SPE-203954-MS

An Optimization-Based Facility Network Solver for Well Allocation in Reservoir Simulation

K. Wiegand, Y. Zaretskiy, K. Mukundakrishnan, and L. Patacchini, Stone Ridge Technology

This paper was prepared for presentation at the SPE Reservoir Simulation Conference, available on-demand, 26 October 2021 – 25 January 2022. The official proceedings were published online 19 October 2021.

Abstract

When coupling reservoir simulators to surface network solvers, an often used strategy is to perform a rule or priority-driven allocation based on individual well and group constraints, augmented by back-pressure constraints computed periodically by the network solver. The allocation algorithm uses an iteration that applies well-established heuristics in a sequential manner until all constraints are met. The rationale for this approach is simply to maximize performance and simulation throughput; one of its drawbacks is that the computed allocation may not be feasible with respect to the overall network balance, especially in cases where not all wells can be choked individually. In the work presented here, the authors integrate the well allocation process into the network flow solver, in the form of an optimization engine, to ensure that the solution conforms to the network rate and pressure balance equations. Results for three stand-alone test cases are discussed.

Introduction

Field development plans and mid/long term reservoir management decisions, especially for conventional fields, are usually evaluated using numerical reservoir simulation. The models may include coupling to surface production or injection networks, in order to simulate a coherent physical object up to a well-defined boundary, which could for example be a field separator with known pressure or an injection module with known capacity.

The integration between subsurface and surface simulations can be implemented in a fully implicit manner, see for example the proposed formulations by Coats et al. (2004) and Cao et al. (2015), which in principle guarantees stability and yields smooth production and injection profiles. The implicit treatment however requires a monolithic approach for the solution of the combined network and reservoir equations, bearing a cost.

An alternative is for the network model to balance surface flow streams, determining well rates based on inflow performance relationhips (IPRs) computed by the reservoir simulator's well model. The foundation of such approach is the observation that surface and subsurface flow timescales are very different, i.e., the network can be balanced considering essentially frozen conditions in the near wellbore regions. The idea, which traces back to Hepguler et al. (1997), Ghorayeb et al. (2003) and Kosmala et al. (2003), provides the

possibility to couple different software components, a flexibility which has been deeply exploited (Ghorayeb et al. 2005; Rotondi et al. 2008). In the network-IPR coupling logic, the coupling location can be chosen to be either down hole or at the wellhead. The reservoir and network models can be coupled periodically (e.g., monthly), at every time-step, or at every Newton iteration. The tighter the coupling, the higher the computational effort, eventually limiting the size and complexity of networks being solved.

A first approach to well control is to implement the allocation strategy into the reservoir simulator's code, using a field scheduler, field manager, or field controller depending on naming preferences. Typically, the controller asks the network solver to evaluate the unconstrained network solution based on IPRs computed by the reservoir well model; the unconstrained well pressures and rates obtained from the network solution are then used to provide bounds/potential values to the allocation process which can be rule-based, priority-based, or even optimization-based (Guyaguler et al. 2008). While this may lead to sub-optimal solutions, it is nevertheless very efficient and useful for reservoir engineers (Ghorayeb et al. 2005).

It is important to note that the above approach applies to both IPR-based and fully implicit coupling, although the lack of common software makes comparison and benchmark between fully implicit coupling and IPR-based coupling a challenge (see for example Patacchini et al. (2016) and Su et al. (2016) for descriptions of fully implicit and IPR-based coupling of the reservoir models and the injection system of the same field using different commercial solutions, or also Darche et al. (2019)). The approach is schematized in Fig. 1.

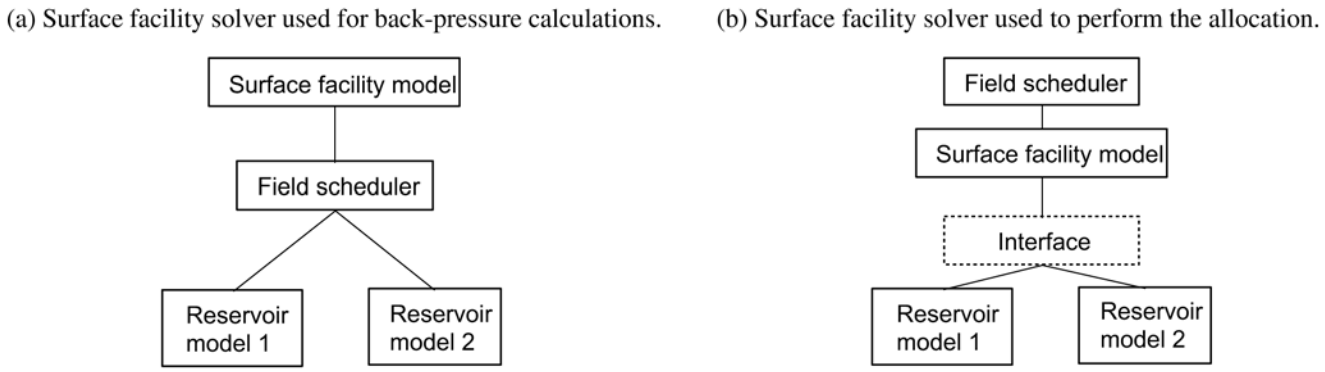

Figure 1—(a) coupling of (multiple) reservoir models to a surface network model used for back-pressure calculations, where the field scheduler performs the well allocation. (b) coupling of (multiple) reservoir models to a surface network model where allocation is performed by the network solver itself, following objectives dictated by the field scheduler.

Alternatively, the field scheduler may exploit an optimisation capability of the network flow solver package (Rotondi et al. 2008; Darche et al. 2019), to whom the allocation task is delegated. A reference example of this approach is provided by Davidson and Beckner (2003), where sequential quadratic programming (SQP) is used to maximize an objective function under network constraints.

In this publication, we present the development of a gradient-based optimisation capability inside a Facility Network Solver (FNS). FNS was developed as a network flow solver to balance pressure and flow rates in production and injection networks according to well IPRs and a fixed rate or pressure condition at the separator/terminal node. Flow-line pressure drops are computed using Vertical Lift Performance (VLP) tables, generated by a third-party tool. FNS is intended to work either as a stand-alone executable, or integrated with a reservoir simulator in form of a library.

The new gradient-based optimisation feature aims at increasing the benefit of FNS for reservoir engineers by providing an alternative to simulator built-in heuristics and potential/deliverability based balancing logic when optimizing hydrocarbon production under constraints.

The paper is organized as follows. After a simplified description of the network solver formulation, we provide a simple example illustrating the added value of a network optimizer, and follow with a description of the adjoint implementation and optimizer algorithm. Finally, we present three standalone (i.e., without coupling to a reservoir model) examples illustrating FNS capabilities.

Network Solver Formulation

Variables selection

FNS supports both injection and production networks, having either a simple tree structure or possibly bifurcated flow lines, as illustrated in Fig. 2 for the production case.

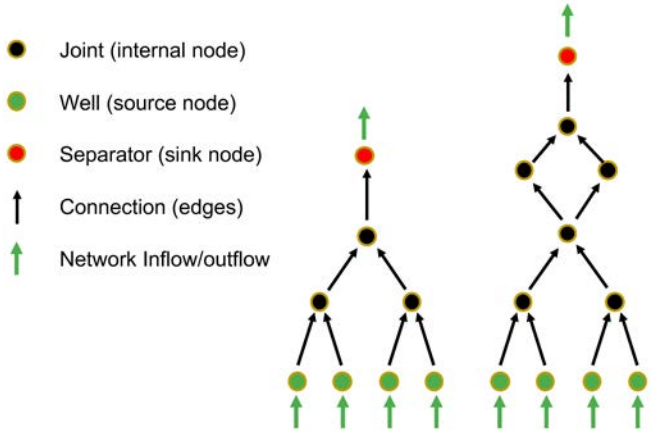

Figure 2—Schematic example of two production networks supported by FNS. Left: the network has a simple tree structure. Right: the network has a bifurcation.

