ISSN 0965-5425, Computational Mathematics and Mathematical Physics, 2021, Vol. 61, No. 2, pp. 177–193. © Pleiades Publishing, Ltd., 2021.
Russian Text © The Author(s), 2021, published in Zhurnal Vychislitel'noi Matematiki i Matematicheskoi Fiziki, 2021, Vol. 61, No. 2, pp. 189–205.

OPTIMAL CONTROL

Multimethod Algorithms for Solving Complicated Optimal Control Problems

A. I. Tyatyushkin[a]

[a] *Institute of System Dynamics and Control Theory, Siberian Branch, Russian Academy of Sciences, Irkutsk, 664033 Russia*

e-mail: tjat@icc.ru

Received December 19, 2019; revised August 20, 2020; accepted September 16, 2020

Abstract—Optimal control problems with terminal conditions without control constraints, problems with a free right trajectory end and with control constraints, and optimization problems with parameters and parameter and control constraints are considered. For each of these classes of problems, multimethod algorithms involving numerical optimal control methods that are most efficient for the given class are designed. The performance of the proposed algorithms is demonstrated using numerical solutions of complicated real-world problems.

Keywords: multimethod optimization algorithms, optimal control, problems with parameters, gradient methods, maximum principle, numerical methods, iteration

DOI: 10.1134/S0965542521020123

INTRODUCTION

According to the multimethod approach for solving optimal control problems, several iterative optimization methods are used in parallel to find the solution of the same problem. The basic difficulty arising when the multimethod approach is applied to the numerical solution of optimal control problems is associated with choosing an efficient method for continuing the optimization process after the convergence of the current method has worsened. Modern operating systems make it possible to organize parallel computational threads for parallel computations based on several methods. Each thread can implement an iterative process for a single optimization method, and a problem can be solved simultaneously by applying several methods. On multiprocessor computers, it is more convenient to implement each method on an individual processor. After the current approximation is found, all the methods are estimated, for example, in terms of the increment of the functional, the most efficient of them is chosen to continue the optimization process, and the approximation produced by this method is sent out to the other methods as an initial approximation for executing the next iteration.

Continuing the iterative process until obtaining an approximation that satisfies the optimality criterion with prescribed accuracy, we find an approximate solution of the problem. This solution is found by a multimethod algorithm consisting of a sequence of steps of different methods used in the optimization process to accelerate its convergence. For example, when three methods are used in parallel (see Fig. 1), the best approximation is determined by the maximum increment of the functional produced by each of them at a given iteration step:

$$u_{i_0} = \operatorname*{argmax}_{i\in\{1,2,3\}} (I(u_i^k) - I(u_i^{k-1})).$$

Then this approximation is sent out to all three methods for executing the next iteration: $u_i^{k+1} = u_{i_0}$, $i = 1, 2, 3$.

Thus, the multimethod approach for solving real-world optimal control problems is implemented in the form of parallel iterative optimization processes with the choice of the best approximation; this approach yields solutions with automatic application of different optimization methods, thus significantly enhancing the efficiency and reliability of numerical solutions found with prescribed accuracy in optimal control applications.

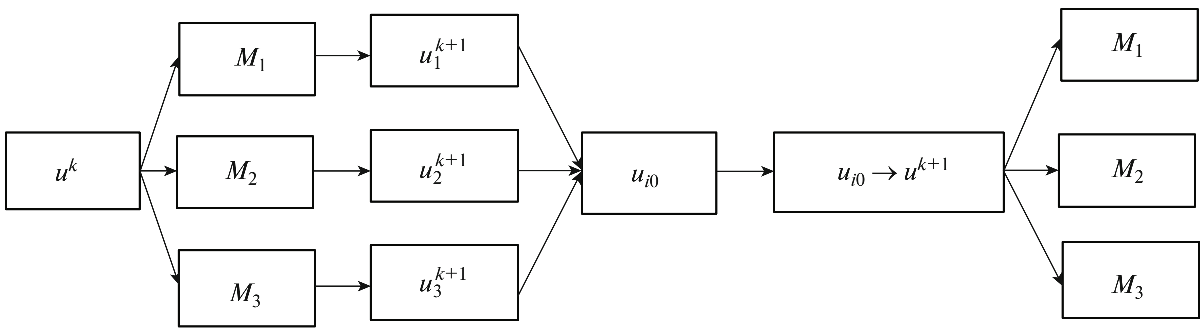

Fig. 1. Schematic of executing the $(k+1)$th iteration in a multimethod algorithm consisting of three methods: M_1, M_2, and M_3.

1. PROBLEM WITHOUT CONTROL CONSTRAINTS

First, we consider an optimal control problem (see [1]) with equality constraints and without control constraints:

$$x = f(x,u,t), \quad t \in T = [t_0, t_1], \quad x(t) \in R^n, \quad u(t) \in R^r, \quad x(t_0) = x^0, \tag{1.1}$$

$$I_0(u) \to \min, \tag{1.2}$$

$$I_j(u) = 0, \quad j = \overline{1,m}, \tag{1.3}$$

where

$$I_j(u) = \varphi^j(x(t_1)) + \int_{t_0}^{t_1} F^j(x,u,t)dt, \quad m \le n.$$

The gradients of functionals (1.2), (1.3) are given by

$$\nabla I_j(u) = -H_u^j(\psi, x, u, t), \tag{1.4}$$

where $H^j(\psi,x,u,t) = \psi'_j(t)f(x,u,t) - F^j(x,u,t)$ is the Pontryagin function from [1] and $\psi_j(t)$ is the solution of the adjoint system

$$\psi_j = -f'_x(x,u,t)\psi_j + F_x^j(x,u,t), \quad \psi(t_1) = -\psi_x^j(x(t_1)). \tag{1.5}$$

Consider a numerical method for solving problem (1.1)–(1.3) based on applying the first and second variations. At the first phase of this method, the iterative process is implemented with strongly violated terminal conditions and the penalty functional to be minimized is

$$J(u) = \varphi(x(t_1)) + \int_{t_0}^{t_1} F^0(x,u,t)dt, \tag{1.6}$$

where

$$\varphi(x(t_1)) = \varphi^0(x(t_1)) + \sum_{j=1}^{m} K_j[\varphi^j(x(t_1)) + x_{n+j}(t_1)]^2,$$

here, $K_j \ge 0$ and $x_{n+j}(t_1)$, $j = \overline{1,m}$, are the solutions of the additional equations (to system (1.1))

$$\dot{x}_{n+j} = F^j(x,u,t), \quad x_{n+j}(t_0) = 0, \quad j = \overline{1,m}.$$

After the iteration at the first phase ceases to converge, we pass to the second phase of the method, at which the original functional (1.2) is minimized and the variation $\delta u(t)$ is constructed taking into account the linearized boundary conditions.

Now suppose that $H(\psi,x,u,t) = \psi' f(x,u,t) - F^0(x,u,t)$,

$$\dot{\psi} = -H_x(\psi,x,u,t), \quad \psi(t_1) = -\varphi_x(x(t_1)), \tag{1.7}$$

and x^k and ψ^k are the solutions of systems (1.1) and (1.7) found for a given control $u^k(t)$. Then the problem of constructing a suitable variation $\delta u^k(t)$ is stated in the form of the linear-quadratic problem

$$I(\delta u) = \frac{1}{2}\delta x'(t_1)\varphi_{xx}(x^k(t_1))\delta x(t_1) - \int_{t_0}^{t_1} H_u'\delta u dt - \frac{1}{2}\int_{t_0}^{t_1}[\delta u' H_{uu}\delta u + 2\delta u' H_{ux}\delta x + \delta x' H_{xx}\delta x]dt \to \min, \quad (1.8)$$

$$\delta\dot{x} = f_x(x^k, u^k, t)\delta x + f_u(x^k, u^k, t)\delta u, \quad \delta x(t_1) = 0. \quad (1.9)$$

Here, H_u, H_{uu}, H_{ux}, and H_{xx} are $r\times 1$, $r\times r$, $r\times n$, and $n\times n$ matrices of partial derivatives of H computed for the control $u^k(t)$ and the trajectories $x^k(t)$, $\psi^k(t)$.

