters of subroutine F. They are neither used nor changed by METHOD. The meaning of JS, MMS, PAR and IPAR is briefly outlined in 3.2.1. For further information see 4.1.

N, XEND, TOL and HMAX are input parameters and must not be changed by METHOD. X, Y and H are input parameters but are changed by METHOD, i.e. are returned with different numerical values.

Important: If the numerical integration breaks down before XEND is reached then BNDSCO expects the output H = 0 . BNDSCO requires integrations to be performed with very high precision. Hence preferable integrators are Runge–Kutta–Fehlberg methods of order 7/8 or extrapolation methods. The program package listed in the appendix contains an implementation of the extrapolation method (SUBROUTINE DIFSYB) which satisfies the standards required by BNDSCO.

The parameter list of METHOD deviates from the usual list of an integrator. Compared to the usual list it is augmented by the part J, L, JS, MMS, PAR, IPAR. A standard integrator can easily be adapted to the form required by BNDSCO by adding exactly this set

of variables. In this case the additional parameters should be declared like this:

```
INTEGER J, L, MMS, JS(MMS,*), IPAR(*)
DOUBLE PRECISION  PAR(*)
```

Moreover, the variables J, L, JS, MMS, PAR, IPAR must be added wherever the subprogram to evaluate the right–hand side of the ODE–system is called.

3.2.4 The Printout Produced by BNDSCO

The input value of KP controls the output produced by BNDSCO. The following options do exist.

KP = – 1: No printout. Useful e.g. for homotopies. Even in this case at least the final results on X, Y and XS should be stored or printed out. Appropriate commands have to be implemented in the calling routine.

KP = 0: First the input X, Y, XS, N, MS, EPS and ITMAX are printed out. Then information on the performance of the Newton–method is given (only if ITMAX > 0). With each iteration a two line message is printed out. The first line contains:

1) The iteration number k is printed in the column headed by "IT" .
2) The values of $T_i\left[F(Z^{(k)}),\ \Delta Z_w^{(k)}\right]$, i = 0, 1, 2, 3, are printed in the columns headed by "ABS.ERR.", "LEVEL 1", "LEVEL 2", "LEVEL 3", respectively (the definitions of T_i, i = 0, 1, 2, 3, are given in 3.1.6).
3) Under "NEW" information is given about how the multiple shooting matrix was obtained:
   NEW = 0: numerical differentiation
   NEW > 0: Broyden update. The number given under NEW is the number of Broyden updates performed subsequently, including the present one.
4) Under "RANK" the rank (M –1) (N + MS) of the multiple shooting matrix is given.
5) Under "COND(E)" and "NORM(E)" the approximations σ and ρ for the condition number and norm of the scaled multiple shooting matrix M* are given, respectively. The definitions of σ, ρ and M* are given in 3.1.5.

The second line refers to the "simplified Newton step": Subsequently numbers for

$$T_i \left[F(Z^{(k+1)}), \overline{\Delta Z_w^{(k+1)}} \right], \quad i = 0, 1, 2, 3,$$

are printed out. Additionally λ_k is printed out under "REL. FC." (the definition of λ_k is given in 3.1.6).

After the last iteration the values of X, XS, and Y are printed out. If the program aborted an additional message is printed out. If ITMAX = 0 then additionally the solutions of the initial value problems

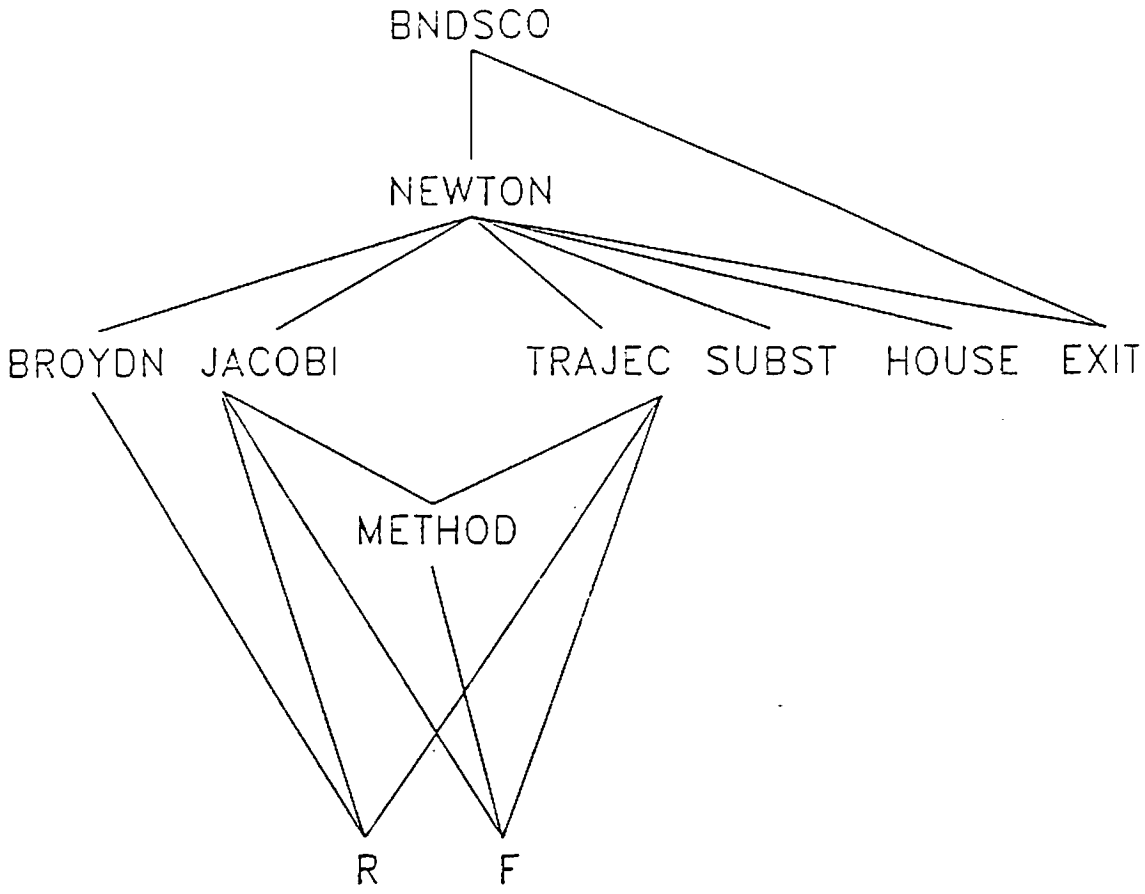
$$y_j' = f(x, y_j), \; x_j \le x \le x_{j+1}, \; y_j(x_j) = Y_j^{(o)}, \quad j = 1,..., M-2$$

are given, i.e. x_{j+1} and $y_j(x_{j+1})$ are printed out. Furthermore the test functions T_o, T_1 (as defined in 3.1.6) are computed for the input data.

KP = 1: Additionally to the output produced with KP = 0 the arrays X, Y and XS are printed out after each iteration.

3.2.5 The Subprogram Structure of BNDSCO

In the diagram below the modular structure of the BNDSCO–package is shown. If two subprograms X and Y are combined with a line and Y stands below X then X contains a call on Y.



The names of the subprograms are chosen as to reflect the task within the algorithm. NEWTON coordinates the Newton–method outlined in 3.1.2. In particular, the λ–strategy described in 3.1.6 is performed in NEWTON. TRAJEC performs the piecewise integration of the trajectory (step 1 and 2 in 3.1.2). JACOBI performs step 3 in 3.1.2, the computation of the multiple shooting matrix M, which is the Jacobian of the function F whose zero is sought by NEWTON. BROYDN performs the Broyden–update of M as outlined in 3.1.4.2. HOUSE decomposes M into the product of a unitary matrix and an upper triangular matrix by application of a sequence of Householder transformations (see also 3.1.5). SUBST performs the backward substitution to obtain the Newton correction based on the decomposed form of M (see also 3.1.5). EXIT processes the whole output produced by BNDSCO. METHOD is the integrator described in 3.2.3. F and R are user supplied subroutines described in chapter 4.

3.2.6 Error Exits

In case the multiple shooting method terminates abnormally, routine BNDSCO returns with KP containing a negative integer which denotes the nature of the error that occurred. For KP ≥ 0 on input the following error messages (in quotes) are printed out.

KP = – 2: "Iteration terminates after ITMAX iteration steps".
Restarting the program with a bigger value ITMAX is sensible only if the test functions T_i, i = 0, 1, 2, 3, show decreasing tendency.

KP = – 3: "BOUNDSCO terminates at singular trajectory".
The numerical solution of an initial value problem fails.

KP = – 4: "Modified Newton–method fails to converge".
The relaxation factor λ_k is reduced below its lower limit FCMIN which is set FCMIN = 0.01 in BNDSCO. Usually it does not make sense to further decrease FCMIN.

KP = – 5: "Numerical differentiation terminated".
During the computation of the multiple shooting matrix via numerical differentiation the integration of an initial value problem has failed. Decreasing ETA (see section 3.2.2) may solve the problem. One might also think of a special choice of increments (for the numerical differentiation) on "critical components". This would require careful examination of the problem and changes in BNDSCO itself.

KP = – 6: "Singular iteration matrix".
The scaled multiple shooting matrix M* is considered singular (see also chapter 3.1.5).

KP = – 7: "Inconsistent estimation of the switching points".
Along an iteration a switching point has been shifted outside the interval [X(1), X(M)]. Possibly a wrong switching structure is assumed.

KP = –8: "Inconsistent estimation of the initial data."
One of the dimension parameters MSMAX, MMAX, MMS, NMS, NDW, NDIW is less than the minimum value given in 3.2.1.

