NTNU
Det skapande universitetet

Kontroll av oljeutslipp frå havbotn

Control of Oil Spill at Sea Bed

Eivind Vrålstad

Master i energi og miljø
Oppgåva levert: Juli 2011
Hovudrettleiar: Tor Ytrehus, EPT

Noregs teknisk-naturvitskapelege universitet

NTNU
Det skapende universitet

MASTERKONTRAKT
- uttak av masteroppgave

1. Studentens personalia

Etternavn, fornavn	Fødselsdato
Vrålstad, Eivind	**22. jul 1984**
E-post **eivind.vralstad@gmail.com**	Telefon **41446991**

2. Studieopplysninger

Fakultet **Fakultet for Ingeniørvitenskap og teknologi**	
Institutt **Institutt for energi- og prosessteknikk**	
Studieprogram **Master i energi og miljø**	Studieretning **Varme- og energiprosesser**

3. Masteroppgave

Oppstartsdato **01. mar 2011**	Innleveringsfrist **26. jul 2011**
Oppgavens (foreløpige) tittel **Kontroll av oljeutslipp fra havbunn**	
Oppgavetekst/Problembeskrivelse Bakgrunn Ved massive oljeutslipp på havbunnen er det av avgjørende betydning å hindre at oljen får strømme fritt ut i omgivelsene og blande seg i vannmassene, slik at den spres over store områder med havstrømmer, bølger og vind, og fører til store miljøproblemer langs kyster og strender, samt i gyte- og oppvekstområder for fisk og sjødyr. Oppsamling av slike utslipp etter at oljen har nådd opp til overflaten viser seg alltid å være problematisk; særlig i områder som er utsatt for mye vær og vind og høye bølger. Med bakgrunn i den store utblåsningen ifm. havariet av BP sin oljeplattform "Deepwater Horizon" i Mexicogulfen sommeren 2010, ble det av veileder til denne oppgaven (T.Y.) skissert en hydrodynamisk limning som i prinsippet samler utslippet på havbunnen og transporterer oljen gjennom et stigerør til et oppsamlingsfartøy på overflaten. Denne løsningen krever en del utstyr som må foreligge på forhånd i ren beredskapsøyemed, og i denne oppgaven blir det fokusert på de strømningsmekaniske forholdene som må etableres og kontrolleres i grensesnittet mellom lekkasjepunkt og stigerør mot overflaten. Oppgaven bearbeides ut fra følgende punkter 1. En kort gjennomgang av dag...	
Hovedveileder ved institutt **Professor Tor Ytrehus**	Medveileder(e) ved institutt
Merknader **1 uke ekstra p.g.a påske.**	

4. Underskrift

Student: Jeg erklærer herved at jeg har satt meg inn i gjeldende bestemmelser for mastergradsstudiet og at jeg oppfyller kravene for adgang til å påbegynne oppgaven, herunder eventuelle praksiskrav.

Partene er gjort kjent med avtalens vilkår, samt kapitlene i studiehåndboken om generelle regler og aktuell studieplan for masterstudiet.

.....
Sted og dato

.....	.....
Student	**Hovedveileder**

Originalen oppbevares på fakultetet. Kopi av avtalen sendes til instituttet og studenten.

Control of Oil Spill at Sea Bed

Stud.techn. Eivind Vrålstad

July 26, 2011

Summary

Inspired by the 2010 «Deepwater Horizon»/Macondo well oil spill in the Gulf of Mexico, a new system of containment and collection directly from a leaking wellhead at large depths is proposed and modeled. This system consists of a vertical riser, lowered from a vessel at the surface onto the leak spot, with a capture dome on the lower end. The system is made leak-free by imposing a lower pressure on the inside of this dome than the hydrostatic sea floor pressure, sealing the connection between the capture dome and the sea floor hydrodynamically.

This model, which also includes a crude model of the entire well, is subject to computer simulations, and is discretized using the method of characteristics. All simulations are 1D, one-phase. Emphasis is put on the start-up of the system. MATLAB is used as platform for the computer simulations, and all simulations are done on a 64-bit laptop.

It turns out the trickiest part of start-up is the period between the flow in the riser is stagnant until pressure in the capture dome sinks below wellhead pressure, merging the two subsystems. A solution to this seems to be a slow, small, nonlinear lowering in top riser pressure in two steps, first until sealing and merging is achieved. After this, the system is easier to handle, and can be set to any end pressure required to set the capture dome entirely into the sea floor.

Samandrag

Med inspirasjon frå lekkasja etter «Deepwater Horizon»/Macondobrønnulukka i 2010, er det foreslått og modellert eit nytt system for innfanging og oppsamling av olje rett frå brønnhovudet ved lekkasje på store havdjup. Dette systemet består av eit stigerøyr som er senka frå eit farty ned på lekkasjestaden, med ein oppsamlingskuppel i nedre enden. Systemet blir tetta ved å påføre eit lågare indre trykk i denne kuppelen enn det hydrostatiske havbotntrykket, slik at koplinga mellom kuppelen og havbotnen blir hydrodynamisk foresgla.

Denne modellen, som også innheld ein grov modell for heile brønnen, gjennomgår datasimuleringar, kor han er diskretisert med karakteristikkmetoden. Alle simuleringane er i 1D, einfase. Det er lagt hovudvekt på oppstarten av systemet. Som plattform for datasimuleringane er MATLAB nytta, og alle simuleringar er gjort på ei 64-bits bærbar datamaskin.

Det viser seg at den vanskelegaste delen av oppstarten er tida frå strøyminga i stigerøyret er stillestående, til trykket i oppsamlingskuppelen synk under brønnhovudtrykket, slik at dei to systema blir eitt. Ei løysing på dette ser ut til å vere ein sakte, liten, ikkjelineær senking av stigerøyrets topptrykk i to steg, først til forsegling og samanslåing er oppnådd. Etter dette blir systemet meir medgjørleg, og kan enkelt påførast kva sluttrykk ein treng for å få oppsamlingskuppelen til å setje seg i havbotnen.

Preface/Forord

Først vil eg gjerne få sei at innspurten til denne oppgåva etter hendingane i Oslo og på Utøya 22. juli har vore vanskeleg. Måtte Noreg aldri oppleve slik ondskap att.

