GPU-Accelerated Stochastic Predictive Control of Drinking Water Networks

Sampathirao, A. K., Sopasakis, P., Bemporad, A., & Patrinos, P. (2017). GPU-Accelerated Stochastic Predictive Control of Drinking Water Networks. IEEE Transactions on Control Systems Technology, 26(2), 551-562. https://doi.org/10.1109/TCST.2017.2677741

Published in:
IEEE Transactions on Control Systems Technology

Document Version:
Peer reviewed version

Queen's University Belfast - Research Portal:
Link to publication record in Queen's University Belfast Research Portal

Publisher rights
Copyright 2017 IEEE. This work is made available online in accordance with the publisher's policies. Please refer to any applicable terms of use of the publisher.

General rights
Copyright for the publications made accessible via the Queen's University Belfast Research Portal is retained by the author(s) and / or other copyright owners and it is a condition of accessing these publications that users recognise and abide by the legal requirements associated with these rights.

Take down policy
The Research Portal is Queen's institutional repository that provides access to Queen's research output. Every effort has been made to ensure that content in the Research Portal does not infringe any person's rights, or applicable UK laws. If you discover content in the Research Portal that you believe breaches copyright or violates any law, please contact openaccess@qub.ac.uk.

Download date: 19. Jun. 2020
GPU-accelerated stochastic predictive control of drinking water networks

Ayjay K. Sampathirao, Pantelis Sopasakis, Alberto Bemporad Fellow, IEEE and Panagiotis Patrinos

Abstract—Despite the proven advantages of scenario-based stochastic model predictive control for the operational control of water networks, its applicability is limited by its considerable computational footprint. In this paper we fully exploit the structure of these problems and solve them using a proximal gradient algorithm parallelizing the involved operations. The proposed methodology is applied and validated on a case study: the water network of the city of Barcelona.

Index Terms—Stochastic model predictive control (SMPC), Graphics processing units (GPU), Drinking water networks.

I. INTRODUCTION

A. Motivation

Water utilities involve energy-intensive processes, complex in nature (dynamics) and form (topology of the network), of rather large scale and with interconnected components, subject to uncertain water demands from the consumers and are required to supply water uninterruptedly. These challenges call for operational management technologies able to provide reliable closed-loop behavior in presence of uncertainty. In 2014, the IEEE Control Systems Society identified many aspects of the management of complex water networks as emerging future research directions.

Stochastic model predictive control is an advanced control scheme which can address effectively the above challenges and has already been used for the management of water networks [2], [3]. However, unless restrictive assumptions are adopted regarding the form of the disturbances, such problems are known to be computationally intractable [3], [4]. In this paper we combine an accelerated dual proximal gradient algorithm with general-purpose graphics processing units (GPGPUs) to deliver a computationally feasible solution for the control of water networks.

B. Background

The pump scheduling problem (PSP) is an optimal control problem for determining an open-loop control policy for the operation of a water network. Such open-loop approaches are known since the 80’s [5], [6]. More elaborate schemes have been proposed such as [7] where a nonlinear model is used along with a demand forecasting model to produce an optimal open-loop 24-hour-ahead policy. Recently, the problem was formulated as a mixed-integer nonlinear program to account for the on/off operation of the pumps [8]. Heuristic approaches using evolutionary algorithms, genetic algorithms, and simulated annealing have also appeared in the literature [9]. However, a common characteristic and shortcoming of these studies is that they assume to know the future water demand and they do not account for the various sources of uncertainty which may alter the expected smooth operation of the network.

The effect of uncertainty can be attenuated by feedback from the network combined with the optimization of a performance index taking into account the system dynamics and constraints as in PSP. This, naturally, gives rise to model predictive control (MPC) which has been successfully used for the control of drinking water networks [10], [11]. Recently, Bakker et al. demonstrated experimentally on five full-scale water supply systems that MPC will lead to a more efficient water supply and better water quality than a conventional level controller [12]. Distributed and decentralized MPC formulations have been proposed for the control of large-scale water networks [13], [14] while MPC has also shown to be able to address complex system dynamics such as the Hazen-Williams pressure-drop model [15].

Most MPC formulations either assume exact knowledge of the system dynamics and future water demands [11], [14] or endeavor to accommodate the worst-case scenario [10], [16]–[18]. The former approach is likely to lead to adverse behavior in presence of disturbances which inevitably act on the system while the latter turns out to be too conservative as we will later demonstrate in this paper.

When probabilistic information about the disturbances is available it can be used to refine the MPC problem formulation. The uncertainty is reflected onto the cost function of the MPC problem deeming it a random variable; in stochastic MPC (SMPC) the index to minimize is typically the expectation of such a random cost function under the (uncertain) system dynamics and state/input constraints [19], [20].

SMPC leads to the formulation of optimization problems over spaces of random variables which are, typically, infinite-dimensional. Assuming that disturbances follow a normal probability distribution facilitates their solution [4], [21], [22]; however, such an assumption often fails to be realistic. The normality assumption has also been used for the stochastic control of drinking water networks aiming at delivering high quality of services – in terms of demand satisfaction – while minimizing the pumping cost under uncertainty [2].

An alternative approach, known as scenario-based stochastic MPC, treats the uncertain disturbances as discrete random
variables without any restriction on the shape of their distribution \[23\]–\[25\]. The associated optimization problem in these cases becomes a discrete multi-stage stochastic optimal control problem \[26\]. Scenario-based problems can be solved algorithmically, however, their size can be prohibitively large making them impractical for control applications of water networks as pointed out by Goryashko and Nemirovski \[17\]. This is demonstrated by Grosso et al. who provide a comparison of the two approaches \[3\]. Although compression methodologies have been proposed – such as the scenario tree generation methodology of Heitsch and Römisch \[27\] – multi-stage stochastic optimal control problems may still involve up to millions of decision variables.

Graphics processing units (GPUs) have been used for the acceleration of the algorithmic solution of various problems in signal processing \[28\], computer vision and pattern recognition \[29\] and machine learning \[30\], \[31\] leading to a manifold increase in computational performance. To the best of the authors’ knowledge, this paper is the first work in which GPU technology is used for the solution of a stochastic optimal control problem.

C. Contributions

In this paper we address this challenge by devising an optimization algorithm which makes use of the problem structure and sparsity. We exploit the structure of the problem, which is dictated by the structure of the scenario tree, to parallelize the involved operations. Then, the algorithm runs on a GPU hardware leading to a significant speed-up.

We first formulate a stochastic MPC problem using a linear hydraulic model of the water network while taking into account the uncertainty which accompanies future water demands. We propose an accelerated dual proximal gradient algorithm for the solution of the optimal control problem and report results in comparison with a CPU-based solver.

Finally, we study the performance of the closed-loop system in terms of quality of service and process economics using the Barcelona drinking water network as a case study. We show that the number of scenarios allows us to refine our representation of uncertainty and trade the economic operation of the network for reliability and quality of service.