Increasing the number of nodes is always a good first guess to help errors KP = – 3 and also KP = – 7. On the other hand usually M > 40 is not reasonable. Note that the subintervals do not have to be chosen equidistantly. Nodes should be placed more densely in areas with rapid changes and areas that caused error code KP = – 3.

3.2.7 Supplementary Checks on a Solution

Besides boundary and switching conditions the solution of an optimal control problem also satisfies certain sign conditions along whole subintervals. These conditions can not be considered in BNDSCO and have to be checked after BNDSCO converged. Practically this can be done using ITMAX = 0 (see also chapter 3.2.1).

Explicitly these sign conditions are:

a) unconstrained problem:

Legendre–Clebsch condition or – more general – the correctness of the control vector in the sense that it is a solution of the Pontryagin Minimum Principle (see chapter 2.1.3). Equation (7) in 2.1.3 alone does not guarantee this property.

b) constrained problem:

- no constraint violation on unconstrained arcs,
- sign conditions on parameter μ (Equ.(23) in chapter 2.2.4),
- sign conditions on switching functions (see also chapter 2.2.8.2).

4. Subroutines to be Provided by the User

4.1 Subroutine F

4.1.1 The Parameter List of F

Subroutine F evaluates the system $y' = f_k(x, y(x))$ of ordinary differential equations. The structure of f changes with the index k, which is defined by the position of x relative to the switching points: $x \in [\xi_k, \xi_{k+1}]$ (see also 3.1.1). The parameter list of F is

(X, Y, DY, J, L, JS, MMS, PAR, IPAR)

Following variables are input parameters and must not be changed by F:

X: double precision variable; value of the independent variable.

Y: double precision array; recommended dimension declaration is Y(*); contains the dependent variables.

J: integer variable. J is the number of the subinterval where "subinterval" refers to the partition induced by the multiple shooting nodes and the switching points.

L: integer variable. L is the number of the subinterval in the partition defined only by the multiple shooting nodes. The relation to J is $J = L + K$ where K is the number of switching points being smaller than X.

JS: integer array. Recommended dimension declaration is JS(MMS,*). JS contains information about the position of X relative to the switching points:

JS(J, K) = −1 for $X \le XS(K)$,

JS(J, K) = +1 for $X > XS(K)$.

With the information contained in JS the user is enabled to identify the position of X relative to the switching points and the correct index k in the sequence of ODE–systems $y' = f_k(x, y(x))$ in 3.1.1:

$k = 0$ and $X \le XS(1)$ if JS(J, 1) = −1,

$k = MS$ and $X > XS(MS)$ if JS(J, MS) = 1 and

$k = K$ and $XS(K) < X \le XS(K+1)$ if JS(J, K) * JS(J, K+1) = −1.

MMS: integer variable; leading dimension of JS.

The only output parameter is DY (recommended declaration statement DY(*)) which contains the vector $f_k(x, y(x))$.

PAR and IPAR are arrays whose length is fixed by the user. PAR and IPAR are not changed by BNDSCO and its subprograms. PAR and IPAR are reserved for problem dependent model parameters of the user which can thus be accessed without defining common blocks.

PAR: double precision array
IPAR: integer array

Remark: Usually subroutine F will be called several thousand times during a BNDSCO run. Hence the computational effort essentially depends on efficient programming of F.

4.1.2 Execution of a Newton Method

The Minimum Principle often requires the numerical solution of complicated nonlinear algebraic equations systems (Equ. (7) in 2.1.3, Equ.(17) in 2.2.3). It is necessary to do this with very high accuracy. A recommended stop criterion is

$$\left|\frac{\Delta u_i}{u_i}\right| < \text{epmach}$$

for all control variables u_i . Δu_i and epmach denote the Newton correction for u_i and the machine precision, respectively. It can be expected that from one integration step to the next one u_i changes only marginally. Hence the last computed value for u_i is usually an excellent initial guess for the next iteration cycle. Also relaxation factors are not necessary in these cycles. Initial guesses for u_i can be stored on PAR.
To further reduce the computational effort, the Newton matrix may be kept constant over several iteration steps. To improve efficiency, the user should make sure that values which do not depend on u_i will be computed only once (outside the iteration loop) in each call of F.

4.1.3 Check H ≡ const.

In optimal control problems a simple property of the Hamiltonian H is known which can be used to check subroutine F for programming errors. If the state equations are autonomous (as shown in chapter 2.1.2.2 they can always be made autonomous), then the Hamiltonian is constant (see remark 2) in 2.1.3). This property is independent of initial conditions and boundary conditions and holds for both constrained an unconstrained arcs. It must be confirmed by a simple numerical experiment: Integrate the state equations and the adjoint equations by calling METHOD (N, F, X, Y, XEND, TOL, HMAX, H) with arbitrary input values for X, Y, XEND. The relative change of the Hamiltonian must not exceed the magnitude of TOL.

4.2 Subroutine R

4.2.1 The Parameter List of R

Subroutine R evaluates boundary and switching conditions r_i and executes jump conditions h_k. Again the explanations refer to the nomenclature in 3.1.1. The parameter list of R is

(YA, YB, ZZ, W, NYA, NSK, J, L, LS, JS, MMS, PAR, IPAR)

The following parameters are input parameters and must not be changed by R:

YA: Array of length N + MS. The first N components contain the dependent variables at the point $x = x_1$.

YB: Array of length N + MS. The first N components contain the dependent variables at the point $x = x_f$.

LS: Indicates from where and for what purpose R has been called.

- LS = −1: R has been called to initialize NYA and NSK and to evaluate those boundary conditions which have been marked by the NYA–array.
- LS = 0: R has been called to evaluate the two–point boundary conditions.
- LS = K > 0: R has been called at switching point XS(K) in order to evaluate the switching conditions and execute the jump conditions associated with this point.

J, L, JS, MMS: see description of subroutine F in 4.1.1.

In complicated problems it is sometimes necessary to call subroutine F in R. This is the reason why these parameters have been added to the list.

ZZ is a "transient" which is changed by R if nontrivial jump conditions are executed.

ZZ: Array of length N + MS. If R is called at a switching point XS(K) then the first N components of ZZ contain the left hand limits of the independent variables at the point X = XS(K).
On output the first N components of ZZ contain the right hand limits of the dependent variables computed according to the jump condition h_k.

The remaining MS components of YA, YB and ZZ contain the switching points (XS(I), I = 1, MS).

W, NYA and NSK are output parameters:

W: Array of length N + MS. After executing R W contains the values r_i (boundary and switching conditions) in arbitrary sequence. (For further information see chapter 4.2.2.)

NYA: Integer array of length N + MS to mark prescribed initial values. If the initial value of y_I is given explicitly say $y_I(x_1) = y_{I1}$ and W(J) = YA(I) – y_{I1} is the corresponding boundary condition then the statement NYA(I) = J must be added in R. All remaining components of NYA will be set zero by BNDSCO.

NSK: Integer array of length N + MS to mark switching conditions. If W(J) refers to a switching condition at the switching point XS(K) then this has to be indicated by setting NSK(J) = K. All remaining components of NSK will be set zero by BNDSCO.

PAR and IPAR have the same meaning as in subroutine F (see chapter 4.1.1).

4.2.2 Evaluation of Boundary and Switching Conditions and Execution of Jump Conditions

For the evaluation of two–point boundary conditions r_i ($y(x_1)$, $y(x_f)$) the arrays YA, YB contain the numerical values of $y(x_1)$ and $y(x_f)$, respectively. The computed values of r_i have to be stored in the array W.
On the first call of R (with LS = – 1) all those W(J) are evaluated which belong to a

prescribed initial value and are marked by a statement of the form NYA(I) = J for some I. In case one of the conditions is not satisfied, i.e. W(J) ≠ 0 for some J as above the following message is printed out:

"warning: initial data and boundary conditions are inconsistent".

Then the computation is continued with the actually given Y(I, 1), regardless of the value of W(J).

The jumps $y(\xi_k^+) = h_k(\xi_k, y(\xi_k^-))$ are executed if and only if R is called with LS = K to make sure that these jumps, in fact, will be executed only at switching point XS(K).
In this case at the entry of subroutine R $y(\xi_k^-)$ and ξ_k are stored in ZZ(I), I = 1, N, and ZZ(N + k) , respectively. With these data the "jumps" h_k can be evaluated and the new values $y(\xi_k^+)$ can be assigned to the elements ZZ(I), I = 1, N . In fact, variable LS is needed only to control the execution of jump conditions. Furthermore LS may be used to avoid unnecessary evaluations of boundary and switching conditions at points where it is not needed.
Of course the calculation of the switching conditions $r_i(\xi_{k_i}, y(\xi_{k_i}^-))$ has to be performed before the jump conditions at ξ_{k_i} are executed.

Important: The components of vector W have to be scaled appropriately by the user. As described in chapter 3.1.6 the boundary and switching conditions enter the first "monotonicity test" without any further scaling and hence have a direct influence on the Newton method.
The weighting of the components of W should be such that these components are all of about the same order. As an example a boundary condition prescribing the value YI on $y_I(x_f)$ should be formulated in the following form:

W ( ) = YB (I) / YI − 1. DO

5. An Example

5.1. Problem Formulation

The following problem is adapted from Bryson, Ho [3], pp. 121–123.

