Time Dependent Contamination Source Determination for Municipal Water Networks using Large Scale Optimization

C. D. Laird, L. T. Biegler *
Department of Chemical Engineering, Carnegie Mellon University, Pittsburgh, PA, 15213, USA
B. van Bloemen Waanders, R. A. Bartlett
Sandia National Laboratories, Albuquerque, NM, 87109, USA

December 4, 2003

Abstract

Concern about malicious contamination of municipal drinking water networks requires us to consider additional protection measures over physical security alone. In the event of an accidental contamination or malicious attack, knowledge of the time and location of the contamination source can help infrastructure personnel stop the contamination and propose control strategies for containment or flushing. In this work, we develop a large scale, nonlinear program for identifying contamination sources given concentration measurements from a limited sensor grid installed in the water network. In a previous work [16], we demonstrated the potential for optimization techniques on the contamination source determination problem, but showed that the direct sequential approach was insufficient to solve the fully time dependent problem. In this current work, we use a direct simultaneous approach, converging the network model and optimization problems simultaneously. To obtain reasonable problem sizes, we present an origin tracking algorithm that reformulates the pipe expressions (partial differential equations in time and space) into a system of algebraic expressions in time alone. This algorithm provides a straightforward mathematical representation of the pipe boundary concentrations and is efficient for large networks with many source and output nodes. After reformulating the optimization constraints, we can solve the resulting nonlinear programming problem with large scale optimization tools. The fully time dependent solution gives complete injection profiles, identifying both the time and location of potential sources of contamination. We demonstrate the effectiveness of this formulation on a model for a real municipal water network.

*Corresponding Author, email: lb01@andrew.cmu.edu

1 Introduction

The threat of accidental drinking water contamination is not new. More recently, however, concern over intentional contamination of municipal water networks has required us to consider novel protection measures. Drinking water networks are especially vulnerable to biological and chemical attack due to the land area encompassed by the network and the large number of access points. Any water outlet, such as a fire hydrant or even a household water faucet, can be an access point for backflow contamination into the network. Physical security can only provide a limited amount of protection. As an alternative to physical security alone, sensors could be installed in the network to detect contamination and initiate a means of protection from within the network itself.

It is generally assumed that contaminant sensors would be costly to purchase, install, and maintain, making it desirable to consider as few sensors as possible. Determining the optimal placement of a limited number of sensors is an area of current research [2, 10]. Unfortunately, a limited sensor grid alone can only provide a coarse indication of the source of the contamination. An accurate measure of the contamination source would help utilities and security personnel stop the contamination as soon as possible and formulate an effective control strategy. Our proposal is to use the time varying concentration data from an installed sensor grid and formulate an inverse problem that more accurately estimates the time and location of the contamination sources.

Inverse problems like this one, are fundamentally different in nature from standard simulation problems. Traditional water quality simulations assume injections of secondary species are known. They solve forward in time to find the propagation of these species through the network. Simulation of the output state of a model based on known inputs is referred to as the forward problem. By contrast, inverse techniques attempt to find the unknown inputs that give rise to a partially known output state. In the contaminant source determination problem, injections are unknown. Instead, we wish to calculate them based on concentration measurements from the network. Inverse problems, like this one, are inherently ill-conditioned and pose unique difficulties not present in the forward problem [1, 17, 6, 11].

Nonlinear programming provides a framework for the inverse problem. In Section 1.1 we formulate an infinite dimensional least squares minimization, subject to the constraints of the water network model. We make no assumption about the time or location of the contamination and introduce fully time dependent injection terms at every network node. The water network model contains algebraic and ordinary differential equations that are a function of time alone and partial differential equations for network pipes that are a function of both time and displacement. A naive discretization of this system produces a large scale, nonlinear math programming problem that is intractable by current solution techniques. To overcome this difficulty, we consider formulations that remove the need to discretize along

the length of the pipes.

We present an origin tracking algorithm in Section 2 that reformulates the partial differential pipe expressions into algebraic constraints with time delays. This algorithm characterizes the time delay between concentrations at the pipe boundaries, removing the need to discretize in space. In Section 3, we present the time discretized inverse formulation as a quadratic programming problem that is tractable for current large scale tools. For reasonable scenarios, this quadratic program has nonunique solutions and regularization is necessary to force a unique solution.

In Section 4, we demonstrate the behavior of this formulation on some small numerical test problems and also demonstrate the effectiveness of this approach on a real municipal water network model with a variety of contamination locations and sensor configurations. Section 5 then concludes the paper and outlines areas for future work.

1.1 Problem Formulation

Water distribution systems are often described by a network of links and nodes, where links represent pipes, pumps, or valves, and nodes represent sources, tanks, or junctions. Considering every pipe in a large municipal water network is often unreasonable and most models collapse regions of the network, modeling individual industrial or residential areas as single network nodes. When simulating these systems, it is usually assumed that the low concentrations of secondary species in the network do not dramatically affect fluid flow properties. Therefore, the flow and concentration calculations can be decoupled into two separate models, the hydraulic model and the water quality model. Generally, algorithms assume the consumption demands are constant over the hydraulic timestep, and solution of the hydraulic model gives piecewise constant profiles for the network flow rates. The water quality solution can then be determined by a variety of existing techniques [7, 14, 13, 19, 15]. In this work, we are concerned only with the water quality analysis and assume that the flow profiles are known, from flow measurements, hydraulic simulations, historical data, or some combination.

With known flow rates and velocities as inputs, we develop the water quality model for the network, using $\mathcal{P}$, $\mathcal{J}$ and $\mathcal{S}$ to refer to the complete sets of all pipes, junctions, and storage tanks respectively. We will use $\bar{c}_i(x,t), i \in \mathcal{P}$ to represent the concentration in the pipes and $\hat{c}_k(t), k \in \mathcal{N}$ to represent the concentration at the nodes, where $\mathcal{N} = \mathcal{J} \cup \mathcal{S}$ is the complete set of all nodes, including junctions and storage tanks. Here, $t \in [0..t_f]$ is time, and $x \geq 0$ is the displacement along a pipe. In developing the model, we will need to refer to connections and concentrations at pipe boundaries. Choosing to define these boundaries based on flow direction, Figure 1 shows the notation used for a flow direction from left to right. The designation $x{=}\mathcal{I}_i(t)$ refers to the boundary where fluid is entering

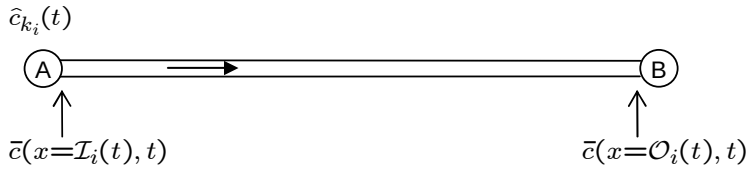

Figure 1: Link Boundary Designation. $\mathcal{I}_i(t)$ indicates the inlet of the link, based on the current flow direction, while $\mathcal{O}_i(t)$ indicates the outlet of the link. The index $k_i(t)$ always refers to the node connected at the inlet.

the link, and $k_i(t)$ is the index of the node connected at this boundary, in this case, node A. The designation $x{=}\mathcal{O}_i(t)$, refers to the boundary where fluid is leaving the link. Note that these designations are time dependent and change with the flow direction. In the case of stagnant flow, the boundary designations are arbitrary with the only restriction that they refer to opposing ends of the pipe. Pumps and valves are modeled as zero length pipes, and reservoirs are modeled as junctions with known external sources. We assume there is no decay reaction for the contaminant, although first order decay can easily be included in the formulation.

While the outer regions of the network may have low flow rates and stochastic demand patterns, plug flow is assumed valid along the main distribution lines represented in the collapsed model. The contaminant profiles in pipe i are then governed by the partial differential equation for advective transport,

$$\frac{\partial \bar{c}_i(x,t)}{\partial t} + u_i(t)\frac{\partial \bar{c}_i(x,t)}{\partial x} = 0, \tag{1}$$

with the additional inlet and initial conditions,

$$\bar{c}_i(x{=}\mathcal{I}_i(t), t) = \hat{c}_{k_i(t)}(t), \tag{2}$$

$$\bar{c}_i(x, t{=}0) = 0, \tag{3}$$

where $u_i(t)$ is the known fluid velocity.