D. Mathematical preliminaries

Let \(\mathbb{R}^n = \mathbb{R} \cup \{+\infty\}\) denote the set of extended-real numbers. The set of nonnegative integers \(\{k_1, k_1 + 1, \ldots, k_2\}, k_2 \geq k_1\) is denoted by \(\mathbb{N}[k_1, k_2]\). For \(x \in \mathbb{R}^n\) we define \([x]_+\) to be the vector in \(\mathbb{R}^n\) whose \(i\)-th element is \(\max\{0, x_i\}\). For a matrix \(A \in \mathbb{R}^{n \times m}\) we denote its transpose by \(A^\prime\).

The indicator function of a set \(C \subseteq \mathbb{R}^n\) is the extended-real valued function \(\delta(\cdot; C) : \mathbb{R}^n \to \mathbb{R}\) and it is \(\delta(x; C) = 0\) for \(x \in C\) and \(\delta(x; C) = +\infty\) otherwise. A function \(f : \mathbb{R}^n \to \mathbb{R}\) is called proper if there is a \(x \in \mathbb{R}^n\) so that \(f(x) < \infty\) and \(f(x) > -\infty\) for all \(x \in \mathbb{R}^n\). A proper convex function \(f : \mathbb{R}^n \to \mathbb{R}\) is called lower semi-continuous or closed if for every \(x \in \mathbb{R}^n\), \(f(x) = \liminf_{z \to x} f(z)\). For a proper closed convex function \(f : \mathbb{R}^n \to \mathbb{R}\), we define its conjugate as \(f^\ast(y) = \sup_x \{yx - f(x)\}\). We say that \(f\) is \(\sigma\)-strongly convex if \(f(x) - \frac{\sigma}{2}\|x\|^2_2\) is a convex function. Unless otherwise stated, \(\|\cdot\|\) stands for the Euclidean norm.

II. Modeling of Drinking Water Networks

A. Flow-based control-oriented model

Dynamical models of drinking water networks have been studied in depth in the last two decades \[11\], \[14\], \[32\]. Flow-based models are derived from simple mass balance equations of the network which lead to the following pair of equations

\[
\begin{align*}
x_{k+1} &= Ax_k + Bu_k + G_d d_k, \\
oparagraph \noindent 0 &= E u_k + E_d d_k,
\end{align*}
\]

where \(x \in \mathbb{R}^{n_x}\) is the state vector corresponding to the volumes of water in the storage tanks, \(u \in \mathbb{R}^{n_u}\) is the vector of manipulated inputs and \(d \in \mathbb{R}^{n_d}\) is the vector of water demands. Equation (1a) forms a linear time-invariant system with additive uncertainty and (1b) is an algebraic input-disturbance coupling equation with \(E \in \mathbb{R}^{n_x \times n_u}\) and \(E_d \in \mathbb{R}^{n_x \times n_d}\) where \(n_x\) is the number of functions in the network.

The maximum capacity of the tanks and the maximum pumping capacity of each pumping station is described by the following bounds:

\[
\begin{align*}
\text{u}_{\text{min}} &\leq u_k \leq \text{u}_{\text{max}}, \\
\text{x}_{\text{min}} &\leq x_k \leq \text{x}_{\text{max}}.
\end{align*}
\]

The above formulation has been widely used in the formulation of model predictive control problems for DWNs \[2\], \[10\], \[14\].

B. Demand prediction model

The water demand is the main source of uncertainty that affects the dynamics of the network. Various time series models have been proposed for the forecasting of future water demands such as seasonal Holt-Winters, seasonal ARIMA, BATS and SVM \[10\], \[33\]. Such models can be used to predict nominal forecasts of the upcoming water demand along a horizon of \(N\) steps ahead using measurements available up to time \(k\), denoted by \(d_{k+j|k}\). Then, the actual future demands \(d_{k+j}\) which are unknown to the controller at time \(k\) — can be expressed as

\[
d_{k+j}(\epsilon_j) = \hat{d}_{k+j|k} + \epsilon_j,
\]

where \(\epsilon_j\) is the demand prediction error which is a random variable on a probability space \((\Omega, \mathcal{F}, P)\) and for convenience we define the tuple \(\epsilon_j = (\epsilon_0, \epsilon_1, \ldots, \epsilon_j)\) which is a random variable in the product probability space. We also define \(d_k = (\hat{d}_{k|k}, \ldots, \hat{d}_{k+N-1|k})\).

III. Stochastic MPC for DWNs

In this section we define the control objectives for the controlled operation of a DWN and we formulate the stochastic MPC problem.
A. Control objectives

We define the following three cost functions which reflect our control objectives. The economic cost quantifies the production and transportation cost

\[ \ell^e(u_k, k) = W_\alpha (\alpha_1 + \alpha_2, x)_k, \]

where the term \( \alpha_1, x \) is the water production cost, \( \alpha_2, x \) is the pumping (electricity) cost and \( W_\alpha \) is a positive scaling factor.

The smooth operation cost is defined as

\[ \ell^s(\Delta u_k) = \Delta u_k W_s \Delta u_k, \]

where \( \Delta u_k = u_k - u_{k-1} \) and \( W_s \in \mathbb{R}^{n_x \times n_x} \) is a symmetric positive definite weight matrix. It is introduced to penalize abrupt switching of the actuators (pumps and valves).

The safety operation cost penalizes the drop of water level in the tanks below a given safety level. An elevation above this safety level ensures that there will be enough water in unforeseen cases of unexpectedly high demand and also maintains a minimum pressure for the flow of water in the network. This is given by

\[ \ell^s(x_k) = W_x \text{dist}(x_k | C_s), \]

where \( \text{dist}(x | C) = \inf_{y \in C} \|x - y\|_2 \) is the distance-to-set function, \( C_s = \{x | x \geq x_s\} \), and \( x_s \in \mathbb{R}^{n_x} \) is the safety level and \( W_x \) is a positive scaling factor.

These cost functions have been used in many MPC formulations in the literature \cite{2, 10, 34}. A comprehensive discussion on the choice of these cost functions can be found in \cite{13}.

The total stage cost at a time instant \( k \) is the summation of the above costs and is given by

\[ \ell(x_k, u_k, u_{k-1}, k) = \ell^e(u_k, k) + \ell^s(\Delta u_k) + \ell^s(x_k). \]

B. SMPC formulation

We formulate the following stochastic MPC problem with decision variables \( \pi = \{u_{k+j} | k, x_{k+j+1} | k\} \in [0, N-1] \)

\[ V^*(p, q, d_k, k) = \min_{\pi} \mathbb{E} V(\pi, p, q, k), \]

where \( \mathbb{E} \) is expectation operator and

\[ V(\pi, p, q, k) = \sum_{j=0}^{N-1} \ell(x_{k+j} | k, u_{k+j} | k, u_{k+j-1} | k, k+j), \]

subject to the constraints

\[ x_{k+j} = \pi(p, q, d_k, k), \]

where the term \( \alpha \) is a positive scaling factor.