Node-based equations for tree structures will consistently use the indices (i) to refer to the node itself, (k) to refer to the single downstream node, and j to refer to (each of) the upstream nodes, as illustrated in Fig. 3. The (W) index will refer to a well node, whose treatment is particular.

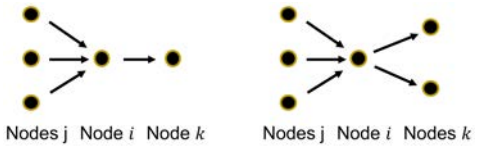

Figure 3—Definition of indexes (i), (k) and (j) for node-based equations. Left: the network has a simple tree structure. Right: the network has a bifurcation.

When the network has a simple tree structure, the variable set reduces to a single pressure per well and rate controlled separator/terminal node, and to a set of component rates and the pressure at joint nodes. Flow equations for bifurcated streams are slightly more complicated and require an additional unknown for the edges (connections) of the graph, at least in situations where multiple out-flowing edges are present at a node.

The focus of this paper being the network optimizer, we will here succinctly present the basic structure of the nodes equations for networks with a tree structure. A possible formulation supporting bifurcations has been presented by Cao et al. (2015).

Well nodes

Well nodes use a single pressure balance equation with the well pressure as primary unknown. Said pressure can either be the bottom-hole pressure, in which case the first edge past the well node would represent the well tubing, or the wellhead pressure.

Assuming either a simple choke or a hydraulic correlation for computing the pressure drop across the outgoing flow line, the pressure equation for the well node (i) is one of

$$0 = p_k - p_W - \Delta p_{(\text{choke})}, \tag{1a}$$

$$0 = p_W - H\left(p_k, F_{(\text{sep})}\left(\mathbf{U}_W\left(p_W\right)\right)\right), \tag{1b}$$

where the first equation is used in case of the simple choke, and the second equation is used in case of a hydraulic pressure drop model. Chokes by convention have a negative Δp in direction of the flow and details of the hydraulics (such as dependence on phase ratios, etc.) are hidden inside the definition of the function H.

Well component rates U_W are a function of the pressure unknown p_W, and are provided to FNS via IPR (inflow performance relationship) tables. In case of coupling to a reservoir simulator, these are readily computed by the well solve.

Hydraulic pressure drop models, often referred to as VFP (vertical flow performance) curves even when applied to horizontal network pipes, are usually tabulated versus phase flow rates at standard or stock-tank conditions. The function $F_{(\text{sep})}$ takes a vector of component rates and passes it through a separator or pseudo-separator to obtain standard or stock-tank rates.

If the computed well pressure lies outside the valid range given by the IPR table, the user-supplied choke or the well hydraulics is augmented by an additional choke that is computed by FNS. In the example above, the equation suing well hydraulics changes to

$$0 = p_W - H\left(p_k, F_{(\text{sep})}\left(\mathbf{U}_W\left(p_W\right)\right)\right) + \Delta p_{(\text{choke})}, \tag{1c}$$

whereas equation 1a remains unchanged, but the value of $\Delta_{p_{(\text{choke})}}$ is being adjusted. As long as the additional choke is required, that is, the well pressure without the choke lies outside the IPR table range, FNS switches the primary variable of the well node to the choke Δp.

Joint nodes

Joint nodes feature N component rate balance equations and the same type of constitutive pressure balance as for well nodes:

$$0 = \mathbf{U}_i - \sum_{j \in J} \mathbf{U}_j, \tag{1d}$$

$$0 = p_k - p_i - \Delta p_{(\text{choke})}, \tag{1e}$$

$$0 = p_i - H\left(p_k, F_{(\text{sep})}\left(\mathbf{U}_i\right)\right), \tag{1f}$$

where J is the set of upstream nodes. The primary unknowns are the outgoing component rates U_i and the pressure p_i. Separator/terminal nodes with fixed pressure use a dummy equation (so that we do not need to change the size of the Jacobian in case the type of boundary condition changes), while separator nodes with fixed rate solve for their pressure. If - for example - a phase rate (e.g., a total liquid rate) is specified, the balance equation at the separator/terminal node is

$$0 = q_\Gamma - \left[F_{(\text{sep})}\left(\sum_{j \in J} \mathbf{U}_j\right)\right]^T S, \tag{1g}$$

where $S \in \{0|1\}^{N^{phase}}$ depicts the phase combination ($N_{phase} = 3$). Component rate boundary equations are similar, but do not require the separator function $F_{(sep)}$.

The formulae shown above are those for non-bifurcated production streams and are straightforward in their application. Injection streams are similar in their formulation with the natural flow direction reversed.

Pressure Constraints

FNS allows the insertion of simple network pressure constraints at well and joint node locations. The constraints limit the allowed pressure range at dedicated node positions and FNS corrects violations by inserting an additional choke value and switching from the pressure unknown to the choke p(choke). Hence, the pressure equation for joint nodes resembles that of the well nodes.

The need for a network optimizer: Example for non-local Choke Placement

The problem

The variable substitution mechanism mentioned above works sufficiently well to handle simple pressure constraints that are either added by the user or imposed by IPR table ranges. Since the substitution is triggered by a local constraint violation at the node where the choke is going to be placed, the algorithm does not handle situations where a non-local choke placement is required, and does not consider any optimization. An example for a non-local choke placement is the following.

Consider the production network shown in Fig. 4 and the corresponding well IPR tables shown in Fig. 6. The usable overlapping operating range for the three wells is a rather narrow band from roughly 75 to 77.8 [Bar].

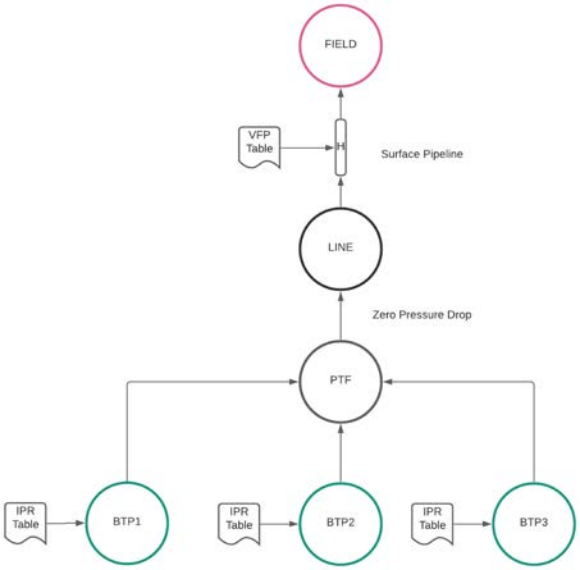

Figure 4—Production network for Choke Placement Example.

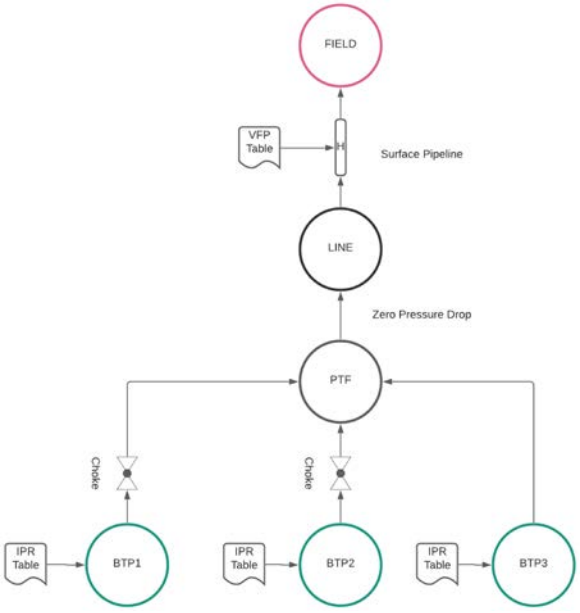

Figure 5—Production network for Choke Placement Example, after choke placement.