For this problem, the Hamiltonian is defined as

$$\mathcal{H}(\delta\psi, \delta x, \delta u, t) = \delta\psi' f_x\delta x + \delta\psi' f_u\delta u + H_u'\delta u + \frac{1}{2}(\delta u' H_{uu}\delta u + 2\delta u' H_{ux}\delta x + \delta x' H_{xx}\delta x)$$

and the adjoint system is given by

$$\delta\dot{\psi} = -f_x'\delta\psi - H_{ux}'\delta u - H_{xx}\delta x, \quad \delta\psi(t_1) = -\varphi_{xx}\delta x(t_1). \quad (1.10)$$

The condition $\mathcal{H}_{\delta u} = 0$ is used to find the solution of variational problem (1.8), (1.9):

$$\delta u = -H_{uu}^{-1}(f_u'\delta\psi + H_{ux}\delta x + H_u). \quad (1.11)$$

Substituting this formula into Eqs. (1.9) and (1.10) yields the linear two-point boundary value problem

$$\delta\dot{x} = C\delta x - f_u H_{uu}^{-1} f_u'\delta\psi - f_u H_{uu}^{-1} H_u, \quad (1.12)$$

$$\delta\dot{\psi} = -(H_{xx} - H_{ux}' H_{uu}^{-1} H_{ux})\delta x - C'\delta\psi + H_{ux} H_{uu}^{-1} H_u, \quad (1.13)$$

where

$$\begin{gathered} C = f_x - f_u H_{uu}^{-1} H_{ux}, \\ \delta x(t_0) = 0, \quad \delta\psi(t_1) = -\varphi_{xx}\delta x(t_1). \end{gathered} \quad (1.14)$$

A standard method for solving the problem consists in applying the Cauchy formula, which relates the boundary conditions with the help of the $2n\times 2n$ transition matrix $\Phi(t_0, t_1)$:

$$\begin{pmatrix}\delta x(t_1)\\ \delta\psi(t_1)\end{pmatrix} = \Phi(t_0, t_1)\begin{pmatrix}\delta x(t_0)\\ \delta\psi(t_0)\end{pmatrix} + \begin{pmatrix}\rho(t_1)\\ \eta(t_1)\end{pmatrix},$$

where $(\rho(t), \eta(t))$ is the solution of the Cauchy problem for system (1.12), (1.13) with $\delta x(t_1) = \delta x_0 = 0$ and $\delta\psi(t_0) = \delta\psi_0 = 0$.

After splitting the matrix $\Phi(t_0, t_1)$ into four equal blocks and taking into account the equality $\delta x(t_0) = 0$, the last equation can be rewritten as

$$\begin{gathered} \delta x(t_1) = \Phi_{12}\delta\psi_0 + \rho(t_1), \\ \delta\psi(t_1) = \Phi_{22}\delta\psi_0 + \eta(t_1). \end{gathered} \quad (1.15)$$

Substituting boundary condition (1.14) into system (1.15), we obtain the following equation for determining the initial values $\delta\psi_0$:

$$(\Phi_{22} + \varphi_{xx}\Phi_{12})\delta\psi_0 = -\varphi_{xx}\rho(t_1) - \eta(t_1). \quad (1.16)$$

The blocks Φ_{12} and Φ_{22} can be computed by integrating the matrix equation

$$\dot{\Phi} = A(t)\Phi, \quad \Phi(t_0, t_0) = E,$$

where $A(t)$ is the $2n\times 2n$ coefficient matrix of system (1.12), (1.13).

Another method, which requires half as much CPU time, can be described as follows.

Setting $\delta x(t_0) = 0$ and $\delta\psi(t_0) = e^i$, $i = \overline{1,n}$ (where e^i are the vectors of the standard orthonormal basis), we integrate the $2n$-dimensional system (1.12), (1.13) n times. Each of the vectors obtained by the integration is used as the $(n+i)$th column of the matrix Φ; as a result, we obtain its blocks Φ_{12} and Φ_{22}.

Solving the system of linear algebraic equations (1.16) yields a vector $\delta\psi_0$. Next, integrating system (1.12), (1.13) in direct time and applying formula (1.11) at the solution $(\delta x(t), \delta\psi(t))$, we find the desired variation $\delta u(t)$. Formula (1.11) is applicable only if the matrix H_{uu} is sufficiently well conditioned. To preserve the numerical stability of the method in the general case, the matrix H_{uu} is replaced in practice by $H_{uu} + W$, where W is a positive definite matrix. The found variation is used to construct a new approximation $u^k + \alpha_k\delta u^k$, where $\alpha_k = \operatorname*{argmin}_{\alpha\geq 0} J(u^k + \alpha\delta u^k)$. The iterative process of the first phase of the method terminates when $J(u^k) - J(u^{k+1}) \leq \varepsilon, \varepsilon \geq 0$. Since we minimized the penalty functional (1.6), the required accuracy of satisfying the boundary conditions may not be attained. Then we pass to the second phase, which solves problem (1.12), (1.13) taking into account the linearized boundary conditions (1.3).

Assume that, after introducing additional equations into system (1.1), boundary conditions (1.3) are reduced to the form

$$\varphi^i(x(t_1)) = 0, \quad i = \overline{1,m}, \quad m \leq n, \tag{1.17}$$

or $\varphi(x(t_1)) = 0$, where $\varphi = (\varphi^1, \varphi^2, \ldots, \varphi^m)'$ and the functional is given by

$$I_0(u) = \varphi^0(x(t_1)).$$

The boundary conditions (1.17) are linearizes around the point $x^k(t_1)$:

$$\varphi(x^k(t_1)) + \varphi_x(x^k(t_1))\delta x(t_1) = 0. \tag{1.18}$$

Some of the components δx_i^f, $i = \overline{1,m}$, of the vector $\delta x(t_1)$ are related by equalities (1.18), while the remaining $n - m$ free components have to satisfy the terminal conditions

$$\delta\psi^c(t_1) = -\varphi^0_{xx}(x^k(t_1))\delta x^0(t_1). \tag{1.19}$$

Splitting the above-introduced matrices Φ_{12} and Φ_{22} into blocks according to the components of the vectors $\delta x(t_1) = (\delta x^f, \delta x^c)$ and $\delta\psi(t_1) = (\delta\psi^f, \delta\psi^c)$, we rewrite Eqs. (1.15) as

$$\begin{aligned}
\Phi^{11}_{12}\delta\psi^f_0 + \Phi^{12}_{12}\delta\psi^c_0 + \rho^f &= \delta x^f,\\
\Phi^{21}_{12}\delta\psi^f_0 + \Phi^{22}_{12}\delta\psi^c_0 + \rho^c &= \delta x^c_1,\\
\Phi^{11}_{12}\delta\psi^f_0 + \Phi^{12}_{12}\delta\psi^c_0 + \eta^f &= \delta\psi^f_1,\\
\Phi^{21}_{12}\delta\psi^f_0 + \Phi^{22}_{12}\delta\psi^c_0 + \eta^c &= \delta\psi^c_1.
\end{aligned}$$

Substituting boundary conditions (1.18), (1.19) into these equations gives a system of linear algebraic equations for the initial value vector $\delta\psi_0 = (\delta\psi^f_0, \delta\psi^c_0)$:

$$\begin{aligned}
(\Phi^{21}_{22} + \varphi^0_{xx}\Phi^{21}_{12})\delta\psi^f_0 + (\Phi^{22}_{22} + \varphi^0_{xx}\Phi^{22}_{12})\delta\psi^c_0 &= -\varphi^0_{xx}\rho^0 - \eta^c,\\
(\varphi_{x^f}\Phi^{11}_{12} + \varphi_{x^c}\Phi^{21}_{12})\delta\psi^f_0 + (\varphi_{x^f}\Phi^{12}_{12} + \varphi_{x^c}\Phi^{22}_{12})\delta\psi^c_0 &= -\varphi_x\rho - \varphi,
\end{aligned} \tag{1.20}$$

where

$$\varphi^0_{xx} = \varphi^0_{xx}(x^k(t_1)), \quad \varphi_x = (\varphi_{x^f}, \varphi_{x^c}) = \varphi_x(x^k(t_1)), \quad \varphi = \varphi(x^k(t_1)).$$

Therefore, the variation $\delta u(t)$ constructed at the solution $(\delta x(t), \delta\psi(t))$ of system (1.12), (1.13) with initial values $(0, \delta\psi_0)$ satisfies boundary conditions (1.17) in the linear approximation and simultaneously minimizes the quadratic approximation of functional (1.2).

Algorithm 1

Let us describe the numerical scheme including both phases of the described method. The first phase consists of the following steps.

Step 1. Given the control $u^k(t)$, integrate Eq. (1.1) and store the trajectory $x^k(t)$ at integration nodes.

Step 2. Integrate Eq. (1.7) in reverse time and store the solution $\psi^k(t)$ at integration nodes.