C. Scenario trees

A scenario tree is the structure which naturally follows from the finiteness assumption of \( \Omega_j \) and is illustrated in Fig. 3. A scenario tree describes a set of possible future evolutions of the state of the system known as scenarios. Scenario trees can be constructed algorithmically from raw data as in \cite{27}.

The nodes of a scenario tree are partitioned in stages. The (unique) node at stage \( k = 0 \) is called root and the nodes at the last stage are the leaf nodes of the tree. We denote the number of leaf nodes by \( n_s \). The number of nodes at stage \( k \) is denoted by \( \mu(k) \) and the total number of nodes of the tree is denoted by \( \mu \). A path connecting the root node with a leaf node is called a scenario. Non-leaf nodes define a set of children; at a stage \( j \in \mathbb{N}[0, N-1] \) for \( i \in \mu(j) \) the set of children of the \( i \)-th node is denoted by \( \text{child}(j, i) \subseteq \mathbb{N}[1, \mu(j+1)] \). At stage \( j \in \mathbb{N}[1, N] \) the \( i \)-th node \( i \in \mathbb{N}[1, \mu(j)] \) is reachable from a single node at stage \( k = 1 \) known as its ancestor which is denoted by \( \text{anc}(j, i) \in \mathbb{N}[1, \mu(j-1)] \).

Fig. 1: Collection of possible upcoming demands at a given time instant. These results were produced using the SVM model and the data in \cite{10}.

Fig. 2: The closed-loop system with the proposed stochastic MPC controller running on a GPU device.
The probability of visiting a node $i$ at stage $j$ starting from the root is denoted by $p^i_j$. For all $j \in \mathbb{N}_N$ we have that $\sum_{i=1}^{\mu(j)} p^i_j = 1$ and for all $i \in \mathbb{N}_{[1,\mu(k)]}$ it is $\sum_{l \in \text{child}(j,i)} p^j_{l+1} = p^i_j$.

We define the maximum branching factor at stage $j$, $b_j$, to be the maximum number of children of the nodes at this stage. The maximum branching factor serves as a measure of the complexity of the tree at a given stage.

### D. Reformulation as a finite-dimensional problem

We shall now exploit the above tree structure to reformulate the optimal control problem (8) as a finite-dimensional problem. The water demand, given by (3), is now modeled as

$$V_i = \sum_{j=0}^{\mu(i)} \sum_{k=1}^{\mu(2)} \ell^d(x_{k+j}^i, k) = \min_{\pi, p, q, \tilde{d}, k} \mathbb{E}V(\pi, p, q, k) + V_s(\pi, p),$$

subject to

$$x_{k+j}^i = p, \quad u_{k-1}^i = q,$$

$$u_{\min} \leq u_{k+j}^i \leq u_{\max}; \quad j \in \mathbb{N}_{[0,N-1]}, i \in \mathbb{N}_{[1,\mu]}.$$

and system equations (10) and (12).

### IV. Solution of the Stochastic Optimal Control Problem

In this section we extend the GPU-based proximal gradient method proposed in [35] to solve the SMPC problem (16). For ease of notation we will focus on the solution of the SMPC problem at $k=0$ and denote $x_{j|0} = x_j, u_{j|0} = u_j, d_{j|0} = \tilde{d}_j$.

#### A. Proximal gradient algorithm

For a closed, proper extended-real valued function $g : \mathbb{R}^n \to \mathbb{R}$, we define its proximal operator with parameter $\gamma > 0$, $\text{prox}_{\gamma g} : \mathbb{R}^n \to \mathbb{R}^n$ as

$$\text{prox}_{\gamma g}(v) = \text{argmin}_{x \in \mathbb{R}^n} \left\{ g(x) + \frac{1}{2\gamma} \|x - v\|_2^2 \right\}.$$ (17)

The proximal operator of many functions is available in closed form [36, 37]. When $g$ is given in a separable sum form, that is

$$g(x) = \sum_{i=1}^{\mu} g_i(x_i),$$ (18a)

then,

$$\left(\text{prox}_{\gamma g}(v)\right)_i = \text{prox}_{\gamma g_i}(v).$$ (18b)

This is known as the separable sum property of the proximal operator.

Let $z \in \mathbb{R}^{n_z}$ be a vector encompassing all states $x^i_j$ for $j \in \mathbb{N}_{[0,N]}$ and $i \in \mathbb{N}_{[1,\mu]}$ and inputs $u^i_j$ for $j \in \mathbb{N}_{[0,N-1]}, i \in \mathbb{N}_{[1,\mu(j)+1]}$; this is the decision variable of problem (16).

Let $f : \mathbb{R}^{n_z} \to \mathbb{R}$ be defined as

$$f(z) = \sum_{j=0}^{N-1} \sum_{i=1}^{\mu(j)} p^i_j \left( \ell^w(u^i_j) + \ell^A(\Delta u^i_j) \right) + \delta(u^i_j | \Phi_1(d^i_j)) + \delta(x^i_{j+1}, u^i_j, x^i_{j+1} | \Phi_2(d^i_j)).$$ (19)
where $\Delta u^j_j = u^j_j - u^t_{j-1}$ and $\Phi_1(d)$ is the affine subspace of $\mathbb{R}^{n_u}$ induced by (10), that is
\[ \Phi_1(d) = \{ u : Eu + Ed = 0 \}, \tag{20} \]
and $\Phi_2(d)$ is the affine subspace of $\mathbb{R}^{2n_x + n_u}$ defined by the system dynamics (12)
\[ \Phi_2(d) = \{(x_{k+1}, x_k, u) : x_{k+1} = Ax_k + Bu + Gd_d \}. \tag{21} \]

We define the auxiliary variables $\zeta$ and $\zeta'$ which stand for copies of the state variables $x^i_j$ — that is $\zeta^i_j = \zeta'^i_j = x^i_j$ — and the auxiliary variable $\psi$ which is a copy of input variables $\psi^i_j = u^i_j$. The reason for the introduction of these variables will be clarified in Section IV-C.

We introduce the variable $t = (\zeta, \zeta', \psi) \in \mathbb{R}^{n_t}$ and define an extended real valued function $g : \mathbb{R}^{n_t} \to \mathbb{R}$ as
\[ g(t) = \sum_{j=0}^{N-1} \sum_{i=1}^{\mu(i)} \ell^j_i (\zeta^i_{j+1}) + \ell^d_i (\zeta^i_{j+1}) + \delta(\psi^i_j | U), \tag{22} \]
where $U = \{ \psi \in \mathbb{R}^{n_u} : \text{umin} \leq \psi \leq \text{umax} \}$.

