energies

Article

Electric Submersible Pump Lifted Oil Field: Basic Model for Control, and Comparison of Simulation Tools

Bernt Lie

Department of Electrical Engineering, Information Technology and Cybernetics, University of South-Eastern Norway, 3918 Porsgrunn, Norway; bernt.lie@usn.no

Abstract: Optimal operation of petroleum production is important in a transition from energy systems based on fossil fuel to sustainable systems. One sub-process in petroleum production deals with transport from the (subsea) well-bore to a topside separator. Good control design for such operation requires a dynamic model of the petroleum flow from the well-bore to the separator. Here, such a dynamic model is considered for liquid production (oil/water) using an electric submersible pump (ESP) to aid in counteracting gravity and friction forces. Based on an existing model used for industrial control design, one goal is to report a complete dynamic model in a single paper. Emphasis is put on dimensionless equipment models for the simple change of units, and the model is developed from physical laws for easy replacement of sub-models, if needed. All the necessary information (equations, parameters) for model implementation is provided, and two candidate equation-based modeling languages are selected and compared: Modelica and ModelingToolkit [MTK] for Julia. The simulation results are virtually identical for the two languages and make sense from physics; however, there is a minor discrepancy in one plot—likely caused by slight differences in accuracy in handling initialization in the implicit algebraic equations. The implementation structures of the model in Modelica and MTK are similar. Modelica is a mature and excellent modeling tool, handles large-scale models, and has tools for producing C code and integration with other tools. MTK is still in rapid development, supports more model types than Modelica, and is integrated in an eco-system with excellent support for control design, optimization, model fitting, and more. To illustrate the suitability of using the developed model for control design, a simple PI controller is designed within the eco-system of MTK/Julia.

Keywords: oil production; ESP lift; dimensionless model; dynamic model; simulation tool; Modelica; ModelingToolkit; Julia

check for updates

Citation: Lie, B. Electric Submersible Pump Lifted Oil Field: Basic Model for Control, and Comparison of Simulation Tools. *Energies* **2024**, *17*, 507. https://doi.org/10.3390/en17020507

Academic Editor: Konstantinos Kyprianidis

Received: 14 December 2023
Revised: 8 January 2024
Accepted: 16 January 2024
Published: 20 January 2024

1. Introduction

This paper builds on material in Lie [1]. Petroleum products have been key energy carriers for more than a century. The current focus on climate (https://sdgs.un.org/goals; accessed on 13 December 2023) implies a change towards sustainable energy carriers. To succeed in this change, a transition period from the use of fossil fuel is necessary. In the transition, improved operation of petroleum production through model-based optimal operation will be necessary. Petroleum production entails slow (reservoir; months) and fast (reservoir-to-separator; seconds) subsystems; this is a focus of the on-going research project "DigiWell" (DigiWell: see Funding). The vertical transport of petroleum from the oil well to the surface requires sufficient pressure to counteract gravitational and friction forces. If the oil-well heel pressure is insufficient for such transport, either (i) gas is injected in the vertical pipe to "blow" the petroleum fluids to the surface ("gas lifted"), or (ii) an electric submersible pump (ESP) is installed in the vertical pipe to increase the pressure ("ESP-lifted") sufficiently. Here, the focus is on the dynamics of transport from the reservoir formation to a surface manifold via an ESP, and further horizontal transport from the manifold to a separator.

Industrial simulation tools typically put the main emphasis on the dynamics of the *reservoir* (time constant: months) and use steady state models for the reservoir-to-surface transport. This emphasis is inadequate for daily operation and control. Here, a simplified, yet complete, dynamic model for oil transport from the reservoir to the separator is discussed. The model provides an understanding of the dynamic behavior of such systems, and is suitable for industrial control design, as well as for control and petroleum production studies. Emphasis is put on simple, yet stringent, model development, while avoiding *unit* complexities in the variables.

The authors of Sharma and Glemmestad [2] (see also Sharma [3]) provide a dynamic model of oil transport from the reservoir to the separator suitable for control design. In Binder et al. [4], an older model is discussed; other models are typically CFD models, which are too complex for control design.

Sharma's model considers a case with four vertical pipes from the oil reservoirs to a single manifold, with two horizontal pipes from the manifold to a single separator. Each vertical pipe has an ESP and a choke valve at a common manifold entrance; the pump speeds can be manipulated individually. The horizontal pipes have booster pumps to counteract friction effects. The original ESP model includes induction motors, but the dynamics of the pump actuator is fast, and is neglected in later work. The ESP model in Sharma and Glemmestad [2] is novel—the authors use a simple model of a booster pump, and use a valve model based on the ANSI/ISA S75.01 standard (http://integrated.cc/cse/ISA_750101_SPBd.pdf; accessed on 13 December 2023). The model in Sharma and Glemmestad [2] was re-structured and simplified in Lie [1], emphasizing dimensionless equipment models, thereby eliminating some of the complexity found in common industry models.

The dynamic model with ESP in Sharma and Glemmestad [2] is mainly relevant for the production of heavy oil. Several papers use this model in advanced industrial control studies, including Krishnamoorthy et al. [5], Santana et al. [6], Delou et al. [7].

Mixtures of liquid oil and water form an emulsion when stirred (e.g., in a multi-stage ESP). For such emulsions, the viscosity—and hence, the friction—varies dramatically with the water content (Justiniano and Romero [8]). Sharma and Glemmestad [2] assume an unrealistic linear viscosity dependence on the water fraction in pipe friction. Creation of an emulsion will likely modify the behavior of the ESP. Furthermore, real systems will contain some gas, which will also modify the behavior of the ESP (see, e.g., Zhu et al. [9–11]).

Although control-relevant models of ESP-aided oil production from bore-well heel to top-side separator are not new, it is difficult to find presentations with complete information provided in a concise way in a single location. This paper aims to fill that gap, while also demonstrating the advantage of using dimensionless equipment models. As a consequence, the paper contains all the necessary equations for a complete model, with the model parameters provided in an appendix. Although some models are over-simplified, the presentation should make it clear how more realistic models can replace the models used here. The simulation of realistic model topologies is greatly enhanced by using equation-based modeling languages, and a secondary aim is to demonstrate this. Simulation results are provided both to demonstrate the physical realism of the model, and also to allow for comparison with other implementations. Simulation languages embedded in an extensive eco-system will make control for optimal operation much simpler. A third aim is, thus, to demonstrate this possibility via a simple tuning of a controller based on the model at hand.

Section 2 gives an overview of the transport system from the oil reservoir via a manifold to a separator, and the key equipment models. Section 3 develops a simple mechanistic model of the system. Section 4 contrasts two modeling languages for simulation: Modelica and Julia's ModelingToolkit. Section 5 illustrates model behavior and the use of modeling/simulation tools for analysis and control. Finally, Section 6 provides some conclusions.

2. System Description

Production of a mixture of water and crude oil in the liquid phase is considered, where the evaporation of liquids is assumed to be negligible. In the sequel, for an *extensive* variable x, $\dot{x}$ denotes the flow rate of x. More generally (i.e., for both extensive and intensive variables), $\frac{dx}{dt}$ denotes the time derivative of any variable x.

2.1. System Topology

Oil production *systems* merge several boreholes from the same or different reservoirs through vertical pipes into a manifold. Normally, more than one horizontal transport pipe is needed from the manifold to a separator for sufficient transport capacity—too large a diameter in pipes leads to unwanted laminar flow, while too small a diameter leads to high velocity and high friction loss. Water is commonly injected into the manifold to reduce the friction loss in the horizontal pipes. The added water is typically recycled from the separator, and is at close to the production temperature, thus making inclusion of the energy balance less important. Figure 1 shows a system with n_{w} wells/vertical pipes via a common manifold to n_{t} transportation/horizontal pipes leading to the separator.

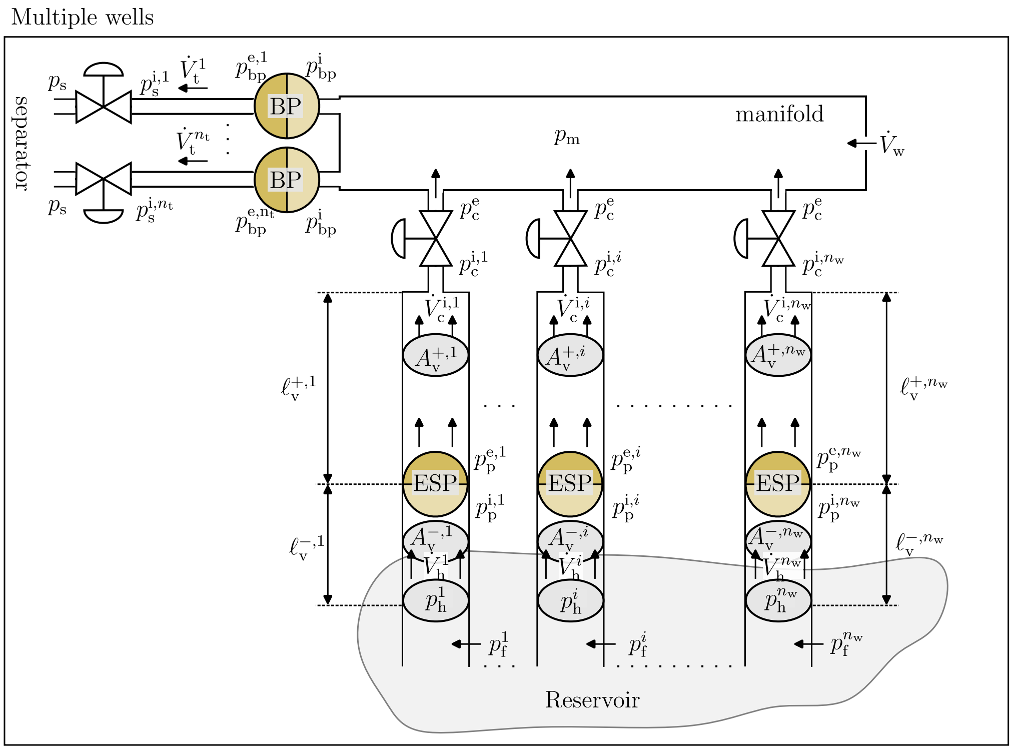

Multiple wells
separator
p_{s}
$p_{\mathrm{s}}^{\mathrm{i},1}$
$\dot{V}_{\mathrm{t}}^{1}$
$p_{\mathrm{bp}}^{\mathrm{e},1}$
$p_{\mathrm{bp}}^{\mathrm{i}}$
BP
$\dot{V}_{\mathrm{t}}^{n_{\mathrm{t}}}$
BP
p_{s}
$p_{\mathrm{s}}^{\mathrm{i},n_{\mathrm{t}}}$
$p_{\mathrm{bp}}^{\mathrm{e},n_{\mathrm{t}}}$
$p_{\mathrm{bp}}^{\mathrm{i}}$
manifold
p_{m}
$\dot{V}_{\mathrm{w}}$
$p_{\mathrm{c}}^{\mathrm{e}}$
$p_{\mathrm{c}}^{\mathrm{i},1}$
$p_{\mathrm{c}}^{\mathrm{i},i}$
$p_{\mathrm{c}}^{\mathrm{i},n_{\mathrm{w}}}$
$\dot{V}_{\mathrm{c}}^{\mathrm{i},1}$
$\dot{V}_{\mathrm{c}}^{\mathrm{i},i}$
$\dot{V}_{\mathrm{c}}^{\mathrm{i},n_{\mathrm{w}}}$
$A_{\mathrm{v}}^{+,1}$
$A_{\mathrm{v}}^{+,i}$
$A_{\mathrm{v}}^{+,n_{\mathrm{w}}}$
$\ell_{\mathrm{v}}^{+,1}$
$\ell_{\mathrm{v}}^{+,n_{\mathrm{w}}}$
ESP
$p_{\mathrm{p}}^{\mathrm{e},1}$
$p_{\mathrm{p}}^{\mathrm{e},i}$
$p_{\mathrm{p}}^{\mathrm{e},n_{\mathrm{w}}}$
$p_{\mathrm{p}}^{\mathrm{i},1}$
$p_{\mathrm{p}}^{\mathrm{i},i}$
$p_{\mathrm{p}}^{\mathrm{i},n_{\mathrm{w}}}$
$A_{\mathrm{v}}^{-,1}$
$A_{\mathrm{v}}^{-,i}$
$A_{\mathrm{v}}^{-,n_{\mathrm{w}}}$
$\ell_{\mathrm{v}}^{-,1}$
$\ell_{\mathrm{v}}^{-,n_{\mathrm{w}}}$
$\dot{V}_{\mathrm{h}}^{1}$
$\dot{V}_{\mathrm{h}}^{i}$
$\dot{V}_{\mathrm{h}}^{n_{\mathrm{w}}}$
p_{h}^{1}
p_{h}^{i}
$p_{\mathrm{h}}^{n_{\mathrm{w}}}$
p_{f}^{1}
p_{f}^{i}
$p_{\mathrm{f}}^{n_{\mathrm{w}}}$
Reservoir

Figure 1. Multiple well system with n_{w} wells, possibly coming from different reservoirs, and n_{t} transport pipes to the separator; from Lie [1] and based on Sharma and Glemmestad [2].

In Figure 1, p_{f}^{j} is the reservoir *formation* pressure for well j, p_{h}^{j} is the bore-hole *heel* pressure, $\dot{V}_{\mathrm{h}}^{j}$ is the volumetric heel flow rate, $A_{\mathrm{v}}^{-,j}$ is the vertical cross-sectional area below the ESP, $\ell_{\mathrm{v}}^{-,j}$ is the length of the vertical pipe below the ESP, $p_{\mathrm{p}}^{\mathrm{i},j}$ is the inlet pressure to the (ESP) pump, $p_{\mathrm{p}}^{\mathrm{e},j}$ is the effluent pressure after the (ESP) pump, $A_{\mathrm{v}}^{+,j}$ is the vertical cross-sectional area above the ESP, $\ell_{\mathrm{v}}^{+,j}$ is the length of the vertical pipe above the ESP, $\dot{V}_{\mathrm{c}}^{\mathrm{i},j}$ is the volumetric flow rate at the inlet to the choke valve, $p_{\mathrm{c}}^{\mathrm{i},j}$ is the pressure at the inlet to the choke valve, and $p_{\mathrm{c}}^{\mathrm{e},j}$ is the pressure at the effluent from the choke valve. Next, p_{m} is the manifold pressure, while $\dot{V}_{\mathrm{w}}$ is the water volumetric flow rate added to the manifold to reduce the viscosity. At the outlet from the manifold, $p_{\mathrm{bp}}^{\mathrm{i},j}$ is the influent pressure to the booster pump (BP), $p_{\mathrm{bp}}^{\mathrm{e},j}$ is the effluent pressure after the booster pump, $\dot{V}_{\mathrm{t}}^{j}$ is the volumetric flow rate in a transport pipe from the manifold to the separator, $p_s^{i,j}$ is the pressure at the inlet to the valve into the separator, and $p_s^{e,j} = p_s$ is the separator pressure.