Assuming complete mixing, we write the mass balance for junction k as,

$$\hat{c}_k(t) = \frac{\left(\sum_{i\in\Gamma_k(t)} Q_i(t)\ \bar{c}_i(x{=}\mathcal{O}_i(t), t)\right) + m_k(t)}{\left(\sum_{i\in\Gamma_k(t)} Q_i(t)\right) + Q_k^{ext}(t) + Q_k^{inj}(t)}, \tag{4}$$

where $\Gamma_k(t)$ is the set of all pipes flowing into node k at time t. $Q_i(t)$ is the known volumetric flow rate from pipe i, and $Q_k^{ext}(t)$ is the volumetric flow rate for known external sources (e.g. reservoir flow). In this application, all known sources are assumed to have zero contaminant concentration. Instead, we model the unknown contaminations as time dependent mass booster injections [12], $m_k(t)$, to every node and assume negligible flow rates. In practice, the volumetric flow rates of these injections, $Q_k^{inj}(t)$, will not be known and we set them to a small quantity relative to other network flow rates.

We write the differential mass balance for storage tank k as,

$$\begin{aligned} V_k(t)\frac{d\hat{c}_k(t)}{dt} = &\left(\sum_{i\in\Gamma_k(t)} Q_i(t)\ \bar{c}_i(x{=}\mathcal{O}_i(t),t)\right) + m_k(t) \\ &- \left[\left(\sum_{i\in\Gamma_k(t)} Q_i(t)\right) + Q_k^{ext}(t) + Q_k^{inj}(t)\right]\hat{c}_k(t), \end{aligned} \tag{5}$$

with the initial condition,

$$\hat{c}_k(t{=}0) = 0. \tag{6}$$

Here, we assume there is no contaminant present in the system initially, but we could also formulate (6) to include the initial tank concentration as an unknown.

Since we have assumed that the flow pattern is independent of the composition, the volumetric flow rates, $Q_i(t)$ and Q_k^{ext}, and the tank volumes, $V_k(t)$, are predetermined. As a result, $\bar{c}_i$, $\hat{c}_k$, and m_k are the only variables and equations (1-6) form a linear time varying system. We then write a weighted least squares formulation of the inverse problem using p_i to represent the equations (1-3) for the links, j_k to represent equation (4) for the junctions, and s_k to represent equations (5) and (6) for the storage tanks as,

$$\min_{m(t),\bar{c}(x,t),\hat{c}(t)} \Psi = \sum_{r\in\Theta_s}\sum_{k\in\mathcal{N}_s} \frac{1}{2}\int_0^{t_f} w_k(t)\left(\hat{c}_k(t) - \hat{c}_k^{\star}(t)\right)^2\ \delta(t-t_r)\ dt$$

$$\begin{aligned} \text{s.t.} \qquad p_i(\ \bar{c}_i(x,t), \hat{c}(t)\ ) &= 0 \qquad && \forall i \in \mathcal{P}, \\ j_k(\ \bar{c}(x,t), \hat{c}_k(t), m_k(t)\ ) &= 0 && \forall k \in \mathcal{J}, \\ s_k(\ \bar{c}(x,t), \hat{c}_k(t), m_k(t)\ ) &= 0 && \forall k \in \mathcal{S}, \\ m_k(t) &\geq 0 && \forall k \in \mathcal{N}. \end{aligned} \tag{7}$$

The subset, $\mathcal{N}_s \subseteq \mathcal{N}$, is the set of nodes with installed sensors, and the measured concentrations, $\hat{c}_k^{\star}(t)$, will not be known continuously in time, but rather at discrete sampling points in Θ_s, described using the standard delta function, δ. The function, $w_k(t)$, is a time

dependent weight for the concentration errors. We use a flow based weighting function for each sensor node, shifting the error measure from a concentration basis to a mass basis.

One could view the traditional forward problem as that of setting the mass injections, $m_k(t)$, and solving (using (1-6)) for the pipe and node concentrations, $\bar{c}_i(t)$, and $\hat{c}_k(t)$ respectively. In the optimization problem (7), the solution vectors are the complete time profiles of the unknown variables $\bar{c}_i(x,t)$, $\hat{c}_k(t)$, and $m_k(t)$, for all $i \in \mathcal{P}$, $k \in \mathcal{N}$. The values of the injection terms, $m_k(t)$, are the profiles of interest, where significantly positive values for a particular node, k, indicate a potential contaminant source location. Problem (7) is an infinite dimensional optimization problem subject to algebraic, ordinary differential, and partial differential constraints.

1.2 Solution Techniques

Solution approaches for differential programming problems can be separated into two general classes, direct and indirect [3]. *Indirect* methods use a variational approach to write the first order optimality conditions as a boundary value problem. *Direct* methods, on the other hand, apply optimization tools directly to a discretized form of the differential model. Categories of direct methods differ in their treatment of model constraints and algorithms exist to solve the model constraints sequentially, simultaneously, or by some blend of the two.

Direct Sequential methods discretize the independent variables (control variables or inversion parameters) only. Given an initial guess for the profiles of these variables, standard solution techniques for the forward problem are used to evaluate the model at each iteration of the optimization and calculate values for the objective function. Derivative information is required with respect to the independent variables at each of the discretized points and can be calculated by various techniques, including the sensitivity equations, adjoint equations, or finite difference. The optimization problem itself is in the space of the independent variables only and is small by comparison. However, calculation of derivative information can be computationally expensive. Also, the forward problem is often solved using an iterative technique and noise in the solution of the forward problem can cause convergence difficulties in the optimization.

Direct Simultaneous methods fully discretize all the unknown variables in the problem and solve the resulting system as a large scale optimization problem with algebraic constraints. The forward problem is converged only once, along with the optimality conditions. Accurate analytical derivatives are often straightforward and efficient to calculate, and significant computational gains over the standard sequential approach are possible using this more intrusive technique. A review of direct and indirect techniques as applied to optimization of differentially constrained problems can be found in Cervantes and Biegler [4].

Both the simultaneous and sequential techniques require some form of the water qual-

ity model. Traditional water quality simulation methods can be classified as *Eulerian* or *Lagrangian* [13, 15, 19]. Eulerian methods discretize the network model in both time and space, tracking the concentration at fixed points or volumes within the pipe. Lagrangian methods discretize in time alone and track the concentrations of discrete volume elements as they move through the network.

In a previous work [16], we applied a direct sequential algorithm to solve problem (7). The model constraints representing the forward problem, (1-6) , were solved at each iteration using the existing water network simulation package, EPANET [12]. We then used a standard sequential quadratic programming tool to perform the optimization. Although the Lagrangian formulation used by EPANET provided efficient solution of the forward problem, it was not clear how to efficiently calculate derivatives, so finite differences were used. The unreasonable computational cost of calculating finite differences across the model prevented complete discretization of the time dependent injection terms, $m_k(t)$. Instead, the time discretized profiles were reduced to single scalar parameters, allowing solutions for constant injection or initial condition contaminations only. Nevertheless, this approach demonstrated the potential for optimization techniques. In this current work, we use a direct simultaneous approach to overcome the difficulties encountered with the sequential method and solve the fully time dependent problem.

The simultaneous approach requires an explicit mathematical representation of the discretized water quality model. While the Lagrangian technique provides efficient simulation, it is not obvious how to formulate this model in a simultaneous setting. On the other hand, Eulerian techniques provide a straightforward representation, but the complete model, discretized in both time and space, produces a nonlinear program that is intractable for current tools. Let n_t and n_x be the number of discretizations in time and space respectively. The fully discretized model has a constraint count on the order of $|\mathcal{P}| \cdot n_t \cdot {n_x}^1 + |\mathcal{J}| \cdot n_t + |\mathcal{S}| \cdot n_t$, and an approximate variable count of $|\mathcal{P}| \cdot n_t \cdot n_x + 2(|\mathcal{J}| \cdot n_t + |\mathcal{S}| \cdot n_t)$. The problem size grows too large because of the need to discretize the pipe concentrations in both time and space. To overcome this, we recognize that the injection terms, node concentrations, and objective function are all dependent on time alone. As well, the node constraints only require pipe concentrations at the pipe boundaries. The only constraint that directly uses the spatial discretization is the pipe equation (1). Since the solution to this equation at either boundary can be represented by an algebraic expression with time delay, we are guided to seek a formulation of the network model that does not require a discretization in space. In particular, we desire a model reformulation with the following properties:

- We want to include the water quality model within an optimization problem that may

[1] In practice, the number of discretizations in x_i would be different for each pipe i, but if we consider n_x to be some averaged value, the argument remains the same.

have additional constraints and requirements. As such, the formulation should have a straightforward mathematical representation.

- The discretized model must be a reasonable size for current large scale nonlinear programming tools. This likely requires a reformulation of the pipe constraints that removes the need to discretize in space.

- Applications often require concentration variables for many nodes at many points in time. In particular, our optimization formulation requires concentration variables at all sensor nodes and all sample times. Any model preprocessing should be efficient for a large number of both source and output nodes.