Now the finite-dimensional optimization problem (16) can be written as:
\[ \hat{V}^* = \min_{z,t} f(z) + g(t), \tag{23a} \]
\[ \text{s.t. } Hz = t, \tag{23b} \]
\[ H = \begin{bmatrix} I_{n_x} & 0 \\ 0 & I_{n_u} \end{bmatrix}. \tag{24} \]

The Fenchel dual of (23) is written as [38, Corol. 31.2.1]:
\[ \hat{D}^* = \min_{y} f^*(-H'y) + g^*(y), \tag{25} \]
where $y$ is the dual variable. The dual variable $y$ can be partitioned as $y = (\zeta^i_j, \zeta'^i_j, \psi^i_j)$, where $\zeta^i_j$ and $\zeta'^i_j$ are the dual variables corresponding to $\zeta^i_j$ and $\zeta'^i_j$ respectively. We also define the auxiliary variable of state copies $\hat{\xi}^i_j := (\zeta^i_j, \zeta'^i_j)$.

According to [39, Thm. 11.42], since function $f(z) + g(Hz)$ is proper, convex and piecewise linear-quadratic, then the primal problem (23) is feasible whenever the dual problem (25) is feasible and, furthermore, strong duality holds, i.e., $\hat{V}^* = \hat{D}^*$. Moreover, the optimal solution of (23) is given by $y^* = \nabla f^*(-H'y^*)$ where $y^*$ is any solution of (25). Applying [39, Prop. 12.60] to $f^*$ and since $f$ is lower semi-continuous, proper and $\sigma$-strongly convex — as shown at the end of Appendix A — its conjugate $f^*$ has Lipschitz-continuous gradient with a constant $1/\sigma$.

An accelerated version of proximal-gradient method which was first proposed by Nesterov in [40] is applied to the dual problem. This leads to the following algorithm:
\[ w^\nu = y^\nu + \theta^\nu (\theta_{\nu-1}^{-1} - 1) (y^\nu - y^{\nu-1}), \tag{26a} \]
\[ z^\nu = \text{argmin}_z \{ z, H'w^\nu + f(z) \}, \tag{26b} \]
\[ t^\nu = \text{prox}_{l^1}\left( \lambda^{-1}l^1H'w^\nu + Hz^\nu \right), \tag{26c} \]
\[ y^{\nu+1} = \left( \sqrt{\theta^\nu_2 + 4\theta^\nu_1 \theta^\nu_2} - \theta^\nu_1 \right), \tag{26d} \]
\[ \theta^\nu_{\nu+1} = \frac{1}{2} \left( \sqrt{\theta^\nu_2 + 4\theta^\nu_1 \theta^\nu_2} - \theta^\nu_1 \right), \tag{26e} \]
starting from a dual-feasible vector $y^0 = y^{-1} = 0$ and $\theta_0 = \theta_{-1} = 1$.

In the first step (26a) we compute an extrapolation of the dual vector. In the second step (26b) we calculate the dual gradient, that is $z^\nu = \nabla f^*(-H'w^\nu)$, at the extrapolated dual vector using the conjugate subgradient theorem [38, Thm. 23.5]. The third step comprises of (26c) and (26d) where we update the dual vector $y$ and in the final step of the algorithm we compute the scalar $\theta^\nu$ which is used in the extrapolation step.

This algorithm has a convergence rate of $O(1/\nu^2)$ for the dual iterates as well as for the ergodic primal iterate defined through the recursion $z^\nu = (1 - \theta^\nu)z^{(\nu-1)} + \theta^\nu z^\nu$, i.e., a weighted average of the primal iterates [21].

### B. Computation of primal iterate

The most critical step in the algorithm is the computation of $z^\nu$ which accounts for most of the computation time required by each iteration. This step boils down to the solution of an unconstrained optimization problem by means of dynamic programming where certain matrices (which are independent of $w^\nu$) can be computed once before we run the algorithm to facilitate the online computations. These are (i) the vectors $\beta^j_i, \tilde{\beta}^j_i, e^j_i$ which are associated with the update of the time-varying cost (see Appendix A) and (ii) the matrices $\Lambda, \Phi, \Psi, \hat{B}$ (see Appendix B). The latter are referred to as the factor step of the algorithm and matrices $\Lambda, \Phi, \Psi, \hat{B}$ are independent of the complexity of the scenario tree.

The computation of $z^\nu$ at each iteration of the algorithm requires the computation of the aforementioned matrices and is computed using Algorithm 1 to which we refer as the solve step. Computations involved in the solve step are merely matrix-vector multiplications. As the algorithm traverses the nodes of the scenario tree stage-wise backwards (from stage $N - 1$ to stage 0), computations across the nodes at a given stage can be performed in parallel. Hardware such as GPUs which enable us to parallelize such operations lead to a great speed-up as we demonstrate in Section V.

### C. Computation of dual iterate

Function $g$ given in (22) is given in the form of a separable sum
\[ g(t) = g(\zeta, \zeta', \psi) = g_1(\zeta) + g_2(\zeta') + g_3(\psi), \tag{27} \]
where
\[ g_1(\zeta) = \sum_{j=0}^{N-1} \sum_{i=1}^{\mu(j)} \ell^j_i (\zeta^i_{j+1}), \tag{28a} \]
\[ g_2(\zeta') = \sum_{j=0}^{N-1} \sum_{i=1}^{\mu(j)} \ell^d_i (\zeta^i_{j+1}), \tag{28b} \]
\[ g_3(\psi) = \sum_{j=0}^{N-1} \sum_{i=1}^{\mu(j)} \delta(\psi^i_j | U). \tag{28c} \]

Functions $g_1(\cdot)$ and $g_2(\cdot)$ are in turn separable sums of distance functions from a set and $g_3(\cdot)$ is an indicator function. Their proximal mappings can be easily computed as in...
Algorithm 1 Solve step

Require: Output of the factor step (See Appendices A and B), i.e., \( \Lambda, \Phi, \Psi, \tilde{B}, \tilde{u}_j^0, \beta_j^0, \gamma_j^0, p, q \) and \( w^0 = (\tilde{c}_j^0, \tilde{c}_j^1, \psi_j^0) \).

\( q_N \leftarrow 0 \), and \( r_i^N \leftarrow 0 \), \( \forall i \in \mathbb{N}_{[1,n_s]} \).

