DAS Departamento de Automação e Sistemas
CTC Centro Tecnológico
UFSC Universidade Federal de Santa Catarina

Optimal oil production network control using Modelica

Monografia submetida à Universidade Federal de Santa Catarina

como requisito para a aprovação da disciplina:

DAS 5511: Projeto de Fim de Curso

Marco Aurélio Schmitz de Aguiar

Florianópolis, agosto de 2013

Aknowlegment

I would like to express my special thanks of gratitude to my supervisor at NTNU, Andres Codas Duarte, as well to my supervisor at UFSC, Professor Eduardo Camponogara, who gave knowledge, advices and incentive to do this project.

I would also like to thank my parents and friends who helped to keep cheerful and focused, even from distance.

I would like to thank the NTNU, the Department of Engineering Cybernetics, and the IO Center for receiving me and giving me the support so I could have an nice stay and a good work environment.

At last, I would like to thank to Brazilian Government and the CNPQ for giving me the opportunity for this exchange program called “Ciência sem fronteiras” (Science without borders). Which I believe can be very beneficial to transform the country in one of the leading centers for scientific research.

Resumo

Na ultima década, diversos trabalhos desenvolveram técnicas para otimização estática da produção. Ao mesmo tempo, engenheiros de controle vem usando otimização para controlar a dinâmica de poços, separadores e outro elementos da rede de produção. Este trabalho visa preencher o *gap* entre essas duas áreas distintas. Ele tenta responder uma questão comum que surge ao desenvolver problemas de otimização estática, o que está ocorrendo com as dinâmicas do sistema?

Para esse propósito, um modelo dinâmico da rede de produção de petróleo foi desenvolvido. Ele contém os principais elementos: poço, *manifold*, *pipeline*, separador, compressor de gás e linha de injeção de *gas-lift*. Uma linguagem de modelagem de sistemas chamada *Modelica* foi usada para transcrever estes modelos para eles serem usados no ambiente computacional JModelica.org. Esse ambiente conecta ferramentas do estado da arte para simulação e otimização de sistemas dinâmicos.

Além de criar uma ferramenta que possibilite a compreensão do que está ocorrendo com as dinâmicas do sistema, este trabalho se propõe a resolver um problema que não seja notado em outras formulações. Um problema no qual uma manutenção programada dos compressores reduzirá sua capacidade de fluxo, se não manejado de forma adequada, poderá levar a um transitório onde é necessário queimar parte da produção no *flare*.

Três problemas de otimização são formulados e comparados. O primeiro, uma formulação simplista que faz o *tracking* de um *setpoint*. Na sequência, é criado um problema de maximização da produção. Por último, é formulada uma combinação de *tracking* com minimização do uso do *flare*.

Os problemas de otimização são discretizados usando o *collocation method*, um método que faz a formulação explícita do problema e descarta o uso de simuladores. A otimização é realizada usando o *nonlinear solver IPOPT*. Os Jacobianos, as Hessianas e os gradientes são obtidos com a ferramenta de diferenciação automática *CasADi*.

Após a analise dos resultados, verificamos que é necessário o tratamento dinâmico do problema e a ausência desta abordagem leva a condições na qual o *flare* é usado. Para evitar tal situação tanto a formulação de maximização de produção, quanto a formulação de *tracking* com alta penalização do *flare* podem ser usadas.

Abstract

In the last decade several works have developed techniques to solve the steady state production optimization problem. At the same time, control engineers have used optimization to control the dynamics of wells, separators and other production network elements. This work aims to fill the gap between this two distinct areas of study. It tries to answer a common question that arises when developing static optimization problems, what is happening with the system dynamics?

For this, a dynamic model of an oil and gas production network was developed. It contains the main elements: wells, manifolds, pipelines, separators, gas compressors and gas-lift injection line. A system modelling language named Modelica was used to transcript these models, so they could be used in the computational environment JModelica.org. This environment connects state-of-the-art tools to simulation and optimization of dynamic systems.

Beside creating a tool that allows the understanding of what is happening in the dynamic domain, this works proposes to solve a dynamic problem that might be unnoticed in other formulations. A problem where a scheduled compressors maintenance will reduce the compressors flow capacity and if not well handled may lead to a temporary flare condition.

Three optimization problems are formulated and compared. At first an naive approach that only tracks a setpoint is proposed. After, a production maximization problem is created. At last, a combination of tracking and flare minimization is suggested.

Each optimization problem is discretized using the collocation method, a explicit formulation that discards the use of simulators. The optimization is solved using the IPOPT nonlinear solver. The Jacobians, Hessians, and gradients are obtained using the CasADi automatic differentiation tool.

After the analysis of the results, it is seen that the an approach that treats the dynamics is necessary and the lack of such treatment leads to a flaring condition. Both, the production maximization and the tracking formulation with high penalization, were able to solve the problem without using the flare.

Contents

Chapter 1: Introduction

A current trend in the science and society is the search for more efficient and effective means to explore the natural resources. This trend combined with a uprising demand for fossil energy culminated in the necessity of the oil industry to enhance the development of new technologies.

Between these technologies, the “smart field” [24] contrasts among the others. This technology develops hardwares and softwares to allow the optimal operation of oilfields. The development became a challenge to the industry and the academic partners. In the software part, many works have been developing nonlinear [21, 2] and piecewise linear [14, 9, 30] formulations for maximize the static production of oil fields.

Static models are easier to be formulated when compared to dynamic models, it can be done by interpolating simulators or field data. The optimization of a static can obtain results faster than dynamic models, but it ignores how the system is behaving in the continuous time.

In the dynamic system area, several works model and control a single element of network, but hardly more than one element is treated per time. In this work, an oil production network is modeled but for this dynamic models are used. In such way, it aims to be able to solve problems that requires the whole network without missing the transitory characteristic.

The production is composed by:

- Gas-lift wells that extract oil, gas, and water from the reservoir.
- Sub-sea manifolds that gathers the flow from different wells.
- Pipelines and risers that transport the flow from the sea bed to the processing unit.
- Separators that separate the incoming three phase flow to three different outlets, the separation efficiency may change depending on the conditions.
- Compressors that increases the pressure from the separator gas outlet to the exportation line.

Gas-lift injection line and manifold that take part of the flow of the exportation line and redirect then to be inject in the wells.

The models equations were coded using a system modeling language named Modelica, this language permit the description of systems in a intuitive way. The computational environment JModelica.org was used to compile the production network model, this environment also interface the model with state-of-the-art simulation and optimization tools.

This work may be relevant to answer some of questions that may rise when doing static optimization, however the tool developed allows to go further create new formulations. A practical problem is created to explore the system model and it possible applications it is created. A scheduled compressor maintenance will restrict the compressor flow capacity. In such condition, it might be necessary to flare part of the production if the problem is not well handled.

To solve this problem three optimization formulations were created. At first, it is created a optimal tracking problem that follows the static production maximization solution. The second approach is a production maximization problem. The third formulation combines the track following of the first formulation with a flare minimization. Each formulation is compared qualitative using figures and qualitative using the solver report.

The model is discretized using the collocation method, an explicit formulation for solving DAEs that discard the use of simulators. The open-source nonlinear solver IPOPT is used to solve the optimization problems. the CasADi automatic differentiation tool is used to obtain the Jacobians, Hessians, and gradients.

Chapter 2 describes the models used and how they are coupled. Chapter 3 discusses about the tools used to accomplish this work. Chapter 4 shows the discretization method and the nonlinear formulation. Chapter 5 show and argument the results. Chapter 6 gives the final conclusions and the possibility for future works.

Chapter 2: Model and Background

In this chapter, the models that were used to describe the oil network will be formulated.

The chapter begins with an overview of the main elements of network and how they interact. Afterwards each of the models is specified based on literature references and considering some changes to fit the purpose of this work. At last it is given a description of how the models are connected to form the network.

2.1: Network

This works wishes to develop a model that is suitable for simulation, control, and optimization. For this, not only the component dynamics must be considered but also the interactions between them and the restrictions that it may lead.

For instance, the literature usually assumes that there is unlimited lift-gas availability or it is bounded by some arbitrary value. However, when analyzing the network as a compound system we can see that the limit on gas injection depends on the mass balance of how much is produced, exported, and injected. For these reasons a macroscopic view of the system is necessary to rightfully control and optimize a network.

The network starts with the producing elements, the well takes as input the lift-gas flow rate and as output, the production mass flow. Besides, it is related to the pressure with the gas-lift manifold and production manifold.

The production manifold gathers the flow from many inlets and redirects to only one outlet. This is used to connect the wells to the pipeline.

The pipeline and the riser are modeled together, since they have similar functionality and dynamics. As input it takes the flow coming from the well and delivers it to the separator. It shares the pressure with the manifold at the inlet and with the separator at the outlet. Between the inlet and outlet a pressure drop is created due to friction and gravitational effects.

The separator takes the three phase flows coming from the riser and discharges in three outlets. If the separation is not perfect, oil and gas flows within water through

the water outlet. In the same way, part of the gas leaves through the oil outlet. The separation efficiency is given by the mass flow through the separator and by the separator pressure, which is a free variable.

The compressor is responsible for increasing the pressure so that the gas can be transported by the pipeline to the shore. Before the compressor there is a exhaustion valve that leads to the flare and in case the compressor can not process all the flow part can be burned.

The system components and free variables of the network are presented in the Figure 2.1. The decision variables are highlighted with a circle and all other variables can be determined using the system equations.

The free variables are:

- w_{gl}^{w} is the lift-gas injection rate in the well w.
- p^{s}, the pressure of the separator s.
- w_{fl}^{c}, the gas mass flow to be burned in the flare that is located before compressor c.
- w_{sl}^{c}, the recirculation flow through the anti-surge line in compressor c.

After this overview, each model will be specified in a different section, starting from the bottom until the export line.

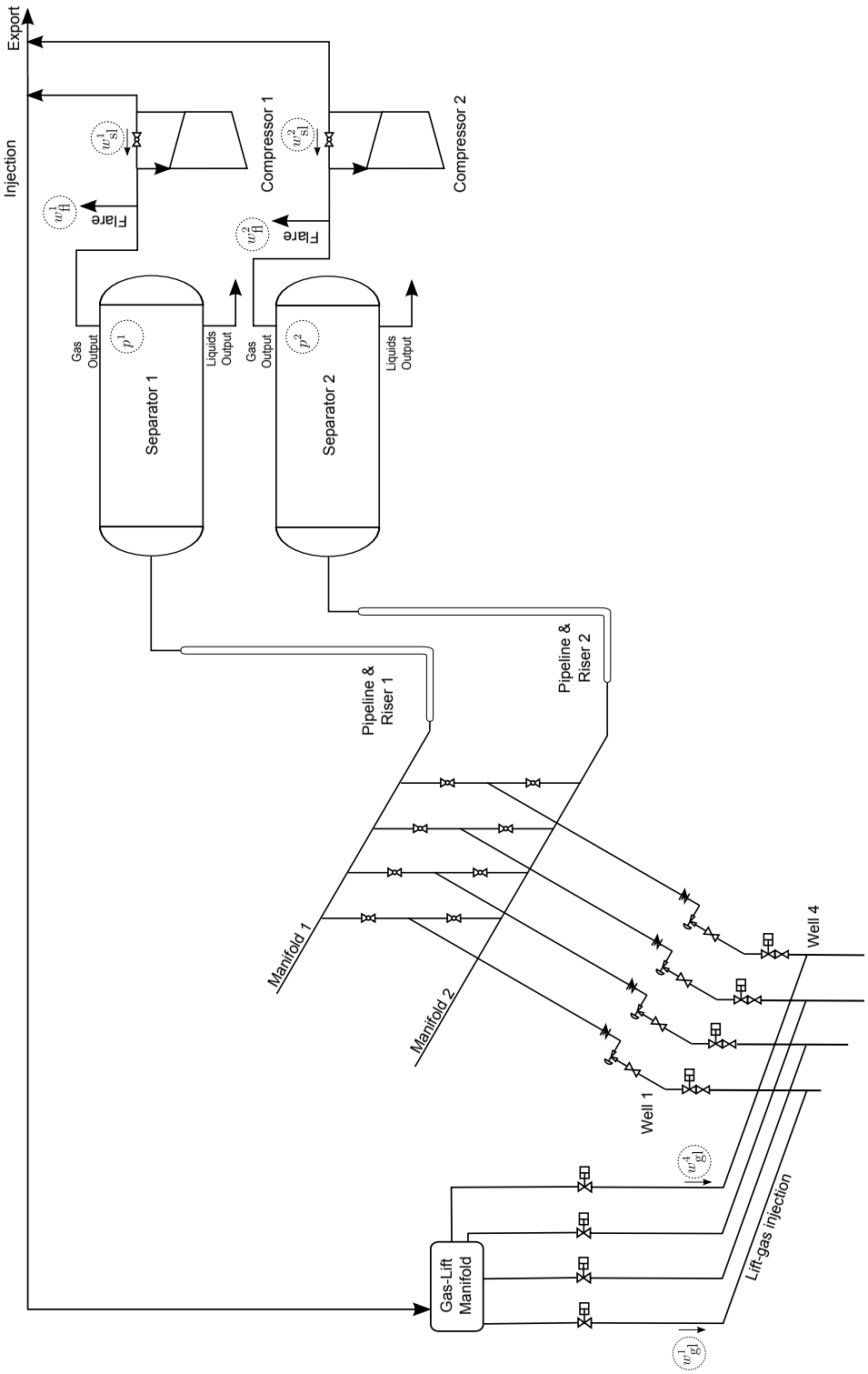

Figure 2.1: Network Schematic

2.2: Well Model

The well is the main actuator of an oil reservoir, it is composed by several different parts. In Figure 2.2 we summarize a gas-lifted well representing the following components: gas-lift choke, injection valve, annulus, tubing, casing, packer, and production choke.

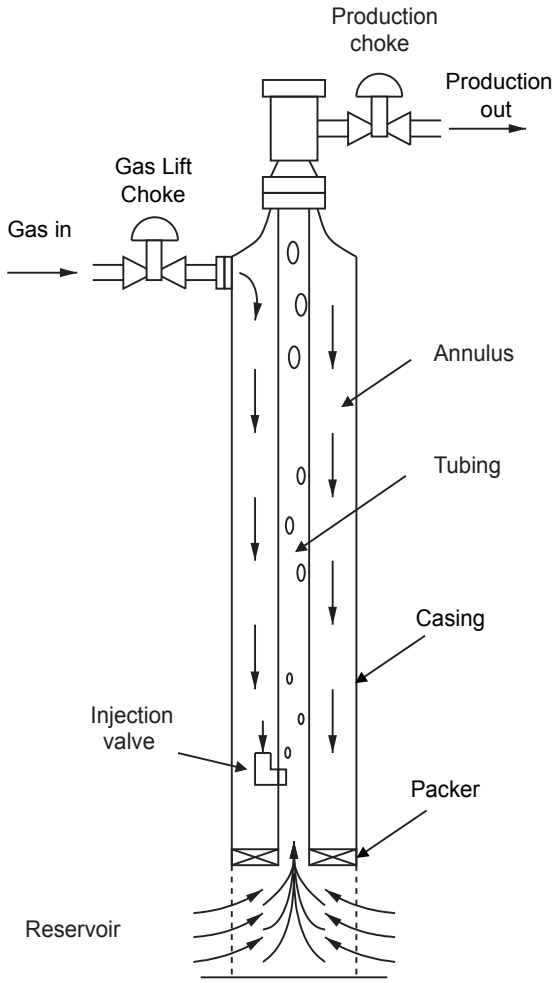

Figure 2.2: Well schematic [11]

The operation of a gas-lifted well can be described by the simple steps:

1. The injected gas passes through the gas-lift choke and builds up the annulus pressure.

2. As the pressure in the annulus exceeds the pressure in the tubing, gas starts to flow through the injection valve.

3. The injected gas mixes with the fluid emanating from the reservoir and reduces the overall density.

4. With a lower density the counter-pressure induced by the fluid is reduced making easier for the fluid to flow to the surface.

A gas-lifted well may operate in different regimes [17, 16]. Some of them with periodic dynamics as a consequence of an intermittent gas rate through the injection valve. This oscillatory behavior is not the aim of this work. Thus, it is assumed that *casing-heading* and *density wave* are avoided by a lower level control and operational constraints or are non existent.

For the gas-lifted well dynamic model we searched in the literature for the most used models. A Hammerstein Model [28] was discarded because it is tuned to characterize the dynamics but not the nonlinearities and discontinuities that are intrinsic of the system, besides this model need to be fitted to simulator data.

Plucenio's Model [27] is based on Partial Derivatives Equations (PDEs) that might be good to represent the well alone, however it is dispendious to represent in a complex network once it generates a high number of states.

Binder's Model [8] is an extension of Eikrem's Model, a model that has been developed and used in the last 10 years [11, 12, 1, 18]. An earlier version of the model was compared with OLGA simulator [18], even with the simplicity the results have considerable fidelity. The model has only 3 ODEs and some algebraic equations and still it can represent most of the characteristics and restrictions in the well. For these reasons, this is the model that it will be used.

The main assumptions of the model are:

- Oil Production is described by Vogel's Equation [35].
- The gas production and water production are given by the water-liquid ratio (watercut) and gas-oil ratio (GOR).
- The distribution of the masses of the three fluids happens without transport delay.
- Friction between the fluids and the annulus/tubing wall is not considered.
- Pressures are obtained by the ideal gas law and gravitational affect of liquids.

The model describes the dynamical system using 3 states and 3 inputs — w_{gl}, u_{pc}, and p_{ds}:

$$\dot{m} = \begin{bmatrix} \dot{m}_{\mathrm{ga}} = w_{\mathrm{in}} - w_{\mathrm{gi}} \\ \dot{m}_{\mathrm{gt}} = w_{\mathrm{gr}} + w_{\mathrm{gi}} - w_{\mathrm{gp}} \\ \dot{m}_{\mathrm{lt}} = w_{\mathrm{lr}} - w_{\mathrm{lp}} \end{bmatrix} \begin{matrix} \text{Mass of gas in the Annulus} \\ \text{Mass of gas in the Tubing} \\ \text{Mass of liquid in the Tubing} \end{matrix} \qquad (2.1)$$
$$w_{\mathrm{out}} = f_{\mathrm{c}}(m, p_{\mathrm{ds}}, u_{\mathrm{pc}}) \qquad \text{Choke flow}$$

In the annulus, the input of lift gas w_{in} increases the annulus gas mass m_{ga}, while the flow through the injection valve w_{gi} reduces it. The gas coming from the reservoir w_{gr} adds up to the gas coming from the annulus w_{gi} and is reduced from the gas flowing through the production choke w_{gp} giving the dynamic of gas mass in the tubing. The liquid mass in the tubing is the time integral of the liquid flowing from the reservoir w_{lr} discounted by the outgoing liquid flow through the production choke w_{lp}. The output vector of gas, oil, and water flows w_{out} is as function of the states $m = (m_{\mathrm{ga}}, m_{\mathrm{gt}}, m_{\mathrm{lt}})$ (Eq. 2.1), the opening in the production choke u_{pc}, and the choke downstream pressure p_{ds}, which depends on the pipeline states. The whole set of equations is presented in Appendix A.1.

The main difference between Eikrem's and Binder's models is that the last one considers not only production of oil but also water.

The main difference between Eikrem's and Binder's models is that, in Binder's model, the reservoir produces not only oil and gas, but water as well. Therefore, some changes were made in the density and pressure equations as can be seen in [8].

Both models consider the opening of the production choke u_{pc} as a controllable variable, however to reduce the complexity of the problem we assume that it is always fully open, since any other position of the choke would reduce the oil production which is against the objective of maximizing production.

An additional modification to Binder's model was necessary to fit the non-return characteristic in the gas-lift choke. There should be flow in the choke only if the pressure in the annulus is lower than the pressure before the choke, in the gas-lift manifold. Due to this limitation, the lift-gas injection must be within the bounds:

$$0 \leq w_{\mathrm{gl}} \leq w_{\mathrm{gl}}^{\max} = f_{\mathrm{gv}}(p_{\mathrm{gm}} - p_{\mathrm{ta}}) \qquad (2.2)$$

being $w_{\mathrm{gl}}^{\max}$ the flow when the gas-lift choke is fully open. It can be found using the choke function, which is dependent of the pressure difference between the gas-lift

manifold (p_{gm}) and the top of the annulus (p_{ta}).

2.3: Manifold

The manifold is a coupling element. It is modeled with no dynamics or complex equations. It is used to connect many components to a single one, Figure 2.3. It can be a three-phase flow manifold, as those in the production line, or a single-phase flow manifold, as the lift-gas manifold.

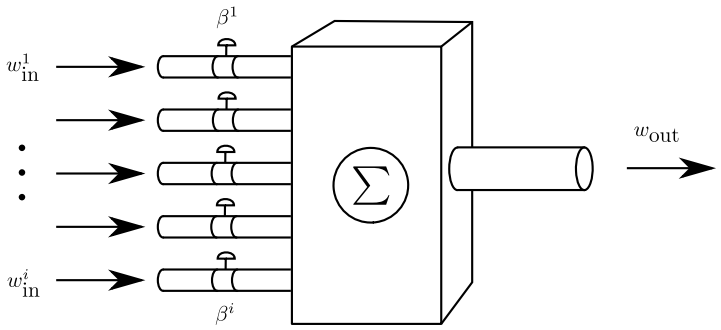

Figure 2.3: Manifold Schematic

The outlet flow, in the manifold, is the sum of all inlet flows. The flow is assumed to be perfectly mixed. The pressure at the inlet and outlet are considered the same.

$$w_{\mathrm{out}} = \sum_{i \in \mathcal{I}} \beta^i w_{\mathrm{in}}^i \tag{2.3a}$$

$$p_{\mathrm{in}}^i = p_{\mathrm{out}}, \qquad \forall i \in \mathcal{I} \tag{2.3b}$$

$$\beta^i \in \{0, 1\}, \qquad \forall i \in \mathcal{I} \tag{2.3c}$$

where w_{out} is the outlet flow and p_{out} is outlet pressure. w_{in}^i is the inlet flow and p_{in}^i the inlet pressure of the input i. β^i is the opening of the inlet valve i, which can be either fully open ($\beta^i = 1$) or fully closed ($\beta^i = 0$). $\mathcal{I}$ is the set of manifold inputs. The model is also the Appendix A.3.