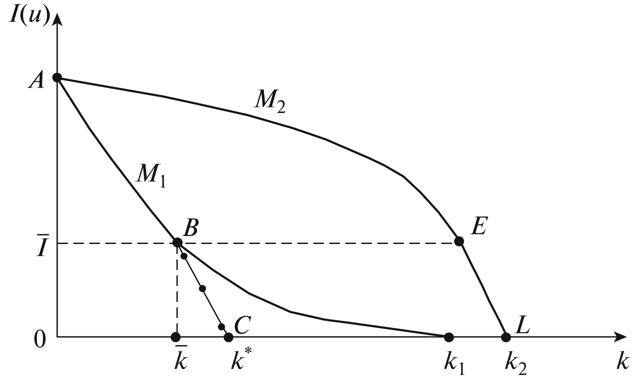

Fig. 2. Decrease in the functional at iterations of the methods M_1, M_2, and l in the multimethod algorithm.

Step 3. With the coefficients computed using the control $u^k(t)$ and the solutions $x^k(t)$, $\psi^k(t)$, integrate system (1.12), (1.13) $n+1$ times with initial conditions

$$\delta x(t_0) = 0, \quad \delta\psi(t_0) = e^i, \quad i = \overline{1,n}; \quad \delta x(t_0) = 0, \quad \delta\psi(t_0) = 0.$$

Set the $(n + j)$th columns of the matrix Φ equal to $(\delta x'(t_1), \delta\psi'(t_1))'$, $j = \overline{1,n}$.

Step 4. Use the blocks Φ_{12} and Φ_{22} of the matrix Φ to form system (1.16) and find its solution $\delta\psi^0$.

Step 5. Integrate system (1.12), (1.13) with $\delta x(t_0) = 0$ and $\delta\psi(t_0) = \delta\psi^0$, compute the variation $\delta u^k(t)$ at each integration node by applying formula (1.11), and store the results.

Step 6. Find the parameter $\alpha_k = \operatorname*{argmin}_{\alpha\geq 0} J(u^k + \alpha\delta u^k)$ by solving ν Cauchy problems (1.1) with the use of linesearch.

Step 7. Construct the new approximation $u^{k+1}(t) = u^k(t) + \alpha_k\delta u^k(t), t \in T$. If $J(u^k) - J(u^{k+1}) > \varepsilon$, then go to Step 1 with $k = k+1$; otherwise, go to the second phase of the algorithm.

The basic difference of the second phase is that the vector $\delta\psi^0$ is determined by solving system (1.20) rather than system (1.16). The second phase consists of the same Steps 1–7, except for Step 4, where system (1.20) is formed instead of (1.16), and except for Step 2, where Eq. (1.5) with $j = 0$ is integrated instead of (1.7).

In fact, the second phase of Algorithm 1 is a version of the quasilinearization method (see [2]), which converges quadratically, but requires a rather good initial approximation, which is provided by the first phase of the algorithm. Thus, the most efficient algorithm is one in which the sequence of approximations is produced by two optimization methods if we are able to determine a suitable moment of switching between them.

Graphically, the decrease in the functional $I(u)$ at iterations of the multimethod algorithm is depicted by the polygonal line made up of the plots of the constituent methods. Figure 2 demonstrates the performance of the multimethod algorithm in the case of two methods, M_1 and M_2. The plots present the decrease in the functional at iterations of these methods. Then these plots are combined to show the decrease in the functional at iterations of the multimethod algorithm (see the curve ABC, where the segment BC is obtained by parallel translation of the curve EL). According to Fig. 2, a zero value of the functional is reached after k_1 iterations of the method M_1 and k_2 iterations of the method M_2. The multimethod algorithm follows the method M_1 until the $\bar{k}$th iteration (curve AB) and then switches to the method M_2 (curve BC), since, starting at $\bar{k}$, the rate of decaying the functional in M_2 is higher. As a result, a zero value of the functional in the multimethod algorithm is reached after k^* iterations, where k^* is much fewer than the number of iterations required in each of the methods M_1 and M_2.

2. PROBLEMS WITH CONTROL CONSTRAINTS

To solve the practically most important class of problems with control constraints

$$u(t) \in U, \quad t \in T, \tag{2.1}$$

where U is a compact subset of R^r, we construct multimethod algorithms based on the maximum principle (see [3, 4]) and on gradient-type methods (see [5–8]).

Assume that introducing a penalty functional yields a problem with a free right end. Suppose that the solution $x^k(t)$ of Eq. (1.1) is found for an admissible control $u^k(t) \in U,\ t \in T$. Solving Eq. (1.5) with $u = u^k(t),\ x = x^k(t),\ j = 0$, we find $\psi^k = \psi_0(t)$ and compute

$$\bar{u}^k(t) = \operatorname*{argmax}_{u\in U} H(\psi^k, x^k, u, t), \quad t \in T. \tag{2.2}$$

Define the scalar function

$$w_k(u(t), t) = H(\psi^k, x^k, u, t) - H(\psi^k, x^k, u^k, t), \tag{2.3}$$

which, for $u = \bar{u}(t)$, obviously satisfies the inequality

$$w_k(\bar{u}^k(t), t) \geq 0, \quad t \in T. \tag{2.4}$$

Let $\tau_k \in T$ be a maximizer of the function $w_k(\bar{u}^k(t), t)$:

$$w_k(\bar{u}^k(\tau_k), \tau_k) = \max_{t\in T} w_k(\bar{u}^k(t), t). \tag{2.5}$$

Then the first-order necessary condition (the maximum principle [1, 2]) is formulated as follows: if $u^k(t)$ is an optimal control in problem (1.2), (1.1), (2.1), then

$$w_k(\bar{u}^k(\tau_k), \tau_k) = 0. \tag{2.6}$$

Assume that condition (2.6) does not hold for the given $u^k(t) \in U$ and found $x^k(t),\ \psi^k(t),\ \bar{u}^k(t)$:

$$w_k(\bar{u}^k(\tau_k), \tau_k) > 0.$$

Then we can find a new control for which the value of functional (1.2) is less than $I_0(u^k)$.

The interval $T_\varepsilon^k \subseteq T$ is constructed according to the following rule:

$$T_\varepsilon^k = [\tau_k - \varepsilon(\tau_k - t_0^k), \tau_k + \varepsilon(t_1^k - \tau_k)], \quad \varepsilon \in [0,1], \tag{2.7}$$

where t_0^k and t_1^k are the nearest left and right discontinuity points of the function $w_k(\bar{u}^k(t), t)$. This interval with measure $\operatorname{mes} T_\varepsilon^k = \alpha_k(\varepsilon) = \alpha_k\varepsilon,\ 0 \leq a_k \leq t_1 - t_0,\ \varepsilon \in [0,1]$, has the following properties:

(i) $\alpha_k(\varepsilon) \to 0$ as $\varepsilon \to 0$;

(ii) as $\varepsilon \to 0$, the interval contracts to the point τ_k;

(iii) for all $\varepsilon \in [0,1]$, the function $w_k(\bar{u}^k(t), t)$ is continuous on T_ε.

Next, we find the parameter

$$\varepsilon_k = \operatorname*{argmin}_{\varepsilon\in[0,1]} I_0(u_\varepsilon^k), \tag{2.8}$$

where

$$u_\varepsilon^k = \begin{cases} \bar{u}^k(t), & t \in T_\varepsilon, \\ u^k(t), & t \in T \setminus T_\varepsilon, \end{cases} \tag{2.9}$$

and determine the new approximation

$$u^{k+1}(t) = u_{\varepsilon_k}^k(t), \quad t \in T, \quad k = 0, 1, \ldots. \tag{2.10}$$

The numerical scheme for the described method can be described as follows. Its convergence was proved in [4].

Algorithm 2

Step 1. Specify a boundary control $u^0(t),\ t \in T$; set $k = 1$.

Step 2. Integrate system (1.1) with $u = u^k(t)$ in direct time and store $x^k(t)$ at integration nodes.

Step 3. Integrate adjoint system (1.5) in reverse time. Specifically, at each integration node, find and store control (2.2), compute the value of the function $w_k(\overline{u}^k(t), t)$, and determine the maximizer τ_k.

Step 4. If $w_k(\overline{u}^k(\tau_k), \tau_k) \le \varepsilon$, then terminate the process. Otherwise, go to Step 5.

Step 5. Solve problem (2.8), (2.9), (2.7) by applying the linesearch procedure. Since control (2.9) is equal to $u^k(t)$ at points $t \in T$ satisfying the inequality $t < \tau_k - \varepsilon(\tau_k - t_0^k)$, the integration of system (1.1) should start at the left node t_k nearest to $\tau_k - \varepsilon(\tau_k - t_0^k)$, at which $x(t_k) = x^k(t_k)$ was found at Step 2. Taking into account the structure of control (2.9), the initial point can also be specified as the "largest" node $t_k \in \{t_k \in T_\varepsilon^k : u^k(t_k) = \overline{u}^k(t_k)\}$. Since the search for ε_k involves repeated integrations of system (1.1), this choice of the initial point can significantly reduce the time required for solving the problem.