<u>Problem</u>: Find a control function $a(t)$, $0 \le t \le 1$,
such that the cost functional

$$J[a] := \int_0^1 \frac{1}{2} a(t)^2 \, dt$$

is minimized subject to the constraints

$$\dot{v} = a, \qquad v(0) = -v(1) = 1,$$

$$\dot{x} = v, \qquad x(0) = x(1) = 0,$$

$$x(t) \le \ell \text{ for } 0 \le t \le 1.$$

This problem can be transformed into a Mayer problem by introducing the additional variable z, satisfying $\dot{z} = \frac{1}{2} a^2$, $z(0) = 0$. We then have the equivalent problem

Minimize

$$J[a] = z(1)$$

subject to the constraints

$$\dot{z} = \frac{1}{2} a^2, \qquad z(0) = 0$$

$$\dot{v} = a, \qquad v(0) = -v(1) = 1,$$

$$\dot{x} = v, \qquad x(0) = x(1) = 0,$$

$$x(t) \le \ell \text{ for } 0 \le t \le 1$$

5.2 Necessary Conditions

In the following let $y := (x, v, z)^T$ and $\lambda := (\lambda_x, \lambda_v, \lambda_z)^T$ denote the state and Lagrange multiplier vectors, respectively.
The constraint $S(y) = x - \ell \leq 0$ is of <u>order q = 2;</u>

$$S^{(1)}(y) = \frac{dS}{dt} = v \ , \ S^{(2)}(y, a) = \frac{d^2S}{dt^2} = a \ .$$

The conditions at the beginning of a constrained arc are

$$N(y) := \begin{bmatrix} S^{(0)}(y) \\ S^{(1)}(y) \end{bmatrix} = \begin{bmatrix} x - \ell \\ v \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} .$$

According to 2.2.3, Equ.(14), the Hamiltonian is given by

$$H(y, \lambda, a) = \lambda_x v + \lambda_v a + \frac{1}{2} \lambda_z a^2 .$$

According to chapter 2.2.3, Equ.(15), the adjoint differential equations are given by

$$\dot{\lambda} = -H_y^T = \begin{bmatrix} 0 \\ -\lambda_x \\ 0 \end{bmatrix} .$$

a and μ are computed from

$$\left.\begin{aligned} 0 &= H_a \\ 0 &= \mu \end{aligned}\right\} \ a = -\frac{\lambda_v}{\lambda_z} \ , \ \mu = 0$$

on unconstrained arcs and from

$$\left.\begin{aligned} 0 &= H_a + \mu\, S_a^{(2)} \\ 0 &= S^{(2)} \end{aligned}\right\} \quad a = 0 \;,\; \mu = -\lambda_v$$

on constrained arcs (see chapter 2.2.3, Equ.(17)).

The transversality condition on λ is as given in chapter 2.1.5:

$$\lambda_z(1) = \frac{\partial J}{\partial z(1)} = 1 \,.$$

The jump conditions at the entry point t_1 of the constrained arc are described in chapter 2.2.4, Equ.(20):

$$\lambda(t_1^+)^T = \lambda(t_1^-)^T - \sigma^T N_y = \lambda(t_1^-)^T - [\sigma_1 \,,\, \sigma_2 \,,\, 0]$$

According to chapter 2.2.4, Equ. (28), the jump conditions at a touch point t_b are

$$\lambda(t_b^+)^T = \lambda(t_b^-)^T - \ell_o \;\; S_y = \lambda(t_b^-)^T - [\ell_o, 0, 0] \;\; .$$

At a point t which is either a touch point or a junction between a constrained and unconstrained arc, the continuity of the Hamiltonian is a further switching condition (see chapter 2.2.4, Equs. (22), (27)):

$$0 = H(t^+) - H(t^-)$$

Using conditions stated earlier this can be reduced to

$$v(t_b) = 0$$

at a touch point t_b , or

$$\lambda_v(t_i^-) = 0$$

at a switching point t_i matching constrained and unconstrained arcs.

5.3 Formulation of Multi Point Boundary Value Problems (MPBVP)

Starting from the necessary conditions stated in chapter 5.2 MPBVPs are formulated for the three most simple switching structures (see also chapters 2.1.4 and 2.2.6).

As shown in chapter 5.2 λ_z is continuous in all possible switching points. Hence, together with $\lambda_z(1) = 1$ and $\dot{\lambda}_z = 0$ we find $\lambda_z \equiv 1$.
Furthermore we note that variable z does not appear in any of the necessary conditions, so that the optimal trajectory can be determined independently of z. Hence z and λ_z will be left out in the remainder.

5.3.1 TPBVP for the Unconstrained Problem

If $\ell \geq 0$ is chosen such that the constraint $x - \ell \leq 0$ is never violated, then the optimal trajectory solves the following TPBVP:

differential equations		boundary conditions
$\dot{x}$	$= v$	$x(0) = 0$
$\dot{v}$	$= -\lambda_v$	$x(1) = 0$
$\dot{\lambda}_x$	$= 0$	$v(0) = 1$
$\dot{\lambda}_v$	$= -\lambda_x$	$v(1) = -1$

5.3.2 MPBVP in Case of a Touch Point

differential equations	boundary conditions
$\dot{x} = v$	$x(0) = 0$
$\dot{v} = -\lambda_v$	$x(1) = 0$
$\dot{\lambda}_x = 0$	$v(0) = 1$
$\dot{\lambda}_v = -\lambda_x$	$v(1) = -1$
$\dot{\ell}_o = 0$	

Switching and jump condition at touch point t_b:

$$x(t_b) = \ell \quad , \quad v(t_b) = 0 \ ,$$
$$\lambda_x(t_b^+) = \lambda_x(t_b^-) - \ell_o \ .$$

5.3.3 MPBVP in Case of a Constrained Arc

differential equations	boundary conditions
$\dot{x} = v$	$x(0) = 0$
$\dot{v} = a$	$x(1) = 0$
$\dot{\lambda}_x = 0$	$v(0) = 1$
$\dot{\lambda}_v = -\lambda_x$	$v(1) = -1$
$\dot{\sigma}_1 = 0$	
$\dot{\sigma}_2 = 0$	

$$a = \begin{cases} 0 & \text{for } t_1 \le t \le t_2 \\ -\lambda_v & \text{otherwise} \end{cases}$$

Switching and jump conditions at t_1:

$$x(t_1) = \ell, \qquad v(t_1) = 0, \qquad \lambda_v(t_1^-) = 0,$$
$$\lambda_x(t_1^+) = \lambda_x(t_1^-) - \sigma_1, \qquad \lambda_v(t_1^+) = \lambda_v(t_1^-) - \sigma_2.$$

Switching conditions at t_2:

$$\lambda_v(t_2) = 0$$

5.4 Solution of the Multi Point Boundary Value Problems

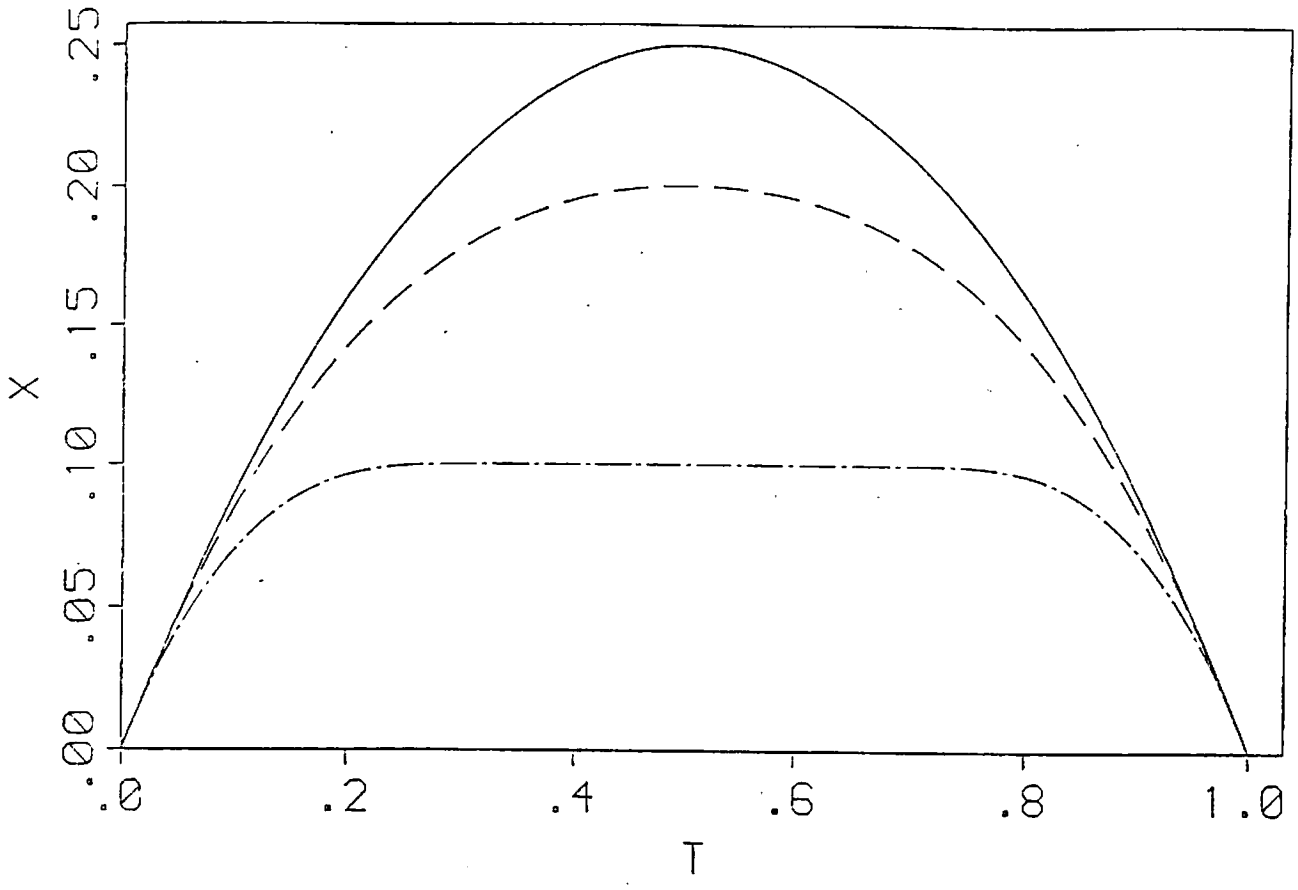
The MPBVPs in chapter 5.3 can be solved analytically as shown in Bryson, Ho [3], pp. 121–123. The equations also have been implemented in subroutine LSG (see chapter 5.5). Depending on parameter ℓ the following switching structures are found to solve the problem:

$\ell \geq \frac{1}{4}$: no touch points or constrained arcs. The optimal trajectory is a solution of 5.3.1.