2.4: Pipeline-Riser Model

In this section it is modeled the pipeline and the riser, elements that connect the subsea manifold to the offshore platform. For the same reasons that were explained in the well modeling, we are not interested in oscillatory behaviors.

During the literature research several models were found, which focus on modeling the turbulent regimes. Slug flow models are described by Taitel [34, 33], Di Meglio [10], Stasiak [31, 32], Masella [23], and Jahanshahi [19]. The last one presents a table (p. 1639 — Table 1) which compares his model with the state-of-the-art Multiphase Flow Simulator *OLGA* and other early proposed models. The conclusion that we can obtain from the table is that the model proposed by Jahanshahi [19] has a fair trade off in accuracy and complexity. It contains a reduced number of states when compared to models that use Partial Differential Equations (PDE).

Jahanshahi [19] assumes that:

- The system has two components: a horizontal flowline and a vertical riser.
- Each component has two state variables, one for the liquid mass and another for the gas mass.
- The connection is made by a virtual valve and the valve opening is given by a set of equations related to the height of liquid in the lowest part of the pipeline.
- The boundary conditions are the constant inflow of gas w_g and liquid w_l (with constant watercut) and the static pressure p_s at the end of the pipeline.
- In both elements, the gas is assumed to be ideal and the pressure drop due to friction is disregarded.
- The horizontal pipeline has a stratified flow, whereas a perfect mix is assume in the riser.

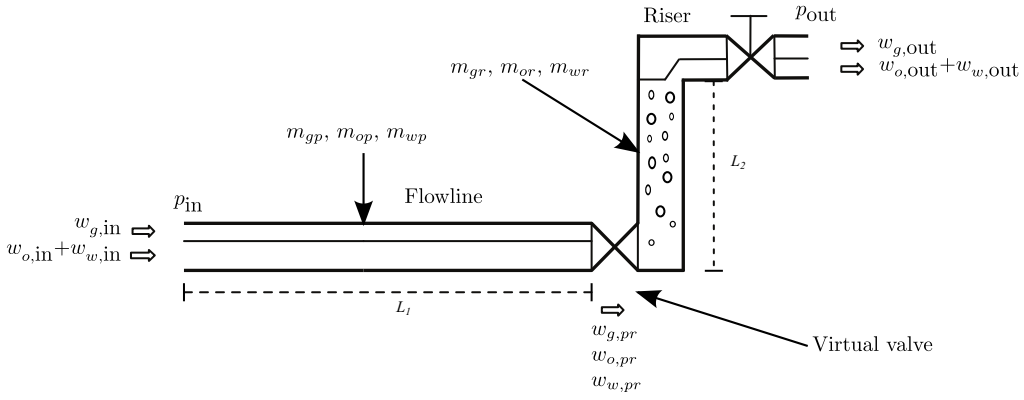

Figure 2.4: Pipeline and Riser Schematic

The flowline and riser have different flow regimes [37, 26]. As discussed in Section 2.2 regimes with oscillatory dynamics are out of the scope of this work. Therefore the model was simplified to avoid such conditions.

The assumption of constant watercut at the inlet will not hold if different production wells are connected to the same pipeline. For this reason, the states related to the liquid phase were split in 2, water and oil, in both sections.

The final model has 6 states. In the pipeline m_{gp}, m_{op}, and m_{wp}, are the mass of gas, oil, and water, respectively. In the riser m_{gr}, m_{or}, and m_{wr}, once again, are the mass of gas, oil, water. Figure 2.4 sketches the system with the modifications.

The set of equations of the model can be summarized as:

$$
\begin{aligned}
\dot{m} \quad &= \begin{bmatrix} \dot{m}_{gp} = w_{g,in} - w_{g,pr} \\ \dot{m}_{op} = w_{o,in} - w_{o,pr} \\ \dot{m}_{wp} = w_{w,in} - w_{w,pr} \\ \dot{m}_{gr} = w_{g,pr} - w_{g,out} \\ \dot{m}_{or} = w_{o,pr} - w_{o,out} \\ \dot{m}_{wr} = w_{o,pr} - w_{w,out} \end{bmatrix} \quad \begin{matrix} \text{Gas mass in the pipeline} \\ \text{Oil mass in the pipeline} \\ \text{Water mass in the pipeline} \\ \text{Gas mass in the riser} \\ \text{Oil mass in the riser} \\ \text{Water mass in the riser} \end{matrix} \\
y_{p,out} \quad &= f_1(m) \qquad \text{Outflow} \\
p_{ds} \quad &= f_2(m) \qquad \text{Pressure at the entrance}
\end{aligned}
\tag{2.4}
$$

The vector of states $m = (m_{gp}, m_{op}, m_{wp}, m_{gr}, m_{or}, m_{wr})$ do the balance of mass in the system. The system inputs are the inflow of gas $w_{g,in}$, oil $w_{o,in}$, and water $w_{w,in}$. The flow through the virtual valve that connects the two elements are the gas flow $w_{g,pr}$, the oil flow $w_{o,pr}$, and the water flow $w_{w,pr}$. The system outlet flow is composed by the gas flow $w_{g,out}$, oil flow $w_{o,out}$, and water flow $w_{w,out}$.

The model equations including the changes can be found in Appendix A.2. For the original model it is suggested the reference [19].

2.5: Separator

The separator developed in this section is inspired by the Sayda's physical-based model [29]. This model includes transient dynamics, but since its settling time is considerable faster than those in the wells and pipelines they will be discard.

This element will determine how much hydrocarbons are flowing along the water through the water outlet and how much gas within the oil goes through the oil outlet.

The separator is basically a tank with fluid flowing through it. It has a three-phase flow inlet and three outlets: a water outlet — that may discharge gas and oil as well, an oil outlet — that can discharge gas also, and a gas outlet. The separator is divided in

four volumes: the mixture, the water phase, the oil phase, and the gas phase. Figure 2.5 shows a schematic.

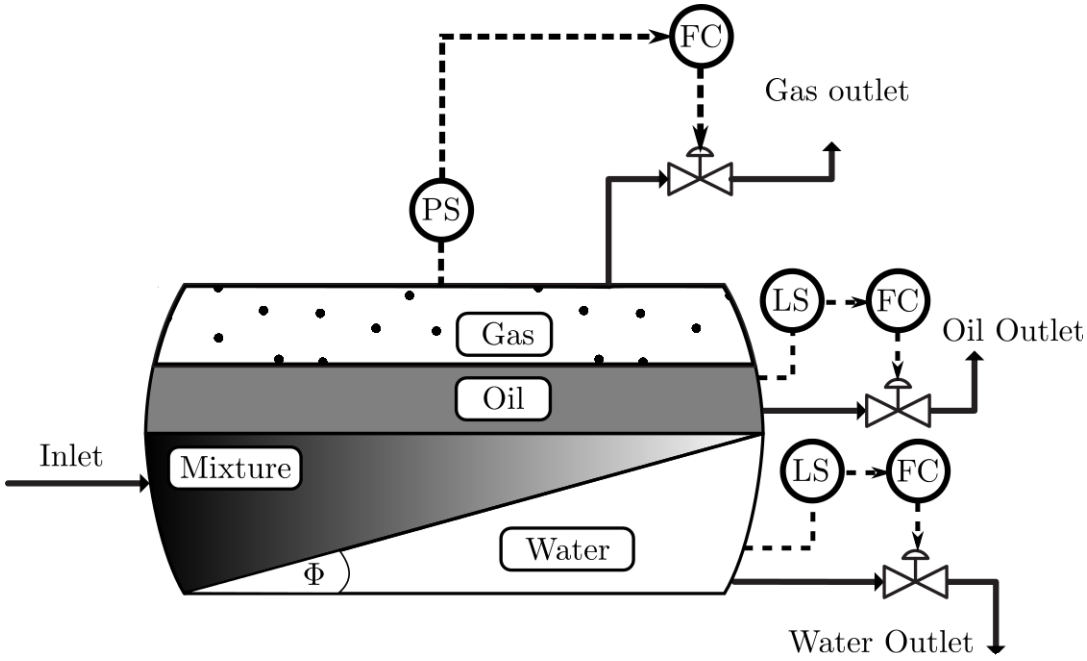

Figure 2.5: Separator Schematic

The separator pressure can be determined by the mass and volume of gas inside of the separator using the ideal gas equation. Since the separator length is relatively short when compared to pipelines, the pressure drop due to the friction through the separator is so small that it is negligible and pressure of all outlets are equal to the inlet pressure:

$$p_{\text{in}} = p_{\text{out}} \tag{2.5}$$

Being p_{in} the inlet pressure and p_{out} the pressure in all outlets.

The separator outlet pressure is defined as a controlled variable and it is assumed that a lower-level control loop controls the outlet gas valve to keep the pressure in the determined reference. This regulation dynamic is faster than other dynamics, therefore it is not considered in this model. In addition, two level controllers for the water-mixture level and the oil level are assumed to keep them in a predetermined reference. Since the heights and pressures have no dynamics, there is no mass accumulation inside the separator, therefore we can assure that all the inlet flow is leaving by one of the outlets.

$$q_{\text{in}} = \sum_{p \in \mathcal{P}} q_{\text{out}}^{p} \tag{2.6}$$

where q_{in} is a three-phase mass flow and each q^p_{out} is an outlet mass flow. $\mathcal{P}$ is the set of the outlets, they can be g, o or w, standing for the gas, oil, and water outlet.

What determines the separation efficiency between water and the mixture is the angle Φ of the interface between this two elements. This angle is the same as the angle formed by the movement of water droplets inside of the mixture. The droplets speed vector is composed by the speed of water flowing through the separator (horizontal component) and gravitational effect (vertical component), which is given by Stoke's law [22]:

$$\Phi = \tan^{-1}\left(\frac{v_v}{v_h}\right) \tag{2.7}$$

where Φ is the surface angle, v_h is the horizontal component of the water flow, and v_v is the vertical component induced by the gravitational effect.

The separation is perfect if the surface touches the oil phase, as it happens in Figure 2.5. Otherwise we need to virtually extend the separator to have perfect separation, as it happens in Figure 2.6. The perfect separation happens when the water and oil phases meet. It does not happen in L anymore, but the perfect separation happens on L_1. The ratio of the volumes of the mixture inside and virtually outside of the separator gives the water-mixture separation index. If the separation is not perfect, oil and gas flow through the water outlet.

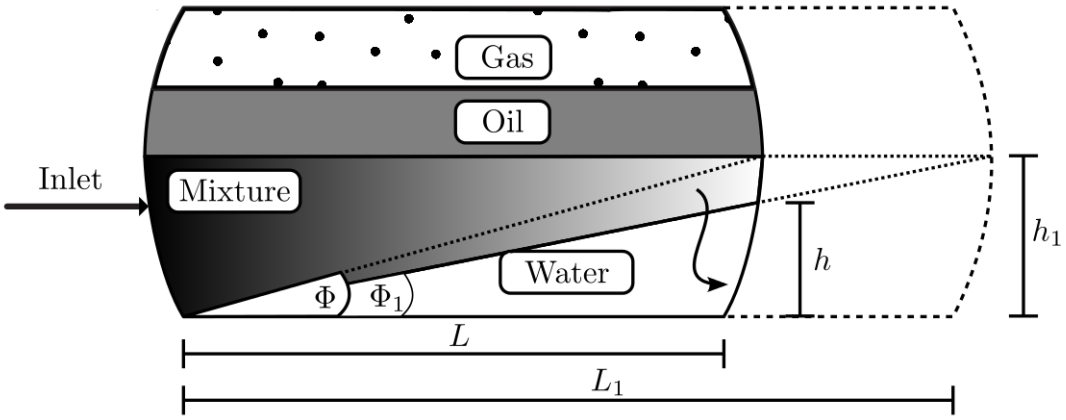

Figure 2.6: Virtual Separator Schematic

The oil-gas separation efficiency is based on Raoult's law and Dalton's law of partial pressures [25], which basically says that the composition of a fluid at equilibrium depends on the fluid pressure and the vapor pressure of the components. In mathematical format:

$$x_i = y_i \frac{P_{\text{v},i}}{P} \tag{2.8}$$

where x_i is the mole fraction of the component in the liquid phase. y_i is the mole fraction of the component in the vapor phase. We assume that the gas has only one component and the oil will not vaporize to the gas phase, in which case $y_i = 1$. $P_{\mathrm{v},i}$ is the vapor pressure of the component i, and P is the mixture pressure. Let g be the only the gas component, then:

$$x_g = \frac{P_{\mathrm{v},g}}{P} \tag{2.9}$$

Using Eq. 2.9 we can find the amount of gas that flows out through the oil outlet and how much flow through the gas outlet. The whole equations set appears in Appendix A.4.

2.6: Compressor

Three compressor models in the study of Grong [13] were examined to find one that suits this work. However, a new model was elaborated and it might fit better our purposes.

The first of the three models is a physical-based model which describes very well the dynamics of the compressor in the simulations as is shown in [13]. Nevertheless, the performance relies on physical design parameters which are difficult to measure and non-existent in the commercial compressor data sheets. For this reason, it is very difficult to implement.

To overcome the lack of information to build a physical model, the following models will only use typical data given by manufacturers. The compressor map, Figure 2.7, is given in the compressor data sheet and contains the compressor static behavior. On its abscissa axis there is the compressor volumetric inlet flow and on the ordinate axis there is the pressure gain through the compressor. Moreover, the chart is composed by several lines. Each line represents the compressor characteristic for a given compressor speed. Figure 2.7 presents two boundaries with dashed lines. The surge line is a limit which separate the map of the unstable flow region. The physical explanation is similar to the stall effect in airplanes wings. The other boundary is the choke line, this limit represent the maximum flow that the compressor can handle.

The next model [13] uses a 4th order polynomial function to approximate all speed lines in the compressor map. The parameters of the polynomial function are speed dependent and are fitted in a least square way from experimental data. The

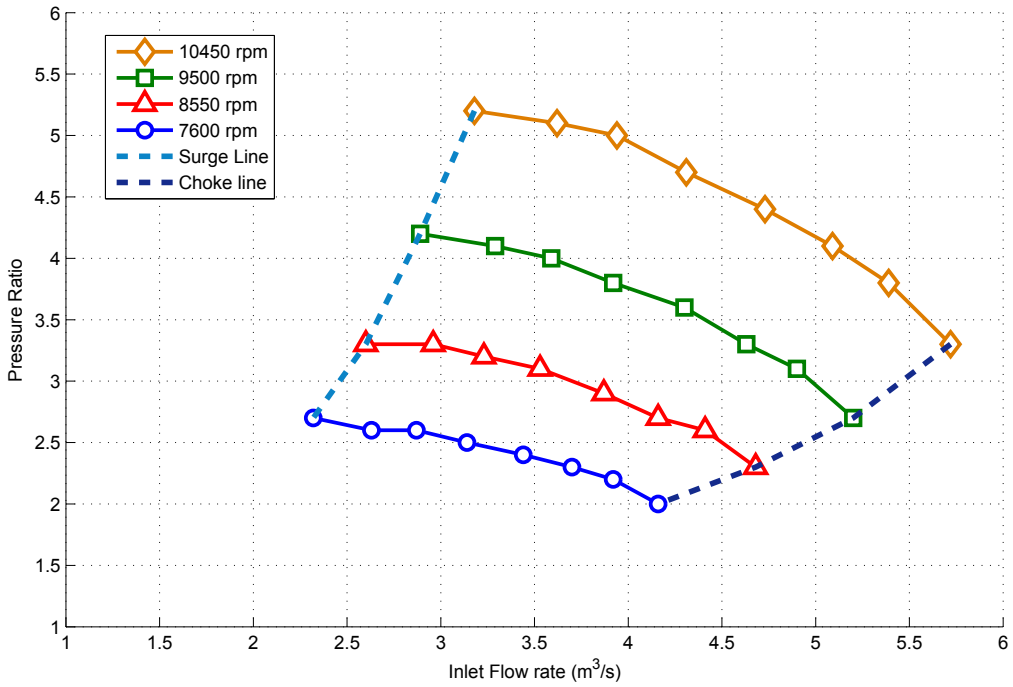

Figure 2.7: Compressor map

fidelity is not very good if compared to the other models in the studies thereof, and the calculation time is very similar to the physical model [13].

The third model of Grong, Table Lookup Model, creates a 4th order polynomial approximation for each speed line. The operation of the compressor must be done over the predetermined speed lines; however, new speed lines can be created by interpolating the compressor map data. The model fits more precisely than the prior models however it takes much longer to calculate [13].

In a very similar way, the previous models use the information from the compressor map to define the dynamics of the compressor.

The fourth model does not have states, since the dynamics of the compressor are much faster than those in the wells and pipelines. Figure 2.8 shows the basic representation of the model.

This model has the objective to assure that the compressor is running within the map limits. For this, the map boundaries are approximated by polynomial and distance vectors from the operation point to each boundary are calculated. If all vectors have negative values it means that system is operating inside of the map. The boundaries, in Figure 2.8, are: 10450 rpm — the maximum speed (upward), 7600 rpm — the

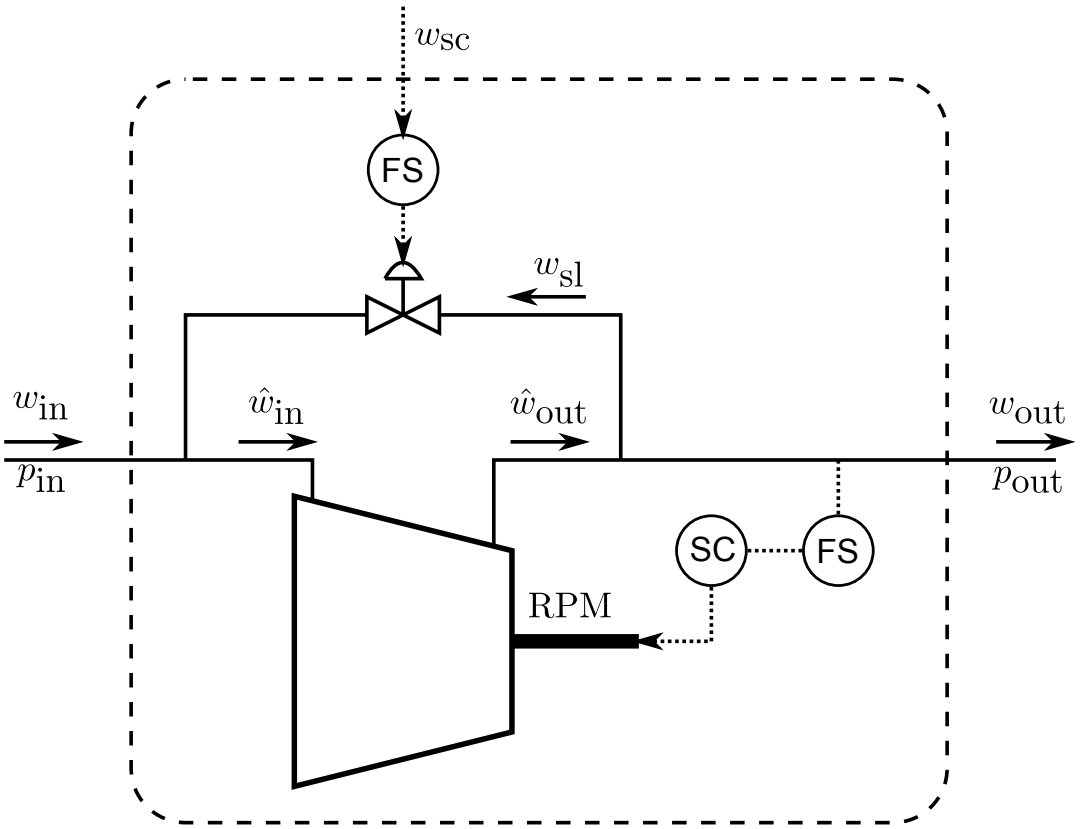

Figure 2.8: Compressor model schematic

minimum speed (downward), surge line (leftward), and choke line (rightward)

It is assumed that there is a lower level controller for the speed of the compressor to ensure that it is keeping the output pressure constant and at its desired reference. Since it has no dynamics, all the mass flow that enters in the system compressor leaves it:

$$w_{\mathrm{in}} = w_{\mathrm{out}} \tag{2.10}$$

The compressor map is expressed in volumetric flow while the rest of the system is modeled with mass flow, so we convert the flow using the ideal gas equation and the conditions at the compressor inlet:

$$q_{\mathrm{vol}} = \frac{w_{\mathrm{in}} RT}{p_{\mathrm{in}} M_{\mathrm{g}}} \tag{2.11}$$

where q_{vol} is the volumetric flow, w_{in} is the mass flow, R is the ideal gas constant, T is the temperature at the entrance of the compressor, p_{in} is the pressure upstream, p_{out} is the pressure downstream, and M_{g} is the gas molar mass. And the pressure ratio

through the compressor is defined:

$$r_{\mathrm{p}} = \frac{p_{\mathrm{out}}}{p_{\mathrm{in}}} \tag{2.12}$$

The map is valid for normalized conditions of inlet pressure and temperature, so to be used in different conditions it need be adjusted:

$$\overline{q}_{\mathrm{vol}} = q_{\mathrm{vol}} \frac{T}{T_{\mathrm{map}}} \frac{p_{\mathrm{map}}}{p_{\mathrm{in}}} \tag{2.13a}$$

$$\overline{r}_{\mathrm{p}} = r_{\mathrm{p}} \frac{T}{T_{\mathrm{map}}} \tag{2.13b}$$

The approximation of each of the four boundary lines of the compressor map is made by a second order equation. The upward boundary $f_{\mathrm{u}}(q)$, the downward boundary $f_{\mathrm{d}}(q)$, the leftward boundary $f_{\mathrm{l}}(q)$, and rightward boundary $f_{\mathrm{r}}(q)$ are defined:

$$f_{\mathrm{i}}(\overline{q}_{\mathrm{vol}}) = a_i \overline{q}_{\mathrm{vol}}^2 + b_i \overline{q}_{\mathrm{vol}} + c_i, \quad \forall i \in \{u, d, l, r\} \tag{2.14}$$

The parameters a_i, b_i, and c_i for each i are obtained by solving an unconstrained least squares problem with the compressor map data. The result of the approximation is shown in Figure 2.9.