For simplicity, it is assumed that $A_v^{-,j} = A_v^{+,j} = A_v$. All the vertical pipes are assumed to be connected to the same manifold pressure p_m; hence, *the effluent choke pressure* satisfies $p_c^{e,j} = p_c^e = p_m$ for all j. The influent pressure to the booster pumps, $p_{bp}^{i,j}$, are all assumed to be equal to the outlet pressure from the manifold, and have the same value, $p_{bp}^{i,j} = p_{bp}^{i} = p_m$. Likewise, all the transport pipes end up in the same separator: $p_s^{e,j} = p_s$ for all j. Here, a choke at the end of the transport pipes has been neglected.

2.2. Fluid Properties

The petroleum fluid properties are important. The density ρ varies with the pressure p and the temperature T, $\rho(p,T)$. Neglecting the temperature dependence, and assuming constant *isothermal compressibility* β_T (good thermodynamics books define the isothermal compressibility; see also https://en.wikipedia.org/wiki/Compressibility; accessed on 13 December 2023), the isothermal compressibility is the inverse of the bulk modulus.

$$\beta_T = \frac{1}{\rho}\frac{\partial \rho}{\partial p}\bigg|_T = \text{constant}$$

leads to $\rho(p)$, given as

$$\rho = \rho_0 \exp(\beta_T(p - p_0)), \tag{1}$$

where (ρ_0, p_0) is some reference state.

Defining the water cut χ_w as $\chi_w \triangleq \dot{V}_w/\dot{V}$, the volumetric flow rate of water divided by the total flow rate of the fluid, the total density ρ can be expressed as

$$\rho = \chi_w \rho_w + (1 - \chi_w)\rho_o, \tag{2}$$

where ρ_w and ρ_o are the constant densities of pure water and crude oil, respectively.

In reality, water and crude oil have different isothermal compressibilities. Here, we simplify and assume an overall value for β_T. Using the data in Table A1 of Appendix A, the density ρ varies by ca. 10 kg/m^3 with pressure variation in the range 25–225 bar (Figure 2).

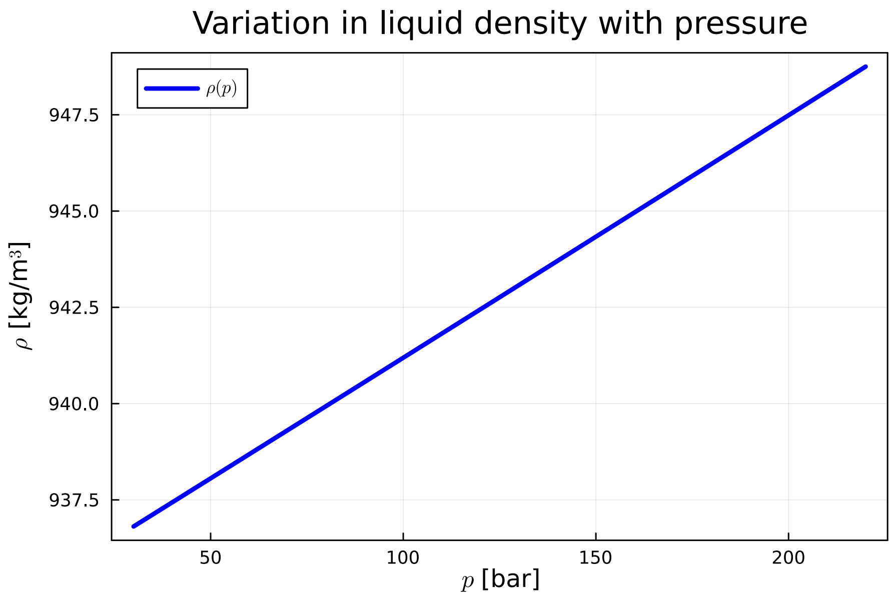


Figure 2. Typical variation in density for production fluid in the pressure range of interest.

Thus, we assume constant density in the pipes, but a pressure-dependent density is assumed in the manifold.

The authors of Sharma and Glemmestad [2] proposed a simple linear mixing rule for *the kinematic* viscosity ν:

$$\nu = \chi_{\mathrm{w}} \nu_{\mathrm{w}} + (1 - \chi_{\mathrm{w}}) \nu_{\mathrm{o}}. \tag{3}$$

With ν known, *the dynamic* viscosity μ can be computed (if needed) as

$$\mu = \nu \rho.$$

The linear interpolation model of Equation (3) is used here to faciliate comparison with the results in Sharma and Glemmestad [2], even though it is not physically realistic [8].

2.3. Well-Bore Production

The total production from the reservoir (formation pressure p_{f}) relates the volumetric petroleum fluid rate $\dot{V}_{\mathrm{h}}$ at the well-bore heel as $\dot{V}_{\mathrm{h}} \propto p_{\mathrm{f}} - p_{\mathrm{h}}$, where p_{h} is the heel pressure and the proportionality constant C_{pi} is the *productivity index*,

$$\dot{V}_{\mathrm{h}} = C_{\mathrm{pi}} \cdot (p_{\mathrm{f}} - p_{\mathrm{h}});$$

C_{pi} is unit-dependent. Here, we instead propose a dimensionless form,

$$\dot{V}_{\mathrm{h}} = \dot{V}_{\mathrm{pi}}^{\mathrm{c}} \frac{p_{\mathrm{f}} - p_{\mathrm{h}}}{p_{\mathrm{pi}}^{\varsigma}}, \tag{4}$$

where $\dot{V}_{\mathrm{pi}}^{\mathrm{c}}$ is the productivity index *capacity* in the same unit as $\dot{V}_{\mathrm{h}}$, and $p_{\mathrm{pi}}^{\varsigma}$ is the scaling pressure with the same unit as $p_{\mathrm{f}}, p_{\mathrm{h}}$.

2.4. Pump Models

2.4.1. Electric Submersible Pump

Pump models are often given as

$$\Delta p_{\mathrm{p}} = \rho g h_{\mathrm{p}}, \tag{5}$$

where $h_{\mathrm{p}} = h_{\mathrm{p}}(\dot{V}, f_{\mathrm{p}})$ is the pump *head* with a volumetric flow rate $\dot{V}$ and a control input f_{p}, rotational pump frequency Hz.

The authors of Sharma and Glemmestad [2] provide values for the minimal, maximal, and best-efficiency-point flow rates,

$$\frac{\dot{V}_{\mathrm{min}}}{\dot{V}_{\mathrm{min},0}} = \frac{f_{\mathrm{p}}}{f_{\mathrm{p},0}} \tag{6}$$

$$\frac{\dot{V}_{\mathrm{max}}}{\dot{V}_{\mathrm{max},0}} = \frac{f_{\mathrm{p}}}{f_{\mathrm{p},0}} \tag{7}$$

$$\frac{\dot{V}_{\eta}}{\dot{V}_{\eta,0}} = \frac{f_{\mathrm{p}}}{f_{\mathrm{p},0}}. \tag{8}$$

In Sharma and Glemmestad [2], a comprehensive model for the pump head of a *multi-stage ESP* is developed. To ease change in the units, their model is rewritten here in dimensionless form

$$\frac{h_{\mathrm{p}}(\dot{V}, f_{\mathrm{p}})}{h_{\mathrm{p},0}} = \left(\frac{f_{\mathrm{p}}}{f_{\mathrm{p},0}}\right)^2 + \sum_{j=1}^{3} a_j \left(\frac{f_{\mathrm{p}}}{f_{\mathrm{p},0}}\right)^{2-j} \left(\frac{\dot{V}}{\dot{V}^{\varsigma}}\right)^j. \tag{9}$$

In Equation (9), $h_{\mathrm{p},0}$ is a nominal scaling head, f_{p} is the pump rotational frequency in the same unit as that of the nominal rotational frequency $f_{\mathrm{p},0}$, $\dot{V}$ is the actual volumetric flow rate out of the pump, $\dot{V}^{\varsigma}$ is the scaling flow rate, and $a_1, \ldots, a_3$ are dimensionless model parameters. (Here, a_j is dimensionless, while in Sharma [3], the parameters a_j

have dimensions). This implies that the values of a_j here are different from those of a_j in Sharma and Glemmestad [2], Sharma [3]. In Sharma and Glemmestad [2], a head curve plot is provided; the result in Figure 3 based on the dimensionless model in Equation (9) is identical to their plot.

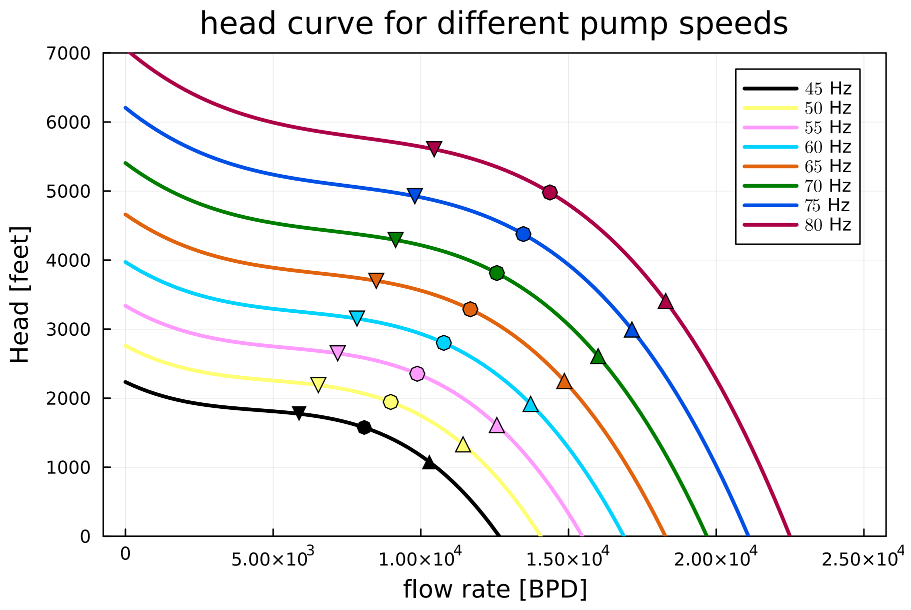


Figure 3. ESP pump head in ft as a function of the volumetric flow rate in bbl/d for selected pump speeds. The lower, best-efficiency-point, and maximum flow rates are indicated.

In addition, Sharma and Glemmestad [2] provide a model for the mechanical power requirement $\dot{W}_{\mathrm{p}}^{\mathrm{m}} = \dot{W}_{\mathrm{p}}^{\mathrm{m}}(\dot{V}, f_{\mathrm{p}})$ for operating the pump ("brake horse power", BHP, in the original publication), again rewritten in dimensionless form,

$$\frac{\dot{W}_{\mathrm{p}}^{\mathrm{m}}}{\dot{W}_{\mathrm{p},0}^{\mathrm{m}}} = \left(\frac{f_{\mathrm{p}}}{f_{\mathrm{p},0}}\right)^{3} + \sum_{j=1}^{4} b_j \left(\frac{f_{\mathrm{p}}}{f_{\mathrm{p},0}}\right)^{3-j} \left(\frac{\dot{V}}{\dot{V}_{\varsigma}}\right)^{j}. \tag{10}$$

In Equation (10), $\dot{W}_{\mathrm{p},0}^{\mathrm{m}}$ is the nominal scaling power consumption to operate the pump, $b_1, \ldots, b_4$ are dimensionless model parameters, while f_{p} and $\dot{V}$ are as above.

The actual power added to the fluid is

$$\dot{W}_{\mathrm{p}} = \Delta p_{\mathrm{p}} \dot{V}, \tag{11}$$

which gives the efficiency as

$$\eta = \frac{\dot{W}_{\mathrm{p}}}{\dot{W}_{\mathrm{p}}^{\mathrm{m}}} = \frac{\Delta p_{\mathrm{p}} \dot{V}}{\dot{W}_{\mathrm{p}}^{\mathrm{m}}}, \tag{12}$$

where it is assumed that $\dot{W}_{\mathrm{p}}$ and $\dot{W}_{\mathrm{p}}^{\mathrm{m}}$ have the same units. Sharma and Glemmestad [2] include an efficiency curve plot; the result of Equation (12) based on a dimensionless model is identical to their plot.

2.4.2. Booster Pump

For the *booster pump* in the horizontal pipes, a simpler model is suggested in Sharma and Glemmestad [2], rewritten in dimensionless form as

$$\frac{\Delta p_{\mathrm{bp}}\left(f_{\mathrm{bp}}\right)}{\Delta p_{\mathrm{bp},0}} = \left(\frac{f_{\mathrm{bp}}}{f_{\mathrm{bp},0}}\right)^{2}, \tag{13}$$

where $\Delta p_{\text{bp}}\left(f_{\text{bp}}\right)$ is the pressure increase at the given pump frequency/speed f_{bp}, in the same units as $\Delta p_{\text{bp},0}$, which is the pressure increase at the nominal pump frequency $f_{\text{bp},0}$.

2.4.3. Pump Control Input

In reality, the pump speed (f_{p}, f_{bp}) is not a control input. Instead, a motor is used to control the torque applied to the aggregate of the motor and the pump.

In Sharma [3], a model of the induction motor driving the ESP is developed. The experience of [3] is that the motor dynamics are much faster than those of the mechanical system; hence, in most of his work, Sharma [3] neglects the motor dynamics. However, it is not clear whether Sharma [3] considers the mechanical dynamics of accelerating the pump itself. These dynamics would be described by the kinetic energy balance in rotational form (the "swing equation"), which can be written as

$$\frac{\mathrm{d}K}{\mathrm{d}t} = P_{\text{i}} - \dot{W}_{\text{p}}$$

with kinetic energy K

$$K = \frac{1}{2} J_{\text{t}} \omega_p^2,$$

where J_{t} is the total moment of inertia for the pump, the motor, and a possible flywheel, while P_{i} is the input power from the motor (control input), and $\dot{W}_{\text{p}}$ is as in Equation (11).

