2024 IEEE 63rd Conference on Decision and Control (CDC)
December 16-19, 2024. MiCo, Milan, Italy

Differentiable Simulator For Dynamic & Stochastic Optimal Gas & Power Flows

Criston Hyett[1], Laurent Pagnier[1], Jean Alisse[2], Igal Goldshtein[2],
Lilah Saban[2], Robert Ferrando[1], and Michael Chertkov[1]

[1]Program in Applied Mathematics & Department of Mathematics, University of Arizona, Tucson, AZ, USA
[2]Noga, The Israel Independent System Operator, Haifa, Israel

***Abstract*— In many power systems, particularly those isolated from larger intercontinental grids, reliance on natural gas is crucial. This dependence becomes particularly critical during periods of volatility or scarcity in renewable energy sources, further complicated by unpredictable consumption trends. To ensure the uninterrupted operation of these isolated gas-grid systems, innovative and efficient management strategies are essential. This paper investigates the complexities of achieving synchronized, dynamic, and stochastic optimization for autonomous transmission-level gas-grid infrastructures. We introduce a novel methodology grounded in differentiable programming, which synergizes symbolic programming, a conservative numerical method for solving gas-flow partial differential equations, and automated sensitivity analysis powered by SciML/Julia. Our methodology refines the co-optimization landscape for gas-grid systems by grounding gas dynamics in physics-adherent simulation. We demonstrate efficiency and precision of the methodology by solving a stochastic optimal gas flow problem, phrased on an open source model of Israel's gas grid model.**

I. Introduction & Background

The surge in renewable energy integration has heightened the variability in power demand, intensifying the fluctuations represented by the duck curve. Concurrently, the shift from coal to cleaner "bridge fuels" like natural gas places increased dependence on the gas infrastructure. This reliance extends beyond power generation to include transmission-level gas systems, which are also impacted by residential, commercial distribution, and exports. The disparate response times between gas and power networks – seconds for power systems versus hours for gas systems – add complexity to real-time and day-ahead coordination across these sectors. Earlier research, like that of [1] and [2], integrated gas dynamics into day-ahead planning through optimization models that simplified gas network constraints. More recent efforts have developed linear approximations for pipe segments to balance computational efficiency against model fidelity, aiding their incorporation into optimization frameworks [3]. Yet, efficiently and scalably addressing the full nonlinearity inherent in gas system dynamics, especially under stress and uncertainty, continues to pose a significant challenge.

The challenge we face is formally defined as solving a PDE-constrained optimization problem, which is schematically represented as:

$$\begin{aligned} &\min_{\{u^{(s)}(t),q(t)\}} \sum_{s\in\mathcal{S}} \int_0^T C^{(s)}(u^{(s)}(t),q(t))\,dt, \\ &\text{s.t. } \forall s,\ \forall t\in[0,T]:\ \frac{du^{(s)}(t)}{dt} = g(u^{(s)}(t),q(t)), \end{aligned} \tag{1}$$

where $u^{(s)}(t)$ denotes the time-evolving state space associated with scenario, $s \in \mathcal{S}$ and $q(t)$ stands for the time-evolving but scenario-independent control degrees of freedom. The space $\mathcal{S}$ represents uncertainty associated, e.g., with stochastic fluctuation in demand and renewable generation. The term $C^{(s)}(u^{(s)}(t), q(t))$ denotes the cumulative cost; in what follows we choose a multi-objective encapsulating the discrepancy between aggregated energy generation and demand, operational costs of gas generators, and pressure constraints at the gas-grid nodes. Note that we allow the cost to depend on the sample; we will use this generality to cover the case of uncertain demands later in the paper. For a given sample, $u^{(s)}(t)$ represents the spatiotemporal gas flows, gas densities, and, indirectly via the gas equation of state, pressures over the gas-grid. $q(t)$ are controlled boundary conditions, specifying gas supply/extraction at the various nodes of the gas-grid where gas supply/generators are positioned. The equation $\frac{du^{(s)}(t)}{dt} = g(u^{(s)}(t), q(t))$ enforces the gas-flow equations; such that for each scenario s, $u^{(s)}(t)$ is a solution to the Euler equations describing the gas flow, subject to the controlled boundary conditions $q(t)$. A detailed explanation is provided in Section II.

In this paper, we propose a novel approach to solving Eq. (1), aiming to enhance the fidelity of gas accounting in short-term to day-ahead planning of power generation in a computationally efficient manner. Our solution crafts a differentiable simulator by leveraging the principles of differentiable programming (DP) [4], combined with an efficient explicit staggered-grid method [5], symbolic programming and the robust capabilities of the SciML sensitivity ecosystem [6], [7]. As we delve further, it will become evident that our approach adeptly addresses the intertwined challenges of nonlinearity, dimensionality, and stochastic modeling.

In the proposed framework, DP facilitates the calculation of gradients by seamlessly solving the gas-flow PDE across a network. This is realized by auto-generating the corresponding adjoint equations, providing flexibility in formulating the

forward pass. The approach not only supports sensitivity analysis but, with a judicious selection of algorithms, proficiently manages scalability issues in parameter spaces, all while preserving the intricate nonlinear dynamics.