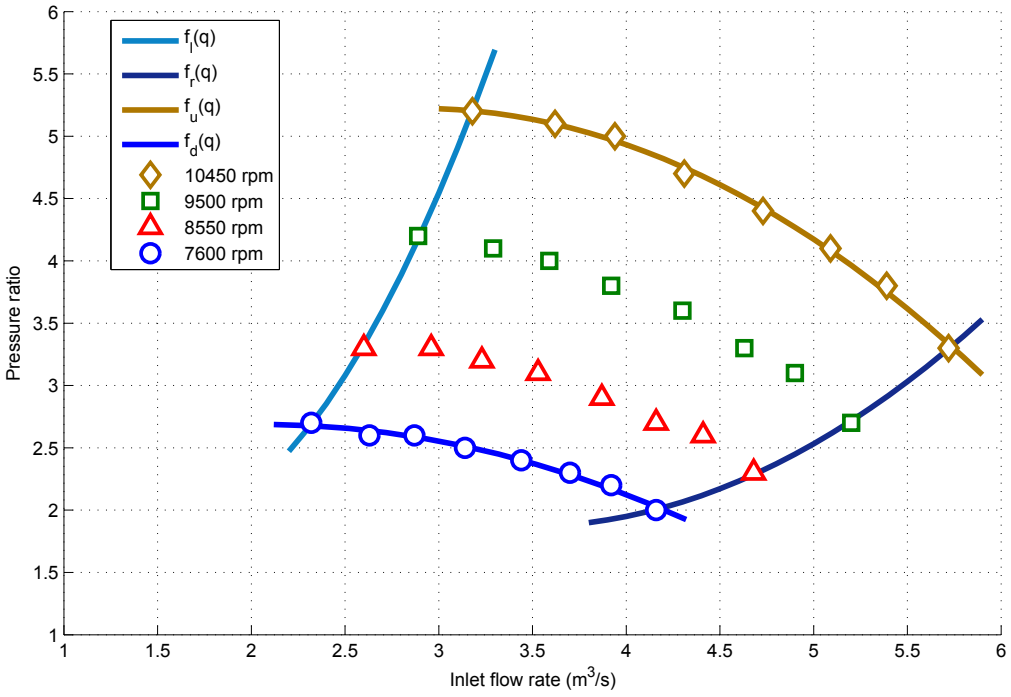

Figure 2.9: Compressor map approximation

Given a pressure ratio $\overline{r}_{\mathrm{p}}$ we can find projection of the leftward and rightward

boundaries in the flow rate axis is solving the second order equation:

$$\overline{q}_{l,\mathrm{vol}} = \frac{-b_l + \sqrt{b_l^2 - 4a_l(c_l - \overline{r}_\mathrm{p})}}{2a_l} \tag{2.15a}$$

$$\overline{q}_{r,\mathrm{vol}} = \frac{-b_r + \sqrt{b_r^2 - 4a_r(c_r - \overline{r}_\mathrm{p})}}{2a_r} \tag{2.15b}$$

To measure the distance from the current operating point to the boundaries it was created four variables:

$$\nu_\mathrm{u} = \overline{r}_\mathrm{p} - f_\mathrm{u}(\overline{q}_\mathrm{vol}) \tag{2.16a}$$

$$\nu_\mathrm{d} = f_\mathrm{d}(\overline{q}_\mathrm{vol}) - \overline{r}_\mathrm{p} \tag{2.16b}$$

$$\nu_\mathrm{l} = \overline{q}_{l,\mathrm{vol}} - \overline{q}_\mathrm{vol} \tag{2.16c}$$

$$\nu_\mathrm{r} = \overline{q}_\mathrm{vol} - \overline{q}_{r,\mathrm{vol}} \tag{2.16d}$$

$$\tag{2.16e}$$

The inequalities in Eq. 2.17 ensure that the compressor is operating inside of the map.

$$\nu_b \leq 0, \qquad \forall b \in \{u, d, l, r\} \tag{2.17}$$

We can increase the operation region using a recirculation line. The recirculation line redirects part of the outlet flow to the inlet. In this way we can increase the inflow of the compressor to avoid the operation before the surge line. For this we write:

$$\widehat{w}_\mathrm{in} = w_\mathrm{in} + w_\mathrm{sl} \tag{2.18a}$$

$$w_\mathrm{out} = \widehat{w}_\mathrm{out} - w_\mathrm{sl} \tag{2.18b}$$

where w_sl is the flow through the surge control line, $\widehat{w}_\mathrm{in}$ is the new compressor inflow and $\widehat{w}_\mathrm{out}$ is the new outflow. Look in Figure 2.8 for a better understanding.

If we replace w_in in the Eq. 2.11 by $\widehat{w}_\mathrm{in}$, then we can move to the right, parallel to the abscissa axis in the compressor map, making it possible to leave the infeasible region.

The whole set of equations of this model is shown in a compact format in Appendix A.5.

2.7: Models Coupling

In this section we will discuss how we can connect all the subsystems to form the oil production network with the lift-gas injection loop.

The network inputs are the vector lift-gas injection in the wells w_{gl}, the vector separator pressure p, the vector of exhaustion flare valve flow w_{fl}, and vector of surge control line flow in the compressor w_{sc}.

$$w_{\mathrm{gl}} = [w_{\mathrm{gl}}^1, \ldots, w_{\mathrm{gl}}^{N_w}]^T \tag{2.19a}$$

$$p = [p^1, \ldots, p^{N_s}]^T \tag{2.19b}$$

$$w_{\mathrm{fl}} = [w_{\mathrm{fl}}^1, \ldots, w_{\mathrm{fl}}^{N_c}]^T \tag{2.19c}$$

$$w_{\mathrm{sc}} = [w_{\mathrm{sc}}^1, \ldots, w_{\mathrm{sc}}^{N_c}]^T \tag{2.19d}$$

where:

- N_w is the number of wells in the network,
- N_p is the number of pipelines,
- N_s is the number of separators,
- N_c is the number of compressors.

With these parameters the following sets can be defined:

- $\mathcal{W} = \{1, \ldots, N_w\}$ is the set of wells in the network,
- $\mathcal{P} = \{1, \ldots, N_p\}$ is the set of pipelines,
- $\mathcal{S} = \{1, \ldots, N_s\}$ is the set of separators,
- $\mathcal{C} = \{1, \ldots, N_c\}$ is the set of compressors.

Besides, we have the sets the connects two elements:

- the set $\mathcal{S}_P$ that have pairs of separators and pipelines that are connected ($(p, s) \in \mathcal{S}_p \rightarrow p \in \mathcal{P}, s \in \mathcal{S}$),
- the set $\mathcal{C}_S$ contains pairs of compressors and separators that are connected ($(s, c) \in \mathcal{C}_S \rightarrow s \in \mathcal{S}, c \in \mathcal{C}$).

Let the subscripts be a reference to the model, being: w a well model, m a manifold, p a pipeline mode, s a separator model, and c a compressor model. The superscript is the index for a specific element of a set. Let x be a vector of state variables of a specific model, f the state derivative function, the vector y be the model algebraic variables, and g be the algebraic function that defines y. In y there are all variables that are not states. For instance x^w_{w} has the well state variables of well w in the set $\mathcal{W}$. We have the system states:

$$\dot{x}^w_{\mathrm{w}} = f_{\mathrm{w}}(x^w_{\mathrm{w}}, y^w_{\mathrm{w}}) \qquad \forall w \in \mathcal{W} \tag{2.20a}$$

$$\dot{x}^p_{\mathrm{p}} = f_{\mathrm{p}}(x^p_{\mathrm{p}}, y^p_{\mathrm{p}}) \qquad \forall p \in \mathcal{P} \tag{2.20b}$$

and the system algebraic equations:

$$g_{\mathrm{w}}(x^w_{\mathrm{w}}, z^w_{\mathrm{w}}) = 0 \qquad w \in \mathcal{W} \tag{2.21a}$$

$$g_{\mathrm{p}}(x^p_{\mathrm{p}}, z^p_{\mathrm{p}}) = 0 \qquad p \in \mathcal{P} \tag{2.21b}$$

$$g_{\mathrm{s}}(y^s_{\mathrm{s}}) = 0 \qquad s \in \mathcal{S} \tag{2.21c}$$

$$g_{\mathrm{c}}(y^c_{\mathrm{c}}) = 0 \qquad c \in \mathcal{C} \tag{2.21d}$$

The following equations connect the system free variables with the respective element variables. The variables and components can be localized using Figure 2.10.

$$w^w_{\mathrm{in}} = w^w_{\mathrm{gl}} \qquad \forall w \in \mathcal{W} \tag{2.22a}$$

$$p^s_{\mathrm{out}} = p^s \qquad \forall (s) \in \mathcal{S} \tag{2.22b}$$

$$w^c_{\mathrm{in}} = w^s_{\mathrm{out}} - w^c_{\mathrm{fl}} \qquad \forall (c, s) \in \mathcal{C}_s \tag{2.22c}$$

$$w^c_{\mathrm{sl}} = w^c_{\mathrm{sc}} \qquad \forall c \in \mathcal{C} \tag{2.22d}$$

where w^w_{in} is the well model flow input, p^s_{out} is the separator model pressure output, w^c_{in} is the compressor model flow input, w^s_{out} is the separator model flow output, and w^c_{sl} is the compressor model anti-surge line flow.

The equation that connects the well with the gas-lift manifold is:

$$p^w_{\mathrm{in}} = p_{\mathrm{gm}} \qquad \forall w \in \mathcal{W} \tag{2.23}$$

where p_{gm} is the gas-lift manifold pressure, and p^w_{in} is the well input pressure of the well w.

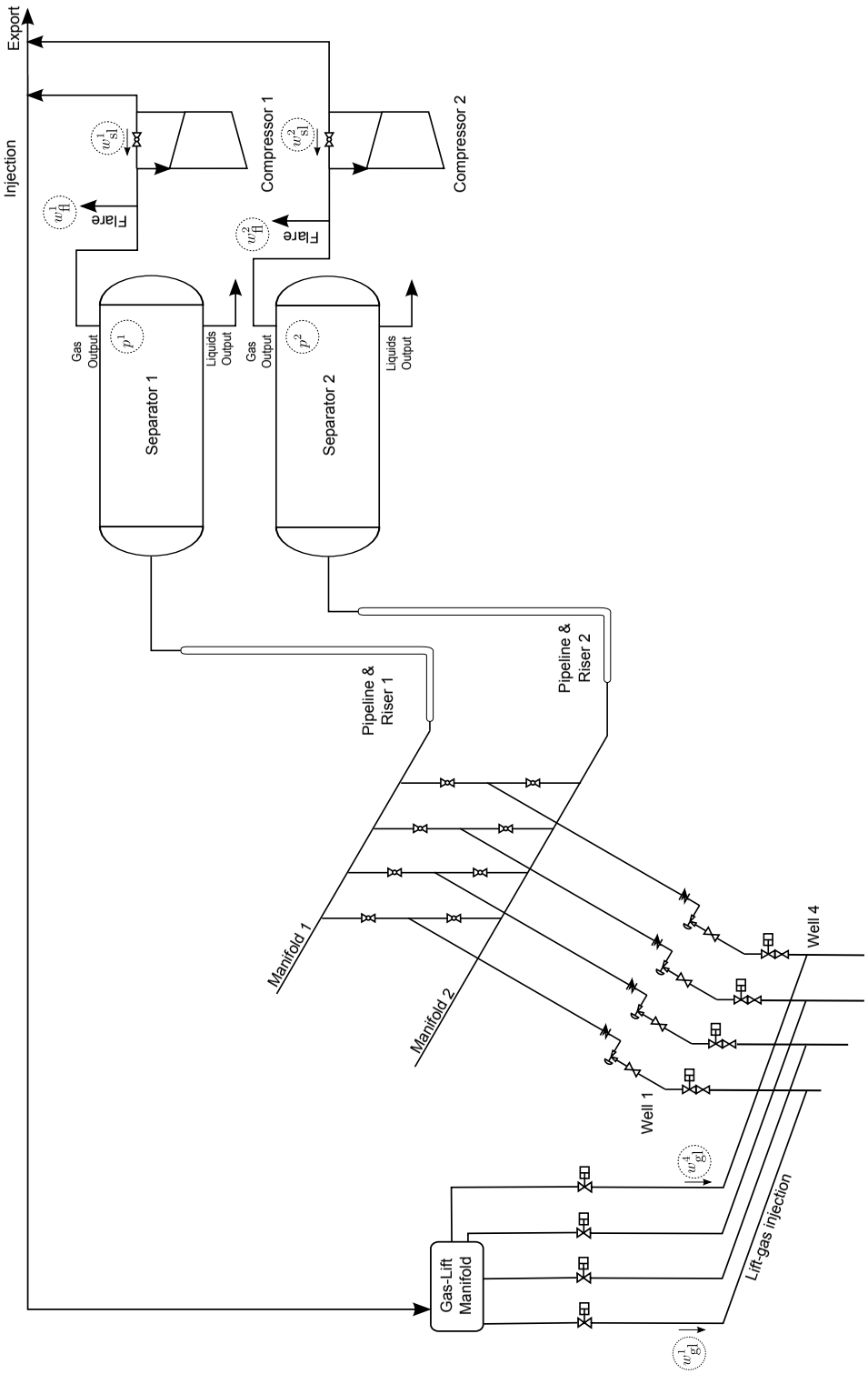

Figure 2.10: Network Schematic

The manifold equations that connect wells and pipelines are:

$$w_{\text{in}}^{p} = \sum_{w \in \mathcal{W}} \beta_{w,p} w_{\text{out}}^{w} \qquad \forall p \in \mathcal{P} \tag{2.24a}$$

$$p_{\text{out}}^{w} = \sum_{p \in \mathcal{P}} \beta_{w,p} p_{\text{in}}^{p} \qquad \forall w \in \mathcal{W} \tag{2.24b}$$

where w_{in}^{p} is the pipeline input pressure, $\beta_{w,p}$ is the manifold routing binary variable, w_{out}^{w} is the well outlet flow, p_{out}^{w} is the well pressure at the outlet, and p_{in}^{p} is the pipeline inlet pressure.

The set of equations that connect the pipeline and the separator are:

$$w_{\text{in}}^{s} = w_{\text{out}}^{p} \qquad \forall (s,p) \in \mathcal{S}_p \tag{2.25a}$$

$$p_{\text{in}}^{s} = p_{\text{out}}^{p} \qquad \forall (s,p) \in \mathcal{S}_p \tag{2.25b}$$

where w_{in}^{s} is the separator inlet flow, and p_{in}^{s} is the separator inlet pressure.

The connection of the separators and the compressors is made by Eq. 2.22c and by:

$$p_{\text{in}}^{c} = p_{\text{out}}^{s} \qquad \forall (c,s) \in \mathcal{C}_s \tag{2.26a}$$

$$p_{\text{out}}^{c} = p_{\text{exp}} \qquad \forall c \in \mathcal{C} \tag{2.26b}$$

where p_{in}^{c} is the compressor inlet pressure, p_{out}^{s} is the separator outlet pressure, p_{out}^{c} is the compressor outlet pressure, and p_{exp} is the exportation line pressure.

The exportation line equations are given by:

$$w_{\text{exp}}^{g} = \sum_{c \in \mathcal{C}} w_{\text{out}}^{c} - w_{\text{gm}} \tag{2.27a}$$

$$w_{\text{exp}}^{o} = \sum_{s \in \mathcal{S}} w_{\text{o,out}}^{s,\text{o}} \tag{2.27b}$$

$$w_{\text{exp}}^{w} = \sum_{s \in \mathcal{S}} w_{\text{w,out}}^{s,\text{w}} \tag{2.27c}$$

where w_{exp}^{g} is the amount of gas that is exported, w_{out}^{c} is the compressor outlet flow, w_{gm} is the flow to the gas-lift manifold, w_{exp}^{o} is the amount of oil that is exported, $w_{\text{o,out}}^{s,\text{o}}$ is the oil flow through the separator oil outlet, w_{exp}^{w} is the exported water, and $w_{\text{w,out}}^{s,\text{w}}$ is the water flow through the separator water outlet.

The gas-lift manifold equations are:

$$p_{\mathrm{gm}} = p_{\mathrm{exp}} \tag{2.28a}$$

$$w_{\mathrm{gm}} = \sum_{w \in \mathcal{W}} w_{\mathrm{gl}}^{w} \tag{2.28b}$$

The whole set of equations is in Appendix A.6.

Chapter 3: Modelica and JModelica.org

In this chapter it is presented the tools used to model the system and solve the optimization problem.

First the Modelica language is presented, commenting its uses and applications. After, it is given a brief explanation about the Modelica extension named Optimica. And at last it is discussed JModelica.org, a toolbox that implements a Modelica and Optimica compiler and gathers several important tools for solving optimal problems.

3.1: Modelica

Modelica is a non-proprietary object-oriented language developed to model dynamic behavior of technical systems in a simple way. The models are described using differential, algebraic, and discrete equations. Models can also be composed by other models in a hierarchical structure.

Unlike other languages, as MATLAB or C++, Modelica has a syntax very close to the mathematical formulation. The focus of this language is to spare the user from programming repetitive routines and leaving him time to focus on the development of the model itself.

Modelica is a textual language, however there are several environments that provide a graphical user interface. Some implementations are free and others commercial. Free environments are: OpenModelica (Linköping University) and JModelica.org (Lund University). Commercial environments are: CATIA Systems (Dassault Systemes), Dymola (Dynasim), MapleSim (MapleSoft), MathModelica (Wolfram Research), and others.

The language has academic and industrial applications in many areas *e.g.*: vehicle dynamics, hydraulics, combustion, air conditioning, electrical and electronic systems, and control. Industrial partners are essentially automotive companies such as Audi, BMW, Daimler, Ford, Toyota, and VolksWagen. Conferences on this language have been happening almost every year since 2000 in Europe and North America, with more than 600 papers published in these conferences.

Some examples of the usage of the language can be found in Appendix B.

3.2: Optimica

Modelica does not support natively formulations of optimization problems, since it was developed for simulation purposes only. Some of the main elements for optimization are not present, as the objective function, constraints, or the optimization horizon.

The objective of Optimica [4] is to offer the user a way to express optimization problems using Modelica models. Optimica is a Modelica extension designed to permit a high-level specification of dynamic optimization problems based on Modelica models. It was first defined in [20].

Dynamic optimization problem presents three levels of information. In the first level the canonical description of the optimization problem. This contains the cost function, free variables and parameters, the optimization interval, and the constraints. In the second level it is specified the transcription method for discretization of the problem from the continuous time to a discrete time mesh. The third level is the algorithm level, where the tolerances and algorithm tuning parameters are specified.

The Optimica fits in the first level, giving an easy way to represent such kind of information. The remaining levels are usually dealt by the Optimica implementation, because this task is critical and very error prone. Although, the implementation gives you options to define the way it is done.

Optimica introduces three main features in Modelica standard:

- Optimization class, a new class of objects can be instantiated. It represents the optimization problem. Necessarily it needs an objective function and optimization variables. It is important to note that the prediction horizon can be an optimization variable.
- A constraint section inside of optimization elements. In this section we can declare equality and inequality variable constraints.
- Access variables in time. With the Modelica standard we can not access variables in a specific time. With Optimica we are able to make constraints like: $x(t_f) = 0$ (variable x at the final time t_f must be equal to 0) or $y(5) \leq 1$ (variable y at time 5 must be less than 1)

3.3: JModelica.org

JModelica.org [4, 5, 3] is an open-source Modelica-based platform for optimization and simulation of dynamic systems. It aims to be a viable and stable platform for industrial applications, at the same time that it has flexibility for the academia develop new algorithms and studies.

The JModelica environment consists of a collection of software modules, including compilers for Modelica and Optimica, a code generator for C, a run-time library in C, a simultaneous optimization algorithm, and a library for integration with the scripting language Python. Also internally it connects with three other very used tools: Sundials, CasADi and IPOPT.

Sundials [15] is a package of ODE and DAE solvers that was developed to be a robust time integrator and nonlinear solver. The most used solver is CVODE an ODE solver with the ability to solve some special types of DAE that can automatically calculate sensibilities.

CasADi [6] is s symbolic framework for automatic differentiation and numeric optimization. It applies both symbolic and numeric differentiation in low-level for quick and efficient calculations. It has an important use for ODE and DAE systems where the solvers need the system Jacobian for simulation and sensibility calculation.

IPOPT [36] is a package for large-scale nonlinear optimization. It is based on the Interior-Point Method. Being more specific, it is a primal-dual interior-point algorithm with a filter line-search method. Although, we can simplify this by saying that is an algorithm that can robustly find local minima for non-convex problems and global minima for convex problems [36]. It also has an automatic problem scaling function, some heuristics for speeding up convergence, and some techniques to increase the robustness.

Together, these software form a complete tool for formulating and solving dynamic optimization problems based on Modelica models. The data flow of the JModelica.org is illustrated in Figure 3.1. It is typical that the solution of dynamic optimization problems require multiple iterations, where the cost function, the constraints, the transcription method, and even the model are refined in order to obtain satisfactory results. The result then typically needs to be analyzed and the optimization formulation parameters adjusted. The use of high-level description languages then relieves the user of the costly and error-prone task of encoding the model and optimization formulations

in less suitable languages. In effect, the focus of the design process is shifted from encoding of the problem into formulation of the problem, which translates into more efficient design processes.

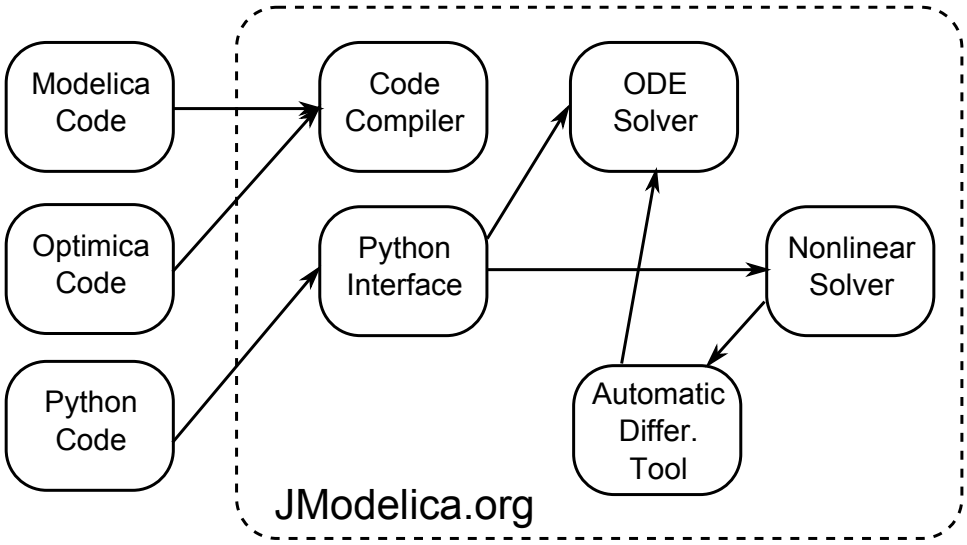

Figure 3.1: Network Schematic

Chapter 4: Nonlinear Model Predictive Control

In this chapter we present a method to discretize the continuous model proposed in Chapter 2, afterward it is shown several optimization problems that uses the network model.