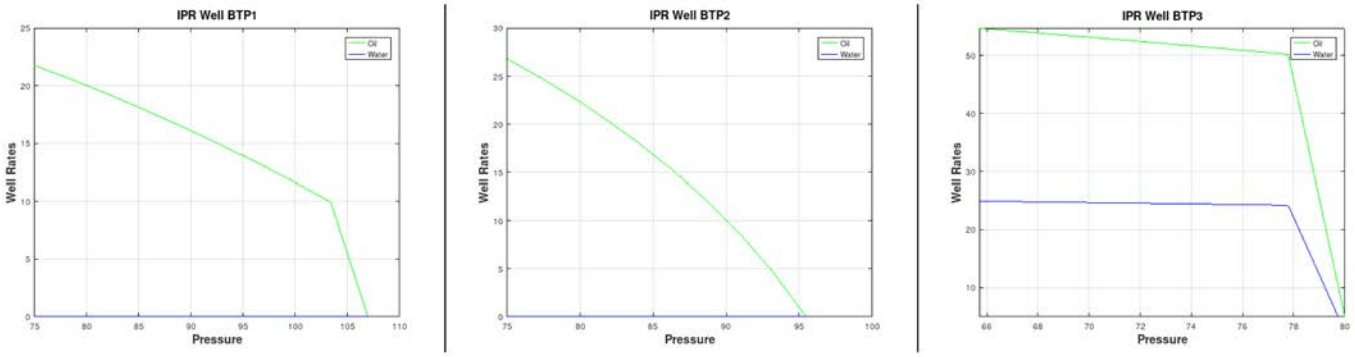

Figure 6—IPR Tables for the three wells of the network illustrated in Fig. 4. Left: well 1, Middle: well 2, Right:well 3. Pressure is in [Bar], and Rates are in [sm^3/D].

The data in Tab. 1 shows the FNS solution for two boundary pressures that are 0.1 [Bar] apart. The output has been partially truncated to focus on the important points.

Discussion

An increased boundary pressure (here: a shift of 0.1 [Bar]) reduces the rate and consequently also the pressure drop over the hydraulic flow-line. This leads to an increased pressure at nodes 'LINE' and 'PTF'. All three wells share the pressure of node 'PTF' via zero pressure drop flow-lines, however, the pressure is out of the range for IPR table 3. The constraint violation triggers the variable substitution mechanism but unless the user allows the placement of a positive choke that acts like a pump, no solution can be found where the well pressure matches the platform pressure. Hence, well BTP3 is shut in. The strong reduction of the field rate then requires chokes to be placed at the remaining two wells that produce the allowed maximum.

Table 1—FNS output for two boundary pressures 0.1 [Bar] apart, for the network of Fig. 4.

NODE NAME	ACTIVE Y/N/I	PRESSURE Bar	OIL RATE m^3/day	GAS RATE m^3/day
FIELD	YES	24.600	95.424	5703245.449
LINE	YES	77.772	95.424	5703245.449
PTF	YES	77.772	95.424	5703245.449
BTP1	YES	77.772	20.826	1244700.756
BTP2	YES	77.772	24.399	1458271.455
BTP3	YES	77.772	50.199	3000273.238

NODE NAME	ACTIVE Y/N/I	PRESSURE Bar	OIL RATE m^3/day	CHOKE PRESSURE Bar
FIELD	YES	24.700	48.587	0.000
LINE	YES	45.835	48.587	0.000
PTF	YES	45.835	48.587	0.000
BTP1	YES	75.000	21.773	-29.165
BTP2	YES	75.000	26.814	-29.165
BTP3	INOP	77.784	0.000	0.005

Table 2—FNS output for the optimized solution to the problem of Fig. 4, i.e., as per Fig. 5.

NODE NAME	ACTIVE Y/N/I	PRESSURE Bar	OIL RATE m^3/day	CHOKE PRESSURE Bar
FIELD	YES	25.000	95.058	0.000
LINE	YES	77.657	95.058	0.000
PTF	YES	77.657	95.058	0.000
BTP1	YES	78.175	20.688	-0.518
BTP2	YES	78.068	24.126	-0.411
BTP3	YES	77.657	50.244	0.005

Table 3—Unconstrained Node pressures and rates for test case #1.

NODE NAME	ACTIVE Y/N/I	PRESSURE Bar	OIL RATE m^3/day	GAS RATE m^3/day	WATER RATE m^3/day	CHOKE PRESSURE Bar
SEP	YES	30.000	12980.837	1298083.699	858.806	0.000
J4	YES	30.000	12980.837	1298083.699	858.806	0.000
J5	YES	39.919	3393.541	339354.067	16.968	0.000
W4	YES	166.606	3393.541	339354.067	16.968	0.000
J6	YES	39.924	2337.829	233782.934	116.891	0.000
W2	YES	146.622	2337.829	233782.934	116.891	0.000
J7	YES	30.000	7249.467	724946.698	724.947	0.000
J9	YES	40.466	7249.467	724946.698	724.947	0.000
J11	YES	52.152	5612.241	561224.083	561.224	0.000
J10	YES	52.152	5612.241	561224.083	561.224	0.000
W5	YES	254.388	5612.241	561224.083	561.224	0.000
W3	YES	137.256	1637.226	163722.616	163.723	0.000

A simple solution to avoid the well shut-in is to expand the range of the IPR table for well 3. However, the maximum pressure was most likely related to an economic well constraint (a minimum rate) and expanding the range for FNS will probably not help because the well and group control logic will most likely shut in the well for a violation of the economic constraint.

The solution to the problem is to actually place a choke at one of the first two wells (or both), even if there is no constraint violation of IPR tables 1 and 2. This will allow to operate well BTP3 at maximum pressure - producing 50 m^3 oil per day, while wells BTP1 and BTP2 produce only slightly less fluids than before. In order to compute this solution, the IPR table ranges need to be converted to state constraints for the optimizer so that FNS can compute a solution without the need to shut-in wells.

Figure 5 shows the non-local choke placements. They are non-local in the sense that the constraint violation occurred at node BTP3 whereas the chokes are placed at nodes BTP1 and BTP2.

The data in Tab. 2 shows the optimized solution for a boundary pressure of 25 [Bar]. In the rest of the paper, we explain some details of the optimization process.

Network Optimization

State and Control Variables

Choosing the right state and control variables for the optimizer is crucial. In an industrial environment, the operator is constrained by what can *actually* be controlled and/or measured. For instance, if the flow of several wells is co-mingled into a riser sitting on the sea floor, no individual well control may be possible. Or, individual valves for wells do exist but have become inoperable over time. Hence, control variables need to reflect controls that can be physically manipulated.

In our case these are operable chokes and pumps inside the network and gas lift rates where applicable. State variables are then all primary unknowns of the network solver and derived quantities. The former are the component rates and node pressures, while the latter are phase ratios and combined flows of well groups. For using the network flow solver inside the optimization loop, we make the following adjustments:

1. The flow solver is altered to execute in two modes: Normal Mode and Optimization Mode.
2. We use the normal mode to compute an initial solution for the network, including well chokes.
3. The normal mode is slightly altered to allow positive choke values (pumps) to avoid well shut-ins.
4. In optimization mode, IPR tables utilize the full physical range and constraints are handled by the optimizer.
5. In optimization mode, variable substitution is disabled, and chokes are supplied by the optimizer. From the viewpoint of the flow solver, these are external variables treated as constants in the regular flow equations - just like constant pressure drop connections.
6. Derivatives with respect to control variables are computed in optimization mode and placed into a secondary matrix. This matrix is rectangular by nature and degenerates into a vector in case we only use a single control variable.