In this manuscript, we leverage recent efforts[8] in symbolic computing to decouple a model specification (i.e., PDE, boundary conditions and numerical method) and its implementation (compiled code). Historically, the developer has needed to implement both the numerical method and the code representation, worrying about low-level details like parallelism, memory access, etc. These low-level details often compete with correctness of the implementation in the developer's attention. ModelingToolkit.jl is a julia package that transforms an ODE specification to compiled code - compiling and optimizing according to the target hardware (e.g. CPU/GPU). In our work, we discretize the PDE symbolically, then stitch the resulting systems of ODEs into a single system - similar to [9] but with the added detail of network topology. This system is then compiled into executable code. This approach ensures the code exactly implements the model, abstracting away from low-level implementation details. Additionally, this weakens the coupling between component pieces - allowing future users to extend this modular methodology directly, e.g. implementing compressors simply by implementing a new network component type, and writing equations describing its interaction with incident pipes.

Motivated by the everyday operational challenges characteristic of Israel's power system, as expounded in [10] and its associated references, we design and solve a dynamic, stochastic problem that integrates power and gas flows over an operational timeframe ranging from several hours to an entire day. The example provided demonstrates the following unique aspects of the system:

(a) Limited availability or operational restrictions of gas compressors;
(b) Notable fluctuations in renewable resources and power loads, with curtailment being inadmissible under the normal operational paradigms assumed in this research;
(c) An intentionally over-engineered power system, ensuring power lines remain within thermal boundaries during standard operations.

Note critically that while the example presented in the following contains gas network specializations (e.g., (a)), because of the generality of symbolic programming and automatic differentiation, the proposed methodology is not restricted to these simplifying assumptions on the gas network.

The remainder of the manuscript is structured as follows: In Section II, we elucidate our gas modeling methodology, elaborate our fundamental optimization problem, and delineate our strategy for its resolution. Experimental results for a representative regional gas network are presented in Section III. Finally, the manuscript culminates with conclusions and suggested future directions in Section IV.

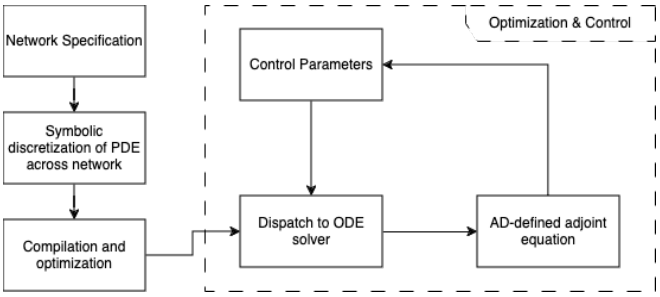

Fig. 1. Schematic of methodology

II. Methodology

A. Solving PDE Constrained Optimization

In this Section, we elucidate our strategy to address Eq. (1). Essentially, two predominant methodologies emerge for tackling the PDE-constrained optimization challenge:

1) **Constraint Matrix Encoding:** The prevailing methodology (demonstrated in [1][11][12] among others); the gas network PDEs are integrated into a constraint matrix that grows as discretization becomes finer. The resulting approximation to the constrained optimization in Eq. (1) can then be fed into any optimizer compatible with equality constraints. A notable merit of this approach is its flexibility in harnessing advanced optimization techniques and the ability to consider all timepoints simultaneously. However, the methodology grapples with potential pitfalls such as the emergence of unphysical solutions, failure to locate a feasible point (especially as the constraint matrix grows), non-adherence to constraints during intermediary timeframes, and the curse of dimensionality. To elaborate on the emergence of unphysical solutions, we demonstrate in Appendix IV-A), the resulting adjoint equation may be a poor approximation to the truth - so that even finding an optimal point may not guarantee a physical solution.
2) **Differentiable Programming:** This strategy (to the authors' knowledge never deployed in this field) leverages the continuous adjoint method (Appendix IV-A) to calculate gradients with respect to control parameters *through the PDE solver*. This method ensures the PDE solution u remains physically valid throughout the optimization process. Further, u converges to a well-defined solution as the grid undergoes refinement ($\Delta x, \Delta t \to 0$), which cannot be guaranteed for the constraint-matrix approach. However, challenges arise in generalizing the calculation of gradients through the PDE solver, dimensionality of the discretized PDE growing as the grid refines, and induced temporal computational complexity due to the Courant-Friedrichs-Lewy (CFL) condition for hyperbolic PDEs[13].

In the present study, we adopt the second approach and implement it as outlined in Fig. (1). We confront the aforementioned challenges by

1) **Using symbolic programming to discretize the PDE, and pre-compute coupling conditions at nodes.** This eliminates unnecessary memory accesses during simulation, ensures compatibility across network topologies, and creates an auto-differentiation friendly system of ODEs. Further, efficient memory access and the composability of julia enable easy parallelization across threads, processors, or accelerators[7].
2) **Using a conservative, method-of-lines discretization to allow for temporal integration using high-order, strong-stability preserving numerical integrators** to mitigate temporal computational complexity induced by the hyperbolic structure of the PDE.
3) **Leveraging modern, source-to-source Automatic Differentiation (AD) tools to automatically define and solve the corresponding adjoint equation**; allowing for freedom in expanding the network component library, and favorable scaling when computing gradients of high-dimensional parameterizations.[4]

Collectively, our proposed methodology bridges the gap between low-fidelity, optimization-centric 'constraint matrix' methods suited for long-term planning, and the demand for a physics-based tool, tailored for medium-term to real-time planning and analysis; essential for operational coordination between power and gas utilities.

B. Gas-Flow Equations

We begin by discussing the dynamics of a single pipe. The governing partial differential equations (PDEs) for the Gas Flow (GF), describing the dynamics of density $\rho(x,t)$ and mass-flux $\phi(x,t)$ along the pipe coordinate x with respect to time t, are provided as follows [14],[15],[16]:

$$\partial_t \rho + \partial_x \phi = 0, \tag{2}$$

$$\partial_t \phi + \partial_x p = -\frac{\lambda}{2D}\frac{\phi|\phi|}{\rho}, \tag{3}$$

where λ is the Darcy-Weisbach friction factor.

