Optimal Flood Control

Peter Dickinson[*] *Joost Hulshof*[†] *André Ran*[†]
Majid Salmani[‡] *Martijn Zaal*[†]

Abstract

A mathematical model for optimal control of the water levels in a chain of reservoirs is studied. Some remarks regarding sensitivity with respect to the time horizon, terminal cost and forecast of inflow are made.

1 Introduction

For the Study Week Mathematics and Industry a problem was proposed by Deltares concerning the computation of optimal controls for the settings of hydraulics. The long term goal that Deltares has is to provide methods for optimal control of the settings of the hydraulics in the Dutch river system as a whole. At present each of the hydraulics is treated separately and not in combination. A "toy problem" was provided by Deltares to use in the study week. See [32]. This simple problem consists of a chain of four reservoirs, with an inflow into the first reservoir which is predictable on a reasonable time-scale. The model will be explained in the first section of this paper.

At present there is a method to compute the optimal control in use at Deltares. The questions posed by Deltares were two-fold: is this a good method, and are there ways to improve on it? In addition, it would be very nice if there was an efficient way to adjust optimal control to changing forecasts of inflow. During the study week the team also discussed issues connected to the modeling of the system.

2 The model

In this section, the problem will be modeled and formulated as an optimal control problem.

The water levels in a chain of reservoirs can be represented by a vector $\mathbf{x} = (x^1, \ldots, x^m)$. A differential equation for $\mathbf{x}$ can be derived by studying the flow of water through the chain of reservoirs. The water flowing out of a certain reservoir can be partially controlled by a hydraulic structure, corresponding to a control $\mathbf{u} = (u^1, \ldots, u^m)$. If the water level rises above a certain critical level x^j_{cr}, extra

[*] Rijksuniversiteit Groningen
[†] VU University Amsterdam
[‡] University Bremen

water will flow out of the reservoir through a so-called spillway. Hence, the volume of water flowing out of a reservoir is given by

$$u^j - g^j(x - h_{\mathrm{cr}})$$

where g^j is a function satisfying $g(y) = 0$ for $y < 0$. The water flowing into a reservoir is simply the flow out of the previous reservoir. The flow of water into the first reservoir is denoted by p, which is, for the time being, assumed to be given. Taking the surface area of the reservoirs, which is assumed to be independent of the water level, into account, this results in the following differential equation for $\mathbf{x}$.

$$\begin{cases} \dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}, p) \\ \mathbf{x}(0) = \mathbf{x}_0, \end{cases} \tag{1}$$

where $\mathbf{x}_0$ is a given initial condition, and $\mathbf{f}$ is given by

$$f^j(\mathbf{x}, \mathbf{u}, p) = \begin{cases} \frac{1}{A^1}\left(p - u^1 - g^1\left(x^1 - h^1_{\mathrm{cr}}\right)\right), & \text{if } j = 1, \\ \frac{1}{A^j}\left(u^{j-1} + g^{j-1}\left(x^{j-1} - h^{j-1}_{\mathrm{cr}}\right) - u^j - g^j\left(x^j - h^j_{\mathrm{cr}}\right)\right), & \text{otherwise.} \end{cases} \tag{2}$$

Here, the numbers A^j indicate the surface area of the reservoirs, which are assumed to be independent of the water levels for simplicity.

There are certain restrictions to the controlled outflow. Each hydraulic structure has a maximum flow of $u^j_{\max}$, and the setting of the structures can only be changed at certain predetermined moments. More precisely, $\mathbf{u}(t) = \mathbf{u}_i$ for $t \in (\frac{i-1}{n}, \frac{i}{n}]$.

The objective is to keep the water levels as close as possible to a given set point level $h^j_{\mathrm{sp}} < h^j_{\mathrm{cr}}$, minimizing the spillover, while adjusting the hydraulic structures a little as possible. More precisely, the objective function

$$J(\mathbf{u}, \mathbf{x}) = \phi(\mathbf{x}(T)) + \int_0^T L(\mathbf{x}(t), \mathbf{u}(t))\,\mathrm{d}t + \frac{1}{2}\sum_{i=1}^{n} \|\mathbf{u}_i - \mathbf{u}_{i-1}\|^2 \tag{3}$$

is minimized, where $\mathbf{u}_0$ is the (given) initial setting of the hydraulic structures, and L represents the penalty for deviation from the set point level and use of the spillway. The function ϕ indicates terminal costs and is not specified here.

In this particular model, specific choices for L and g will be used:

$$L(\mathbf{x}, \mathbf{u}) := \sum_{j=1}^{m} w^j_{\mathrm{sp}}(x - x_{\mathrm{sp}})^2 + w^j_{\mathrm{cr}}\left((x - x_{\mathrm{cr}})^+\right)^2, \tag{4}$$

$$g^j(y) := C^j\left((x - x_{\mathrm{cr}})^+\right)^{\frac{3}{2}}. \tag{5}$$

Putting all equations together, the optimal control problem is

$$\begin{cases} \text{minimize} & J(\mathbf{u}, \mathbf{x}) \\ \text{under} & \dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}, p) \\ & \mathbf{x}(0) = \mathbf{x}_0 \\ & 0 \le u^j_i \le u^j_{\max} \quad (j = 1, \ldots m) \end{cases} \tag{6}$$

The problem as outlined above may be found in [32], where also a case study was undertaken for the case of a chain of four reservoirs.

