Konrad-Zuse-Zentrum
für Informationstechnik Berlin

Takustraße 7
D-14195 Berlin-Dahlem
Germany

MARC C. STEINBACH

On PDE Solution in Transient Optimization of Gas Networks

ZIB-Report 04-46 (November 2004)

ON PDE SOLUTION IN TRANSIENT OPTIMIZATION OF GAS NETWORKS

MARC C. STEINBACH

ABSTRACT. Operative planning in gas distribution networks leads to large-scale mixed-integer optimization problems involving a hyperbolic PDE defined on a graph. We consider the NLP obtained under prescribed combinatorial decisions—or as relaxation in a branch and bound framework, addressing in particular the KKT systems arising in primal-dual interior methods. We propose a custom solution algorithm using sparse local projections, based on the KKT systems' structural properties induced by the discretized gas flow equations in combination with the underlying network topology. The numerical efficiency and accuracy of the algorithm are investigated, and detailed computational comparisons with a control space method and with the multifrontal solver MA27 are provided.

1. INTRODUCTION

The topic of this paper is operative planning, or transient technical optimization (TTO), in gas networks. This planning level addresses the task of controlling the network load distribution over the next 24 to 48 hours to satisfy the actual demand subject to physical, technical, and contractual constraints as well as target values for gas production, storage, purchase, and sale determined by the mid-term planning. The objective is to minimize the variable operating costs, which are dominated by the cost for the gas transport, that is, the fuel consumption of compressors. Due to reliable temperature forecasts we can neglect demand uncertainty and hence use a deterministic model, but the operative planning problem involves PDE constraints (gas flow) as well as substantial combinatorial aspects (start-up and shut-down of compressors, opening or closing of valves, possibly the direction of flow on some lines), leading to a currently intractable PDE constrained mixed-integer optimization problem.

Typical subjects of the early literature include dynamic programming techniques for steady-state optimization (in tree-structured networks) [38], later surveyed in [7], or sequential linearization for nonlinear mixed-integer models on more general network topologies [28]. The papers [3, 6, 8] study the technical difficulties as well as criteria for the comparison and evaluation of compressor optimization based on mixed-integer models. Related topics include optimization of single compressor stations by simulated annealing [39], or optimization of gas networks by Nash equilibria [27]. Probably the most intensively studied subject is transient network simulation: commercial simulation tools such as SIMONE [40] are available for this purpose, based on highly detailed physical models of gas dynamics and compressor behavior [21, 22, 23, 24]. The authors of this system also propose a gradient method for transient network optimization under given binary decisions [19, 20, 35]. More recently, an extended simplex method was developed for a quasi-stationary model [9]. First approaches for the mixed-integer TTO problem, with rather coarse approximations of nonlinearities, are developed in [31, 32] and later in [17]. To address the full TTO problem, our own work aims at a future integration with linear mixed-integer approaches that are currently being developed [25, 26]. In the same context,

---

2000 *Mathematics Subject Classification.* 93C20, 65F50, 76D55, 90C30, 90C35.

Key words and phrases. Gas network, operative planning, PDE constrained optimization, KKT system factorization, sparse projection method.

This work has been supported by the Federal Ministry of Education and Science (BMBF) under grant 03STM5B4.

stochastic models for mid-term planning are investigated in [18, 36]. For basic texts on gas dynamics we refer to [30, 37].

As in [13, 14], we focus on the nonlinear aspects in this paper, assuming that combinatorial decisions are externally given—ideally by an enclosing mixed-integer optimization framework such as branch and bound. In [13] we have developed a suitable NLP model and validated it on a small test network using the general purpose SQP code SNOPT [16]. The highly structured NLP model is characterized by an underlying network providing the coupling between

- the gas flow in pipes governed by a (discretized) PDE,
- the compressors as nonlinear control elements,
- further active and passive linear elements.

As the majority of network elements are pipes, the PDE defined on the network graph is largely responsible for the overall complexity. Our goal here is the construction of KKT solvers for interior methods that are sufficiently fast to act as standalone decision support tool in operative planning, or as subproblem solvers for the NLP relaxations within a mixed-integer optimization framework. Although iterative solvers are well-studied both for KKT systems in PDE constrained optimization and for classical (linear) network flow problems, the situation at hand appears to be mostly unexplored. No preconditioners are known for this problem type. On the other hand, efficient direct algorithms can be constructed for tree-structured networks. Real gas distribution networks are more complex but do not have too many loops on the large scale. This suggests to investigate direct KKT solvers that exploit the specific problem structure, which is the approach pursued in the following.

The material is organized as follows. In Section 2 we summarize the overall network model developed in [13], then formulate the complete NLP model and highlight its structural properties. The KKT systems obtained in primal-dual interior methods are presented in Section 3. In Section 4 we discuss two direct solution algorithms that exploit the structural characteristics of the discretized PDE and network topology by different projection techniques. The new recursive algorithm with sparse local projections developed here, in particular, achieves linear complexity in the number of timesteps. An extensive computational study comparing the two solvers with each other and with the well-known multifrontal code MA27 is then provided in Section 5, addressing runtime and memory requirements as well as solution accuracy. We conclude the paper by discussing promising directions of future research.

2. NLP Model

The model consists of discretized dynamic equations for the network elements, a terminal condition, simple bounds on all variables, and a linear objective. Details are given in [13]; here we summarize the overall model with some slight modifications to ease presentation.

2.1. **Network Topology and Planning Horizon.** The network is modeled as a directed graph $G = (\mathcal{N}, \mathcal{A})$ whose vertex set consists of provider nodes $\mathcal{N}_+$ (sources), customer nodes $\mathcal{N}_-$ (sinks), and interior nodes $\mathcal{N}_0$ (junctions),

$$\mathcal{N} = \mathcal{N}_+ \cup \mathcal{N}_- \cup \mathcal{N}_0. \tag{1}$$

The arc set consists of pipes $\mathcal{A}_{\text{pi}}$, connections $\mathcal{A}_{\text{cn}}$, compressors $\mathcal{A}_{\text{cs}}$, valves $\mathcal{A}_{\text{vl}}$, and regulators $\mathcal{A}_{\text{rg}}$,

$$\mathcal{A} = \underbrace{\mathcal{A}_{\text{pi}} \cup \mathcal{A}_{\text{cn}}}_{\text{passive}} \cup \underbrace{\mathcal{A}_{\text{cs}} \cup \mathcal{A}_{\text{vl}} \cup \mathcal{A}_{\text{rg}}}_{\text{active (controlled)}}. \tag{2}$$

Individual arcs will be denoted as $a \in \mathcal{A}$ or, using the tail and head $i, j \in \mathcal{N}$, as $ij \in \mathcal{A}$. The flow is directed from i to j.

We consider a uniform time grid $t = 0, 1, \ldots, t_e$ on the finite horizon $I = [0, t_e]$. Subinterval $I_t = (t-1, t)$ is referred to as period t and has physical length Δt.

Basic state variables are the node pressures p_{jt} and the arc inflows and outflows $q^{\text{in}}_{at}, q^{\text{out}}_{at}$ at time instances $t = 1, \ldots, t_e$. Fixed initial states are given at $t = 0$. When considering a single arc $a = ij \in \mathcal{A}$, we write q_{it}, q_{jt} instead of $q^{\text{in}}_{at}, q^{\text{out}}_{at}$. Control variables are the pressure changes Δp_{at} at regulators and compressors; they are constant in each period $t = 1, \ldots, t_e$.

2.2. **Model Equations.** The node and arc equations in the following are valid for all $t \in \{1, \ldots, t_e\}$, where we assume that all discrete decisions are externally prescribed (compressors: on/off, valves and regulators: open/closed). Each constraint is given a name for later reference.

Standard flow balance equations hold at every internal node $j \in \mathcal{N}_0$,

$$c^{\text{flow}}_{jt} = \sum_{i\colon ij\in\mathcal{A}} q^{\text{out}}_{ijt} - \sum_{k\colon jk\in\mathcal{A}} q^{\text{in}}_{jkt} = 0. \tag{3}$$

At the customer nodes $j \in \mathcal{N}_-$ we have to include the predicted demands D_{jt},

$$c^{\text{flow}}_{jt} = \sum_{i\colon ij\in\mathcal{A}} q^{\text{out}}_{ijt} - \sum_{k\colon jk\in\mathcal{A}} q^{\text{in}}_{jkt} - D_{jt} = 0. \tag{4}$$

Interior nodes could thus be modeled as customer nodes with zero demand. We keep the distinction since, in contrast to water distribution networks [4, 5], the number of customer nodes in gas networks is typically small.