These equations are valid under the assumption that the gas velocity is much smaller than the speed of sound, c_s, in the gas ($\phi/\rho \ll c_s$). This is a reasonable approximation for the typical flows we consider, see e.g. [14].

To provide a complete description, it is necessary to relate the pressure $p(x,t)$ and density $\rho(x,t)$ using an equation of state:

$$p = Z(\rho, T)RT\rho, \tag{4}$$

where $Z(\rho,T)$ denotes the compressibility factor. For clarity, we adopt the ideal gas law to model the equation of state, where $Z(\rho,T)RT$ is replaced by a constant, c_s^2, with c_s representing the speed of sound in the gas. Notably, there are more accurate models available (e.g., CNGA [17]), and the methodology we present here is agnostic to the specific choice of model.

The system of Eqs. (2,3,4) is also supplemented by the boundary conditions, for example given profile of injection/consumption, at both ends of the pipe of length l,

$$q(0,t) \text{ and } q(L,t). \tag{5}$$

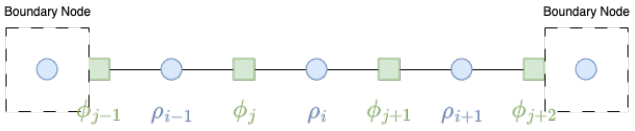

Fig. 2. Schematic of spatially staggered grid with variable locations

To extend the described equations from a single pipe to a network, the boundary condtions (5) need to appropriately couple pipe boundaries together, depending on the network topology. This is explained with the numerics in the next section.

C. Explicit Numerical Method for the Forward Path

To solve Eqs. (2,3,4) for a single pipe and their network generalizations, we employ a conservative, explicit, staggered spatial grid. This approach is inspired by the staggered-grid (in space and time) method of Gyrya & Zlotnik [5], but with modification consisted in rejecting the staggering in time.

To motivate this seemingly subtle modification from prior methods, consider that the computational complexity in simulating this PDE over interested timescales is dominated by the temporal complexity resulting from long time integration using timesteps compliant with the CFL condition. The CFL condition is a global condition dependent on the maximum local wave-speed across the network - *a priori* one can only upper bound this maximum using the speed of sound, leading to a very restrictive and costly CFL condition. Rejecting the staggering in time allows for deploying the method-of-lines discretization introduced below, to high-order, *adaptive*, strong-stability preserving ODE integrators - effectively allowing the ODE integrator to discover the (time-varying) maximum wave-speed. In practice, for nominal flows, this allows our timesteps to increase substantially beyond what is required by a speed-of-sound derived CFL - mitigating temporal computational complexity while retaining accuracy.

As illustrated in Fig. (2), we let the staggered grids for ρ and ϕ be denoted by x_i, x_j respectively, such that $x_j = x_i + \Delta x/2$. Futher, denote $\rho_i^n := \rho(x_i, t_n)$ and $\phi_j^n := \phi(x_j, t_n)$. Then we take centered differences on the staggered grid of Eqs. (2,3) to obtain

$$\frac{d\rho_i^n}{dt} = -\frac{1}{\Delta x}\left(\phi_j^n - \phi_{j-1}^n\right), \tag{6}$$

$$\frac{d\phi_j^n}{dt} = -\left(\frac{p_{i+1}^n - p_i^n}{\Delta x} + \frac{\lambda}{2D}\frac{\phi_j^n|\phi_j^n|}{\rho_j^n}\right). \tag{7}$$

Here $\rho_j^n \approx \frac{1}{2}(\rho_{i-1}^n + \rho_i^n)$. As we are interested in integration in day-ahead planning of energy generation, we control Dirichlet boundary conditions on nodal mass flows $\{q_i\}_{i\in\text{nodes}}$ (relating to generated power through heat-rate curves). These boundaries are resolved according to the numerical method using a boundary discretization shown in Fig. (3). The density updates for these junctions are evaluated

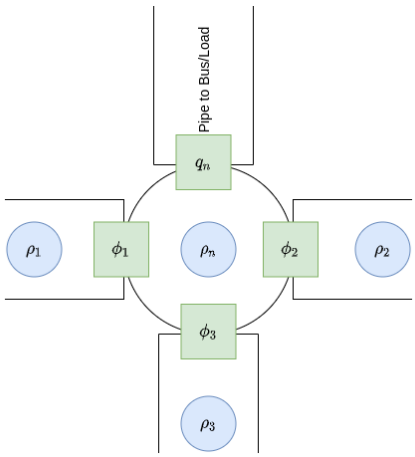

Fig. 3. An example of a 3-pipe junction discretization, with a fictitious pipe setting the nodal flow boundary condition.

using conservation of mass at boundary node ℓ

$$\frac{d\rho_\ell^n}{dt}\sum_{k\in\partial\ell}\left(\frac{\Delta x}{2}\right)S_{kl} = q_\ell + \sum_{k\in\partial\ell}\text{sgn}_{kl}S_{kl}\phi_{kl}^n(x=x_\ell), \tag{8}$$

where S_{kl} is the cross-section area of pipe from node k to node l, and sgn_{kl} keeps track of the directionality of the mass flux. ρ_{kl}, ϕ_{kl} denote the l-side boundary values of density and mass flux for the pipe from node k to node l. $\phi_{kl}^n(x=x_\ell)$ is approximated by a second-order, one-sided stencil. After solving for the density at the node, the flux update at the ends of the pipes can proceed using the momentum equation (7).