Når dette er sagt, er det mange eg gjerne vil takke frå heile mi ferd gjennom studentlivet. Eg vil takke Trondheim for å ha vore min studieby i sju år. Eg vil takke EMIL og Smørekoppen Revy, Studentersamfundets Interne Teater og UKA for å ha vist meg kven eg eigentleg er. Og sjølvsagt E. C. Dahls Bryggeri.

Eg vil vidare takke Tor Ytrehus for rettleiinga med oppgåva, og at eg fekk lov til å byrje på ho så langt utanfor vanleg semester.

Til slutt vil eg takke IA for å ha halde ut med meg i hus dei siste vekene.

Trondheim, 26. juli 2011
Eivind Vrålstad

Contents

Summary I
Samandrag III
Preface/Forord V
Contents VIII
Nomenclature IX
List of Figures XI

1 Introduction **1**

2 Background **3**
2.1 Deepwater Horizon, the Macondo Well 3
2.2 Current Technologies of Oil Spill Containment 4
2.2.1 Shovels and Buckets 4
2.2.2 Booms 5
2.3 Seafloor Efforts at the Macondo Well 5
2.3.1 The Containment Dome and Top Kill 6
2.3.2 The Top Hat Procedures 6

3 Description of the Proposed Collection System **9**
3.1 In Detail 9

4 Fluid Dynamics **13**
4.1 Equation of Motion 13
4.2 Continuity Equation 15
4.3 Speed of Sound and Bulk Modulus of Elasticity 17
4.4 The Friction Factor 19
4.5 The Well 19
4.6 The Capture Dome 20
4.7 The Top Valve 20
4.8 Thermodynamic Properties of the Fluids 20
4.9 Other Properties 21

5 The Method of Characteristics **23**
5.1 The + and - Characteristics 23
5.2 Steady-State Solution 27
5.3 Interpolations 28
5.4 The Numerical Method 30
5.5 Stability 31
5.6 Interpolation Errors 32

6 The MATLAB Programs **35**
6.1 `storeinitdata.m` . . . 35
6.2 `blowout.m` . . . 35
6.3 `steadystate.m` . . . 35
6.4 `transient.m` . . . 36
6.5 The Grid . . . 36

7 Results **39**
7.1 Blowout . . . 39
7.2 Steady-State . . . 41
7.3 Variability of The Speed of Sound and Other Secondary Variables 43
7.3.1 The Speed of Sound, a . . . 44
7.3.2 The Real Area, A . . . 45
7.3.3 The Density, ρ . . . 45
7.4 Transient Start-Up . . . 46
7.4.1 Single Linear Opening of Top Valve . . . 46
7.4.2 Velocity Peaks and Pressure Change . . . 48
7.4.3 Effect of Grid Variation . . . 51
7.4.4 Double Linear Opening of Top Valve . . . 53
7.4.5 Nonlinear Opening of Top Valve . . . 57
7.4.6 Nonlinear Opening – Fine Tuning of the 1^{st} Opening . . . 60

8 Conclusion **63**
8.0.7 Further Work . . . 63

References **65**

A MATLAB Code **67**
A.1 storeinitdata.m . . . 67
A.2 blowout.m . . . 69
A.3 steadystate.m . . . 75
A.3.1 plotSteadyState.m . . . 81
A.4 transient.m . . . 85
A.4.1 plotTransient.m . . . 95
A.4.2 rho_p_init.m . . . 99

B Full Background (Norwegian) **101**

Nomenclature

A	Area of pipe
a	Speed of sound
D	Pipe diameter
E	Young's modulus of elasticity
e	Pipe wall thickness
f	Darcy-Weisbach friction factor
g	Gravitational acceleration [9.81 m/s^2]
H	Vertical position below sea level
K	Bulk modulus of elasticity
L	Pipe length
m	Mass
PI	Productivity index
p	Pressure
Q	Volumetric flow rate
Q_0	Volumetric flow rate at standard conditions (1 atm, 300 K)
Re	Reynolds number
SG	Specific gravitation, SG $= \frac{\rho_{\text{oil}}}{\rho_{\text{water}}}$
T	Temperature
t	Time
V	Velocity
α	Pipe slope
θ	Grid mesh ratio
λ	Multiplier in the characteristics method
μ	Dynamic viscosity
ν	Kinematic viscosity
ρ	Mass density
τ	Shear stress

List of Figures

2.1 The «Deepwater Horizon» drilling rig. 3
2.2 «Full City» cleanup. 5
2.3 "Top hat #4". 6
3.1 System sketch. 10
4.1 Free-body diagram for application of equation of motion. 13
4.2 Contnuity. 15
5.1 Characteristic lines. 25
5.2 Interpolations. 28
5.3 Stability 31
5.4 Interpolation error, xt plane 33
7.1 Steady-state pressures at various volumetric flux and well diameter. 39
7.2 Steady-state velocities at various volumetric flux and well diameter. 40
7.3 Steady-state results of a reduction in p_{top} through the system, relative to the blowout scenarios. 43
7.4 Variation of a, A and ρ through a steady-state system. Series are divided by their maximum values. 44
7.5 Pressures at transient start-up with linear reduction in p_{top}. 48
7.6 Velocities at transient start-up with linear reduction in p_{top}. 49
7.7 Surface plot of V for the 10 s, 15 bar transient start-up with linear reduction in p_{top}. 50
7.8 Pressures at transient start-up with linear reduction in p_{top}, comparison between different M values. 51
7.9 Velocities at transient start-up with linear reduction in p_{top}, comparison between different M values. 52
7.10 Pressures at transient start-up with double linear reduction in p_{top}. 53
7.11 Velocities at transient start-up with double linear reduction in p_{top}. 56
7.12 Velocities at transient start-up with double linear reduction in p_{top}. 57
7.13 Pressures at transient start-up with double nonlinear reduction in p_{top}. 58
7.14 Velocities at transient start-up with double nonlinear reduction in p_{top}. 59
7.15 Velocities at transient start-up with double nonlinear reduction in p_{top}. 60
7.16 Pressures when adjusting the first nonlinear reduction in p_{top}. . . . 61
7.17 Velocities when adjusting the first nonlinear reduction in p_{top}. . . . 62

1 Introduction

The goal of this master thesis is to simulate a system to collect oil directly from blown-out wells at large depths. The inspiration comes from the 2010 «Deepwater Horizon» spill. The design is simple; only consisting of a vertical riser lowered from a vessel at the surface. The system is not intended as a stopper to the well leakage itself, but as a temporary measure to contain and limit the oil spill, until the blowout can be stopped.