4.1: Problem Discretization

This section was based on [7], it starts with a recapitulation of techniques for solving ODEs, after, the collocation method is presented followed by an example of use, and at last it is shown how to formulate optimization problems using the collocation method.

4.1.1: ODE Solvers

The representation made in Chapter 2 uses DAEs to describe the system, therefore to obtain the system answer over time a Initial Value Problem (IVP) need to be solved. This means that we have an initial condition and DAEs, by integrating the system a system time-dependent solution is obtained.

Let Eq. 4.1a define the system in the time interval $t \in [0, T]$, the analytical solution of this ODE is given by 4.1b.

$$\dot{x} = f(x, t) \tag{4.1a}$$

$$x(t) = x(0) + \int_0^t f(x, t)\, dt \tag{4.1b}$$

where x is a vector of states, $f(\cdot)$ is nonlinear time variant function. Due to the size and the complexity of the systems such analytical integral is usually very dispendious to find, becoming impractical to use. To overcome this problem numerical approximation are used.

Numerical methods approximates the continuous time dynamic system by a discrete time dynamic system. They divide the time interval $[0, T]$ into sub-intervals $[t_{i-1}, t_i]$ with $i \in [1, \ldots, N]$, assuming a stepsize h_i for each i interval. Being $t_i = t_{i-1} + h_i$, the states approximation $z_i \approx x(t_i)$, where $x(t_i)$ is the exact solution Eq. 4.1b in the time t_i.

In the interval endpoints, the current and the subsequent states approximation must be equal to maintain the continuity. Figure 4.1 shows a comparison between the analytical solution and a numerical approximation using the Euler method and how the time grid is created. The system is a simple linear system $\dot{x} = -\frac{x}{2}$, with $x(0) = 1$. The analytical solution (blue line) is $x(t) = e^{-\frac{t}{2}}$.

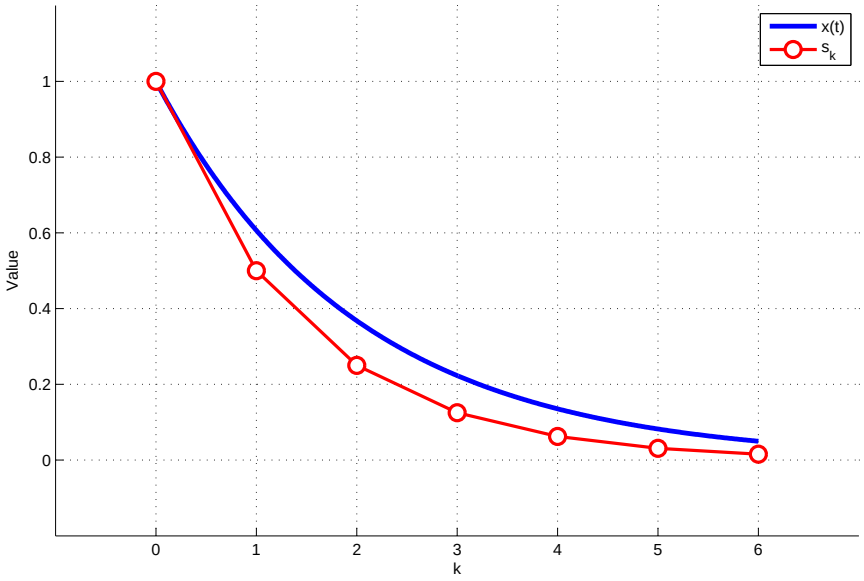

Figure 4.1: Example using Euler Method, $h = 1$ and N=6

Numerical methods can be categorized in two big groups: explicit or implicit methods.

The explicit methods uses the information of the current states and it derivative to find the next state. It is easier to implement and faster to compute, however a bad tuning can lead to instabilities in the solver. A very common explicit method is the Euler method (Eq. 4.2), it uses only the prior state and it derivative to find the next step. Figure 4.1 shows an application of this method for a linear system, where $h_i = 1 \; \forall i$.

$$z_i = z_{i-1} + h_i f(z_{i-1}, t_{i-1}) \tag{4.2}$$

A more complex method, but widely used is the explicit fourth order Runge-Kutta (ERK) method. The ERK uses not only one evaluation per time interval but it combines the derivative of four points inside of the time interval to obtain the next state approximation, as Eq. 4.3 shows. This method have a higher accuracy compared to the Euler method. But still, as explicit method it may have some stability issues

depending on the system convergence rate.

$$z_i = z_{i-1} + \frac{1}{6}h(k_1 + 2k_2 + 2k_3 + k_4) \tag{4.3a}$$

$$k_1 = f(t_{i-1}, z_{i-1}) \tag{4.3b}$$

$$k_2 = f(t_{i-1} + \frac{1}{2}h, z_{i-1} + \frac{h}{2}k_1) \tag{4.3c}$$

$$k_3 = f(t_{i-1} + \frac{1}{2}h, z_{i-1} + \frac{h}{2}k_2) \tag{4.3d}$$

$$k_4 = f(t_{i-1} + h, z_{i-1} + hk_3) \tag{4.3e}$$

where k_1 is the derivative at the start of the interval. k_2 is the derivative at the middle of the interval using the state calculated using k_1. k_3 is the derivative at the middle of the interval, again, but using k_2. k_4 is the derivative at the end of the interval, calculated using k_3. It is clearly notable that dependencies are progressive, k_2 depends on k_1, but not the other way around.

The implicit methods use not only the current state but the next state and it derivative to solve the DAEs. It has a greater improvement in the stability, however it is necessary to solve an equation to find the next state at each time interval. Among the implicit methods the most commons are backward Euler method, implicit Runge-Kutta (IRK), and collocation method.

The backward Euler allows us to clearly see that the state calculation is not straight forward and a equation need to be solved, usually using some iterative algorithm. Eq. 4.4 shows the Backward Euler method, to calculate the next state it uses the current state and the derivative of the next state. This dependency of the next state on it derivative makes necessary the use of a iterative solver. Usually a Newton-Raphson method is used, but for more complex systems a nonlinear solvers is necessary.

$$z_i = z_{i-1} + h_i f(z_i, t_i) \tag{4.4}$$

In the same way that explicit Runge-Kutta enhances the accuracy of the forward Euler, the implicit Runge-Kutta (IRK) makes the backward Euler method more precise. Different from the ERK, the IRK have a crossed relation in calculation of k values, for instance the value of k_1 is dependent of k_4. This cross dependence makes necessary to solve equations.

The collocation method can be formulated as a IRK, it has some properties that makes it interesting for dynamic optimization. This topic will be cover in the following

subsection.

4.1.2: Collocation Method

The Collocation method is a special case of IRK where the interval are approximated by a n-th order polynomial.

The time is divided in N intervals, each $i \in [1, \ldots, N]$ intervals have a timestep h_i. For every timestep a polynomial of $K+1$ order approximates the state. We denominate this polynomial by $z^K(t)$, it can be represented in several ways i.e. power series, Newton divided difference, or B-splines. However to develop the collocation method it is preferred the Lagrangian interpolation polynomials, the reason is that this class of polynomial have stability properties, null error for some kind of problems and the formulation is easier because polynomial coefficients have the same bounds as the states, which means that a state constraint can be applied to the approximation coefficients directly.

The approximation with $K+1$ interpolation points in the sub-interval i is defined:

$$t = t_{i-1} + h_i \tau \tag{4.5a}$$

$$z^K(t) = \sum_{j=0}^{K} \ell_j(\tau) z_{ij} \tag{4.5b}$$

where:

$$t \in [t_{i-1}, t_i], \tag{4.6a}$$

$$\tau \in [0, 1] \tag{4.6b}$$

$$\ell_j(\tau) = \prod_{k=0, \neq j}^{K} \frac{(\tau - \tau_k)}{(\tau_j - \tau_k)} \tag{4.6c}$$

with $\tau_0 = 0$, $\tau_j < \tau_{j+1}$, $j = 0, \ldots, K-1$. In which the variables represent:

- τ is the variable that select a specific time in the interval, being 0 the beginning of the interval and 1 the end.
- τ_k and τ_j are collocation points, these values are taken from tables.
- $\ell_j(\cdot)$ is the Lagrangian interpolation base.
- z_{ij} is the state value ate the collocation point j in the interval j, this is the free variable that should be adjusted.

Differentiating Eq. 4.5a:

$$dt = h_i \, d\tau \tag{4.7}$$

Taking the the derivative of the Eq. 4.5b we have

$$\frac{dz^K(t)}{d\tau} = \sum_{j=0}^{K} \frac{d\ell_j(\tau)}{d\tau} z_{ij} \tag{4.8}$$

In the collocation points $t_{ik} = t_{i-1} + h_i \tau_k$, it is desired that the model and the polynomial approximation have the same derivative, so we state:

$$\frac{dz^K(t_{ik})}{dt} = f(z^K(t_{ik}), t_{ik}), \quad k = 1, \ldots, K \tag{4.9}$$

Using the derivative in Eq. 4.7, the derivative in Eq. 4.9, and the state derivative define in Eq. 4.8 we have:

$$\sum_{j=0}^{K} z_{ij} \frac{d\ell_j(\tau_k)}{d\tau} = h_i f(z_{ik}, t_{ik}), \quad k = 1, \ldots, K \tag{4.10}$$

This is the definition for a single sub-interval i. In the case that we have $N > 1$, we need to ensure the continuity of the system stating:

$$z_{1,0} = z_0 \tag{4.11a}$$

$$z_{i+1,0} = \sum_{j=0}^{K} \ell_j(1) z_{ij}, \quad i = 1, \ldots, N-1 \tag{4.11b}$$

$$z_f = \sum_{j=0}^{K} \ell_j(1) z_{Nj} \tag{4.11c}$$

where $z_{1,0}$ is the first state value at the begin of the first sub-interval, $z_{i+1,0}$ is the value of the state at the begin of the interval i, and z_f is the value of the state at the end of the simulation period.

Summarizing, this method creates a time mesh, each time interval is approximated by a polynomial of $K + 1$ order represented in the Langrangrian base. The polynomials are differentiated and forced to be equal to the system derivative. The value of the state at the beginning of the sub-interval is equal to the value at the end of the prior sub-interval. To clarify the understanding, in the next subsection an example will be explained.

4.1.3: Collocation Method Example

For this example, we define the system:

$$\frac{dz}{dt} = z^2 - 2z + 1, \quad z(0) = -3 \tag{4.12}$$

with $t \in [0, 1]$. This system has an analytical solution given by $z(t) = (4t - 3)/(4t + 1)$. However we are going to calculate the numerical approximation using the collocation method in the Radau collocation points with $K = 3$ the collocation points are $\tau_0 = 0$, $\tau_1 = 0.155051$, $\tau_2 = 0.644949$, and $\tau_3 = 1$.

Using the Eq. 4.10 with $N = 1$ elements, and $h = 1/N$ we have

$$\sum_{j=0}^{3} z_{ij} \frac{d\ell_j(\tau_k)}{d\tau} = h(z_{ij}^2 - 2z_{ik}), \quad k = 1, \ldots, 3, \ i = 1 \tag{4.13}$$

Developing the Lagrangian base described in the Eq. 4.6c and taking it derivative we find:

$$\frac{d\ell_0(\tau_k)}{d\tau} = -30\tau_k^2 + 36\tau_k - 9 \tag{4.14a}$$

$$\frac{d\ell_1(\tau_k)}{d\tau} = 46.7423\tau_k^2 - 51.2392\tau_k - 10.0488 \tag{4.14b}$$

$$\frac{d\ell_2(\tau_k)}{d\tau} = -23.7423\tau_k^2 + 20.5925\tau_k - 1.38214 \tag{4.14c}$$

$$\frac{d\ell_3(\tau_k)}{d\tau} = 10\tau_k^2 - \frac{16}{3}\tau_k + \frac{1}{3} \tag{4.14d}$$

$$\tag{4.14e}$$

Substituting Eq. 4.14 in Eq. 4.13 we find:

$$\begin{aligned} &z_0(-30\tau_k^2 + 36\tau_k - 9) + z_1(46.7423\tau_k^2 - 51.2392\tau_k + 10.0488) \\ &z_2(-23.7423\tau_k^2 + 20.5925\tau_k - 1.38214) + z_3\left(10\tau_k^2 - \frac{16}{3}\tau_k + \frac{1}{3}\right) \\ &= (z_k^2 - 2z_k + 1), \quad k = 1, \ldots, 3 \end{aligned} \tag{4.15}$$

Solving this equations system we have $z_1 = -1.65701$, $z_2 = 0.032053$, $z_3 = 0.207272$ with $z_0 = -3$. Figure 4.2 compares the solution using the collocation method and the analytical solution. To increase the fidelity of the approximation we could increase the number of time discretizations N or the number of collocation point K.

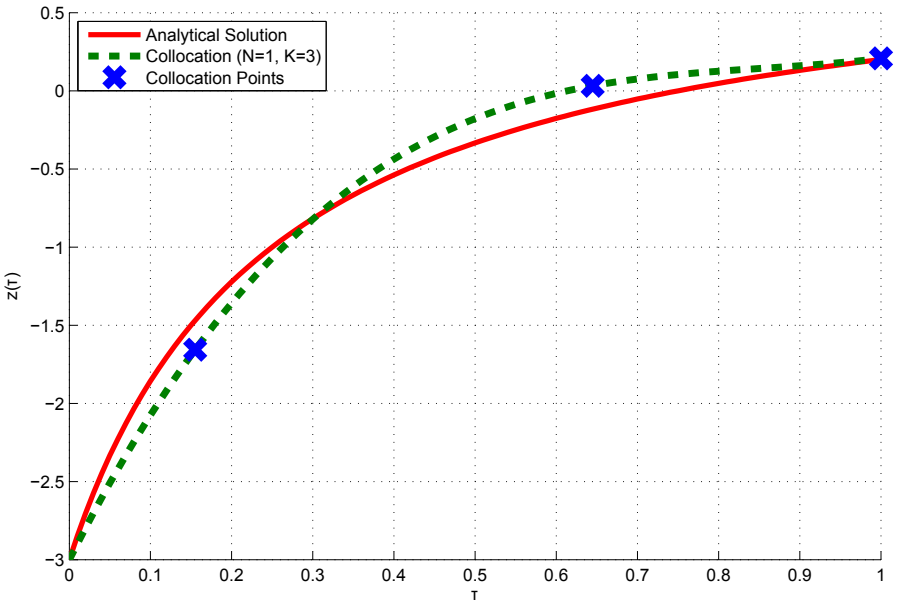

Figure 4.2: Example analytical and approximation solution

4.1.4: NLP Optimization Formulation

Assuming a continuous time dynamic optimization problem in the format:

$$\min \psi(t, x(t), y(t), u(t)) \tag{4.16a}$$

$$\text{s.t.: } \dot{x} = f(t, x(t), y(t), u(t)) \tag{4.16b}$$

$$g(t, x(t), z(t), u(t)) = 0 \tag{4.16c}$$

$$x^L \leq x(t) \leq x^U \tag{4.16d}$$

$$y^L \leq y(t) \leq y^U \tag{4.16e}$$

$$u^L \leq u(t) \leq u^U \tag{4.16f}$$

$$t_0 \leq t \leq t_f \tag{4.16g}$$

where:

- ψ is objective function, usually a time integral.
- $x(t)$, $z(t)$, and $u(t)$ are states, algebraic variables, and controls, respectively.
- $f(\cdot)$ and $g(\cdot)$ are states function and algebraic variables function, respectively.
- The states boundaries are x^L and x^U.
- The algebraic variables boundaries are x^L and x^U.

- The controls boundaries are x^L and x^U.

The state variables can be discretized using the collocation method show in Subsection 4.1.2, but the control profiles and the algebraic variables can also be represented using the Lagrangian interpolation polynomials. Being $u(t)$ the control profiles and the algebraic variables $y(t)$. Using the Lagrangian base inside of the i sub-interval they are defined as:

$$u(t) = \sum_{j=1}^{K} \bar{\ell}_j(\tau) u_{ij} \tag{4.17a}$$

$$y(t) = \sum_{j=1}^{K} \bar{\ell}_j(\tau) y_{ij} \tag{4.17b}$$

$$\tag{4.17c}$$

where:

$$\bar{\ell}_j(\tau) = \prod_{k=1,\neq j}^{K} \frac{(\tau - \tau_k)}{(\tau_j - \tau_k)} \tag{4.18}$$

In the case that the objective function is a time integral, in the format:

$$\psi(x(t), y(t), u(t)) = \int_{t_0}^{t_f} h(x(t), y(t), u(t))\, dt \tag{4.19}$$

we can transform this in a state $\phi(t)$:

$$\dot{\phi} = h(x(t), y(t), u(t)) \tag{4.20}$$

And replace the objective function by $\min \phi(t_f)$, and use the collocation method to approximate $\phi(t)$ like the other state variables.

Using this approximation we can transform the optimization problem in Eq. 4.16 in to the finite element problem:

$$\min \phi(t_f) \tag{4.21a}$$

$$\text{s.t.: } \sum_{j=0}^{K} z_{ij} \frac{d\ell_j(\tau_k)}{d\tau} - h_i f(t, z_{ik}, y_{ik}, u_{ik}, t_{ik}) = 0, \quad k \in \{1, \ldots, K\}, i \in \{1, \ldots, N\} \tag{4.21b}$$

$$g(t, z_{ik}, y_{ik}, u_{ik}, t_{ik}) = 0, \quad k \in \{1, \ldots, K\}, i \in \{1, \ldots, N\} \tag{4.21c}$$

$$z_{i+1,0} = \sum_{j=0}^{K} \ell_j(1) z_{ij}, \quad i \in \{1, \ldots, N-1\} \tag{4.21d}$$

$$z_f = \sum_{j=0}^{K} \ell_j(1) z_{Nj} \tag{4.21e}$$

$$x^L \leq z_{ik} \leq x^U \tag{4.21f}$$

$$y^L \leq y_{ik} \leq y^U \tag{4.21g}$$

$$u^L \leq u_{ik} \leq u^U \tag{4.21h}$$

$$z_{1,0} = z_0 \tag{4.21i}$$

Note that the variable bounds are same of the continuous time formulation, they are applied straight to the collocation points, this ease of application of constraints are one of the reasons that the Lagrangian interpolation polynomial is chosen. However, such constraints can only me ensured at the collocation points and not in the interval between them.

Assuming that z approximates the oil network states, y is the algebraic variable, and $f(\cdot)$ and $g(\cdot)$ their functions as defined in Subsection 2.7 we can optimize the system over a objective function. The objective function will be define in the next section.

An important remark about optimizing using Modelica and Optimica is that the optimization problem that we need to state should follow the format in Eq. 4.16 (continuous time) and the computational environment, in this case JModelica.org, will automatically discretize the problem to the problem in Eq.4.21.

4.2: Objective Function

4.2.1: Tracking Problem

The objective of this formulation is to bring the system from the initial conditions to given reference.

Defined a reference for some of the states, algebraic variables, or inputs, we can formulate a minimization problem to obtain the optimal transient using a quadratic objective function. This objective function integrates the weighted quadratic error between the states, algebraic variables, and controls, and their references.

$$\begin{aligned}\min \psi = \int_{t_0}^{t_f} (z - z_{\text{ref}})^T Q_z (z - z_{\text{ref}}) + (y - y_{\text{ref}})^T Q_y (y - y_{\text{ref}}) \\ + (u - u_{\text{ref}})^T R_u (u - u_{\text{ref}})\ dt\end{aligned} \tag{4.22}$$

where Q_z is the states weighting factor, Q_y is the algebraic variables weighting factor, and R_u is the inputs weighting factor.

This formulation is very common and very flexible allowing the control of different variables of the system. Such objective function could be used to make field production follow a production planing, make the wells to follow production references, or make the system try at most to follow input reference.

4.2.2: Production Maximization Formulation

In this formulation, the developed dynamic model is used to maximize the oil production. This formulation aims to replace the traditional static optimization. Even if it calculation is more costly, it can be more effective for treating events that occurs in a minute or hourly timescale, *i.e.* temporary constraints.

The objective function will maximize the integral of the profit function, which is composed by the gains with oil and gas exportation, and the cost of water treatment, flaring, and gas-lift injection.

$$\max \psi = \int_{t_0}^{t_f} k_g q^g_{\text{exp}} + k_o q^o_{\text{exp}} + k_w q^w_{\text{exp}} + k_{\text{fl}} w_{\text{fl}} + k_{\text{gl}} w_{\text{gm}}\ dt \tag{4.23}$$

where k_g is the gas profit, k_o is the oil production profit, k_w is the water treatment cost, k_{fl} is the flaring cost, k_{gl} the gas-lift injection cost.

4.3: Test Case: Compressor Scheduled Maintenance

To show the potential of the developed model, a test case that goes beyond the system normal operational condition is generated.

The production maximization can be achieved by static optimization, however it may be inefficient for the case that the system is no constant for all the optimization period.

For test purpose, it is suggested that a scheduled compressor maintenance will change the operational condition of the compressor map. It makes the compressor unable to handle the flow at the static production maximum.