3 Lagrange multipliers

3.1 Optimality conditions

Ignoring the range constraint on $\mathbf{u}(t)$ for the time being, optimality conditions for (6) can be derived by studying the critical points of the extended Lagrange functional

$$\mathcal{L}(\mathbf{x}, \mathbf{u}, \boldsymbol{\lambda}, p) = \phi(\mathbf{x}(T)) + \int_0^T L(\mathbf{x}, \mathbf{u}) + \boldsymbol{\lambda}(\mathbf{f} - \dot{\mathbf{x}})\,\mathrm{d}t + \frac{1}{2}\sum_{i=1}^{n} \|\mathbf{u}_i - \mathbf{u}_{i-1}\|^2, \tag{7}$$

where $\boldsymbol{\lambda}$ is interpreted as a row vector. For notational convenience, introduce $H := L + \boldsymbol{\lambda}\mathbf{f}$.

Choosing a smooth variation $\boldsymbol{\xi}$ for $\mathbf{x}$,

$$\begin{aligned} \left\langle \frac{\partial \mathcal{L}}{\partial \mathbf{x}}, \boldsymbol{\xi} \right\rangle &= \frac{d\phi}{d\mathbf{x}}(\mathbf{x}(T))\boldsymbol{\xi}(T) + \int_0^T \frac{\partial H}{\partial \mathbf{x}}\boldsymbol{\xi} - \boldsymbol{\lambda}\dot{\boldsymbol{\xi}}\,\mathrm{d}t \\ &= \left(\frac{d\phi}{d\mathbf{x}}(\mathbf{x}(T)) - \boldsymbol{\lambda}\right)\boldsymbol{\xi}(T) + \int_0^T \left(\frac{\partial H}{\partial \mathbf{x}} + \dot{\boldsymbol{\lambda}}\right)\boldsymbol{\xi}\,\mathrm{d}t + \boldsymbol{\lambda}(0)\boldsymbol{\xi}(0), \end{aligned} \tag{8}$$

where derivatives with respect to vectors are interpreted as row vectors. Note that the last term vanishes because there is an initial condition for $\mathbf{x}$. Hence, requiring that the variation of $\mathcal{L}$ with respect to $\mathbf{x}$ is zero results in a terminal value problem for $\boldsymbol{\lambda}$:

$$\begin{cases} \dot{\boldsymbol{\lambda}} = -\dfrac{\partial H}{\partial \mathbf{x}} \\ \boldsymbol{\lambda}(T) = \dfrac{d\phi}{d\mathbf{x}}(\mathbf{x}(T)) \end{cases} \tag{9}$$

Since $\mathbf{u}$ is required to be piecewise constant, the variation with respect to $\mathbf{u}$ is actually a partial derivative:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{u}_i} = \int_{\frac{i-1}{n}}^{\frac{i}{n}} \frac{\partial H}{\partial \mathbf{u}_i}\,\mathrm{d}t - \mathbf{u}_{i+1} + 2\mathbf{u}_i - \mathbf{u}_{i-1}, \tag{10}$$

if $i < n$. In case $i = n$, the last three terms become $\mathbf{u}_i - \mathbf{u}_{i-1}$. Equivalently, one could introduce an artificial variable $\mathbf{u}_{n+1}$ and require that $\mathbf{u}_{n+1} = \mathbf{u}_n$.

As expected, since $\frac{\partial H}{\partial \boldsymbol{\lambda}} = \mathbf{f}$, setting the variation of $\mathcal{L}$ with respect to $\boldsymbol{\lambda}$ equal to zero gives back (1).

All variations together result in the optimality conditions

$$\begin{cases} \dot{\mathbf{x}} = \mathbf{f} \\ \dot{\boldsymbol{\lambda}} = -\dfrac{\partial H}{\partial \mathbf{x}} \\ \mathbf{u}_{i+1} - 2\mathbf{u}_i + \mathbf{u}_{i-1} = \displaystyle\int_{\frac{i-1}{n}}^{\frac{i}{n}} \frac{\partial H}{\partial \mathbf{u}_i}\,\mathrm{d}t \\ \mathbf{x}(0) = \mathbf{x}_0 \\ \boldsymbol{\lambda}(T) = \dfrac{d\phi}{d\mathbf{x}}(\mathbf{x}(T)) \\ \mathbf{u}_{n+1} - \mathbf{u}_n = \mathbf{0} \end{cases} \tag{11}$$

Note that this is almost a coupled system of ordinary differential equations: only the equation for u is discretized. The artificial condition $\mathbf{u}_{n+1}-\mathbf{u}_n = \mathbf{0}$ can be considered to be the discretized analog of the terminal condition $\dot{\mathbf{u}}(T) = \mathbf{0}$.