\( \text{for} \ j = N - 1, \ldots, 0 \ \text{do} \)

\( \text{for} \ i = 1, \ldots, \mu(k) \ \text{do} \) \{in parallel\}

\[ \sigma_j^0 \leftarrow r_j^{i+1} + \beta_j^i, \forall j \in \text{child}(j, i) \]

\[ \psi_j^i \leftarrow \frac{1}{2\sigma_j^i} \left( \Phi_j^i \tilde{c}_j^k + q_j^{i+1} + \Psi_j^i \tilde{w}_j^i + \Lambda_j^i \sigma_j^0 \right), \]

\( \forall i \in \text{child}(j, i) \)

\[ r_j^i \leftarrow \sum_{i \in \text{child}(j, i)} \sigma_j^i + \tilde{B}^i (\tilde{c}_j^0 + q_j^{i+1}) + L \tilde{w}_j^i \]

\[ q_j^i \leftarrow A' \sum_{i \in \text{child}(j, i)} \tilde{c}_j^i + q_j^{i+1} \]

\( \text{end for} \)

\( \text{end for} \)

\( x_N^0 \leftarrow p, u_{-1} \leftarrow q, \)

\( \text{for} \ j = 0, \ldots, N - 1 \ \text{do} \)

\( \text{for} \ i = 1, \ldots, \mu(k) \ \text{do} \) \{in parallel\}

\[ \nu_j^i \leftarrow w_j^{\text{anc}(j,i)} + v_j^i \]

\[ \tilde{u}_j^i \leftarrow L \nu_j^i + \tilde{u}_j^i \]

\[ x_j^i+1 \leftarrow A x_j^{\text{anc}(j,i)} + \tilde{B} \nu_j^i + e_j^i \]

\( \text{end for} \)

\( \text{end for} \)

\( \text{return} \ \{x_j^i\}_{j=1}^N, \ \{u_j^i\}_{j=0}^{N-1} \)

Appendix C and essentially are element-wise operations on the vector \( t \) that can be fully parallelized.

D. Preconditioning and choice of \( \lambda \)

First-order methods are known to be sensitive to scaling and preconditioning can remarkably improve their convergence rate. Various preconditioning method such as [42], [43] have been proposed in the literature. Here, we employ a simple diagonal preconditioning which consists in computing a diagonal matrix \( \tilde{H}_D \) with positive diagonal entries which approximates the dual Hessian \( H_D \) and use \( \tilde{H}_D^{1/2} \) to scale the diagonal vector \([44] 2.3.1\). Since the uncertainty does not affect the dual Hessian, we take this preconditioning matrix for a single branch of the scenario tree and use it to scale all dual variables.

In a similar way, we compute the parameter \( \lambda \). We choose \( \lambda = 1/L_{\tilde{H}_D} \) where \( L_{\tilde{H}_D} \) is the Lipschitz constant of the dual gradient which is computed as \( \|H\|/\sigma \) as in [44]. It again suffices to perform the computation for a single branch of the scenario tree.

E. Termination

The termination conditions for the above algorithm are based on the ones provided in [41]. However, rather than checking these conditions at every iteration, we perform always a fixed number of iterations which is dictated by the sampling time. We may then check the quality of the solution \textit{a posteriori} in terms of the duality gap and the term \( \|H z^\nu - t^\nu\|_\infty \).

V. CASE STUDY: THE BARCELONA DWN

We now apply the proposed control methodology to the drinking water network of the city of Barcelona using the data found in [2], [10]. The topology of the network is presented in Figure 4. The system model consists of 63 states corresponding to the level of water in each tank, 114 control inputs which are pumping actions and valve positions, 88 demand nodes and 17 junctions. The prediction horizon is \( N = 24 \) with sampling time of 1 hour. The future demands are predicted using the SVM time series model developed in [10].

A. Performance of GPU-accelerated algorithm

Accelerated proximal gradient (APG) was implemented in CUDA-C v6.0 and the matrix-vector computations were performed using cuBLAS. We compared the GPU-based implementation with the interior-point solver of Gurobi. Active-set algorithms exhibited very poor performance and we did not include the respective results.

All computations on CPU were performed on a 4\( \times \)2.60GHz Intel i5 machine with 8GB of RAM running 64-bit Ubuntu v14.04 and GPU-based computations were carried out on a NVIDIA Tesla C2075.

The dependence of the computational time on the size of the scenario tree is reported in Figure 5 where it can be noticed that there is speed-up of 10\( \times \) to 25\( \times \) in the computational times with CUDA-APG compared to Gurobi. Furthermore, the speed-up increases with the number of scenarios.

The optimization problems we are solving here are of noticeably large size. Indicatively, the scenario tree with 493 scenarios counts approximately 2.52 million dual decision variables (1.86 million primal variables) and while Gurobi requires 1329s to solve it, our CUDA implementation solves it in 58.8s; this corresponds to a speed-up of 22.6\( \times \).

In all of our simulations we obtained a sequence of control actions across the tree nodes \( U_{\text{appp}}^* = \{u_j^i\} \) which was, element-wise, within \( \pm 0.029 m^3/s \) (1.9\%) from the solution produced by Gurobi. The maximum primal residual was \( \|Hz - z\|_\infty = 1.7446 \). Moreover, we should note that the control action \( u_0^i \) computed by APG with 500 iterations was consistently within \( \pm 0.0025 m^3/s \) (0.08\%) from the Gurobi solution. Given that only \( u_0^i \) is applied to the system while all other control actions \( u_j^i \) for \( j \in \mathbb{N}_{[1,N-1]} \) and \( i \in \mathbb{N}_{[1,\mu(j)]} \) are discarded, 500 iterations are well sufficient for convergence.
B. Closed-loop performance

In this section we analyse the performance of SMPC with different scenario-trees. This analysis is carried for a period of 7 days ($H_s = 168$) from 1st to 8th July 2007. Here, we compare the operational cost and the quality of service of various scenario-tree structures.

The weighting matrices in the operational cost are chosen as $W_a = 2 \cdot 10^4$, $W_u = 10^5 \cdot I$ and $W_x = 10^7$, respectively and $\gamma_d = 5 \cdot 10^7$. The demand is predicted using SVM model presented in [10]. The steps involved in SMPC using GPU based APG in closed-loop is summarized in Algorithm 2.

Algorithm 2 Closed-loop of DWN with SMPC with proximal-operator

Require: Scenario-tree, current state measurement $x_0$ and previous control $u_{-1}$.
Compute $\Lambda, \Phi, \Psi$ and $B$ as in Appendix B
Precondition the original optimization problem and compute $\lambda$ as in Section IV-D

Loop

Step 1. Predict the future water demands $d_k$ using current and past demand data.
Step 2. Compute $\alpha_j, \beta_j, \epsilon_j$ as in Appendix A
Step 3. Solve the optimization problem using APG on GPU using iteration (26) and Algorithm 1
Step 4. Apply $u_0$ to the system, update the previous

end loop

For the performance assessment of the proposed control methodology we used various controllers summarized in Table I. The corresponding computational times are presented in Figure 5.