To emulate the reduction of the compressor flow capacity the equations that approximate the left and right border of the compressor map that in Section 2.6 were defined as

$$\overline{q}_{l,\mathrm{vol}} = \frac{-b_l + \sqrt{b_l^2 - 4a_l(c_l - \overline{r}_{\mathrm{p}})}}{2a_l} \tag{4.24a}$$

$$\overline{q}_{r,\mathrm{vol}} = \frac{-b_r + \sqrt{b_r^2 - 4a_r(c_r - \overline{r}_{\mathrm{p}})}}{2a_r} \tag{4.24b}$$

are redefined as:

$$\overline{q}_{l,\mathrm{vol}} = f_c(t)\frac{-b_l + \sqrt{b_l^2 - 4a_l(c_l - \overline{r}_{\mathrm{p}})}}{2a_l} \tag{4.25a}$$

$$\overline{q}_{r,\mathrm{vol}} = f_c(t)\frac{-b_r + \sqrt{b_r^2 - 4a_r(c_r - \overline{r}_{\mathrm{p}})}}{2a_r} \tag{4.25b}$$

with:

$$f_c(t) = 1 - k_{\mathrm{cm}}\theta(t - t_t) \tag{4.26}$$

where k_{cm} is a scaling factor that says how much the compressor capability were reduced and $\theta(t - t_t)$ is a Heaviside function (also known as unit step function), and t_t is the event triggering time.

The $\theta(t - t_t)$ is a discontinuous function and it may lead to problems when optimizing the system because the automatic differentiation tool CasADi can not handle it, thus it is approximated by sigmoid function:

$$\theta(t - t_t) \approx \frac{1}{1 + e^{-2(t - t_t)}} \tag{4.27}$$

The Figure 4.3 shows the approximation in a time range of 20 seconds, with the

trigger at 10 seconds.

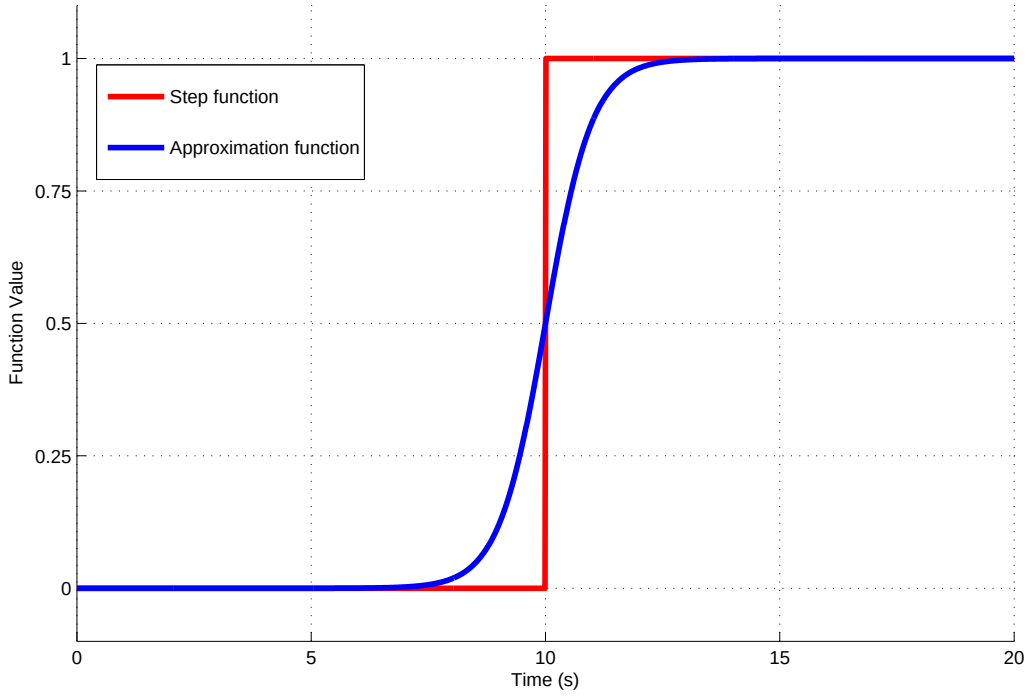

Figure 4.3: Step function approximation

The approximation takes about five seconds to change from zero to one. If we consider that the system dynamics are about thousand seconds, then we can say that this approximation is good enough.

4.3.1: Solving Approaches

Using the formulations defined in Section 4.2 three different approaches are suggested, two of them requires a statical optimal solution for production maximization problem. As the static optimization is out of the scope of this work, an heuristic is proposed.

The heuristic used to find the statical optimal solution uses the the production maximization formulation defined in Subsection 4.2.2 to obtain a control profile. After at transitory moment, the control remains steady for a long period, this steady value is assumed to be the static optimal control. For example, in the Figure 5.4 the optimal control can be obtained at 15000 seconds.

The three approaches are:

1. Naive Approach: using the tracking formulation (Eq. 4.22) makes the system to follow the static optimal control at any cost, the intention of this approach is to simulate the conditions in which no dynamic optimization is used. The injection rates reference and separator pressures reference are defined by the static production maximization solution.

2. Optimal Transient: Obtained solving the production maximization problem in Eq. 4.23. This formulation changes the values of gas-lift injection, separator pressure, anti-surge flow rate, and flare flow rate to obtain the best transient. A high cost on flaring is used to make the algorithm to avoid flaring conditions.

3. Hybrid Approach: use the tracking formulation, as the naive approach, although with a much higher weighting at the flare penalization to avoid situations with flaring.

The naive approach represents the simple application of the static optimization result. The parametrization of Eq. 4.22 have $Q_z = Q_y = 0$ and the quadratic terms related surge control (w_{sl}) are nulls ($R_{u,w_{\mathrm{sl}}} = 0$). As the flaring amount (w_{fl}) is a system input, if we do not define how it should be assigned the optimal solution may have a unnecessary flaring, so a weighting must be given for the flaring. However the objective of this formulation is tracking not flaring reduction, the terms related to tracking should be much larger than those related to the flaring ($(w_{\mathrm{fl}})^T R_{u,w_{\mathrm{fl}}} w_{\mathrm{fl}} \ll (w_{\mathrm{gl}} - w_{\mathrm{gl,ref}})^T R_{u,w_{\mathrm{gl}}} (w_{\mathrm{gl}} - w_{\mathrm{gl,ref}}) + (p^s - p^s_{\mathrm{ref}})^T R_{u,p^s} (p^s - p^s_{\mathrm{ref}})$).

Meanwhile, the Hybrid Approach tries to reach the static optimal at same time that it void to use the flare. Compared to the first approach in which the objective is to follow the optimal static control settings using as less flaring as possible, this approach balance tracking with flaring reduction. The reason is that the using of flaring can be really costly and might be more important to avoid it than to maximize the production. To make this approach an higher value for k_{fl} is adopted, ensuring that the term related to the flare have about the same magnitude than tracking term ($(w_{\mathrm{fl}})^T R_{u,w_{\mathrm{fl}}} w_{\mathrm{fl}} \approx (w_{\mathrm{gl}} - w_{\mathrm{gl,ref}})^T R_{u,w_{\mathrm{gl}}} (w_{\mathrm{gl}} - w_{\mathrm{gl,ref}}) + (p^s - p^s_{\mathrm{ref}})^T R_{u,p^s} (p^s - p^s_{\mathrm{ref}})$)

Chapter 5: Results

In this chapter, the results of the optimizations using the model and the formulations presented in this work are shown. First, an oil production network instance is formulated. Afterwards, it is presented the computation setup used to run the optimization problems. The results of the tracking problem, maximization problem, and the compressor problem approaches are analysed qualitatively. At last, the computational results are reviewed.

5.1: Instance definition

The synthetic oil network was create. It has 8 wells, grouped in two clusters, each group is connected to one of the two subsea manifold, those are connected to two pipeline-risers that brings to production to the surface. At the processing unit, two separators separates the income flow, at the gas outlet of each separator a flare valve is preceded by a compressor. Compressors outflow are divided in two, part goes to the exportation line and part goes to the gas-lift manifold and is reinjected in the wells. The Figure 5.1 illustrates the network.

The parametrization of the wells are displayed in Table 5.1. Besides those parameters, all wells have the same value for reservoir pressure (25×10^6 Pa) and the valve coefficients: gas-lift choke (1.6×10^4), injection valve (1.6×10^4), production choke (1.4×10^3).

Table 5.1: Wells parameters

Well	Annular Vol. m^3	Tubing. vol m^3	Tubing inj. point m	Watercut $^{kg}/_{kg}$	GOR $^{kg}/_{kg}$	$Q_{\max}$ $^{m^3}/_{s}$
1	30	18	400	0.35	0.08	0.025
2	30	18	400	0.80	0.12	0.050
3	25	15	800	0.30	0.10	0.035
4	30	18	600	0.50	0.09	0.010
5	25	15	500	0.25	0.09	0.020
6	30	18	400	0.35	0.08	0.025
7	30	18	400	0.80	0.12	0.050
8	25	15	800	0.30	0.10	0.035

The pipelines have $13\ km$ of horizontal length and $600\ m$ of vertical length, both

parts with $20\ cm$ of diameter, the roughness is $2.8 \times 10^{-5}\ m$ The temperature of fluid inside of the pipeline is assumed to be at $330\ K$.

The separator has $2.62\ m$ of the horizontal length and $1.46\ m$ of diameter. The water droplets have an average diameter of $500\ \mu m$. The water-mixture level is fixed at $0.73\ m$ and the oil level is fixed at $1.02\ m$.

The gas injection must be at least $1\ kg/s$ and not higher than $3\ kg/s$ in each well. At the separators, the minimum pressure is $50 \times 10^5\ Pa$ and maximum pressure is $60 \times 10^5\ Pa$.

The compressor map before the scheduled maintenance have the boundaries as in Figure 5.2, after a reduction of 35% it turns in the map shown in Figure 5.3.

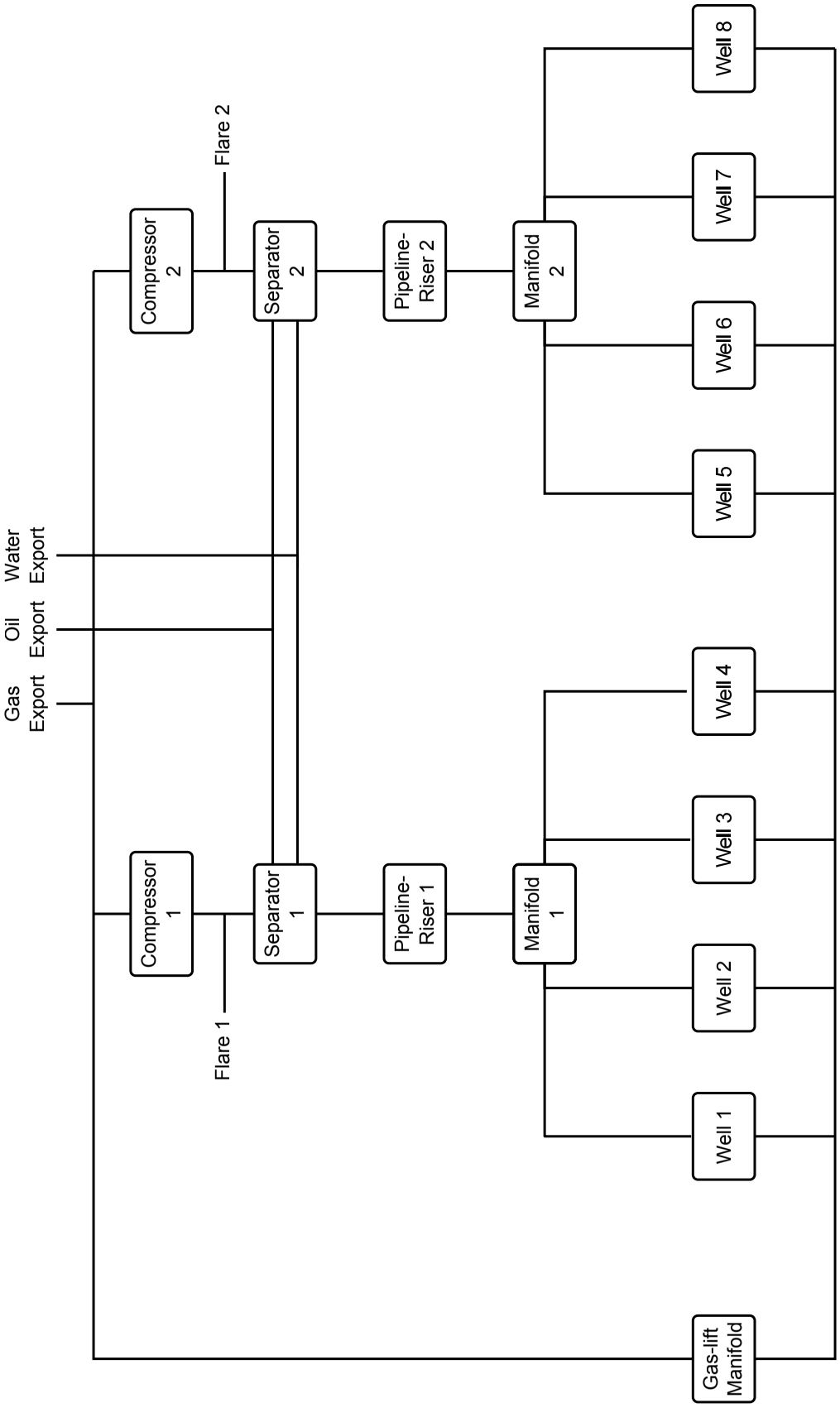

Figure 5.1: Network instance schematic

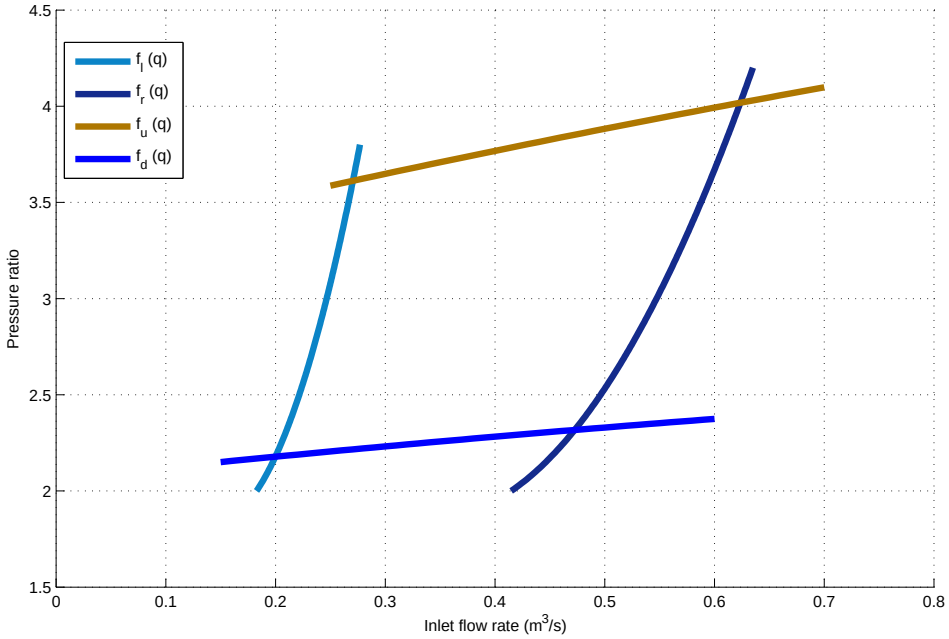

Figure 5.2: Compressor map before maintenance

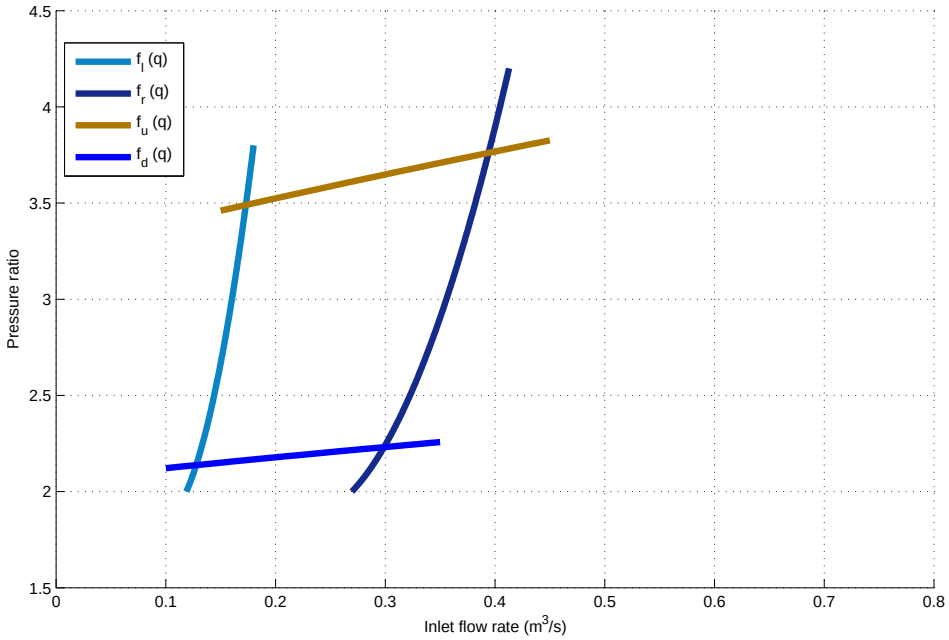

Figure 5.3: Compressor map after maintenance

5.2: Computational Setup

To solve this problems it was used an Asus notebook model N76V, equipped with a processor Intel i7-3610QM @ 2.30 GHz, 8GB of RAM, and a SAMSUNG 830 solid state disk.

This machine is running Windows 7 64 bits. The JModelica.org is version 1.9, which comes with Python 2.8, Sundials package version 2.4, and CasaADi 1.6. The IPOPT general tolerance was set to 10^{-4}, the maximum number of iterations was 100, and the MUMPS parameter "mumps_mem_percent" was 50...

5.3: Maximization Problem

The parameters of Table 5.2 were used to test the objective function defined in Section 4.2.2. The system at the initial condition have already reached the steady state after the application of the control lower bounds: the gas injections of $1kg/s$ and separators pressure of 50×10^5. The system initial guess assumes a gas injections of $1.5kg/s$ in all wells and pressure of 50×10^5 in both separators.

Table 5.2: Flow values

Coefficient	Value
k_g	1
k_o	10
k_w	-1
k_{fl}	-1000
k_{gl}	-1

Figure 5.4 shows the control setup for the production maximization. The algorithm gives an initial blow at the beginning to rise the mass states, since the system was operating in the lower condition. After about 4000 seconds the system settle and remains constant until the last 5000 seconds. At the end of the simulation, the algorithm gives a final blow in the compressor and turns all injections to the minimum, this happens because it increases the derivative of the objective function, as it can be seen in Figure 5.5. To avoid this kind of behavior, some techniques can be used *e.g.* input blocking. The flare values obtained by the solver are so small that can be considered zero.

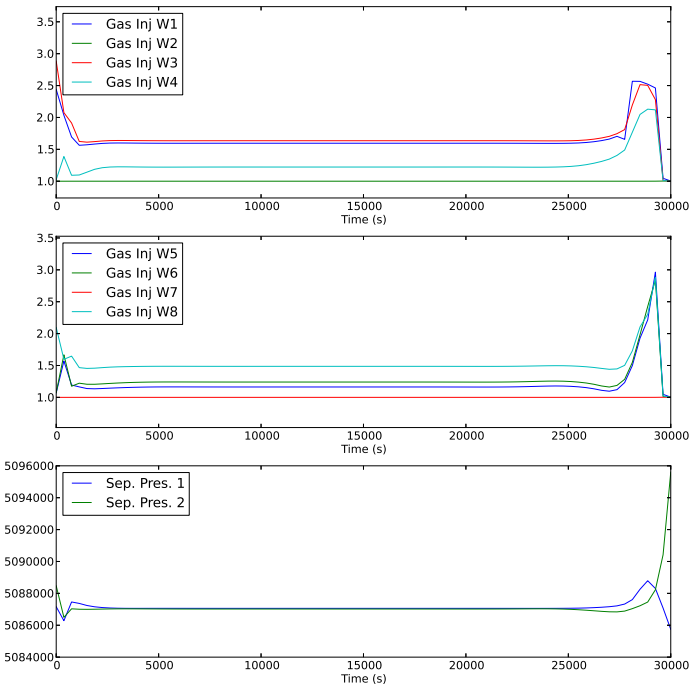

Figure 5.4: Maximization optimal controls

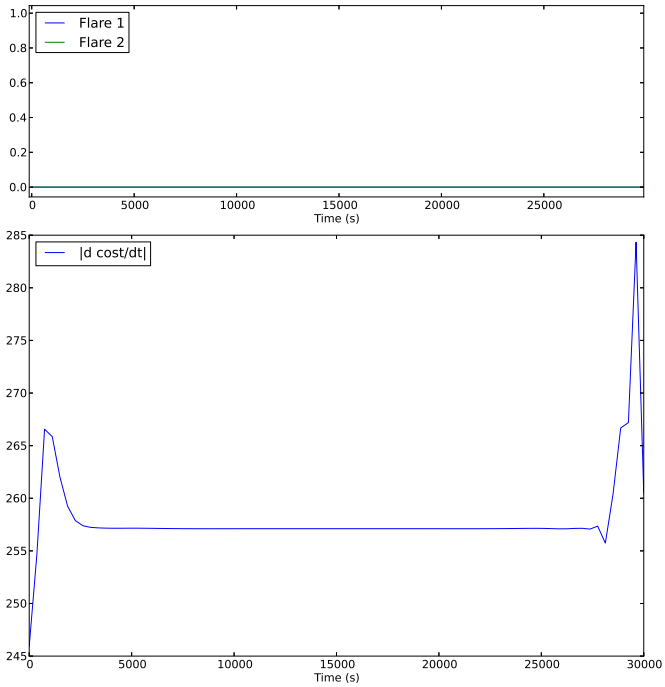

Figure 5.5: Maximization flare and derivative of cost function

5.4: Tracking Problem

The tracking problem was solved using the formulation in Eq. 4.22 with the parameters:

$$Q_{z,w_{\text{out}}} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0.001 \end{bmatrix} \tag{5.1a}$$

$$R_{u,w_{gl}} = 0.1 \tag{5.1b}$$

$$R_{u,w_{fl}} = 1000 \tag{5.1c}$$

The system reference to be tracked are the the gas production of $5\ kg/s$ and the oil production of $25\ kg/s$. All the other parameters were set to zero.

The system initial conditions was the inputs lower bounds: gas injections at $1\ kg/s$ and separator pressure at $50 \times 10^5\ Pa$. The algorithm first guess is obtained using injections at $1.5\ kg/s$ and separators pressure at $50 \times 10^5\ Pa$.

Figure 5.6 shows that after 50000 seconds the controls settle and remain in constant until the end of the prediction window, were the the same behavior that happens in the maximization happens in this formulation.