In Sharma [3], the moment of inertia is given as approximately $J \approx 71\,\text{kg}\,\text{m}^2$. It is not clear whether this is the motor moment of inertia or the total moment of inertia. However, using such a moment of inertia leads to a pump time constant which is still much faster than the dynamics of the flow $\dot{V}_{\text{v}}$, etc.; hence, the pump/motor dynamics is neglected here, and for simplicity, it is assumed that f_{p} is a control input.

2.5. Valve Models

The authors of Sharma and Glemmestad [2] base their valve models on the ANSI/ISA S75.01 standard (http://integrated.cc/cse/ISA_750101_SPBd.pdf; accessed on 13 December 2023). Here, instead, a dimensionless description is proposed with extension to a control input,

$$\dot{m} = \dot{m}_{\text{v}}^{\text{c}} \cdot f(u_{\text{v}}) \frac{\rho_{\text{i}}}{\rho_{\text{e}}} \sqrt{\frac{(p_{\text{i}} - p_{\text{e}})/p^{\varsigma}}{\rho_{\text{i}}/\rho^{\varsigma}}}, \tag{14}$$

where $\dot{m}_{\text{v}}^{\text{c}}$ is the valve mass flow rate *capacity*, $u_{\text{v}} \in [0,1]$ is the valve control signal, $f : [0,1] \to [0,1]$ is the valve characteristics, $\rho_{\text{i}}, \rho_{\text{e}}$ are the influent and effluent densities, respectively, $p_{\text{i}}, p_{\text{e}}$ are the influent and effluent pressures, respectively, while $\rho^{\varsigma}, p^{\varsigma}$ are the scaling density and pressure, respectively.

2.6. Friction Loss

The friction drop along the pipe can be given by the Darcy–Weisbach model (e.g., https://en.wikipedia.org/wiki/Darcy%E2%80%93Weisbach_equation; accessed on 13 December 2023) as

$$\frac{\Delta p_{\text{f}}}{\ell} = f_{\text{D}} \frac{\rho}{2} \frac{v^2}{D}, \tag{15}$$

where f_{D} is Darcy's friction factor given by Colebrook's implicit expression (the Colebrook equation, sometimes known as the Colebrook–White equation). One explicit *approximation* to Colebrook's expression is due to Swamee and Jain [12],

$$\frac{1}{\sqrt{f_{\text{D}}}} = -2 \cdot \log_{10}\left(\frac{5.74}{N_{\text{Re}}^{0.9}} + \frac{\epsilon/D}{3.7}\right), \tag{16}$$

where N_{Re} is the Reynolds number,

$$N_{\mathrm{Re}} = \frac{\rho v D}{\mu} = \frac{vD}{\nu}, \tag{17}$$

where μ is the *dynamic* viscosity, ν is the *kinematic* viscosity, and ϵ is the "roughness height" of the pipe internal surface. The linear velocity v is related to the volumetric flow rate $\dot{V}$ by

$$\dot{V} = vA, \tag{18}$$

where A is the cross-sectional area of the pipe.

2.7. Why Dimensionless Models?

2.7.1. Example: ESP Pump Model

As a first example, consider the ESP model in Equation (9). In the original formulation in Sharma and Glemmestad [2] (slight change in notation),

$$h_{\mathrm{p}} = \bar{a}_0 \frac{f_{\mathrm{p}}}{f_{\mathrm{p},0}} + \bar{a}_1 \frac{f_{\mathrm{p}}}{f_{\mathrm{p},0}} \dot{V} + \bar{a}_2 \dot{V}^2 + \bar{a}_3 \frac{f_{\mathrm{p},0}}{f_{\mathrm{p}}} \dot{V}^3, \tag{19}$$

where the parameters $\bar{a}_j$ have rather complicated units and the equation is hard-coded with $\left[\dot{V}\right] = \mathrm{gal/min}$, while $\left[h_{\mathrm{p}}\right] = \mathrm{ft}$ (for quantity x, $[x]$ is the unit of the quantity). In practice, (i) either the remainder of the model has to be posed using these units, or (ii) one has to operate with several copies of the variables, e.g., $\dot{V}_{\mathrm{gal/min}}$ and $\dot{V}$, and remember to correctly convert between these versions of the flow rate. Both of these approaches are error-prone, and also require several versions of the variables.

A much better solution is to write the model in dimensionless form. The simplest way to do this for the model in Equation (19), is as

$$\frac{h_{\mathrm{p}}}{h_{\mathrm{p},0}} = \tilde{a}_0 \frac{f_{\mathrm{p}}}{f_{\mathrm{p},0}} + \tilde{a}_1 \frac{f_{\mathrm{p}}}{f_{\mathrm{p},0}} \frac{\dot{V}}{\dot{V}^{\varsigma}} + \tilde{a}_2 \left(\frac{\dot{V}}{\dot{V}^{\varsigma}}\right)^2 + \tilde{a}_3 \frac{f_{\mathrm{p},0}}{f_{\mathrm{p}}} \left(\frac{\dot{V}}{\dot{V}^{\varsigma}}\right)^3. \tag{20}$$

If we choose $h_{\mathrm{p},0} \equiv 1\,\mathrm{ft}$ and $\dot{V}^{\varsigma} \equiv 1\,\mathrm{gal/min}$, then, $\left[\tilde{a}_j\right] \equiv \left[\bar{a}_j\right]$. Suppose we want to generate the plot in Figure 3. Because that figure plots h_{p} in ft (the "native" unit), while the flow rate $\dot{V}$ is given in bbl/d (the "native" unit is gal/min), this result is produced by choosing $\dot{V}^{\varsigma} = 1\,\mathrm{gal/min} = 34.29\,\mathrm{bbl/d}$, which can easily be found using, e.g., the WolframAlpha app (e.g., the Microsoft Store).

In practice, it may be better to choose a more natural scaling unit, e.g., SI units. In that case, it is necessary to change the parameters $\bar{a}_j$; for the parameters in Equation (9), the scaling parameters are in SI units, where $\bar{a}_j \rightarrow a_j$ and a_j are given in Table A2. To find a_j, choose $h_{\mathrm{p},0} = 1\,\mathrm{ft} = 0.3048\,\mathrm{m}$, $\dot{V}^{\varsigma} = 1\,\mathrm{gal/min} = 6.309 \times 10^{-5}\,\mathrm{m^3/s}$, and write Equation (20) as

$$\underbrace{\frac{h_{\mathrm{p}}}{h_{\mathrm{p},0}\tilde{a}_0}}_{\rightarrow h_{\mathrm{p},0}} = \frac{f_{\mathrm{p}}}{f_{\mathrm{p},0}} + \underbrace{\frac{\tilde{a}_1}{\tilde{a}_0 \dot{V}^{\varsigma}}}_{a_1} \frac{f_{\mathrm{p}}}{f_{\mathrm{p},0}} \dot{V} + \underbrace{\frac{\tilde{a}_2}{\tilde{a}_0 \left(\dot{V}^{\varsigma}\right)^2}}_{a_2} \dot{V}^2 + \underbrace{\frac{\tilde{a}_3}{\tilde{a}_0 \left(\dot{V}^{\varsigma}\right)^3}}_{a_3} \frac{f_{\mathrm{p},0}}{f_{\mathrm{p}}} \dot{V}^3,$$

where the new $h_{\mathrm{p},0}$ is given in unit m, a_1, a_2, a_3 are the new, dimensionless parameters in Equation (9), while the new scaling flow rate $\rightarrow \dot{V}^{\varsigma} = 1\,\mathrm{m^3/s}$. With this modified correlation $(a_1, \ldots, a_3)$ for h_{p} using SI units, the dimensionless form is as in Equation (9).

2.7.2. Example: Control Valve

The ANSI/ISA S75.01 standard (http://integrated.cc/cse/ISA_750101_SPBd.pdf; accessed on 13 December 2023) for *compressible* (i.e., $\dot{m}_{\mathrm{i}} = \dot{m}_{\mathrm{e}} = \dot{m}$, $\rho_{\mathrm{i}} \neq \rho_{\mathrm{e}}$), non-choked fluids without fitting is

$$C = \frac{\dot{m}}{N_6 \frac{\rho_{\mathrm{i}}}{\rho_{\mathrm{e}}} \sqrt{\frac{p_{\mathrm{i}} - p_{\mathrm{e}}}{\rho_{\mathrm{i}}}}}, \tag{21}$$

where, C is the valve coefficient, $\dot{m}$ is the mass flow rate through the valve, ρ_{i} is the influent density, ρ_{e} is the effluent density, p_{i} is the influent pressure, p_{e} is the effluent pressure, and N_6 is used to handle unit conversion. Typically, tabular values for N_6 are given which are valid for different combinations of units for $\dot{m}$, ρ, and p. This makes change to the units rather complicated. The dimensionless formulation as in Equation (14) greatly simplifies the use of the valve model in different units, and also includes a control valve characteristic.

3. Dynamic Model

3.1. Balance Laws

The model is based on the total mass balance (manifold) and the linear momentum balance (pipes) (see any good book on balance models, such as Bird et al. [13]). The total mass balance is expressed as

$$\frac{\mathrm{d}m}{\mathrm{d}t} = \dot{m}_{\mathrm{i}} - \dot{m}_{\mathrm{e}}, \tag{22}$$

where m is the accumulated mass in the system, t is the time, $\dot{m}$ is the mass flow rate, and the indices (i, e) denote influent and effluent, respectively.

The linear momentum balance is

$$\frac{\mathrm{d}\mathfrak{m}}{\mathrm{d}t} = \dot{\mathfrak{m}}_{\mathrm{i}} - \dot{\mathfrak{m}}_{\mathrm{e}} + F, \tag{23}$$

where $\mathfrak{m}$ is the linear momentum given as $\mathfrak{m} = mv$ with linear velocity v, $\dot{\mathfrak{m}}$ is the momentum flow rate given as $\dot{\mathfrak{m}} = \dot{m}v$, and F is the total force. With constant fluid density, $\dot{\mathfrak{m}}_{\mathrm{i}} = \dot{\mathfrak{m}}_{\mathrm{e}}$, the momentum balance reduces to Newton's law, $\frac{\mathrm{d}\mathfrak{m}}{\mathrm{d}t} = F$.

3.2. Vertical Pipes with ESP

We assume constant density in the pipes, causing the volumetric vertical flow rate $\dot{V}_{\mathrm{v}}$ to be the same everywhere: $\dot{V}_{\mathrm{h}} = \dot{V}_{\mathrm{c}}^{\mathrm{i}} = \dot{V}_{\mathrm{v}}$. Furthermore, Equation (23) reduces to Newton's law. The momentum is given as $\mathfrak{m} = \dot{m}v$ with $\dot{m} = \rho \dot{V}_{\mathrm{v}}$, and v is related to $\dot{V}_{\mathrm{v}}$ by Equation (18). The total force is $F = F_{\mathrm{p}} + F_{\mathrm{b}} - F_{\mathrm{f}} - F_{\mathrm{g}}$, with

- Pressure forces at the inlet and outlet of the pipe,

$$F_{\mathrm{p}} = p_{\mathrm{h}}A - p_{\mathrm{c}}^{\mathrm{i}}A. \tag{24}$$

- Possible pressure boost due to a pump,

$$F_{\mathrm{b}} = \Delta p_{\mathrm{p}} A, \tag{25}$$

  with Δp_{p} given by Equations (5) and (9).
- Friction loss,

$$F_{\mathrm{f}} = \Delta p_{\mathrm{f}} A, \tag{26}$$

  with Δp_{f} given by Equations (15)–(18).
- Flow against gravity, with a vertical height h,

$$F_{\mathrm{g}} = \Delta p_{\mathrm{g}} A, \tag{27}$$

  with

$$\Delta p_{\mathrm{g}} = \rho_{\mathrm{v}} g h. \tag{28}$$

In addition, we need information about how the flow rate $\dot{V}_{\mathrm{v}}$ relates to the bottom hole pressure via the productivity index, Equation (4), and how the flow rate $\dot{V}_{\mathrm{v}}$ relates to the choke valve flow, Equation (14).

The most structured formulation would be to pose the momentum balance (here, Newton's law) as the differential equation, and add all the necessary algebraic equations. However, the OpenModelica DAE solver struggles with such a formulation. The valve equation Equation (14) is implicit in the pressure difference; in the iteration to find $\Delta p_{\mathrm{v}} = p_{\mathrm{i}} - p_{\mathrm{e}}$, if Δp_{v} becomes negative, the square root gives a complex number, and the simulation crashes (it was not tested whether ModelingToolkit can handle this implicit algebraic equation). Instead, if the differential variable is changed to $\dot{V}_{\mathrm{v}}$, then the valve equation can be inverted and expressed as $\Delta p_{\mathrm{v}} \propto \dot{V}_{\mathrm{v}}^2$.