Our proposed approach is related to the recently developed particle backtracking algorithm. This algorithm, proposed by Zierolf et al. [19], and extended by Shang et al. [15], reduces the water quality model down to its essential elements. The algorithm calculates impact coefficients which describe the concentrations of selected output nodes as a function of network sources and tank concentrations. The impact coefficients are calculated by tracking a particle in reverse time, from the output node, back through the network, to the source nodes. The time delays associated with paths through the network are described exactly and numerical error is only introduced by discretized tank equations. This algorithm could be used to evaluate the time delay expressions for every sensor node at all sample times, considering a mass secondary input to each node. This technique satisfies our first two criteria, providing a reduced model with a straightforward mathematical representation. Unfortunately, difficulties include the cost of evaluating time delay expressions for a large number of nodes at many points in time. Instead we propose a technique that considers the time delays of each pipe individually and scales efficiently to large networks when considering all nodes in the network model.

2 Reformulating the Pipe Constraints

The goal of the algorithm described here is to reformulate the partial differential pipe equations (1-3), into a set of algebraic constraints. We remove the need to discretize along the length of the pipes by considering concentrations at the pipe boundaries only. We then characterize the time delays associated with each pipe in the network. Of course, these time delays are themselves time dependent and will be affected by changing flow conditions. Choosing any appropriate time discretization, $l \in \Theta$, we can discretize the pipe boundary

concentrations, node concentrations, and injection terms,

$$\bar{c} = \begin{bmatrix} \bar{c}_i(x{=}\mathcal{I}_i(t_l), t_l) \\ \vdots \\ \bar{c}_i(x{=}\mathcal{O}_i(t_l), t_l) \\ \vdots \end{bmatrix}, \quad \hat{c} = \begin{bmatrix} \hat{c}_k(t_l) \\ \vdots \end{bmatrix}, \quad m = \begin{bmatrix} m_k(t_l) \\ \vdots \end{bmatrix}, \qquad \begin{matrix} \forall i \in \mathcal{P}, \\ \forall k \in \mathcal{N}, \\ \forall l \in \Theta. \end{matrix} \tag{8}$$

With this discretization, the algebraic expressions for the concentrations at the pipe boundaries can be represented as a linear system,

$$\bar{P}\bar{c} + \hat{P}\hat{c} = 0, \tag{9}$$

where $\bar{P}$ and $\hat{P}$ will be matrices of zeros and ones.

To demonstrate this, consider a network where the volumetric flow rates, $Q_k(t)$, and fluid velocities, $u_i(t)$, are known and can reverse and stagnate. The concentration at any point in pipe i originated from one of three possibilities: the initial concentration in the pipe at time zero, $\bar{c}(x, t{=}0)$, or the concentration of either connected node at the current or previous timestep. Since we assume there is no initial concentration in the pipe, the right hand side of equation (9) is zero. The matrices $\bar{P}$ and $\hat{P}$ link discretized pipe boundary concentrations with the appropriate discretized node concentrations, characterizing the time delays in the network pipes. An expression for the concentration at one boundary will always be straightforward to define as $\bar{c}(x = \mathcal{I}(t_l), t_l) = \hat{c}_{k_i(t_l)}$. For example, in the case of flow from left to right, the concentration at the left side of the pipe will always be equivalent to that of the connected node at the current timestep. With general flow patterns, the relationship for the alternate end is more difficult to characterize.

To find this relationship, we propose an origin tracking algorithm based on the water quality method presented by Liou and Kroon [7]. Our algorithm exploits the efficiency of the Lagrangian technique, but while traditional Lagrangian methods track the actual concentration value of fluid elements as they move through the network, we instead track the origin of each fluid element in the pipe.

To formalize this idea, Figure 2 shows pipe i with flow conditions from left to right. For each timestep, $l \in \Theta$, we wish to find expressions for the concentration at each pipe boundary. At timestep $l{=}1$, a fluid element is created at the left boundary and we can record its originating node as "A" and its originating timestep as "1". The concentration at this end of the pipe is equivalent to the immediate concentration from node A, that is $\bar{c}(x{=}\mathcal{I}_i(t_1), t_1) = \hat{c}_A(t_1)$. The concentration at the right boundary of the pipe is equal to the initial loading in the pipe, $\bar{c}(x{=}\mathcal{O}_i(t_1), t_1) = 0$. As time progresses, the element advances through the pipe. Assuming the element is pushed from the pipe at timestep $l{=}5$, the

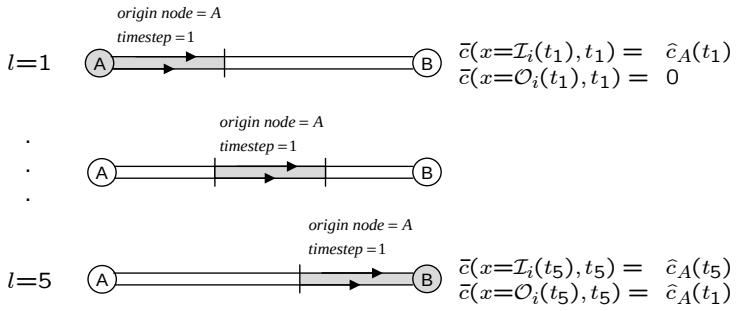

Figure 2: General Pipe Diagram. This figure illustrates the flow of a single volume element through a pipe. Tracking this element allows us to determine the approximate time delays at each point in time.

pipe concentration at the left boundary is still equivalent to the current value from node A, $\bar{c}(x{=}\mathcal{I}_i(t_5), t_5) = \hat{c}_A(t_5)$. The concentration at the right boundary, however, is now the same as the value from node A at timestep 1, $\bar{c}(x{=}\mathcal{O}_i(t_1), t_1) = \hat{c}_A(t_1)$. With this approach we can find time delay expressions defining the boundary concentrations for each pipe at every timestep. The result of this characterization is a set of algebraic expressions, which are linear in $\bar{c}$ and $\hat{c}$.

2.1 Origin Tracking Algorithm

The origin tracking algorithm is used to generate the linear system (9) that replaces equations (1-3) in optimization problem (7). In developing this algorithm, the Lagrangian method of Liou and Kroon is simplified in two ways. First, we assume that the set of discrete points in time are known and fixed. This allows us the flexibility to work with any discretization scheme selected for the differential tank equations. Second, we do the analysis on a pipe by pipe basis, not on the network as a whole. The processing and memory requirements are then linear with the number of pipes and efficient for large networks. Although these simplifications can introduce estimation errors in the time delays associated with paths through the network, they provide efficient scaling and favorable sparsity in the model.

A description of the origin tracking algorithm is shown below, where i is the current pipe, and l is the current timestep. In each tracked volume element, we store the originating node, the originating timestep, and the current position of each of the element boundaries within the pipe.

Algorithm 1. Origin Tracking Algorithm

Step 0. Initialize Overall Algorithm

- let $i = 0$, the first pipe in the network

Step 1. Initialize Pipe Iterations

- clear the fluid elements
- let $l = 0$
- set $\bar{c}_i(x{=}\mathcal{I}_i(t_l), t{=}t_l) = \bar{c}_i(x{=}\mathcal{O}_i(t_l), t{=}t_l) = 0$

Step 2. Advance Elements

- $\Delta x{=}u_i(t_l) \cdot (t_l - t_{l-1})$
- advance all currently tracked elements by Δx

Step 3. Add New Elements

- create a new element at the top or bottom of the list depending on flow direction
  - record the originating node as $k_i(t_l)$, the current "inlet" node
  - record the originating timestep as l

Step 4. Write Time Delay Expressions

- if stagnant flow,
  - set the concentrations equal to the expressions from the last timestep
- otherwise,
  - set $\bar{c}(x{=}\mathcal{I}_i(t_l), t{=}t_l) = \hat{c}_{k_i(t_l)}(t_l)$
  - if there is no element bracketing a pipe boundary
    - set $\bar{c}(x{=}\mathcal{O}_i(t_l), t{=}t_l) = 0$
  - otherwise,
    - read the data from the element bracketing the pipe boundary, store the originating node as "n" and the originating timestep as τ
      - set $\bar{c}(x{=}\mathcal{O}_i(t_l), t{=}t_l) = \hat{c}_n(t_\tau)$

Step 5. Crop Elements

- remove any elements that have advanced outside the pipe boundary

· crop the length of any overhanging element

Step 6. Continue with the next timestep

· $l = l + 1$

· if $t_l \leq t_f$, goto Step 2.

Step 7. Continue with the next pipe

· $i = i + 1$

· if $i \in \mathcal{P}$, goto Step 1.