D. Optimization Formulation: Cost Function

In our pursuit to devise a scalable framework that aptly accommodates optimization challenges akin to the archetype presented in Eq. (1), we pivot our attention to a paradigmatic problem: the minimization of an integrated objective spanning time and evaluated under the cloak of uncertainty. This uncertainty, in our presentation reflected through scenarios $s \in \mathcal{S}$, pertains to uncertain energy demand $D^{(s)}(t)$, e.g., resulting from variable renewable generation. The time interval $t \in [0, T]$ typically encapsulates a pre-established performance window.

Our control parameters, symbolized by nodal flows $q(t)$, permit adjustments within our dynamic and stochastic optimization

$$\min_{(q(t)|t\in[0,T])}\sum_{s\in\mathcal{S}}\int_0^T C^{(s)}(u^{(s)}(t), q(t))\,dt. \tag{9}$$

Here, $u^{(s)}(t) := \{\rho^{(s)}(x,t), \phi^{(s)}(x,t)\}_{\forall x\in\text{nodes \& pipes}}$ denote the solution of the PDE defined by Eqs. (6-8), with nodal flows (injections/consumptions) $q(t)$; and the specific per-time t and per-scenario s cost is expanded as:

$$C^{(s)}(u^{(s)}(t), q(t)) = \alpha\left(D^{(s)}(t) - \sum_{i\in\text{nodes}} G_i(q_i(t))\right)^2 + \beta\sum_{i\in\text{nodes}} E_i(q_i(t)) + \gamma\sum_{x\in\text{nodes}} V(p^{(s)}(x,t)), \tag{10}$$

constrained by the gas-flow PDEs and associated boundary conditions over gas-grid network detailed earlier.

The first term in Eq. (10) aims to minimize the cumulative mismatch between energy demand $D^{(s)}(t)$ and the sum of generation at each node i and at each moment of time t, $G_i(q_i(t))$, with $q_i(t)$ representing nodal flows (our control variable). $G_i(q_i(t))$ is a so-called heat rate curve, mapping mass flow (in kg/s) to power production (in MW). Here the assumption is that any residual mismatch, if not optimal, can

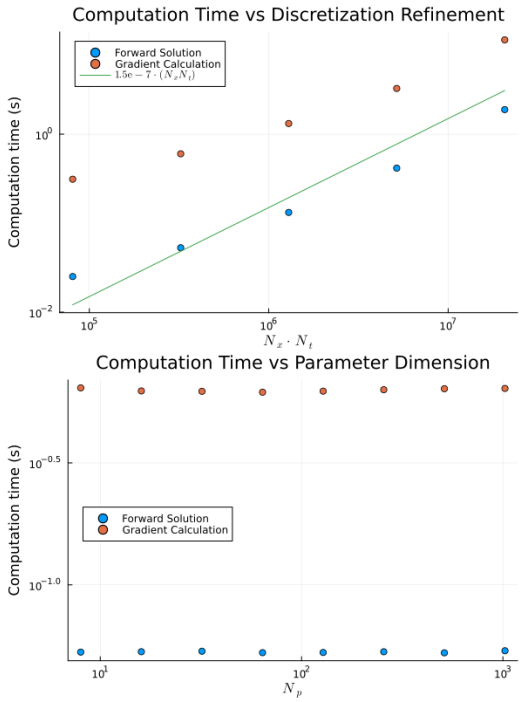

Fig. 4. Computational complexity scaling for the forward and adjoint calculations, (top) as a function of dimensionality of discretized PDE, and (bottom) as a function of parameter dimension.

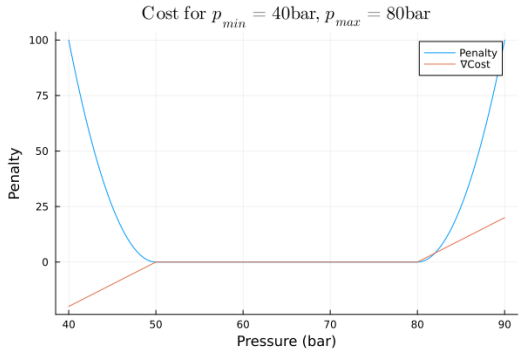

Fig. 5. Quasi-quadratic penalty for violating pressure constraints, with $40-80$ bar shown here, but configurable on a per-node basis.

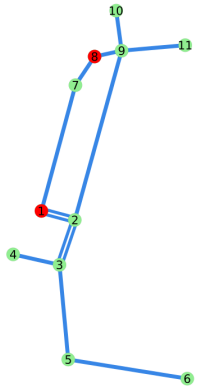

Fig. 6. 11-node network diagram, with supply nodes (injection into the network) in red, demand nodes (consumption from the network) in green.

be adjusted by either shedding demand or introducing a generation reserve, at a certain cost. The second term in Eq (10), $E_i(q_i(t))$, represents the cost of operating power generator run on gas and located at the node i at the gas withdrawal rate $q_i(t)$. The third term in Eq. (10), $\sum_{x\in\text{nodes}} V(p^{(s)}(x,t))$, is chosen to be a quasi-quadratic cost to penalize pressure constraint violations across the network (refer to Fig. 5): with $p_{\min}(x)$ and $p_{\max}(x)$ denoting pre-set pressure boundaries at system nodes. The influence of the multi-objective cost C's components can be modulated using the hyperparameters α, β, and γ; found as coefficients to the individual terms in Eq[10].