If a well choke was necessary to be placed by FNS for the initial solution, but a choke is actually not allowed at the present position (because it needs to be moved further downstream or upstream), the optimizer keeps the choke as control variable. However, an equality constraint will be added to drive the value of the choke down to zero while increasing choke pressure drop(s) elsewhere.

Some necessary Notation

Only a few pieces of notation are required, which we introduce here.

- $y := y(x)$ is the vector of primary unknowns in the network flow solver and will be referred to as the vector of state variables. For example, the entries for a joint node in a black-oil model would be $\{U_g, U_o, U_w, p\}$, where 'g', 'o' and 'w' indicate the reference gas, oil and water components or stock-tank phases.
- For reasons that become clear shortly, the flow equations inside FNS are denoted by $c(y(x), x)$ or $c(y, x)|_{y=y(x)}$ depending on the context.
- The Jacobian matrix of partial derivatives inside the flow solver will be denoted by $c_y(y(x), x)$ or $c_y(y, x)|_{y=y(x)}$ depending on the context. We always consider the Jacobian from the *last* Newton iteration at which FNS converged to a solution.
- x is the vector of control variables and presently is restricted to choke values and gas lift rates. The input deck informs the optimizer *where* chokes are allowed to be placed and which hydraulic

pipe segments support variable gas lift rates. The actual number of control variables is determined inside FNS based on consistency checks and pre-existing chokes.

- $c_x \in \mathbb{R}_{n_y \times n_x}$, where n_x is the number of control variables, contains the derivatives of the flow equations with respect to the local control variables. This matrix is in general not square.

Control Variable Derivatives

Consider the following abstract optimization problem (Heinkenschloss 2008):

$$\min_x \ \widehat{\Psi}(x) := \Psi(y(x), x) \tag{2a}$$

where $y(x) \in \mathbb{R}^{n_y}$ is the solution to the state equation:

$$c(y, x) = 0, \tag{2b}$$

and where

$$\Psi : \mathbb{R}^{n_y+n_x} \to \mathbb{R}, \ c : \mathbb{R}^{n_y+n_x} \to \mathbb{R}^{n_y}. \tag{2c}$$

Under certain assumptions, the implicit function theorem states the existence of a differentiable function $y : \mathbb{R}^n x \to \mathbb{R}^{n_y}$ defined by $c(y(x), x) = 0$. Its derivative is the solution to the following equation:

$$c_y(y, x)\big|_{y=y(x)} \cdot y_x(x) = -c_x(y, x)\big|_{y=y(x)}, \tag{2d}$$

or short

$$y_x(x) = -c_y(y(x), x)^{-1} \cdot c_x(y(x), x). \tag{2e}$$

The derivative $y_x(x)$ is called the sensitivity of the state variables with respect to the control variables. The gradient of $\Psi(x)$, using the sensitivity, is now given by:

$$\nabla\widehat{\Psi}(x) = y_x(x)^T \nabla_y \Psi(y(x), x) + \nabla_x \Psi(y(x), x) \tag{2f}$$

$$= -c_x(y(x), x)^T \cdot c_y(y(x), x)^{-T} \cdot \nabla_y \Psi(y(x), x) + \nabla_x \Psi(y(x), x). \tag{2g}$$

If we define $\lambda(x) := -c_y(y(x), x)^{-T} \cdot \nabla_y \Psi(y(x), x)$, the gradient of Ψ can be written as:

$$\nabla\widehat{\Psi}(x) = \nabla_x \Psi(y(x), x) + c_x(y(x), x)^T \cdot \lambda(x), \tag{2h}$$

where $\lambda(x)$ is the solution to the *Adjoint* equation:

$$c_y(y(x), x)^T \cdot \lambda(x) = -\nabla_y \Psi(y(x), x). \tag{2i}$$

The above tells us that we can compute the gradient of the objective (and of any constraint function) by reusing the flow solvers Jacobian from the last Newton iteration of the converged network solution. This is of immense practical value given that a single choke placed randomly inside the network changes the pressures and rates almost everywhere in the presence of hydraulic flow lines. It allows derivative computations for the optimizer without the algorithm being aware of the network topology and the details of the flow equations. Although second order derivatives could be computed in a similar manner, we will not compute them because of the computational costs involved, and the fact that the flow solver itself does not have a need for them. We illustrate the process of computing the gradient in Fig. 7.

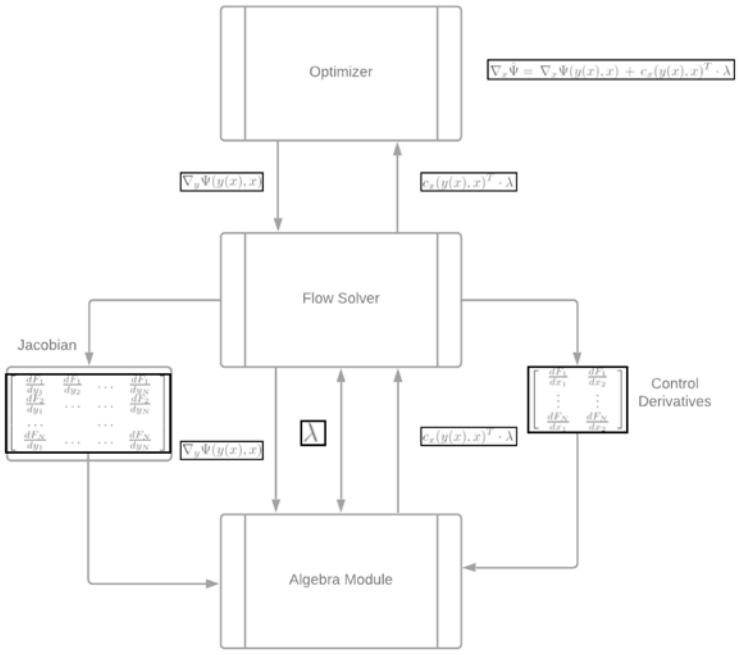

Figure 7—Gradient Computation via FNS Adjoint.

A Merit Function for Evaluation and Step Computation

Our optimization strategy is based on the following requirements:

1. It must be fast enough not to slow down the reservoir simulation when used in coupled mode.
2. It must be able to handle infeasible steps, however, the network equations solved by the flow solver need to be physically feasible.

Given the requirements above, we define a *merit* function that combines the objective and all state and control constraints into a single expression:

$$\hat{\Psi}^{\mu}(x) := \Psi^{\mu}(y(x), x) = f(y(x), x) + \frac{1}{\mu} \cdot \sum_{i=1}^{N_I} max\left(0, g_i\left(y(x), x\right)\right)^2 + \frac{1}{\mu} \cdot \sum_{i=1}^{N_E} h_i^2(y(x), x), \tag{3a}$$

where $f(y(x), x)$ denotes the objective function, and the g_i, h_i are the inequality and equality state and control constraints. The facility network flow equations enter the picture as active equality constraint, that is, we assume the constraint to be satisfied and use the computed derivatives for the adjoint computation. The design of the objective and the inequality constraints is such that the minimization of the merit function leads to the desired optimal result. Hence, we are interested in a solution to the minimization problem:

$$\min_x \hat{\Psi}^{\mu}(x), \tag{3b}$$

$$c(y(x), x) = 0. \tag{3c}$$

Since $c(y(x), x) = 0$ is always fulfilled, the above optimization problem is actually solved as an *unconstrained* minimization problem. In order to arrive at a practical algorithm, we solve problem 3b repeatedly for a sequence of smaller (positive) μ. This allows the optimization procedure to first make larger strides to reduce the objective, only to tighten the constraints more and more in subsequent iterations. Hence, we have an outer iteration over μ and an inner iteration to solve the unconstrained minimization problem for the given μ.