To assess the performance of closed-loop operation of the SMPC-controlled network we used the key performance indicators (KPIs) reported in [3], [45]. For a simulation time length $H_s$, the performance indicators are computed by

$$KPI_E = \frac{1}{H_s} \sum_{k=1}^{H_s} (\alpha_1 + \alpha_2 \| u_k \|_1)$$  
$$KPI_{\Delta U} = \frac{1}{H_s} \sum_{k=1}^{H_s} \| \Delta u_k \|^2$$  
$$KPI_S = \sum_{k=1}^{H_s} \| x_s - x_k \|_1$$  
$$KPI_R = \frac{1}{H_s} \sum_{k=1}^{H_s} \| x_k \|_1 \times 100\%.$$  

$KPI_E$ is the average economic cost, $KPI_{\Delta U}$ measures the average smoothness of the control actions, $KPI_S$ corresponds to the total amount of water used from storage and $KPI_R$ is the percentage of the safety volume $x_s$ contained into the average volume of water.

| Controller | $b_k$ | scenarios | primal variables | dual variables |
|------------|-------|-----------|-----------------|---------------|
| CE-MPC    | 1     | 1         | 4248            | 5760          |
| SMPC$_1$  | [4, 2]| 6         | 24072           | 32540         |
| SMPC$_2$  | [6, 8]| 30        | 118059          | 16080         |
| SMPC$_3$  | [6, 8]| 114       | 430287          | 583440        |
| SMPC$_4$  | [8, 8]| 146       | 551355          | 747600        |
| SMPC$_5$  | [10, 8]| 242       | 915621          | 1241520       |
| SMPC$_6$  | [12, 8]| 363       | 1145044         | 1553280       |
| SMPC$_7$  | [12, 8]| 404       | 1520861         | 2062320       |
| SMPC$_8$  | [12, 10]| 493     | 1856022         | 2516640       |

Table I: Various controllers used to assess the closed-loop performance of the proposed methodology. The numbers in the bracket denote the first maximum branching factors, $b_j$, of the scenario tree while all subsequent branching factors are assumed to be equal to 1.

1) Risk vs Economic utility: Figure 6 illustrates the trade-off between economic and safe operation: The more scenarios we use to describe the distribution of demand prediction error, the safer the closed-loop operation becomes as it is reflected by the decrease of KPI$_S$. Stochastic MPC leads to a significant decrease of economic cost compared to the certainty-equivalence approach, however, the safer we require the operation to be, the higher the operating cost we should expect.

2) Quality of service: A measure of the reliability and quality-of-service of the network is KPI$_R$ which reflects the tendency of water levels to drop under the safety storage levels.
As expected, the CE-MPC controller leads to the most unsafe operation, whereas SMPC\textsubscript{0} leads to the lowest value.

3) Network utility: Network utility is defined as the ability to utilize the water in the tanks to meet the demands rather than pumping additional water and is quantified by KPI\textsubscript{R}. In Table II we see the dependence of KPI\textsubscript{R} on the number of scenarios of the tree. KPI\textsubscript{R} remains always within reasonable limits; on average we operate away from the safety storage limit. The decrease in KPI\textsubscript{R} on may observe is because as more scenarios are employed, the more accurate the representation of uncertainty becomes and the system does not need to operate, on average, too far away from \( x_s \).

4) Smooth operation: We may notice that the introduction of more scenarios results in an increase in KPI\textsubscript{\Delta U}. Then, the controller becomes more responsive to accommodate the need for a less risky operation, although the value of KPI\textsubscript{\Delta U} is not greatly affected by number of scenarios.

![Fig. 6: The figure shows the trade-off between risk and economic utility in terms of scenarios. The KPI\textsubscript{E} represent the economical utility and KPI\textsubscript{S} shows the risk of violation.](image)

**Fig. 6**

![Fig. 7: A pumping action using SMPC\textsubscript{4} (expressed in % of \( u_{\text{max}} \)) and the corresponding weighted time-varying cost \( W_{\alpha}\beta_{2,k} \) in economic units.](image)

**Fig. 7**

C. Implementation details

At every time instant \( k \) we need to load onto the GPU the state measurement and a sequence of demand predictions (see Figure 3), that is \( d_k \). This amounts to 8.4kB and is rapidly uploaded on the GPU (less than 0.034ms). In case we need to update the scenario-tree values, that is \( \epsilon_k \), and for the case of SMPC\textsubscript{S} we need to upload 3.52MB which is done in 3.74ms. Therefore, the time needed to load these data on the GPU is not a limiting factor.

VI. CONCLUSIONS

In this paper we have presented a framework for the formulation of a stochastic model predictive control problem for the operational management of drinking water networks and we have proposed a novel approach for the efficient numerical solution of the associated optimization problem on a GPU.

We demonstrated the computational feasibility of the algorithm and the benefits for the operational management of the system in terms of performance (which we quantified using certain KPIs from the literature).

APPENDIX A

ELIMINATION OF INPUT-DISTURBANCE COUPLING

In this section we discuss how the input-disturbance equality constraints can be eliminated by a proper change of input variables and we compute the parameters \( \beta_{j}^{\prime}, \hat{u}_{j}, e_{j} \) \( \forall i \in \mu(j), j \in \mathbb{N}_{N} \) which are then provided as input to Algorithm I. These depend on the nominal demand forecasts \( d_{k+j} \) and on the time-varying economic cost parameters \( \alpha_{2,k+j} \) for \( j \in \mathbb{N}_{[0,N-1]} \), therefore, they need to be updated at every time instant \( k \).

The affine space \( \Phi_{1}(d) \) introduced in (20) can be written as

\[
\Phi_{1}(d) = \{ v \in \mathbb{R}^{n_{e}} : u = Lv + \hat{u}(d) \},
\]

(30)

where \( L \in \mathbb{R}^{n_{e} \times n_{e}} \) is a full rank matrix whose range spans the nullspace of \( E \), i.e., for every \( v \in \mathbb{R}^{n_{e}} \), we have \( Lv \) is in the kernel of \( E \) and \( \hat{u}(d) \) satisfies \( E\hat{u}(d) + E_{d}d = 0 \).