3.2 Deltares's approach

A very intuitive strategy for solving (6) is to first solve the initial value problem for $\mathbf{x}$ in terms of the control $\mathbf{u}$, substituting the solution into the criterion J, and minimizing the resulting function of $\mathbf{u}$. More precisely, the dependence of $\mathbf{x}$ on $\mathbf{u}$ can be written and used to study the optimum, at least formally:

$$\Phi(\mathbf{u}) := \phi(\mathbf{x}(T;\mathbf{u})) + \int_0^T L(\mathbf{x}(t;\mathbf{u}),\mathbf{u}(t))\,\mathrm{d}t + \frac{1}{2}\sum_{i=1}^{n} \|\mathbf{u}_i - \mathbf{u}_{i-1}\|^2 \tag{12}$$

where $\mathbf{x}(.;\mathbf{u})$ is the solution of the initial value problem (1).

The partial derivatives of Φ with respect to the $\mathbf{u}_i$'s now involve an extra term representing the $\mathbf{u}$-dependence.

$$\begin{aligned}\frac{\partial\Phi}{\partial\mathbf{u}_i} = \frac{\mathrm{d}\phi}{\mathrm{d}\mathbf{x}}(\mathbf{x}(T;\mathbf{u}))\frac{\partial\mathbf{x}(T;\mathbf{u})}{\partial\mathbf{u}_i} + \int_0^T \frac{\partial L}{\partial\mathbf{x}}\frac{\partial\mathbf{x}(t;\mathbf{u})}{\partial\mathbf{u}_i}\,\mathrm{d}t \\ + \int_{\frac{i-1}{n}}^{\frac{i}{n}} \frac{\partial L}{\partial\mathbf{u}_i}\,\mathrm{d}t - \mathbf{u}_{i+1} + 2\mathbf{u}_i - \mathbf{u}_{i-1},\end{aligned} \tag{13}$$

Assuming some smoothness, $\frac{\partial\mathbf{x}(t;\mathbf{u})}{\partial\mathbf{u}_i}$ satisfies the following initial value problem

$$\begin{cases} \dfrac{\partial}{\partial t}\dfrac{\partial\mathbf{x}(t;\mathbf{u})}{\partial\mathbf{u}_i} = \dfrac{\partial\mathbf{f}}{\partial\mathbf{x}}\dfrac{\partial\mathbf{x}(t;\mathbf{u})}{\partial\mathbf{u}_i} + \dfrac{\partial\mathbf{f}}{\partial\mathbf{u}}\chi_{(\frac{i-1}{n},\frac{i}{n}]}(t), \\ \dfrac{\partial\mathbf{x}(0;\mathbf{u})}{\partial\mathbf{u}_i} = \mathbf{0}. \end{cases} \tag{14}$$

Introducing $\boldsymbol{\lambda}$ as the solution of the terminal value problem (9), the first integral in (13) can be simplified:

$$\begin{aligned}\int_0^T \frac{\partial L}{\partial\mathbf{x}}(\mathbf{x}(t;\mathbf{u}),\mathbf{u}(t))\frac{\partial\mathbf{x}(t;\mathbf{u})}{\partial\mathbf{u}_i}\,\mathrm{d}t &= \int_0^T \left(\frac{\partial H}{\partial\mathbf{x}} - \boldsymbol{\lambda}\frac{\partial\mathbf{f}}{\partial\mathbf{x}}\right)\frac{\partial\mathbf{x}(t;\mathbf{u})}{\partial\mathbf{u}_i}\,\mathrm{d}t \\ &= -\int_0^T \left(\dot{\boldsymbol{\lambda}} + \boldsymbol{\lambda}\frac{\partial\mathbf{f}}{\partial\mathbf{x}}\right)\frac{\partial\mathbf{x}(t;\mathbf{u})}{\partial\mathbf{u}_i}\,\mathrm{d}t \\ &= -\int_0^T \left(\dot{\boldsymbol{\lambda}} + \boldsymbol{\lambda}\frac{\partial\mathbf{f}}{\partial\mathbf{x}}\right)\frac{\partial\mathbf{x}(t;\mathbf{u})}{\partial\mathbf{u}_i}\,\mathrm{d}t\end{aligned} \tag{15}$$

Integrating by parts, the term involving $\frac{\partial\mathbf{f}}{\partial\mathbf{x}}$ drops out, leaving only a term involving $\frac{\partial\mathbf{f}}{\partial\mathbf{u}}$ and initial and terminal values:

$$\int_0^T \frac{\partial L}{\partial\mathbf{x}}(\mathbf{x}(t;\mathbf{u}),\mathbf{u}(t))\frac{\partial\mathbf{x}(t;\mathbf{u})}{\partial\mathbf{u}_i}\,\mathrm{d}t = -\frac{d\phi}{d\mathbf{x}}(\mathbf{x}(T;\mathbf{u}))\frac{\partial\mathbf{x}(T;u)}{\partial\mathbf{u}_i} + \int_{\frac{i-1}{n}}^{\frac{i}{n}} \boldsymbol{\lambda}\frac{\partial\mathbf{f}}{\partial\mathbf{u}}\,\mathrm{d}t \tag{16}$$

Substituting this back into (13), it follows that

$$\frac{\partial \Phi}{\partial \mathbf{u}_i} = \int_{\frac{i-1}{n}}^{\frac{i}{n}} \frac{\partial H}{\partial \mathbf{u}_i} \,\mathrm{d}t - \mathbf{u}_{i+1} + 2\mathbf{u}_i - \mathbf{u}_{i-1} = \frac{\partial \mathcal{L}}{\partial \mathbf{u}_i}(\mathbf{x}(,;\mathbf{u}), \mathbf{u}, \boldsymbol{\lambda}(.;\mathbf{u}), p) \tag{17}$$

where $\mathbf{x}$ and λ solve (1) and (9), respectively. Apparently, the derivative of Φ can be computed using the Lagrange multipliers.