Figure 5.7 shows that the flaring is null. The gas, oil, and water exportation are kept constant for the all the prediction window after a transitory in the start.

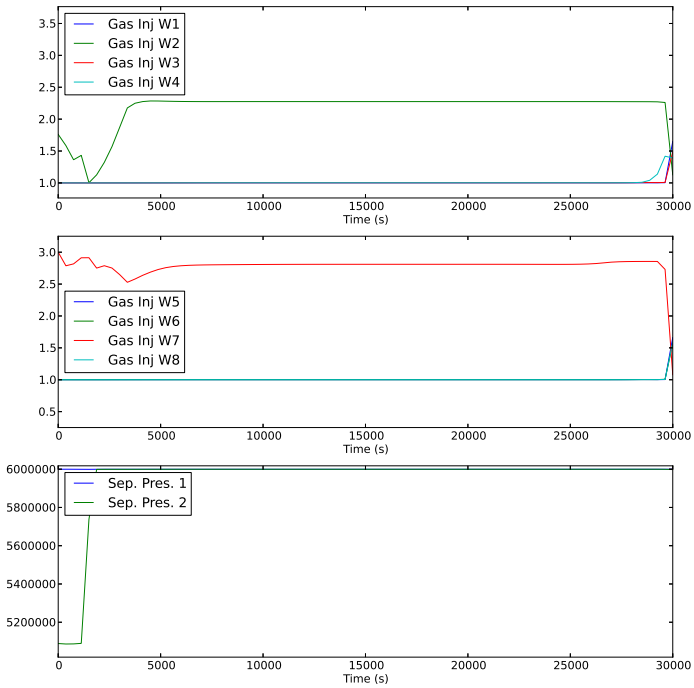

Figure 5.6: Tracking optimal control profiles

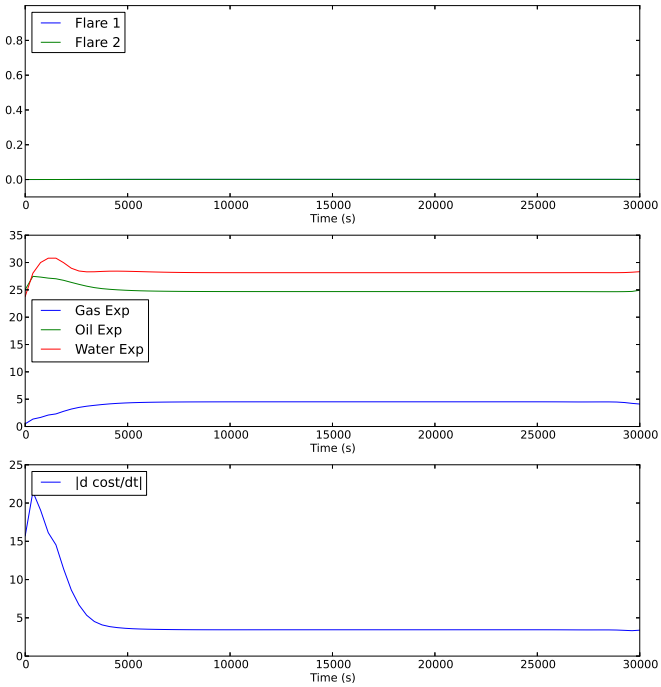

Figure 5.7: Tracking flare, flow exportation, and derivative of cost function

5.5: Compressor Maintenance Results

In this section we show the results of the methods discussed in Section 4.3. In all the cases the system is operating at the static maximum point, which was obtained using the heuristic explained in Subsection 4.3.1. First, the naive approach results are shown and discussed, followed by the transient optimization, and at last, the hybrid approach is reviewed.

The initial guess for the algorithm are $1.5\ kg/s$ of gas injection for all wells and a pressure of $50 \times 10^5 Pa$ in both separators. The plots in this section were chopped in the last 5000 seconds since the "final blow" effect that happens at the end of prediction window, as seen in the last sections, are not interesting for the analysis of the approaches.

5.5.1: Naive Approach

As explained in Section 4.3, this approach requires a steady state optimal solution and an heuristic is used to obtain it. The control assumed to be optimal at steady state are those at 15000 seconds in the Figure 5.4. For the tracking problem the parameters are $R_{u,w_{\mathrm{gl}}} = 10$, $R_{u,p_{\mathrm{s}}} = 10^{-11}$, and $R_{u,w_{\mathrm{Fl}}} = 0.1$. The reason for such small value for the separator pressure tracking is that the pressure is around 5.5 MPa, while the mass flow is in the range from 1 to 3 kg/s.

In Figure 5.8, we can see that the inputs are almost all constant during the first period, showing that the system manage to keep on the reference. A small change in the pressure happens as the system foresee the changes.

After the compressor maintenance, the system is not able to keep the injections reference anymore because there is no gas to feed the wells freely. The solver tries to distribute the available gas, so the injections get as closer as possible to their references. The separator pressure could not remain at the reference since the compressor map changes makes that the static optimal operation point become unfeasible. Figure 5.9, shows that as a reflex of the restraint of the compressor flow capability, the flare needed to be used.

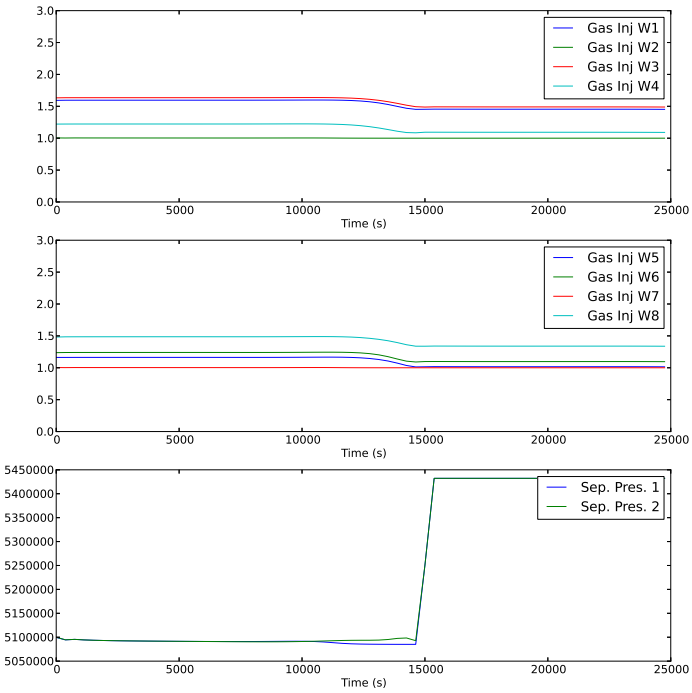

Figure 5.8: Naive approach control settings

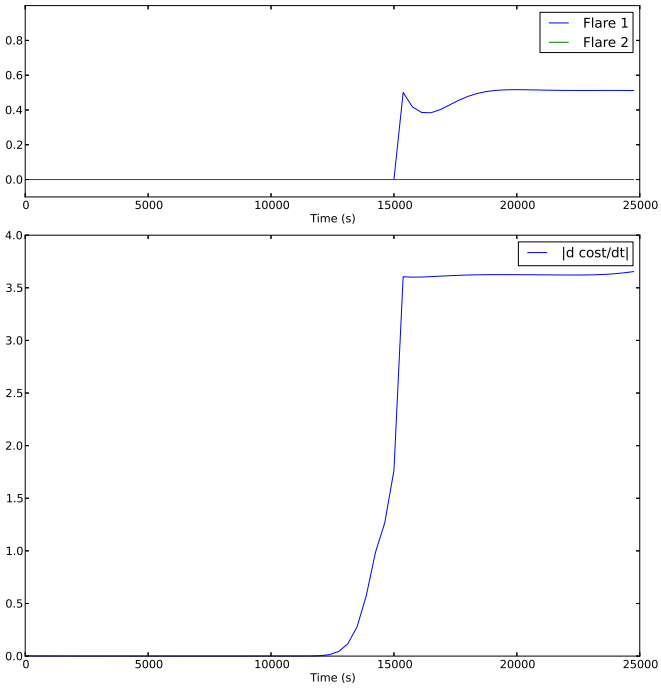

Figure 5.9: Naive approach flare and cost derivative

5.5.2: Transient Optimization

For the transient optimization approach, the coefficients needed in the formulation Eq. 4.23 are given in Table 5.2.

Figure 5.10 shows the control profiles for the maximization problem. We can see that the algorithm foresee the reduction in the compression capability and reduces the flow right before the compressor constraint. As the in the naive approach, the changes forces to increase the pressure in both separators to keep the feasibility. But differently, this formulation can completely suppress the flaring during the transition, as shown in Figure 5.11.

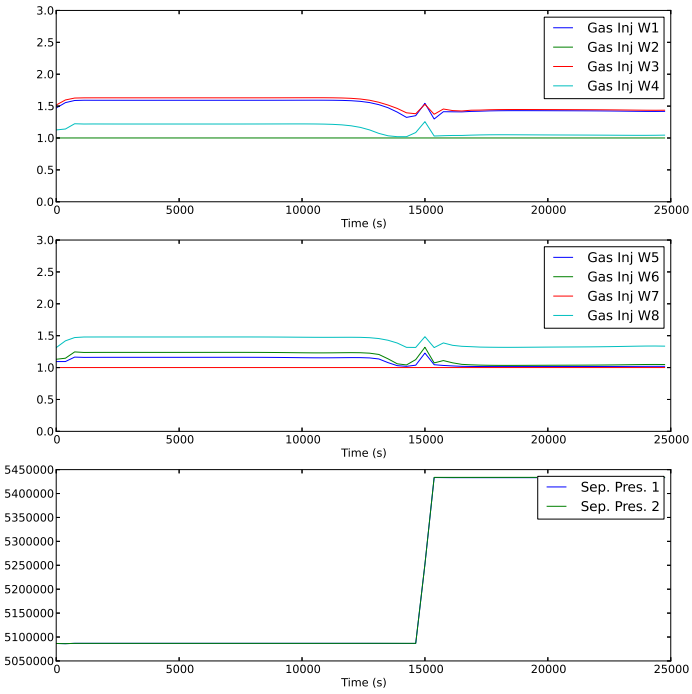

Figure 5.10: Optimal transient control settings

5.5.3: Hybrid Approach

For the hybrid approach, the coefficients of the objective function in Eq. 4.22 were set as $R_{u,w_{\mathrm{gl}}} = 10$, $R_{u,p_{\mathrm{s}}} = 10^{-11}$, and $k_{\mathrm{fl}} = 10000$.

The consequence of this changes, in relation to the naive approach, is a control action much similar to the production maximization approach. As it can be seen in Figure 5.12, the algorithm reduces injection foreseeing the compressor changes, as in

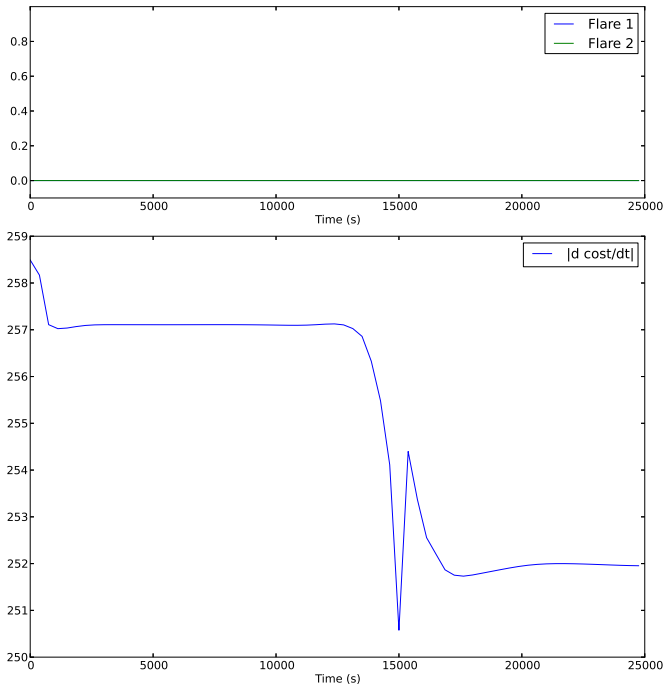

Figure 5.11: Optimal transient flare and cost derivative

the production maximization. However, the separator pressures behaves like the naive approach having some a movement before the compressor capacity reduction.

As shown in Figure 5.13, this formulation can also completely operate during the transition without using the flare, which can not be achieved by the simple application of the static optimal control.

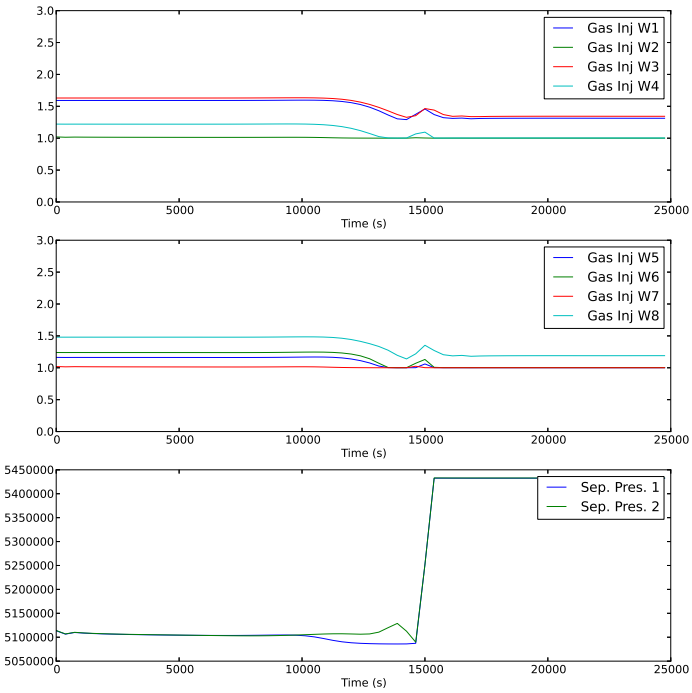

Figure 5.12: Hybrid approach control settings

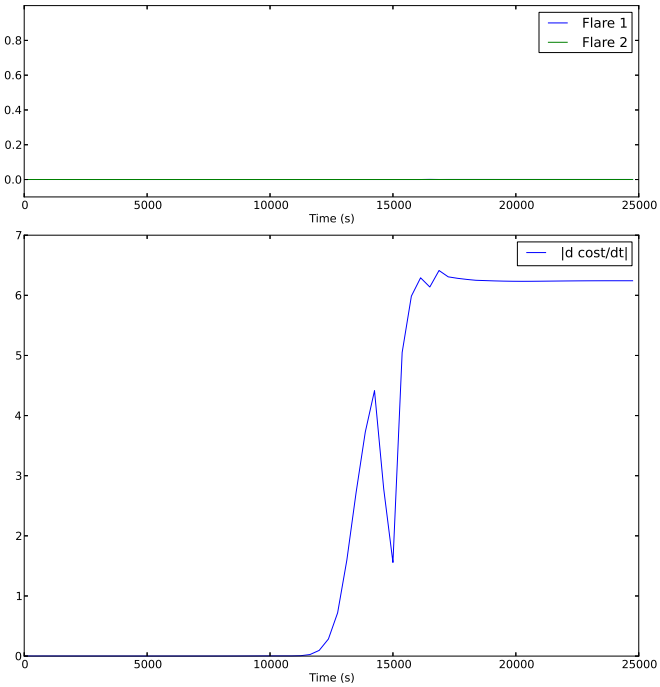

Figure 5.13: Hybrid approach flare and cost derivative

5.6: Computation Analysis

The computational analysis of the methods describe in this section were made based in 5 items: solving time, number of iterations, number of variables, number of equality constraints, and number of inequality constraints.

The first part of the Table 5.3 presents the test of the formulations. We can see that both perform very similarly, even taking an extra iteration the tracking problem have a faster solving time, when compared to the production maximization. The tracking formulation have a slightly higher number of variables and equality constraints, while both have the same number of inequality constraints.

Table 5.3: Formulations and Approaches computational indicators

Method	Solving Time	Iter.	Variables	Eq. Constr.	Ineq. Constr.
Tracking Problem	21.64	22	63216	60976	1288
Production Max.	22.72	21	62493	60253	1288
Naive Approach	24.05	24	63216	60976	1288
Transient Opt.	24.15	27	62332	60092	1288
Hybrid Approach	21.79	23	63216	60976	1288

The second part of the Table 5.3 have the three approaches for the compressor problem. Their solving time do not have a big difference, however the hybrid approach have a faster computing time. While the transient optimization takes more iterations to solve the problem, each iteration is faster if compared to the naive approach, since they have about the same solving time. The transient optimization have a greater number of variables, but this difference may not be the reason for the an increased number of iterations. The number of equality constraint for all there cases are also very similar.

The author points that the comparison of the computational performance can be a little bit subjective, since the methods performance have a big dependence on the algorithm's first guess and the in the problem formulation, and some times they need to be tuned to get the results in the reasonable time.

Chapter 6: Conclusion

In this work, we followed the construction of an dynamic model of network containing models were developed and taken references. This network were composed by the most common elements: wells, manifolds, pipelines, separators, and compressor. However, due to the simplicity and flexibility of the tool that was used, this network could be expanded or have your elements replaced to other studies.

An interesting tool was presented in this work, the JModelica.org which implements compiler for Modelica and Optimica, and gather tools for simulation and optimization. Modelica and Optimica allows to model and create dynamic optimization problems with easiness.

The optimization of this network was made using sophisticated method for dynamic optimization, the collocation method. This is one the preferred methods when a simulator is not desired. The method was shown, after a quick review of ODE solving techniques, followed by and explanatory example and demonstration of how it can be used to formulate nonlinear optimization problems.

An problem was created in order to demonstrate the application of the developed model, the optimization method, and the computational tools. A situation in which the a compressor scheduled maintenance reduces compressor capability. For this problem, three formulations are proposed.

An synthetic oil production network is created to test and compare the methods. The methods have their performance evaluated quantitatively and qualitatively.

As result, we can see that the simple application of the static optimal solution, naive approach, may lead to undesirable conditions. The transient optimization and the hybrid approach were able to operate without creating undesirable flaring conditions. Both were able to leave an maximum operation point and move to another condition without using the flare to relive the gas overflow. Due to system dynamic this problem is not trivial to solve.

Thinking in an application of this work for production maximization, we can have either a dynamic production maximization approach or an layered approach, in which a static production maximization and tracking problem that handles the real time problems. Such layered would have the advantage of allowing more complex problems in

the static optimization layer, possibly including an the wells routing problem.

As continuation of this work, some points can be highlighted:

- in the case in which an static optimization problem define a optimal well-pipeline routing, using the model developed in this work formulate an optimization problem that finds the best time and order to the rerouting be made.
- Make the Modelica codes for the developed model more user friendly and release then, so other researchers can use and extend it for new oil-related researches.
- The JModelica.org passes to CasADi all the states and algebraic equations, however some special type of algebraic equations can be ease simplified. The application some algorithm that simplify the model equations reducing the number of variables and equations could lead to a better computational performance.

Bibliography

[1] Ole M. Aamo, G.O. Eikrem, H.B. Siahaan, and Bjarne A. Foss. Observer Design for Multiphase Flow in Vertical Pipes with Gas Lift - Theory and Experiments. Modeling, Identification and Control: A Norwegian Research Bulletin, 26(2):65–80, 2005.

[2] Marco Aurelio Schmitz Aguiar, Thiago Lima Silva, and Eduardo Camponogara. A Mixed-Integer Convex Formulation for Optimal Operation of Gas-Lifted Oil Fields with Facility, Routing, and Pressure Constraints. In EngOpt 2012 - International Conference on Engineering Optimization, pages 1–10, Rio de Janeiro - Brazil, 2012.

[3] J. Åkesson, K.-E. Årzén, M. Gäfvert, T. Bergdahl, and H. Tummescheit. Modeling and optimization with Optimica and JModelica.org – Languages and tools for solving large-scale dynamic optimization problems. Computers & Chemical Engineering, 34(11):1737–1749, November 2010.

[4] Johan Åkesson, M. Gäfvert, and T. Tummescheit. JModelica – an Open Source Platform for Optimization of Modelica Models. 6th Vienna International Conference on Mathematical Modelling, 2009.

[5] Joel Andersson, J. Åkesson, F Casella, and Moritz Diehl. Integration of CasADi and JModelica.org. In 8th International Modelica Conference, 2011.

[6] Joel Andersson, Johan Åkesson, and Moritz Diehl. CasADi – A symbolic package for automatic differentiation and optimal control. In Recent Advances in Algorithmic Differentiation, volume 87, pages 297–307. Springer Berlin Heidelberg, 2012.

[7] L.T. Biegler. Nonlinear Programming: Concepts, Algorithms, and Applications to Chemical Processes. SIAM e-books. Society for Industrial and Applied Mathematics (SIAM, 3600 Market Street, Floor 6, Philadelphia, PA 19104), 2010.

[8] Benjamin Julian Tø mte Binder. Production Optimization in a Cluster of Gas-Lift Wells. Master's thesis, Norwegian University of Science and Technology, 2012.

[9] Andrés Codas and Eduardo Camponogara. Mixed-integer linear optimization for optimal lift-gas allocation with well-separator routing. European Journal of Operational Research, 217(1):222–231, February 2012.

[10] Florent Di Meglio, Glenn-Ole Kaasa, and Nicolas Petit. A first principle model for multiphase slugging flow in vertical risers. In Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference, pages 8244–8251. IEEE, December 2009.

[11] Gisle Eikrem, Ole Aamo, and Bjarne Foss. On Instability in Gas Lift Wells and Schemes for Stabilization by Automatic Control. SPE Production & Operations, 2008.

[12] Gisle Eikrem, Lars Imsland, and Bjarne Foss. Stabilization of gas-lifted wells based on state estimation. In IFAC Symposium Adchem, pages 1–6, 2004.

[13] Torbjørn Sønstebø Grong. Modeling of Compressor Characterisics and Active Surge Control. Master's thesis, Norwegian University of Science and Technology, 2009.

[14] Vidar Gunnerud and Bjarne Foss. Oil production optimization – A piecewise linear model, solved with two decomposition strategies. Computers & Chemical Engineering, 34(11):1803–1812, November 2010.

[15] AC Hindmarsh, PN Brown, and KE Grant. SUNDIALS: Suite of nonlinear and differential/algebraic equation solvers. ACM Transactions on Mathematical Software, 31(3):363–396, 2005.