Step 6. Given the found ε_k, the accuracy δ of the computed $I_0(u^k)$, and the step size h, if $I_0(u^k_{\varepsilon_k}) \ge I_0(u^k) - \delta$ and $t_1^s - t_0^s > h$, then reduce the interval T_ε^k by setting $\varepsilon = 2^{-s}$ (s is the number of reductions in T_ε^k) and go to Step 5.

If $I_0(u^k_{\varepsilon_k}) < I_0(u^k) - \delta$, then set $u^{k+1} = u^k_{\varepsilon k}$, $k = k + 1$ and go to Step 2. Otherwise, terminate the iterative process.

Since the iterations of the algorithm are performed in the class of piecewise constant controls, the optimality condition (2.8) for the resulting control may not be satisfied with the prescribed accuracy (although the value of $\int_T w_k(u^k(t), t)dt$ is sufficiently small). In contrast to gradient methods, this algorithm is also applicable to optimal control problems in which the vector function $f(x, u, t)$ is not differentiable with respect to u and the set U is not convex or even connected.

As was noted in [4, 8, 9], algorithms based on the maximum principle often lead to a control stuck on the boundary, which deteriorate their convergence. This effect is caused by the fact that, in certain systems (e.g., in control-linear ones), the solution of problem (1.2) is reached on the boundary and, hence, control approximations of the form (2.9), (2.10) also have boundary values. With this approximation, the convergence of the algorithm at the last iterations is ensured by a very small step size, which leads to excessive CPU times. The convergence of the iterative process in this situation can often be recovered or improved by applying a simpler technique, namely, by constructing a convex combination of the controls $u^k(t)$ and $\overline{u}^k(t)$:

$$u^{k+1}(t) = u^k(t) + \alpha_k[\overline{u}^k(t) - u^k(t)], \quad \alpha_k \in [0, 1], \tag{2.11}$$

on the interval T_ε^k.

The ceasing of convergence can also be caused by the fact that the integration nodes remaining within the reduced interval T_ε^k are too few to construct a control variation for which the functional increment is greater (in absolute value) than the integration errors.

Consider a numerical method in which a new approximation is specified as a control generated using the maximum principle on the set

$$T_\varepsilon = \{t \in T : w_k(\overline{u}^k(t), t) \ge \varepsilon w_k(\overline{u}^k(t_k), \tau_k)\}, \quad \varepsilon \in [0, 1]. \tag{2.12}$$

The set T includes all points $t \in T$ of violation of the maximum principle and consists of several disjoint intervals if the function $w_k(\overline{u}^k(t), t)$ has several extrema. By varying ε, it is possible to find ε_k for which control (2.9) minimizes the functional. For many problems, the convergence of Algorithm 2 is improved when interval (2.7) is replaced by set (2.12). When the maximum of $w_k(\overline{u}^k(t), t)$ is reached on a set T_{ε_k} of positive measure, the algorithm may cease to converge. Then the iterations of Algorithm 2 are continued with T_ε^k given by formula (2.7).

Finally, the numerical scheme for constructing a new approximation can be described as follows.

Algorithm 3

Step 1. Integrate Eq. (1.1) with $u = u^k(t)$ and store the trajectory $x^k(t)$ at integration nodes.

Step 2. Integrate the adjoint system (1.5) in reverse time and, at each integration node, compute and store the control $\bar{u}^k(t)$ and the scalar function

$$w_k(t) = w_k(\bar{u}^k(t), t).$$

Step 3. Find the point $\tau_k = \underset{t\in T}{\operatorname{argmax}}\, w_k(t)$. If $w_k(\tau_k) \le \varepsilon$, then terminate the iterative process.

Step 4. Solve the linesearch problem $I(u_\varepsilon^k) \to \min$. Specifically, for each $\varepsilon \in [0,1]$ used in the linesearch procedure, integrate Eq. (1.1) starting at the node t_k where the inequality $w_k(t) > \varepsilon w_k(\tau_k)$ holds for the first time, and choose the control by using the formula

$$u_\varepsilon^k(t) = \begin{cases} \bar{u}^k(t) & \text{if} \quad w_k(t) \ge \varepsilon w_k(\tau_k), \\ u^k(t) & \text{if} \quad w_k(t) < \varepsilon w_k(\tau_k), \quad t \in T. \end{cases}$$

Step 5. If $I_0(u_{\varepsilon_k}^k) \ge I_0(u^k) - \delta$ and $\operatorname{mes} T_{\varepsilon_k} > h$, then execute Step 5 of Algorithm 2, where T_ε^k is given by formula (2.7) and the first approximation is specified as T_{ε_k}.

Step 6. If the iteration of Algorithm 2 also does not improve the control $u^k(t)$, then, on the interval $T_{\varepsilon_k}^k$ obtained at Step 4, perform an iteration of the conditional gradient method (2.11), where α_k is again determined using linesearch.

Step 7. If Steps 4–6 produce ε_k or α_k for which the new control approximation ensures a smaller value of the functional, then go to Step 1 (with $k := k+1$). Otherwise, terminate the iterative process.

Note that this algorithm, like the one described above, is intended for problems with control constraints and a free right end.

3. GENERAL OPTIMAL CONTROL PROBLEM WITH PARAMETERS

Now we consider a more general optimal control problem, namely, one with state constraints and a right-hand side of the system depending not only on controls, but also on parameters. The initial values of the system can also depend on parameters, and their choice usually ensures, for example, an optimal "start" of the process.

Before solving this complicated problem, we first reduce it to a finite-dimensional one and then construct a multimethod algorithm for finding an optimal control.

Consider a controlled process depending on parameters, namely,

$$\begin{gathered} \dot{x} = f(x,u,w,t), \quad x(t) \in E^n, \quad u(t) \in E^r, \quad t \in T = [t_0, t_1], \\ x(t_0) = \Theta(v), \quad w \in R^p, \quad v \in R^n, \end{gathered} \tag{3.1}$$

with terminal conditions

$$I_i(u) = h_i(x(t_1)) = 0, \quad i = \overline{1,m}, \tag{3.2}$$

and state constraints

$$J_i(u,v) = g_i(x(t),t) = 0, \quad t \in T, \quad i = \overline{1,s}. \tag{3.3}$$

The control and parameters obey the constraints

$$c_i(u,t) = 0, \quad t \in T, \quad i = \overline{1,l}, \tag{3.4}$$

$$u^{\mathrm{l}}(t) \le u(t) \le u^{\mathrm{u}}(t), \quad t \in T, \tag{3.5}$$

$$v^{\mathrm{l}} \le v \le v^{\mathrm{u}}, \quad w^{\mathrm{l}} \le w \le w^{\mathrm{u}}, \tag{3.6}$$

where the functions $c_i(u,t)$, $i = \overline{1,l}$, are continuously differentiable with respect to u and piecewise continuous in t, while $\Theta(v)$ is a continuously differentiable vector function. The functions determining conditions (3.1)–(3.3) satisfy the assumptions stated above; additionally, they are assumed to be continuously differentiable with respect to the parameters.

The task is, among the controls and parameters satisfying constraints (3.4)–(3.6), to find ones that ensure the fulfillment of conditions (3.3) for controlled process (3.1) and drive it to a state point where conditions (3.2) hold with the prescribed accuracy and the functional

$$I_0(u) = \varphi(x(t_1)) \tag{3.7}$$

reaches its smallest value.

3.1. Reduction to a Finite-Dimensional Problem

To construct a finite-dimensional problem, on the given interval T, we introduce a grid with nodes $t_0, t^1, \ldots, t^N$ such that

$$t_0 = t^0 < t^1 < \ldots < t^N = t_1. \tag{3.8}$$

This grid is allowed to be nonuniform.