Hopefully, putting a collection device like this directly over of a leaking wellhead will prove much more efficient than existing methods of containment and dispersion, and perhaps also be much cheaper.

The initially oil-filled vertical riser that is lowered onto the point of leakage from a vessel. Here, a capture dome on the end of the riser totally encapsulates the leaking wellhead, and then utilizes the pressure difference created by unequal density in the sea water and the oil to make a hydrodynamic seal between the capture dome and the sea floor.

The focus in this report is mainly to simulate the start-up process and confirm that the system actually can work. This includes making a simple model of the well, and also finding an optimal diameter for the riser, based on steady-state simulations.

Finally, a transient model is developed to simulate the start-up of the system. This is then fine-tuned in an attempt to find an optimal start-up procedure.

A numerical model based on the method of characteristics is applied in both the steady-state and the transient simulations.

It is not the intent of this thesis to answer questions regarding the hydrodynamic seal between the capture dome and the sea floor.

2 Background

The background for this master thesis is the 2010 Macondo well blowout in the Gulf of Mexico. This blowout showed the inadequacies of todays oil spill management at large depths. Vast amounts of raw oil spewed out for almost three months before the spill was stopped.

To have an adequate response available will become essential in the years to come, as the depths increase, also in the Norwegian sector. The Macondo well was situated at $\sim 1,500$ m, and many new Norwegian prospects, like the Barents Sea and the seas outside Lofoten, all have depths of this order. These areas are also extremely vulnerable to oil spills, as the ecosystems here are essential in Norwegian fishery.

Figure 2.1: The «Deepwater Horizon» drilling rig. Picture: Wikimedia Commons.

Luckily, the Norwegian sector has still not seen an oil spill of this order. But there is no guarantee of this not happening, so to give this a Norwegian spin, the cleanup efforts from the much smaller «Full City» spill in Southern Norway in 2009 have been used as comparison, and a basis for the walk-through of elements of spill treatment not directly concerning the «Deepwater Horizon» specifics. Even though the orders of magnitude are different, many key elements in clean-up are the same, albeit on a different scale.

2.1 Deepwater Horizon, the Macondo Well

On April 20, 2010, the *Macondo* well in the Gulf of Mexico blew out, sinking the «Deepwater Horizon» drilling rig. According to the National Commission on the BP Deepwater Horizon Oil Spill and Offshore Drilling,[1] the disaster cost 11 crew members' lives, and spilled a total of $4.9 \cdot 10^6$ bbl* ($\simeq .8 \cdot 10^6$ m^3) of crude oil into the ocean. The spill caused severe pollution along some of the most vulnerable coastline in the USA, which in turn had catastrophic effects on wildlife, fishery and tourism.

By comparison, the «Full City» accident resulted in a spill of only 294 m^3, but still this killed between 2,000 and 2,500 seabirds, and seriously polluted a large stretch of the Norwegian coastline. It was also very costly; with a final cleanup cost of over NOK 230 million.[2] Or almost NOK 1 million per spilled cubic meter.

*Barrels [bbl] are standard volume measuring units in the oil industry. 1 bbl $\approx .159$ m^3

The «Deepwater Horizon»/Macondo oil spill revealed the weaknesses of todays technologies of oil spill containment. Out of the $4.9 \cdot 10^6$ bbl spilled, about 17 % were directly recovered from the wellhead (see Sec. 2.3.2 for details), 5 % were burned, 3 % were skimmed, and 8 % were chemically dispersed. Altogether, only 33 % of the total spill was handled, from it was discovered on April 22 until it was finally capped on July 15 2011. By then, the spill had become the largest in the history of USA.

2.2 Current Technologies of Oil Spill Containment

The current technologies of oil spill containment and management can be divided into two categories, either those that collect and store the oil, and those that burn or in other ways disperse or dissolve the oil. The first of these is by far the most environmentally sound one, as burning and dispersion both tend to create new problems, like coral death, high CO_2 emissions, etc.

Generally, there are two phases of an oil spill. First, there is the acute phase, where oil has just leaked, or is still leaking. In this phase, containment of the spill and efforts to stop it are central. In the second phase (cleanup), however, the spill is stopped, and focus shifts to cleaning up what is already contaminated.

In the event of a broken well, like the «Deepwater Horizon» accident, the first goal must be to close the well entirely. Until then, however, something must be done to prevent the oil from spreading. A number of methods of cleanup and containment already exist, some of which will be explained further in the section to come.

2.2.1 Shovels and Buckets

The title might not be entirely covering, but still, a large quantity of any oil spill that reaches land has to be physically removed from the shoreline. This is very labor-intensive, and during the 2009 «Full City» spill, hundreds of volunteers only removed 72 m^3 of oil in this way. This is however a necessary action when the oil first reaches shore, as oil that reaches shore degrades very slowly.

This method of turning every stone to find the oil does not help limiting the amount of oil spread from a leak, and it is only practical to initiate this first after the spill itself is stopped. After all, to use many man-hours to remove oil, only for the oil to return the next day is not the most wise use of resources in this situation.

Fig. (2.2) shows a beach in Krogshavn, near Langesund in Telemark, heavily polluted by the «Full City» spill. The top picture shows the beach during cleanup, with a number of people removing a thick layer of heavy oil from the beach, while the bottom picture shows how the beach looked like after cleanup.

Of course, many other tools than buckets and shovels are used, such as skimmers and various suction devices and other mechanized tools. Still, turning every stone in a shoreline is a process that takes many man-hours, and has not yet been mechanized. Until that happens, a cleanup depends on the help of numerous volunteers and professionals to get a good result.

Figure 2.2: A beach during and after clean-up of the «Full City» spill. Picture: Kystverket.

2.2.2 Booms

Booms floating barriers used to contain or direct the oils lighter than water to control a spill. They are used in several ways, both as a mean of containment and collection, but also to prevent surface oils from reaching specific areas. These include natural reserves and other especially vulnerable areas.

The booms are most effective during calm weather, with little or no waves. This means that they are often ineffective in weather when accidents tend to happen. So, if an oil tanker goes down in a storm, little can be done to prevent an oil spill with booms until the weather clears.

In the case of a ship going ashore, these booms can be quite effective, if they encircle the vessel and are combined with a skimmer (a tool which separates surface oils from water) to collect the surface oil. There is very little a boom can do with the heavier oils, so these mostly sink to the bottom.