At the provider nodes $j \in \mathcal{N}_+$ we consider the pressures to be given,

$$c^{\text{press}}_{jt} = p_{jt} - \hat{p}_{jt} = 0, \tag{5}$$

typically specified as hourly profiles according to contractual agreements.

The pipes $a = ij \in \mathcal{A}_{\text{pi}}$ have nonlinear pressure and flow relations obtained from implicit Euler discretizations in space and time [13],

$$c^{\text{cont}}_{at} = \frac{\rho_{jt} - \rho_{jt_-}}{\Delta t} + \frac{q_{jt} - q_{it}}{L_a} = 0, \tag{6}$$

$$c^{\text{loss}}_{at} = A_a \frac{p_{jt} - p_{it}}{L_a} + g \frac{h_j - h_i}{L_a} \rho_{jt} + \frac{\lambda(q_{jt})}{2D_a} \frac{q^2_{jt}}{\rho_{jt}} = 0, \tag{7}$$

$$c^{\text{state}}_{at} = p_{jt} - \gamma(T_{jt}) z(p_{jt}, T_{jt}) \rho_{jt} = 0. \tag{8}$$

Here we abbreviate $t_- = t - 1$ and, as mentioned above, $q_{it} = q^{\text{in}}_{at}$, $q_{jt} = q^{\text{out}}_{at}$. The friction coefficient λ and the compressibility factor z are given empirically.

Connections $a = ij \in \mathcal{A}_{\text{cn}}$ are short pipes with a constant relative pressure loss $c_a \in (0, 1]$ and no change in the flow rate,

$$c^{\text{press}}_{at} = p_{jt} - c_a p_{it} = 0, \quad c^{\text{flow}}_{at} = q_{jt} - q_{it} = 0. \tag{9}$$

The valves $a = ij \in \mathcal{A}_{\text{vl}}$ are control elements that can be either open or closed,

$$c^{\text{press}}_{at} = p_{jt} - p_{it} = 0, \qquad c^{\text{flow}}_{at} = q_{jt} - q_{it} = 0 \qquad \text{(open)}, \tag{10}$$

$$c^{\text{press}}_{at} = q_{it} = 0, \qquad c^{\text{flow}}_{at} = q_{jt} = 0 \qquad \text{(closed)}. \tag{11}$$

Regulators (or control valves) $a = ij \in \mathcal{A}_{\text{rg}}$ reduce the pressure by a controlled positive amount $\Delta p_{at} \in [\Delta p^-_a, \Delta p^+_a] \subset \mathbf{R}_+$,

$$c^{\text{press}}_{at} = p_{jt} - p_{it} + \Delta p_{jt} = 0, \qquad c^{\text{flow}}_{at} = q_{jt} - q_{it} = 0 \qquad \text{(open)}, \tag{12}$$

$$c^{\text{press}}_{at} = q_{it} = 0, \qquad c^{\text{flow}}_{at} = q_{jt} = 0 \qquad \text{(closed)}. \tag{13}$$

In order to move the gas, compressors increase the pressure by a controlled nonnegative amount Δp_{at}, taking their fuel B_{at} from the inflow,

$$\begin{aligned} c_{at}^{\text{press}} &= p_{jt} - p_{it} - \Delta p_{jt} = 0, \\ c_{at}^{\text{flow}} &= q_{jt} - q_{it} + B_{at} = 0, \\ c_{at}^{\text{fuel}} &= \beta_a(p_{it}, p_{jt}, q_{jt}) - B_{at} = 0 \qquad \text{(on)}, \end{aligned} \tag{14}$$

$$\begin{aligned} c_{at}^{\text{press}} &= p_{jt} - p_{it} = 0, \\ c_{at}^{\text{flow}} &= q_{jt} - q_{it} = 0, \\ c_{at}^{\text{fuel}} &= B_{at} = 0 \qquad \text{(off)}. \end{aligned} \tag{15}$$

The fuel consumption depends nonlinearly on the pressures and throughput,

$$\beta_a(p_{it}, p_{jt}, q_{jt}) = C_a N_{at} = C_{at} q_{jt} z(p_{it}, T_{it}) \left[\left(\frac{p_{jt}}{p_{it}} \right)^{\frac{\kappa-1}{\kappa}} - 1 \right],$$

where N_{at} is the compressor's power consumption and C_a, C_{at} are products of several constants [13].

The objective is to minimize the overall fuel cost,

$$\phi = \sum_{t=1}^{t_e} \phi_t = \sum_{a \in \mathcal{A}_{\text{cs}}} c_a \sum_{t=1}^{t_e} \frac{B_{at_-} + B_{at}}{2} \Delta t \to \min. \tag{16}$$

Finally we have a terminal constraint on the total network gas content,

$$c_e = \sum_{ij \in \mathcal{A}_{\text{pi}}} L_{ij} \frac{\rho_{it_e} + \rho_{jt_e}}{2} - m_{\min} = 0. \tag{17}$$

Actually $m_{\min}$ is a lower bound on the gas mass, but the inequality constraint is always active and therefore formulated as an equality.

2.3. **NLP Formulation and Structure.**

The vector of all NLP variables is denoted

$$y = (y_1, \dots, y_{t_e}) \tag{18}$$

and the initial states are z_0 where

$$y_t = (u_t, z_t), \quad z_t = (p_t, q_t, s_t), \quad t = 0, \dots, t_e. \tag{19}$$

States z_t consist of node pressures $p_t = (p_{it})_{i \in \mathcal{N}}$, arc flows $q_t = (q_{at}^{\text{in}}, q_{at}^{\text{out}})_{a \in \mathcal{A}}$, and further states in pipes and compressors, $s_t = (s_{at})_{a \in \mathcal{A}}$. Controls are the pressure changes $u_t = (u_{at})_{a \in \mathcal{A}}$ in compressors and regulators (constant on I_t):

$$\begin{array}{llll} s_{ijt} = \rho_{jt}, & ij \in \mathcal{A}_{\text{pi}}, & u_{at} = \Delta p_{at}, & a \in \mathcal{A}_{\text{cs}}, \\ s_{at} = B_{at}, & a \in \mathcal{A}_{\text{cs}}, & u_{at} = \Delta p_{at}, & a \in \mathcal{A}_{\text{rg}}. \end{array} \tag{20}$$

In the remaining arc types, s_{at} and u_{at} are empty. All variables have simple bounds modeling technical or contractual restrictions, $y \in Y = [y^-, y^+]$.

Thus we can write the NLP in the separable form

$$\begin{aligned} \underset{y \in Y}{\text{minimize}} \quad & \sum_{t=1}^{t_e} \phi_t(z_t) \\ \text{subject to} \quad & c_t(z_{t-1}, u_t, z_t) = 0, \quad t = 1, \dots, t_e, \\ & c_e(z_{t_e}) = 0, \end{aligned}$$

where ϕ_t and c_e denote the period t cost (16) and the terminal constraint (17), respectively, and c_t collects the equality constraints of period t,

$$c_t(z_{t-1}, u_t, z_t) = \begin{pmatrix} \{c^{\text{flow}}_{at}(z_t)\}_{a\in\mathcal{N}_-\cup\mathcal{N}_0} \\ \{c^{\text{press}}_{at}(z_t)\}_{a\in\mathcal{N}_+} \\ \{c^{\text{cont}}_{at}(z_{t-1}, z_t), c^{\text{loss}}_{at}(z_t), c^{\text{state}}_{at}(z_t)\}_{a\in\mathcal{A}_{\text{pi}}} \\ \{c^{\text{press}}_{at}(z_t), c^{\text{flow}}_{at}(z_t)\}_{a\in\mathcal{A}_{\text{cn}}\cup\mathcal{A}_{\text{vl}}} \\ \{c^{\text{press}}_{at}(u_t, z_t), c^{\text{flow}}_{at}(z_t)\}_{a\in\mathcal{A}_{\text{rg}}\cup\mathcal{A}_{\text{cs}}} \\ \{c^{\text{fuel}}_{at}(z_t)\}_{a\in\mathcal{A}_{\text{cs}}} \end{pmatrix}.$$