E. Solving the optimization

To solve the optimization in Eq(9), we iterate over three steps: (i) given a guess of our control $q(t)$, solve the forward problem Eqs(6,7,8) for $u(t) = \{\rho(t), \phi(t)\}$; (ii) solve the adjoint ODE Eqs(15,16) backward in time for $\lambda(t)$; (iii) use the solutions $u(t), \lambda(t)$ to compute the update to our control, Eq(19). Derivation and further exposition can be found in Section IV. A complete description, including implementation details, as well as full consideration of application specific trade-offs of different sensitivity methods for differentiable programming can be found in [18].

III. Results

To exemplify the methodology, we solve an optimal gas flow problem phrased on a previously studied reduced model of Israel's gas grid [10]. The reduced model, shown in Fig. (6), has 11 nodes, with a total pipe length of approximately 550km. We minimize the objective in Eq. (9) over a time horizon of 10hrs, encompassing a morning ramp in energy use. The demand curves $D(t)$ are aggregated from publicly available data; the gas cost $G(q)$ is taken as a constant; the efficiency curves $E_i(q(t))$ take one of three constant values representing efficient, nominal, and inefficient turbines; and the pressure limits are set as $p_{min} =$ 60bar, $p_{max} =$ 80bar. We use a box-constrained Limited-memory Broyden–Fletcher–Goldfarb–Shanno (LBFGS) optimizer; the constraints enforcing max and min injection/consumption at each node. Each node has hourly flow-rate control parameters, so the dimension of the optimization space is $110 = 11\text{nodes} * 10\text{hrs}$.

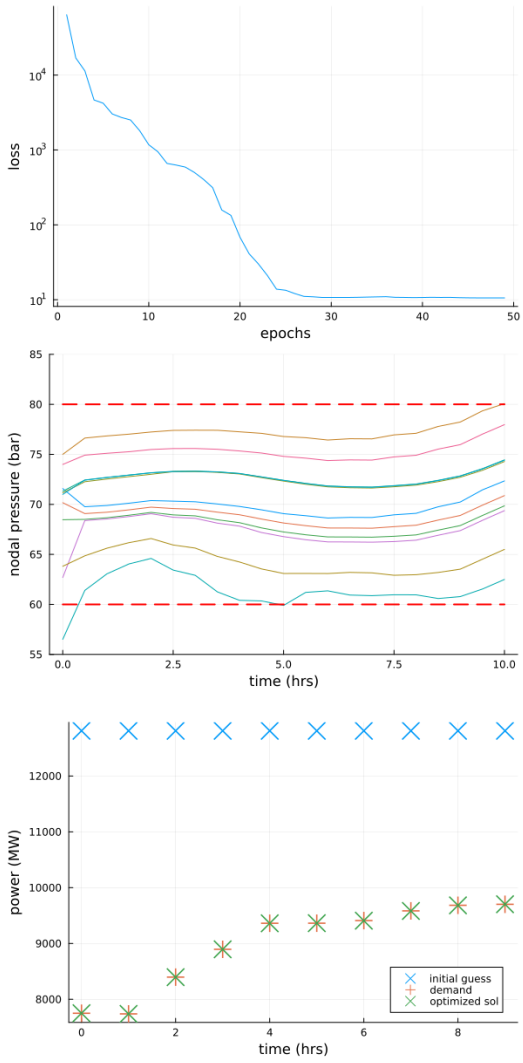

Fig. 7. Optimal gas flow results from an 11-node network, spanning $\approx$ 550km. The OGF is solved across a time horizon of 10 hours, using representative data of a morning ramp in the energy demand. (Top) Shows the quick convergence of LBGFS, despite the initial guess yielding a large penalty. (Middle) shows the evolution of pressure at each node using the optimized injections/withdrawals. Note that despite the dynamic initial conditions being outside the pressure window, the optimization quickly rectifies and holds all pressures in the acceptable range marked by the dashed lines. (Bottom) shows that we meet demand during the morning ramp, without waste.

The optimization results, considering a deterministic gas consumption profile, are depicted in Fig. 7. We observe an exponential decrease in the loss function, with the algorithm converging to a stable minimum within 30 iterations. System pressures remain within specified limits, except for the uncontrolled initial data, which does not contribute to the loss calculation. Notably, at node 6 – the system's lowest pressure point – there is a proactive pressure increase to accommodate the expected rise in consumption during the morning peak. This optimization strategy, navigating the

system's nonlinearities, proves crucial for operators in making informed real-time decisions. Impressively, the optimizer achieves exact demand fulfillment, despite starting very far from the optimum.

Subsequently, we applied the software to perform optimization under uncertain consumption patterns, assuming a normal distribution with a standard deviation equal to 5% of the current consumption level. This scenario aims to simulate the uniform response of all generators to the variability inherent in renewable energy sources. The findings, illustrated in Fig. (8), indicate the network's ability to achieve a reduced minimum. This improvement is attributed to effectively managing the pressures at nodes 7 and 8 (the nodes with the highest pressure) to remain below 80bar towards the latter part of the simulation, thereby mitigating less frequent low-pressure breaches at node 6, the node with the lowest pressure.

All code required to run and analyze these simulations as well as smaller test networks is available at `https://github.com/cmhyett/DiffGasNetworks`.