In the case of a blowout at sea bed, the booms will also be of limited help, especially at depths as large as with the Macondo blowout. By the time it has reached the surface, it has had a long time to mix with the sea water and become a large, semi-concentrated plume of oil. The area at which the oil comes to the surface will also be shifting, so that there is very little help in booms other than to seal off special areas in a blowout scenario or to divert or direct the spill.

2.3 Seafloor Efforts at the Macondo Well

The fact that the Macondo well blowout lasted for so long, made it a testing ground for a number different techniques to contain oil directly from the blowout spot. Many of these failed imminently, while others were more effective, and eventually helped stopping the well flow. The procedures here are mostly cited from the descriptions given in the end report by the commission appointed to investigate the accident.[1]

The very first attempt was to stop the flow using ROVs (remote-controlled submarines) to release the emergency valve on the *blowout preventer*[†] (BOP), but these efforts failed when it turned out it had been severely damaged in the accident, stopping the closing valve half-way.

2.3.1 The Containment Dome and Top Kill

When it turned out the BOP would not close, the first effort in containing the oil was to put a bulky containment dome on the leak, encapsulating the whole BOP. This, however, was stopped very early by the formation of gas hydrates caused by the mixing of gas leaking from the wellhead and cold water leaking in from the sea floor.

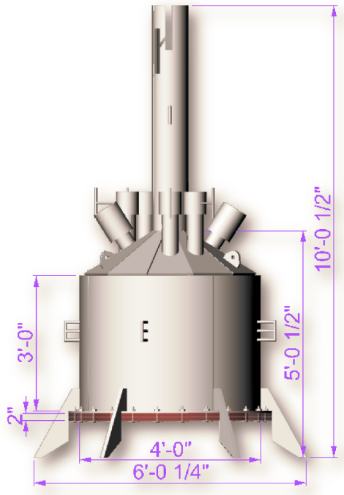

Figure 2.3: "Top hat #4".
Picture: US Dept. of Energy.

When this failed, they turned to another procedure, "Top Kill". This is a method consisting of pumping heavy drilling mud into the well, heavy enough to sink to the bottom of the well and plug it. This is a quite normal procedure that has worked many times before, but it failed to stop the Macondo well.

2.3.2 The Top Hat Procedures

The top hat procedure consisted of a dome much smaller than the containment dome, that was designed to fit atop the uppermost intact flange of the BOP. It was lowered from a vessel at sea level, and the first test design (Top Hat #4) proved effective enough to collect almost all of the gas, and up to 15,000 bbl/day (0.028 m^3/s) of oil. To prevent formation of hydrates, the top hat was also fitted with a methanol injector.

When this procedure was operational, another vessel connected to the BOP itself, and collected and additional 10,000 bbl/day (0.018 m^3/s) of oil. By then, the well was blowing out at a rate of 68,000 bbl/day (0.12 m^3/s), according to Crone (2010).[3] The estimates made by the well operator British Petroleum, however, were much smaller. So, the top hat was under-dimensioned, and not even the combined effort of both systems was nearly enough to collect all blown-out oil.

[†]A blowout preventer is a device that sits on top of the wellhead to prevent a blowout.

Even the top hat was insufficient in collecting the oil from the well, but it worked to some degree. Eventually, it was removed and replaced by a much more heavy-duty system, bolted onto the BOP, replacing its top cap. This system was eventually able to assist the capping of the well.

But even though it did not remove more than nearly 1/3 of the oil from the well when in full operation, the top kill procedure was a semi-success. Had it been large enough to swallow the full extent of the blowout, it might actually have saved the Gulf of Mexico and the coast of the USA from a large portion of the spill.

3 Description of the Proposed Collection System

Looking at the summary of the «Deepwater Horizon»/Macondo spill in the last section, it seems as though there is a need for a new method of containment and collection of oil directly from the point of leakage. It should be designed for large depths, and be large enough to handle even very large spills. To ensure a quick response, it should also have a universal shape and method of connection, so it does not depend largely on what kind of equipment is installed on the sea floor.

Luckily, the leaking oil that emerges on the surface has one key feature to make it easily collected from the sea floor – it is lighter than the sea water. While sea water weighs around $1,000$ kg/m^3, a normal crude oil only weighs about $800 - 900$ kg/m^3. Therefore, if a vertical riser was lowered onto the point of leakage, and the seal between its end and the well was made tight, the flow of oil would just continue to the surface.

In fact, the top pressure would be higher than atmospheric because of the difference in densities. In a frictionless flow, this would account to about 10-20 bar on a 1,000 m riser with the already given densities for oil and sea water.

However, this case requires the bottom end to be sealed tightly, so that the well and riser become one system. The solution to this is a proposed hydrodynamic seal between the riser end bit and the sea floor itself. This is possible, since a release in pressure at the top will make the riser end stick to the sea floor.

Some water will probably enter the system as a result of this, causing two problems, namely higher density and hydrate formation. The latter can be solved by injecting methanol, like in the "top hat" procedure. The first can hopefully be solved through optimizing the "capture dome" design on the lower end of the riser. This is however not the object of this report, and instead it is only assumed that a vacuum pressure of about 10 bar at the connection will make the seal tight.

3.1 In Detail

As previously stated, the proposed system consists of a vertical riser that is lowered on top of the leaking well head, to collect the oil using a tight hydrodynamic seal between the sea floor and itself. The different parts of the system, including a simplified simulation of the well, is shown in Fig. (3.1).

The key in this system is the end bit, or the capture dome. This will have to be equipped with valves that allow oil to flow quite uninterrupted into the ocean while not in operation. Then, as start-up commences, these will close to allow the pressure inside the capture dome to sink below the hydrostatic sea floor pressure and squeeze the capture dome into the sea floor. This design will hopefully make it relatively simple to operate and install, even at 1500 m below sea level, as in «Deepwater Horizon»/Macondo case.