The following formulation is used in OpenModelica and ModelingToolkit:

$$\frac{\mathrm{d}\dot{V}_{\mathrm{v}}}{\mathrm{d}t} = \frac{p_{\mathrm{h}} - p_{\mathrm{i}}^{\mathrm{c}} + \Delta p_{\mathrm{p}} - \Delta p_{\mathrm{f}} - \Delta p_{\mathrm{g}}}{\rho_{\mathrm{v}} \ell / A} \tag{29}$$

$$\rho_{\beta}^{0} = \chi_{\mathrm{w}} \rho_{\mathrm{w}} + (1 - \chi_{\mathrm{w}}) \rho_{\mathrm{o}} \tag{30}$$

$$\nu = \chi_{\mathrm{w}} \nu_{\mathrm{w}} + (1 - \chi_{\mathrm{w}}) \nu_{\mathrm{o}} \tag{31}$$

$$\mu = \rho_{\beta}^{0} \nu \tag{32}$$

$$\rho_{\mathrm{v}} = \rho_{\beta}^{0} \exp\left(\beta_T \left(p_{\mathrm{c}}^{\mathrm{i}} - p_{\beta}^{0}\right)\right) \tag{33}$$

$$p_{\mathrm{h}} = p_{\mathrm{f}} - p_{\mathrm{pi}}^{\varsigma} \frac{\dot{V}_{\mathrm{v}}}{\dot{V}_{\mathrm{pi}}^{\mathrm{c}}} \tag{34}$$

$$\dot{m}_{\mathrm{v}} = \rho_{\mathrm{v}} \dot{V}_{\mathrm{v}} \tag{35}$$

$$p_{\mathrm{i}}^{\mathrm{c}} = p_{\mathrm{m}} + p_{\mathrm{v}}^{\varsigma} \frac{\rho_{\mathrm{v}}}{\rho_{\mathrm{v}}^{\varsigma}} \left(\frac{\dot{m}_{\mathrm{v}}}{\dot{m}_{\mathrm{v}}^{\varsigma}}\right)^2 \frac{1}{f_{\mathrm{c}}^2(u_{\mathrm{v}})} \tag{36}$$

$$\begin{aligned} h_{\mathrm{p}} = h_{\mathrm{p},0} \Bigg(\left(\frac{f_{\mathrm{p}}}{f_{\mathrm{p},0}}\right)^2 + a_1 \frac{f_{\mathrm{p}}}{f_{\mathrm{p},0}} \frac{\dot{V}_{\mathrm{v}}}{\dot{V}^{\varsigma}} \\ + a_2 \left(\frac{\dot{V}_{\mathrm{v}}}{\dot{V}^{\varsigma}}\right)^2 + a_3 \frac{f_{\mathrm{p},0}}{f_{\mathrm{p}}} \left(\frac{\dot{V}_{\mathrm{v}}}{\dot{V}^{\varsigma}}\right)^3 \Bigg) \end{aligned} \tag{37}$$

$$\Delta p_{\mathrm{p}} = \rho_{\mathrm{v}} g h_{\mathrm{p}} \tag{38}$$

$$v_{\mathrm{v}} = \frac{\dot{V}_{\mathrm{v}}}{A} \tag{39}$$

$$N_{\mathrm{Re}} = \frac{v_{\mathrm{v}} d_{\mathrm{v}}}{\nu_{\mathrm{v}}} \tag{40}$$

$$f_{\mathrm{D}}^{\mathrm{v}} = \frac{1}{4\left(\log_{10}\left(\frac{5.74}{N_{\mathrm{Re}}^{0.9}} + \frac{\epsilon_{\mathrm{v}}/d_{\mathrm{v}}}{3.7}\right)\right)^2} \tag{41}$$

$$\Delta p_{\mathrm{f}} = \ell \cdot f_{\mathrm{D}}^{\mathrm{v}} \frac{\rho_{\mathrm{v}}}{2} \frac{v_{\mathrm{v}}^2}{d_{\mathrm{v}}} \tag{42}$$

$$\Delta p_{\mathrm{g}} = \rho_{\mathrm{v}} g h. \tag{43}$$

If we only consider the model of a single vertical pipe, we need to specify (i) initial state (e.g., $\dot{V}_\mathrm{v}$), (ii) all "input" variables, i.e., p_f, f_p, p_m, and possibly the water cut χ_w, and (iii) all the parameters, i.e., ρ_w, ρ_o, ν_w, ν_o, p^0_β, ℓ, A, p^ς_pi, $\dot{V}^\mathrm{c}_\mathrm{pi}$, p^ς_v, $\rho^\varsigma_\mathrm{v}$, $\dot{m}^\mathrm{c}_\mathrm{v}$, $h_\mathrm{p,0}$, $f_\mathrm{p,0}$, $\dot{V}^\varsigma$, a_1,a_2, a_3, g, d_v, ν_v, ϵ_v, h.

3.3. Manifold

We assume a perfectly mixed manifold. Assuming a constant manifold volume V_m, and adding water at the flow rate $\dot{V}_\mathrm{w}$ to dilute the fluid to a specified manifold water cut $\chi^\mathrm{m}_\mathrm{w}$, thus reducing friction loss in the pipe towards the separator, $\dot{V}_\mathrm{w}$ must be approximately

$$\dot{V}_\mathrm{w} = \frac{\chi^\mathrm{m}_\mathrm{w} - \chi_\mathrm{w}}{1 - \chi^\mathrm{m}_\mathrm{w}} \dot{V}_\mathrm{v}. \tag{44}$$

The total mass balance for the manifold can then be expressed as

$$\frac{\mathrm{d}p_\mathrm{m}}{\mathrm{d}t} = \frac{1}{\rho_\mathrm{m} V_\mathrm{m} \beta_T} \left(\rho_\mathrm{v}\dot{V}_\mathrm{v} + \rho_\mathrm{w}\dot{V}_\mathrm{w} - \rho_\mathrm{m}\dot{V}_\mathrm{t}\right) \tag{45}$$

$$\rho^0_\beta = \chi^\mathrm{m}_\mathrm{w}\rho_\mathrm{w} + (1-\chi^\mathrm{m}_\mathrm{w})\rho_\mathrm{o} \tag{46}$$

$$\rho_\mathrm{m} = \rho^0_\beta \exp\left(\beta_T\left(p_\mathrm{m} - p^0_\beta\right)\right) \tag{47}$$

$$\dot{V}_\mathrm{w} = \frac{\chi^\mathrm{m}_\mathrm{w} - \chi_\mathrm{w}}{1 - \chi^\mathrm{m}_\mathrm{w}} \dot{V}^\mathrm{i}_\mathrm{c} \tag{48}$$

In practice, the water cut χ_w and the flow rate $\dot{V}^\mathrm{i}_\mathrm{c}$ are not known perfectly, and it is necessary to use a feedback control system to manipulate $\dot{V}_\mathrm{w}$ instead of using Equation (44). Here, for simplicity, Equation (44) is still used.

For the manifold model, we must know (i) the initial manifold pressure, (ii) the vertical inflow $\dot{V}_\mathrm{v}$, and the horizontal transport flow $\dot{V}_\mathrm{t}$ from the manifold to the separator, as well as the manifold water cut $\chi^\mathrm{m}_\mathrm{w}$, and (iii) the parameters.

3.4. Transport Pipe

For simplicity, we will neglect the separator inlet valve, and assume that $p^{\mathrm{i},j}_\mathrm{s} \equiv p_\mathrm{s}$. It is straightforward to reverse this assumption.

The model of the horizontal pipe from the manifold to the separator is almost identical to the vertical pipe from the reservoir to the manifold. The essential differences are (i) no gravity pressure drop, (ii) a simpler booster pump model, (iii) neglect of the pressure drop from the pipe into the separator, and (iv) no need for a productivity index model. The complete model is

$$\frac{\mathrm{d}\dot{V}_\mathrm{t}}{\mathrm{d}t} = \frac{p_\mathrm{m} - p_\mathrm{s} + \Delta p_\mathrm{bp} - \Delta p^\mathrm{t}_\mathrm{f}}{\rho_\mathrm{t}\ell_\mathrm{t}/A_\mathrm{t}} \tag{49}$$

$$\rho^{0,\mathrm{t}}_\beta = \chi^\mathrm{m}_\mathrm{w}\rho_\mathrm{w} + (1-\chi^\mathrm{m}_\mathrm{w})\rho_\mathrm{o} \tag{50}$$

$$\nu_\mathrm{t} = \chi^\mathrm{m}_\mathrm{w}\nu_\mathrm{w} + (1-\chi^\mathrm{m}_\mathrm{w})\nu_\mathrm{o} \tag{51}$$

$$\mu_\mathrm{t} = \rho^{0,\mathrm{t}}_\beta \nu_\mathrm{t} \tag{52}$$

$$\rho_\mathrm{t} = \rho^0_\beta \exp\left(\beta_T\left(p_\mathrm{m} - p^0_\beta\right)\right) \tag{53}$$

$$\Delta p_\mathrm{bp} = \Delta p_\mathrm{bp,0}\left(\frac{f_\mathrm{bp}}{f_\mathrm{bp,0}}\right)^2 \tag{54}$$

$$v_\mathrm{t} = \frac{\dot{V}_\mathrm{t}}{A_\mathrm{t}} \tag{55}$$

$$N_\mathrm{Re,t} = \frac{v_\mathrm{t} d_\mathrm{t}}{\nu_\mathrm{t}} \tag{56}$$

$$f_{\mathrm{D}}^{\mathrm{t}} = \frac{1}{4\left(\log_{10}\left(\frac{5.74}{N_{\mathrm{Re,t}}^{0.9}} + \frac{\epsilon_{\mathrm{t}}/d_{\mathrm{t}}}{3.7}\right)\right)^2} \tag{57}$$

$$\Delta p_{\mathrm{f}}^{\mathrm{t}} = \ell_{\mathrm{t}} \cdot f_{\mathrm{D}}^{\mathrm{t}} \frac{\rho_{\mathrm{t}}}{2} \frac{v_{\mathrm{t}}^2}{d_{\mathrm{t}}}. \tag{58}$$

Again, we need to know the initial condition of the differential variable ($\dot{V}_{\mathrm{t}}$), the inputs ($\chi_{\mathrm{w}}^{\mathrm{m}}$, f_{bp}, p_{m}, p_{s}), and the parameters.

3.5. Combined Model

For illustration, we use two vertical pipes, one manifold, and one horizontal transport pipe from the manifold to the separator; the authors of Sharma and Glemmestad [2] use four vertical pipes, one manifold, and two horizontal transport pipes. Both Modelica and Julia's ModelingToolkit have support for building classes/reusable models. Because of the similarity between the models for vertical and horizontal pipes, it would be possible to collect these in the same class/constructor and to just differentiate between them with a function argument. The manifold model should be a separate class, though.

With re-usability of such classes/constructors, modeling of the combined system simply consists of (i) instantiating one model per unit (two vertical pipes, one horizontal transport pipe, and the manifold), and (ii) connecting the various instances. Specifically, the vertical pipes should see the same manifold pressure p_{m}, the vertical transport pipe should have the same inlet pressure as the manifold pressure p_{m}, and the influent volumetric flows to the manifold should be the sum of the flows from the vertical pipes. The viscosity diluting water feed $\dot{V}_{\mathrm{w}}$ now being

$$\dot{V}_{\mathrm{w}} = \frac{\sum_{i=1}^{2}\left(\chi_{\mathrm{w}}^{\mathrm{m}} - \chi_{\mathrm{w}}^{i}\right)\dot{V}_{\mathrm{v}}^{i}}{1-\chi_{\mathrm{w}}^{\mathrm{m}}}, \tag{59}$$

the effluent volumetric flow from the manifold is still $\dot{V}_{\mathrm{t}}$.

For a proper re-usable implementation, the connections should be performed using *connectors* (supported by both Modelica and ModelingToolkit). Connectors are not implemented here.

4. Simulation Tools

Modelica is a mature language dating back to the 1990s; ModelingToolkit [MTK] is some 4–5 years old and is still evolving rapidly. MTK is more general than Modelica, and is also integrated in the larger eco-system of a general scripting language (Julia). Currently, MTK does not support a free graphical flow-sheeting tool, and it is unclear whether MTK allows for as large models as OpenModelica. Both tools have extensive support for building libraries.

The combined model was solved using the free languages/tools OpenModelica [14,15] and ModelingToolkit [16] for Julia. To illustrate the similarity between the OpenModelica code and a current formulation using MTK, parts of the Modelica code and the MTK code for the reservoir heel–to–manifold are provided in Appendix B. The listings show that the Modelica code and the ModelingToolkit code have a high degree of similarity. A few things to note:

1. In Modelica, the independent temporal variable has a fixed name (`time`), and the time differentiation operator has a fixed name (`der`). In ModelingToolkit, both of these can be freely named by the user. In order to make the unit models work together (e.g., in a standard library), it is, however, necessary to standardize on a name for time (commonly `t`); differentiation can be given a name, e.g., `Dt = Differential(t)` or similar.
2. In Modelica, the quantities need to be specified with a type (e.g., `Real`), and are prepended with a qualifier (e.g., `constant`, `parameter`) except for the variables. For

Julia and ModelingToolkit, the data type is inferred, unless explicitly stated. In the code (Appendix B), quantities in MTK are grouped within `begin...end` blocks in *macros* (identifiers prepended by `@`, e.g., `@parameters`).

3. Modelica has a simple way to handle implicit algebraic equations, and in many cases an initial guess of the algebraic variable is not required (see variable `p_c__i` in the Modelica code). In ModelingToolkit, the initial values for the unknowns after structural simplification ("states") must be provided with numeric values (see variable `p_c__i` in the MTK code, Appendix B).
4. In ModelingToolkit, the initial values of the differential variables can be changed outside of the code; hence, default values can be written as `Vd_v(t)=23.15e-3`. In Modelica, only *parameters* can be changed outside of the code (after compilation); hence, a parameter has been defined to hold the default initial value `Vd_v(start = Vd_v0, fixed = true`.
5. Modelica uses symbol = for mathematical equality; MTK uses symbol ~ since Julia already uses symbol = for assignment.

The default solver in OpenModelica is excellent, although here it struggled if the model was posed as a DAE formulation with *momentum* as a differential variable. ModelingToolkit can use solvers from the large, high-quality DifferentialEquations.jl package [17]. With ModelingToolkit, more thought is currently required when choosing the solver, accuracies, etc., compared to OpenModelica. Also, OpenModelica handles step-changes in inputs well, while for the DifferentialEquations.jl solvers, it is often necessary to specify the time points where the step changes occur. On the other hand, the solutions from ModelingToolkit include interpolation functions, which yield smooth solutions with considerably fewer data points than for Modelica.

The results presented in Section 5 compare numerical solutions for the reservoir heel-to-manifold system for ModelingToolkit vs. OpenModelica.

OpenModelica's support for linearization and plotting can be accessed from Julia via the OMJulia API [18]. ModelingToolkit is integrated in the Julia eco-system, with support for linearization, plotting, control systems analysis, random variables, etc., and overall has more possibilities than OpenModelica if further analysis is required.

OpenModelica is currently reported to handle models up to approximately 10^6 variables/equations. Various conference presentations indicate that ModelingToolkit currently can solve models of up to approximately 10^5 variables/equations (on-going work on a JuliaSimCompiler.jl for a commercial extension of Julia will increase the possible system size). The model above (`Reservoir_2_Manifold`) is reported by OpenModelica to have 15 variables (differential+algebraic) and 15 equations. In ModelingToolkit, linear equations with "observed" variables are stripped off from the model (function `structural_simplify()`) before solving the model. The above `Reservoir_2_Manifold` model in the listing is reported to have two "states" and two equations by ModelingToolkit. It is not clear whether the ModelingToolkit claim of 10^5 variables/equations is before or after the "observed" variables are stripped off.

Other commonly used languages for scientific computing are MATLAB (commercial) and Python (free). Compared to both of these languages, Julia (free) has a more extensive set of differential equation solvers (Julia's DifferentialEquations.jl package can be accessed from Python and R). Neither MATLAB nor Python offer equation-based modeling languages with library/re-use support as do Modelica or ModelingToolkit; MathWorks does offer Simscape (commercial) with MATLAB integration for such use, however (https://se.mathworks.com/products/simscape.html; accessed on 13 December 2023).

5. Results

5.1. Reservoir Heel to Manifold

The parameters, initial conditions, and system inputs are given in Appendix A. Figure 4 shows the input variation for the reservoir heel–to–manifold (`R2M`) case.