This can be exploited when numerically solving the optimal control problem: given an initial guess for $\mathbf{u}$, one can find the derivative of Φ, which can, of course, be used to improve the guess, by computing the Lagrange multipliers. The efficiency of this method obviously also depends on the quality of the initial guess for $\mathbf{u}$. Another appealing property of this approach is that it is easy to add the bounds on $\mathbf{u}$. Indeed, at each step of the algorithm one checks to see whether the newly computed $\mathbf{u}$ satisfies the bounds. If not, $\mathbf{u}$ is at either its maximum or its minimum, and one follows the boundary of the feasible set until the gradient is pointing into the feasible set again.

The method explained above is to some extent classical in optimal control theory, see, e.g., [3, 9, 16].

3.3 Summary of the algorithm

Starting from an initial guess of $\mathbf{u}$ one solves the system of equations forward in time for $\mathbf{x}$ and determines $\mathbf{x}(T)$ this way. Then, one solves for $\lambda(t)$ backwards in time. Using these, one finds $\frac{\partial \Phi}{\partial \mathbf{u}_i}$, and adjust $\mathbf{u}$ using this. Then one repeats the cycle, until convergence is reached.

Obviously, the selection of an initial guess of $\mathbf{u}$ is important in case the algorithm converges slowly, or takes a lot of time to execute. One may use the fact that for the case of no disturbance the system is linear and the cost function quadratic, in which case direct application of standard optimal LQ control (see e.g. [9]) gives us an initial guess. However, as it turns out, this initial guess may not be feasible in the sense that it violates $0 \le u_i^j \le u_{max}^j$. In the application to the toy problem this was observed in practice. Just taking the optimal LQ-control input and multiplying it with the characteristic function of $[0, u_{max}]$ gives an initial guess that may be better than just taking the zero input as initial guess.

4 Some remarks

Implementing the method above for the extremely simple case of one reservoir was carried out as a bachelor thesis project by one of the students (Hidde Kok) at VU University. The case considered was that of one reservoir with area $A = 60000$, with critical level $h_{\mathrm{cr}} = 0.2$, and set-point water level at $h_{\mathrm{sp}} = 0$. The weights in the cost function were set to $w_{\mathrm{sp}} = 1$ and $w_{\mathrm{cr}} = 10$. The final state was penalized with $2(x(T) - h_{\mathrm{sp}}) + 2\max(0, x(T) - h_{\mathrm{cr}})$. The perturbation p was set at $p = 50$ for the duration of the period of 20 to 50 minutes from the start. The program did not pose any restriction on the values of the input. We compared with the maximum level of u set to $u_{\max} = 20$, and with the minimum level $u_{\min} = 0$.

The student's program was run for two cases: one with time horizion two hours, one with time horizon ten hours. The two graphs below show the results.

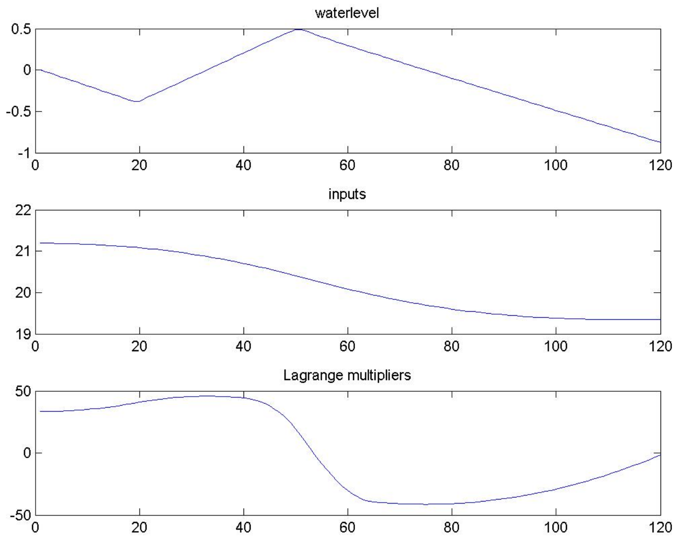

Figure 1: Two hour time horizon, no bounds on the input.