$$\lim_{\mu \to 0} \min_x \hat{\Psi}^{\mu}(x), \tag{3d}$$

$$c(y(x), x) = 0. \tag{3e}$$

Gradient Computation

We now turn our attention to computing the gradient with the help of the adjoint as this was demonstrated for the abstract problem above. Denote an arbitrary term of equation 3a by the generic function $c_i(y(x), x)$, and - to avoid misunderstandings about which derivatives are partial and which are not - define:

$$\hat{c}_i^{\mu}(x) := c_i^{\mu}(y(x), x) \tag{4a}$$

$$\nabla_x \hat{c}_i^{\mu}(x) = \nabla_x c_i^{\mu}(y(x), x) + \left[\frac{\partial c_i^{\mu}(y(x), x)}{\partial y} \cdot \frac{\partial y}{\partial x}\right]^T \tag{4b}$$

$$= \nabla_x c_i^{\mu}(y(x), x) + \left[\frac{\partial y}{\partial x}\right]^T \cdot \nabla_y c_i^{\mu}(y(x), x). \tag{4c}$$

Of course, we do not have the matrix $\left[\frac{\partial y}{\partial x}\right]$. But if we compute the solution to the adjoint equation

$$c_y(y(x), x)^T \cdot \lambda(x) = -\nabla_y c_i^{\mu}(y(x), x), \tag{4d}$$

the gradient $\nabla_x \hat{c}_i^{\mu}(x)$ can be written as:

$$\nabla_x \hat{c}_i^{\mu}(x) = \nabla_x c_i^{\mu}(y(x), x) + c_x(y(x), x)^T \cdot \lambda(x). \tag{4e}$$

Here, $c_y(y(x), x)$ and $c_x(y(x), x)$ are the Jacobian and the matrix of control derivatives from the flow solver. The computation above requires one linear solve to compute the vector of adjoint variables λ. Since our merit function lumps all derivatives together, we can achieve a major reduction in computational time by computing the adjoint and gradient for the sum of the penalty terms. For example, let

$$\hat{c}^{\mu}(x) := \sum_i c_i^{\mu}(y(x), x).$$

Then the adjoint equation becomes:

$$c_y\left(y\left(x\right), x\right)^T \cdot \lambda\left(x\right) = \sum_i \mu \cdot \nabla_y c_i\left(y\left(x\right), x\right), \tag{5a}$$

and the gradient is computed as:

$$\nabla_x \hat{c}\left(x\right) = -\sum_i \mu \cdot \nabla_x c_i\left(y\left(x\right), x\right) + c_x\left(y\left(x\right), x\right)^T \cdot \lambda\left(x\right). \tag{5b}$$

The adjoint and gradient computation for these "lumped" penalty terms effectively reduces the number of required linear solves to one, because all constraint terms and the objective can be combined this way.

Optimization Algorithm

Presently, we prefer to avoid a Hessian (or the approximation of it) and addressing issues of ill-conditioning and negative curvature, which need to be considered when developing a robust second order optimization code. Furthermore, second order methods require additional computational effort for computing the derivatives. Another issue is that we cannot expect to find a feasible starting point without significant effort. Our first version of the optimization procedure then uses a simple steepest descent based method based on the merit function's gradient, combined with a line search. The algorithm is outlined below.

Algorithm 1: Optimization Loop.

Result: Optimal feasible x
$k := 0$
$x_k \leftarrow x_{init}$
$\widehat{\Psi}^{\mu}_k := \widehat{\Psi}^{\mu}(x_k)$
$S_k := \{c^{\mu}_i : c^{\mu}_i(x_k) \geq 0\}$
$\mu = 0.1$
while $\mu > \mu_{min}$ **do**
  while $\neg optimal$ **do**
    $s_k := -\nabla\widehat{\Psi}^{\mu}(x_k)$
    $\alpha := \alpha_{init}$
    do
      $\alpha := \alpha \cdot 0.9$
      $\widehat{\Psi}^{\mu}_{k+1} := \widehat{\Psi}^{\mu}(x_k + \alpha \cdot s_k)$
      $S_{k+1} := \{c^{\mu}_i : c^{\mu}_i(x_k + \alpha \cdot s_k) \geq 0\}$
    while $(\widehat{\Psi}^{\mu}_{k+1} \geq \widehat{\Psi}^{\mu}_k \wedge |S_{k+1}| > |S_k| \wedge \alpha > \alpha_{min})$;
    if $\alpha > \alpha_{min}$ **then**
      $x_{k+1} := x_k + \alpha \cdot s_k$
    else
      break;
    end
    $\mu := \mu \times 0.5$
  end
end

In algorithm 1, $|S_k|$ denotes the number of active or violated constraints, and the condition in the do-while loop ensures that a step does not increase the cardinality of the set S_k. However, experiments have shown that relaxing this condition and actually allowing $|S_{k+1}| > |S_k|$ sometimes improves the convergence rate. The factors for decreasing μ and are values that work for most cases but may need adjustment based on case heuristics. For example, cases that are very sensitive to small pressure changes benefit from a larger factor (more closer to unity) when decreasing. In our first experiments, we only modify choke values and don't use gas-lift as additional control variable. Since the range of values for the choke is much more restricted, the initial step direction can be computed as $s_k := -\nabla\Psi^{\mu}(x_k)/\|\nabla\Psi^{\mu}(x_k)\|$ or a slightly scaled version of it. This brings the choke adjustment per inner iteration into a reasonable range for an initial value of close to unity.

Testing for Optimality

The choice of a simple steepest descent step deprives us of the mathematical tools to check for necessary and sufficient optimality conditions. Since $\Psi^{\mu}(x)$ contains the constraint terms, one would need "good" Lagrange multiplier estimates to at least check for first order optimality. Such estimates can be obtained by combining μ and the individual c_i in certain ways (depending on the kind of constraint), however, this approach strongly depends on the way the updates to x and μ are computed during inner and outer iterations. In practice, a simple algorithm like the one shown above will not be able to produce iterates such that either $-\mu/c^{\mu}_i(x_k)$ or $c^{\mu}_i(x_k)/\mu$ come even close to a good Lagrange multiplier estimate. Hence, the "test" for optimality in algorithm 1 above is to simply break the inner loop if the line search fails and continue the outer loop till μ reaches its minimum.

Regularization

Using any kind of optimization procedure to compute an optimal well allocation at discrete points in time faces the problem of well oscillations. A trivial example is a hand-crafted script that maintains plateau by

choking less economic wells, e.g., wells with high water cut. If we have two or more wells with nearly identical water cut, and if we only need a slight reduction in the total field rate to reach plateau, chances are that we will switch between wells in subsequent reduction loops. This is simply due to the fact that reduced wells - who are been given a break so to speak - often see a slight reduction in their water cut. If coning is present, even in light form, then we can expect that reducing the well's rate will improve the water cut. Hence, candidates for reduction switch between discrete invocations of the allocation procedure. Using a numerical optimization scheme does not eliminate the problem. The objective function, $f(y(x), x)$ in our example, will most certainly contain terms to penalize unwanted production and encourage the production of oil. As long as no wells are added or shut-in between calls to the optimizer at subsequent time steps, a simple regularization term that penalizes individual well variations is often sufficient. If the well count changes, if wells are worked over (that is, the open perforations or perforation parameters are changed during the simulation run), or if wells are shut-in or re-opened, a more complex penalty strategy is required that takes the overall effect of the changes into account. By providing a "built-in" optimization capability in FNS with a reservoir simulator, we hope to avoid well oscillations that sometimes can be observed when coupling an external network optimization tool with a reservoir simulator.