Substituting \( u_{j}^{\prime} = Lv_{j} + \hat{u}_{j}, \forall i \in \mu(j), j \in \mathbb{N}_{N} \) in the dynamics \( \Phi_{2}(d) \) in (21) gives

\[
\Phi_{2}(d) = \{ (x_{j+1}, x_{j}, v) : x_{j+1} = Ax_{j} + Bv + e, \newline
B = BL, e = B\hat{u} + G_{d}d \},
\]

(31)

and we define

\[
e_{j}^{\prime} = B\hat{u}_{j} + G_{d}d_{j}.
\]

(32)

Now the cost in (19) is transformed as:

\[
\sum_{j=0}^{N-1} \mu(j) \sum_{i=1}^{\mu(j)} p_{j}^{\epsilon}(\ell^{w}(u_{j}^{i}) + \ell^{\Delta}(\Delta u_{j}^{i})) = \newline
\sum_{j=0}^{N-1} \mu(j) \sum_{i=1}^{\mu(j)} p_{j}^{\epsilon}(\ell^{w}(v_{j}^{i}) + \ell^{\Delta}(\Delta v_{j}^{i}, \hat{u}_{j}^{i})),
\]

(33)
where
\[
\bar{R} = W_n L, \\
\tilde{R} = L' \bar{R}, \\
\tilde{R} = L' \bar{R}, \\
\delta_j = W_\alpha (\alpha_1 + \alpha_2 j + k) L,
\]
(34a)
(34b)
(34c)
(34d)
(34e)
(34f)

By substituting and expanding \(\Delta v_j^i\) and \(\Delta u_j^i\) in \(\ell(\Delta v_j^i, \Delta u_j^i)\) the cost in (34g) becomes
\[
\sum_{j=0}^{N-1} \mu(j) \sum_{i=1}^{\mu(j)} p_j^i (\ell^w(u_j^i) + \ell^c(\Delta u_j^i)) = \\
\sum_{j=0}^{N-1} \mu(j) \sum_{i=1}^{\mu(j)} p_j^i v_j^i - 2p_j^i v_j^i \bar{v}_j + \beta_j^i v_j^i,
\]
(35)

where
\[
p_j^i = p_j^i + \sum_{l \in child(j,i)} p_{j+1}^l, \\
\beta_j^i = p_j^i \tilde{\alpha}_j + 2p_j^i \tilde{R} (p_j^i u_j^i - \tilde{u}^{anc(j,i)} - \sum_{l \in child(j,i)} p_{j+1}^l \tilde{u}_{j+1}^i).
\]
(36a)
(36b)

Now \(u_j^i, v_j^i, \beta_j^i\) are calculated from (30), (32) and (36b) respectively. Using our assumption that \(L\) is full-rank, we can see that \(\tilde{R}\) is a positive definite and symmetric matrix, therefore, \(f\) is strongly convex.

**APPENDIX B**

**FACTOR step**

Algorithm (1) solves the unconstrained minimization problem (26b), that is
\[
z^* = \arg\min_z \{z, H' y + f(z)\},
\]
(37)

where \(z = \{x_j^i, u_j^i\}, y = \{\tilde{c}_j, \tilde{\psi}_j\}\) for \(i \in \mathbb{N}_{1, \mu(j)}\) and \(j \in \mathbb{N}_{0, N}\), \(f(z)\) is given by (19) and \(H\) is given by (24).

Substituting \(H\) the optimization problem becomes
\[
z^* = \arg\min_z \sum_{j=0}^{N-1} \mu(j) \sum_{i=1}^{\mu(j)} p_j^i (\ell^w(u_j^i) + \ell^c(\Delta u_j^i)) + \hat{\psi}_j^i x_j^i + \tilde{\chi}_j^i x_j^i + \delta(u_j^i | \Phi_1(d_j^i)) + \delta(x_j^i+1, u_j^i, x_j^{anc(j,i)} | \Phi_2(d_j^i)),
\]
(38)

where \(\hat{\psi}_j^i := (\tilde{c}_j, \tilde{\psi}_j)\).

The input-disturbance coupling constraints imposed by \(\delta(u_j^i | \Phi_1(d_j^i))\) in the above problem are eliminated as discussed in Appendix A. This changes the input variable from \(u_j^i\) to \(v_j^i\) given by (30) and the cost function as in (35). We, therefore, replace the decision variable \(z\) with \(\tilde{z} := \{x_j^i, v_j^i\}\) and the optimization problem (38) reduces to
\[
z^{**} = \arg\min_{\tilde{z}} \sum_{j=0}^{N-1} \sum_{i=1}^{\mu(j)} p_j^i v_j^i \tilde{R} v_j - 2p_j^i v_j^i \bar{v}_j + \beta_j^i v_j^i + \hat{\psi}_j^i x_j+1 + \hat{\psi}_j^i \bar{v}_j
\]
\[
+ \delta(x_j+1, v_j^i, x_j^{anc(k,i)} | \Phi_2(d_j^i)),
\]
(39)

where \(u_j^i = L v_j^i + \tilde{u}_j^i\).

The above problem is an unconstrained optimization problem with quadratic stage cost which is solved using dynamic programming [46]. This method transforms the complex problem into a sequence of sub-problems solved at each stage.

Using dynamic programming we find that the transformed control actions \(v_j^*\) have to satisfy
\[
v_j^* = v_j^\prime + \frac{1}{2p_j^i} (\Phi(\tilde{c}_j + \tilde{\psi}_j) + \bar{\psi}_j^i) + \Lambda(\beta_j^i + r_{j+1})^i,
\]
(40)

where
\[
\Lambda = -\bar{R}^{-1}, \\
\Phi = \bar{A} B', \\
\Psi = \Lambda L.
\]
(41a)
(41b)
(41c)

Matrix \(\bar{R}\) is symmetric and positive definite, therefore, we can compute once its Cholesky factorization so that we obviate the computation of its inverse.

The \(r_{j+1}^i, r_{j+1}^i\) in (40) correspond to the linear cost terms in the cost-to-go function at node \(i\) of stage \(j + 1\). At stage \(j\), these terms are updated by substituting the \(v_j^*\) as:
\[
r_j^s = \sum_{l \in child(j-1, s)} \sigma_j^s + \bar{B}'(\tilde{\gamma}_j^s + \tilde{\psi}_j^s) + \bar{L}\bar{\psi}_j^s,
\]
(42a)
\[
q_j^s = \bar{A}' \sum_{l \in child(j-1, s)} \tilde{\gamma}_j^s + q_{j+1}^s,
\]
(42b)

where \(s = \text{anc}(j, i)\).

Equations (40) and (42) form the solve step as in Algorithm (1). Matrices \(\Lambda, \Phi\) and \(\Psi\) are required to be computed once.

**APPENDIX C**

**PROXIMAL OPERATORS**

Function \(g\) in (27) is a separable sum of distance and indicator functions and its proximal is computed according to (18). The proximal operator of the indicator of a convex closed set \(C\), that is
\[
\chi_C(x) = \begin{cases} 
0, & \text{if } x \in C \\
+\infty, & \text{otherwise}
\end{cases}
\]

is the projection operator onto \(C\), i.e.,
\[
\text{prox}_{\chi_C}(v) = \text{proj}_C(v) = \arg\min_{y \in C} \|v - y\|,
\]
(43)
When $g$ is the distance function from a convex closed set $C$, that is
\[
g(x) = \mu \operatorname{dist}(x \mid C) = \inf_{y \in C} \mu \| x - y \|
\]

Then proximal operator of $g$ given by \cite{37}
\[
\operatorname{prox}_{\lambda g}(v) = \begin{cases} 
  x + \frac{\mu}{\lambda} \frac{x - \operatorname{proj}_C(x)}{\operatorname{dist}(x \mid C)}, & \text{if dist}(x \mid C) > \lambda \\
  \operatorname{proj}_C(x), & \text{otherwise}
\end{cases}
\]

ACKNOWLEDGMENT
This work was financially supported by the EU FP7 research project EFFINET “Efficient Integrated Real-time monitoring and Control of Drinking Water Networks,” grant agreement no. 318556.