IV. Conclusion & Path Forward

The chief technical contribution of this manuscript is the integration of symbolic programming, automatic differentiation, and gas-flow PDE numerics to develop a more accurate, physics-based approach for addressing optimization and control issues in gas networks. This was demonstrated through the solution of a stochastic optimal gas flow problem. Our development efforts concentrated on:

1) **Efficiency:** We prioritized the forward solution's efficiency and the gradient computation's scalability using the adjoint method. Achievements in efficiency resulted from combining symbolic programming, high-performance ODE integrators, and advanced AD tools.
2) **Consistency:** For PDE-constrained optimization, especially in short-term or real-time planning, it is crucial to maintain precise physical solutions irrespective of grid refinement choices. The differentiable programming framework ensures consistent convergence within the physical domain and offers error assurances.
3) **Flexibility:** The methodology's design allows adaptable network and component configurations, supporting a range of applications from uncertainty quantification to inverse problems and data assimilation. By preserving symbolic representations until execution and employing Automatic Differentiation for derivative calculations, our approach facilitates selecting the most appropriate gradient computation method from available library of options, ensuring both application breadth and solution specificity.

We demonstrated our ability to utilize these characteristics to effectively solve an Optimal Gas Flow (OGF) problem under uncertainty, preserving essential system properties and the complete nonlinear dynamics of a representative regional gas network.

Future endeavors will focus on integrating this methodology into the broader scope of gas network optimization and control. Comparison and integration with optimization-centric (e.g., constraint-matrix and mixed integer optimizers) approaches is a promising avenue to more completely understand and mitigate the weaknesses of any individual approach. Additionally, such work provides the capability to bridge the OGF timescales of long-term planning, day-ahead scheduling, and real-time mitigation of intraday variations or emergencies.

Although our current model omits gas compressors due to specific characteristics of the Israel's system, the extension of this method to include compressors and valves in the network library is straightforward. Also important are representative equations of state - while the ideal gas law is convenient for exposition, utilization of this methodology in more extreme environments and at higher degrees of accuracy necessitate the adoption of more expressive equations of state. Both the addition of network components and enriching the equation of state are facilitated by the inherent generality of our method's design.

Moreover, while our Optimal Gas Flow (OGF) model under uncertainty effectively managed the expected cost, the realm of stochastic optimal control offers the capability to target more specific objectives, such as managing the higher moments of nodal pressures. Proactively addressing and planning for these uncommon occurrences within gas-grid system coordination remains a dynamic and critical field of research, see e.g., [19].

Finally, there have been significant theoretical advancements in multi-fidelity methods for outer-loop applications[20]. By integrating the methodology presented in this manuscript with constraint-matrix and machine learning techniques, one might develop a comprehensive multi-fidelity approach. This integration promises to be both efficient and versatile, offering high-fidelity solutions to the underlying PDEs across networks. Such an approach has the potential for broader applicability, extending beyond gas networks to encompass a wider spectrum of complex systems.

Appendices

A. Adjoint Method

In this appendix we present the adjoint (conjugated gradient) method used to compute gradient steps to resolve the minimization problem Eq. (1).

Let us drop index (s) in Eq. (1) to simplify notations (as if it would be just one sample), and restate Eq. (1) in the Lagrangian form:

$$\min_{(q(t),u(t)|t\in[0,T])} \max_{(\lambda(t)|t\in[0,T])} \mathcal{L}(q(t),u(t),\lambda(t)|t\in[0,T]), \tag{11}$$

$$\mathcal{L} \coloneqq \int_0^T \left(C(u(t),q(t)) + \lambda(t)^T\left(\dot{u}(t) - g(u(t),q(t))\right)\right) dt,$$

where $(\lambda(t)|t\in[0,T])$ is the functional Lagrangian multiplier introduced to enforce the underlying differential equation at $t\in[0,T]$:

$$\dot{u}(t) = g(u(t),q(t)). \tag{12}$$

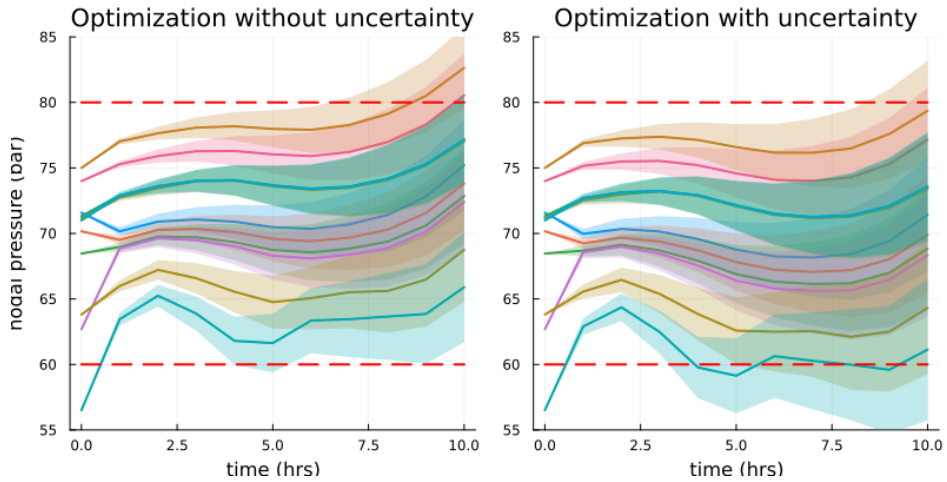

Fig. 8. Results of optimization under uncertainty. We initialize the optimization using the solution in Fig. 7, and then re-perform the optimization, taking the loss as the expectation over samples of noisy consumption as in Eq. (9). Plotted are the mean (solid line) plus/minus a standard deviation (opaque region) for selected nodes. This procedure illustrates the ability to tune for robustness against stochastic fluctuations in the power network.