Numerical Experiments

Test case #1

Our first test case is the simple synthetic production network depicted in Fig. 8.

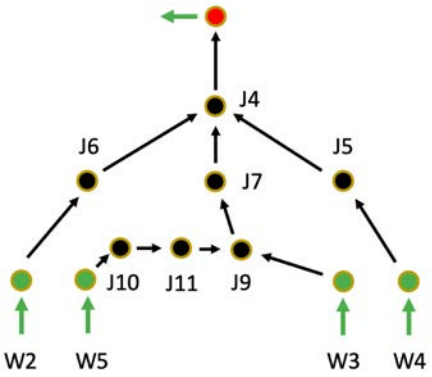

Figure 8—Production network for test case #1.

The wells have different oil/water ratios, with well W4 producing the most oil per m^3 of liquid and wells W3 and W5 having the largest water cut. Well W5 is the most productive well, supplying roughly half of the field oil rate in the unconstrained case.

Table 3 lists node pressures and liquid rates for the unconstrained field case, where wells are choked only by the network back-pressure.

For our optimizer test, we imposed a total liquid constraint of 10, 000 $[m^3/D]$ at node J4. Chokes were allowed at individual wells only, resulting in a total of 4 control variables for the optimizer. Since we need an objective to "pull" the solution towards a minimum, we associated a negative cost multiplier with the field oil production and a positive cost factor with the field water production. This is in line with traditional field reduction methods that need to know which wells are preferred economically.

We first show in Fig. 9 and Fig. 10 two plots illustrating how the field rates and merit function values evolve during the optimizer's iterations. The observable jumps in Fig. 10 are due to the outer iterations where we tighten μ.

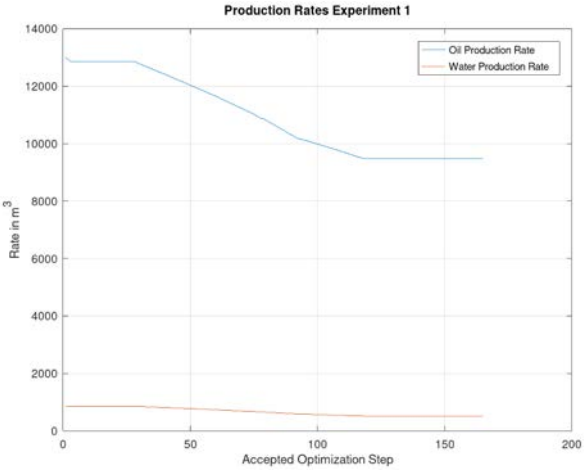

Figure 9—Oil and Water production rates (in [sm^3/D]) for test case #1.

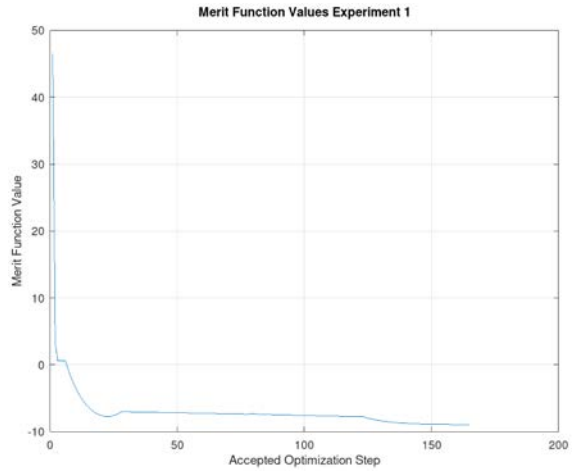

Figure 10—Evolution of the merit function with optimization step for test case #1.

We then report in Tab. 4 rounded solution values. The result is close to being optimal. The optimal oil rate for this case is 9, 490sm^3/D. The total run time on a high end laptop is approximately 180 [ms].

Test Case #2

Our second test case uses a small section of a deep-water flow network as shown in Fig. 11. Here again, the objective is to limit the liquid rate. Since the riser is very sensitive to the water cut in the transported fluid, we need the optimizer to reach the reduction goal by mostly reducing wells with high water rates.

Table 4—Node pressures and rates for optimized result - test case #1.

NODE NAME	ACTIVE Y/N/I	PRESSURE Bar	OIL RATE m^3/day	GAS RATE m^3/day	WATER RATE m^3/day	CHOKE PRESSURE Bar
SEP	YES	30.000	9489.266	948926.628	554.000	0.000
J4	YES	30.000	9489.266	948926.628	554.000	0.000
J5	YES	39.918	3311.889	331188.875	16.559	0.000
W4	YES	166.688	3311.889	331188.875	16.559	-2.021
J6	YES	39.911	1605.944	160594.402	80.297	0.000
W2	YES	153.941	1605.944	160594.402	80.297	-21.013
J7	YES	30.000	4571.434	457143.351	457.143	0.000
J9	YES	39.990	4571.434	457143.351	457.143	0.000
J11	YES	51.926	4406.829	440682.867	440.683	0.000
J10	YES	51.926	4406.829	440682.867	440.683	0.000
W5	YES	255.593	4406.829	440682.867	440.683	-38.262
W3	YES	166.708	164.605	16460.484	16.460	-40.891

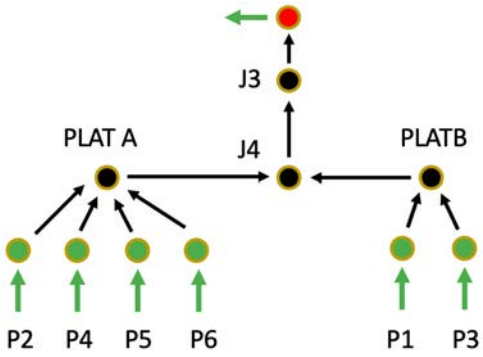

Figure 11—Production network for test case #2.

Table 5 shows the pressures and rates for the unconstrained case. The wells on the left side (PLAT A) of the diagram share the same IPR tables, and so do the wells on the right side (PLAT B). Our expectation is that only the wells on the left side will be chocked in order to meet the constraint of 150, 000[*stdb/D*] total liquid rate.

Table 5—Unconstrained Node pressures and rates for test case #2.

NODE NAME	ACTIVE Y/N/I	PRESSURE PSI	OIL RATE stdb/day	GAS RATE Mscf/day	WATER RATE stdb/day	CHOKE PRESSURE PSI
SEP-1	YES	500.000	94230.368	202761.437	85975.297	0.000
J3	YES	500.000	94230.368	202761.437	85975.297	0.000
J4	YES	1291.399	94230.368	202761.437	85975.297	0.000
PLAT_A	YES	1302.595	58764.258	90085.605	85975.295	0.000
P2	YES	1302.595	14691.064	22521.401	21493.824	0.000
WH2	YES	2901.278	14691.064	22521.401	21493.824	0.000
P4	YES	1302.595	14691.064	22521.401	21493.824	0.000
WH4	YES	2901.278	14691.064	22521.401	21493.824	0.000
P5	YES	1302.595	14691.064	22521.401	21493.824	0.000
WH5	YES	2901.278	14691.064	22521.401	21493.824	0.000
P6	YES	1302.595	14691.064	22521.401	21493.824	0.000
WH6	YES	2901.278	14691.064	22521.401	21493.824	0.000
PLAT_B	YES	1294.791	35466.110	112675.832	0.002	0.000
P1	YES	1294.791	17733.055	56337.916	0.001	0.000
WH1	YES	2220.041	17733.055	56337.916	0.001	0.000
P3	YES	1294.791	17733.055	56337.916	0.001	0.000
WH3	YES	2220.041	17733.055	56337.916	0.001	0.000

We first show in Fig. 12 and Fig. 13 two plots illustrating how the field rates and merit function values evolve during the optimizer's iterations. The observable jumps in Fig. 13 are due to the outer iterations where we tighten μ.