$\frac{1}{4} > \ell \geq \frac{1}{6}$: touch point at $t = \frac{1}{2}$; the optimal trajectory is a solution of 5.3.2.

$0 < \ell < \frac{1}{6}$: constrained arc $[3\ \ell\ ,\ 1 - 3\ \ell]$; the optimal trajectory is a solution of 5.3.3.

In all cases state x(t) is axis symmetric with respect to $t = \frac{1}{2}$. The time histories of x(t) for $\ell = 0.4$ (solid line), $\ell = 0.2$ (dashed line) and $\ell = 0.1$ (dashed–dotted line)are given in the following figure.



5.5 FORTRAN Programs to Solve the Example Problem Using BNDSCO

The following main program subsequently solves the BVPs 5.3.1 (for $\ell = 0.4$), 5.3.2 (for $\ell = 0.2$) and 5.3.3 (for $\ell = 0.1$). The components PAR(1) and IPAR(1) serve to distinguish the three cases: PAR(1) contains the value of ℓ, IPAR(1) takes the values 1, 2 and 3 to distinguish the different switching structures. In each case subroutine LSG first computes the arrays Y and XS as the exact solutions of the respective problem (The solution formulae are given in Bryson, Ho [3], pp. 120 – 123.). Then all components are multiplied by a "perturbation factor" DEVIAT = .97. The printout of the run can be found at the end of the appendix and contains further comments.