Observing that almost all constraints depend on the current state z_t only while the previous state z_{t-1} and the control u_t enter linearly, the period t equality constraints can be rewritten

$$c_t(z_{t-1}, u_t, z_t) = L_t z_{t-1} + B_t u_t + \tilde{c}_t(z_t), \quad t = 1, \dots, t_e,$$

yielding the linearized primal constraints system

$$\begin{bmatrix} B_1 & A_1 & & & & \\ & L_2 & B_2 & A_2 & & \\ & & \ddots & \ddots & \ddots & \\ & & & L_{t_e} & B_{t_e} & A_{t_e} \\ & & & & & F_{t_e} \end{bmatrix} \begin{bmatrix} \Delta u_1 \\ \Delta z_1 \\ \vdots \\ \Delta u_{t_e} \\ \Delta z_{t_e} \end{bmatrix} = \begin{bmatrix} h_1 \\ h_2 \\ \vdots \\ h_{t_e} \\ e \end{bmatrix},$$

where

$$\begin{aligned} A_t &= \nabla \tilde{c}_t(z_t) = \nabla_{z_t} c_t(z_{t-1}, u_t, z_t), & h_t &= -c_t(z_{t-1}, u_t, z_t), \\ F_{t_e} &= \nabla c_e(z_{t_e}), & e &= -c_e(z_{t_e}). \end{aligned}$$

Thus we obtain the separable Lagrangian

$$\begin{aligned} \mathcal{L}(u, z, \lambda, \eta) &= \sum_{t=1}^{t_e} \mathcal{L}_t(z_{t-1}, u_t, z_t, \lambda_t) + \mathcal{L}_e(z_{t_e}, \eta), \\ \mathcal{L}_t(z_{t-1} u_t, z_t, \lambda_t) &= \phi_t(z_t) - \lambda_t^* c_t(z_{t-1}, u_t, z_t), \\ \mathcal{L}_e(z_{t_e}, \eta) &= \eta^* c_e(z_{t_e}), \end{aligned}$$

and the dual feasibility system

$$\begin{bmatrix} 0 & & & & \\ & H_1 & & & \\ & & \ddots & & \\ & & & 0 & \\ & & & & H_{t_e} \end{bmatrix} \begin{bmatrix} \Delta u_1 \\ \Delta z_1 \\ \vdots \\ \Delta u_{t_e} \\ \Delta z_{t_e} \end{bmatrix} - \begin{bmatrix} B_1^* & & & \\ A_1^* & L_2^* & & \\ & B_2^* & \ddots & \\ & A_2^* & \ddots & L_{t_e}^* \\ & & \ddots & B_{t_e}^* \\ & & & A_{t_e}^* & F_{t_e}^* \end{bmatrix} \begin{bmatrix} \Delta\lambda_1 \\ \Delta\lambda_2 \\ \vdots \\ \Delta\lambda_{t_e} \\ \Delta\eta \end{bmatrix} = \begin{bmatrix} d_1 \\ f_1 \\ \vdots \\ d_{t_e} \\ f_{t_e} \end{bmatrix},$$

where

$$\begin{aligned} H_t &= \nabla_{z_t z_t} \mathcal{L}(u, z, \lambda, \eta) = \nabla_{z_t z_t} \mathcal{L}_t(z_{t-1}, u_t, z_t, \lambda_t) = -\nabla_{z_t z_t}[\tilde{c}_t(z_t)^* \lambda_t], \\ d_t &= -\nabla_{u_t} \mathcal{L}(u, z, \lambda, \eta) = -\nabla_{u_t} \mathcal{L}_t(z_{t-1}, u_t, z_t, \lambda_t) = B_t^* \lambda_t, \\ f_t &= -\nabla_{z_t} \mathcal{L}(u, z, \lambda, \eta) \\ &= -\nabla_{z_t}[\mathcal{L}_t(z_{t-1}, u_t, z_t, \lambda_t) + \mathcal{L}_{t+1}(z_t, u_{t+1}, z_{t+1}, \lambda_{t+1}) + \mathcal{L}_e(z_{t_e}, \eta)] \\ &= -\nabla \phi_t(z_t) + A_t^* \lambda_t + L_{t+1}^* \lambda_{t+1} + \delta_{t t_e} F_{t_e}^* \eta. \end{aligned}$$

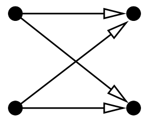

FIGURE 1. Unnatural subnetwork creating a structural singularity: flows in the four connections are not uniquely determined by the inlet and outlet flows

3. KKT SYSTEMS

First computational experiments with our optimization model [13] have been conducted with the general purpose SQP software SNOPT/SnadiOpt [16, 15]. The present work aims at specialized KKT solvers to be employed within primal-dual interior methods. In what follows, we reorder the variables as

$$y = (z, u) \in \mathbf{R}^N, \quad z = (z_1, \dots, z_{t_e}) \in \mathbf{R}^{N_z}, \quad u = (u_1, \dots, u_{t_e}) \in \mathbf{R}^{N_u},$$

where $N_z = n_z t_e$, $N_u = n_u t_e$, and $N = N_z + N_u$. Since all NLP inequalities are simple bounds, the (reduced) KKT system then takes the form

$$\begin{bmatrix} H + \Phi_z & & A^* & F^* \\ & \Phi_u & B^* & \\ A & B & & \\ F & & & \end{bmatrix} \begin{bmatrix} \Delta z \\ \Delta u \\ -\Delta\lambda \\ -\Delta\eta \end{bmatrix} = - \begin{bmatrix} f \\ d \\ h \\ e \end{bmatrix} \in \mathbf{R}^{N_z+N_u+N_z+1}. \tag{21}$$

Here Φ_z, Φ_u are positive diagonal matrices containing state and control barrier terms, respectively, the Hessian H and control operator B are block-diagonal, the state operator A is block-bidiagonal because of the implicit Euler scheme in time, and F corresponds to the single linear terminal constraint,

$$A = \begin{bmatrix} A_1 & & & \\ L_2 & A_2 & & \\ & \ddots & \ddots & \\ & & L_{t_e} & A_{t_e} \end{bmatrix}, \qquad \begin{aligned} H &= \mathrm{Diag}(H_1, \dots, H_{t_e}), \\ B &= \mathrm{Diag}(B_1, \dots, B_{t_e}), \\ F &= (0, \dots, 0, F_{t_e}). \end{aligned} \tag{22}$$

It can be shown that the local state operators $A_t \in \mathbf{R}^{n_z \times n_z}$ are nonsingular under natural assumptions on the network composition. (A simple "counterexample" is shown in Fig. 1.) Thus, in particular, every subset of rows has full row rank. The full KKT system is also nonsingular since Φ_z, Φ_u are positive. These facts will be exploited in constructing direct solvers for the large and sparse system (21).

4. KKT SOLUTION

4.1. **Control Space Projection.** A first structured KKT solver has been presented in [14], which generalizes the classical control space projection (condensing recursion) [2] to the sparse implicit state equation of interest, $A\Delta z + B\Delta u = h$. This algorithm employs sparse factorizations of A_t obtained with MA28 [10, 11] to construct factorizations of A and A^* as forward and backward recursions over time. The factors are then used to eliminate Δz and $\Delta\lambda$, yielding the control space KKT system

$$\begin{bmatrix} S & \bar{F}^* \\ \bar{F} & \end{bmatrix} \begin{bmatrix} \Delta u \\ -\Delta\eta \end{bmatrix} = \begin{bmatrix} \bar{d} \\ \bar{e} \end{bmatrix} \in \mathbf{R}^{N_u+1}.$$

Here the projected Hessian and the projected terminal constraint matrix are

$$S = \Phi_u + \bar{B}^*(H + \Phi_z)\bar{B} \quad \text{and} \quad \bar{F} = F\bar{B} \quad \text{with} \quad \bar{B} = A^{-1}B.$$

Block rows of the block lower triangular transformation $\bar{B}$ are calculated during the factorization to accumulate S forward in time, whereas matrix-vector products with $A^{-1}B$ and

B^*A^{-*} are calculated during forward and backward substitution. The projected system is solved by a dense Cholesky factorization of S. This way the algorithm needs storage only for S and for the factors of A_t, plus workspace to hold twice the largest block row of $\bar{B}$ ($2n_z N_u$ elements).

A key ingredient of this approach is the cheap factorization of A_t. For tree-structured networks it can actually be achieved in complexity $O(|\mathcal{A}|)$ which may, however, be unstable. Using MA28, we obtain a stable factorization at comparable effort even for realistic networks (depending on the topology). This is confirmed by the empirical observation that the required storage always remains below twice the number of original entries in our tests. Memory and runtime complexity of the algorithm are therefore $O(N_u^2)$ and $O(N_u^3)$. Assembling and factorizing S with $O(N_u^3)$ dominates the computational effort (> 99.5% on all instances considered below). Thus, since factorizing A_t is inexpensive and the control space dimension N_u is independent of the space discretization and comparatively small, the algorithm is well suited for networks with a moderate number of compressors and regulators under coarse time discretizations but possibly fine space discretizations.