Next we rewrite Eq. (11) splitting variation over $(q(t)|t \in [0,T])$ from variation over $(u(t)|t \in [0,T])$ and $(\lambda(t)|t \in [0,T])$:

$$\min_{(q(t)|t\in[0,T])} \tilde{\mathcal{L}}(q(t)|t \in [0,T]), \tag{13}$$

$$\tilde{\mathcal{L}}(q(t)|t \in [0,T]) := \min_{(u(t)|t\in[0,T])} \max_{(\lambda(t)|t\in[0,T])} \mathcal{L}. \tag{14}$$

Given the control function $(q(t)|t \in [0,T])$ our task becomes to find $\tilde{\mathcal{L}}(q(t)|t \in [0,T])$ which we do replacing minimization over $(u(t)|t \in [0,T])$ and maximization over $(\lambda(t)|t \in [0,T])$ by finding the stationary point of $\mathcal{L}$, that is solving respective Euler-Lagrange (EL) equations:

$$t \in]0,T[: \quad \frac{\delta\mathcal{L}}{\delta u(t)} = 0 \implies \partial_u C = \lambda^T \partial_u g + \frac{d}{dt}\lambda^T, \tag{15}$$

$$\frac{\delta\mathcal{L}}{\delta u(T)} = 0 \implies \lambda(T) = 0, \tag{16}$$

$$t \in [0,T]: \quad \frac{\delta\mathcal{L}}{\delta q(t)} = 0 \implies \partial_q C + \lambda^T \partial_q g = 0. \tag{17}$$

We solve the system of the EL by, first, running Eqs. (12) forward in time, and thus finding $(u(t)|t \in [0,T])$. Once $u(T)$ is found, and $\lambda(T)$ is fixed to zero, according to Eq. (16), we solve Eqs. (15) running them backwards in time to find $(\lambda(t)|t \in [0,T])$. Finally, we arrive at the following EL expression for $\tilde{\mathcal{L}}(q(t)|t \in [0,T])$

$$\tilde{\mathcal{L}} = \mathcal{L}(q(t), u(t), \lambda(t)|t \in [0,T])\Big|_{\text{Eqs. (12,15,16))}}. \tag{18}$$

What is now left is to optimize over $(q(t)|t \in [0,T])$ in Eq. (13). This last step is done iteratively – starting with a guess for $(q(t)|t \in [0,T])$ and updating it at each step according to, $t \in [0,T]$:

$$q^{(new)}(t) = q(t) - \eta \cdot \frac{\delta\tilde{\mathcal{L}}(q(t')|t' \in [0,T])}{\delta q(t)}, \tag{19}$$

where η is the so-called learning rate and at each iteration we update the functional $\tilde{\mathcal{L}}(q(t')|t' \in [0,T])$ according to the procedure described above and resulting in Eq. (18).

To avoid laborious encoding of derivative functions, the functional forms – such as partial derivatives of C and g – dependent on the current state $u(t)$ and control $q(t)$, are determined and evaluated using source-to-source Automatic Differentiation (AD).

We can now clearly point to a main difference between the constraint-matrix and differentiable programming approaches. When encoding the PDE into the constraint matrix, a choice of $\Delta x, \Delta t$ is made. This choice is likely far coarser than would be necessary to solve the PDE (otherwise the dimension of the equality constraints could easily reach the tens of millions for moderately sized networks), and results in particular choices t_i for which the solutions, $u(t_i)$, $\lambda(t_i)$ are available. This in turn reduces the fidelity of the approximation to Eq. (15), so that solutions $u(t)$ may or may not be physical, depending on the spectrum of the differential operator Eq. (15).

Contrast this with the differentiable programming approach, where the differential Eqs. (12,15) can be approximated at any time since it is truly solved continuously. Even when the forward solution $u(t)$ is not available at a needed time, one can use interpolation, or checkpointing (shown in Fig. 9) to obtain the needed value. This guarantees the gradient moves us along the solution manifold and never off of it.

B. *Differentiable Programming*

Source-to-source differentiation, particularly from Zygote.jl [21], is a transformational capability that allows reverse-mode automatic differentiation (AD) through programming language constructs – enabling optimized adjoint function evaluation without the need to write the derivatives by hand. This freedom ensures correctness, and allows for generality in construction of the forward pass [22].

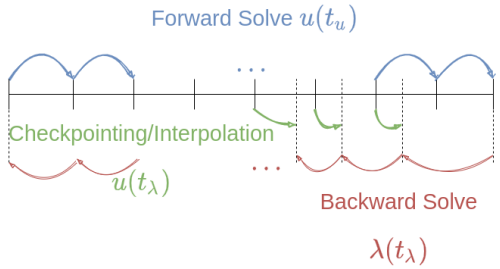

Fig. 9. Schematic of adjoint method implementation, where the adjoint ODE utilizes checkpointing to accelerate queries of the state u at required times.[23]

In order to compute the right hand side of Eq. (19), the adjoint ODE Eq. (15) is solved for $\lambda(t)$, and the term $\partial_q g$ is found via source-to-source reverse-mode AD. This method to compute the adjoint has computational cost that scales linearly with the forward pass, and with the number of parameters [23]. Notably, the sensitivity backend used evaluates the most efficient approach, and dispatches the appropriate sensitivity method (e.g., forward or adjoint sensitivities). We emphasize the adjoint method here, because of its preferable scaling with respect to high-dimensional parameterizations.

References

[1] A. Zlotnik, L. Roald, S. Backhaus, M. Chertkov, and G. Andersson, "Coordinated scheduling for interdependent electric power and natural gas infrastructures," *IEEE Transactions on Power Systems*, vol. 32, no. 1, pp. 600–610, 2016.