To illustrate this algorithm on a simple example, consider a 1000 foot long pipe, i, where the velocity is 30 feet per minute from left to right for the first 40 minutes, -30 feet per minute (right to left) for the next 40 minutes, and then stagnant. Using a timestep of 10 minutes, the progression of the algorithm is illustrated in Figure 3. For the initial timestep, $l = 0$, the boundary concentrations are both equal to the initial value. At l=1 (t=10 minutes), the initial fluid in the pipe has been displaced by 300 feet. With flow from left to right, the concentration at the left pipe boundary is equivalent to the immediate concentration from node A (i.e. $\mathcal{I}_i(t_1) = x_i^A$, $\bar{c}_i(x{=}x_i^A, t{=}t_1) = \hat{c}_A(t_1)$). The concentration at the right pipe boundary is still the initial value (i.e. $\mathcal{O}_i(t_1) = x_i^B$, $\bar{c}_i(x{=}x_i^B, t{=}t_1) = 0$). As we proceed three more timesteps, we completely push the initial fluid from the pipe. For $l = 4$, the concentration at the left boundary is still the same as the node A, but the concentration at the right boundary is now equivalent to that of node A from three timesteps previous, $\bar{c}_i(x{=}x_i^B, t{=}t_4) = \hat{c}_A(t_1)$

Reversing the flow, we begin pushing the volume elements back towards the inlet. At $l = 5$, the concentration at the left boundary is the immediate value from node B, whereas the fluid element leaving the left boundary of the pipe originated from node A at timestep 3. Advancing three more timesteps, we have pushed out all fluid originating from node A. The concentration at the right boundary is still the immediate value from node B, and the concentration at the left boundary is now a delayed value from node B. Finally, in timestep $l = 9$, the flow is stagnant and the concentration expressions are duplicated from the previous timestep.

Performing this algorithm for each pipe in the network, we can completely describe the linear system (9). The computational cost of the algorithm is linear in the number of pipes yet describes relationships for all pipe boundaries at all discretized points in time. Note that this algorithm requires flow data and network structure only and is performed with no prior knowledge of the source terms, m. This resulting linear system provides a straightforward mathematical representation that characterizes the time delays and can easily be included in the discretized optimization problem.

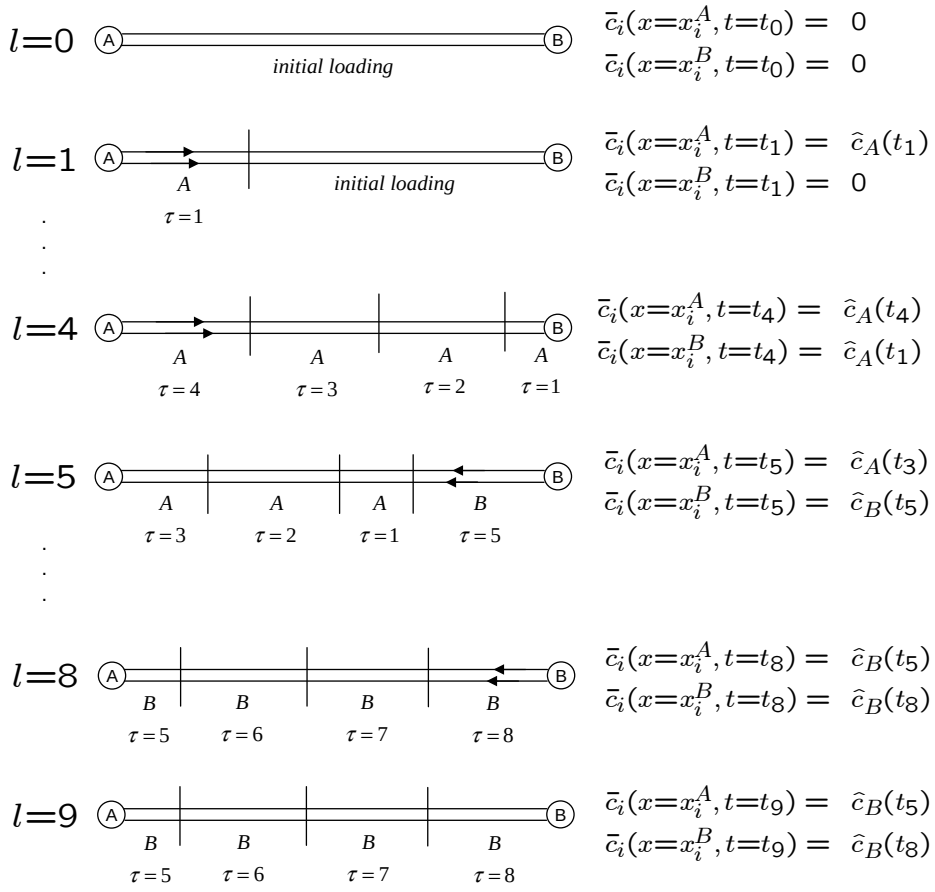

Figure 3: Algorithm Example. This figure illustrates the origin tracking algorithm on a brief example with varying flow direction.

3 Discretized Nonlinear Program

Using the reformulation of the pipe constraints illustrated in Section 2, we can discretize problem (7) in time alone, producing a reasonably sized nonlinear program. Although the origin tracking algorithm allows the use of any discretization scheme, accuracy requirements on the pipe time delays tend to govern the stepsize and we gain little benefit from a higher order method. In this paper we use a simple backward Euler technique. However, we have implemented more advanced techniques, like collocation on finite elements, with similar results.

Choosing equally spaced intervals, $h{=}t_l - t_{l-1}$, we write the discrete form of equation (5) as,

$$\left[\frac{V(t_{l+1})}{h} + \left(\sum_{i\in\Gamma_k(t_{l+1})} Q_i(t_{l+1})\right) + Q_k^{ext}(t_{l+1}) + Q_k^{inj}(t_{l+1})\right]\hat{c}_k(t_{l+1}) - \frac{V(t_{l+1})}{h}\hat{c}_k(t_l) - \left(\sum_{i\in\Gamma_k(t_{l+1})} Q_i(t_{l+1})\bar{c}_i(x{=}\mathcal{O}_i, t_{l+1})\right) - m_k(t_{l+1}) = 0$$

for all $k \in \mathcal{S}, l \in \Theta$, where the initial condition is given by,

$$\hat{c}_k(t{=}0) = 0. \tag{10}$$

The discretized junction equations from (4) are,

$$\hat{c}_k(t_l) - \frac{\left(\sum_{i\in\Gamma_k(t_l)} Q_i(t_l)\bar{c}_i(x{=}\mathcal{O}_i, t_l)\right) + m_k(t_l)}{\left(\sum_{i\in\Gamma_k(t_l)} Q_i(t_l)\right) + Q_k^{ext}(t_l) + Q_k^{inj}(t_l)} = 0, \tag{11}$$

for all $k \in \mathcal{J}, l \in \Theta;\ l \neq 0$. Difficulty arises when all flows to a junction are stagnant. In this case, we replace equation (11) with $\hat{c}_k(t_l) = \hat{c}_k(t_{l-1})$ or $\hat{c}_k(t_0) = 0$, if required. The set of discretized equations (9-11), for the pipes, junctions, and tanks can be written as the linear system,

$$\begin{bmatrix} \bar{P} & \hat{P} & 0 \\ \bar{N} & \hat{N} & M \end{bmatrix} \begin{bmatrix} \bar{c} \\ \hat{c} \\ m \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \tag{12}$$

where $\bar{c}$, $\hat{c}$, and m are the complete vectors of discretized concentrations described by (8). The junction and tank equations are grouped together and $[\ \bar{N}\ \hat{N}\ M\ ]$ is the Jacobian of these discretized node equations with respect to the unknown variables. Discretizing the objective from (7) we have,

$$\Psi_D(\hat{c}) = \frac{1}{2}\left[\hat{c} - \hat{c}^\star\right]^T W \left[\hat{c} - \hat{c}^\star\right]. \tag{13}$$

Here, W is the diagonal matrix of flow based weights for the concentration error and will only have nonzero entries corresponding to sensor nodes at sample times. These weights are all positive, so W is a positive semidefinite matrix[2]. Minimizing the quadratic objective (13), subject to the linear constraint equations (12), and bounds on the injection terms, $m \geq 0$, we produce a convex quadratic program. As such, any solution of this problem is a global minimum, but the solution, $[\bar{c}, \hat{c}, m]$ is not necessarily a unique minimizer.

To illustrate this, consider a simple two node network with flow from node A to node B and a sensor at node B only. Assume, given current flow rates, that the travel time between the nodes is one hour. If the sensor at node B registers concentration $c^\star$ at time $t^\star$, it is clear that the contamination could have been injected at node B at time $t^\star$. But it could also have been injected at node A at time $t^\star$ minus one hour. In fact, if we allow the possibility of multiple injections, any linear combination of these two injections summing to $c^\star$ is a possible solution, giving us infinitely many solutions to the optimization problem.