Note the dimension statements at the beginning of the main program. All dimensions only depend on three problem dependent parameters: the maximum number of nodes, the maximum number of switching points and the maximum dimension of the ODE–system. This is the way how to minimize the number of problem dependent constants in the declaration of arrays.

```
      PROGRAM MAIN
C-----------------------------------------------------------------------
C     MAIN PROGRAM CALLING BNDSCO. EXAMPLE TAKEN FROM:
C     BRYSON, A.E.JR., HO, Y.-C.: APPLIED OPTIMAL CONTROL. WALTHAM,
C     MASS.: GINN & CO. (1969), PP. 120 - 123.
C-----------------------------------------------------------------------
      IMPLICIT LOGICAL(A-Z)
      INTEGER          MMAX, MSMAX, MMS, NMAX, NDW, NDIW, NMS, NY
C-----------------------------------------------------------------------
C     BASIC DIMENSION STATEMENTS:
C     MMAX : MAXIMUM NUMBER OF NODES
C     MSMAX: MAXIMUM NUMBER OF SWITCHING POINTS
C     NMAX : MAXIMUM DIMENSION OF THE ODE-SYSTEM
C-----------------------------------------------------------------------
      PARAMETER ( MMAX = 10, MSMAX = 2, NMAX = 6)
C-----------------------------------------------------------------------
C     DERIVATION OF FURHTER DIMENSIONS
C-----------------------------------------------------------------------
      PARAMETER (    NMS  = NMAX + MSMAX,
     &               NY   = NMS * MMAX,
     &               MMS  = MMAX + MSMAX,
     &               NDW  = NMS * (8 + MMAX*(10+6*NMS)),
     &               NDIW = 5 * NMS )
C-----------------------------------------------------------------------
C     ARRAYS IN THE BNDSCO-LIST
C-----------------------------------------------------------------------
      DOUBLE PRECISION X(MMAX), Y(NY), XS(MSMAX), WORK(NDW), PAR(1)
      INTEGER          IWORK(NDIW), JS(MMS,MSMAX), IPAR(1)
C-----------------------------------------------------------------------
C     SIMPLE VARIABLES IN THE BNDSCO-LIST
C-----------------------------------------------------------------------
      DOUBLE PRECISION TOL
      INTEGER          N, M, MS, KS, ITMAX, KP, NFILE
C-----------------------------------------------------------------------
C     FURHTER VARIABLES
C-----------------------------------------------------------------------
      DOUBLE PRECISION DEVIAT
      INTEGER          I
C-----------------------------------------------------------------------
C     EXTERNAL SUBROUTINES
C-----------------------------------------------------------------------
      EXTERNAL F, R, DIFSYB
C-----------------------------------------------------------------------
C     VARIOUS INITIAL VALUES
C-----------------------------------------------------------------------
      DATA  M   , KS , TOL   , ITMAX , NFILE , DEVIAT /
     &      10  , 0  , 1.D-5, 30    , 6     , .97D0  /
C-----------------------------------------------------------------------
C     DEFINE THE NODES
C-----------------------------------------------------------------------
      DO 50 I=1,M
         X(I) = DBLE(I-1) / DBLE(M-1)
   50 CONTINUE
```

```
C-----------------------------------------------------------------------
C     BNDSCO-CALL FOR THREE DIFFERENT CASES:
C     I = IPAR(1) = 1: UNCONSTRAINED CASE FOR L=.4 IN BRYSON/HO
C     I = IPAR(1) = 2: CONSTRAINED CASE WITH A TOUCH POINT FOR L=.2
C     I = IPAR(1) = 3: TRAJECTORY WITH A CONSTRAINED SUBARC FOR L=.1
C     PAR(1) IS THE VALUE OF THE PARAMETER L IN THE
C     BRYSON/HO-EXAMPLE.
C     SUBR. LSG COMPUTES THE SOLUTION OF THE PROBLEM AND PERTURBS IT
C     WITH THE FACTOR DEVIAT (DEVIAT=1: NO PERTURBATION).
C-----------------------------------------------------------------------
      PAR(1) = .8D0
      DO 100 I=1,3
         KP    = 0
         N     = 3 + I
         MS    = I - 1
         IPAR(1) = I
         PAR(1)  = PAR(1) * .5D0
         CALL LSG(X,XS,Y,NMS,N,M,MS,PAR(1),IPAR(1),DEVIAT)
         CALL BNDSCO(F,R,DIFSYB,X,XS,Y,WORK,IWORK,JS,N,M,MS,KS,TOL,
     &              ITMAX,KP,MMAX,MSMAX,MMS,NMS,NDW,NDIW,NFILE,
     &              PAR,IPAR)
  100 CONTINUE
C-----------------------------------------------------------------------
      STOP
      END
C
C***********************************************************************
      SUBROUTINE F(X,Y,DY,J,L,JS,MMS,PAR,IPAR)
C***********************************************************************
C    INPUT:
C
C    X     INDEPENDENT VARIABLE
C    Y     ARRAY OF MAXIMUM DIMENSION 6. (Y(I),I=1,4) IS THE VECTOR
C          (X,V,LAMBDA-X,LAMBDA-V) IN THE BRYSON/HO-EXAMPLE. Y(5)
C          AND Y(6) ARE THE JUMPS IN LAMBDA-X AND LAMBDA-V,
C          RESPECTIVELY.
C    J, L, JS, MMS : SEE DESCRIPTION OF BNDSCO.
C    PAR  PAR(1) IS THE PARAMETER L IN THE BRYSON/HO-EXAMPLE.
C    IPAR IPAR(1) IS THE CODE FOR THE SWITCHING STRUCTURE:
C          IPAR(1) = 1: UNCONSTRAINED CASE
C          IPAR(1) = 2: TOUCH POINT
C          IPAR(1) = 3: CONSTRAINED SUBARC
C
C    OUTPUT:
C
C    DY    ARRAY OF MAXIMUM DIMENSION 6. TIME DERIVATIVES OF Y.
C-----------------------------------------------------------------------
      IMPLICIT LOGICAL (A-Z)
C-----------------------------------------------------------------------
C     FORMAL PARAMETERS
C-----------------------------------------------------------------------
      DOUBLE PRECISION X, Y(*), DY(*), PAR(*)
      INTEGER          J, L, MMS, JS(MMS,*), IPAR(*)
C-----------------------------------------------------------------------
C     LOCAL VARIABLES
C-----------------------------------------------------------------------
      DOUBLE PRECISION A
```

```
C-----------------------------------------------------------------------
C     A IS THE OPTIMAL CONTROL
C-----------------------------------------------------------------------
      A = - Y(4)
      IF ((IPAR(1).EQ.3) .AND. (JS(J,1)*JS(J,2).EQ.-1)) A = 0.D0
C-----------------------------------------------------------------------
C     STATE AND ADJOINT EQUATIONS
C-----------------------------------------------------------------------
      DY(1) = Y(2)
      DY(2) = A
      DY(3) = 0.D0
      DY(4) = - Y(3)
      IF (IPAR(1).GT.1) DY(5) = 0.D0
      IF (IPAR(1).GT.2) DY(6) = 0.D0
C-----------------------------------------------------------------------
      RETURN
      END
C
C
C***********************************************************************
      SUBROUTINE R(YA,YB,ZZ,W,NYA,NSK,J,L,LS,JS,MMS,PAR,IPAR)
C***********************************************************************
C     BOUNDARY, SWITCHING AND JUMP CONDITIONS FOR THE
C     BRYSON/HO-EXAMPLE
C-----------------------------------------------------------------------
C     YA, ZZ AND YB ARE THE INITIAL, INTERMEDIATE AND FINAL VALUES OF
C     THE VECTOR (X,V,LAMBDA-X,LAMBDA-V) IN THE BRYSON/HO-EXAMPLE.
C     THE 5TH AND 6TH COMPONENT ARE THE JUMPS IN LAMBDA-X AND
C     LAMBDA-V, RESPECTIVELY.
C     FURTHER DESCRIPTION OF THE PARAMETERS: SEE DOCUMENTATION
C     MEANING OF PAR AND IPAR: SEE SUBR. F
C-----------------------------------------------------------------------
      IMPLICIT LOGICAL (A-Z)
C-----------------------------------------------------------------------
C     FORMAL PARAMETERS
C-----------------------------------------------------------------------
      DOUBLE PRECISION YA(*), YB(*), ZZ(*), W(*), PAR(*)
      INTEGER          J, L, MMS, JS(MMS,*), LS, NYA(*), NSK(*), IPAR(*)
C-----------------------------------------------------------------------
C     LOCAL VARIABLES: ICASE WILL TAKE THE VALUE OF IPAR(1).
C-----------------------------------------------------------------------
      INTEGER ICASE
C-----------------------------------------------------------------------
C     INITIAL VALUES FOR NYA AND NSK
C-----------------------------------------------------------------------
      ICASE = IPAR(1)
      IF (LS.EQ.-1) THEN
         GOTO (100, 200, 300), ICASE
  300       NSK(7) = 1
            NSK(8) = 2
  200       NSK(5) = 1
            NSK(6) = 1
  100       NYA(1) = 1
            NYA(2) = 2
      ENDIF
```

```
C-----------------------------------------------------------------------
C     EVALUATION OF THE TWO-POINT BOUNDARY CONDITIONS
C-----------------------------------------------------------------------
      IF (LS.LE.0) THEN
         W(1) = YA(1)
         W(2) = YA(2) - 1.D0
         W(3) = YB(1)
         W(4) = YB(2) + 1.D0
      ENDIF
C-----------------------------------------------------------------------
C     JUMPS AND SWITCHING CONDITIONS AT THE FIRST SWITCHING POINT
C-----------------------------------------------------------------------
      IF (LS.EQ.1) THEN
         IF (ICASE.GT.1) THEN
            W(5)  = ZZ(1) / PAR(1) - 1.D0
            W(6)  = ZZ(2)
            ZZ(3) = ZZ(3) - ZZ(5)
         ENDIF
         IF (ICASE.EQ.3) THEN
            W(7)  = ZZ(4)
            ZZ(4) = ZZ(4) - ZZ(6)
         ENDIF
      ENDIF
C-----------------------------------------------------------------------
C     SWITCHING CONDITION AT THE SECOND SWITCHING POINT
C-----------------------------------------------------------------------
      IF ((LS.EQ.2) .AND. (ICASE.EQ.3)) W(8) = ZZ(4)
C-----------------------------------------------------------------------
      RETURN
      END
C
C
C***********************************************************************
      SUBROUTINE LSG(X,XS,Y,NMS,N,M,MS,AL,ICASE,DEVIAT)
C***********************************************************************
C     SOLUTION OF THE BRYSON/HO-EXAMPLE AND SUBSEQUENT
C     PERTURBATION
C-----------------------------------------------------------------------
C     INPUT (READ ONLY):
C
C     X          ARRAY OF DIMENSION M. MULTIPLE SHOOTING NODES IN
C                ASCENDING ORDER
C     NMS        FIRST DIMENSION OF Y, NMS.GE.N+MS REQUIRED
C     N          NUMBER OF THE DEPENDENT VARIABLES
C     M          NUMBER OF MULTIPLE SHOOTING NODES = FIRST DIMENSION
C                OF X = SECOND DIMENSION OF Y
C     MS         NUMBER OF SWITCHING POINTS
C     AL         PARAMETER L IN THE BRYSON/HO-EXAMPLE
C     ICASE      CODE FOR THE SWITCHING STRUCTURE:
C                =1: UNCONSTRAINED CASE (L > 1/4)
C                =2: CONSTRAINED CASE WITH A TOUCH POINT (1/6 <= L <= 1/4)
C                =3: TRAJECTORY WITH A CONSTRAINED SUBARC (0 < L <= 1/6)
C     DEVIAT     THE SOLUTION IS PERTURBED WITH THE FACTOR DEVIAT:
C                THE EXACT VALUE OF Y(I,J) IS REPLACED BY Y(I,J)*DEVIAT
C                FOR ALL I,J EXCEPT FOR THE FIRST TWO INITIAL VALUES.
C                THE EXACT VALUE OF XS(I) IS REPLACED BY XS(I)*DEVIAT
C                FOR 1 <= I <= MS.
```

```
C
C     OUTPUT:
C
C     Y          ARRAY OF DIMENSION (NMS,M). SOLUTION MATRIX OF
C                BNDSCO
C                PERTURBED WITH THE FACTOR DEVIAT. Y(*,J) SERVES AS AN
C                INITIAL GUESS FOR THE VECTOR (X,V,LAMBDA-X,LAMBDA-V)
C                IN THE BRYSON/HO-EXAMPLE AT THE NODE X(J). Y(5,J) AND
C                Y(6,J) TAKE THE ROLE OF THE JUMPS IN LAMBDA-X AND
C                LAMBDA-V, RESPECTIVELY.
C     XS         ARRAY OF DIMENSION MS. SWITCHING POINTS PERTURBED
C                WITH THE FACTOR DEVIAT.
C-----------------------------------------------------------------------
      IMPLICIT LOGICAL (A-Z)
C-----------------------------------------------------------------------
C     FORMAL PARAMETERS
C-----------------------------------------------------------------------
      INTEGER          NMS, N, M, MS, ICASE
      DOUBLE PRECISION X(M), XS(*), Y(NMS,M), DEVIAT, AL
C-----------------------------------------------------------------------
C     LOCAL VARIABLES
C-----------------------------------------------------------------------
      DOUBLE PRECISION AL3, T, B3, B4, AL31
      INTEGER          I, J
C-----------------------------------------------------------------------
C     SWITCHING POINTS AND AUXILIARY VARIABLES DEPENDENT ON AL AND
C     ICASE
C-----------------------------------------------------------------------
      IF (ICASE.EQ.2) THEN
         B3 = 1.D0 - 3.D0 * AL
         B4 = 1.D0 - 4.D0 * AL
         XS(1) = .5D0
      ENDIF
      IF (ICASE.EQ.3) THEN
         AL3 = 3.D0 * AL
         XS(1) = AL3
         XS(2) = 1.D0 - AL3
      ENDIF
C-----------------------------------------------------------------------
C     LOOP FOR THE NODES
C-----------------------------------------------------------------------
      DO 100 I=1,M
C-----------------------------------------------------------------------
C        SOLUTION FOR ICASE=1
C-----------------------------------------------------------------------
         IF (ICASE.EQ.1) THEN
            Y(1,I) = X(I) * (1.D0 - X(I))
            Y(2,I) = 1.D0 - 2.D0 * X(I)
            Y(3,I) = 0.D0
            Y(4,I) = 2.D0
         ENDIF
C-----------------------------------------------------------------------
C        SOLUTION FOR ICASE=2
C-----------------------------------------------------------------------
         IF (ICASE.EQ.2) THEN
            IF (X(I).LE.XS(1)) THEN
               T = X(I)
```

```
            ELSE
               T = 1.D0 - X(I)
            ENDIF
            Y(1,I) = ((B4 * T - B3) * 4.D0 * T + 1.D0) * T
            Y(2,I) = (12.D0 * B4 * T - 8.D0 * B3) * T + 1.D0
            Y(3,I) = 24.D0 * B4
            Y(4,I) = 8.D0 * (B3 - 3.D0 * B4 * T)
            Y(5,I) = 48.D0 * B4
            IF (X(I).GT.XS(1)) THEN
               Y(2,I) = - Y(2,I)
               Y(3,I) = - Y(3,I)
            ENDIF
         ENDIF
C-------------------------------------------------------------------------
C        SOLUTION FOR ICASE=3
C-------------------------------------------------------------------------
         IF (ICASE.EQ.3) THEN
            IF (X(I).LE.XS(1)) THEN
               T = X(I)
            ELSE
               T = 1.D0 - X(I)
            ENDIF
            AL31 = 1.D0 - T / AL3
            IF ((X(I).LE.XS(1)) .OR. (X(I).GE.XS(2))) THEN
               Y(1,I) = AL * (1.D0 - AL31**3)
               Y(2,I) = AL31**2
               IF (X(I).GE.XS(2)) Y(2,I) = - Y(2,I)
            ELSE
               Y(1,I) = AL
               Y(2,I) = 0.D0
            ENDIF
            Y(3,I) = 2.D0 / AL3**2
            IF (X(I).GE.XS(1)) Y(3,I) = - Y(3,I)
            Y(4,I) = 2.D0 * AL31 / AL3
            Y(5,I) = 4.D0 / AL3**2
            Y(6,I) = 3.D0 * Y(5,I) * (1.D0 / 6.D0 - AL)
         ENDIF
C-------------------------------------------------------------------------
  100 CONTINUE
C-------------------------------------------------------------------------
C     THE SOLUTION IS PERTURBED WITH DEVIAT
C-------------------------------------------------------------------------
      IF (DEVIAT.NE.1.D0) THEN
         DO 210 I=1,M
            DO 220 J=1,N
               IF ((I.GT.1).OR.(J.GT.2)) Y(J,I) = Y(J,I) * DEVIAT
  220       CONTINUE
  210    CONTINUE
         IF (MS.GE.1) XS(1) = XS(1) * DEVIAT
         IF (MS.GE.2) XS(2) = XS(2) * DEVIAT
      ENDIF
C-------------------------------------------------------------------------
      RETURN
      END
```

6. References

[1] Bock, H.G.
Optimierungsverfahren Software und praktische Anwendungen. Carl–Cranz–Lehrgang R1.06, Heidelberg, 1983.