A step change in the formation pressure (red curve), as in Figure 4, is not very realistic; such changes are normally slow. The manifold pressure (blue curve) is not normally an input function, but rather is a dependent variable in the overall system, as in Section 5.2, and, thus, also normally varies slowly. The pump speed (green curve) is, however, a control variable, and can change rapidly. Nonetheless, the inputs in Figure 4 help provide useful information about time constants in the system.

Figure 5 shows the response in (vertical) volumetric flow rate $\dot{V}_{\mathrm{v}}$, with comparison between Julia (red, solid) and OpenModelica (blue, dash-dot).

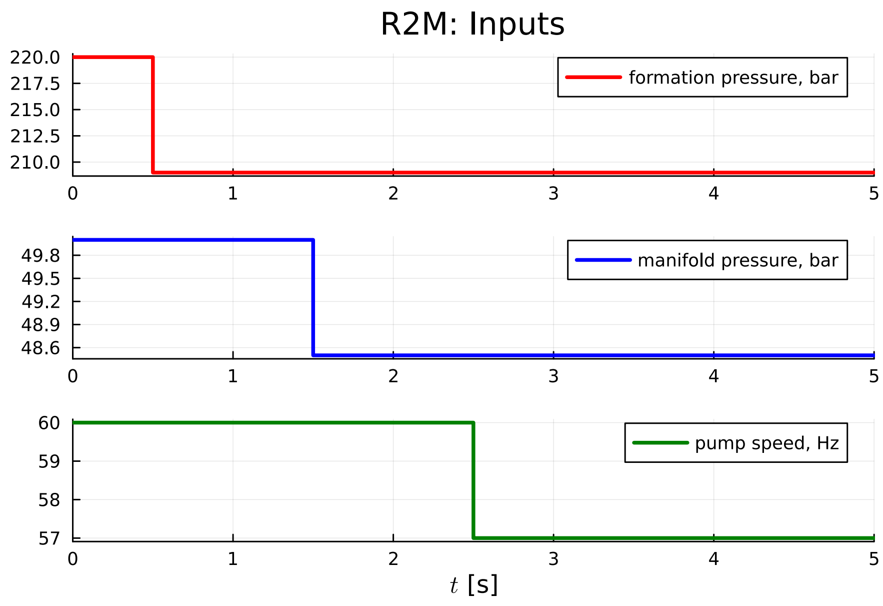


Figure 4. Input variation in experiment; Table A5.

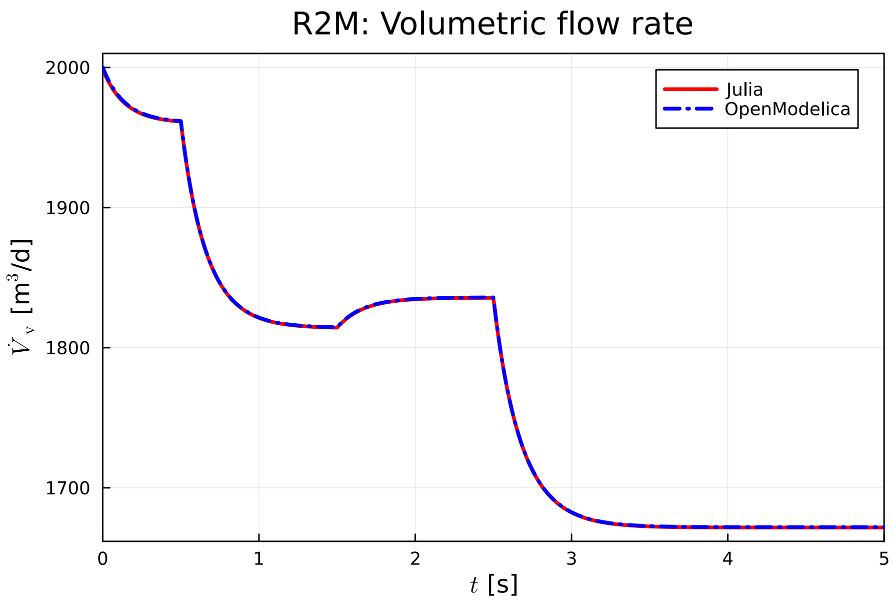


Figure 5. Response in volumetric flow rate to step inputs.

An important observation is that the volumetric flow rate is *continuous* under the step changes in Figure 4. This makes sense, in that the momentum of the fluid (oil-water) is substantial. Time constants are in the range of 0.2–0.5 s.

Figure 6 shows the response in the choke valve inlet pressure, $p_{\mathrm{c}}^{\mathrm{i}}$.

Observe that the choke valve inlet pressure for many types of step changes is continuous, but that it changes *discontinuously* upon a step change in manifold pressure at $t = 1.5$ s.

Again, this makes sense—when the manifold pressure drops (see Figure 4), the choke valve inlet pressure must also drop in proportion so that the flow through the valve changes continuously (see Equation (14)).

So far, ModelingToolkit/Julia and OpenModelica have provided virtually identical simulation results (Figures 5 and 6). Figure 7 shows the pipe friction loss, Δp_f.

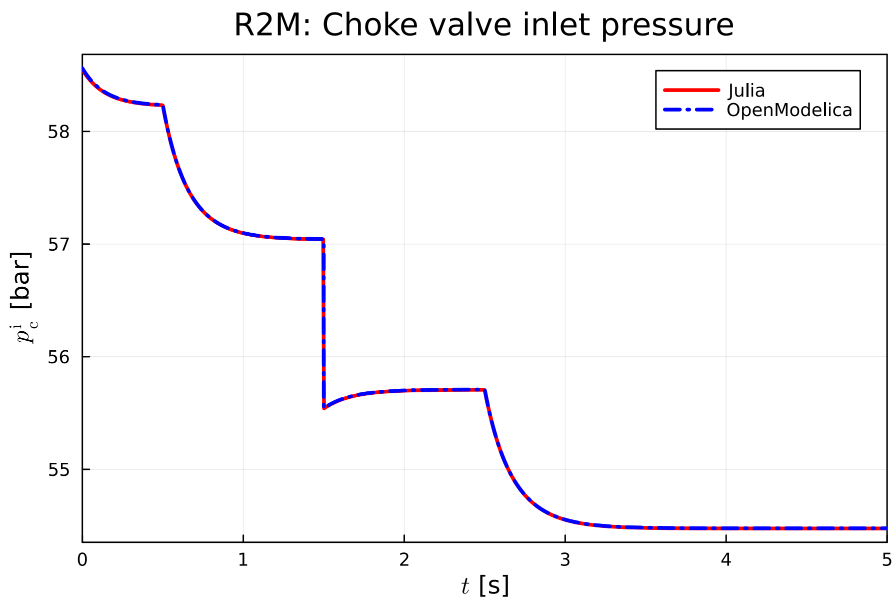


Figure 6. Response in choke valve inlet pressure to step inputs.

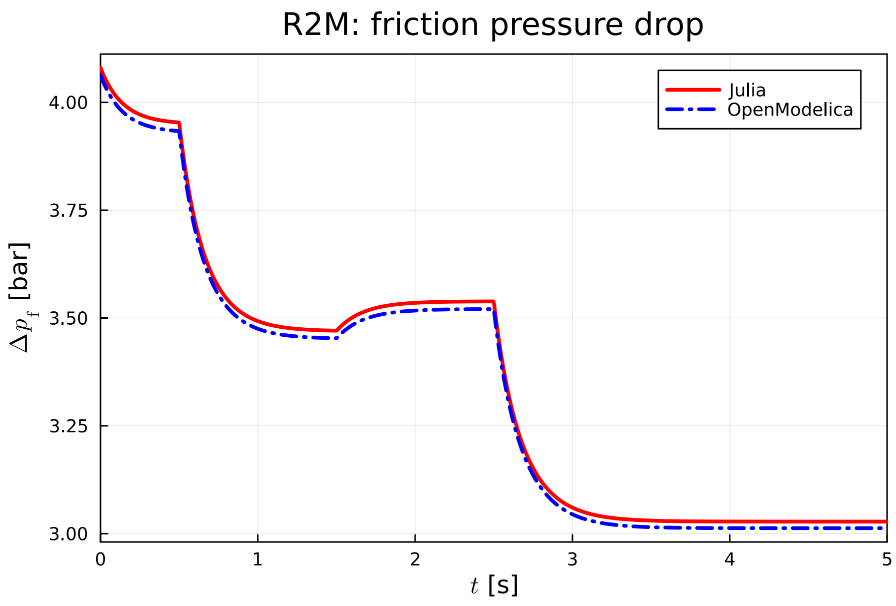


Figure 7. Response in pipe friction pressure drop.

It is interesting to contrast the magnitude of the friction loss in Figure 7 and how much smaller it is than the pressure boost in the ESP. Obviously, if a more realistic viscosity model than linear interpolation had been used, and in particular if emulsification occurs [8], the friction pressure drop might increase considerably with an unfortunate mixing fraction of oil and water.

Apart from this, it is interesting to observe a slight discrepancy between the result from ModelingToolkit/Julia and OpenModelica in this case. This discrepancy is maintained during a multitude of tests with different solvers and accuracy for the DifferentialEquation.jl solvers [17] and the solvers supported by OpenModelica. It appears that there is a minor discrepancy at $t = 0$ for Δp_f, which propagates throughout the solution. Because the codes and the initial conditions are the same for both implementations, a natural suspicion is that the difference is due to different handling of the implicit algebraic equation at the initial time, and that Δp_f is rather sensitive to such an inaccuracy.

5.2. Reservoir Heel-to-Separator

The parameters, initial conditions, and system inputs are given in Appendix A. For vertical pipe #2, the scaling pump head $h_{\mathrm{p},0}$ is set to 80% of the value suggested in Appendix A. For this more complete system (two vertical pipes, one horizontal transport pipe), there is no observed difference between the solution from ModelingToolkit/Julia and OpenModelica.

Figure 8 shows the input variation for the reservoir heel–to–separator (`R2S`) case. Because of slower dynamics for this larger system, the locations of the step changes have been changed, and the step change in the manifold pressure (Figure 4) has been replaced by a step change in the separator pressure (Figure 8).

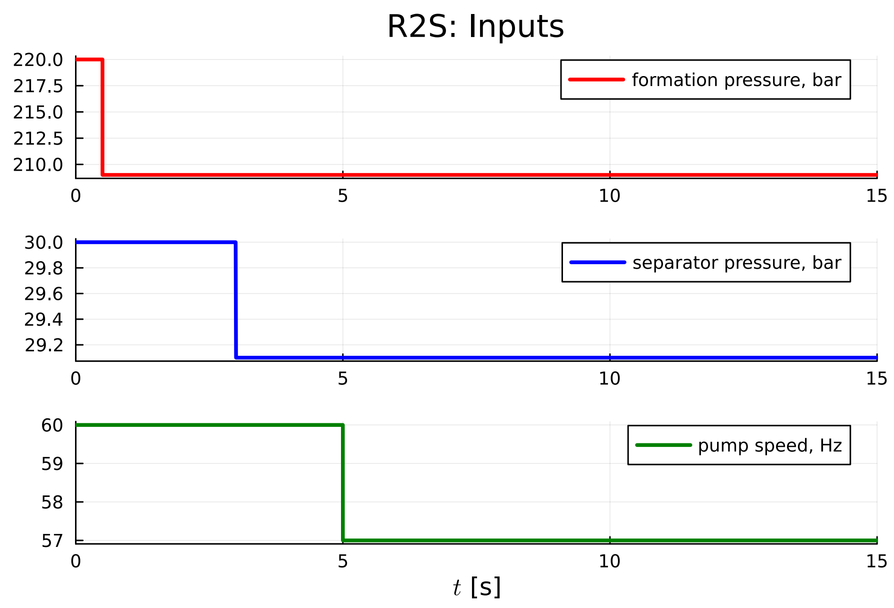


Figure 8. Input variation in experiment; Table A6.

For comments on the formation pressure and pump speed inputs, see Section 5.1. Although the separator pressure is not normally an input function, there may be action applied to the separator that may create relatively quick changes in the separator pressure.

Figure 9 shows the pressures in front of the choke valves for the vertical pipes, as well as the manifold pressure.

Figure 9 demonstrates the positive pressure drop over the choke valves, and that they are different for the two valves ($\Delta p_\mathrm{c}^j = p_\mathrm{c}^{\mathrm{i},j} - p_\mathrm{m}$). Therefore, one should expect different flows through the two valves. Because the manifold pressure is a dependent (dynamic) variable in this case, there is no discontinuity in the pressures of Figure 9. In general, for this extended system there is no visible difference between the OpenModelica and Julia solutions.

The resulting time constants and the overall behavior for the extended system (represented by Figure 9) are similar to those in Sharma [3]. Note that in Figure 9, some oscillatory behavior/overshoot is seen. This is to be expected due to the elasticity of the oil/water mixture with the given non-zero value of the isothermal compressibility β_T.

In order to understand the effect of parameter uncertainty in a model, Monte Carlo simulations are useful. Figure 10 shows the vertical flow rates from the reservoir to the manifold in the two pipes, as well as the flow from the manifold to the separator (thick, solid lines), and the effect of uncertain productivity indices in Well 1, $\dot{V}^{c,1}_{pi} \sim \mathcal{N}(7 \times 10^{-4}, 10^{-4})$, and uncertain isothermal compressibility in the petroleum fluid, $\beta_T \sim \mathcal{U}_{[0.3/1.5\times10^9, 3/1.5\times10^9)]}$.

ModelingToolkit has support for efficient Monte Carlo studies; this is comparatively more complicated using Modelica + OMJulia.

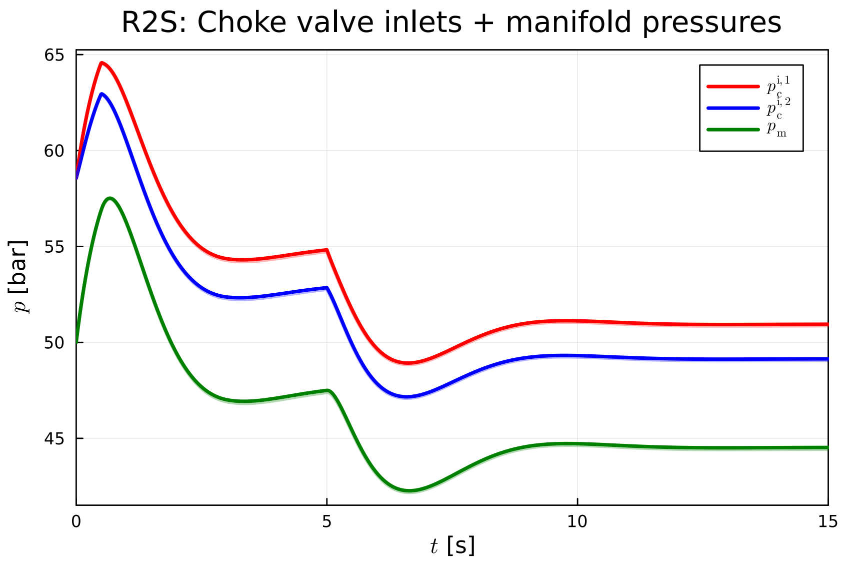


Figure 9. Pressures in front of choke valve into manifold for vertical pipes (red, blue) and manifold pressure (green).