The control functions $u^i(t),\ i = \overline{1, r}$, are sought only at nodes (3.8), while the intermediate values $u^i(t),\ i = \overline{1, r}$, are obtained using the piecewise constant approximation

$$u^i(t) = u^i(t^j) = u^i_j, \quad t \in [t^j, t^{j+1}],$$

or the piecewise linear approximation

$$u^i(t) = [(t^{j+1} - t)u_j + (t - t^j)u_{j+1}](t^{j+1} - t^j), \quad t \in [t^j, t^{j+1}]. \tag{3.9}$$

Then the finite-dimensional problem approximating problem (3.1)–(3.7) has the form

$$\begin{gathered}
\dot{x} = f(x, u, w, t), \quad t \in T = [t_0, t_1], \quad x(t_0) = \Theta(v), \\
h_i(x(t^N)) = 0, \quad i = \overline{1, m}, \\
g_i(x(t^j), t^j) = 0, \quad i = \overline{1, s}, \quad j = \overline{0, N}, \\
c_i(u_j, t^j) = 0, \quad i = \overline{1, l}, \quad j = \overline{0, N}, \\
v^{\mathrm{l}} \le v \le v^{\mathrm{u}}, \quad w^{\mathrm{l}} \le w \le w^{\mathrm{u}}, \\
\varphi(x(t^N)) \to \min, \quad u^{\mathrm{l}}_j \le u_j \le u^{\mathrm{u}}_j, \quad j = \overline{0, N},
\end{gathered} \tag{3.10}$$

where

$$u^{\mathrm{l}}_j = u^{\mathrm{l}}(t^j), \quad u^{\mathrm{u}}_j = u^{\mathrm{u}}(t^j), \quad j = \overline{0, N}.$$

Note that the controlled process (3.1) in approximating problem (3.10) remains continuous and, in the course of the computations, it is modeled by the numerical integration method with the required (sufficiently high) accuracy.

3.2. Numerical Solution of the Finite-Dimensional Problem

With the help of the functions $H^j(\psi_j, x, u, t) = \psi'_j(t) f(x, u, t)$ and the adjoint system

$$\dot{\psi}_j = -f_x(x, u, t)' \psi_j(t), \quad \psi_j(t_1) = -\varphi^j_x(x(t_1)),$$

the gradients of the functionals $I_j(u),\ j = \overline{0, m}$, are traditionally defined by the formulas

$$\nabla I_j(u) = -H^j_u(\psi_j, x, u, t), \quad j = \overline{0, m}.$$

For each $t \in T$, the gradients of $J_j(u, t),\ j = \overline{1, s}$, can be computed in a similar manner:

$$\nabla J_j(u, t) = -\bar{H}^j_u(\Phi_j, x, u, t, \tau), \quad t_0 \le \tau \le t \le t_1,$$

where $\bar{H}^j(\Phi_j, x, u, t, \tau) = \Phi'_j(t,\tau) f(x, u, \tau)$ and $\Phi_j(t,\tau)$, $j = \overline{1,s}$, are the solutions of the adjoint system

$$\frac{\partial \Phi_j(t,\tau)}{\partial \tau} = -\frac{\partial f(x,u,\tau)}{\partial x}\Phi_j(t,\tau), \qquad \tau \in [t_0, t],$$

with boundary conditions

$$\Phi_j(t,t) = -\frac{\partial g^j(x(t))}{\partial x}, \qquad j = \overline{1,s}.$$

We linearize the constraints in the approximating problem. The Jacobian matrix of the linearized constraints is made up of the gradients $\nabla I_i, i = \overline{1,m}$, and $\nabla J_j(t)$, $j = \overline{1,s}$, $t \in T$. Since the right-hand sides and initial values of system (3.1) depend additionally on parameters, we need to know the gradients of the functionals $I_i, i = \overline{1,m}$, and $J_j(t)$, $j = \overline{1,s}$, $t \in T$, with respect to these parameters (see [3, 8]):

$$\begin{aligned} \nabla_v I_i(u^k, w^k, v^k) &= -\psi_i(t_0)'\,\Theta_v(v^k), \qquad i = \overline{1,m}, \\ \nabla_w I_i(u^k, w^k, v^k) &= -\int_{t_0}^{t_1} \psi_i(t)' f_w(x^k, u^k, w^k, t)dt, \end{aligned} \tag{3.11}$$

$$\nabla_w J_i(u^k, w^k, v^k, t^j) = -\int_{t_0}^{t^j} \Phi_i(t)' f_w(x^k, u^k, w^k, t)dt, \tag{3.12}$$

$$\nabla_v J_i(u^k, w^k, v^k, t^j) = -\Phi_i(t_0)'\,\Theta(v^k), \qquad i = \overline{1,s}, \qquad j = \overline{1,N}. \tag{3.13}$$

Now suppose that $u^k(t^j)$ and the corresponding $x^k(t^j)$, $j = \overline{1,N}$, have been found on grid (3.8) at the kth iteration of the outer method. To calculate the gradients with respect to the control $\nabla_u I_i, i = \overline{1,m}$, system (3.10) is integrated m times from t_1 to t_0 with various initial conditions. Simultaneously, gradients (3.11) are computed using quadrature rules for evaluating the integrals. Next, we find the gradients of the functionals $J_i(t^j)$, $j = \overline{1,N}$, $i = \overline{1,s}$. For this purpose, the Cauchy problem is solved s times for each node of grid (3.8), i.e., the system is integrated $s \cdot N$ times on average on a half of the interval T.

The resulting solutions are used to compute the components of the gradients $\nabla_u I_i, i = \overline{1,m}$, and $\nabla_u J_i(t^j)$, $i = \overline{1,s}$, $j = \overline{1,N}$. Taking into account the control approximation, their values are equal to

$$\int_{t^j}^{t^{j+1}} \psi^k(t)' f_u(x^k, u_j^k, w^k, t)dt$$

in the case of a piecewise constant approximation and to

$$\frac{1}{t^{j+1} - t^j}\left[\int_{t^{j-1}}^{t^j} \psi^k(t)' f_u(x^k, \bar{u}^k(t), w^k, t)(t - t^{j-1})dt + \int_{t^j}^{t^{j+1}} \psi^k(t)' f_u(x^k, \bar{u}^k(t), w^k, t)(t^{j+1} - t)dt\right] \tag{3.14}$$

in the case of piecewise linear approximation (3.9). Here, $\bar{u}^k(t)$ is computed using formula (3.9) with $u_j = u_j^k$, $u_{j+1} = u_{j+1}^k$.

The resulting values of the control gradients $\nabla I_i, i = \overline{1,m}$, and $\nabla J_j(t)$, $j = \overline{1,s}$, $t \in T$, and the gradients with respect to the parameters computed using formulas (3.11)–(3.13) make up the coefficient matrix of the linearized constraints. It is supplemented with the block of elements $\partial c_i / \partial u_j$ corresponding to the control constraints to become a matrix of special block structure, which is denoted by A.

3.3. Reduced Gradient Algorithm from [6]

After introducing vector notation for equalities (3.2)–(3.4), the augmented Lagrangian (see [7]) for problem (3.1)–(3.7) is defined as

$$
\begin{aligned}
L &= \varphi(x(t_1)) - \lambda^{k'}[h(x(t_1)) - \bar{h}^L] + \frac{\rho}{2}[h(x(t_1)) - \bar{h}^L]'[h(x(t_1)) - \bar{h}^L] - \int_{t_0}^{t_1} \mu^{k'}(t)[g(x(t),t) - \bar{g}^L]dt \\
&+ \frac{\rho}{2}\int_{t_0}^{t_1}[g(x(t),t) - \bar{g}^L]'[g(x(t),t) - \bar{g}^L]dt - \int_{t_0}^{t_1}\gamma^k(t)[c(u,t) - \bar{c}^L]dt + \frac{\rho}{2}\int_{t_0}^{t_1}[c(u,t) - \bar{c}^L]'[c(u,t) - \bar{c}^L]dt,
\end{aligned}
\tag{3.15}
$$

where

$$
\bar{h}^L = h(x^k(t_1)) + h_x(x^k(t_1))\delta x(t_1), \quad \bar{g}^L = g(x^k(t),t) + g_x(x^k(t),t)\delta x(t),
$$

$$
\bar{c}^L = c(u^k(t),t) + c_u(u^k(t),t)\delta u(t), \quad \delta u = u - u^k, \quad \delta x = x - x^k.
$$

Next, constraints (3.2) and (3.3) are linearized at the kth approximation:

$$
I^k + \sum_{j=0}^{N}\nabla_u I^k(t^j)'(u_j - u_j^k) + \nabla_w I^{k'}(w - w^k) + \nabla_v I^{k'}(v - v^k) = 0, \tag{3.16}
$$

$$
J_j^k + \sum_{i=0}^{j}[\nabla_u J^k(t^j)'(u_i - u_i^k) + \nabla_w J^k(t^j)'(w - w^k) + \nabla_v J^k(t^j)'(v - v^k)] = 0, \quad j = \overline{0,N}. \tag{3.17}
$$

Here, $I = (I_1, I_2, \ldots, I_m)$ and $J = (J_1, J_2, \ldots, J_s)$. Therefore, we have m constraints (3.16) and $(N+1)s$ constraints (3.17), which represent linearized (h^L, g_j^L) constraints (3.2) and (3.3) written in explicit form (in terms of u, w, v); moreover, equalities (3.3) specified at each time $t \in T$ are replaced by N equalities determined at nodes of grid (3.8).