To handle larger networks, the idea was initially to replace the expensive operations with S by a conjugate gradients method using some natural preconditioner, such as the diagonal blocks of S or some wider band about the diagonal. It turned out, however, that S is diagonally non-dominant with slowly decaying off-diagonal entries, and that the eigenvalues are not well clustered. This led us to investigate alternative approaches, resulting in the algorithm described below.

4.2. **Local Projection Algorithm.** A much faster solution algorithm featuring essentially the same stability properties is based on the *implicit tree-sparse recursion* developed in [33, 34], in combination with *sparse local projections*. We drop the distinction of state and control variables and split the rows of $L_t\Delta z_{t-1} + B_t\Delta u_t + A_t\Delta z_t = h_t$ into *local constraints* and *transition equations*. The latter are derived from the continuity equation $c_{at}^{\text{cont}}(z_{t-1}, z_t)$ and provide the complete coupling between time steps; they consist precisely of the rows that have nonzero entries in L_t. In the notation of [33, 34], these transition equations take the implicit form

$$G_t\Delta y_{t-1} = P_t\Delta y_t + h_t. \tag{23}$$

The remaining rows do not involve z_{t-1} (but do include the terminal constraint at $t = t_e$) and constitute the local constraints

$$F_t^y\Delta y_t = e_t^y. \tag{24}$$

Dual feasibility conditions complete the KKT system:

$$(H_t + \Phi_t)\Delta y_t + P_t^*\Delta\lambda_t - G_{t+1}^*\Delta\lambda_{t+1} - F_t^{y*}\Delta\mu_t^y = f_t. \tag{25}$$

Here $\Delta\lambda_t, \Delta\mu_t^y$ are the dual variables associated with transition equations and local constraints, respectively, and $H_t + \Phi_t$ now denotes the barrier Hessian with respect to y_t rather than just z_t. In what follows we absorb Φ_t into H_t. The respective dimensions of (23), (24), and (25) are

$$l_t = |\mathcal{A}_{\text{pi}}|, \quad l_t^y = |\mathcal{N}| + 2|\mathcal{A}| + |\mathcal{A}_{\text{cs}}| + \delta_{tt_e}, \quad n_t = n_{zt} + n_{ut}.$$

The full system now takes the form

$$\begin{bmatrix} H & G^* & F^{y*} \\ G & & \\ F^y & & \end{bmatrix} \begin{bmatrix} y \\ -\lambda \\ -\mu^y \end{bmatrix} = \begin{bmatrix} f \\ h \\ e^y \end{bmatrix}, \tag{26}$$

with appropriately defined matrices H, G, F^y (for details see [33, 34]).

Note that the number of transition equations and the null space dimension of F_t^y are both small compared to the number of local constraints, $l_t \ll l_t^y$ and $n_t - l_t^y \ll l_t^y$. This allows us to *decouple space and time*. The local constraints (spatial coupling) are handled independently in each timestep, in that a basis of their null space is constructed. The transition equations (temporal coupling) are then projected into that null space and solved by a recursion over time, yielding a direct KKT solution algorithm whose computational complexity is linear in the number of timesteps. Such an approach is generally infeasible in PDE constrained optimization because of prohibitively expensive projections. In our context, however, the specific structure admits local projections at reasonable cost.

The sparse projection can be executed parallel in time and works as follows. For $t = 1, \dots, t_e$ we first factorize

$$F_t^y = L_t^y \begin{pmatrix} I & 0 \end{pmatrix} U_t \in \mathbf{R}^{l_t^y \times n_t}, \tag{27}$$

where $L_t^y \in \mathbf{R}^{l_t^y \times l_t^y}$ and $U_t \in \mathbf{R}^{n_t \times n_t}$ are nonsingular with L_t^y lower triangular; for further details see Section 4.3. Next, the dense projected constraint matrices (null space operators) are calculated as

$$\begin{aligned} G_{t2} &= G_t U_{t-1}^{-1} \begin{pmatrix} 0 \\ I \end{pmatrix}, \\ P_{t2} &= P_t U_t^{-1} \begin{pmatrix} 0 \\ I \end{pmatrix}, \qquad H_{t22} = \begin{pmatrix} 0 & I \end{pmatrix} U_t^{-*} H_t U_t^{-1} \begin{pmatrix} 0 \\ I \end{pmatrix}. \end{aligned} \tag{28}$$

In contrast to the dense version of [33], however, we do not explicitly form the operators associated with the large complement of the null space,

$$\begin{aligned} G_{t1} &= G_t U_{t-1}^{-1} \begin{pmatrix} I \\ 0 \end{pmatrix}, \\ P_{t1} &= P_t U_t^{-1} \begin{pmatrix} I \\ 0 \end{pmatrix}, \qquad \begin{pmatrix} H_{t11} \\ H_{t12} \end{pmatrix} = U_t^{-*} H_t U_t^{-1} \begin{pmatrix} I \\ 0 \end{pmatrix}. \end{aligned}$$

The latter are only required in form of sparse matrix-vector products during forward and backward substitution, as follows. For $t = 1, \dots, t_e$ we first calculate

$$\begin{aligned} \Delta y_{t1} &= (L_t^y)^{-1} e_t^y, \\ \bar{f}_t &= U_t^{-*} \left[f_t - H_t U_t^{-1} \begin{pmatrix} I \\ 0 \end{pmatrix} \Delta y_{t1} \right], \\ \bar{h}_t &= h_t - G_t U_{t-1}^{-1} \begin{pmatrix} I \\ 0 \end{pmatrix} \Delta y_{t-1,1} + P_t U_t^{-1} \begin{pmatrix} I \\ 0 \end{pmatrix} \Delta y_{t1}. \end{aligned} \tag{29}$$

Here Δy_{t1} is the primal solution vector corresponding to the complement of the null space, and $\bar{f}_t, \bar{h}_t$ are transformed right-hand sides entering the projected KKT system. Backward and forward recursions on the projected system yield the primal null space solution components Δy_{t2} and multipliers $\Delta\lambda_t$ [33], from which Δy_t and $\Delta\mu_t^y$ are finally obtained for $t = t_e, \dots, 1$:

$$\begin{aligned} \Delta\bar{\mu}_t^y &= \bar{f}_{t1} - \begin{pmatrix} I & 0 \end{pmatrix} U_t^{-*} \left[P_t^* \Delta\lambda_t - G_{t+1}^* \Delta\lambda_{t+1} + H_t U_t^{-1} \begin{pmatrix} 0 \\ I \end{pmatrix} \Delta y_{t2} \right], \\ \Delta\mu_t^y &= -(L_t^y)^{-*} \Delta\bar{\mu}_t^y, \\ \Delta y_t &= U_t^{-1} \begin{pmatrix} \Delta y_{t1} \\ \Delta y_{t2} \end{pmatrix}. \end{aligned} \tag{30}$$

The projection reduces the full KKT system (26) to

$$\begin{bmatrix} H_{22} & G_2^* \\ G_2 & \end{bmatrix} \begin{bmatrix} y_2 \\ -\lambda \end{bmatrix} = \begin{bmatrix} \bar{f}_2 \\ \bar{h} \end{bmatrix}, \tag{31}$$

with local representation for $t = 1, \dots, t_e$:

$$H_{t22}\Delta y_{t2} + P_{t2}^*\Delta\lambda_t - G_{t+1,2}^*\Delta\lambda_{t+1} = \bar{f}_{t2}$$
$$G_{t2}\Delta y_{t-1,2} = P_{t2}\Delta y_{t2} + \bar{h}_t.$$

Except for the missing local constraints, this is identical to the original system and can be interpreted as an implicit version of the linear-quadratic regulator problem. The small dimension thus enables the fast recursive solution. Again, we refer to [33] for more details.