[16] Bin Hu. Characterizing gas-lift instabilities. PhD thesis, Norwegian University of Science and Technology, 2004.

[17] Bin Hu and Michael Golan. Gas-lift Instability Resulted Production Loss and Its Remedy by Feedback Control: Dynamical Simulation Results. In Proceedings of SPE International Improved Oil Recovery Conference in Asia Pacific. Society of Petroleum Engineers, October 2003.

[18] Lars Imsland. Topics in Nonlinear Control: Outpit Feedback Stabilization and Control of Positive Systems. PhD thesis, Norwegian University of Science and Technology, 2002.

[19] Esmaeil Jahanshahi and Sigurd Skogestad. Simplified Dynamical Models for Control of Severe Slugging in Multiphase Risers. In Bittanti Sergio, editor, 8th IFAC World Congress, pages 1634–1639, August 2011.

[20] Åkesson. Johan. Optimica – An Extension of Modelica Supporting Dynamic Optimization. In 6th International Modelica Conference 2008, volume 312, Bielefeld, Germany, January 2008.

[21] Vassileios D. Kosmidis, John D. Perkins, and Efstratios N. Pistikopoulos. Optimization of Well Oil Rate Allocations in Petroleum Fields. Industrial & Engineering Chemistry Research, 43(14):3513–3527, July 2004.

[22] Horace Lamb. Hydrodynamics. Cambridge University Press, 1994.

[23] JM Masella, QH Tran, D Ferre, and C Pauchon. Transient simulation of two-phase flows in pipes. International Journal of Multiphase Flow, 24:739–755, 1998.

[24] Gustavo Moises, Tuerte Rolim, and Jose Formigli. GeDIg: Petrobras Corporate Program for Digital Integrated Field Management. In Proceedings of Intelligent Energy Conference and Exhibition, pages 1–7. Society of Petroleum Engineers, February 2008.

[25] Pierre Perrot. A to Z of Thermodynamics. Oxford University Press, 1998.

[26] PF Pickering, GF Hewitt, MJ Watson, and CP Hale. The prediction of flows in production risers-truth & myth. IIR Conference, pages 1–16, 2001.

[27] Agustinho Plucenio, C. A. Ganzaroli, and Daniel J. Pagano. Stabilizing gas-lift well dynamics with free operating point. In 2012 IFAC Workshop on Automatic Control in Offshore Oil and Gas Production, pages 95–100, 2012.

[28] Agustinho Plucenio and Daniel J. Pagano. Gas-lift optimization and control with nonlinear MPC. Advanced Control of Chemical Processes, 7(1), 2009.

[29] AF Sayda and JH Taylor. Modeling and Control of Three-Phase Gravilty Separators in Oil Production Facilities. In American Control Conference ACC '07, 2007.

[30] Thiago Lima Silva, Andres Codas, and Eduardo Camponogara. A Computational Analysis of Convex Combination Models for Multidimensional Piecewise-Linear Approximation in Oil Production Optimization. 2012 IFAC Workshop on Automatic, pages 292–298, 2012.

[31] Marina Stasiak. Tecnica de controle para supressão de golfadas em risers de produção de petroleo. Master's thesis, Universidade Federal de Santa Catarina, 2012.

[32] Marina Stasiak, Daniel J. Pagano, and Agustinho Plucenio. A new discrete slug-flow controller for production pipeline risers. In 2012 IFAC Workshop on Automatic Control in Offshore Oil and Gas Production, pages 122–127, Trondheim, Norway, 2012.

[33] Y Taitel and D Barnea. Simplified transient simulation of two phase flow using quasi-equilibrium momentum balances. International Journal of Multiphase Flow, 23(3):493–501, June 1997.

[34] Yehuda Taitel, Ovadia Shoham, and JP Brill. Simplified transient solution and simulation of two-phase flow in pipelines. Chemical Engineering Science, 44(6):1353–1359, January 1989.

[35] J.V. Vogel. Inflow Performance Relationships for Solution-Gas Drive Wells. Journal of Petroleum Technology, 20(1), January 1968.

[36] Andreas Wächter and Lorenz T. Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. Mathematical Programming, 106(1):25–57, April 2005.

[37] MJ Watson. Flow regime transitions and associated phenomena. PhD thesis, London University, 1999.

Appendix A : Dynamic Models

A.1: Well Model

These are the equations of the model described in 2.2.

Mass balances:

$$\dot{m}_{\mathrm{ga}} = w_{\mathrm{gl}} - w_{\mathrm{gi}} \tag{A.1a}$$

$$\dot{m}_{\mathrm{gt}} = w_{\mathrm{gr}} + w_{\mathrm{gi}} - w_{\mathrm{gp}} \tag{A.1b}$$

$$\dot{m}_{\mathrm{lt}} = w_{\mathrm{gl}} - w_{\mathrm{gi}} \tag{A.1c}$$

Mass flows:

$$w_{\mathrm{gi}} = C_{\mathrm{iv}} \sqrt{\rho_{\mathrm{gi}} \max(0, p_{\mathrm{ai}} - p_{\mathrm{ti}})} \tag{A.2a}$$

$$w_{\mathrm{p}} = C_{\mathrm{pc}} \sqrt{\rho_{\mathrm{p}} \max(0, p_{\mathrm{ai}} - p_{\mathrm{ti}})} \tag{A.2b}$$

$$w_{\mathrm{gp}} = \frac{m_{\mathrm{gt}}}{m_{\mathrm{t}}} w_{\mathrm{p}} \tag{A.2c}$$

$$w_{\mathrm{lp}} = \frac{m_{\mathrm{lt}}}{m_{\mathrm{t}}} w_{\mathrm{p}} \tag{A.2d}$$

$$w_{\mathrm{op}} = (1 - r_{\mathrm{wc}}) w_{\mathrm{lp}} \tag{A.2e}$$

$$w_{\mathrm{wp}} = r_{\mathrm{wc}} w_{\mathrm{lp}} \tag{A.2f}$$

$$w_{\mathrm{lr}} = \rho_{\mathrm{l}} Q_{\max} \left(1 - (1 - C) \left(\frac{p_{\mathrm{bh}}}{p_{\mathrm{r}}} \right) - C \left(\frac{p_{\mathrm{bh}}}{p_{\mathrm{r}}} \right)^2 \right) \tag{A.2g}$$

$$w_{\mathrm{gr}} = r_{\mathrm{glr}} w_{\mathrm{lr}} \tag{A.2h}$$

$$w_{\mathrm{gl}}^{\max} = C_{\mathrm{gl}} \sqrt{\rho_{\mathrm{gl}} \max(0, p_{\mathrm{gm}} - p_{\mathrm{ta}})} \tag{A.2i}$$

Pressures

$$p_{\mathrm{ai}} = \left(\frac{RT_{\mathrm{a}}}{V_{\mathrm{a}} M_{\mathrm{g}}} + \frac{g}{2A_{\mathrm{a}}} \right) m_{\mathrm{ga}} \tag{A.3a}$$

$$p_{\mathrm{p}} = \frac{RT_{\mathrm{t}} m_{\mathrm{gt}}}{M_{\mathrm{g}} V_{\mathrm{t}} - M_{\mathrm{g}} \rho_{\mathrm{l}}^{-1} m_{\mathrm{lt}}} - \frac{g m_{\mathrm{t}}}{2A_{\mathrm{t}}} \tag{A.3b}$$

$$p_{\mathrm{ti}} = p_{\mathrm{p}} + \frac{g m_{\mathrm{t}}}{A_{\mathrm{t}}} \tag{A.3c}$$

$$p_{\mathrm{bh}} = \frac{\left(1 + r_{\mathrm{glr}} + \dfrac{r_{\mathrm{glr}} g M_{\mathrm{g}} L_{\mathrm{w}}}{2RT_{\mathrm{t}}} \right) p_{\mathrm{ti}} + \rho_{\mathrm{l}} g L_{\mathrm{w}}}{1 + r_{\mathrm{glr}} - \dfrac{r_{\mathrm{glr}} g M_{\mathrm{g}} L_{\mathrm{w}}}{2RT_{\mathrm{t}}}} \tag{A.3d}$$

$$p_{\mathrm{ta}} = \left(\frac{RT_{\mathrm{a}}}{V_{\mathrm{a}} M_{\mathrm{g}}} - \frac{g}{2A_{\mathrm{a}}} \right) m_{\mathrm{ga}} \tag{A.3e}$$

Densities:

$$\rho_{\mathrm{gi}} = \frac{M_{\mathrm{g}}}{RT_{\mathrm{a}}} p_{\mathrm{ai}} \tag{A.4a}$$

$$\rho_{\mathrm{p}} = \frac{\rho_{\mathrm{l}} M_{\mathrm{g}} p_{\mathrm{p}} m_{\mathrm{t}}}{\rho_l R T_{\mathrm{t}} m_{\mathrm{lt}} + M_{\mathrm{g}} p_{\mathrm{p}} m_{\mathrm{gt}}} \tag{A.4b}$$

$$\rho_{\mathrm{l}} = r_{\mathrm{wc}} \rho_{\mathrm{w}} + (1 - r_{\mathrm{wc}}) \rho_{\mathrm{o}} \tag{A.4c}$$

$$\rho_{\mathrm{gl}} = \frac{M_{\mathrm{g}}}{RT_{\mathrm{a}}} p_{\mathrm{gm}} \tag{A.4d}$$

Table A.1: Well model variables

Name	Type	Explanation
$\dot{m}_{\mathrm{ga}}$	State	Mass of gas in the annulus
$\dot{m}_{\mathrm{gt}}$	State	Mass of gas in the tubing
$\dot{m}_{\mathrm{lr}}$	State	Mass of liquid in the tubing
w_{gi}	Algebraic	Mass flow through the injection valve
w_{p}	Algebraic	Mass flow through production choke
w_{gp}	Algebraic	Gas production
w_{lp}	Algebraic	Liquid production
w_{op}	Algebraic	Oil production
w_{wp}	Algebraic	Water production
w_{lr}	Algebraic	Liquid extracted from reservoir
w_{gr}	Algebraic	Gas extracted from the reservoir
ρ_{gi}	Algebraic	Gas density upstream of the injection valve
ρ_{p}	Algebraic	Production density upstream of the production choke
p_{ai}	Algebraic	Pressure upstream of the injection valve
p_{p}	Algebraic	Pressure upstream of the production valve
p_{ti}	Algebraic	Pressure downstream of the injection
p_{bh}	Algebraic	Pressure at the bottom hole
p_{ai}	Algebraic	Pressure upstream of the injection valve

Table A.2: Well model variables

Name	Type	Explanation
C_{iv}	Tunning	Injection valve coefficient
C_{pc}	Tunning	Production Choke
r_{wc}	Reservoir	Reservoir watercut
Q_{max}	Reservoir	Reservoir theoretical maximum flow
C	Constant	Voegel's constant
p_{r}	Reservoir	Reservoir pressure
r_{gor}	Reservoir	Gas-oil ratio
R	Constant	Ideal gas constant
T_{a}	Well	Annulus temperature
g	Constant	Gravity
V_{a}	Well	Annulus Volume
M_{g}	Parameter	Gas molar mass
A_{a}	Well	Annulus section area
T_{t}	Well	Tubing temperature
V_{t}	Well	Tubing Volume
A_{t}	Well	Tubing section area
r_{glr}	Reservoir	Gas-liquid ratio
L_{w}	Well	Tubing length from reservoir to injection point
ρ_{l}	Parameter	Liquid mass density
ρ_{w}	Parameter	Water mass density
ρ_{o}	Parameter	Oil mass density

A.2: Flowline Model

These are the equations of the model described in 2.4.

Mass balances:

$$\dot{m}_{\mathrm{g,p}} = w_{\mathrm{g,in}} - w_{\mathrm{g,lp}} \tag{A.5a}$$

$$\dot{m}_{\mathrm{o,p}} = w_{\mathrm{o,in}} - w_{\mathrm{o,lp}} \tag{A.5b}$$

$$\dot{m}_{\mathrm{w,p}} = w_{\mathrm{w,in}} - w_{\mathrm{w,lp}} \tag{A.5c}$$

$$\dot{m}_{\mathrm{g,r}} = w_{\mathrm{g,lp}} - w_{\mathrm{g,out}} \tag{A.6a}$$

$$\dot{m}_{\mathrm{o,r}} = w_{\mathrm{o,lp}} - w_{\mathrm{o,out}} \tag{A.6b}$$

$$\dot{m}_{\mathrm{w,r}} = w_{\mathrm{w,lp}} - w_{\mathrm{w,out}} \tag{A.6c}$$

Horizontal pipeline:

$$V_{\mathrm{g,p}} = V_{\mathrm{p}} - \frac{m_{\mathrm{o,p}}}{\rho_{\mathrm{o}}} - \frac{m_{\mathrm{w,p}}}{\rho_{\mathrm{w}}} \tag{A.7a}$$

$$p_{\mathrm{in}} = \frac{m_{\mathrm{g,p}} R T_{\mathrm{p}}}{V_{\mathrm{g,p}} M_{\mathrm{g}}} \tag{A.7b}$$

$$\rho_{\mathrm{g,p}} = \frac{m_{\mathrm{g,p}}}{V_{\mathrm{g,p}}} \tag{A.7c}$$

$$\rho_{\mathrm{l,p}} = \frac{\rho_{\mathrm{o}}\rho_{\mathrm{w}}}{\rho_{\mathrm{w}} m_{\mathrm{o,p}} + \rho_{\mathrm{o}} m_{\mathrm{w,p}}} (m_{\mathrm{o,p}} + m_{\mathrm{w,p}}) \tag{A.7d}$$

$$\alpha_{\mathrm{l,p}} = \frac{\rho_{\mathrm{w}} m_{\mathrm{o,p}} + \rho_{\mathrm{o}} m_{\mathrm{w,p}}}{\rho_{\mathrm{o}}\rho_{\mathrm{w}} V_{\mathrm{p}}} \tag{A.7e}$$

$$U_{\mathrm{s,l,p}} = \frac{\rho_{\mathrm{w}} w_{\mathrm{o,in}} + \rho_{\mathrm{o}} w_{\mathrm{w,in}}}{\rho_{\mathrm{o}}\rho_{\mathrm{w}} \pi r_{\mathrm{p}}^2} \tag{A.7f}$$

$$\mu_{\mathrm{l,p}} = \frac{m_{\mathrm{o,p}}}{m_{\mathrm{o,p}} + m_{\mathrm{w,p}}} \mu_{\mathrm{o}} + \frac{m_{\mathrm{w,p}}}{m_{\mathrm{o,p}} + m_{\mathrm{w,p}}} \mu_{\mathrm{w}} \tag{A.7g}$$

$$Re_{\mathrm{p}} = \frac{2\rho_{\mathrm{l,p}} U_{\mathrm{s,l,p}} r_{\mathrm{p}}}{\mu_{\mathrm{l,p}}} \tag{A.7h}$$

$$f_{\mathrm{p}} = \left\{ -1.8 \log_{10} \left[\left(\frac{\varepsilon_{\mathrm{p}}}{3.7 D_{\mathrm{p}}} \right)^{1.11} + \frac{6.9}{Re_{\mathrm{p}}} \right] \right\}^{-2} \tag{A.7i}$$

$$\Delta P_{\mathrm{f,p}} = \frac{\alpha_{\mathrm{l,p}} L_{\mathrm{p}} \rho_{\mathrm{l,p}} f_{\mathrm{p}} U_{\mathrm{s,l,p}}^2}{4 r_{\mathrm{p}}} \tag{A.7j}$$

Vertical riser:

$$V_{\mathrm{g,r}} = V_{\mathrm{r}} - \frac{m_{\mathrm{o,r}}}{\rho_{\mathrm{o}}} - \frac{m_{\mathrm{w,r}}}{\rho_{\mathrm{w}}} \tag{A.8a}$$

$$p_{\mathrm{r,t}} = \frac{\rho_{\mathrm{g,r}} R T_{\mathrm{r}}}{M_{\mathrm{g}}} \tag{A.8b}$$

$$\rho_{\mathrm{g,r}} = \frac{m_{\mathrm{g,r}}}{V_{\mathrm{g,r}}} \tag{A.8c}$$

$$\rho_{\mathrm{l,r}} = \frac{\rho_{\mathrm{o}}\rho_{\mathrm{w}}}{\rho_{\mathrm{w}} m_{\mathrm{o,r}} + \rho_{\mathrm{o}} m_{\mathrm{w,r}}} (m_{\mathrm{o,r}} + m_{\mathrm{w,p}}) \tag{A.8d}$$

$$\rho_{\mathrm{m,r}} = \frac{m_{\mathrm{g,r}} + m_{\mathrm{o,r}} + m_{\mathrm{w,r}}}{V_{\mathrm{r}}} \tag{A.8e}$$

$$\alpha_{\mathrm{l,r}} = \frac{\rho_{\mathrm{w}} m_{\mathrm{o,r}} + \rho_{\mathrm{o}} m_{\mathrm{w,r}}}{\rho_{\mathrm{o}}\rho_{\mathrm{w}} V_{\mathrm{r}}} \tag{A.8f}$$

$$U_{\mathrm{s,l,r}} = \frac{w_{\mathrm{o,in}}\rho_{\mathrm{w}} + w_{\mathrm{w,in}}\rho_{\mathrm{o}}}{\rho_{\mathrm{o}}\rho_{\mathrm{w}} A_{\mathrm{r}}} \tag{A.8g}$$

$$U_{\mathrm{s,g,r}} = \frac{w_{\mathrm{g,in}}}{\rho_{\mathrm{g,r}} A_{\mathrm{r}}} \tag{A.8h}$$

$$U_{\mathrm{m,r}} = U_{\mathrm{s,l,r}} + U_{\mathrm{s,g,r}} \tag{A.8i}$$

$$\mu_{\mathrm{l,r}} = \frac{m_{\mathrm{o,r}}}{m_{\mathrm{o,r}} + m_{\mathrm{w,r}}} \mu_{\mathrm{o}} + \frac{m_{\mathrm{w,r}}}{m_{\mathrm{o,r}} + m_{\mathrm{w,r}}} \mu_{\mathrm{w}} \tag{A.8j}$$

$$Re_{\mathrm{r}} = \frac{2\rho_{\mathrm{m,r}} U_{\mathrm{m,r}} r_{\mathrm{r}}}{\mu_{\mathrm{l,r}}} \tag{A.8k}$$

$$f_{\mathrm{r}} = \left\{ -1.8 \log_{10} \left[\left(\frac{\varepsilon_{\mathrm{p}}}{3.7 D_{\mathrm{r}}} \right)^{1.11} + \frac{6.9}{Re_{\mathrm{r}}} \right] \right\}^{-2} \tag{A.8l}$$

$$\Delta P_{\mathrm{f,r}} = \frac{\alpha_{\mathrm{l,r}} L_{\mathrm{r}} \rho_{\mathrm{l,r}} f_{\mathrm{r}} U_{\mathrm{s,l,p}}^2}{4 r_{\mathrm{r}}} \tag{A.8m}$$

Connection between pipeline and riser:

$$p_{\mathrm{l,p}} = p_{\mathrm{in}} - \Delta P_{\mathrm{f,p}} \tag{A.9a}$$

$$p_{\mathrm{l,r}} = p_{\mathrm{r,t}} + \Delta P_{\mathrm{f,r}} + \rho_{\mathrm{m,r}} g L_{\mathrm{r}} \tag{A.9b}$$

$$A_{\mathrm{g,p}} = \frac{V_{\mathrm{g,p}}}{V_{\mathrm{p}}} A_{\mathrm{p}} \tag{A.9c}$$

$$A_{\mathrm{l,p}} = A_{\mathrm{p}} - A_{\mathrm{g,p}} \tag{A.9d}$$

$$\alpha_{\mathrm{l,t}} = \frac{2(m_{o,r} + m_{w,r})}{V_{\mathrm{r}} \rho_{\mathrm{l,r}}} - \frac{A_{\mathrm{l,p}}}{A_{\mathrm{p}}} \tag{A.9e}$$

$$\rho_{\mathrm{t,m}} = \alpha_{\mathrm{l,t}} \rho_{\mathrm{l,r}} + (1 - \alpha_{\mathrm{l,t}}) \rho_{\mathrm{g,r}} \tag{A.9f}$$

$$w_{\mathrm{g,lp}} = K_{\mathrm{g,p}} A_{\mathrm{g,p}} \sqrt{\rho_{\mathrm{g,p}} \max\{0, p_{\mathrm{lp,p}} - p_{\mathrm{lp,r}}\}} \tag{A.9g}$$

$$w_{\mathrm{o,lp}} = K_{\mathrm{l,p}} A_{\mathrm{O,p}} \sqrt{\rho_{\mathrm{o}} \max\{0, p_{\mathrm{lp,p}} - p_{\mathrm{lp,r}}\}} z_{\mathrm{o,p}} \frac{m_{\mathrm{o,r}}}{m_{\mathrm{g,r}} + m_{\mathrm{o,r}} + m_{\mathrm{w,r}}} \tag{A.9h}$$

$$w_{\mathrm{w,lp}} = K_{\mathrm{l,p}} A_{\mathrm{W,p}} \sqrt{\rho_{\mathrm{w,p}} \max\{0, p_{\mathrm{lp,p}} - p_{\mathrm{lp,r}}\}} z_{\mathrm{w,p}} \frac{m_{\mathrm{w,r}}}{m_{\mathrm{g,r}} + m_{\mathrm{o,r}} + m_{\mathrm{w,r}}} \tag{A.9i}$$

$$w_{\mathrm{g,out}} = K_{\mathrm{g,r}} \sqrt{\rho_{\mathrm{t,m}} \max\{0, p_{\mathrm{r,t}} - p_{\mathrm{out}}\}} \frac{m_{\mathrm{g,r}}}{m_{\mathrm{g,r}} + m_{\mathrm{o,r}} + m_{\mathrm{w,r}}} \tag{A.9j}$$

$$w_{\mathrm{o,out}} = K_{\mathrm{l,r}} \sqrt{\rho_{\mathrm{t,m}} \max\{0, p_{\mathrm{r,t}} - p_{\mathrm{out}}\}} \frac{m_{\mathrm{o,r}}}{m_{\mathrm{g,r}} + m_{\mathrm{o,r}} + m_{\mathrm{w,r}}} \tag{A.9k}$$

$$w_{\mathrm{w,out}} = K_{\mathrm{l,r}} \sqrt{\rho_{\mathrm{t,m}} \max\{0, p_{\mathrm{r,t}} - p_{\mathrm{out}}\}} \frac{m_{\mathrm{w,r}}}{m_{\mathrm{g,r}} + m_{\mathrm{o,r}} + m_{\mathrm{w,r}}} \tag{A.9l}$$

A.3: Manifold

These are the equations of the model described in 2.3.

$$w_{\mathrm{out}} = \sum_{i \in \mathcal{I}} \beta^i w_{\mathrm{in}}^i \tag{A.10a}$$

$$p_{\mathrm{in}}^i = p_{\mathrm{out}}, \qquad \forall i \in \mathcal{I} \tag{A.10b}$$

$$\beta^i \in \{0, 1\} \qquad \forall i \in \mathcal{I} \tag{A.10c}$$

Table A.3: Manifold model variables

Name	Type	Explanation
w_{in}^{i}	Variable	Manifold i-th inlet flow
p_{in}^{i}	Variable	Manifold i-th inlet pressure
w_{out}	Variable	Manifold outlet flow
p_{out}	Variable	Manifold outlet pressure
β^{i}	Binary Variable	Opens/Closes the inlet

A.4: Separator

These are the equations of the model described in 2.5.