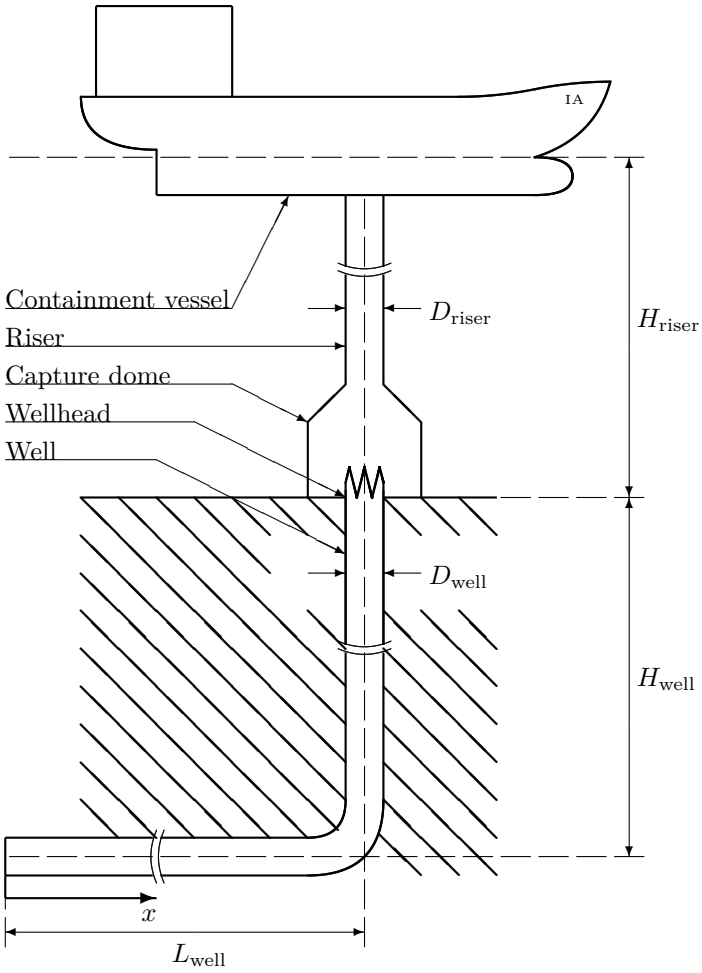

Figure 3.1: System sketch.

To be able to simulate this start-up, a simplified model of the well is also needed. This is further described in Sec. 4.5. This will make the well flow rate rise as the pressure at the wellhead is lowered.

Also, one or more production vessels capable of handling the amount of oil blown out will have to be available on the surface. It is only assumed that this capacity is sufficient for any flow rate.

The key to control the system sits mainly in one component, namely a valve on top of the riser. It is assumed that this valve is able to make pressure drop along any curve, independent of any actual valve characteristics.

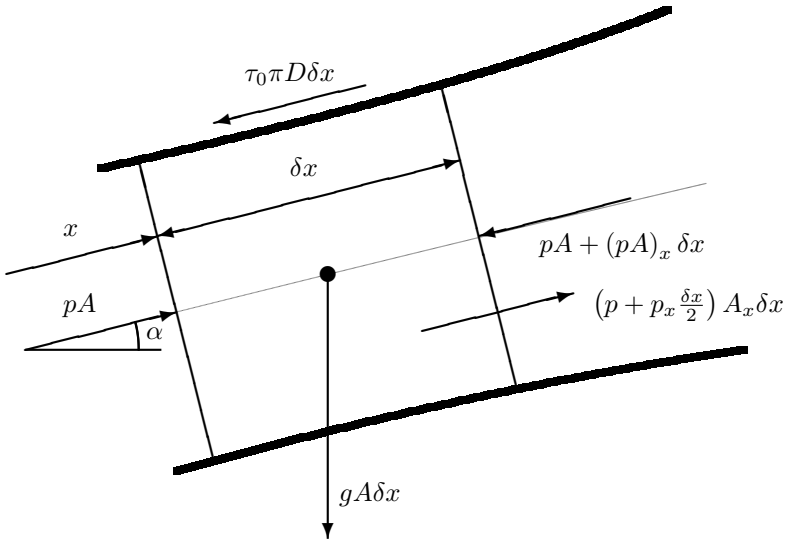

Figure 4.1: Free-body diagram for application of equation of motion.

4 Fluid Dynamics

4.1 Equation of Motion

To calculate any flow an equation of motion, or a momentum equation is needed. This is the sum of all forces that act upon the fluid. An arbitrary control volume of length δx, with a cross section A and a pressure p, closed off by the pipe wall and the cross sections is shown in Fig. (4.1), together with all forces that act upon the control volume. There is normal pressure on the cross sections, pA and $pA + (pA)_x \, \delta x$ and peripheral pressure, $\left(p + p_x \frac{\delta x}{2}\right) A_x \delta x$. The gravity has an x component, $\rho g A \sin \alpha$, and finally, the shear stress τ_0 that acts in negative x direction. Abbreviated notations are used for simplicity, e.g. $f_x = \frac{\partial f}{\partial x}$ and $\dot{V} = V V_x + V_t$. Summarizing these elements, yields

$$pA - [pA + (pA)_x \, \delta x] + \left(p + p_x \frac{\delta x}{2}\right) A_x \delta x \\ -\tau_0 \pi D \delta x - \rho g A \delta x \sin \alpha + \rho A \dot{V} = \rho A \delta x \dot{V}$$

δx can be assumed as small, so $(\delta x)^2 \ll 1$,

$$p_x A + \tau_0 \pi D + \rho g A \sin\alpha + \rho A \dot{V} = 0 \tag{4.1}$$

The shear stress τ_0 is assumed to behave as in a steady, fully developed flow even though it is transient. Further, the flow regime must be determined, by applying some coarse approximations of the order of magnitude of various flow and pipe elements to evaluate the Reynolds number,

$$Re_D = \frac{\rho V D}{\mu} \tag{4.2}$$

Approximate values of $\rho \sim 1000$ kg/m^3, $V \sim 1$ m/s, $D \sim 10^{-1}$ m, and $\mu \sim 10^{-5}$ kg/m $\cdot$ s, produce $Re_D \sim 10^7$, which means the flow is fully turbulent. The high relation between length and diameter also implies that the flow is fully developed.

The shear stress τ_0 can now be evaluated, using the $Darcy-Weisbach$ friction factor, f. The Darcy-Weisbach relation is given by[4]

$$\Delta p = \rho f \frac{L}{d} \frac{V^2}{2} \tag{4.3}$$

where L is the length of the pipe, and D is the diameter. In a steady flow, the force from the pressure gradient on the cross section $\pi/4D^2$ will equal that of the shear stress on the wall πDL, making

$$\Delta p \frac{\pi}{4} D^2 = \tau_0 \pi D L \tag{4.4}$$

Combining Eqs. (4.3) and (4.4), yields a cleaner form of the Darcy-Weisbach equation,

$$\tau_0 = \frac{\rho f V |V|}{8} \tag{4.5}$$

in which the absolute value is used to ensure that friction always opposes the flow direction.