Conditions (3.4) are also linearized:

$$
c(u^k, t^j) + \nabla_u c(u^k, t^j)'(u_j - u_j^k) = 0, \quad j = \overline{0,N}, \tag{3.18}
$$

where $c = (c_1, c_2, \ldots, c_l)$. The primal constraints on the control and the parameters remain unchanged:

$$
u_j^{\mathrm{l}} \le u_j \le u_j^{\mathrm{u}}, \quad j = \overline{1,N}, \tag{3.19}
$$

$$
v_j^{\mathrm{l}} \le v_j \le v_j^{\mathrm{u}}, \quad j = \overline{1,n}, \quad w_i^{\mathrm{l}} \le w_i \le w_i^{\mathrm{u}}, \quad i = \overline{1,p}. \tag{3.20}
$$

For functional (3.15) with variables $x(t)$ determined by system (3.1) with a given $u(t)$, $t \in T$, we consider the finite-dimensional approximation

$$
\begin{aligned}
L &= \varphi(x^N) - \lambda^{k'}[h(x(t^N)) - \bar{h}^L] + \frac{\rho}{2}[h(x(t^N)) - \bar{h}^L]'[h(x(t^N)) - \bar{h}^L] - \sum_{j=0}^{N}\mu_j^{k'}[g(x(t^j),t^j) - \bar{g}^L] \\
&+ \frac{\rho}{2}\sum_{j=0}^{N}[g(x(t^j),t^j) - \bar{g}^L]'[g(x(t^j),t^j) - \bar{g}^L] - \sum_{j=0}^{N}\gamma_j^{k'}[c(u_j,t^j) - \bar{c}^L] + \frac{\rho}{2}\sum_{j=0}^{N}[c(u_j,t^j) - \bar{c}^L]'[c(u_j,t^j) - \bar{c}^L],
\end{aligned}
\tag{3.21}
$$

where

$$
\bar{h}^L = h(x^k(t^N)) + h_x(x^k(t^N))(x(t^N) - x^k(t^N)),
$$

$$
\bar{g}^L = g(x^k(t^j),t^j) + g_x(x^k(t^j),t^j)(x(t^j) - x^k(t^j)),
$$

$$
\bar{c}^L = c(u^k(t^j),t^j) + c_u(u^k(t^j),t^j)(u_j - u_j^k), \quad j = \overline{0,N}.
$$

Functional (3.21), which is, in fact, a multivariable function, is minimized under linear constraints (3.16)–(3.20) by applying the reduced gradient method (see [6]). Note that functional (3.21) assumes the use of the original system (3.1) for computing the trajectory $\{x(t^1), x(t^2), \ldots, x(t^N)\}$ for given parameters v, w and control $u(t^0), u(t^1), \ldots, u(t^N)$, i.e., a complete model of the auxiliary problem is described by relations (3.1) and (3.16)–(3.21).

Let $A[m+(l+s)(N+1)]\times[r(N+1)+p+n]$ denote the coefficient matrix of linear equations (3.16)–(3.18), b be the vector of free terms of dimension $m+(l+s)(N+1)$, and z denote the vector of desired variables $(u_j, j=\overline{0,N}; v; w)$ of dimension $r(N+1)+p+n$. Then the problem under study can be written as

$$\begin{gathered} L(z)\to\min, \\ Az=b, \\ z^{\mathrm{l}}\le z\le z^{\mathrm{u}}. \end{gathered} \tag{3.22}$$

Problem (3.22) is solved by applying the reduced gradient method (see [6]), which differs from the simplex method well-known in linear programming in that, in view of the nonlinearity of the objective function, its successive approximations are not necessarily vertices of the linear constraint polyhedron, but may be its interior points.

3.4. Projected Lagrangian Algorithm (see [6–9])

Now we describe the complete algorithm for solving the original problem (3.1)–(3.6).

Step 1. Given the control u_j^k, $j=\overline{0,N}$, integrate system (3.1) and store the state trajectory points x_j^k, $j=\overline{0,N}$, at nodes of grid (3.8). Here, k is the iteration number (starting from $k=0$).

Linearize the constraints of problem (3.10) at the resulting solution and construct auxiliary problem (3.16)–(3.21).

Step 2. Solve the auxiliary problem of minimizing the augmented Lagrangian (3.21) with linear constraints (3.16)–(3.20) by applying the reduced gradient method.

As a result, new approximations for the control u_j^{k+1}, $j=\overline{0,N}$, the parameters w^{k+1} and v^{k+1}, and the dual variables λ^{k+1} and μ_j^{k+1}, $j=\overline{0,N}$, are found.

Step 3. Check the stopping rule for the iterative process with respect to both primal and dual variables:

$$\left|I_i(u^{k+1},w^{k+1},v^{k+1})\right|/(1+\alpha^{k+1})\le\varepsilon,\quad i=\overline{1,m},$$

$$\left|J_i(u^{k+1},w^{k+1},v^{k+1})\right|/(1+\alpha^{k+1})\le\varepsilon,\quad i=\overline{1,s},$$

where

$$\alpha^{k+1}=\max\{\|u_j^{k+1}\|, j=\overline{0,N}; |w_i|, i=\overline{1,p}; |v_l|, l=\overline{1,n}\};$$

$$\left|\lambda_j^k-\lambda_j^{k+1}\right|/(1+\Theta^{k+1})\le\varepsilon,\quad j=\overline{1,m};$$

$$\left|\mu_{ij}^k-\mu_{ij}^{k+1}\right|/(1+\Theta^{k+1})\le\varepsilon,\quad i=\overline{1,s},\quad j=\overline{0,N};$$

$$\Theta^{k+1}=\max\{|\lambda_j^{k+1}|, j=\overline{1,m}; |\mu_{ij}^{k+1}|, i=\overline{1,s}, j=\overline{0,N}\}.$$

If at least one of these conditions is violated, then go to Step 1 to execute the new $(k+1)$th iteration. If these inequalities hold for the given $\varepsilon>0$, then terminate the iterative process and output the found values u_j^{k+1}, $j=\overline{0,N}$, w^{k+1}, and v^{k+1} as an approximate solution of the optimal control problem.

4. NUMERICAL EXPERIMENTS: REAL-WORLD PROBLEMS WITH SOLUTIONS

Below are examples of applications related to the classes of optimization problems considered in Sections 1–3 with solutions found by applying the multimethod algorithms described in these sections.

4.1. Optimal Control of a Spherical Mobile Robot with Three-Dimensional Control Functions

The problem was formulated by M.M. Svinin and can be found in [10]. Consider a mobile spherical robot moving in a plane. The robot consists of a shell with three engines (rotors) mounted on it. The dynamics of the robot in terms of contact coordinates is described by the system of ordinary differential equations

$$\dot{x} = G(x)J^{-1}(x)J_r\sum_{k=1}^{n} n_k(x)u_k,$$

where the state and control vectors are defined as

$$x \triangleq [u_a, v_a, u_o, v_o, \psi]^{\mathrm{T}}, \quad u \triangleq [\dot{\varphi}_1, \dot{\varphi}_2, \dot{\varphi}_3]^{\mathrm{T}},$$

and φ_i, $i = \overline{1,3}$, denote the angles of rotation of the engines.

The position of a point of contact on the plane is specified by the coordinates u_a, v_a, while its coordinates on the sphere are defined by the angles u_o, v_o.

The matrix and vector quantities are defined as

$$G = \begin{bmatrix} 0 & -R & 0 \\ R & 0 & 0 \\ \sin\psi/\cos v_o & \cos\psi/\cos v_o & 0 \\ \cos\psi & -\sin\psi & 0 \\ \sin\psi\tan v_o & \cos\psi\tan v_o & 1 \end{bmatrix},$$

and the vectors n_1, n_2, n_3 are the columns of the matrix

$$R = \begin{bmatrix} \cos u_o\cos\psi + \sin u_o\sin v_o\sin\psi & \cos v_o\sin\psi & -\sin u_o\cos\psi + \cos u_o\sin v_o\sin\psi \\ -\cos u_o\sin\psi + \sin u_o\sin v_o\cos\psi & \cos v_o\cos\psi & \sin u_o\sin\psi + \cos u_o\sin v_o\cos\psi \\ \sin u_o\cos v_o & -\sin v_o & \cos u_o\cos v_o \end{bmatrix}.$$

The matrix of inertia of the system is defined as

$$J = \begin{bmatrix} (2/3m_o + M)R^2 & 0 & 0 \\ 0 & (2/3m_o + M)R^2 & 0 \\ 0 & 0 & 2/3m_oR^2 \end{bmatrix} + (2J_p + J_r)E,$$

where M is the total mass of the robot, while m_o and m_r denote the mass of the spherical shell and the mass of a rotor, respectively.