[2] G. Byeon and P. Van Hentenryck, "Unit commitment with gas network awareness," *IEEE Transactions on Power Systems*, vol. 35, no. 2, pp. 1327–1339, 2019.

[3] L. S. Baker, S. Shivakumar, D. Armbruster, R. B. Platte, and A. Zlotnik, "Linear system analysis and optimal control of natural gas dynamics in pipeline networks," 2023.

[4] M. Innes, A. Edelman, K. Fischer, C. Rackauckas, E. Saba, V. B. Shah, and W. Tebbutt, "A differentiable programming system to bridge machine learning and scientific computing," *arXiv preprint arXiv:1907.07587*, 2019.

[5] V. Gyrya and A. Zlotnik, "An explicit staggered-grid method for numerical simulation of large-scale natural gas pipeline networks," *Applied Mathematical Modelling*, vol. 65, pp. 34–51, 2019.

[6] C. Rackauckas, Y. Ma, J. Martensen, C. Warner, K. Zubov, R. Supekar, D. Skinner, and A. Ramadhan, "Universal differential equations for scientific machine learning," *arXiv preprint arXiv:2001.04385*, 2020.

[7] C. Rackauckas and Q. Nie, "DifferentialEquations.jl–a performant and feature-rich ecosystem for solving differential equations in Julia," *Journal of Open Research Software*, vol. 5, no. 1, 2017.

[8] Y. Ma, S. Gowda, R. Anantharaman, C. Laughman, V. Shah, and C. Rackauckas, "Modelingtoolkit: A composable graph transformation system for equation-based modeling," 2021.

[9] A. W. Jones, C. Hyett, C. Rackauckas, W. Trust, and C. Z. I. U. States), "MethodOfLines.jl - Automatic finite difference PDE discretization and solving with Julia SciML," May 2024. [Online]. Available: https://doi.org/10.5281/zenodo.11186853

[10] C. Hyett, L. Pagnier, J. Alisse, L. Sabban, I. Goldshtein, and M. Chertkov, "Control of line pack in natural gas system: Balancing limited resources under uncertainty," in *PSIG Annual Meeting*. PSIG, 2023, pp. PSIG–2314.

[11] S. R. Kazi, K. Sundar, S. Srinivasan, and A. Zlotnik, "Modeling and optimization of steady flow of natural gas and hydrogen mixtures in pipeline networks," *International Journal of Hydrogen Energy*, vol. 54, pp. 14–24, 2024.

[12] K. Hoppmann-Baum, F. Hennings, R. Lenz, U. Gotzes, N. Heinecke, K. Spreckelsen, and T. Koch, "Optimal operation of transient gas transport networks," *Optimization and Engineering*, vol. 22, no. 2, pp. 735–781, 2021.

[13] M. Brio, G. M. Webb, and A. R. Zakharian, *Numerical time-dependent partial differential equations for scientists and engineers*. Academic Press, 2010.

[14] A. Osiadacz, "Simulation of transient gas flows in networks," *International Journal for Numerical Methods in Fluids*, vol. 4, no. 1, pp. 13–24, Jan. 1984. [Online]. Available: https://onlinelibrary.wiley.com/doi/10.1002/fld.1650040103

[15] M. C. Steinbach, "On PDE solution in transient optimization of gas networks," *Journal of Computational and Applied Mathematics*, vol. 203, no. 2, pp. 345–361, Jun. 2007. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0377042706002263

[16] M. H. Chaudhry, *Applied hydraulic transients*. Springer, 2014, vol. 415.

[17] E. S. Menon, *Gas Pipeline Hydraulics*. CRC Press, 2005.

[18] F. Sapienza, J. Bolibar, F. Schäfer, B. Groenke, A. Pal, V. Boussange, P. Heimbach, G. Hooker, F. Pérez, P.-O. Persson *et al.*, "Differentiable programming for differential equations: A review," *arXiv preprint arXiv:2406.09699*, 2024.

[19] S. Tokareva, A. Zlotnik, and V. Gyrya, "Stochastic finite volume method for uncertainty quantification of transient flow in gas pipeline networks," *Applied Mathematical Modelling*, vol. 125, pp. 66–84, 2024.

[20] B. Peherstorfer, K. Willcox, and M. Gunzburger, "Survey of multifidelity methods in uncertainty propagation, inference, and optimization," *SIAM Review*, vol. 60, no. 3, pp. 550–591, 2018. [Online]. Available: https://doi.org/10.1137/16M1082469

[21] M. Innes, "Don't unroll adjoint: Differentiating ssa-form programs," 2019.

[22] C. Rackauckas, Sciemon, J. Vaverka, B. S. Zhu, V. L., A. Strouwen, D. P. Sanders, G. Sterpu, J. Ling, P. E. Catach, P. Monticone, W. Dey, V. Churavy, A. Edelman, A. Haslam, A. Lenail, A. Kaushal, C. Laforte, C. Wang, F. Cucchietti, K. Bhogaonker, L. Milechin, F. C. White, M. Payne, S. Schaub, S. Fu, V. Meijer, W. Kirchgässner, and anand jain, "Sciml/scimlbook: v1.1," Nov. 2022. [Online]. Available: https://doi.org/10.5281/zenodo.7347643

[23] Y. Ma, V. Dixit, M. J. Innes, X. Guo, and C. Rackauckas, "A comparison of automatic differentiation and continuous sensitivity analysis for derivatives of differential equation solutions," in *2021 IEEE High Performance Extreme Computing Conference (HPEC)*, 2021, pp. 1–9.