Substituting $\dot{V}$ with

$$\dot{V} = V V_x + V_t \tag{4.6}$$

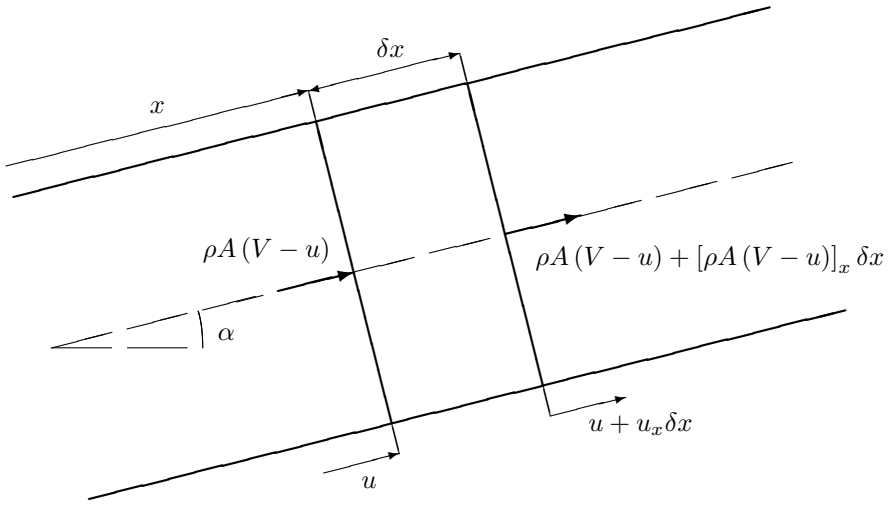

Figure 4.2: Contnuity.

and τ_0 with Eq. (4.5) in Eq. (4.1), yields

$$\boxed{\frac{p_x}{\rho} + VV_x + V_t + g\sin\alpha + \frac{fV\,|V|}{2D} = 0} \tag{4.7}$$

This rearrangement has removed the term A from the energy equation, but it is still valid for variable diameter D.

4.2 Continuity Equation

Continuity is achieved when the change in mass of a control volume equals the amount of mass going in minus the amount going out of it. This will be described this in total derivatives, with a moving control volume at a time t, fixed to the

pipe wall. This means the control volume will only move if the pipe moves, and stretch if the pipe stretches, as shown in Fig. (4.2). From this, the rate of increase of mass in the pipe section as mass out minus mass in, is

$$\frac{\mathrm{D}'}{\mathrm{D}t}\left(\rho A \delta x\right) = -\left[\rho A\left(V-u\right)\right]_x \delta x \tag{4.8}$$

Here, V is the velocity of the fluid and u is the axial velocity of the pipe. The upstream cross section is now at x, and the distance to the downstream one is δx. The total derivative for the axial movement of the pipe is,

$$\frac{\mathrm{D}'}{\mathrm{D}t} = u\frac{\partial}{\partial x} + \frac{\partial}{\partial t} \tag{4.9}$$

and the increase of δx with respect to time is,

$$\frac{\mathrm{D}'}{\mathrm{D}t}\delta x = u_x \delta x \tag{4.10}$$

Eq. (4.8) can now be expanded, using Eq. (4.10),

$$\frac{\mathrm{D}'}{\mathrm{D}t}\left(\rho A\right) + \rho A u_x + \left(\rho A V\right)_x - \left(\rho A u\right)_x = 0 \tag{4.11}$$

and expanded further, using Eq. (4.9),

$$u\left(\rho A\right)_x + \left(\rho A\right)_t + \rho A u_x + \left(\rho A\right)_x V + \rho A V_x - \left(\rho A\right)_x u - \rho A u_x = 0 \tag{4.12}$$

and finally contracted to get a short version of the continuity equation, without total derivatives and the pipe velocity u,

$$\left(\rho A\right)_t + V\left(\rho A\right)_x + \rho A V_x = 0 \tag{4.13}$$

or even shorter,

$$\left(\rho A\right)_t + \left(\rho A V\right)_x = 0 \tag{4.14}$$

The first two terms of Eq. (4.13) can be recognized as the total derivative of ρA with respect to V,

$$\frac{\mathrm{D}}{\mathrm{D}t}(\rho A) = V\frac{\partial}{\partial x}(\rho A) + \frac{\partial}{\partial t}(\rho A) \tag{4.15}$$

Which in turn means,

$$\frac{1}{\rho A}\frac{\mathrm{D}}{\mathrm{D}t}(\rho A) + V_x = 0 \tag{4.16}$$

Employing a dot over the dependent variable as shorthand for $\frac{\mathrm{D}}{\mathrm{D}t}$, and separate ρ and A, yields

$$\frac{1}{\rho A}\left(\dot{\rho}A + \rho\dot{A}\right) + V_x = 0 \tag{4.17}$$

or

$$\boxed{\frac{\dot{A}}{A} + \frac{\dot{\rho}}{\rho} + V_x = 0} \tag{4.18}$$

Which is the final form of the continuity equation with no simplifications made. This is valid for both pressure, density, and diameter changes in both time and length, for pipes of any incline rate.

4.3 Speed of Sound and Bulk Modulus of Elasticity

The speed of sound a is the pressure wave propagation speed in a fluid at constant entropy. This must be described, as the calculations include a pipeline that is too long for pressure change to be instantaneous throughout.

It can be found by defining a control volume around a pressure wave moving at velocity $V = a$. The energy must now be equal at either side of the wave, and we get the differences $\Delta\rho$ and ΔV over the wave (A is held constant), and are limited to potential energy from pressure and kinetic energy from velocity, so that

$$pA + \rho A a^2 = (p + \Delta p)A + \rho A a(a + \Delta V) \tag{4.19}$$

or

$$\Delta p = -\rho a \Delta V \tag{4.20}$$

If a valve is closed in the downstream end of the pipe with a fluid moving a velocity V, a pressure wave will start traveling in the upstream direction. This will lead to a change in pressure, which in turn will lead to a stretching of the pipe in the axial direction, Δs. The shock wave will then move the length of the pipe L in time of L/a, meaning the speed of the pipe stretching becomes $\Delta sa/L$. If the pipe is anchored in the upstream end, the velocity of the fluid in the pipe changes by

$$\Delta V = \Delta sa/L - V_0 \tag{4.21}$$

The mass will not stop moving into the pipe before the wave has reached the upstream end, and will continue moving into the pipe at a mass rate of $\rho A V_0 L/a$. This is possible to accommodate by filling the extra pipe volume generated from Δs, by an increase in pressure Δp, or by an increase in pipe cross-section ΔA,

$$\rho A V_0 \frac{L}{a} = \rho L \Delta A + \rho A \Delta s + L A \Delta p \tag{4.22}$$