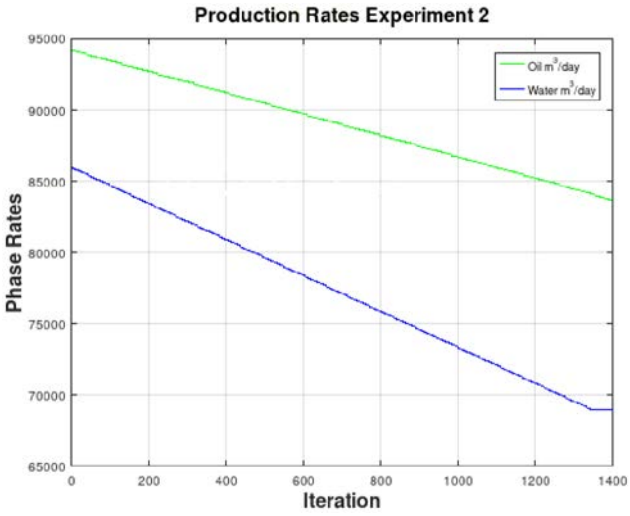

Figure 12—Oil and Water production rates (in [*stb/D*]) for test case #2.

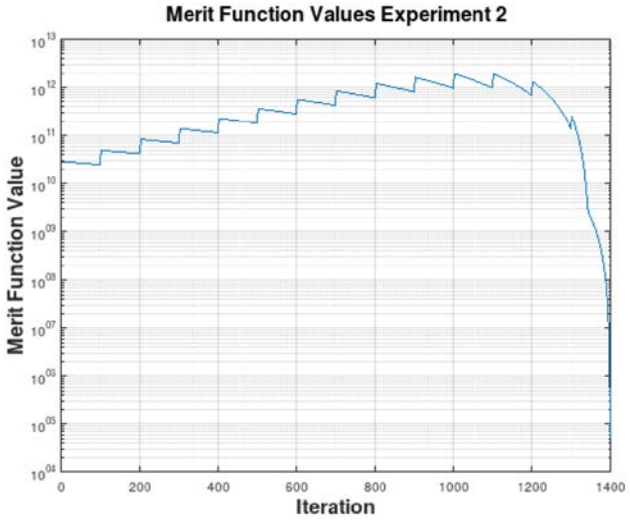

Figure 13—Evolution of the merit function with optimization step for test case #2.

We then report in Tab. 6 rounded optimization solutions. The fact that the rates drops in a linear fashion during the iterations (see Fig. 12) show that there is room for improvement, e.g. pursuing a more aggressive step computation.

Test Case #3

We consider the gathering network of a gas field as depicted in Fig. 14. The network uses flow-line bifurcation in two places to widen the flow path for the transport of gas from the manifolds to the gathering point at which we apply a boundary condition of ≈ 26 [Bar]. The well IPR tables are actually very narrow for this case, however, we selected a boundary pressure that doesn't require the application of chokes for the unconstrained case.

Table 6—Node pressures and rates for optimized Result - test case #2.

NODE NAME	ACTIVE Y/N/I	PRESSURE PSI	OIL RATE stdb/day	GAS RATE Mscf/day	WATER RATE stdb/day	CHOKE PRESSURE PSI
SEP-1	YES	500.000	83675.410	188303.099	68999.987	0.000
J3	YES	500.000	83675.410	188303.099	68999.987	0.000
J4	YES	1205.478	83675.410	188303.099	68999.987	0.000
PLAT_A	YES	1213.088	47161.604	72298.738	68999.985	0.000
P2	YES	1213.088	11790.401	18074.685	17249.996	0.000
WH2	YES	3315.658	11790.401	18074.685	17249.996	-672.643
P4	YES	1213.088	11790.401	18074.685	17249.996	0.000
WH4	YES	3315.658	11790.401	18074.685	17249.996	-672.643
P5	YES	1213.088	11790.401	18074.685	17249.996	0.000
WH5	YES	3315.658	11790.401	18074.685	17249.996	-672.643
P6	YES	1213.088	11790.401	18074.685	17249.996	0.000
WH6	YES	3315.658	11790.401	18074.685	17249.996	-672.643
PLAT_B	YES	1209.351	36513.806	116004.361	0.002	0.000
P1	YES	1209.351	18256.903	58002.181	0.001	0.000
WH1	YES	2152.690	18256.903	58002.181	0.001	-0.495
P3	YES	1209.351	18256.903	58002.181	0.001	0.000
WH3	YES	2152.690	18256.903	58002.181	0.001	-0.495

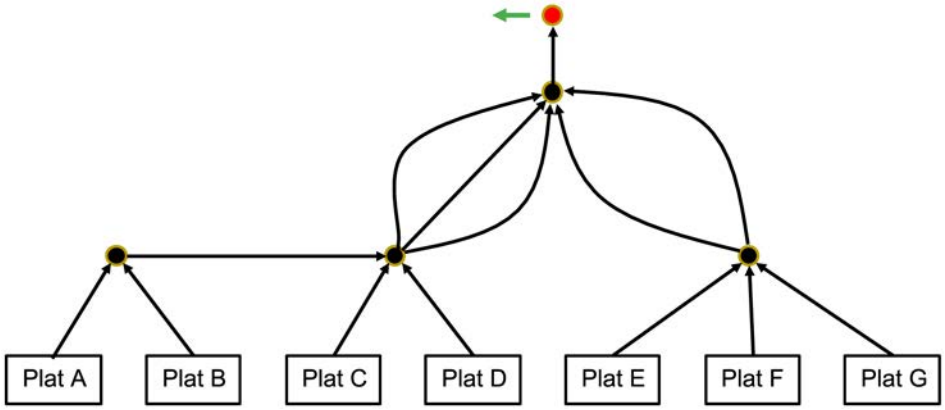

Figure 14—Gas gathering network of test case #3.

We apply a gas rate limit of 50 Million [m^3/D] and penalize high water content as before. The ratio between unconstrained and optimized rates for the different wells is illustrated in Fig. 15, where wells are ordered by increasing water-gas ratio (from top to bottom). Figure 16 shows shows the value of $\Delta_{p_{(\text{choke})}}$ proposed by the optimized for each well, vs. the well WGR (water-gas ratio) in the unconstrained scenario. It appears, as expected, that the trend is towards wells with higher WGR being choked more. The optimal solution however is scattered around the trendline, which would have been difficult to obtain using simple rule-based allocations available in reservoir simulators.

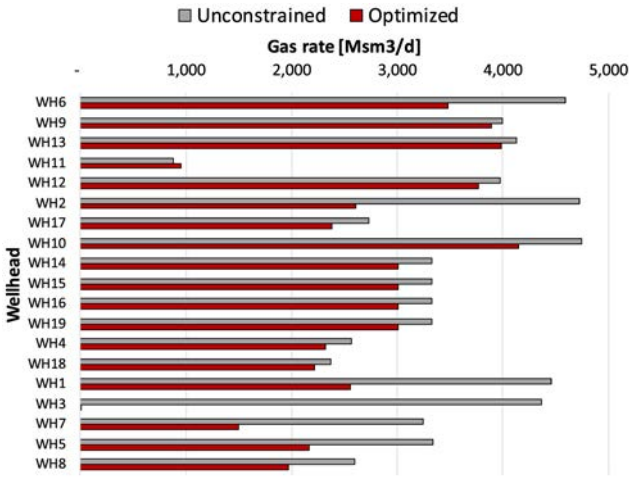

Figure 15—Unconstrained and optimized gas rates for test case #3.