Source inversion problems, like this one, are inherently ill-posed [6, 17, 1]. Rojas [11] classifies ill-conditioned problems in two general categories, discrete ill-posed problems and rank deficient problems. Rank deficient problems have a coefficient matrix where small singular values are clearly separated from the large singular values. The contaminant source determination problem exhibits this deficiency because of the nonuniqueness described above. In order to force a unique solution for this problem, we introduce a regularization term in the objective, $\rho\frac{1}{2}m^T R m$, and write the complete nonlinear program for the contaminant source inversion problem as,

$$\begin{aligned} \min_{\hat{c},m}\ & \frac{1}{2}\left[\hat{c} - \hat{c}^\star\right]^T W \left[\hat{c} - \hat{c}^\star\right] + \rho\frac{1}{2}m^T R m \\ \text{s.t.}\ & \bar{P}\bar{c} + \hat{P}\hat{c} = 0, \\ & \bar{N}\bar{c} + \hat{N}\hat{c} + Mm = 0, \\ & m \geq 0, \end{aligned} \tag{14}$$

[2]In the unusual circumstance where there is a sensor at every node and data is sampled at every timestep, W will be nonsingular and positive definite.

where R is a positive definite regularization matrix. The exact form of R will depend on the type of regularization used. As shown below, any positive value for ρ is sufficient to guarantee a unique solution to problem (14).

A necessary and sufficient condition for a unique solution to the quadratic programming problem is positive definiteness of the reduced Hessian [9] and full rank of the active constraint set. For convenience, we can eliminate the pipe concentrations, $\bar{c}$, and write problem (14) as,

$$\begin{aligned} \min_{\hat{c},m} \quad & \frac{1}{2}\left[\hat{c}-\hat{c}^{\star}\right]^{T} W \left[\hat{c}-\hat{c}^{\star}\right] + \rho\frac{1}{2} m^{T} R m \\ \text{s.t.} \quad & N\hat{c} + Mm = 0, \\ & m \geq 0, \end{aligned} \tag{15}$$

where $N = [\hat{N} - \bar{N}\bar{P}^{-1}\hat{P}]$. Considering a particular solution to this problem, we can partition the injection variables so that M_0 refers to the Jacobian of the node equations with respect to the injection variables with zero values (variables at the lower bound), and M_+ refers to the Jacobian of the node equations with respect to the injection variables that have positive values (away from the bound). The combined Jacobian is then,

$$\begin{bmatrix} N & M_0 & M_+ \\ 0 & I & 0 \end{bmatrix} \begin{bmatrix} \hat{c} \\ m_0 \\ m_+ \end{bmatrix}$$

Using reduction of variables [8], we can produce null space matrix, Z, for the Jacobian of the constraints,

$$Z = \begin{bmatrix} D \\ I \end{bmatrix} \;\; ; \;\; D = -\begin{bmatrix} N & M_0 \\ 0 & I \end{bmatrix}^{-1} \begin{bmatrix} M_+ \\ 0 \end{bmatrix} = -\begin{bmatrix} N^{-1}M_+ \\ 0 \end{bmatrix}$$

and write the reduced Hessian of the Lagrangian,

$$\begin{bmatrix} D^T & I \end{bmatrix} \begin{bmatrix} W & 0 & 0 \\ 0 & \rho R_0 & 0 \\ 0 & 0 & \rho R_+ \end{bmatrix} \begin{bmatrix} D \\ I \end{bmatrix} = M_+^T N^{-T} W N^{-1} M_+ + \rho R_+. \tag{16}$$

Since the weighting matrix, W, will be positive semidefinite, the first term in (16), $M_+^T N^{-T} W N^{-1} M_+$, will be, at least, positive semidefinite. Since R (and hence R_+) is positive definite, any small positive value for ρ will ensure that the reduced Hessian is positive definite. Given full rank of $[NM_+]$, in turn, guarantees that the solution to problem (14) will be unique. In practice,

ρ will need to be sufficiently large to overcome any inaccuracies due to machine precision and ill-conditioning.

The regularized formulation presented in (14) is a well-posed quadratic program that can be solved with a variety of large scale optimization tools. In the next section we will illustrate the effectiveness of this formulation for finding both the time and location of injections for a number of contamination scenarios.

4 Numerical Results

We will first illustrate the behavior of problem formulation (14) on a small fictitious network. Following this discussion, we present numerical results for a real municipal network model withdifferent contamination locations and sensor configurations. The procedure used for numerical testing is illustrated in Figure 4. Using the network model and choosing an injection location, we simulate a contamination scenario with the water network simulation package, EPANET [12]. We have developed a tool to formulate the optimization problem. Our formulation tool requires the horizon information (the start and end time to consider for the inversion), the sensor configuration, and the integration timestep. It then reads the network structure from EPANET input, and the network flow rates and concentrations from EPANET output. Using the origin tracking algorithm, the tool formulates problem (14) as an AMPL [5] model. AMPL is a modeling language that provides both first and second order derivative information using automatic differentiation. IPOPT, a nonlinear interior point optimization package [18], is then used to solve the optimization problem and find solutions for both the network concentrations and the injection profiles. To verify the effectiveness of the formulation, we examine the injection profiles, m, from the optimization solution and compare against the actual profiles used in the simulated contamination.

For all examples, we have selected a 5 minute integration timestep and a 5 minute sample interval. Since the sampling interval is equivalent to the integration timestep, we set the regularization matrix $R = I$, approximating $\sum_{k\in\mathcal{N}} \int_0^{t_f} m_k(t)^2 dt$. For a sampling interval that was longer than the timestep, one could regularize with $\sum_{k\in\mathcal{N}} \int_0^{t_f} \left(\frac{\partial m_k(t)}{\partial t}\right)^2 dt$ to impose a smoothness condition on the injection profiles between sampling intervals. The regularization parameter, ρ, is set to $1{\cdot}10^{-4}$. Fortunately, the numerical results are reasonable over a wide range of values for ρ and no specific tuning is required to find solutions for the different injection locations studied.

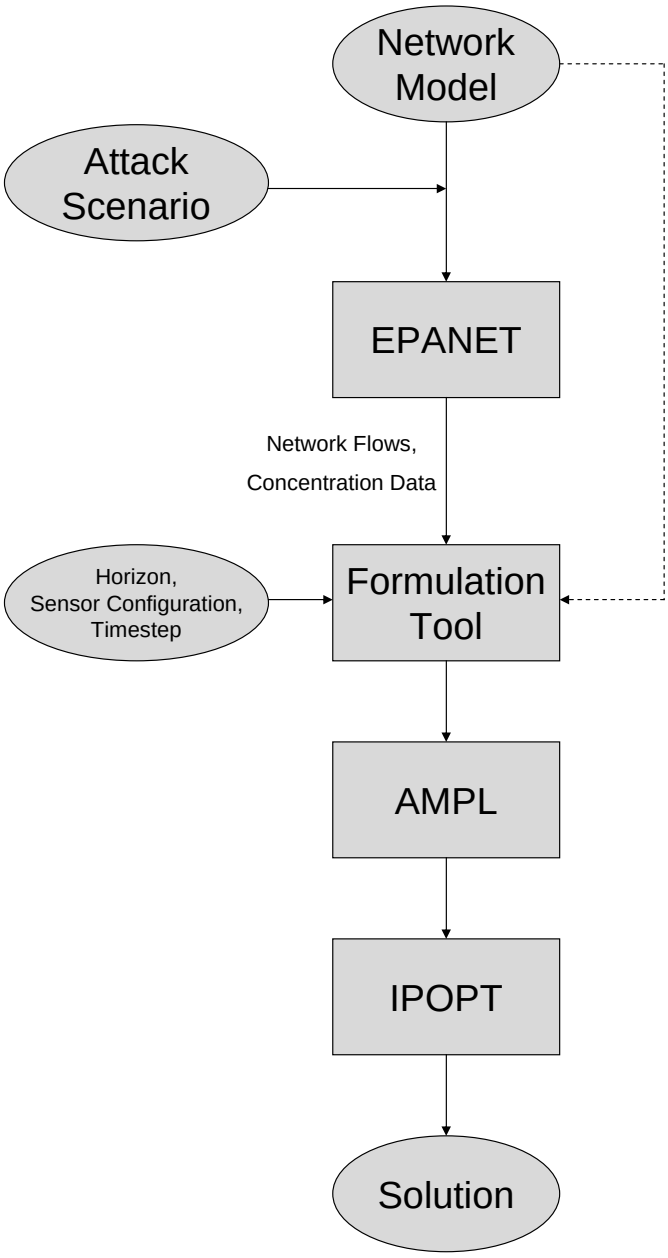

Figure 4: Formulation Test Procedure. EPANET is used to calculate network hydraulics and provide the simulated sensor measurements. We then formulate the optimization problem in AMPL and solve using IPOPT.