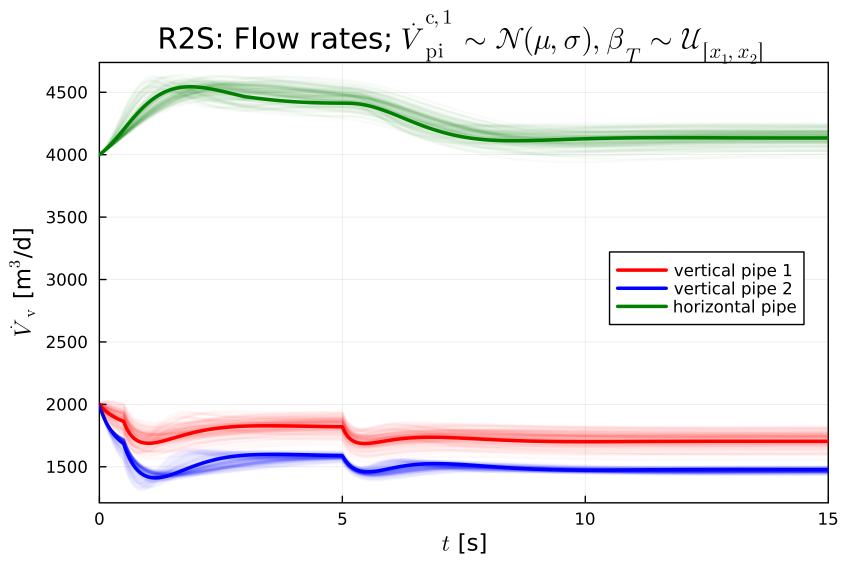
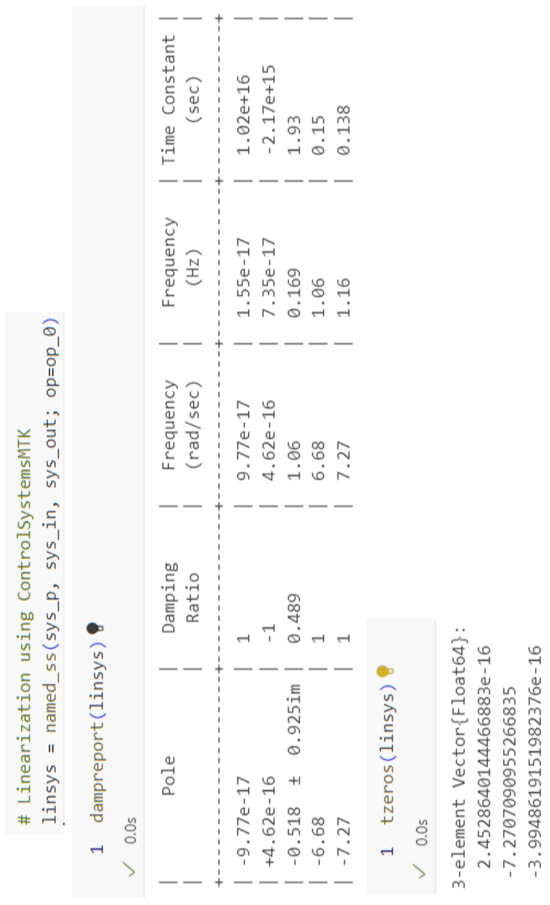


Figure 10. Vertical flow rates (red, blue) from bore-well into manifold, and horizontal flow rate (green) from manifold to separator, with uncertainty productivity capacity and isothermal compressibility.

5.3. Linearized Model

ModelingToolkit.jl has good support for the linearization of models. To linearize a system, it is necessary to provide a model where the inputs have not been defined (`sys_p` in Figure 11), a vector of input variables (`sys_in`), a vector of output variables (`sys_in`), and an operating point (keyword `op`, value `op_0`). If the ModelingToolkitStandardLibrary.jl (similar to Modelica's Standard Library) is used, alternatively, some "virtual" inputs/outputs can be added in the form of *analysis points* which simplifies linearization; this is not discussed further here.

In Figure 11, linearization is performed using the `named_ss` function in ControlSystemsMTK.jl, which has similar arguments as the function `linearize` in ModelingToolkit.jl.



```
# Linearization using ControlSystemsMTK
linsys = named_ss(sys_p, sys_in, sys_out; op=op_0)
```

```
dampreport(linsys)
```

```
|        Pole        |   Damping     |   Frequency   |   Frequency   | Time Constant |
|                    |    Ratio      |   (rad/sec)   |     (Hz)      |     (sec)     |
+--------------------+---------------+---------------+---------------+---------------+
| -9.77e-17          |  1            |  9.77e-17     |  1.55e-17     |  1.02e+16     |
| +4.62e-16          |  -1           |  4.62e-16     |  7.35e-17     |  -2.17e+15    |
| -0.518  ±  0.925im |  0.489        |  1.06         |  0.169        |  1.93         |
| -6.68              |  1            |  6.68         |  1.06         |  0.15         |
| -7.27              |  1            |  7.27         |  1.16         |  0.138        |
```

```
tzeros(linsys)
```

```
3-element Vector{Float64}:
  2.4528640144466883e-16
 -7.270709095552668 35
 -3.994861915198237 6e-16
```

Figure 11. Use of ControlSystemsMTK.jl and ControlSystems.jl for linearization and analysis.

Figure 11 suggests six poles/states in the system, and three transmission zeros. Obviously, the system has four states/differential variables (flow rates $\dot{V}_\mathrm{v}$ for each of the vertical pipes, pressure p_m for the manifold, and flow rate $\dot{V}_\mathrm{t}$ for the horizontal transport pipe). Comparing the poles and zeros, one might suspect that two spurious/"infinitesimal" poles have been added together with two spurious/"infinitesimal" zeros, and that canceling out the tiny poles and zeros should give the correct transfer function. Observe also that there is a *finite* zero at -7.27 that cancels out one of the four true eigenvalues.

It is possible to write one's own linearization code using a (symbolic) Jacobian function applicable to ModelingToolkit.jl models. If one assumes that the original model is a DAE of index 0 or 1, this will indeed give the correct transfer function with four states (and one zero that cancels out one of the eigenvalues). Why does ModelingToolkit.jl produce spurious additional poles/zeros? Possibly because the linearization algorithm in ModelingToolkit.jl makes no assumption of the index of the DAE, thus producing the two spurious "infinitesimal" poles/zeros.

Figure 12 shows a Bode plot of the transfer function from the ESP rotational speed (f_p) (it is assumed that the same speed is used for both ESPs in the vertical pipes) to the flow rate into the separator ($\dot{V}_\mathrm{t}$), as found using Julia + ModelingToolkit.jl based on the transfer function provided by the system in Figure 11. The package ControlSystems.jl also has a function `balance_statespace()`, which, in combination with minimal realization, provides a transfer function with three poles or one time constant and one damped resonator,

$$P(s) \approx \frac{1.094 \times 10^{-3}}{(1+0.15s)\left(1+2\cdot 0.489\frac{s}{1.06}+\left(\frac{s}{1.06}\right)^2\right)}; \tag{60}$$

(see pale curves in Figure 12).

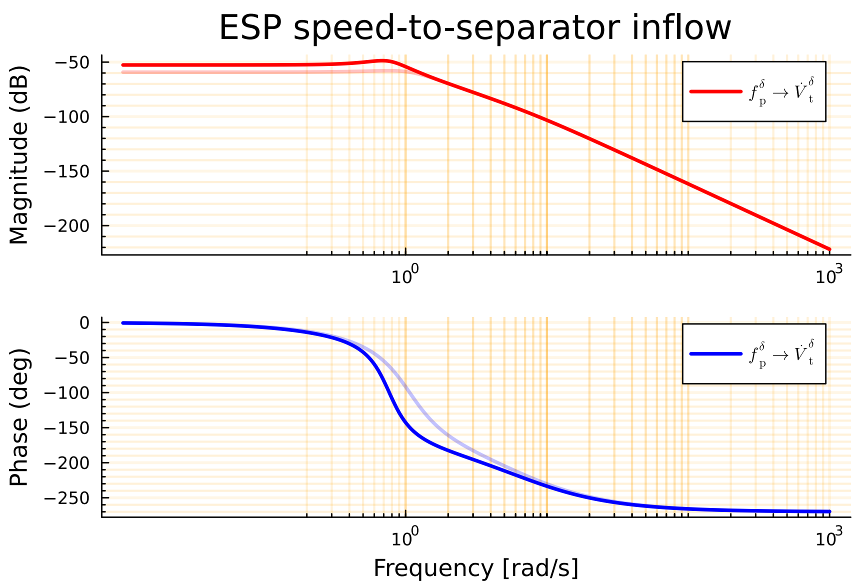


Figure 12. Bode plot of linearized model from ESP speed f_p^δ to volumetric flow $\dot{V}_\mathrm{t}^\delta$ from manifold to separator. Dark lines indicate original model; light lines the balanced-minimal model.

It is possible to instead use Modelica + OMJulia for linearization, and then use ControlSystems.jl for Julia (similar capabilities as MATLAB's Control Toolbox) for plotting and analysis/design. However, control analysis and design is simpler to do if a Julia set-up is also used for modeling and simulation.

5.4. Single-Loop Controller Tuning

Figure 13 shows a unit step response for the linearized model using the convenience function `step()` in package ControlSystems.jl.

A crude approximation of the system is read off Figure 13 as the plant transfer function $P_\approx(s)$

$$P_\approx(s) = K\frac{\exp(-\tau s)}{s} \tag{61}$$

with

$$K \approx 6.27 \times 10^{-3}$$
$$\tau \approx 0.5.$$

A PI controller

$$C(s) = K_\mathrm{p}\frac{1+T_\mathrm{i}s}{T_\mathrm{i}s} \tag{62}$$

based on Skogestad's method [19–21] is used with

$$T_\ell = 1.3 \cdot \tau \tag{63}$$
$$\kappa = 4 \tag{64}$$
$$K_\mathrm{p} = \frac{1}{K(\tau + T_\ell)} \tag{65}$$
$$T_\mathrm{i} = \kappa(\tau + T_\ell), \tag{66}$$

where the tuning parameters are T_ℓ, the closed-loop time constant, and κ, the integral time modifier.

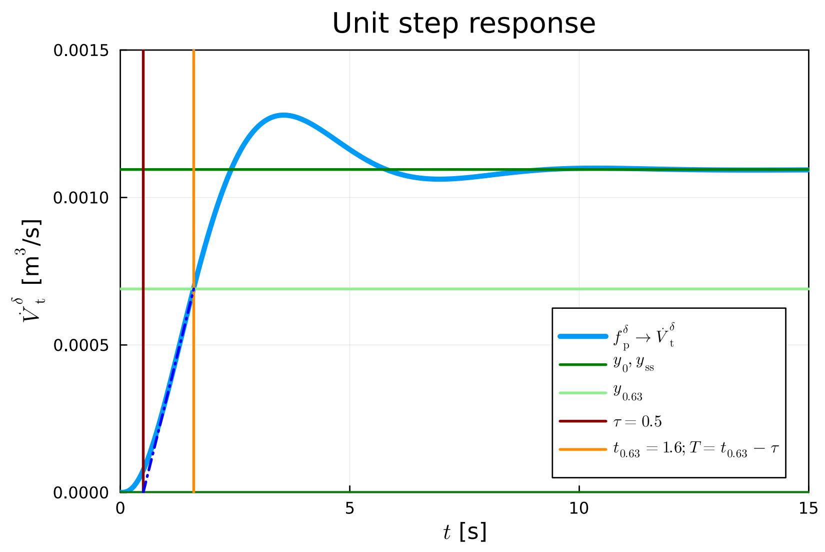


Figure 13. Response in $\dot{V}_t^\delta$ after a unit step in f_p^δ. The dark blue dash-dot line indicates the location of the time delay.

Figure 14 shows a unit reference step response for the closed-loop system [blue line]; here, the utility function `feedback(P*C)` was used to construct the closed-loop system. Observe that the closed-loop time constant T_ℓ is not achieved, and that the resulting system is rather oscillatoric.

The reason for the oscillatoric behavior is that single-loop tuning methods are not designed for oscillatoric systems such as the one described in Equation (60).

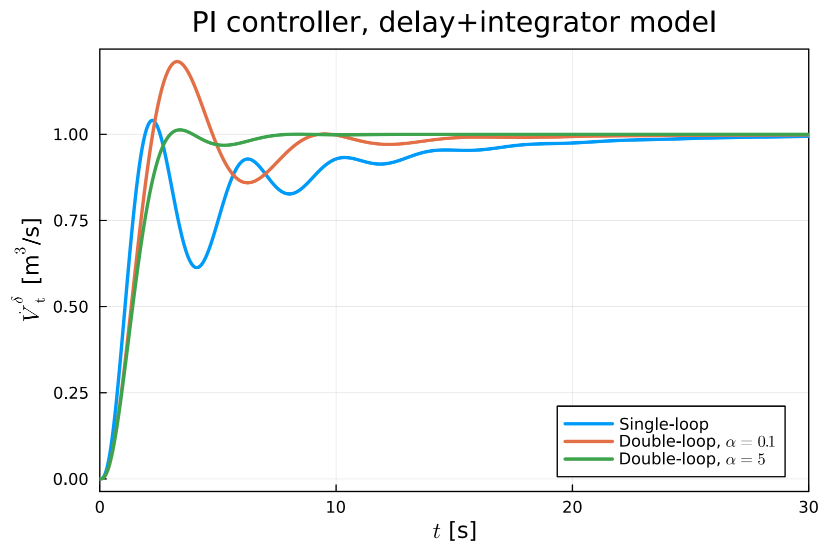


Figure 14. Response in $\dot{V}_t^\delta$ after a unit reference step assuming delay + integrator process model, using Skogestad's method: single-loop [blue], double-loop with $\alpha = 0.1$ [red], and double-loop with $\alpha = 5$ [green].