Substituting ΔV, given by Eq. (4.21), into Eq. (4.22) yields,

$$-\frac{\Delta V}{a} = \frac{\Delta A}{A} + \frac{\Delta p}{\Delta \rho} \tag{4.23}$$

which can be combined with Eq. (4.20) to yield,

$$a = \sqrt{\frac{\Delta p/\rho}{\Delta A/A + \Delta \rho/\rho}} \tag{4.24}$$

This can be simplified by introducing the bulk modulus of elasticity of the fluid,

$$K = \frac{\Delta p}{\Delta \rho/\rho} \tag{4.25}$$

and the change in wall tensile stress, to get a value for ΔA

$$\frac{\Delta A}{A \Delta p} = \frac{D}{Ee} \tag{4.26}$$

in which D is the pipe diameter, E is the Young's modulus of elasticity for the pipe material, and e is the pipe thickness. Eq. (4.27) then becomes,

$$\boxed{a = \sqrt{\frac{K/\rho}{1 + (K/E)\,(D/e)}}} \tag{4.27}$$

which for a very rigid pipe becomes,

$$a = \sqrt{\frac{K}{\rho}} \tag{4.28}$$

4.4 The Friction Factor

The friction factor f in Eq. (4.7) can be found through the equation derived by Colebrook (1939),[4]

$$\frac{1}{f^{1/2}} = -2.0 \log\left(\frac{\epsilon/D}{3.7} + \frac{2.51}{Re_D f^{1/2}}\right) \tag{4.29}$$

in which the Reynolds number, $Re_D = \frac{\rho V D}{\mu}$. This is the basis for the Moody chart of turbulent friction factors, but the interpolation procedure needed to solve this equation makes it quite time-consuming to use in numerics. Therefore the approximation derived by Haaland (1999) is applied,

$$\frac{1}{f^{1/2}} \approx -1.8 \log\left[\frac{6.9}{Re_D} + \left(\frac{\epsilon/D}{3.7}\right)^{1.11}\right] \tag{4.30}$$

This is accurate to within ± 2 % of Eq. (4.29).[4]

4.5 The Well

All oil wells have a productivity index PI, which determines the connection between reservoir pressure p_r, well pressure p_w, and volumetric flow Q. PI is a constant for the well, and is given by

$$\mathrm{PI} = \frac{Q}{p_{\mathrm{res}} - p_{\mathrm{well}}} \tag{4.31}$$

If p_{res} is assumed to equal the hydrostatic pressure of water at the well depth, this can be a boundary condition for the relation between velocity and density at the lower en of the well pipe. If $p_{\text{res}} = \rho_{\text{sea}} g \left(H_{\text{well}} + H_{\text{riser}}\right)$ and $Q = A_{\text{well}} V_{\text{well}}$,

$$V_{\text{well}} = \frac{\text{PI}}{A_{\text{well}}} \left[\rho_{\text{sea}} g \left(H_{\text{well}} + H_{\text{riser}}\right) - p_{\text{well}}\right] \tag{4.32}$$

4.6 The Capture Dome

For the purpose of simplification, the capture dome itself can be assumed to be a frictionless and reflection-free joint between the riser and the well. This means,

$$A_{\text{riser}} V_{\text{riser}} = A_{\text{head}} V_{\text{head}} \tag{4.33}$$

at the point where they two are connected. Therefore, the capture dome does not exist in the simulator. This is of course a radical simplification, and the simulated flow will not act exactly like a real case. However, a fully simulated capture dome would most likely just work as added friction to the system, only dampening velocity change as an effect of pressure change.

The simplified system is therefore less dampened and more prone to velocity change as an effect of pressure fluctuations, increasing the necessity of a fine-tuned opening procedure.

4.7 The Top Valve

Much like the capture dome, the top valve is also very simplified. It is assumed that this valve is able to operate in an ideal way, just lowering the pressure according to any function. The valve diameter is also assumed to be the same as in the rest of the riser. The velocity will of course change as an effect of this, but the relation between the pressure and the velocity will stay the same as in a general boundary condition, as developed in Sec. 5.4.

4.8 Thermodynamic Properties of the Fluids

The bulk modulus of elasticity can in most cases be evaluated as a constant, and this is also assumed for both liquids in this case; sea water and oil. Tab. (4.1) shows density, speed of sound, and bulk modulus of elasticity for the two. For the oil, this is calculated from Macondo well data, with the procedure shown below.

According to the United States Department of Energy, the density of the oil recovered from the Macondo well was $\sim 37°\text{API}$ @ $T = 80\ °\text{F}$ (≈ 300 K, $540\ °\text{R}$), $p = 1$ atm (≈ 101.3 kPA).[5] API gravity[‡] is given by

[‡]American Petroleum Institute gravity. API > 10 means the oil is lighter than water.

Properties of Liquids				
Liquid	ρ [kg/m^3]	a [m/s]	K [GPa]	μ [Pa · s]
Sea water	1,025	1,535	2.42	$1.00{\cdot}10^{-3}$
Oil	840	1,315	1.45	.07

Table 4.1: Properties of sea water and oil.

$$\text{API} = \frac{141.5}{\text{SG}} - 131.5 \tag{4.34}$$

in which SG is the specific gravity,

$$\text{SG} = \frac{\rho_{\text{oil}}}{\rho_{\text{H}_2\text{O}}} \tag{4.35}$$

with $\rho_{\text{H}_2\text{O}} = 10^3$ kg/m^3, this gives $\rho_{oil} = 840$ kg/m^3. Furthermore, the bulk modulus of elasticity can be found, using the formula derived from experimental data by Wostl et al. (1970),[6]

$$\begin{aligned} K = {} & 1.286 \cdot 10^6 + 13.55 \cdot p - 4.122 \cdot 10^4 \cdot \sqrt{T} \\ & - 4.53 \cdot 10^3 \cdot API - 10.59 \cdot (API)^2 + 3.228 \cdot T \cdot API \end{aligned} \tag{4.36}$$

which takes p in psi (1 atm = 14.696 psi) and T in °R as input, and produces K in psi. This is applied to find that for the case above, $K = 2.107 \cdot 10^5$ psi (= 1.45 GPa). Eq. (7.1) is used to find the speed of sound, $a = 1,315$ m/s.