[2] Bryson, A.E., Denham, W.F., Dreyfus, S.E.
Optimal Programming Problems with Inequality Constraints I, Necessary Conditions for Extremal Solutions. AIAA Journal, Vol.1, pp. 2544–2550, 1963.

[3] Bryson, A.E., Ho, Y.–C.
Applied Optimal Control. Waltham, Mass., Ginn and Company, 1969.

[4] Bulirsch, R.
Die Mehrzielmethode zur numerischen Lösung von nichtlinearen Randwertproblemen und Aufgaben der optimalen Steuerung. Carl–Cranz–Lehrgang 1971, Nachdruck Technische Universität München, Mathematisches Institut, 1985.

[5] Deuflhard, P.
A Modified Newton Method for the Solution of Ill–Conditioned Systems of Nonlinear Equations with Application to Multiple Shooting. Num. Math. 22, pp. 298–315, 1974.

[6] Deuflhard, P.
A Relaxation Strategy for the Modified Newton Method. In: *Optimization and Optimal Control* (Eds. Bulirsch, Oettli, Stoer). Heidelberg, Springer Lecture Notes Vol. 477, 1975.

[7] Hiltmann, P.
Numerische Behandlung optimaler Steuerprozesse mit Zustandsbeschränkungen mittels der Mehrzielmethode. Diplomarbeit, Mathematisches Institut der Technischen Universität München, 1983.

[8] Oberle, H.J.
BOUNDSCO – Hinweise zur Benutzung des Mehrzielverfahrens für die numerische Lösung von Randwertproblemen mit Schaltbedingungen. Hamburger Beiträge zur Angewandten Mathematik, Berichte 6, 1987.

[9] Oberle, H.J.
Numerische Berechnung optimaler Steurungen von Heizung und Kühlung für ein realistisches Sonnenhausmodell. TU–Bericht Nr. M831O, Mathematisches Institut der Technischen Universität München, 1983.

[10] Stoer, J.
Einführung in die Numerische Mathematik I. Springer, Heidelberger Taschenbücher Band 105, 1979.

[11] Stoer, J., Bulirsch, R.
Einführung in die Numerische Mathematik II. Springer, Heidelberger Taschenbücher Band 114, 1980.

[12] Stoer, J., Bulirsch, R.
Introduction to Numerical Analysis. Springer, New York, 1983.

```
C***********************************************************************
      SUBROUTINE BNDSCO(F,R,METHOD,X,XS,Y,WORK,IWORK,JS,N,M,MS,KS,TOL,
     +                  ITMAX,KP,MMAX,MSMAX,MMS,NMS,NDW,NDIW,NFILE,
     +                  PAR,IPAR)
C***********************************************************************
C
C     B O U N D S C O
C
C     REALIZATION OF THE MULTIPLE SHOOTING METHOD
C
C     SPECIAL VERSION OF THE PROGRAM  B O U N D S O L
C     FOR THE SOLUTION OF MULTI-POINT BOUNDARY-VALUE PROBLEMS
C     WITH SWITCHING CONDITIONS
C
C     WRITTEN BY
C          H.J. OBERLE
C          UNIVERSITY OF HAMBURG
C
C     REFERENCES :
C
C     BULIRSCH, R.:     DIE MEHRZIELMETHODE ZUR NUMERISCHEN LOESUNG
C                       VON NICHTLINEAREN RANDWERTPROBLEMEN UND
C                       AUFGABEN DER OPTIMALEN STEUERUNG.
C                       REPORT OF THE CARL CRANZ GESELLSCHAFT,
C                       OBERPFAFFENHOFEN, WEST GERMANY, 1971.
C     DEUFLHARD,P.:     A RELAXATION STRATEGY FOR THE MODIFIED NEWTON
C                       METHOD. IN: BULIRSCH, R., OETTLI, W. AND
C                       STOER, J. (EDS.) PROCEEDINGS OF CONFERENCE ON
C                       OPTIMIZATION AND OPTIMAL CONTROL,
C                       OBERWOLFACH.
C                       LECTURE NOTES VOL.477, SPRINGER, 1975.
C     OBERLE, H.J.:     NUMERISCHE BERECHNUNG OPTIMALER STEUERUNGEN
C                       VON HEIZUNG UND KUEHLUNG FUER EIN
C                       REALISTISCHES SONNENHAUSMODELL. TECHNISCHE
C                       UNIVERSITAET MUENCHEN, TU-BERICHT NR. M8310,
C                       1983.
C     OBERLE, H.J.:     BOUNDSCO - HINWEISE ZUR BENUTZUNG DES
C                       MEHRZIEL-VERFAHRENS FUER DIE NUMERISCHE
C                       LOESUNG VON RANDWERTPROBLEMEN MIT
C                       SCHALTBEDINGUNGEN. HAMBURGER BEITRAEGE ZUR
C                       ANGEWANDTEN MATHEMATIK, BERICHTE 6, 1987.
C
C     UPDATE : MARCH 1989
C
C     ***************************************************
C
C     F(X,Y,DY,J,L,JS,MMS,PAR,IPAR) ...:
C                       RIGHT-HAND SIDE OF SYSTEM OF FIRST-ORDER
C                       DIFFERENTIAL EQUATIONS
C                       X, Y(N) : ACTUAL INTEGRATION NODE
C                       DY(N)   : VALUES OF THE DERIVATIVES AT X
C                       J : NUMBER OF SHOOTING INTERVAL (INCLUDING THE
C                       SWITCHING POINTS)
C                       L : NUMBER OF SHOOTING INTERVAL (WITHOUT THE
C                       SWITCHING POINTS)
C                       JS : INDICATES THE POSITION OF THE
C                       ACTUAL INTEGRATION POINT WITH RESPECT TO THE
```

```
C                       SWITCHING POINTS
C                       JS(J,K)= 1, IF X.GT.XS(K) , K=1,..,MS
C                       JS(J,K)=-1, IF X.LT.XS(K) , K=1,..,MS
C                       MMS : DIMENSION PARAMETER
C                       PAR(*) : DOUBLE PRECISION ARRAY FOR MODEL
C                       PARAMETERS OF THE USER
C                       IPAR(*) : INTEGER ARRAY FOR MODEL
C                       PARAMETERS OF THE USER
C
C   R(YA,YB,ZZ,W,NYA,NSK,J,L,LS,JS,MMS,PAR,IPAR) ...:
C                       TWO-POINT BOUNDARY CONDITIONS AND
C                       SWITCHING CONDITIONS
C                  -    YA, YB: VALUES OF Y(A), Y(B)
C                  -    ZZ : VALUE OF Y(X) AT THE ACTUAL SWITCHING POINT
C                  -    W(NMS) : BOUNDARY- AND SWITCHING CONDITIONS
C                  -    NYA(NMS) : INTEGER ARRAY INDICATING THE STATE
C                       VARIABLES FIXED AT THE INITIAL POINT X=A :
C                       NYA(I)=K , IF W(K)=YA(I)-PRESCIBED VALUE
C                  -    NSK(NMS) : INTEGER ARRAY INDICATING THE
C                       SWITCHING CONDITIONS :
C                       NSK(K)=L , IF W(K) DESCRIBES A SWITCHING CONDI-
C                       TION, WHICH HAS TO BE SATISFIED AT XS(L).
C                  -    LS : INTEGER PARAMETER
C                       LS<=0, IF SUBROUTINE R IS CALLED AT X=A, X=B
C                       LS=L, IF SUBROUTINE R IS CALLED AT THE SWIT-
C                       CHING POINT XS(L), L=1,..MS
C                       PAR(*) : DOUBLE PRECISION ARRAY FOR MODEL
C                       PARAMETERS OF THE USER
C                       IPAR(*) : INTEGER ARRAY FOR MODEL
C                       PARAMETERS OF THE USER
C
C   METHOD(N,F,X,Y,XEND,TOL,HMAX,H,J,L,JS,MMS,PAR,IPAR)...:
C                       SUBROUTINE FOR THE SOLUTION OF INITIAL-
C                       VALUE PROBLEMS
C                       METHOD SOLVES THE INITIAL VALUE PROBLEM WITHIN
C                       THE INTERVAL (X,XEND) WITH THE (RELATIVE)
C                       TOLERANCE TOL.
C                       H IS THE INITIAL STEPSIZE, AND THE LAST SUCCESS-
C                       FUL STEPSIZE, RESPECTIVELY.
C                       FAILURE IS INDICATED BY H=0.0
C                       PAR(*) : DOUBLE PRECISION ARRAY FOR MODEL
C                       PARAMETERS OF THE USER
C                       IPAR(*) : INTEGER ARRAY FOR MODEL
C                       PARAMETERS OF THE USER
C
C   ***********************************************************
C
C   --- INPUT PARAMETERS :
C
C   X(M),Y(N,M)         INITIAL DATA
C   XS(MS)              INITIAL GUESS OF SWITCHING POINTS
C   WORK(NDW)           REAL ARRAY USED FOR INTERNAL STORAGE
C   IWORK(NDIW)         INTEGER ARRAY USED FOR INTERNAL STORAGE
```

```
C   JS(MMS,MSMAX)  INTEGER ARRAY USED FOR THE POSITION OF THE
C                  ACTUAL INTEGRATION POINT WITH RESPECT TO
C                  SWITCHING POINTS
C                  JS(J,K)= 1, IF X.GT.XS(K) , K=1,..,MS
C                  JS(J,K)=-1, IF X.LT.XS(K) , K=1,..,MS
C                  J : SHOOTING INTERVAL (INCLUDING THE
C                  SWITCHING POINTS)
C   N              NUMBER OF FIRST-ORDER DIFFERENTIAL EQUATIONS
C   M              NUMBER OF NODES
C   MS             NUMBER OF SWITCHING POINTS
C   MMAX           DIMENSION PARAMETER; M.LE.MMAX
C   MSMAX          DIMENSION PARAMETER; MS.LE.MSMAX
C   MMS            DIMENSION PARAMETER; M+MS.LE.MMS
C   NMS            DIMENSION PARAMETER; N+MS.LE.NMS
C   NDW            DIMENSION PARAMETER;
C                  NDW.GE.NMS*(8+MMAX*(10+6*NMS))
C   NDIW           DIMENSION PARAMETER; NDIW.GE.5*NMS
C   NFILE          FILE NUMBER FOR OUTPUT
C   KS             0          NUMBER OF NODES UNCHANGED
C                  1          SWITCHING POINTS INSERTED
C   TOL            REQUIRED RELATIVE PRECISION OF SOLUTION
C   ITMAX          MAXIMUM PERMITTED NUMBER OF ITERATIONS
C                  ITMAX=0    COMPUTATION OF ONE BASIC TRAJECTORY
C                             (MULTIPLE SHOOTING)
C                  ITMAX=-1   COMPUTATION OF ONE BASIC TRAJECTORY
C                             (SIMPLE SHOOTING)
C   KP             PRINT PARAMETER
C                  KP=-1      NO PRINT
C                  KP= 0      INITIAL DATA
C                             ITERATIVE VALUES OF LEVEL FUNCTIONS
C                             SOLUTION DATA
C                  KP=+1      ADDITIONALLY
C                             ITERATES X(J),Y(I,J),XS(K)
C                             ITERATIVE VALUES AT SWITCHING POINTS
C   PAR            DOUBLE PRECISION ARRAY FOR MODEL PARAMETERS
C                  OF THE USER. PAR IS NOT CHANGED BY BNDSCO AND
C                  ITS SUBPROGRAMS. THE DIMENSION MUST BE FIXED BY
C                  THE USER.
C   IPAR           INTEGER ARRAY FOR MODEL PARAMETERS OF THE
C                  USER. IPAR IS NOT CHANGED BY BNDSCO AND ITS SUB-
C                  PROGRAMS. THE DIMENSION MUST BE FIXED BY THE
C                  USER.
C
C   --- OUTPUT PARAMETERS :
C
C   M              LIKE INPUT M, IF KS=0
C                  NUMBER OF NODES AND SWITCHING POINTS, IF KS=1
C   X(M),Y(N,M)    SOLUTION DATA (OR FINAL DATA, RESPECTIVELY)
C   XS(MS)         SWITCHING POINTS
C   KP             .GT.0      NUMBER OF ITERATIONS PERFORMED
C                             TO OBTAIN THE SOLUTION
C                  .LT.0      TERMINATION OF THE ITERATION
C                  KP=-2      ITERATION TERMINATES AFTER ITMAX
C                             ITERATIONS
C                  KP=-3      INTEGRATION OF AN INITIAL VALUE
C                             PROBLEM TERMINATES
C                  KP=-4      NEWTON METHOD FAILS TO CONVERGE
```

```
C                     KP=-5     NUMERICAL DIFFERENTIATION IS
C                               TERMINATED DUE TO NEIGHBOURING
C                               SINGULAR TRAJECTORIES
C                     KP=-6     SINGULAER ITERATION MATRIX
C                     KP=-7     INCONSISTENT ESTIMATION OF THE
C                               SWITCHING POINTS
C                     KP=-8     INCONSISTENT ESTIMATION OF THE INITIAL
C                               DATA
C
C     ************************************************************
C
C     —— INTERNAL PARAMETERS :
C
C     EPMACH          RELATIVE MACHINE PRECISION
C     EPMIN           MINIMUM PRECISION PARAMETER FOR THE IVP SOLVER
C     ETA             RELATIVE DEVIATION FOR NUMERICAL
C                     DIFFERENTIATION
C     FCMIN           MINIMUM VALUE FOR THE RELAXATION FACTOR
C     COND            MAXIMUM LOWER BOUND OF CONDITION NUMBER FOR
C                     THE MULTIPLE SHOOTING MATRIX
C     KBROY           PARAMETER FOR BROYDEN APPROXIMATIONS
C                     0 : ITERATION WITHOUT BROYDEN UPDATING
C                     1 : BROYDEN UPDATING
C
C
C     ************************************************************
C
C
C
C     .. SCALAR ARGUMENTS ..
      DOUBLE PRECISION TOL
      INTEGER ITMAX,KP,KS,M,MMAX,MMS,MS,MSMAX,N,NDIW,NDW,NFILE,NMS
C     ..
C     .. ARRAY ARGUMENTS ..
      DOUBLE PRECISION WORK(NDW),X(*),XS(*),Y(NMS,*),PAR(*)
      INTEGER IWORK(NDIW),JS(MMS,*),IPAR(*)
C     ..
C     .. SUBROUTINE ARGUMENTS ..
      EXTERNAL F,METHOD,R
C     ..
C     .. LOCAL SCALARS ..
      DOUBLE PRECISION COND,EPMACH,EPMIN,ETA,FC,FCMIN,H,H1,S,SABS,SK,
     +                 SUM1A,SUM2A,SUM3A,T,TD,X1,XX1,XXM
      INTEGER I,I1,I2,I3,I4,I5,IFAIL,IK,ITER,K,K1,K10,K11,K12,K13,K14,
     +        K15,K16,K17,K18,K19,K2,K20,K21,K22,K23,K24,K3,K4,K5,K6,K7,
     +        K8,K9,KBROY,NEW
C     ..
C     .. EXTERNAL SUBROUTINES ..
      EXTERNAL EXIT,NEWTON
C     ..
C     .. DATA STATEMENTS ..
C
      DATA EPMACH/1.0D-14/,EPMIN/1.0D-10/,ETA/1.0D-7/,FCMIN/1.0D-2/,
     +     COND/1.0D20/,KBROY/1/
C     ..
C
```