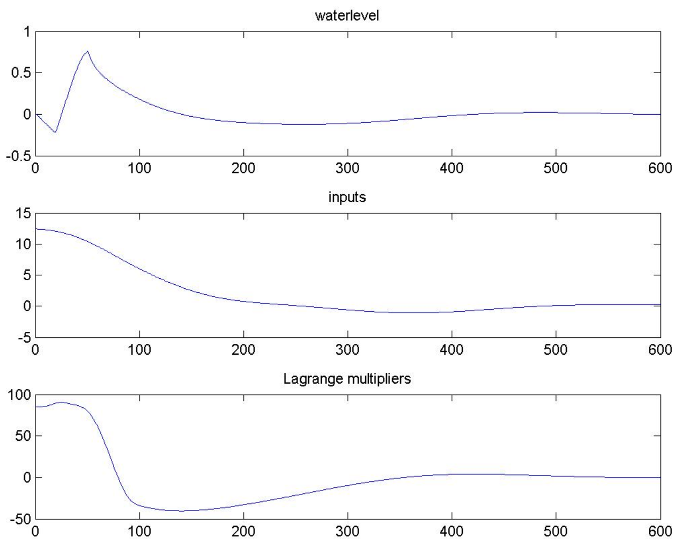

Figure 2: Ten hour time horizon, no bounds on the input.

It is obvious from figure 1 and 2 that the time horizon has a serious effect on the optimal input. This is caused by the fact that the influence of the cost on the final state is spread over a larger period. Also note that the two hour time horizon is not enough to restore the water level to the set point, and that for the two hour time horizon the inputs are above the maximum level, whereas for the ten hour time horizon this never occurs. However, for the ten hour time horizon, the input is (slightly) below zero for a considerable amount of the time. In addition, in the ten hour case we see that the peak of the water level is higher. This is caused by the fact that the cost puts more weight (due to the larger time period) on the deviation from the set point level.

In figures 3 and 4 the same settings are considered, however, now with bounds on the input taken into account. Observe that again we see that there is a substantial difference between time horizon two hours and time horizon ten hours. Comparing to the case without bounds on the input we see that even with a large time horizon the set point level is not reached. This is due to the fact that both the minimal value of the control and the reference level of the forecasted inflow are zero. In practice, however, this will not be the case: the forecasted inflow will always be positive, whereas the minimal value of the control remains zero. This difference might have been taken into account in the model by taking a negative minimal value for the control, or, alternatively, taking a positive reference level for the forecasted inflow. It can be seen from figure 2 that a small negative control is needed to return to the set point level if the water level drops below h_{sp} after the disturbance in the inflow has passed. We did not take this into account in our example in order to stay close to the model in [32].

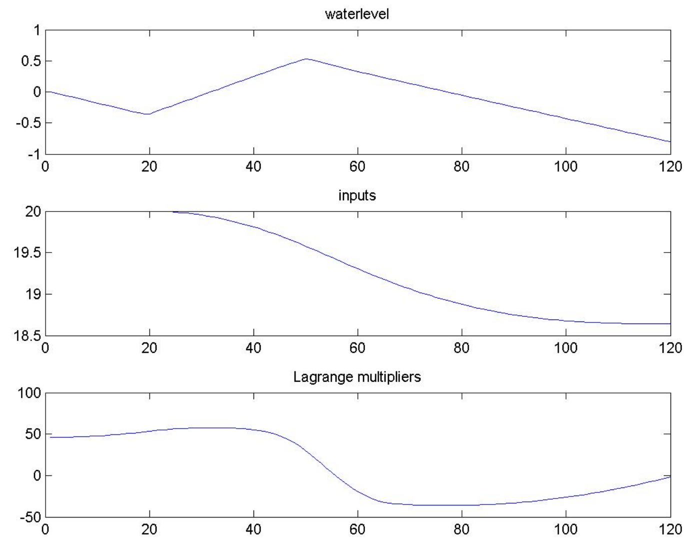

Figure 3: Two hour time horizon taking into account the bounds on the input.

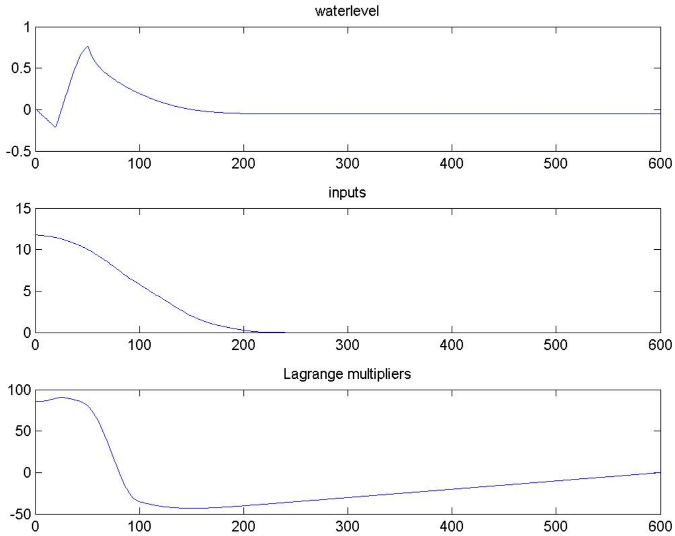

Figure 4: Ten hour time horizon taking into account the bounds on the input.