The problem of optimal control of the spherical robot is to drive it from the point $x(0)$ to the point $x(T)$ so as to minimize the control energy $J = \int_0^T u^T u dt$.

For example, let $x(0) = [0, 0, 0, 0, 0]$ and $x(10) = \left[0, 2, 0, 3, 0, 0, \frac{\pi}{6}\right]$. Then the resulting solutions are physically feasible, and they are presented in Fig. 3. The constraints are satisfied up to 10^{-5}, and the functional value is $J(10) = 5769001$.

4.2. Optimal Control of a Robotic Manipulator

The dynamics of a moving industrial robotic manipulator is described by the system of differential equations

$$\dot{x}_1 = x_2,$$

$$\dot{x}_2 = \frac{[M_1(x,u) - F_1(x)]a_{22} - [M_2(x,u) - F_2(x)]a_{12}(x)}{a_{11}a_{22} - a_{12}(x)a_{21}(x)},$$

$$\dot{x}_3 = x_4,$$

$$\dot{x}_4 = \frac{[M_2(x,u) - F_2(x)]a_{11} - [M_1(x,u) - F_1(x)]a_{21}(x)}{a_{11}a_{22} - a_{12}(x)a_{21}(x)},$$

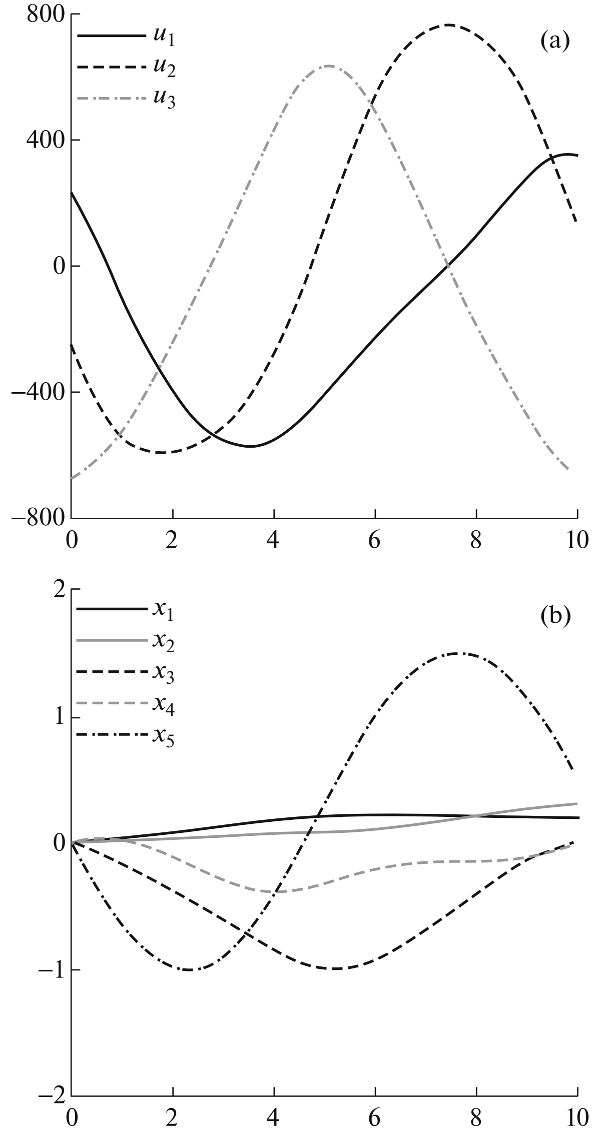

Fig. 3. Controls and trajectories found for a mobile robot.

where

$$M_1(x,u) = -c_1(x_1 - u_1), \quad M_2(x,u) = -c_2(x_3 - x_1 - u_2),$$

$$F_1(x) = -m_2 l_1 R_2 \sin(x_3 - x_1)x_2^2, \quad F_2(x) = m_2 l_2 R_2 \sin(x_3 - x_1)x_4^2,$$

$$a_{11} = m_1\rho_1^2 + m_2 l_1^2, \quad a_{12} = a_{21} = m_2 R_1 l_1 \cos(x_3 - x_1), \quad a_{22} = m_2\rho_2^2.$$

For the considered robot model, $m_1 = 7.62$, $m_2 = 8.73$, $R_1 = 0.239$, $R_2 = 0.251$, $\rho_1 = 0.968$, $\rho_2 = 0.973$, $l_1 = 0.5$, $l_2 = 0.67$, and $c_1 = c_2 = 10$. The trajectory of motion is assumed to satisfy the constraints $|M_i(x,u)| \le 10$ for $i = 1,2$ and $\pi/6 \le x_1(t) \le 5/6\pi$ and $\pi/3 \le x_1(t) - x_3(t) \le 5/6\pi$ for $t \in [0,T]$.

The task is to find a control driving the system from the point $x'(0) = (\pi/6, 0, -\pi/6, 0)$ to the point $x'(T) = (5/6\pi, 0, \pi/3, 0)$ over a minimum time T.

Starting at the approximation $T = 4$, $u_1(t) = 0$, $u_2(t) = 0$, $t \in [0,T]$, a solution was obtained for which the constraint residuals did not exceed 10^{-3} and the functional value was 2.88. The optimal control and the corresponding state coordinates are shown in Fig. 4.

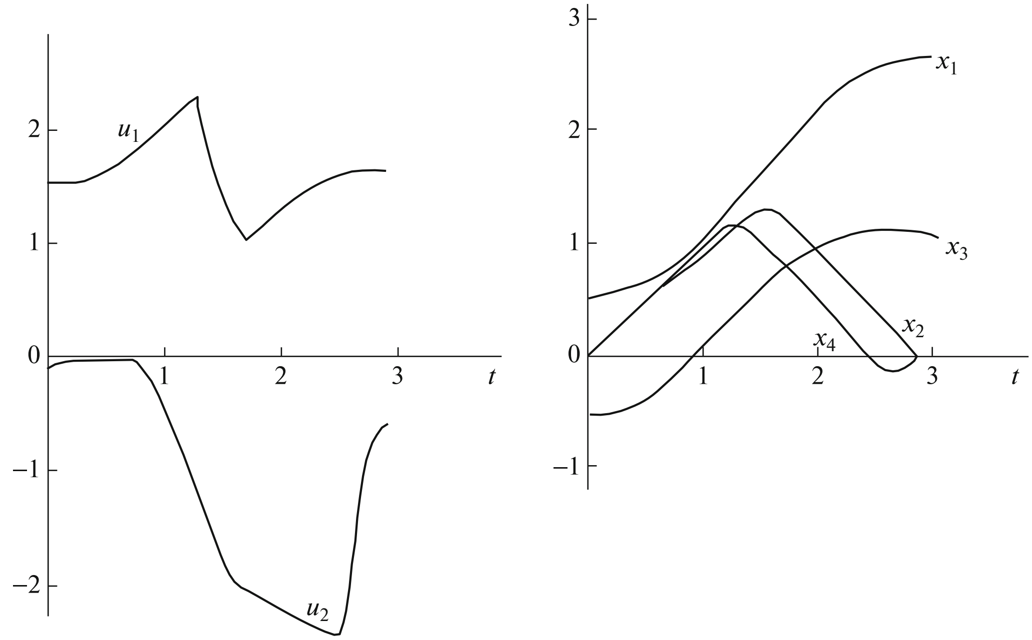

Fig. 4. Plots of the controls and state coordinates for the problem described in Subsection 4.2.