4.3. **Implementation.** Our current (preliminary) implementation stores and factorizes F_t^y in dense form using the LQ factorization DGELQF from the LAPACK linear algebra library [1]. DGELQF calculates a factorization of the form (27) where U_t is an orthogonal matrix,

$$F_t^y = L_t^y \begin{pmatrix} I & 0 \end{pmatrix} U_t, \quad U_t^* U_t = I \in \mathbf{R}^{n_t \times n_t}.$$

No pivoting is performed (or required) here, and the orthogonal projection is inherently stable. Alternatively, one could apply a Gauss factorization with pivoting, in which case U_t is a column-permuted upper triangular matrix (and $(I\ 0)\, U_t$ a column-permuted upper trapezoidal matrix),

$$U_t = \begin{pmatrix} R_t & V_t \\ & I \end{pmatrix} \Pi_t, \quad \begin{pmatrix} I & 0 \end{pmatrix} U_t = \begin{pmatrix} R_t & V_t \end{pmatrix} \Pi_t.$$

Eventually we intend to use a sparse factorization of F_t^y, requiring a code where the factors L_t^y and U_t are individually accessible for the subsequent operations (28), (29), and (30). The latter are implemented using sparse multiplications with H_t, G_t, P_t on dense operands. To this end, the forward substitution (29) is rewritten in the form

$$\Delta y_{t1} = (L_t^y)^{-1} e_t^y,$$
$$w_t = U_t^{-1} \begin{pmatrix} I \\ 0 \end{pmatrix} \Delta y_{t1}, \quad \bar{f}_t = U_t^{-*}[f_t - H_t w_t], \quad \bar{h}_t = h_t - G_t w_{t-1} + P_t w_t,$$

and the operations are rearranged such that a single workspace w suffices to hold each vector w_t in turn. For the factorization we use two workspace matrices

$$W_1 = U_t^{-1} \begin{pmatrix} 0 \\ I \end{pmatrix}, \quad W_2 = H_t W_1,$$

to calculate $H_{t22} = W_1^* W_2$. This is faster than accumulating H_{t22} as sum of symmetric rank-1 and rank-2 products with rows of W_1.

Observe finally that, after calculating Δy_t, we may alternatively obtain the local multipliers as

$$\Delta\mu_t^y = -(L_t^y)^{-*} \begin{pmatrix} I & 0 \end{pmatrix} U_t^{-*} \left[f_t - H_t \Delta y_t - P_t^* \Delta\lambda_t + G_{t+1}^* \Delta\lambda_{t+1} \right].$$

This requires separate storage for the vectors $\bar{f}_{t2}$ in the projected system but can be useful to achieve higher accuracy; cf. Section 5.5.

5. Numerical Results

Here we compare the performance of three direct algorithms for the symmetric indefinite KKT system: the public domain multifrontal solver MA27 [12] from the HSL numerical software library, the control space projection algorithm [14], and the new locally projecting recursive algorithm presented above. The comparison includes:

- runtimes (CPU) for factorizing the KKT matrix,
- memory requirements for the inverse: factors and floating-point workspace (but not the bookkeeping overhead),
- profiles of solution accuracies.

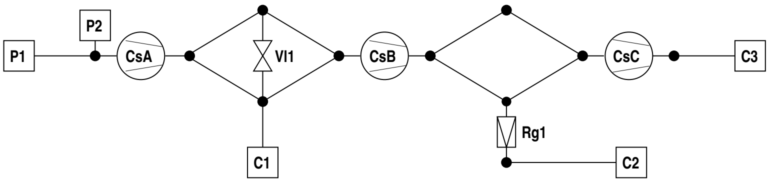

FIGURE 2. Test network T120

TABLE 1. Data of networks used in computations. L_{min}, L_{max}: minimum and maximum pipe lengths; $\mathcal{N}_+, \mathcal{N}_-, \mathcal{N}_0, \mathcal{N}$: node numbers; $\mathcal{A}_{cs}, \mathcal{A}_{rg}, \mathcal{A}_{vl}, \mathcal{A}_{cn}, \mathcal{A}_{pi}, \mathcal{A}$: arc numbers

Name	L_{min}	L_{max}	$\mathcal{N}_+$	$\mathcal{N}_-$	$\mathcal{N}_0$	$\mathcal{N}$	$\mathcal{A}_{cs}$	$\mathcal{A}_{rg}$	$\mathcal{A}_{vl}$	$\mathcal{A}_{cn}$	$\mathcal{A}_{pi}$	$\mathcal{A}$
T120	50.0	120.0	2	3	10	15	3	1	1	1	11	17
T40	25.0	40.0	2	3	27	32	3	1	1	1	28	34
T10	10.0	10.0	2	3	91	96	3	1	1	1	93	99
R235	11.5	235.2	7	10	46	63	13	6	4	12	29	64
R80	11.5	80.0	7	10	60	77	13	6	4	12	43	78
R40	11.5	40.0	7	10	86	103	13	6	4	12	69	104

All computations are performed in core on a 3 GHz Linux PC workstation with 2 GB primary storage.

One difficulty arises in the comparisons: for MA27, neither the memory for the factors nor the workspace for the factorization are accurately predictable. Moreover, the runtime performance improves slightly if more than the minimum required amount is provided. The performance data reported below are obtained by doubling the amount of memory repeatedly until the factorization is successful, starting with twice as many elements as the number of entries in the original KKT system.

5.1. **Networks and Space Discretization.** For the numerical tests we consider two networks with different characteristics: a small test network with just a few control elements and a total length of 920 km depicted in Fig. 2, and a medium-sized network with more than three times as many control elements and a total length of roughly 2500 km, representing the backbone transport network of our industry partner Ruhrgas. Pipelines with up to 120 km and 235 km in length, respectively, correspond to single arcs in the basic networks T120 and R235. We construct refined space discretizations as follows. Each pipeline that exceeds a given maximum length is partitioned into as many arcs of equal length as are needed to remain below the threshold. Thus we obtain three variants of each network, referred to as 'T networks' and 'R networks', with up to approximately hundred nodes and arcs; see Table 1.

5.2. **Time Discretization.** The longest relevant planning horizon, 48 hours, is considered in all test problems. Typical demand scenarios are described in [13]. For the uniform time discretization we use multiples of 48 up to 288 as the number of periods, obtaining timesteps of 60, 30, 20, 15, 12, and 10 minutes. Although there is no need to satisfy a Courant–Friedrichs–Lewy (CFL) type stability condition in our context, some indication on reasonable values of the period length Δt can be obtained by requiring that a flow entering a pipe must not leave it in the same timestep:

$$\Delta t \le \min_{a \in \mathcal{A}_{pi}} \frac{L_a}{v_a^{max}}, \quad v_a^{max} := \max_{t=1,\dots,T} \max \left\{ \frac{q_{at}^{in}}{\rho_{at}^{in}}, \frac{q_{at}^{out}}{\rho_{at}^{out}} \right\}.$$

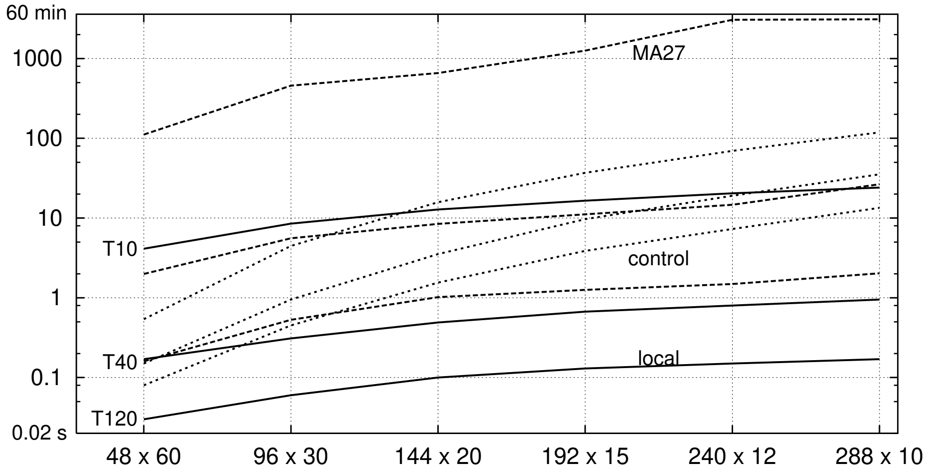

FIGURE 3. CPU time comparison for KKT solvers on T networks

Assuming a maximal gas flow velocity of about $20\,\mathrm{km/h}$ (a realistic value), this inequality yields maximal time steps between $30\,\mathrm{min}$ and $150\,\mathrm{min}$ for the networks in Table 1. This shows that our range of time discretizations contains suitable values for each of the selected space discretizations. However, as we are only studying linear solvers here, computational results will be reported for every possible combination.