5 NLP formulation

In order to solve (6), there are some numerical methods using nonlinear programming techniques, see [10, 11, 9, 7]. These methods use a suitable discretization of the control problem by which it is transcribed into a parametric NLP problem [4, 8, 18]. The discretization methods can be divided in two general categories: Full-discretization and Partial-discretization. In the first method, all state and control variables are discretized, but in the second method, only control variables are discretized. Indeed, the states will be determined by integration. The nature of the problem suggests that partial discretization should be used, as demonstrated in section 2. The state approximations $x_i \in \mathbb{R}^n$ of the values $x(t_i)$ can be achieved recursively as functions of the control variables by an integration scheme such as forward Euler or Runge-Kutta approximation.

The problem (6) defines an NLP of the general form

$$\begin{cases} \text{minimize} & F(z,p) \\ \quad\text{under} & G_i(z,p) = 0 \quad (i = 1, \dots N_e) \\ & G_j(z,p) \leq 0 \quad (j = 1, \dots, N_i) \end{cases} \tag{18}$$

where $F : \mathbb{R}^{N_z} \times P \to \mathbb{R}$, $G_i, G_j : \mathbb{R}^{N_z} \times P \to \mathbb{R}$, N_e and N_i are the numbers of equality and inequality constraints, respectively. Note that the number of decision variables in this NLP(p) equals $N_z = mn$. We shall study the differential properties of optimal solutions to the perturbed problem NLP(p) and the related optimal values of the objective function with respect to p in a neighborhood of nominal parameter p_0.

5.1 Parameter Sensitivity Analysis of NLP(p)

Since perturbations are unavoidable, only having knowledge about the nominal optimal solution is not enough and analysis of the effects of perturbations is necessary. So far we have transformed the optimal control problem with parameter p into the NLP(p) problem (18). After solving (6), independent of the discretization technique, we know the set and the number of active constraints N_a. Assume that $G^a(z^*, p_0)$ denotes the collection of the active constraints at point (z^*, p_0). Similar to the sections above, the Lagrangian function is defined by

$$\mathcal{L}(z, \lambda, p) = F(z, p) + \lambda^T G^a(z, p). \tag{19}$$

The sufficient conditions for the differentiability of the optimal solution (z^*, p_0) with respect to $p \in P \subseteq \mathbb{R}^{N_p}$ are given in the next theorem, [8, 18].

Theorem 5.1. *Suppose $F(z, p)$ and $G^a(z, p)$ are twice differentiable with respect to z and p. And assume z^* be a strong regular local solution of (18) for a fixed parameter p_0 with Lagrange multipliers λ_0, i.e. $G^a(z_0, p_0) = 0$ and*

1. *$\nabla_z G^a(z_0, p_0)$ is full rank (z_0 is regular),*
2. *$\nabla_z \mathcal{L}(z_0, \lambda_0, p_0) = 0$, $\lambda_0^T G^a(z_0, p_0) = 0$ (necessary optimality conditions),*
3. *$\lambda_{i_a} > 0$ for $i_a = 1, \ldots, N_a$ (strict complementarity),*
4. *$v^T \nabla^2_{zz} \mathcal{L}(z_0, \lambda_0, p_0) v > 0$, $\forall v \in \ker(\nabla_z G^a(z_0, p_0))$, $v \neq 0$ (second order sufficient conditions),*

then there exists a neighborhood $\mathcal{P}$ of p_0 such that the problem (18) has a unique strong regular local solution $z(p)$ and $\lambda(p)$. Furthermore, $z(p)$ and $\lambda(p)$ are continuously differentiable functions of $p \in \mathcal{P}(p_0)$ and

$$\begin{pmatrix} \nabla^2_{zz} \mathcal{L}(z_0, \lambda_0, p_0) & \nabla_z G^a(z_0, p_0)^T \\ \nabla_z G^a(z_0, p_0) & 0 \end{pmatrix} \begin{pmatrix} \frac{dz}{dp}(p_0) \\ \frac{d\lambda}{dp}(p_0) \end{pmatrix} = - \begin{pmatrix} \nabla^2_{zp} \mathcal{L}(z_0, \lambda_0, p_0) \\ \nabla^2_p G^a(z_0, p_0) \end{pmatrix} \tag{20}$$

where $\nabla^2_{zz} L$ denotes the Hessian of the Lagrangian.

The proof of the theorem is given in [11] and [17]. Since the coefficient matrix in (20) is non-singular by the assumption of Theorem 5.1, $\frac{dz}{dp}(p_0)$ and $\frac{d\lambda}{dp}(p_0)$ can be calculated explicitly by solving the linear equation system (20). In [11], it is explained how one can check the assumptions of Theorem 5.1 numerically by using the projected or reduced Hessian.

5.2 Real-Time Mission Correction

In this section, a mathematical method to correct the violations of the space mission trajectory is presented. This method is based on the discussion from last section about sensitivity analysis of an NLP(p), and linear approximation

$$\begin{aligned} z(p) &= z(p_0 + \Delta p) \\ &\approx \tilde{z}(p) = z(p_0) + \frac{dz}{dp}(p_0) \Delta p, \end{aligned} \tag{21}$$

which uses the explicit sensitivity differentials achieved by solving linear system (20) for perturbations Δp to modify the control signals.