We then show in Fig. 17 and Fig. 18 two plots illustrating how the field rates and merit function values evolve during the optimizer's iterations. Contrary to the test cases #1 and #2, here there are no observable jumps in the merit function.

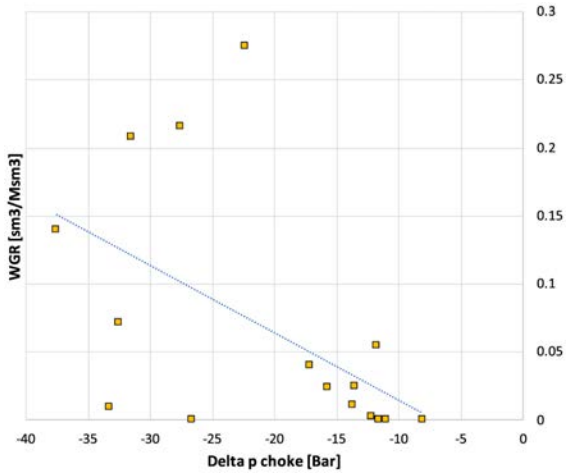

Figure 16—cross-plot of wellhead choke pressure drop vs. WGR for test case #3, after optimization.

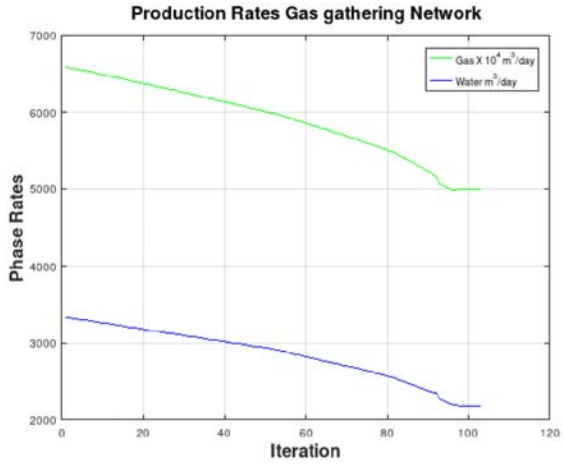

Figure 17—Gas and Water production rates for test case #3. Gas rate tends to its target (50 Million [*m*3/*D*]).

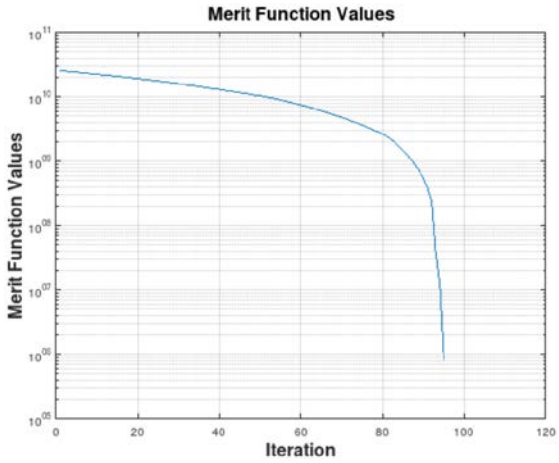

Figure 18—Evolution of the merit function with optimization step for test case #3.

Conclusions and Future Work

A simple first-order gradient-based optimization algorithm has been implemented in a facility network solver, designed to run either standalone, or be coupled to a reservoir simulator. The main advantage of this approach is the integration with the flow solver which simplifies the engineering workflow and allows an efficient way to compute derivatives using the Adjoint method.

The end-goal of this development is to enable coupled reservoir-network simulations where the well allocation is directly performed by the network solver by tuning the network configuration or choke/pump settings in order to achieve a given target under constraints, rather than having the reservoir simulator allocate rates based on rules or priorities subject to a network back-pressure.

I this paper, we presented the development of the optimization algorithm, tested on three selected standalone cases where well IPRs are computed by the reservoir simulator in a preprocessing step (i.e., represent a snapshot of reservoir conditions). The next step will be to apply the optimizer in coupled runs.

Acknowledgements

FNS is developed as part of a cooperative development agreement between Stone Ridge Technology and Eni S.p.A. The authors would in particular like to thank Federica Caresani, Alberto Pizzolato and Paola Panfili for their support in developing and testing FNS.

References

Cao, H., Samier, P., Kalunga, H., Detige, E., Obi, E., et al. 2015. A fully coupled network model, practical issues and comprehensive comparison with other integrated models on field cases. In SPE Reservoir Simulation Symposium. Society of Petroleum Engineers. SPE-173251-MS.

Coats, B., Fleming, G., Watts, J., Rame, M., Shiralkar, G., et al. 2004. A generalized wellbore and surface facility model, fully coupled to a reservoir simulator. *SPE Reservoir Evaluation & Engineering*, **7**(02):132–142.

Darche, G., Marmier, R., Samier, P., Bursaux, R., Guillonneau, N., Long, J., Kalunga, H., Zaydullin, R., Cao, H., et al. 2019. Integrated asset modeling of a deep-offshore subsea development using 2 complementary reservoir-surface coupling workflows. In SPE Reservoir Simulation Conference. Society of Petroleum Engineers. SPE-193929-MS.

Davidson, J. and Beckner, B. 2003. Integrated optimization for rate allocation in reservoir simulation. *speree* **6** (6): 426–432. SPE Reservoir Evaluation & Engineering. SPE-87309-PA.

Ghorayeb, K., Holmes, J., Torrens, R., Grewal, B., et al. 2003. A general purpose controller for coupling multiple reservoir simulations and surface facility networks. In SPE Reservoir Simulation Symposium. Society of Petroleum Engineers. SPE-79702-MS.

Ghorayeb, K., Holmes, J. A., Torrens, R., et al. 2005. Field planning using integrated surface/subsurface modeling. In SPE Middle East Oil and Gas Show and Conference. Society of Petroleum Engineers. SPE-92381-MS.

Guyaguler, B., Byer, T. J., et al. 2008. A new rate-allocation-optimization framework. *SPE Production & Operations*, **23**(04):448–457.

Heinkenschloss, M. 2008. Numerical solution of implicitly constrained optimization problems. Technical Report TR08-05, Department of Computational and Applied Mathematics, Rice University, Houston, TX 77005-1892. Available at http://www.caam.rice.edu/heinken/mh_publications.html.

Hepguler, G., Barua, S., Bard, W., et al. 1997. Integration of a field surface and production network with a reservoir simulator. *SPE Computer Applications*, **9**(03):88–92.

Kosmala, A., Aanonsen, S. I., Gajraj, A., Biran, V., Brusdal, K., Stokkenes, A., Torrens, R., et al. 2003. Coupling of a surface network with reservoir simulation. In SPE Annual Technical Conference and Exhibition. Society of Petroleum Engineers. SPE-84220-MS.

Patacchini, L., Bedewi, M., Farouk, M., et al. 2016. Fully implicit coupling of multiple reservoir models with common water injection facilities. In Abu Dhabi International Petroleum Exhibition & Conference. Society of Petroleum Engineers. SPE-183213-MS.

Rotondi, M., Cominelli, A., Di Giorgio, C., Rossi, R., Vignati, E., Carati, B., et al. 2008. The benefits of integrated asset modelling: lessons learned from field cases. In Europec/EAGE Conference and Exhibition. Society of Petroleum Engineers. SPE-113831-MS.

Su, S. J., Patacchini, L., Mohmed, F., Farouk, M., Ouzzane, D., Draoui, E., Torrens, R., Amoudruz, P., et al. 2016. Coupling production and injection systems with multiple reservoir models: A novel method of optimizing development strategies in a mature giant oilfield. In Abu Dhabi International Petroleum Exhibition & Conference. Society of Petroleum Engineers. SPE-183153-MS.