5.3. **CPU Time.** Let us first investigate the runtime behavior of the three codes. CPU times for the factorization will be illustrated in plots showing the number and length of timesteps on the abscissa in the form "$t_e \times \Delta t$", with Δt in minutes. Separate curves corresponding to the network variants are plotted for each of the three solvers, where solid, dashed, and dotted lines always correspond to the new locally projecting solver, the control space method, and MA27, respectively. The runtime in seconds is indicated on the ordinate in logarithmic scale.

CPU times for the T networks are displayed in Fig. 3. Comparing the control space method with MA27 first, we observe that the former becomes rapidly slower with increasing number of timesteps whereas the latter becomes drastically slower when the space discretization is refined. MA27 requires up to an hour for factorizing the T10 matrix, whereas the control space method takes never more than two minutes. As expected, the new algorithm performs very well, especially on fine time discretizations. Although it slows down considerably with increasing numbers of pipe segments, it factorizes all T120 and T40 instances in less than a second, and even the largest T10 instance in just 24 seconds. Hence the new algorithm can be considered the clear winner, being outperformed only on the two smallest instances of T10 by the control space method. Note finally that MA27 behaves strangely on T10 for 240 and 288 timesteps: on the larger problem it actually runs 10 seconds faster. This is probably due to the fact that the performance depends on the user-provided workspace memory.

The characteristics of each solver are even more clearly pronounced for the realistic R networks; see Fig. 4. Here MA27 is consistently faster than the control space method, and is in turn always outperformed by the new solver. The differences are quite significant especially on fine time grids, where the control space method needs up to 50 minutes, MA27 up to 6 minutes, and the new solver only up to half a minute. It can be expected that the runtime advantage of the new solver increases further with larger networks, even with the current preliminary implementation.

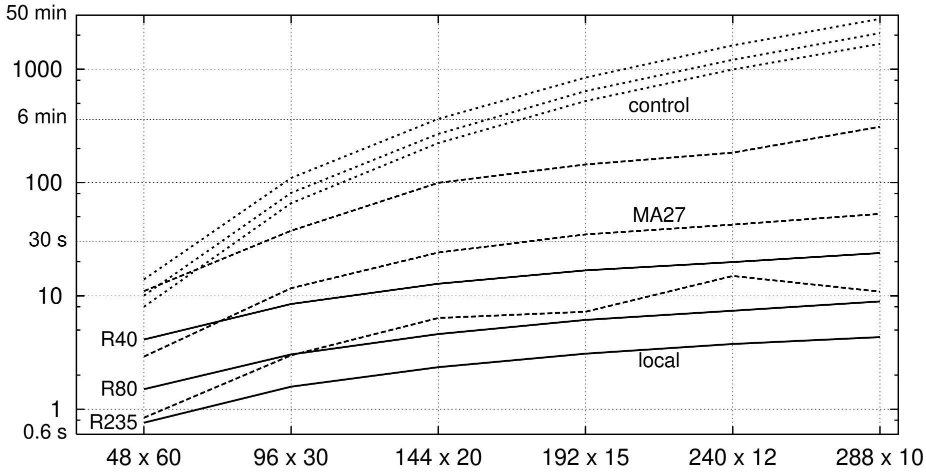

FIGURE 4. CPU time comparison for KKT solvers on R networks

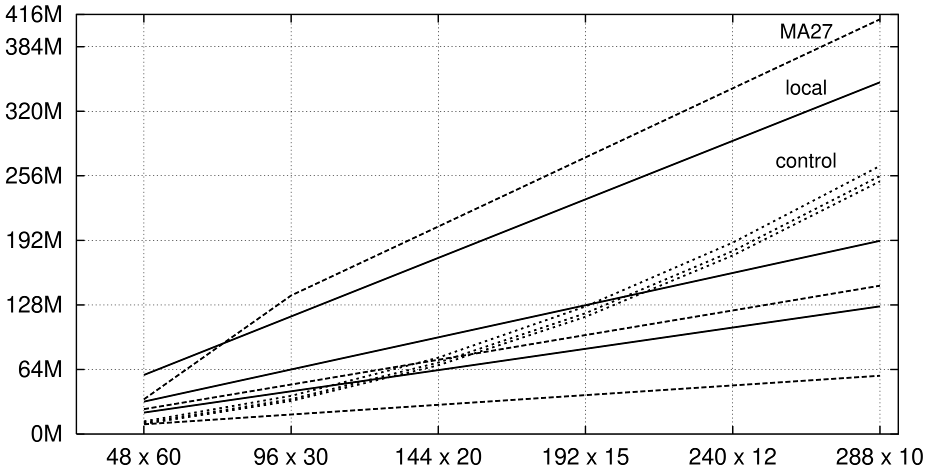

FIGURE 5. Memory comparison for KKT solvers on R networks

5.4. **Memory Requirements.** The plots for memory requirements are organized in the same way as for the CPU times, except that the ordinate displays the total memory in megabytes on a linear scale. Turning to the results, we consider the R networks first; see Fig. 5. The memory required by the new algorithm grows linearly with the number of periods. MA27 appears to show roughly the same behavior, whereas the amount for the control space method grows quadratically. The dependence on the space discretization is almost negligible for the latter (only the workspace grows, not S), while the new solver and especially MA27 show a strong dependence (exactly cubic for the former, apparently also cubic for the latter). In absolute numbers, all problems are solved within less than 412 MB, and the solvers never differ by a factor greater than eight for the same problem.

For the T networks we observe drastic differences displayed in the three plots of Fig. 6. Here the control space method needs very little memory on all instances (because of the small dimension of S), while the new algorithm needs moderate amounts comparable to the R network instances. The requirements of MA27, however, increase disproportionately,