To deal with the linear approximation (21), one has to consider the changes of the active constraints, see [10, 11, 6]. Although (21) results in acceptable real-time approximations for small Δp, it can cause larger deviations from the active constraints for larger Δp and leads to a non-admissible solution as

$$\varepsilon_1 := G^a(\tilde{z}(p), p) \neq 0. \tag{22}$$

Introducing an auxiliary parameter $q \in \mathbb{R}^{N_a}$ for every active constraint in (18), one deals with the following problem

$$\begin{cases} \text{minimize} & F(z,p) \\ \quad \text{under} & G^a(z,p) - q = 0 \end{cases} \tag{23}$$

Choosing the nominal value of q which is $q_0 = 0$, the problem (23) is equivalent to the problem (18). Actually, the parameters can be considered as $(p, q) \in \mathbb{R}^{N_p+N_a}$. Since one of the problems (18) or (23) satisfies the conditions of Theorem 5.1 if the other one does, therefore one can compute the sensitivity differentials $\frac{dz}{dq}(q_0)$ and $\frac{d\lambda}{dq}(q_0)$ in the same way as (20). By using the new sensitivity differentials, we can hope that a better approximation of the form of (21) can be found to improve the optimality and admissibility of the real-time approximation. Considering (21) and (22), this approximation is given by

$$\begin{aligned} z(p) \approx \tilde{z}_2(p) &= \tilde{z}(p) - \frac{dz}{dq}(q_0)\varepsilon_1 \\ &= \tilde{z}(p) + \frac{dz}{dq}(0)G^a(\tilde{z}(p), p). \end{aligned} \tag{24}$$

Let $\tilde{z}_1(p)$ denote the same $\tilde{z}(p)$, then the improving steps (22) and (24) can be considered as an iterative process to construct sequences $(\varepsilon_k)_k$ and $(\tilde{z}_k)_k$ for $k = 1, 2, \ldots$ as the parameter and solution sequences, respectively. Since the nominal solution $z(p_0)$ as well as the sensitivity differentials $\frac{dz}{dp}(p_0)$ and $\frac{dz}{dq}(q_0)$ can be computed off-line, steps like (24) do not need any derivative computational cost. Moreover, the terms of form $\frac{dz}{dq}(0)G^a(\tilde{z}_i(p), p)$, can be considered as a correcting *feedback step* for ε_i-error correction. In the following, the feedback closed loop is briefly presented. The loop continues until a prescribed accuracy ε_∞ is achieved.

1. Initialize $\tilde{z}_1(p) = z(p_0) + \frac{dz}{dp}(p_0)\Delta p$, $k = 1$ and choose the desirable accuracy ε_∞.

2. While $\|G^a(\tilde{z}_k(p), p)\|_2 > \varepsilon_\infty$ do the following

   - $\tilde{z}_{k+1}(p) := \tilde{z}_k(p) - \frac{dz}{dq}(0)G^a(\tilde{z}(p), p)$,
   - $k := k + 1$.

For more details about the feedback rule and the convergence rate of $\tilde{z}_k(p)$, see [12].

References

[1] C.J. Alpert, T.C. Hu, J.H. Huang, A.B. Kahng, and D. Karger. Prim-Dijkstra tradeoffs for improved performance-driven routing tree design. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 14(7):890–896, 1995.

[2] S. Arora. Polynomial time approximation schemes for Euclidean TSP and other geometric problems. In *Proceedings of the 37th IEEE Symposium on Foundations of Computer Science*, pages 2–11, 1996.

[3] M. Athans and P. L. Falb. *Optimal Control*. McGraw-Hill, New York, 1966.

[4] A. Barclay, P. E. Gill, and Rosen J. B. SQP methods and their application to numerical optimal control. In W.H. Schmidt, K. Heier, L. Bittner, and R. Bulirsch, editors, *Variational Calculus, Optimal Control and Applications*, Basel, 1998. Birkhäuser.

[5] Cynthia Barnhart, Ellis L. Johnson, George L. Nemhauser, Martin W. P. Savelsbergh, and Pamela H. Vance. Branch-and-price: Column generation for solving huge integer programs. *Operations Research*, 46:316–329, 1996.

[6] T. J. Beltracchi and G. A. Gabriele. Observations on extrapolations using parameter sensitivity derivatives. In *Advances in Design Automation*, volume 14, pages 165–174. ASME, 1988.

[7] J. T. Betts. *Practical Methods for Optimal Control and Estimation Using Nonlinear Programming*. SIAM, Philadelphia, PA, second edition, 2010.

[8] H.G. Bock and K.J. Plitt. A multiple shooting algorithm for direct solution of optimal control problems. In *IFAC 9th World Congress*, Budapest, Hungary, 1984.

[9] A. Bryson and Y. Ho. *Applied Optimal Control*. Hemisphere/Wiley, 1975.