****

INITIAL DATA

0.00000000D+00	0.00000000D+00	0.10000000D+01	0.00000000D+00	0.19400000D+01
0.11111111D+00	0.95802469D-01	0.75444444D+00	0.00000000D+00	0.19400000D+01
0.22222222D+00	0.16765432D+00	0.53888889D+00	0.00000000D+00	0.19400000D+01
0.33333333D+00	0.21555556D+00	0.32333333D+00	0.00000000D+00	0.19400000D+01
0.44444444D+00	0.23950617D+00	0.10777778D+00	0.00000000D+00	0.19400000D+01
0.55555556D+00	0.23950617D+00	-0.10777778D+00	0.00000000D+00	0.19400000D+01
0.66666667D+00	0.21555556D+00	-0.32333333D+00	0.00000000D+00	0.19400000D+01
0.77777778D+00	0.16765432D+00	-0.53888889D+00	0.00000000D+00	0.19400000D+01
0.88888890D+00	0.95802469D-01	-0.75444444D+00	0.00000000D+00	0.19400000D+01
0.10000000D+01	0.00000000D+00	-0.97000000D+00	0.00000000D+00	0.19400000D+01

N= 4 M=10 MS= 0
PRESCRIBED RELATIVE PRECISION 0.10D-04
MAXIMUM PERMITTED NUMBER OF ITERATIONS 30

****

IT	ABS.ERR.	LEVEL1	LEVEL2	LEVEL3	RELAX.	NEW	COND(M)	NORM(M)
0	0.18D-02	0.45D-02	0.46D-02	0.28D-01		0	0.49D+02	0.26D+01
	0.18D-02	0.44D-02	0.45D-02	0.27D-01	0.010			
1	0.18D-02	0.36D-02	0.37D-02	0.23D-01		1	0.50D+02	0.27D+01
	0.27D-25	0.16D-25	0.26D-24	0.23D-23	1.000			
2	0.27D-25	0.15D-25	0.26D-24	0.23D-23		2	0.51D+02	0.27D+01

****

SOLUTION OBTAINED AFTER 3 ITERATION STEPS

SOLUTION DATA

0.00000000D+00	0.00000000D+00	0.10000000D+01	-0.15099717D-13	0.20000000D+01
0.11111111D+00	0.98765432D-01	0.77777778D+00	-0.15099717D-13	0.20000000D+01
0.22222222D+00	0.17283951D+00	0.55555556D+00	-0.15099717D-13	0.20000000D+01
0.33333333D+00	0.22222222D+00	0.33333333D+00	-0.15099717D-13	0.20000000D+01
0.44444444D+00	0.24691358D+00	0.11111111D+00	-0.15099717D-13	0.20000000D+01
0.55555556D+00	0.24691358D+00	-0.11111111D+00	-0.15099717D-13	0.20000000D+01
0.66666667D+00	0.22222222D+00	-0.33333333D+00	-0.15099717D-13	0.20000000D+01
0.77777778D+00	0.17283951D+00	-0.55555556D+00	-0.15099717D-13	0.20000000D+01
0.88888889D+00	0.98765432D-01	-0.77777778D+00	-0.15099717D-13	0.20000000D+01
0.10000000D+01	-0.25087448D-16	-0.10000000D+01	-0.15099717D-13	0.20000000D+01