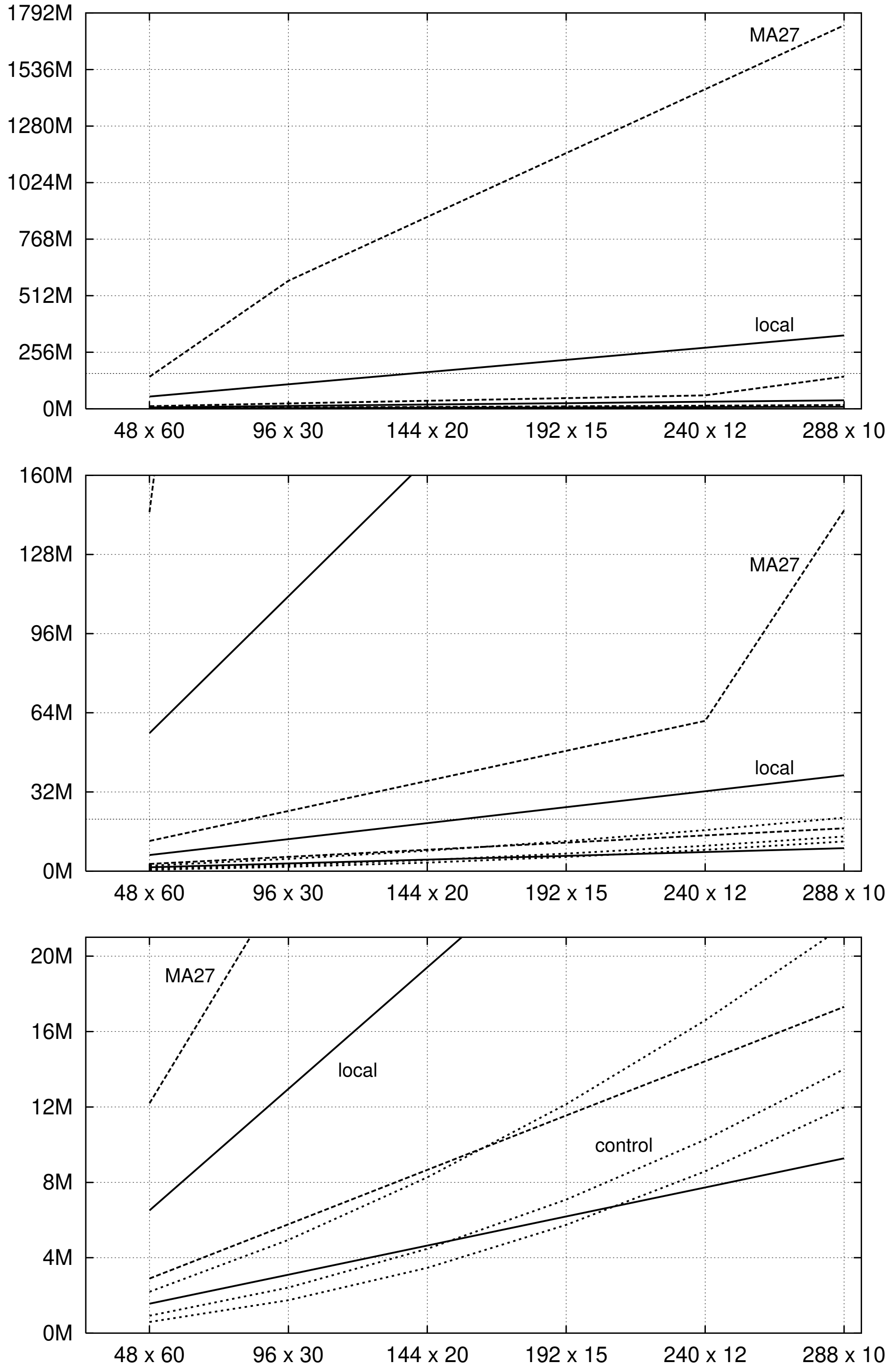

FIGURE 6. Memory comparison for KKT solvers on T networks

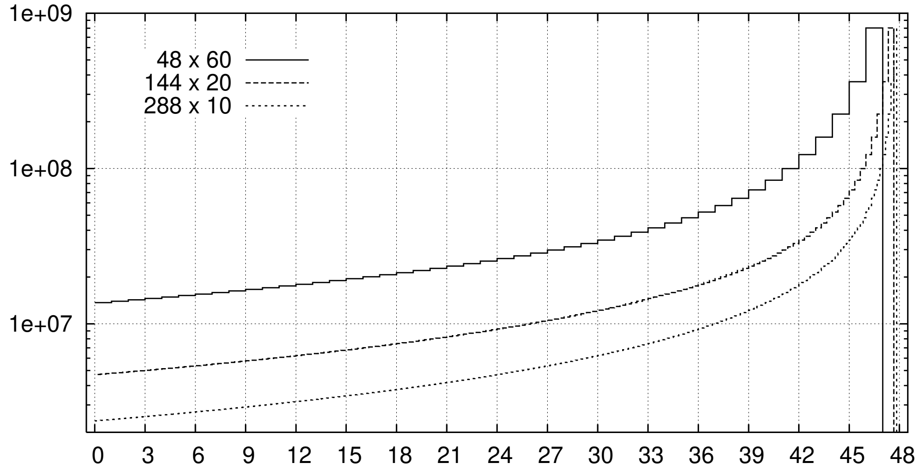

FIGURE 7. Condition numbers of H_{t22} for network T10

growing up to 1736 MB on the T10 instance with 288 periods. This confirms the observation that MA27 runs into difficulties when it has to handle large numbers of pipes. Possible explanations for this behavior could be the increased number of intertemporal constraints (making sparsity pivoting harder), or an increased number of poorly scaled rows (making numerical pivoting harder). Indeed, MA27 has more flexibility for numerical pivoting than the two other solvers, and it does achieve higher solution accuracies; see next section.

5.5. **Solution Accuracy.** To measure the solution accuracy, we generate error profiles as follows. As right-hand side we use the product of the KKT matrix with the vector of all ones, $e = (1, \dots, 1)$. For each solver, the components of the absolute error $\delta y = |y - e|$ are then ordered by increasing magnitude, and we plot the largest absolute error versus the percentage of most accurate components.

As can be expected from the model equations, the condition of the KKT system depends strongly on the scaling of variables or, equivalently, the choice of physical units. Numerical tests confirm the ill-conditioning: if SI base units are used for all variables, none of the algorithms can solve the system to a single digit of accuracy in terms of $\|y - e\|_\infty$. (On the T10 problem with 48 periods, for instance, MA27 yields $\|y - e\|_\infty = 7.8$ using 581 MB and 7:33 minutes. The largest error of the control space method is 4.8×10^6, while the new algorithms stops with a factorization error.) The tests also showed that the condition depends only mildly on the number of timesteps. This is illustrated in Fig. 7, showing for three time discretizations the condition of H_{t22} versus physical time. A detailed discussion of scaling is beyond the scope of this paper, so we just mention that we use a heuristic choice of scaling factors such that the function values and derivatives are reasonably balanced.

In Fig. 8, results are shown for the T10 instance with 144 periods, which is one of the hardest cases. Results for the other problems are similar or better. The control space projection is obviously the least accurate solver and MA27 is the most accurate one, while the new algorithm lies in between. With minor differences, this is basically true for the other problem instances as well. Slightly exceptional behavior occurs for the largest error $\|y - e\|_\infty$: here all solvers yield comparable results, although MA27 still tends to be somewhat more accurate than its competitors.

Looking at the test results separated by primal and dual components in Fig. 9, we see that the primal solution is much more accurate than the dual solution for each of the solvers, which is quite typical for KKT systems. As regards the relative performance of the solvers,

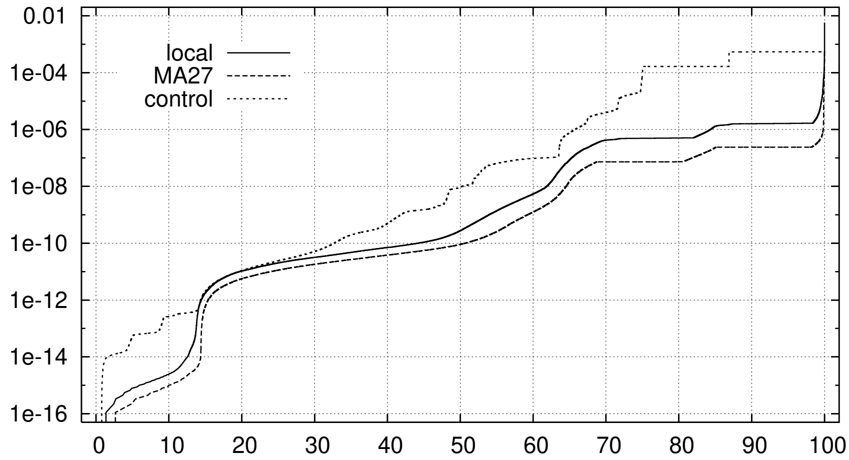

FIGURE 8. Error profiles for network T10 with 144 periods

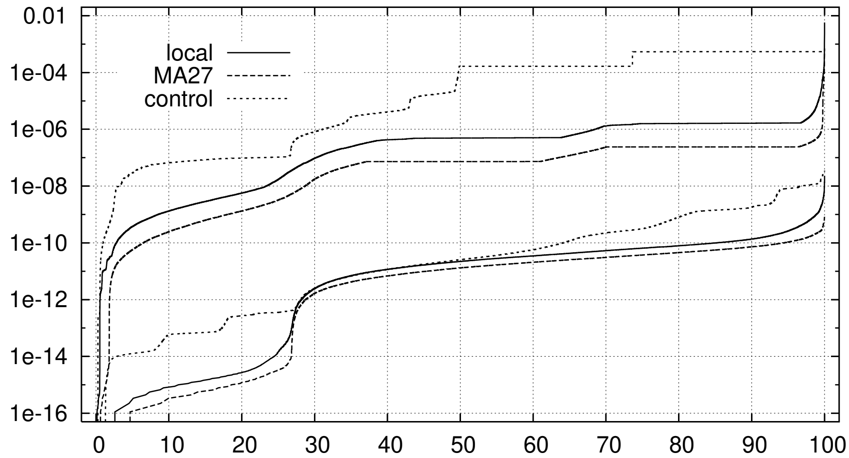

FIGURE 9. Primal and dual errors for network T10 with 144 periods

the same observations as above apply. In any case, the results clearly show that we will need iterative refinement no matter which solver we choose.

6. SUMMARY

Transient optimization in gas networks is difficult even under specified combinatorial decisions. Since general purpose NLP solvers are too slow for realistic problem sizes, our goal is the construction of a highly efficient method for the large, potentially ill-conditioned KKT systems arising in interior methods. The proposed algorithm uses sparse local projections in combination with a recursive solution of the small, dense projected system, exploiting the fact that space and time discretizations of the PDE governing the gas flow can be essentially decoupled. Even in the current preliminary implementation, the new algorithm clearly outperforms a control space method and the public domain multifrontal code MA27, especially on fine discretizations. Moreover, in contrast to MA27, our algorithm has accurately predictable (and generally lower) storage requirements. Because of the currently dense implementation of local factorizations, significant potential for improvements remains. As it turns out, more than 70% of the factors (over 60% of the total memory) are zero entries even without sparsity pivoting. It is straightforward to reduce the memory to roughly 40%, just by storing the factors of F_t^y differently. Further substantial savings in runtime and memory are to be expected from a genuine sparse factorization of the local constraint matrices F_t^y. Ultimately this may even include graph-based pivoting selection to exploit the static structure of a given network.