[10] C. Büskens. *Direkt Optimierungsmethoden zur numerischen Berechung optimaler Steuerungen*. Diploma thesis, Institut für Numerische Mathematik, Universität Münster, Münster, Germany, 1993.

[11] C. Büskens. *Optimierungsmethoden und Sensitivitätsanalyse für optimale Steuerungsprozesse mit Steuer- und Zustands-Beschränkungen*. Dissertation, Institut für Numerische Mathematik, Universität Münster, Münster, Germany, 1998.

[12] C. Büskens. *Echtzeitoptimierung und Echtzeitoptimalsteuerung parametergestörter Probleme*. Habilitation, Fachbereich Mathematik und Physik, Universität Bayreuth, Bayreuth, Germany, 2002.

[13] Cid Carvalho De Souza and Celso Carneiro Ribeiro. Heuristics for the minimum rectilinear Steiner tree problem: new algorithms and a computational study. *Discrete Applied Mathematics*, 45(3):205–220, 1993.

[14] Guy Desaulniers, Jacques Desrosiers, and Marius M. Solomon, editors. *Column generation*, volume 5 of *GERAD 25th Anniversary Series*. Springer, New York, 2005.

[15] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numer. Math.*, 1:269–271, 1959.

[16] Pontryagin et al. *The mathematical theory of optimal processes*. Interscience, New York, 1962.

[17] A. V. Fiacco. *Introduction to Sensitivity and Stability Analysis in Nonlinear Programming*, volume 165 of *Mathematics in Science and Engineering*. Academic Press, New York, 1983.

[18] R. Fletcher. *Practical Methods of Optimization*. Wiley & Sons, Chichester / New York, 1997.

[19] M. R. Garey and D. S. Johnson. The Rectilinear Steiner Tree Problem is NP-Complete. *SIAM Journal on Applied Mathematics*, 32(4):826–834, 1977.

[20] R. Geraerts. Planning short paths with clearance using explicit corridors. In *IEEE International Conference on Robotics and Automation*, pages 1997–2004, 2010.

[21] P. E. Hart, N. J. Nilsson, and B. Raphael. A Formal Basis for the Heuristic Determination of Minimum Cost Paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2):100–107, 1968.

[22] S. Held, B. Korte, D. Rautenbach, and J. Vygen. Combinatorial Optimization in VLSI design. In V. Chvátal and N. Sbihi, editors, *Combinatorial Optimization: Methods and Applications*. IOS Press, to appear.

[23] Renato F. Hentschke, Jaganathan Narasimham, Marcelo O. Johann, and Ricardo L. Reis. Maze routing Steiner trees with effective critical sink optimization. In *Proceedings of the 2007 international symposium on Physical design*, ISPD '07, pages 135–142, New York, NY, USA, 2007. ACM.

[24] Huibo Hou, Jiang Hu, and S.S. Sapatnekar. Non-Hanan routing. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 18(4):436–444, April 1999.

[25] Richard M. Karp. Reducibility among combinatorial problems. In *Complexity of computer computations*, pages 85–103. Plenum Press, New York, 1972.

[26] C. Y. Lee. An algorithm for path connections and its applications. *Electronic Computers, IRE Transactions on*, EC-10(3):346–365, 1961.

[27] Christine R. Leverenz and Miroslaw Truszczynski. The rectilinear Steiner tree problem: algorithms and examples using permutations of the terminal set. In *Proceedings of the 37th annual Southeast regional conference (CD-ROM)*, ACM-SE 37, New York, NY, USA, 1999. ACM.

[28] Chih-Hung Liu, Shih-Yi Yuan, Sy-Yen Kuo, and Szu-Chi Wang. High-performance obstacle-avoiding rectilinear Steiner tree construction. *ACM Transactions on Design Automation of Electronic Systems*, 14:45:1–45:29, 2009.

[29] Dirk Müller, Klaus Radke, and Jens Vygen. Faster min–max resource sharing in theory and practice. *Mathematical Programming Computation*, 3:1–35, 2011. 10.1007/s12532-011-0023-y.

[30] S. Peyer, D. Rautenbach, and J. Vygen. A generalization of Dijkstra's shortest path algorithm with applications to VLSI routing. *Journal of Discrete Algorithms*, 7:377–390, 2009.

[31] J. Scott Provan. An approximation scheme for finding Steiner trees with obstacles. *SIAM J. Comput.*, 17(5):920–934, 1988.

[32] Dirk Schwanenberg, Govert Verhoeven, and Luciano Raso. Nonlinear model predictive control of water resources systems in operational flood forecasting. In *55. IWK*. TU Ilmenau, September 2010.

[33] David Warme, Pawel Winter, and Martin Zachariasen. GeoSteiner [Computer Software]. www.diku.dk/hjemmesider/ansatte/martinz/geosteiner/. (version 3.1).

[34] Martin Zachariasen. Rectilinear full Steiner tree generation. *Networks*, 33(2):125–143, 1999.

[35] Hai Zhou. Efficient Steiner tree construction based on spanning graphs. In *Proceedings of the 2003 international symposium on Physical design*, ISPD '03, pages 152–157, New York, NY, USA, 2003. ACM.