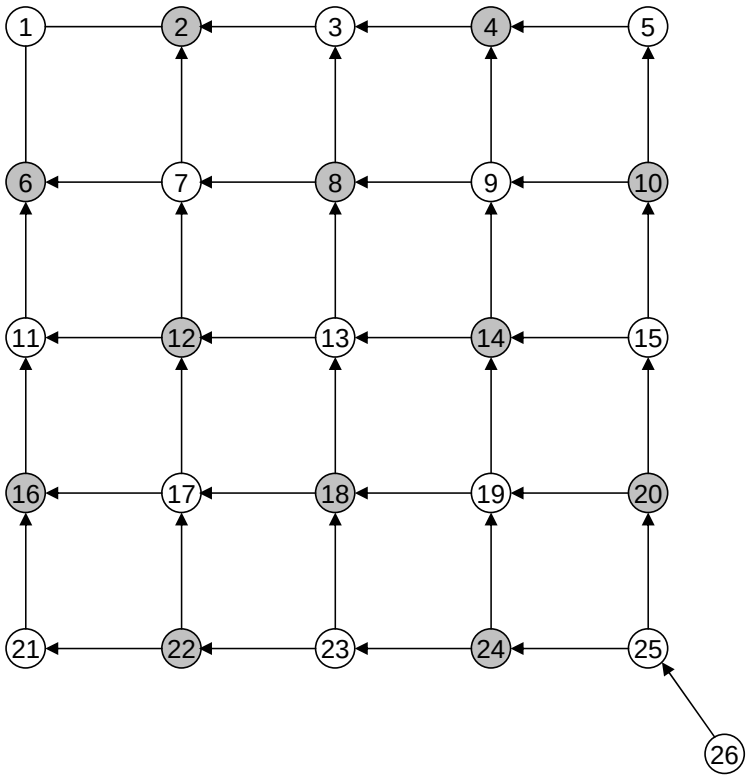

Figure 5: Grid Network Example A small symmetric grid network with sensors installed at every second node, indicated by the shaded circles. Here, we simulate a contamination at node 13.

4.1 Example 1. Symmetric Grid Network

The first example we will consider is a simple grid network, shown in Figure 5, where flows are in the direction indicated, consumption demands exist at the boundary nodes only, and the single known external water source is a reservoir at node 26. The time delays between the nodes range from approximately half an hour to five hours. The shaded nodes indicate installed sensors, and with sensors at every second node, this example is symmetric about the diagonal from node 1 to node 25. We use EPANET to simulate a 30 minute long injection from node 13 at hour 1.

First, we run the optimization at hour 4, considering only the previous 4 hours of sensor and flow data. The solution to the inverse problem is given in Figure 6, where we have solved for the complete injection profiles of all nodes, showing only those with significant values. Even though our regularized problem is guaranteed to have a single unique solution, this

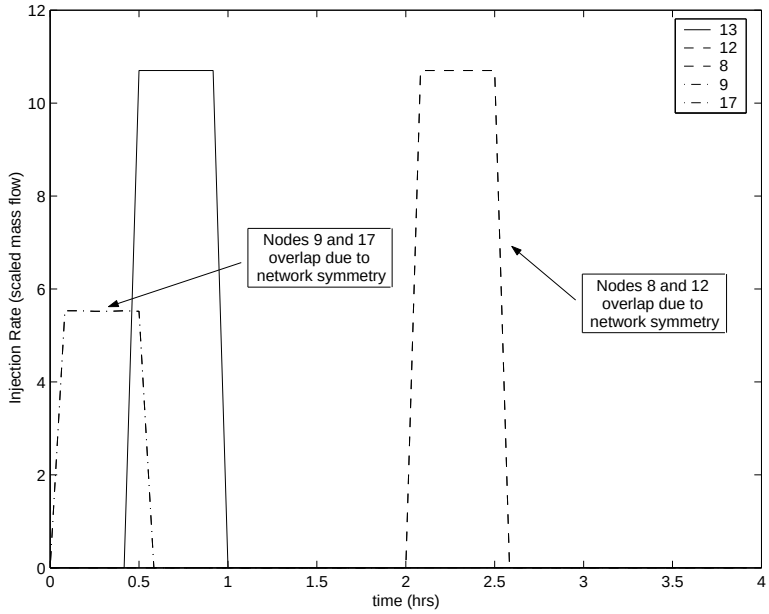

Figure 6: Grid Network Solution 1. Solution for an injection at node 13 using a 4 hour time horizon.

single solution indicates several possible injection scenarios. One possibility is the solution we were hoping for, an injection at node 13 at hour 1. The contamination could also have occured at nodes 9 and 17 at approximately hour 0, or at nodes 8 and 12 at approximately hour 2. Although simultaneous injections may seem unlikely, our formulation makes no restriction on the number of injections or the injection time and solves for these injection scenarios as well.

We can see from the network structure, sensor configuration, and flow direction, that it is impossible to distinguish the actual injection node 13, from injections at nodes 8 and 12. However, we would expect the sensors at nodes 4 and 16 to measure any contaminant that was injected at nodes 9 and 17. In this particular network, the time delays for links 17-16 and 9-4 are just over 4 hours. We ran the optimization at hour 4 and only considered the previous 4 hours of data (only 3 hours past the actual injection). Therefore, the sensor measurements at nodes 4 and 16 do not provide any information about injections at nodes 9 and 17 in the time horizon considered.

Using the same simulated injection, but running the optimization at hour 8, considering the previous 8 hours of simulation data, we find the solution shown in Figure 7. Here, nodes 9 and 17 have now been excluded as possible injection locations. This simple example shows

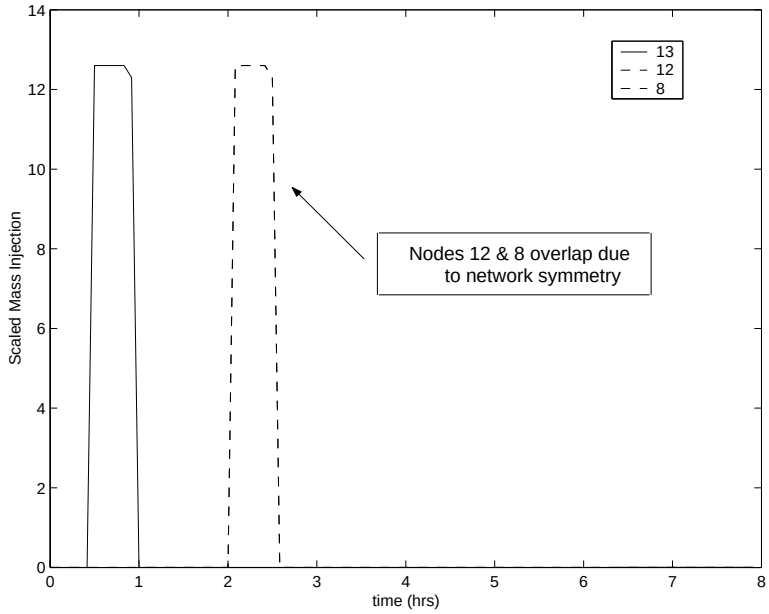

Figure 7: Grid Network Solution 2. Solution for an injection at node 13 with an 8 hour time horizon. Note that nodes 9 and 17 are now excluded as possible injection locations.

the importance of sufficient horizon lengths and concentration measurements of zero.

4.2 Example 2. Real Municipal Network Model

We have tested our formulation on a model for a real municipal water network with 469 nodes and 635 links. We consider four injection locations in the network, A through D, shown in Figure 8. We selected three locations where contaminant spreads readily through a major portion of the network, A,B, and C. Location A is along a major feed line to the network, while locations B and C are interior to the network. We have also selected a node, at location D, with minimal spreading of contaminant through the network. For each of the four locations, we simulate a 16 hour time segment, with a 30 minute injection at hour 8. We would expect an attack or accidental injection to be significantly longer than 30 minutes, but we use this short injection time to illustrate the algorithm performance on a difficult injection scenario. A longer (or continuous) injection would provide even more information for the optimization and, in general, the formulation should be more effective.

We wish to verify the effectiveness of the formulation for a large number of test examples. Visually inspecting each of the profiles, like those shown in Figures 6 and 7 is unreasonable. Instead, we seek a single scalar measure of the effectiveness of the formulation in identifying the actual injection node. After running the optimization we integrate the solution profiles for all the injection terms, $\gamma_k = \int_{t=0}^{t_f} m_k(t)dt$. We then sort the nodes, in descending order of γ_k. This provides a rank for the solution profile of each node in the network. Obviously, we desire the solution profile corresponding to the actual injection node to be ranked first (i.e. the optimization solution estimates the most prominent injection at the node where the actual injection occured). Unfortunately, because of the limited sensor grid and nonuniqueness, this may not be possible.