REFERENCES

[1] E. ANDERSON, Z. BAI, C. BISCHOF, J. DEMMEL, J. DONGARRA, J. DU CROZ, A. GREENBAUM, S. HAMMARLING, A. MCKENNEY, S. OSTROUCHOV, AND D. SORENSEN, *LAPACK Users' Guide*, SIAM, Philadelphia, PA, 1992. `http://www.netlib.org/lapack/`.

[2] H. G. BOCK AND K. J. PLITT, *A multiple shooting algorithm for direct solution of optimal control problems*, in Proc. 9th IFAC World Congress, Budapest, Hungary, 1984, J. Gertler, ed., vol. IX, Oxford, UK, 1985, Pergamon Press, pp. 242–247.

[3] E. A. BOYD, L. R. SCOTT, AND S. WU, *Evaluating the quality of pipeline optimization algorithms*, in 29th Annual Meeting, Pipeline Simulation Interest Group, 1997. Paper 9709.

[4] J. BURGSCHWEIGER, B. GNÄDIG, AND M. C. STEINBACH, *Optimization models for operative planning in drinking water networks*. In preparation.

[5] J. BURGSCHWEIGER, B. GNÄDIG, AND M. C. STEINBACH, *Nonlinear programming techniques for operative planning in large drinking water networks*. In preparation.

[6] R. G. CARTER, *Compressor station optimization: Computational accuracy and speed*, in 28th Annual Meeting, Pipeline Simulation Interest Group, 1996. Paper 9605.

[7] ———, *Pipeline optimization: Dynamic Programming after 30 years*, in 30th Annual Meeting, Pipeline Simulation Interest Group, 1998. Paper 9803.

[8] R. G. CARTER, D. W. SCHROEDER, AND T. D. HARBICK, *Some causes and effect of discontinuities in modeling and optimizing gas transmission networks*, technical report, Stoner Associates, Carlisle, PA, USA, Apr. 1994.

[9] D. DE WOLF AND Y. SMEERS, *The gas transmission problem solved by an extension of the simplex algorithm*, Management Sci., 46 (2000), pp. 1454–1465.

[10] I. S. DUFF, *MA28—a set of FORTRAN subroutines for sparse unsymmetric linear equations*, Report AERE R8730, HMSO, London, 1977.

[11] I. S. DUFF AND J. K. REID, *Some design features of a sparse matrix code*, ACM Trans. Math. Software, (1979), pp. 18–35.

[12] ———, *The multifrontal solution of indefinite sparse symmetric linear equations*, ACM Trans. Math. Software, (1983), pp. 302–325.

[13] K. EHRHARDT AND M. C. STEINBACH, *Nonlinear optimization in gas networks*, ZIB Report ZR-03-46, Konrad-Zuse-Zentrum für Informationstechnik Berlin, 2003.

[14] K. EHRHARDT AND M. C. STEINBACH, *KKT systems in operative planning for gas distribution networks*, ZIB Report ZR-04-21, Konrad-Zuse-Zentrum für Informationstechnik Berlin, 2004.

[15] E. M. GERTZ, P. E. GILL, AND J. MUETHERING, *Users guide for SnadiOpt: a package adding automatic differentiation to SNOPT*, Technical Memorandum ANL/MCS-TM-245, Argonne National Labs, Jan. 2001.

[16] P. E. GILL, W. MURRAY, AND M. S. SAUNDERS, *SNOPT: An SQP algorithm for large-scale constrained optimization*, SIAM J. Optim., 12 (2002), pp. 979–1006.

[17] P. HACKLÄNDER, *Integrierte Betriebsplanung von Gasversorgungssystemen*, Verlag Mainz, Wissenschaftsverlag, Aachen, 2002.

[18] T. HEIDENREICH, *Linearisierung in transienten Optimierungsmodellen des Gastransports*, Diploma thesis, Universität Duisburg, 2002.

[19] T. JENÍČEK, *Steady-state optimization of gas transport*, in Proceedings of 2nd International Workshop SIMONE [29], pp. 26–38.

[20] T. JENÍČEK, J. KRÁLIK, J. ŠTĚRBA, Z. VOSTRÝ, AND J. ZÁVORKA, *Study to analyze the possibilities and features of an optimization system (optimum control system) to support the dispatching activities of Ruhrgas*. Vertrauliche Dokumentation, LIWACOM Informationstechnik GmbH Essen, 1993.

[21] J. KRÁLIK, *Compressor stations in SIMONE*, in Proceedings of 2nd International Workshop SIMONE [29], pp. 93–117.

[22] J. KRÁLIK, P. STIEGLER, Z. VOSTRÝ, AND J. ZÁVORKA, *Modeling the dynamics of flow in gas pipelines*, IEEE Trans. Syst., Man, Cybern., SMC-14 (1984), pp. 586–596.

[23] ———, *A universal dynamic simulation model of gas pipeline networks*, IEEE Trans. Syst., Man, Cybern., SMC-14 (1984), pp. 597–606.

[24] ———, *Dynamic Modeling of Large–Scale Networks with Application to Gas Distribution*, vol. 6 of Studies in Automation and Control, Elsevier Sci. Publ., New York, 1988.

[25] A. MARTIN AND M. MÖLLER, *Cutting planes for the optimization of gas networks*, Preprint 2253, Fachbereich Mathematik, Technische Universität Darmstadt, 2002.

[26] M. MÖLLER, *Mixed Integer Models for The Optimisation of Gas Networks in the Stationary Case*, PhD thesis, Technische Universität Darmstadt, 2004.

[27] T. P. A. PERDICOÚLIS AND L. R. FLETCHER, *Decentralised dynamic optimisation of gas*, in 31st Annual Meeting, Pipeline Simulation Interest Group, 1999. Paper 9910.

[28] K. F. PRATT AND J. G. WILSON, *Optimization of the operation of gas transmission systems*, Transactions of the Institute of Measurement and Control, 6 (1984), pp. 261–269.

[29] *Proceedings of 2nd International Workshop SIMONE on Innovative Approaches to Modeling and Optimal Control of Large Scale Pipeline Networks*, Prague, 1993.

[30] D. RIST, *Dynamik realer Gase*, Springer, Berlin, 1996.

[31] E. SEKIRNJAK, *Mixed integer optimization for gas transmission and distribution systems*. Vortragsmanuskript, INFORMS-Meeting, Seattle, Oct. 1998.

[32] ———, *Transiente Technische Optimierung (TTO-Prototyp)*. Vertrauliche Dokumentation, PSI AG, Berlin, 1999.

[33] M. C. STEINBACH, *Hierarchical sparsity in multistage convex stochastic programs*, in Stochastic Optimization: Algorithms and Applications, S. P. Uryasev and P. M. Pardalos, eds., Dordrecht, The Netherlands, 2001, Kluwer Academic Publishers, pp. 385–410.

[34] ———, *Tree-sparse convex programs*, Math. Methods Oper. Res., 56 (2002), pp. 347–376.

[35] Z. VOSTRÝ, *Transient optimization of gas transport and distribution*, in Proceedings of 2nd International Workshop SIMONE [29], pp. 53–62.

[36] M. WESTPHALEN, *Anwendungen der stochastischen Optimierung im Stromhandel und Gastransport*, Cuvillier Verlag, Göttingen, 2004.

[37] J. F. WILKINSON, D. V. HOLLIDAY, E. H. BATEY, AND K. W. HANNAH, *Transient Flow in Natural Gas Transmission Systems*, American Gas Association, 1964.

[38] P. J. WONG AND R. E. LARSON, *Optimization of tree-structured natural-gas transmission networks*, J. Math. Anal. Appl., 24 (1968), pp. 613–626.

[39] S. WRIGHT, M. SOMANI, AND C. DITZEL, *Compressor station optimization*, in 30th Annual Meeting, Pipeline Simulation Interest Group, 1998. Paper 9805.

[40] J. ZÁWORKA, *Project SIMONE—Achievements and running development*, in Proceedings of 2nd International Workshop SIMONE [29], pp. 1–24.

MARC C. STEINBACH, KONRAD-ZUSE-ZENTRUM FÜR INFORMATIONSTECHNIK BERLIN, DEPARTMENT OPTIMIZATION, TAKUSTR. 7, 14195 BERLIN-DAHLEM, GERMANY

E-mail address: `steinbach@zib.de`

URL: `http://www.zib.de/steinbach`