REFERENCES

[1] V. Havlena, P. Trnka, and B. Sheridan, “Management of complex water networks,” in The Impact of Control Technology (T. Samad and A. Annaswamy, eds.), IEEE Control Systems Society, 2nd ed., 2014. Available at www.ieeeess.org.

[2] J. Grosso, C. Ocampo-Martinez, V. Puig, and B. Joseph, “Chance-constrained model predictive control for drinking water networks,” Journal of Process Control, vol. 24, no. 5, pp. 504 – 516, 2014.

[3] J. Grosso, J. Maestre, C. Ocampo-Martinez, and V. Puig, “On the assessment of tree-based and chance-constrained predictive control approaches applied to drinking water networks,” in 19th IFAC Conference, (Cape town, South Africa), pp. 6240–6245, Aug. 2014.

[4] A. Nemirovski and A. Shapiro, “Convex approximations of chance-constrained programs,” SIAM Journal on Optimization, vol. 17, no. 4, pp. 969–996, 2006.

[5] J. Creasy, “Pump scheduling in water supply: More than a mathematical problem,” Computer Applications in Water Supply, vol. 2, pp. 279–289, 1998.

[6] U. Zessler and U. Shamir, “Optimal operation of water distribution systems,” Journal of Water Resources Planning and Management, vol. 115, no. 6, pp. 735–752, 1989.

[7] G. Yu, R. Powell, and M. Sterling, “Optimized pump scheduling in water distribution systems,” Journal of Optimization Theory and Applications, vol. 83, no. 3, pp. 463–488, 1994.

[8] A. Bagirov, A. Barton, H. Mala-Jetmarova, A. A. Nuaimat, S. Ahmed, N. Sullanova, and J. Yearwood, “An algorithm for minimization of storage facilities,” in 19th IFAC Conference, (Cape Town, South Africa), pp. 6240–6245, Aug. 2014.

[9] G. Calafiore and M. Campi, “The scenario approach to robust control design,” Automatic Control, IEEE Transactions on, vol. 51, pp. 742–753, May 2006.

[10] M. C. Campi, S. Garatti, and M. Prandini, “The scenario approach for systems and control design,” Annual Reviews in Control, vol. 33, no. 2, pp. 149 – 157, 2009.

[11] M. Prandini, S. Garatti, and J. Lygeros, “A randomized approach to stochastic model predictive control,” in Decision and Control (CDC), 2012 IEEE 51st Annual Conference on, (Maui, Hawaii, USA), pp. 7315–7320, Dec. 2012.

[12] A. Shapiro, D. Dentcheva, and A. Ruszczynski, Lectures on Stochastic Programming: Modeling and Theory, Philadelphia: Society for Industrial and Applied Mathematics, 2009.

[13] H. Heitsch and W. Römisch, “Scenario tree modeling for multistage stochastic programs,” Mathematical Programming, vol. 118, no. 2, pp. 371–406, 2009.

[14] M. McCool, “Signal processing and general-purpose computing and gpus [exploratory dsp],” Signal Processing Magazine, IEEE, vol. 24, pp. 109–114, May 2007.

[15] R. Benenson, M. Mathias, R. Timofte, and L. Van Gool, “Pedestrian detection at 100 frames per second,” in Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on, pp. 2903–2910, June 2012.

[16] H. Jang, A. Park, and K. Jung, “Neural network implementation using cuda and openmp,” in Digital Image Computing: Techniques and Applications (DICTA), 2008, pp. 155–161, Dec 2008.

[17] A. Guezhva, S. Dolenko, and I. Persiantsev, Artificial Neural Networks – ICANN 2009: 19th International Conference, Limassol, Cyprus, September 14-17, 2009, Proceedings, Part I, ch. Multifacet Acceleration of Neural Network Computations Using GPU, pp. 373–380. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009.

[18] S. Miyaoka and M. Funabashi, “Optimal control of water distribution systems by network flow theory,” Automatic Control, IEEE Transactions on, vol. 29, pp. 303–311, Apr 1984.

[19] Y. Wang, C. Ocampo-Martinez, V. Puig, and J. Quevedo, “Gaussian-process-based demand forecasting for predictive control of drinking water networks,” in Proceedings of the 9th International Conference on Critical Information Infrastructures Security (CRITIS), (Limassol (Cyprus)), 2014.

[20] S. Cong, S. Puig, and G. Cembrano, “Combining CSP and MPC for the operational control of water networks: Application to the Richmond case study,” in 19th IFAC World Congress, (Cape Town), pp. 6246–6251, 2014.

[21] A. Sampathirao, P. Sopasakis, A. Bemporad, and P. Patrinos, “Distributed solution of stochastic optimal control problems on GPUs,” in 54 IEEE Conf. Decision and Control, (Osaka, Japan), Dec 2015.

[22] N. Parikh and S. Boyd, “Proximal algorithms,” Found. Trends Optim., vol. 1, pp. 127–239, jan 2014.
[37] P. Combettes and J.-C. Pesquet, “Proximal splitting methods in signal processing.” 2010. arXiv:0912.3522v4.

[38] R. Rockafellar, Convex analysis. Princeton university press, 1972.

[39] R. Rockafellar and J. Wets, Variational analysis. Berlin: Springer-Verlag, 3rd ed., 2009.

[40] Y. Nesterov, “A method of solving a convex programming problem with convergence rate $O(1/k^2)$,” Soviet Mathematics Doklady, vol. 72, no. 2, pp. 372–376, 1983.

[41] P. Patrinos and A. Bemporad, “An accelerated dual gradient-projection algorithm for embedded linear model predictive control,” Automatic Control, IEEE Transactions on, vol. 59, pp. 18–33, Jan 2014.

[42] P. Giselsson and S. Boyd, “Metric selection in fast dual forward–backward splitting,” Automatica, vol. 62, pp. 1–10, Dec 2015.

[43] A. Bradley, Algorithms for equilibration of matrices and their application to limited-memory quasi-Newton methods. PhD thesis, Stanford University, 2010.

[44] D. P. Bertsekas, Nonlinear Programming, vol. 1. Belmont, MA: Athena Scientific, 1999.

[45] H. Aigre, J. Baptista, E. Cabrera Jr, and F. Cubillo, Performance indicators for water supply services. Manuals of best practice series. London: IWA Publishing, 2006.

[46] D. P. Bertsekas, Dynamic Programming and Optimal Control. Athena Scientific, 2nd ed., 2000.