5.5. Double-Loop Controller Tuning

In order to better handle the oscillatoric behavior of the system, which has the generic form

$$P_\zeta(s) = \frac{K}{(1 + T_1 s)\left(1 + 2\zeta\frac{s}{\omega_0} + \left(\frac{s}{\omega_0}\right)^2\right)}, \tag{67}$$

an idea of Nandong [22] is pursued, where the control signal is split into an "inner" control signal

$$u_{\mathrm{i}} = -C_{\mathrm{i}}(s)y \tag{68}$$

and an "outer" control signal u_{o},

$$u = u_{\mathrm{i}} + u_{\mathrm{o}}, \tag{69}$$

where the first C_{i} is designed to reduce the oscillations in the system.

Consider the following proper inner controller

$$C_{\mathrm{i}}(s) = K_{\mathrm{c}}^{\mathrm{i}} \frac{1 + T_1 s}{1 + \alpha T_1 s}. \tag{70}$$

Inserting the controller Equation (68) into $y = P_\zeta(s)u$ with u, as in Equation (69), and $P_\zeta(s)$, as in Equation (67), leads to,

$$\begin{aligned} y = P_\zeta(s)u &= P_\zeta(s)(-C_{\mathrm{i}}(s)y + u_{\mathrm{o}}) \\ &\Downarrow \\ (1 + P_\zeta(s)C(s))y &= P(s)u_{\mathrm{o}} \end{aligned}$$

which, after some re-arrangement, gives:

$$\left(1 + \frac{KK_{\mathrm{c}}^{\mathrm{i}}}{1 + \alpha T_1 s} + 2\zeta \frac{s}{\omega_0} + \left(\frac{s}{\omega_0}\right)^2\right) y = \frac{K}{1 + T_1 s} u_{\mathrm{o}}.$$

For *small* values of α, $\alpha \to 0$:

$$\begin{aligned} \left(1 + KK_{\mathrm{c}}^{\mathrm{i}}\right) + 2\zeta \frac{s}{\omega_0} + \left(\frac{s}{\omega_0}\right)^2 &= \left(1 + KK_{\mathrm{c}}^{\mathrm{i}}\right) \times \\ &\cdots \times \left(1 + 2\zeta_{\mathrm{i}} \frac{s}{\omega_{\mathrm{i}}} + \left(\frac{s}{\omega_{\mathrm{i}}}\right)^2\right), \end{aligned}$$

where the closed-loop damping ζ_{i} is given by

$$\begin{aligned} \zeta_{\mathrm{i}} &= \zeta / \sqrt{1 + KK_{\mathrm{c}}^{\mathrm{i}}} \\ &\Downarrow \\ K_{\mathrm{c}}^{\mathrm{i}} &= \frac{(\zeta/\zeta_{\mathrm{i}})^2 - 1}{K}. \end{aligned} \tag{71}$$

A design procedure could thus be:

1. Specify inner loop damping, $\zeta_{\mathrm{i}} \geq 1$ to provide (over-)damping.
2. Compute inner gain $K_{\mathrm{c}}^{\mathrm{i}}$ from Equation (71).
3. Choose a "small" value for α to make the above design valid.

We choose $\zeta_{\mathrm{i}} = 1$, and compute $K_{\mathrm{c}}^{\mathrm{i}}$ from Equation (71). Next, closing the loop from u_{o} to y, where we allow for $\alpha > 0$ to have a realizable controller, the step response is as in Figure 15.

Based on a closed inner loop with $\alpha = 0.1$, as in the lower plot of Figure 15, we find the model parameters in the model of Equation (61) to be:

$$\begin{aligned} K &\approx 0.762 \times 10^{-3} \\ \tau &\approx 0.5. \end{aligned}$$

Tuning an outer loop PI controller with Skogestad's method with an integrator + delay approximation, this time with $T_\ell = 2 \cdot \tau$, the result is as in Figure 14 (red curve). Although the result is considerable faster than what is achieved with the single-loop controller (same figure, blue curve), the result is surprisingly oscillatoric.

It is interesting to observe that with the same value for $T_\ell = 2 \cdot \tau$, and changing α from $\alpha = 0.1$ to $\alpha = 5$, this gives a considerably better response (see Figure 14, green curve).

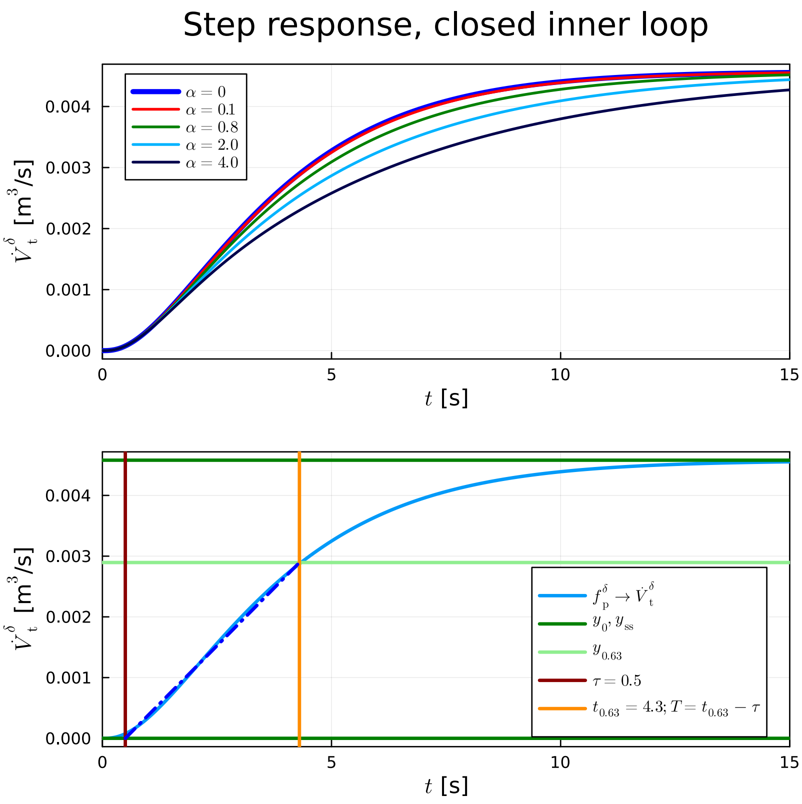


Figure 15. Response in $\dot{V}^\delta_t$ after a unit step in u_o. Upper plot: effect of varying α. Lower plot: indicating steady state and 63% rule for time constant with $\alpha = 0.1$. The dark blue dash-dot line indicates the location of the time delay.

5.6. Controller Implementation with Non-Linear Model

The inner loop controller of Equation (68) with $C_i(s)$ as in Equation (70) admits the following state space realization

$$\frac{\mathrm{d}z_i}{\mathrm{d}t} = -\frac{1}{\alpha T_1} z_i + y$$
$$u_i = \frac{K^i_c}{\alpha^2 T_1}(1-\alpha) z_i - \frac{K^i_c}{\alpha} y,$$

where we must require $\alpha > 0$. The PI controller for the outer loop as in Equation (62) admits the following state space realization:

$$e = r - y$$
$$\frac{\mathrm{d}z_o}{\mathrm{d}t} = \frac{1}{T^o_i} e$$
$$u_o = K^o_p (e + z_o).$$

With $y \leftarrow \dot{V}_t$ as input to the controller together with a reference value $r \leftarrow \dot{V}^{\mathrm{ref}}_t$, the controller produces output $u = u_i + u_o \to f_p$ and is straightforward to implement in ModelingToolkit (or Modelica) and to connect with the reservoir-to-separator model. Using the inner–outer model with $\alpha = 5$, starting in a steady state and injecting a step change in $\dot{V}^{\mathrm{ref}}_t$ gives the response as shown in Figure 16.

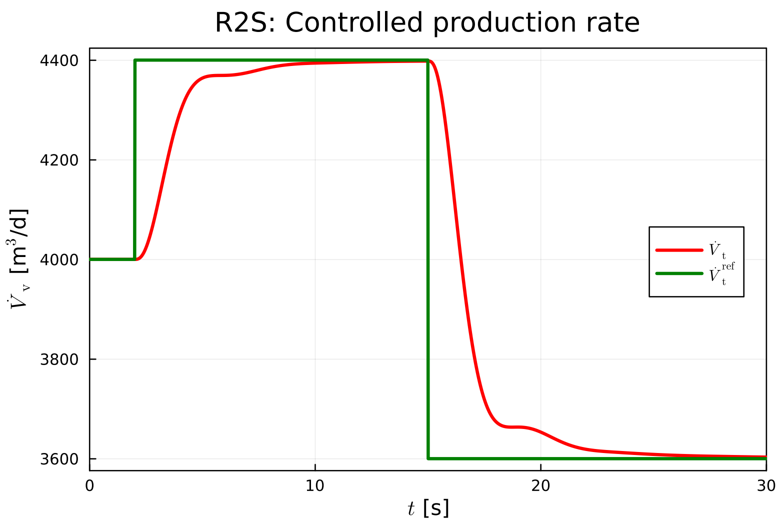


Figure 16. Response in $\dot{V}_t$ after a unit step in $\dot{V}_t^{ref}$, with $\alpha = 5$ in Equation (70).

The response in Figure 16 (red curve; nonlinear model) is similar to the corresponding response (green curve; linear approximation) in Figure 14.

The response in Figure 16 is achieved with an ESP pump speed as in Figure 17.

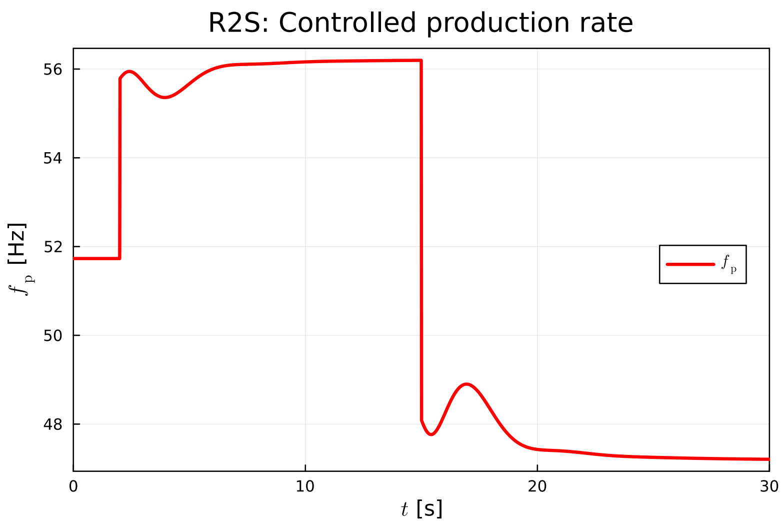


Figure 17. ESP pump speed f_p, with $\alpha = 5$ in Equation (70).

6. Conclusions

Sharma and Glemmestad [2], Sharma [3] provided a novel dynamic model of an ESP-lifted oil production system from the reservoir to the separator. Their pump models neglect the effect of the gas content and fluid viscosity variation, and they use a simplistic viscosity model in friction terms. Also, some of the information is spread over several publications, and the ESP model parameters are difficult to find.

Nevertheless, their model is used in the present work to facilitate comparison. Here, a structured development of the model in Sharma and Glemmestad [2] is provided from the fundamental balance laws with clearly stated simplifying assumptions. Their equipment models are rephrased in dimensionless form, and the merits of dimensionless forms are emphasized. A complete, rephrased mathematical model from the reservoir to the separator is provided, with all the necessary model parameters, initial states, and input functions. This achieves the goal of providing a complete model that can be implemented in a single presentation. Because of the structured model development, it is also relatively straightforward to replace the equipment models with alternative models.

The implementation of realistic and complex oil field topologies is greatly simplified by using modeling languages that facilitate use of model libraries and the re-use of models. The rephrased model of Sharma [3] is implemented in two such languages, Modelica and ModelingToolkit, for comparison. Modelica is a mature language for differential algebraic models, with excellent solvers that can handle large-scale models (in the range of 10^6+ unknowns), and support for C code generation and integration via the FMI standard (FMI: functional mock-up interface, e.g., https://fmi-standard.org/; accessed on 13 December 2023) to other tools. ModelingToolkit [MTK] is newer, is in rapid development, handles a larger class of models (including distributed models), and is tightly integrated within the Julia scripting language, which has excellent support for control design, optimization, and more. It is demonstrated that the form of Modelica and MTK models can be very similar. The purpose of this work is not to implement a library unit of the model. However, the advantages of using modeling languages are clearly pointed out.

To illustrate the possibilities of modeling languages integrated with script languages, an example is given where the MTK model is linearized in Julia. Based on the linearized model, a simple PI controller from the ESP speed to the production flow rate is designed. This design is not meant for industrial use, but illustrates the simplicity of a control design workflow with modern modeling languages. The example also illustrates that standard PI tuning rules struggle with oscillatoric systems (here, due to the compressibility of the oil/water mixture), and that some additional refinement may improve the closed-loop system. However, one would anticipate that an even more complex controller, such as model predictive control [23], would further improve the performance.

The model development and discussion indicates a number of possible extensions, such as (a) more realistic properties (density, viscosity), (b) allowing for distributed properties along the pipes (ModelingToolkit for Julia has support for automatic discretization of PDEs), (c) adding a more realistic system for water dilution in the manifold, (d) inclusion of valves in the manifold–separator pipes, (e) use for advanced control design with constraints, (f) use for optimization, (g) integration with a reservoir model to study how to handle stiffness, and more. Such further work will give more insight into the industrial usefulness of the model.

Funding: The economic support from The Research Council of Norway and Equinor ASA through Research Council project "308817 —Digital wells for optimal production and drainage" (DigiWell) is gratefully acknowledged.

Data Availability Statement: Data are contained within the article.

Acknowledgments: Information/data for the original paper [2] provided by Roshan Sharma and Kushila Jayamanne, both at the University of South-Eastern Norway, is gratefully acknowledged.

Conflicts of Interest: The author declares no conflicts of interest. The background information for the study predates the funded project, and is openly available. Although the study is relevant to the funded project (DigiWell), the funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

Abbreviations

The following abbreviations are used in this manuscript:

CFD	Computational Fluid Dynamics
ESP	Electric Submersible Pump
MTK	ModelingToolkit
R2M	Reservoir Heel-to-Manifold
R2S	Reservoir Heel-to-Separator

Appendix A. Parameters and Operating Conditions

The parameters for the petroleum fluid, nominal vertical pipes, and nominal manifold + horizontal pipe are given in Tables A1–A3. The initial states are given in Table A4, while the input functions are given in Tables A5 and A6.

Table A1. Parameters: petroleum liquid.