While measuring the effectiveness of the formulation, we are interested in two key indicators.

1. The Number of Installed Sensors: We would like the algorithm to identify injection locations using as few sensors as possible. The placement of installed sensors will likely be critical in reducing this number.

2. Identification Time: Note, this does not refer to the time for an optimization to execute, but rather to the time required for the contaminant to reach installed sensors, plus the additional time required to accrue enough information for the optimization to be effective. This constitutes the total ellapsed time, following a contamination, before the system can make a reasonable estimate of the injection location. We give an indicator of this time by

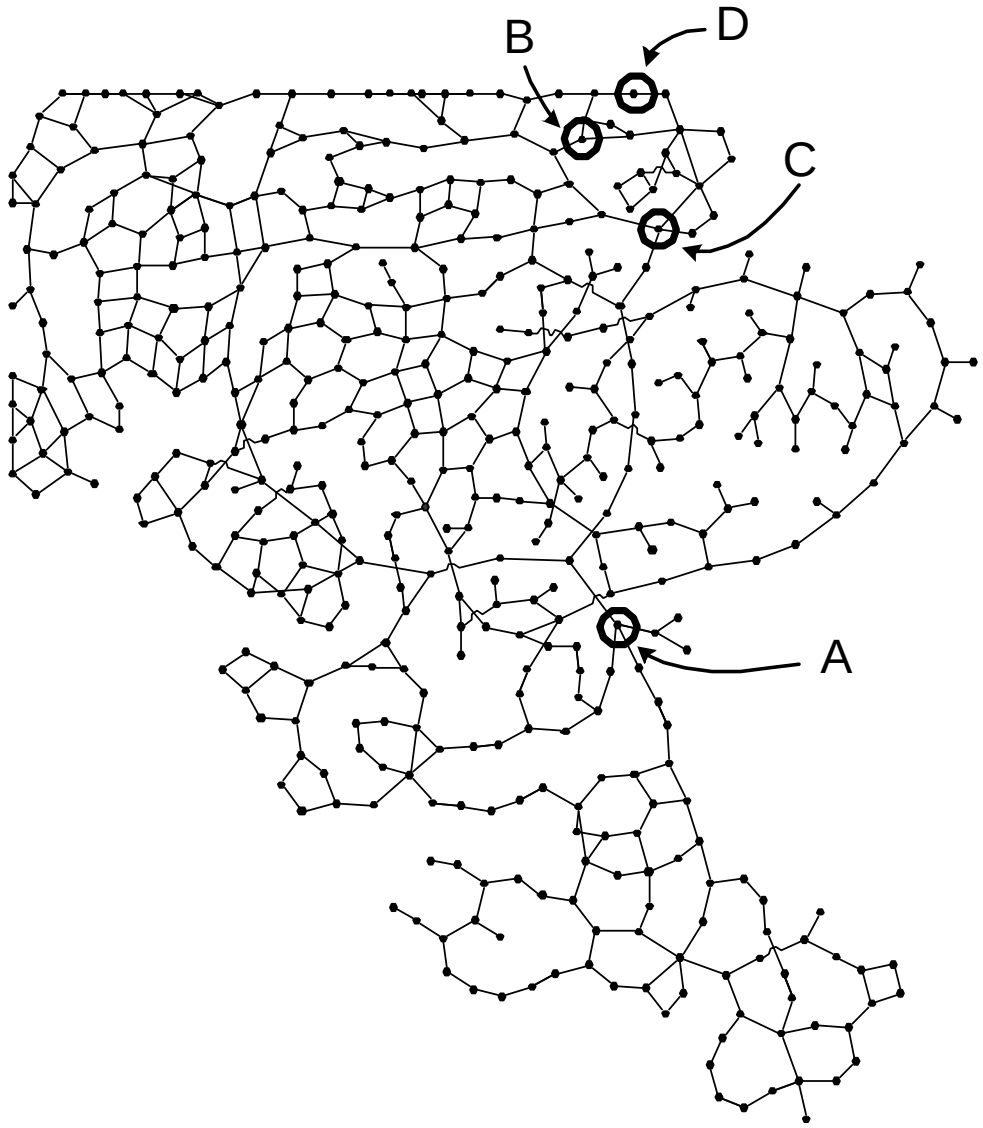

Figure 8: Municipal Water Network Injection Locations. Four simulated injection locations, A through D, are shown in the diagram of the network model.

Figure 9 shows the results for over 1000 tests on the four injection scenarios, using the same formulation parameters as the previous example. For each of the injection locations, we vary both the number of sensors and the time of optimization. Randomizing a list of all the network nodes, we select the first 5 nodes as sensor locations and run the optimization for each of the times shown. We then increase the number of sensors by adding additional sensors from the remaining nodes on the list. In this way, the sensors from the previous configuration remain in their original locations and we only add new sensor locations. By varying the time of the optimization, we are able to deduce the identification time for the particular scenario and sensor configuration. With the 5 minute integration stepsize and 8 hour time horizon, each optimization problem is a large scale nonlinear program with 212352 variables, 167232 equality constraints, and 45120 inequality constraints[3]. For each optimization, we plot the rank of the solution profile for the simulated injection node, shown by the shading scale to the right.

The hashed regions indicate tests where none of the installed sensors had yet registered any contaminant and thus there is no information to run the optimization. At $t = 0$ hours, the time of the simulated injection, no sensors have measured contaminant, therefore we do not expect any result for this entire column. In the top row of each contour plot, sensors are installed at every node, and we expect rank 1 for each of the simulated injection scenarios.

The formulation is very effective for locations A through C, where we can determine the injection location in very little time with few sensors. It is important to remember that we have placed sensors randomly and are only simulating a 30 minute injection. We expect even better results when sensors are placed optimally and injections are longer.

The effectiveness is influenced dramatically by the flow patterns at the injection location. Although injection locations B,C, and D are all within the same region of the network, the effectiveness of the formulation is different for location D. The flow conditions from injection location D do not cause significant spreading of the contaminant through the network, and we expect a higher requirement on the number of installed sensors. Nevertheless, once there are enough sensors to detect the contamination, the formulation is extremely effective at determining the correct injection location and optimal sensor placement should reduce the number of required sensors.

[3]The optimization problem is formulated for 96 timesteps, each with 470 node concentrations, 470 injection terms, 636 pipe concentrations, for both the inlet and the outlet. This gives us $96 \cdot (2 \cdot 470 + 2 \cdot 636) = 212352$ variables, $96 \cdot (1 \cdot 470 + 2 \cdot 636) = 167232$ equality constraints, and $96 \cdot 470 = 45120$ variable bounds.