****

INITIAL DATA

0.00000000D+00	0.00000000D+00 0.93120000D+01	0.10000000D+01	0.46560000D+01	0.31040000D+01
0.11111111D+00	0.89681756D-01 0.93120000D+01	0.65385185D+00	0.46560000D+01	0.25866667D+01
0.22222222D+00	0.14742936D+00 0.93120000D+01	0.39518519D+00	0.46560000D+01	0.20693333D+01
0.33333333D+00	0.17962963D+00 0.93120000D+01	0.19400000D+00	0.46560000D+01	0.15520000D+01
0.44444444D+00	0.19266941D+00 0.93120000D+01	0.50296296D-01	0.46560000D+01	0.10346667D+01
0.55555556D+00	0.19266941D+00 0.93120000D+01	-0.50296296D-01	-0.46560000D+01	0.10346667D+01
0.66666667D+00	0.17962963D+00 0.93120000D+01	-0.19400000D+00	-0.46560000D+01	0.15520000D+01
0.77777778D+00	0.14742936D+00 0.93120000D+01	-0.39518519D+00	-0.46560000D+01	0.20693333D+01
0.88888889D+00	0.89681756D-01 0.93120000D+01	-0.65385185D+00	-0.46560000D+01	0.25866667D+01
0.10000000D+01	0.00000000D+00 0.93120000D+01	-0.97000000D+00	-0.46560000D+01	0.31040000D+01

SWITCHING POINTS

0.48500000D+00

N= 5 M=10 MS= 1
PRESCRIBED RELATIVE PRECISION 0.10D-04
MAXIMUM PERMITTED NUMBER OF ITERATIONS 30

IT	ABS.ERR.	LEVEL1	LEVEL2	LEVEL3	RELAX.	NEW	COND(M)	NORM(M)
0	0.22D-01	0.12D-01	0.15D-01	0.52D-01		0	0.24D+03	0.13D+02
	0.22D-01	0.12D-01	0.15D-01	0.51D-01	0.010			
1	0.22D-01	0.53D-01	0.55D-01	0.47D-01		1	0.27D+03	0.13D+02
	0.17D-04	0.23D-03	0.24D-03	0.23D-03	1.000			
2	0.17D-04	0.23D-03	0.23D-03	0.23D-03		2	0.27D+03	0.13D+02
	0.98D-08	0.68D-07	0.70D-07	0.66D-07	1.000			
3	0.98D-08	0.66D-07	0.68D-07	0.67D-07		3	0.27D+03	0.14D+02
	0.67D-11	0.13D-08	0.13D-08	0.71D-09	1.000			
4	0.67D-11	0.13D-08	0.13D-08	0.58D-09		4	0.26D+03	0.14D+02
	0.23D-12	0.50D-11	0.50D-11	0.95D-12	1.000			
5	0.23D-12	0.50D-11	0.50D-11	0.11D-11		5	0.27D+03	0.14D+02

SOLUTION OBTAINED AFTER 6 ITERATION STEPS

SOLUTION DATA

0.00000000D+00	0.00000000D+00 0.96000001D+01	0.10000000D+01	0.48000000D+01	0.32000000D+01
0.11111111D+00	0.92455418D-01 0.96000001D+01	0.67407407D+00	0.48000000D+01	0.26666667D+01
0.22222222D+00	0.15198903D+00 0.96000001D+01	0.40740741D+00	0.48000000D+01	0.21333333D+01
0.33333333D+00	0.18518519D+00 0.96000001D+01	0.20000000D+00	0.48000000D+01	0.16000000D+01
0.44444444D+00	0.19862826D+00 0.96000001D+01	0.51851852D-01	0.48000000D+01	0.10666667D+01
0.55555556D+00	0.19862826D+00 0.96000001D+01	-0.51851856D-01	-0.48000001D+01	0.10666666D+01
0.66666670D+00	0.18518519D+00 0.96000001D+01	-0.20000000D+00	-0.48000001D+01	0.16000000D+01
0.77777778D+00	0.15198903D+00 0.96000001D+01	-0.40740741D+00	-0.48000001D+01	0.21333333D+01
0.88888890D+00	0.92455418D-01 0.96000001D+01	-0.67407407D+00	-0.48000001D+01	0.26666667D+01
0.10000000D+01	-0.18358592D-17 0.96000001D+01	-0.10000000D+01	-0.48000001D+01	0.32000000D+01

SWITCHING POINTS

0.50000000D+00

****

INITIAL DATA

0.00000000D+00	0.00000000D+00	0.10000000D+01	0.21555556D+02	0.64666667D+01
	0.43111111D+02	0.86222222D+01		
0.11111111D+00	0.72788193D-01	0.38454047D+00	0.21555556D+02	0.40716049D+01
	0.43111111D+02	0.86222222D+01		
0.22222222D+00	0.95309658D-01	0.65198903D-01	0.21555556D+02	0.16765432D+01
	0.43111111D+02	0.86222222D+01		
0.33333333D+00	0.97000000D-01	0.00000000D+00	-0.21555556D+02	-0.79037037D+01
	0.43111111D+02	0.86222222D+01		
0.44444444D+00	0.97000000D-01	0.00000000D+00	-0.21555556D+02	-0.55086420D+01
	0.43111111D+02	0.86222222D+01		
0.55555556D+00	0.97000000D-01	0.00000000D+00	-0.21555556D+02	-0.31135802D+01
	0.43111111D+02	0.86222222D+01		
0.66666667D+00	0.97000000D-01	0.00000000D+00	-0.21555556D+02	-0.71851852D+00
	0.43111111D+02	0.86222222D+01		
0.77777778D+00	0.95309658D-01	-0.65198903D-01	-0.21555556D+02	0.16765432D+01
	0.43111111D+02	0.86222222D+01		
0.88888889D+00	0.72788193D-01	-0.38454047D+00	-0.21555556D+02	0.40716049D+01
	0.43111111D+02	0.86222222D+01		
0.10000000D+01	0.00000000D+00	-0.97000000D+00	-0.21555556D+02	0.64666667D+01
	0.43111111D+02	0.86222222D+01		

SWITCHING POINTS

0.29100000D+00 0.67900000D+00

N= 6 M=10 MS= 2
PRESCRIBED RELATIVE PRECISION 0.10D-04
MAXIMUM PERMITTED NUMBER OF ITERATIONS 30

IT	ABS.ERR.	LEVEL1	LEVEL2	LEVEL3	RELAX.	NEW	COND(M)	NORM(M)
0	0.40D+00	0.27D+00	0.33D-01	0.13D+00		0	0.18D+04	0.61D+02
	0.39D+00	0.26D+00	0.33D-01	0.13D+00	0.010			
1	0.39D+00	0.26D+00	0.25D-01	0.14D+00		1	0.24D+04	0.61D+02
	0.10D-02	0.63D-02	0.68D-02	0.39D-01	1.000			
2	0.10D-02	0.61D-02	0.58D-02	0.12D-01		2	0.23D+04	0.62D+02
	0.78D-05	0.70D-03	0.73D-03	0.13D-02	1.000			
3	0.78D-05	0.68D-03	0.75D-03	0.27D-02		3	0.24D+04	0.63D+02
	0.85D-06	0.76D-05	0.75D-05	0.26D-04	1.000			
4	0.85D-06	0.76D-05	0.78D-05	0.34D-04		4	0.24D+04	0.63D+02
	0.49D-09	0.67D-07	0.75D-07	0.28D-06	1.000			
5	0.49D-09	0.67D-07	0.74D-07	0.25D-06		5	0.24D+04	0.63D+02
	0.76D-11	0.54D-09	0.59D-09	0.20D-08	1.000			
6	0.76D-11	0.54D-09	0.60D-09	0.24D-08		6	0.24D+04	0.63D+02
	0.83D-14	0.61D-12	0.66D-12	0.22D-11	1.000			
7	0.83D-14	0.61D-12	0.66D-12	0.21D-11		7	0.24D+04	0.63D+02

SOLUTION OBTAINED AFTER 8 ITERATION STEPS

SOLUTION DATA

0.00000000D+00	0.00000000D+00	0.10000000D+01	0.22222222D+02	0.66666667D+01
	0.44444445D+02	0.88888889D+01		
0.11111111D+00	0.75039374D-01	0.39643347D+00	0.22222222D+02	0.41975309D+01
	0.44444445D+02	0.88888889D+01		
0.22222222D+00	0.98257379D-01	0.67215363D-01	0.22222222D+02	0.17283951D+01
	0.44444445D+02	0.88888889D+01		
0.33333333D+00	0.10000000D+00	0.38888689D-22	-0.22222222D+02	-0.81481482D+01
	0.44444445D+02	0.88888889D+01		
0.44444444D+00	0.10000000D+00	0.38736780D-22	-0.22222222D+02	-0.56790124D+01
	0.44444445D+02	0.88888889D+01		
0.55555556D+00	0.10000000D+00	0.35394783D-22	-0.22222222D+02	-0.32098766D+01
	0.44444445D+02	0.88888889D+01		
0.66666667D+00	0.10000000D+00	0.32508513D-22	-0.22222222D+02	-0.74074075D+00
	0.44444445D+02	0.88888889D+01		
0.77777778D+00	0.98257379D-01	-0.67215363D-01	-0.22222222D+02	0.17283951D+01
	0.44444445D+02	0.88888889D+01		
0.88888889D+00	0.75039374D-01	-0.39643347D+00	-0.22222222D+02	0.41975309D+01
	0.44444445D+02	0.88888889D+01		
0.10000000D+01	-0.15316614D-16	-0.10000000D+01	-0.22222222D+02	0.66666667D+01
	0.44444445D+02	0.88888889D+01		

SWITCHING POINTS

0.30000000D+00 0.70000000D+00