Parameter
$\beta_T = \frac{1}{1.5\times10^9} \approx 6.67\times10^{-10}\,\mathrm{Pa}^{-1}$
$p_0 = 1\,\mathrm{bar}$
$\rho_\mathrm{o} = 900\,\mathrm{kg/m^3}$
$\rho_\mathrm{w} = 1000\,\mathrm{kg/m^3}$
$\chi_\mathrm{w} = 0.35$
$\rho_0 = \chi_\mathrm{w}\rho_\mathrm{w} + (1-\chi_\mathrm{w})\rho_\mathrm{o}$
$\chi_\mathrm{w}^\mathrm{m} = 0.5$
$\rho_0^\mathrm{m} = \chi_\mathrm{w}^\mathrm{m}\rho_\mathrm{w} + (1-\chi_\mathrm{w}^\mathrm{m})\rho_\mathrm{o}$
$\nu_\mathrm{o} = 100\,\mathrm{cSt} = 100\times10^{-6}\,\mathrm{m^2/s}$
$\nu_\mathrm{w} = 1\,\mathrm{cSt} = 10^{-6}\,\mathrm{m^2/s}$

Table A2. Parameters: vertical pipe.

Parameter
$\ell^- = 100\,\mathrm{m}$
$\ell^+ = 2000\,\mathrm{m}$
$d = 0.1569\,\mathrm{m}$
$\epsilon = 0.0018\,\mathrm{inch} = 45.7\,\mu\mathrm{m} = 45.7\times10^{-6}\,\mathrm{m}$
$\dot{V}_{\mathrm{min},0} = 14.43\times10^{-3}\,\mathrm{m^3/s}$
$\dot{V}_{\eta,0} = 19.83\times10^{-3}\,\mathrm{m^3/s}$
$\dot{V}_{\mathrm{max},0} = 25.24\times10^{-3}\,\mathrm{m^3/s}$
$h_{\mathrm{p},0} = 1210.6\,\mathrm{m}$
$f_{\mathrm{p},0} = 60\,\mathrm{Hz}$
$\dot{V}^\varsigma = 1\,\mathrm{m^3/s}$
$a_1 = -37.57$
$a_2 = 2.864\times10^3$
$a_3 = -8.668\times10^4$
$\dot{W}_{\mathrm{p},0}^\mathrm{m} = 167.733\,\mathrm{kW}$
$b_1 = 52.12$
$b_2 = -768.7$
$b_3 = 38.544\times10^3$
$b_4 = -1.534\times10^6$
$\dot{m}_\mathrm{v}^\mathrm{c} = 25.9\times10^3\,\mathrm{kg/h}$
$f(u_\mathrm{v}) = \begin{cases} 0, & u_\mathrm{v} \le 0.05 \\ \frac{11.1u_\mathrm{v}-0.556}{30}, & 0.05 < u_\mathrm{v} \le 0.5 \\ \frac{50u_\mathrm{v}-20}{30}, & 0.5 < u_\mathrm{v} \le 1 \end{cases}$
$p^\varsigma = 1\,\mathrm{bar}$
$\rho^\varsigma = 1000\,\mathrm{kg/m^3}$
$\dot{V}_\mathrm{pi}^\mathrm{c} = 7\times10^{-4}\,\mathrm{m^3/s}$

Table A3. Parameters: manifold + horizontal pipe.

Parameter
$\ell_\text{m} = 500$
$d_\text{m} = 0.1569$
$\ell_\text{t} = 4000\,\text{m}$
$d_\text{t} = 0.1569\,\text{m}$
$\epsilon = 0.0018\,\text{inch} = 45.7\,\mu\text{m} = 45.7 \times 10^{-6}\,\text{m}$
$\Delta p_\text{bp,0} = 10\,\text{bar}$
$f_\text{bp,0} = 60\,\text{Hz}$

Table A4. Nominal initial states.

Variable
$\dot{V}_\text{v}(t=0) = 2000\,\text{m}^3/\text{d} \approx 0.02315\,\text{m}^3/\text{s}$
$p_\text{m}(t=0) = 50\,\text{bar} = 50 \times 10^5\,\text{Pa}$
$\dot{V}_\text{t}(t=0) = 2000\,\text{m}^3/\text{d} \approx 0.02315\,\text{m}^3/\text{s}$

Table A5. Nominal inputs, reservoir-to-manifold case.

Variable
$p_\text{f}(t) = \begin{cases} 220\,\text{bar}, & t < 0.5\,\text{s} \\ 0.95 \cdot 220\,\text{bar}, & t \geq 0.5\,\text{s} \end{cases}$
$p_\text{s}(t) = \begin{cases} 30\,\text{bar}, & t < 1.5\,\text{s} \\ 0.97 \cdot 30\,\text{bar}, & t \geq 1.5\,\text{s} \end{cases}$
$f_\text{p}(t) = \begin{cases} 60\,\text{Hz}, & t < 2.5\,\text{s} \\ 0.95 \cdot 60\,\text{Hz}, & t \geq 2.5\,\text{s} \end{cases}$
$u_\text{v}(t) = 1.0$
$f_\text{bp} = 60\,\text{Hz}$

Table A6. Nominal inputs, reservoir-to-separator case.

Variable
$p_\text{f}(t) = \begin{cases} 220\,\text{bar}, & t < 0.5\,\text{s} \\ 0.95 \cdot 220\,\text{bar}, & t \geq 0.5\,\text{s} \end{cases}$
$p_\text{s}(t) = \begin{cases} 30\,\text{bar}, & t < 3\,\text{s} \\ 0.97 \cdot 30\,\text{bar}, & t \geq 3\,\text{s} \end{cases}$
$f_\text{p}(t) = \begin{cases} 60\,\text{Hz}, & t < 5\,\text{s} \\ 0.95 \cdot 60\,\text{Hz}, & t \geq 5\,\text{s} \end{cases}$
$u_\text{v}(t) = 1.0$
$f_\text{bp} = 60\,\text{Hz}$

Appendix B. Modelica vs. ModelingToolkit Code

It is of interest to see how similar the *structures* of the Modelica and ModelingToolkit code are. Listing AA1 shows parts of the Modelica code for the reservoir heel–to–manifold; to save space, *a description* of quantities is only included for constant π to illustrate how it is done:

Listing A1. Modelica code structure.

```
model Reservoir_2_Manifold
    // Model of Reservoir-to-Manifold
   //
   // Model constants
   constant Real PI = 3.151592654 "pi";
   constant Real g = 9.81;
   ...
   // Model parameters
   parameter Real ell_m = 100;
   parameter Real ell_p = 2000;
   ...
   // Initial state parameters
   parameter Real Vd_v0 = 23.15e-3;
   //
   // Declaring variables
   // -- differential variables
   Real Vd_v(start = Vd_v0, fixed = true);
   // -- depending on inputs
   Real rho_beta_0;
   ...
   Real p_c__i;
   Real p_h;
   ...
   // -- input variables
   input Real p_f;
   ...
  // Equations constituting the model
  equation
   // Balance equations
   der(Vd_v) = A*(p_h - p_c__i + Dp_p - Dp_f - rho_v*g*h)/(rho_v*ell);
   // Algebraic equations
   // -- depending on inputs
   rho_beta_0 = chi_w*rho_w + (1-chi_w)*rho_o;
   ...
   //
  end Reservoir_2_Manifold;
```

Next, Listing AA2 shows similar parts of the ModelingToolkit code for the reservoir heel–to–manifold:

Listing A2. ModelingToolkit code structure.

```
# Reservoir-to-manifold pipe
@mtkmodel Reservoir_2_Manifold begin
    # Model of Reservoir-to-Manifold
    #
    # Model "constants" and parameters
    @parameters begin
        # -- constants
        PI=3.141592654 , [description="pi"]
        g=9.81
        ...
        # -- parameters
        ell_m=100
        ell_p=2_000
        ...
    end
    # Dependent variables
    @variables begin
        # -- differential variable
```

```
                    Vd_v(t)=23.15e-3
                    # -- algebraic variables
                  rho_beta_0(t)
                    ...
                    p_c__i(t)=58.5e5
                    p_h(t)
                    ...
                end
                # Equations
                @equations begin
                  # Balance equation
                    Dt(Vd_v) ~ A*(p_h - p_c__i + Dp_p - Dp_f - rho_v*g*h)/(rho_v*ell)
                  # Algebraic equations
                  # -- depending on inputs
                    rho_beta_0 ~ chi_w*rho_w + (1-chi_w)*rho_o
                    ...
                end
            end
```

References

1. Lie, B. ESP Lifted Oil Field: Core Model, and Comparison of Simulation Tools. In Proceedings of the 64th International Conference of Scandinavian Simulation Society, SIMS 2023, Västerås, Sweden, 25–28 September 2023; pp. 159–166. [CrossRef]
2. Sharma, R.; Glemmestad, B. Modeling and Simulation of an Electric Submersible Pump Lifted Oil Field. *Int. J. Pet. Sci. Technol.* **2014**, *8*, 39–68.
3. Sharma, R. Optimal Operation of Gas Lifted and ESP Lifted Oil Fields: An Approach Based on Modeling, Simulation, Optimization and Control. Ph.D. Thesis, University of South-Eastern Norway, Notodden, Norway, 2014.
4. Binder, B.J.T.; Pavlov, A.; Johansen, T.A. Estimation of Flow Rate and Viscosity in a Well with an Electric Submersible Pump Using Moving Horizon Estimation. *IFAC-PapersOnLine* **2015**, *48*, 140–146. [CrossRef]
5. Krishnamoorthy, D.; Bergheim, E.M.; Pavlov, A.; Fredriksen, M.; Fjalestad, K. Modelling and Robustness Analysis of Model Predictive Control for Electrical Submersible Pump Lifted Heavy Oil Wells. *IFAC-PapersOnLine* **2016**, *49*, 544–549. [CrossRef]
6. Santana, B.A.; Fontes, R.M.; Schnitman, L.; Martins, M.A.F. An Adaptive Infinite Horizon Model Predictive Control Strategy Applied to an ESP-lifted Oil Well System. *IFAC PapersOnLine* **2021**, *54*, 176–181. [CrossRef]
7. Delou, P.d.A.; de Azevedo, J.P.A.; Krishnamoorthy, D.; de Souza, M.B., Jr.; Secchi, A.R. Model Predictive Control with Adaptive Strategy Applied to an Electrical Submersible Pump in a Subsea Environment. *IFAC PapersOnLine* **2019**, *52*, 784–789. [CrossRef]
8. Justiniano, M.; Romero, O.J. Inversion Point of Emulsions as a Mechanism of Head Loss Reduction in Onshore Pipeline Heavy Oil Flow. *Braz. J. Pet. Gas* **2021**, *15*, 13–24. [CrossRef]
9. Zhu, J.; Zhu, H.; Wang, Z.; Zhang, J.; Cuamatzi-Melendez, R.; Farfan, J.A.M.; Zhang, H.Q. Surfactant Effect on Air/Water Flow in a Multistage Electrical Submersible Pump (ESP). *Exp. Therm. Fluid Sci.* **2018**, *98*, 95–111. [CrossRef]
10. Zhu, H.; Zhu, J.; Zhang, H.Q. Mechanistic Modeling of Gas Effect on Multi-stage Electrical Submersible Pump (ESP) Performance with Experimental Validation. *Chem. Eng. Sci.* **2022**, *252*, 117288. [CrossRef]
11. Zhu, J.; Guo, X.; Liang, F.; Zhang, H.Q. Experimental Study and Mechanistic Modeling of Pressure Surging in Electrical Submersible Pump. *J. Nat. Gas Sci. Eng.* **2017**, *45*, 625–636. [CrossRef]
12. Brkić, D. Review of Explicit Approximations to the Colebrook Relation for Flow Friction. *J. Pet. Sci. Eng.* **2011**, *77*, 34–48. [CrossRef]
13. Bird, R.B.; Stewart, W.E.; Lightfoot, E.N. *Transport Phenomena*, 2nd ed.; John Wiley & Sons: New York, NY, USA, 2002.
14. Fritzson, P. *Principles of Object-Oriented Modeling and Simulation with Modelica 3.3: A Cyber-Physical Approach*, 2nd ed.; Wiley-IEEE Press: Piscataway, NJ, USA, 2015.
15. Fritzson, P.; Pop, A.; Asghar, A.; Bachmann, B.; Braun, W.; Braun, R.; Buffoni, L.; Casella, F.; Castro, R.; Danós, A.; et al. The OpenModelica Integrated Modeling, Simulation and Optimization Environment. In Proceedings of the 1st American Modelica Conference, Cambridge, MA, USA, 9–10 October 2018.
16. Ma, Y.; Gowda, S.; Anantharaman, R.; Laughman, C.; Shah, V.; Rackauckas, C. ModelingToolkit: A Composable Graph Transformation System For Equation-Based Modeling. *arXiv* **2021**, arXiv:2103.05244. https://doi.org/10.48550/arXiv.2103.05244.
17. Rackauckas, C.; Nie, Q. DifferentialEquations.Jl—A Performant and Feature-Rich Ecosystem for Solving Differential Equations in Julia. *J. Open Res. Softw.* **2017**, *5*, 15. [CrossRef]
18. Lie, B.; Palanisamy, A.; Mengist, A.; Buffoni, L.; Sjölund, M.; Asghar, A.; Pop, A.; Fritzson, P. OMJulia: An OpenModelica API for Julia-Modelica Interaction. In Proceedings of the Proceedings of the 13th International Modelica Conference, Regensburg, Germany, 4–6 March 2019; pp. 699–708. [CrossRef]
19. Skogestad, S. Simple Analytic Rules for Model Reduction and PID Controller Tuning. *J. Process. Control* **2003**, *13*, 291–309. [CrossRef]
20. Skogestad, S. Simple Analytic Rules for Model Reduction and PID Controller Tuning. *Model. Identif. Control Nor. Res. Bull.* **2004**, *25*, 85–120. [CrossRef]

21. Haugen, F. Comparing PI Tuning Methods in a Real Benchmark Temperature Control System. *Model. Identif. Control* **2010**, *31*, 79–91. [CrossRef]
22. Nandong, J. Double-Loop Control Structure for Oscillatory Systems: Improved PID Tuning via Multi-Scale Control Scheme. In Proceedings of the 2015 10th Asian Control Conference (ASCC), Kota Kinabalu, Malaysia, 31 May–3 June 2015; IEEE: Piscataway, NJ, USA, 2015. [CrossRef]
23. Maciejowski, J.M. *Predictive Control with Constraints*; Prentice Hall: Harlow, UK, 2002.