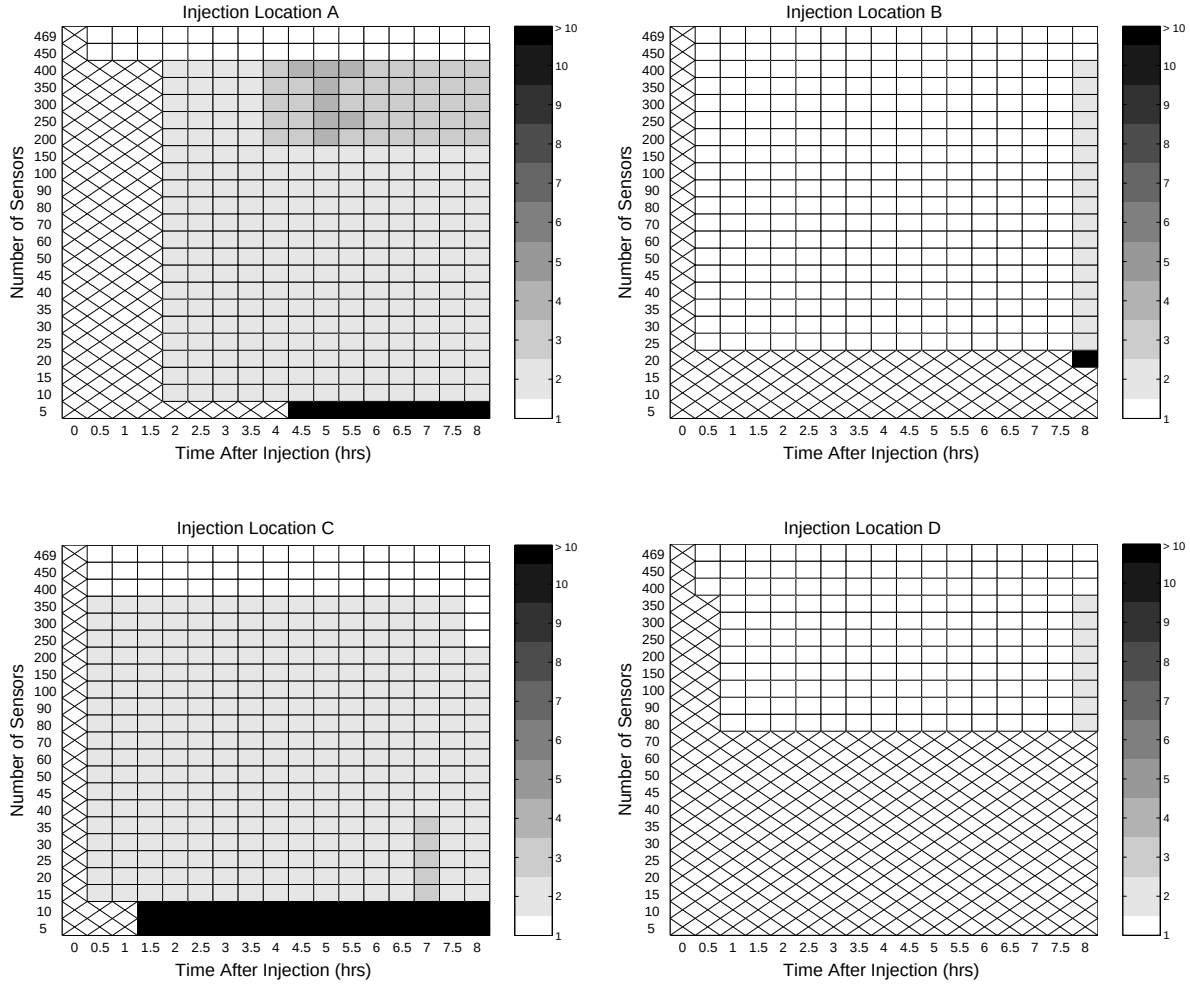

Figure 9: Rank of Optimization Solution for Simulated Injection Node. This figure illustrates the effectiveness of the formulation in determining the correct injection node for simulated injections A through D. The rank of the injection node is shown with the shading to the right of each contour plot. A low ranking indicates that the formulation has been effective at identifying the injection location.

5 Conclusions and Future Work

We have presented an algorithm for reformulating the pipe constraints of network water quality models, dramatically reducing the size of the discretized problem. This algorithm provides a straightforward mathematical representation of the pipe boundary concentrations, and is efficient for large networks with many source and output nodes. Although we have only demonstrated the origin tracking algorithm for the contaminant source determination problem, its usefulness extends to other optimization and control problems.

The results of problem formulation (14) are very promising for the contamination source determination problem and it is important to further develop the technology. Some areas of future work include:

- The results for location D illustrate the importance of determining optimal sensor locations. Current work in this area is useful for detection systems, but the objective measures used may not be appropriate for the contamination source determination problem. With estimates of the likely flow patterns through the network, we could use network observability tools to determine sensor locations that reduce the amount of nonuniqueness in solutions. While our formulation performs well on the test examples studied, we can make no guarantee that the regularized solution includes all possible injection scenarios. Also, to guarantee a unique solution of the unregularized problem for any and all possible injection scenarios, we would require an installed sensor at each and every node in the network. It may be important to reduce the number of possible injection locations to make a guarantee of uniqueness for optimal sensor placement.

- No analysis has yet been done to test the reliability of this formulation in the face of sensor failure or noise in flow rates and sensor measurements. An actual implementation would likely need to include a robust estimation phase that could remove potential outliers in the measurement data. It is also important to quantify the uncertainties associated with the data.

- Many contaminants will experience decay as they propagate through the network. Reaction terms would enter the formulation through the models for the pipes and the storage tanks. The storage tank equations are dependent on time alone and can be easily modified to include reaction terms. The pipe equations are reformulated with the origin tracking algorithm, which could be modified to include first order kinetic decay expressions. Each element has a corresponding fixed volume, and the elapsed time in the pipe is known from the resulting time delay. We would also now require identification of the contamination species to include the correct reaction rate.

- In each of the tests, the time for the optimization to execute on a 2.2 GHz Pentium 4 machine was under two minutes. This demonstrates that the formulation could be used in a real time setting for the particular model studied. It is necessary to examine the performance of this formulation for much larger community networks. Since the time delays associated with large water networks are often long, it is reasonable to assume that one could formulate the contamination source determination problem on a subset of the entire network, formulating problem (14) for a region around the detected contaminant. If multiple injections are suspected, multiple instances could be formulated and solved simultaneously.

In conclusion, we have demonstrated that our problem formulation is very effective at identifying the location of potential contamination sources in the water network. Using the origin tracking algorithm, we can use the direct simultaneous approach to solve the fully time dependent contamination source determination problem, a result that was previously not possible with the direct sequential approach.

6 Acknowledgements

Continued gratitude goes to Sandia National Laboratories and the National Science Foundation (under ITR Grant ACI-021667) for their financial support. The authors would like to specifically thank many individuals at the Computer Science Research Institute, Sandia National Laboratories, Albuquerque, New Mexico and others in the water network research community for their assistance and insight.

References

[1] Volkan Akcelik, George Biros, and Omar Ghattas. Parallel multiscale guass-newton-krylov methods for inverse wave propogation. *Proceedings of the IEEE/ACM SC2002 Conference*, November 2002.

[2] Jonathan Berry, Lisa Fleischer, William Hart, and Cynthia Phillips. Sensor placement in municipal water networks. *Proceedings of the EWRI Conference, Philadelphia*, 2003.

[3] Lorenz T. Biegler, Arturo M. Cervantes, and Andreas Wächter. Advances in simultaneous strategies for dynamic process optimization. Technical report, Department of Chemical Engineering, Carnegie Mellon University, February 2001. CAPD Technical Report B-01-01.

[4] A. Cervantes and Lorenz T. Biegler. Optimization strategies for dynamic systems. *In C. Floudas, P. Pardalos (Eds.), Encyclopedia of Optimization*, 2003.

[5] Robert Fourer, David M. Gay, and Brian W. Kernighan. *AMPL: A Modeling Language for Mathematical Programming.* The Scientific Press (now an imprint of Boyd & Fraser Publishing Co.), Danvers, MA, USA, 1993.

[6] P. K. Lamm. Inverse problems and ill-posedness. *Inverse Problems in Engineering, Theory and Practice, Americal Society of Mechanical Engineers*, pages 1–10, 1993.

[7] C. P. Liou and J. R. Kroon. Modeling the propogation of water-borne substances in distribution networks. *J. Am. Water Works Assoc.*, 79(11):54–58, 1987.

[8] Stephen G. Nash and Ariela Sofer. *Linear and Nonlinear Programming*. The McGraw-Hill Companies, Inc., 1996.

[9] Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer-Verlag New York, Inc., 1999.

[10] Avi Ostfeld and Elad Salomons. An early warning detection system (ewds) for drinking water distribution systems security. *Proceedings of the EWRI Conference, Philadelphia*, 2003.

[11] Marielba Rojas. *A Large-Scale Trust-Region Approach to the Regularization of Discrete Ill-Posed Problems*. PhD thesis, Rice University, May 1998.

[12] L.A. Rossman. *EPANET User's Manual*. Risk Reduction Engineering Laboratory, U.S. EPA, Cincinnati, 2000.

[13] Lewis A. Rossman and Paul F. Boulos. Numerical methods for modeling water quality in distribution systems: A comparison. *Journal of Water Resources Planing and Management*, 122(2):137–146, March/April 1996.

[14] Lewis A. Rossman, Paul F. Boulos, and Tom Altman. Discrete volume-element method for network water-quality models. *Journal of Water Resources Planning and Management*, 119(5), September/October 1993.

[15] Feng Shang, James G. Uber, and Marios M. Polycarpou. Particle backtracking algorithm for water distribution systems analysis. *Journal of Environmental Engineering*, pages 441–450, May 2002.

[16] Bart G. van Bloemen Waanders, Roscoe. A. Bartlett, Lorenz. T. Biegler, and Carl. D. Laird. Nonlinear programming strategies for source detection of municipal water networks. *Presented at EWRI Conference, Philadelphia*, 2003.

[17] Curtis R. Vogel. *Computational Methods for Inverse Problems*. SIAM, Frontiers in Applied Mathematics, 2002.

[18] Andreas Wächter. *An Interior Point Algorithm for Large-Scale Nonlinear Optimization with Applications in Process Engineering*. PhD thesis, Carnegie Mellon University, January 2002.

[19] Michael L. Zierolf, Marios M. Polycarpou, and James G. Uber. Development and autocalibration of an input-output model of chlorine transport in drinking water distribution systems. *IEEE Transactions on Control Sytstems Technology*, 6(4):543–553, July 1998.