Separator:

$$v_{\mathrm{v}} = \frac{2}{9}\frac{(\rho_{\mathrm{h}} - \rho_{\mathrm{w}})g d_{\mathrm{d}}^2}{\mu_{\mathrm{w}}} \tag{A.11a}$$

$$v_{\mathrm{h}} = \frac{L}{\tau} \tag{A.11b}$$

$$\tau = \frac{m_w}{w^w_{\mathrm{w,out}}} \tag{A.11c}$$

$$\theta = \cos^{-1}\left(\frac{R-h}{R}\right) \tag{A.11d}$$

$$\Phi = \tan^{-1}\left(\frac{h_{\mathrm{w}}}{L}\right) \tag{A.11e}$$

$$\Phi_1 = \tan^{-1}\left(\frac{v_{\mathrm{v}}}{v_{\mathrm{h}}}\right) \tag{A.11f}$$

$$L_1 = h\cot\left[\min\left(\Phi, \Phi_1\right)\right] \tag{A.11g}$$

$$h_1 = L\tan\left[\min\left(\Phi, \Phi_1\right)\right] \tag{A.11h}$$

$$\theta_1 = \cos^{-1}\left(1 - \frac{h_1}{R}\right) \tag{A.11i}$$

$$V_{\mathrm{S1}} = R^2 L_1\left\{\theta - 0.5\sin(2\theta) - \frac{3\sin\theta - 3\theta\cos\theta - \sin^3\theta}{3(1-\cos\theta)}\right\} \tag{A.11j}$$

$$V_{\mathrm{S2}} = R^2 L\left\{\theta - 0.5\sin(2\theta) - \frac{3\sin\theta_1 - 3\theta_1\cos\theta_1 - \sin^3\theta_1}{3(1-\cos\theta_1)}\right\} \tag{A.11k}$$

Water phase:

$$\varepsilon = \frac{V_{\mathrm{S2}}}{V_{\mathrm{S1}}} \tag{A.12a}$$

$$w^g_{\mathrm{w,out}} = (1-\varepsilon)w^g_{\mathrm{in}} \tag{A.12b}$$

$$w^o_{\mathrm{w,out}} = (1-\varepsilon)w^o_{\mathrm{in}} \tag{A.12c}$$

$$w^w_{\mathrm{w,out}} = w^w_{\mathrm{in}} \tag{A.12d}$$

Oil Phase:

$$x = \frac{P_{\mathrm{in}}}{P_v} \tag{A.13a}$$

$$w^g_{\mathrm{o,out}} = x\varepsilon w^g_{\mathrm{in}} \tag{A.13b}$$

$$w^o_{\mathrm{o,out}} = \varepsilon w^o_{\mathrm{in}} \tag{A.13c}$$

Gas Phase:

$$w^g_{\mathrm{g,out}} = (1-x)\varepsilon w^g_{\mathrm{in}} \tag{A.14a}$$

Table A.4: Well model variables

Name	Type	Explanation
v_{h}	Algebraic	Droplets' horizontal velocity
τ	Algebraic	Time for water cross the separator
m_{w}	Algebraic	Water mass in separator
w_{gi}	Algebraic	Mass flow through the injection valve
w_{p}	Algebraic	Mass flow through production choke
w_{gp}	Algebraic	Gas production
w_{lp}	Algebraic	Liquid production
w_{op}	Algebraic	Oil production
w_{wp}	Algebraic	Water production
w_{lr}	Algebraic	Liquid extracted from reservoir
w_{gr}	Algebraic	Gas extracted from the reservoir
ρ_{gi}	Algebraic	Gas density upstream of the injection valve
ρ_{p}	Algebraic	Production density upstream of the production choke
p_{ai}	Algebraic	Pressure upstream of the injection valve
p_{p}	Algebraic	Pressure upstream of the production valve
p_{ti}	Algebraic	Pressure downstream of the injection
p_{bh}	Algebraic	Pressure at the bottom hole
p_{ai}	Algebraic	Pressure upstream of the injection valve

A.5: Compressor

These are the equations of the model described in 2.6.

Flows:

$$w_{\text{out}} = w_{\text{in}} \tag{A.15a}$$

$$\widehat{w}_{\text{in}} = w_{\text{in}} + w_{\text{sl}} \tag{A.15b}$$

$$w_{\text{out}} = \widehat{w}_{\text{out}} - w_{\text{sl}} \tag{A.15c}$$

$$q_{\text{vol}} = \frac{\widehat{w}_{\text{in}} RT}{p_{\text{in}} M_{\text{g}}} \tag{A.15d}$$

$$\overline{q}_{\text{vol}} = q_{\text{vol}} \frac{T_{\text{map}}}{T} \frac{p_{\text{in}}}{p_{\text{map}}} \tag{A.15e}$$

Pressures:

$$r_{\text{p}} = \frac{p_{\text{out}}}{p_{\text{in}}} \tag{A.16a}$$

$$\overline{r}_{\text{p}} = r_{\text{p}} \frac{T}{T_{\text{map}}} \tag{A.16b}$$

Quadratic approximations:

$$f_{\text{i}}(\overline{q}_{\text{vol}}) = a_i \overline{q}_{\text{vol}}^2 + b_i \overline{q}_{\text{vol}} + c_i, \quad \forall i \in \{u, d, l, r\} \tag{A.17a}$$

$$\overline{q}_{l,\text{vol}} = \frac{-b_l + \sqrt{b_l^2 - 4a_l(c_l - \overline{q}_{\text{vol}})}}{2a_l} \tag{A.17b}$$

$$\overline{q}_{r,\text{vol}} = \frac{-b_r + \sqrt{b_r^2 - 4a_r(c_r - \overline{q}_{\text{vol}})}}{2a_r} \tag{A.17c}$$

Slack variables:

$$\nu_{\text{u}} = \overline{r}_{\text{p}} - f_{\text{u}}(\overline{q}_{\text{vol}}) \tag{A.18a}$$

$$\nu_{\text{d}} = f_{\text{d}}(\overline{q}_{\text{vol}}) - \overline{r}_{\text{p}} \tag{A.18b}$$

$$\nu_{\text{l}} = \overline{q}_{l,\text{vol}} - \overline{r}_{\text{p}} \tag{A.18c}$$

$$\nu_{\text{r}} = \overline{q}_{\text{vol}} - \overline{r}_{r,\text{p}} \tag{A.18d}$$

$$\nu_b \leq 0, \qquad \forall b \in \{u, d, l, r\} \tag{A.18e}$$

Table A.5: Compressor model variables

Name	Type	Explanation
w_{in}	Input	Inlet flow
p_{in}	Input	Inlet pressure
w_{out}	Variable	Outlet flow
p_{out}	Variable	Outlet pressure
w_{sl}	Input	Flow through surge control line
r_{p}	Variable	Pressure ratio
$\overline{q}_{\mathrm{vol}}$	Variable	Volumetric flow at the map conditions
$\overline{r}_{\mathrm{p}}$	Variable	Pressure ratio at the map conditions
$f_{\mathrm{u}}(\overline{q}_{\mathrm{vol}})$	Functions	Quadratic functions approximating the map upper bound
$f_{\mathrm{d}}(\overline{q}_{\mathrm{vol}})$	Functions	Quadratic functions approximating the map lower bound
$f_{\mathrm{l}}(\overline{q}_{\mathrm{vol}})$	Functions	Quadratic functions approximating the map left bound
$f_{\mathrm{r}}(\overline{q}_{\mathrm{vol}})$	Functions	Quadratic functions approximating the map right bound
$\overline{q}_{l,\mathrm{vol}}$	Variable	Projection of the left bound in the flow axis at the operation point
$\overline{q}_{r,\mathrm{vol}}$	Variable	Projection of the right bound in the flow axis at the operation point

A.6: Production Network

Wells:

$$w_{\text{in}}^{w} = w_{\text{gl}}^{w} \qquad \forall w \in \mathcal{W} \tag{A.19a}$$

$$p_{\text{in}}^{w} = p_{\text{exp}}^{g} \qquad \forall w \in \mathcal{W} \tag{A.19b}$$

Manifold:

$$w_{\text{in}}^{p} = \sum_{w \in \mathcal{W}} \beta_{w,p} w_{\text{out}}^{w} \qquad \forall p \in \mathcal{P} \tag{A.20a}$$

$$p_{\text{out}}^{w} = \sum_{p \in \mathcal{P}} \beta_{w,p} p_{\text{in}}^{p} \qquad \forall w \in \mathcal{W} \tag{A.20b}$$

Separator

$$w_{\text{in}}^{s} = w_{\text{out}}^{p} \qquad \forall (s,p) \in \mathcal{S}_p \tag{A.21a}$$

$$p_{\text{in}}^{s} = p_{\text{out}}^{p} \qquad \forall (s,p) \in \mathcal{S}_p \tag{A.21b}$$

$$p_{\text{in}}^{s} = p^{s} \qquad \forall (s) \in \mathcal{S} \tag{A.21c}$$

Compressor:

$$w_{\text{in}}^{c} = w_{\text{out}}^{s} - w_{\text{fl}}^{c} \qquad \forall (c,s) \in \mathcal{C}_s \tag{A.22a}$$

$$p_{\text{in}}^{c} = p_{\text{out}}^{s} \qquad \forall (c,s) \in \mathcal{C}_s \tag{A.22b}$$

$$p_{\text{out}}^{c} = p_{\text{exp}} \qquad \forall c \in \mathcal{C} \tag{A.22c}$$

Export Line:

$$w_{\text{exp}}^{g} = \sum_{c \in \mathcal{C}} w_{\text{out}}^{c} - w_{\text{gm}} \tag{A.23a}$$

$$w_{\text{exp}}^{o} = \sum_{s \in \mathcal{S}} w_{\text{o,out}}^{s,\text{o}} \tag{A.23b}$$

$$w_{\text{exp}}^{w} = \sum_{s \in \mathcal{S}} w_{\text{w,out}}^{s,\text{w}} \tag{A.23c}$$

Lift gas manifold:

$$p_{\text{gm}} = p_{\text{exp}} \tag{A.24a}$$

$$w_{\text{gm}} = \sum_{w \in \mathcal{W}} w_{\text{gl}}^{w} \tag{A.24b}$$

A.7: Objective Functions

A.7.1: Tracking Problem

$$\begin{aligned}\min \psi = \int_{t_0}^{t_f} &(z - z_{\text{ref}})^T Q_z (z - z_{\text{ref}}) + (y - y_{\text{ref}})^T Q_y (y - y_{\text{ref}}) \\ &+ (u - u_{\text{ref}})^T R_u (u - u_{\text{ref}})\ dt\end{aligned} \tag{A.25}$$

A.7.2: Maximization Problem

$$\max \psi = \int_{t_0}^{t_f} k_g q^g_{\text{exp}} + k_o q^o_{\text{exp}} + k_w q^w_{\text{exp}} + k_{\text{fl}} w_{\text{fl}} + k_{\text{gl}} w_{\text{gm}}\ dt \tag{A.26}$$

A.8: Constraints

1. All masses should be non-negative,
2. All flows should be non-negative,
3. Injection should be smaller than the maximum determined in the well ($w^w_{\text{gl}} \leq w^{w,\max}_{\text{gl}}, \quad \forall w \in \mathcal{W}$),
4. The compressor map violation variables must be non-positive.

Appendix B : Modelica Examples

B.1: Example 1: Van der Pol Oscillator

To show how simple it is to describe a model using Modelica language in this subsection we will model the Van der Pol oscillator.

The Van der Pol oscillator is a non-linear system composed by two states and one input. The state equations are:

$$\dot{x}_1 = x_2 \tag{B.1a}$$

$$\dot{x}_2 = (1 - x_1^2)x_2 - x_1 + u \tag{B.1b}$$

$$y = x_1 \tag{B.1c}$$

where x_1 and x_2 are the states, u is the system input, and y is the system output. Further, $x_{1,0}$ and $x_{2,0}$ are the system initial conditions. Transcribing it to Modelica we have:

```
1  model VanDerPol
2      parameter Real x1_0 = 0; // x_1 initial cond
3      parameter Real x2_0 = 1; // x_2 initial cond
4
5      Real x1(start = x1_0);  // x_1 state
6      Real x2(start = x2_0);  // x_2 state
7
8      input Real u;           // control signal
9      output Real y;          // system output
10 equation
11     der(x1) = x2;
12     der(x2) = (1 - x1^2) * x2 - x1 + u;
13     x1 = y;
14 end VanDerPol;
```

In line 1, we initiate a component declaration saying the type of element it is followed by its name. Line 2 and 3, the initial conditions are declared, `parameter` means that this variable is not time dependent and `Real` means that it is in real domain. Line 5 and 6, states are declared and using `start` we indicate the initial conditions. Line 8 and 9, system input and output are declared using the respective keywords. Line 10

indicates that the equating starts. Line 11 and 12, `der` indicates that we are defining the derivative of x_1 and x_2 Line 13, we define the that the output of the system is x_1, note that as Modelica describe mathematically the model $y = x_1$ and $x_1 = y$ are the same.

B.2: Example 2: The Quadruple-Tank Process

Other very recurrent model for control is the four tank process. It have four cylindrical tanks, two pumps, and two valves. Tanks 3 and 4 leak in tanks 1 and 2, respectively. The pump 1 sends the flow to tank 1 and 4, whilst the pump 2 sends the flow to tank 2 and 3. The configuration is shown in B.1.

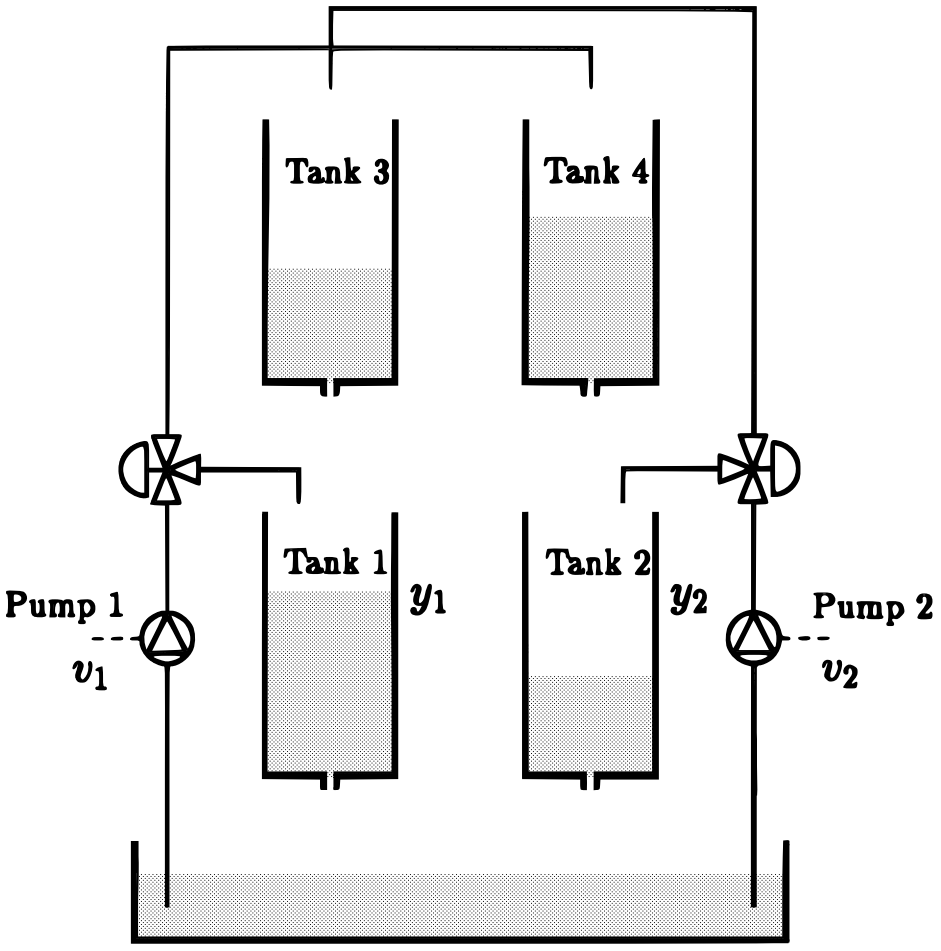

Figure B.1: Quadruple-tank process

The system dynamics are:

$$\frac{dh_1}{dt} = -\frac{a_1}{A_1}\sqrt{2gh_1} + \frac{a_3}{A_1}\sqrt{2gh_3} + \frac{\gamma_1 k_1}{A_1}v_1 \tag{B.2a}$$

$$\frac{dh_2}{dt} = -\frac{a_2}{A_2}\sqrt{2gh_2} + \frac{a_4}{A_2}\sqrt{2gh_4} + \frac{\gamma_2 k_2}{A_2}v_2 \tag{B.2b}$$

$$\frac{dh_3}{dt} = -\frac{a_3}{A_3}\sqrt{2gh_3} + \frac{(1-\gamma_2)k_2}{A_3}v_2 \tag{B.2c}$$

$$\frac{dh_4}{dt} = -\frac{a_4}{A_4}\sqrt{2gh_4} + \frac{(1-\gamma_1)k_1}{A_1}v_1 \tag{B.2d}$$

We can see that this problem is composed by four identical subsystems. To make the coding simpler we will exploit this. Each tank have one state, one inlet flow, and one outlet flow.

Listing B.1: Tank Modelica model

```
1  model Tank
2      import SI = Modelica.SIUnits;
3      import g = Modelica.Constants.g_n;
4
5      parameter SI.Area A = 4.9e-4;
6      parameter SI.Area a = 0.03e-4;
7
8      SI.Height h(start=0.05);
9      input SI.VolumeFlowRate q_in;
10     output SI.VolumeFlowRate q_out;
11 equation
12     A*der(h) = q_in - q_out;
13     q_out = a*sqrt(2*g*h);
14 end Tank;
```

Line 1 starts the model declaration. Line 2 imports the SI units library from Modelica standard library. Line 3 imports the gravitational acceleration from Modelica library. Line 5 defines the tank area, which is a parameter for the model. It could be used `Real` as the example before, however declare the unit of variables and parameters is a good practice in Modelica. Line 6, declaration of the tank orifice area. Line 8, define the state h with initial value of 5 cm. Line 9, the inlet flow is the tank input. Line 10, the outlet flow is tank output. Line 11, we integrate the volume in the tank using the height and the inlet and outlet flow ($A\dot{h} = q_{in} - q_{out}$). Line 12, define the outlet flow.

Now to compose the system we will create four tanks and add two pumps.

Listing B.2: Four Tank Modelica model

```
1  model fourTanks
2      import SI = Modelica.SIunits;
3
4      Tank t1, t2, t3, t4;
5
6      parameter Real k1 = 0.56e-6;
7      parameter Real k2 =  0.56e-6;
8      parameter Real gama1 = 0.30;
9      parameter Real gama2 = 0.50;
10
11     input SI.Voltage v1, v2;
12     output SI.Height y1, y2;
13 equation
14     t1.q_in = t3.q_out + k1*gama1*v1;
15     t2.q_in = t4.q_out + k2*gama2*v2;
16     t3.q_in = k1*(1-gama1)*v1;
17     t4.q_in = k2*(1-gama2)*v2;
18     y1 = t1.h;
19     y2 = t2.h;
20 end fourTanks;
```

Line 4, as Tank was defined we can use it to instantiate objects. Lines 6-9 define the pump parameters. Lines 14-17 define the input of each system. Lines 18-19 indicates the system output.

Using this structure we can ease change the tank model, use a conic tank for instance. The routing of tanks or even the number of tanks can be changed without any effort.

B.3: Example 1: Minimum time problem with Van der Pol Oscillator

Using the model stated in B.1 we are going to develop a optimization problem that the objective is to reach a specific condition in a less time as possible subject to constraints.

Consider that we want to bring to oscillator to origin as fast as possible, but having the control between the bounds $-1 \leq u \leq 1$. For this we state the optimization

problem

$$\min t_f \tag{B.3a}$$
$$\text{s.t.: } \dot{x} = f(x,u) \tag{B.3b}$$
$$x_1(t_f) = 0 \tag{B.3c}$$
$$x_2(t_f) = 0 \tag{B.3d}$$
$$-1 \leq u \leq 1 \tag{B.3e}$$

where x is the Van der Pol oscillator states, u is the control input, $f(x,u)$ is the state function, and t_f is the final time. This system can be transcribed to Optimica as:

Listing B.3: Tank Modelica model

```
1 optimization VDP_Opt(objective = finalTime,
2                           startTime = 0,
3                           finalTime(free=true,min=0.2,initialGuess=1))
4
5    extends VanDerPol;
6 constraint
7   x1(finalTime)=0;
8   x2(finalTime)=0;
9   u >= -1;
10  u <= 1;
11 end VDP_Opt;
```

At line 1, the optimization class indicates that this is a optimization problem, the objective function is the finalTime, simulation starts at 0, and final time is an optimization (free) variable with the minimum value of 0.2 and the initial guess of 1. Line 5 say that this problem will use the definitions of the Van der Pol model. Line 6 indicates the begin of the declaration of constraints. Lines 7-10 implement the system constraints.