4.3. Optimization of an Electric Power System (*EPS*)

A mathematical EPS model was developed at the Melentiev Energy Systems Institute of the Siberian Branch of the Russian Academy of Science. It consists of a family of subsystems describing the generation and consumption of electricity, which are combined in a unified system by electrical network equations. For many years, this model has been successfully used to compute various operation modes of designed EPSes. Consider a small EPS model consisting of m synchronous generators and m steam turbines. Each synchronous machine is described by the Park–Gorev differential equations (without allowance for transient processes in the stator armature), which, after reducing them to the standard form, become

$$\dot{x}_i = x_{m+i}, \quad i = \overline{1, m},$$

$$\dot{x}_{m+i} = \omega_0/(T_{ji}P_{n_i})[x_{5m+i} - U_i^1/X_{d_i}^1(x_{2m+i}\sin x_i + x_{3m+i}\cos x_i) + U_i^2/X_{d_i}^1(x_{2m+i}\cos x_i - x_{3m+i}\sin x_i) - D_i x_{m+i}], \quad i = \overline{1, m},$$

$$\dot{x}_{2m+i} = 1/T_{d_0 i}[x_{4m+i} - x_{2m+i} + (X_{d_i} - X_{d_i}^1)/X_{d_i}^1(U_i^2 \sin x_i - x_{2m+i} + U_i^1 \cos x_i)], \quad i = \overline{1, m},$$

$$\dot{x}_{4m+i} = 1/TR_i[E_{R_0 i} - x_{4m+i} + K_{Ui}(U_{0i} - \sqrt{(U_i^1)^2 + (U_i^2)^2} + K_{I_i}/X_{d_i}^1(-I_{0i}X_{d_i}^1 + ((x_{2m+i})^2 + (x_{3m+i})^2 + (U_i^1)^2 + (U_i^2)^2 - 2\cos x_i(U_i^2 x_{3m+i} + U_i^1 x_{2m+i}) + 2\sin x_i(U_i^1 x_{3m+i} - U_i^2 x_{2m+i}))^{1/2}) + K_{f_i}/2\pi x_{m+i} + K_{f_i}^1/2\pi x_{2m+i}], \quad i = \overline{1, m}.$$

The dynamics of the steam turbines are governed by the equations

$$\dot{x}_{5m+i} = 1/T_{si}[-P_{ni}/\omega_0\sigma_{1i}x_{m+i} + x_{6m+i} - x_{5m+i}], \quad i = \overline{1, m},$$

$$\dot{x}_{6m+i} = 1/T_{p2i}[-P_{ni}/\omega_0\sigma_{2i}x_{m+i} + u_i], \quad i = \overline{1, m}.$$

The state variables x_{jm+i}, $j = \overline{0, 6}$, are the generator rotor angle, slip, the transition emf components in the longitudinal and transverse directions, and the field coil voltage for each $i = \overline{1, m}$. The control U_i changes the setting of the velocity regulator to ensure a stable dynamic transition to the prescribed post-emergency state after emergency load shedding. The right-hand sides of the equations also involve technical parameters of the generators and turbines, whose interpretations are omitted. The number of generators and turbines is set equal to five. Thus, for $m = 5$, the number of differential equations is 35, i.e., $n = 35$.

The electrical network model consists of algebraic equations for node voltages, which also involve the state variables. These equations are usually set in complex variables, but, for the numerical solution, they are rewritten in real variables. For example, in the experiment, the number of equations in complex variables was specified as $N = 14$, while the transition to real variables produced a system of 28 algebraic equations:

$$C_i^1U_i^1 - C_i^2U_i^2 + \sum_{k=1,k\neq i}^{N}(-U_k^1Y_{ik}^1 + U_k^2Y_{ik}^2) = 1/X_{d_i}^1(x_{2m+i}\sin x_i + x_{3m+i}\cos x_i), \quad i = \overline{1,N},$$

$$C_i^1U_i^2 + C_i^2U_i^1 + \sum_{k=1,k\neq i}^{N}(-U_k^1Y_{ik}^2 - U_k^2Y_{ik}^1) = 1/X_{d_i}^1(x_{2m+i}\cos x_i + x_{3m+i}\sin x_4), \quad i = \overline{1,N},$$

where $C_i^1 = P_{ni}/(U_i)^2 + A_i^1$, $C_i^2 = A_i^2 - Q_{ni}/(U_i)^2$, and $(U_i)^2 = (U_i^1)^2 + (U_i^2)^2$.

Additionally, the state and control constraints were specified as

$$|x_i(t) - x_j(t)| \leq \delta_{\max}, \quad i, j = \overline{1,m},$$

$$x_{\min_i} \leq x_{4m+i}(t) \leq x_{\max_i}, \quad i = \overline{1,2m},$$

$$U_{\min_i} \leq U_i(t) \leq U_{\max_i}, \quad i = \overline{1,m}.$$

The objective functional was a terminal state function for the system (at $t = 10$ s) that measures the deviations of some of the state coordinates from prescribed values (e.g., power ratings).

The problem was solved numerically by applying the projected Lagrangian method. After performing 11 outer iterations, each involving about 20 inner iterations of the reduced gradient method, the given equalities were satisfied up to 10^{-6} and all variables remained within the prescribed ranges. The resulting optimal control ensured that the EPS reached the required operation mode in 10 s after the emergency load shedding.

Conclusions. The multimethod computational technique implemented in the form of parallel iterative optimization processes with the choice of the best approximation solves problems with automatic application of different optimization methods, thus significantly enhancing the efficiency and reliability of numerical solutions obtained in optimal control applications. Obtaining numerical solutions at the lowest computational cost is important in the design of computer-controlled robotic and electric power systems.

CONCLUSIONS

Practice has shown that the sequence of approximations produced by a multimethod algorithm for solving complicated control problems is based, as a rule, on several (three to five) numerical methods chosen automatically according to a given criterion in the optimization process. The conducted numerical experiments confirmed the efficiency of this approach as applied to real-world optimal control problems. It was established that the multimethod approach is often the only technique available for obtaining numerical solutions of complicated optimal control problems, since each of the methods taken separately cease to converge before obtaining an optimal solution. Multimethod algorithms can be efficiently implemented in practice relying on modern information technologies and multiprocessor computers. Based on this approach, the software code implementing the multimethod technique for computing optimal control and optimal parameters (see [9–12]) has been successfully used to solve complicated real-world optimal control problems from various fields of science and engineering (see [10–14]). The use of an efficient technique for control computation is especially important in real-time control systems, for example, in control systems for high-maneuverability aircraft. For example, this software was used to solve a series of optimal maneuvering problems in the design of the Su-57 jet fighter aircraft, which has the world's highest maneuverability (see [11]). It is also well known that the successful high-accurate landing of the Buran space shuttle was ensured by onboard software for choosing an optimal initial approximation. Specifically, the onboard computer chose a least cross-wind-dependent initial point for landing and calculated an optimal glide path.

REFERENCES

1. L. S. Pontryagin, V. G. Boltyanskii, R. V. Gamkrelidze, and E. F. Mishchenko, *The Mathematical Theory of Optimal Processes* (Nauka, Moscow, 1969; Gordon and Breach, New York, 1986).
2. R. E. Bellman and R. E. Kalaba, *Quasilinearization and Nonlinear Boundary-Value Problems* (Elsevier, New York, 1965).
3. R. Gabasov and F. M. Kirillova, *Qualitative Theory of Optimal Processes* (Nauka, Moscow, 1971) [in Russian].
4. O. V. Vasil'ev and A. I. Tyatyushkin, "A method for solving optimal control problems that is based on the maximum principle," USSR Comput. Math. Math. Phys. **21** (6), 14–22 (1981).
5. R. Gabasov, F. M. Kirillova, and A. I. Tyatyushkin, *Constructive Optimization Methods,* Part I: *Linear Problems* (Universitetskoe, Minsk, 1984) [in Russian].
6. P. E. Gill, W. Murray, and M. H. Wright, *Practical Optimization* (Academic, London, 1981).
7. Yu. G. Evtushenko, *Methods for Solving Extremal Problems and Their Application in Optimization Systems* (Nauka, Moscow, 1982) [in Russian].
8. R. P. Fedorenko, *Approximate Solution of Optimal Control Problems* (Nauka, Moscow, 1978) [in Russian].
9. A. I. Tyatyushkin, "Package KONUS as applied to the optimization of continuous controlled systems," in *Application Packages*: *Experience of Using* (Nauka, Moscow, 1989), pp. 63–83 [in Russian].
10. A. Yu. Gornov, A. I. Tyatyushkin, and E. A. Finkelstein, "Numerical methods for solving terminal optimal control problems," Comput. Math. Math. Phys. **56** (2), 221–234 (2016).
11. A. I. Tyatyushkin and B. E. Fedunov, "Possibilities of defense from a missile attacking from the back hemisphere by a vertical maneuver," J. Comput. Syst. Sci. Int. **45** (1), 120–127 (2006).
12. A. I. Tyatyushkin, *Multimethod Technique for Optimization of Controlled Systems* (Nauka, Novosibirsk, 2006) [in Russian].
13. A. I. Tyatyushkin, "Numerical optimization methods for controlled systems with parameters," Comput. Math. Math. Phys. **57** (10), 1592–1606 (2017).
14. A. I. Tyatyushkin, "Multimethod optimization of control in complicated applied problems," Comput. Math. Math. Phys. **59** (2), 224–235 (2019).

Translated by I. Ruzanova