4.9 Other Properties

There are also a number of other properties in the system, many of which are connected to what type of material is used in the pipes. Stainless steel is the most likely material, both as liner for the well and for the riser. Various properties of steel are given in Tab. (4.2), where a pipe with diameter 400 mm is chosen, at ASME SCH80S.

The riser will be fully filled with oil at any time, with $\rho_{\text{oil}} < \rho_{\text{sea}}$, so that it may be buoyant. This is countered by the bulk weight of the steel in the riser, so that

$$F_{\text{buoyancy}} = \rho_{\text{sea}} g H_{\text{riser}} \frac{\pi}{4} (D + e)^2 - \rho_{\text{oil}} g H_{\text{riser}} \frac{\pi}{4} D^2 - m/L \cdot g H_{\text{riser}} \tag{4.37}$$

Properties of Riser			
Property	Unit	Size	Source
E	[GPa]	207.0	[7]
ϵ	[m]	$2.0{\cdot}10^{-3}$	[4]
D_{nom}	[m]	$400.0{\cdot}10^{-3}$	[8]
e	[m]	$12.7{\cdot}10^{-3}$	[8]
m/L	[kg/m]	123.3	[8]

Table 4.2: Properties of the steel used in the piping.

which means the riser is buoyant if

$$\rho_{\text{oil}} < \frac{1}{D^2}\left[\rho_{\text{sea}}\left(D+e\right)^2 - \frac{m/L}{\pi/4}\right] \tag{4.38}$$

In this case, the last term becomes 109.93 kg/m^3, which is far less than ρ_{oil}, at 840 kg/m^3. The riser bulk weighs in at 185 Mg, with a resulting gravitational force of 1.81 MN. This means the riser is not buoyant, unless large quantities of hydrocarbon gases are encountered. This is a very real possibility, but it will not be taken into account in the further treatment.

5 The Method of Characteristics

For the sake of repetition, the basic flow equations, namely that of continuity, Eq. (4.18), and that of energy, Eq. (4.7) are given by

$$\frac{\dot{A}}{A} + \frac{\dot{\rho}}{\rho} + V_x = 0 \tag{5.1a}$$

$$\frac{1}{\rho} p_x + V V_x + V_t + g \sin\alpha + \frac{f \cdot V\,|V|}{2D} = 0 \tag{5.1b}$$

These equations are now to be solved numerically in a transient analysis. For this system, the Method of Characteristics is chosen, as it has many advantages when applied to a system with start-up or shut-down. The most prominent being its ability not to "blur" the pressure waves, but keep them crisp and clear throughout the entire pipe.

The following chapter is mainly a summary of Chapter 4 in Wylie & Streeter (1993),[7] with some minor adaptations and fixes.

5.1 The + and - Characteristics

Remembering the compression module, K, from Eq. (5.2)

$$\frac{\dot{\rho}}{\rho} = \frac{\dot{p}}{K} \tag{5.2}$$

and speed of sound, or the pressure propagation speed in a liquid,

$$a = \sqrt{\frac{K/\rho}{1 + (K/A)\left(\dot{A}/\dot{p}\right)}} \tag{5.3}$$

or, as written in Eq. (4.27),

$$a = \sqrt{\frac{K/\rho}{1 + (K/E)\,(D/e)}} \tag{5.4}$$

these can be combined into Eq. (5.1a) to yield,

$$V p_x + p_t + \rho a^2 V_x = 0 \tag{5.5}$$

The Method of Characteristics requires two equations, L_1 and L_2, to describe the problem. In this case, L_1 is Eq. (5.1a) and L_2 is Eq. (5.1b).

$$L_1 = Vp_x + p_t + \rho a^2 \cdot V_x = 0 \tag{5.6a}$$

$$L_2 = \frac{1}{\rho} p_x + VV_x + V_t + g\sin\alpha + \frac{fV\,|V|}{2D} = 0 \tag{5.6b}$$

These can now be combined into one equation, L, by multiplying L_1 by the unknown λ.

$$L = \lambda L_1 + L_2 = 0 \tag{5.7}$$

$$= \lambda \left[Vp_x + p_t + \rho a^2 \cdot V_x \right] + \frac{1}{\rho} p_x + VV_x + V_t + g\sin\alpha + \frac{fV\,|V|}{2D} = 0 \tag{5.8}$$

$$= \lambda \left[p_x \left(V + \frac{1}{\lambda\rho} \right) + p_t \right] + \left[V_x \left(V + \lambda\rho a^2 \right) + V_t \right] + g\sin\alpha + \frac{fV\,|V|}{2D} = 0 \tag{5.9}$$

Introducing the derivative $\frac{d}{dt}$,

$$\frac{dp}{dt} = p_x \frac{dx}{dt} + p_t \tag{5.10}$$

$$\frac{dV}{dt} = V_x \frac{dx}{dt} + V_t \tag{5.11}$$

This can be recognized in Eq. (5.9), so

$$\frac{dx}{dt} = V + \frac{1}{\lambda\rho} = V + \lambda\rho a^2 \tag{5.12}$$

$$\Rightarrow \lambda = \pm \frac{1}{\rho a} \tag{5.13}$$

$$\Rightarrow \frac{dx}{dt} = V \pm a \tag{5.14}$$

Substituting Eqs. (5.10) and (5.11) and Eq. (5.14) into Eq. (5.9) yields,

$$\pm \frac{1}{\rho a} \frac{dp}{dt} + \frac{dV}{dt} + g\sin\alpha + \frac{fV\,|V|}{2D} = 0 \tag{5.15}$$

Eqs. (5.15) and (5.14) can now be written as two sets of ordinary differential equations; the C^+ and C^- characteristics.

$$C^+ : \begin{cases} \frac{1}{\rho a}\frac{dp}{dt} + \frac{dV}{dt} + g\sin\alpha + \frac{fV|V|}{2D} = 0 \\ \frac{dx}{dt} = V + a \end{cases} \tag{5.16a}$$

$$C^- : \begin{cases} -\frac{1}{\rho a}\frac{dp}{dt} + \frac{dV}{dt} + g\sin\alpha + \frac{fV|V|}{2D} = 0 \\ \frac{dx}{dt} = V - a \end{cases} \tag{5.16b}$$

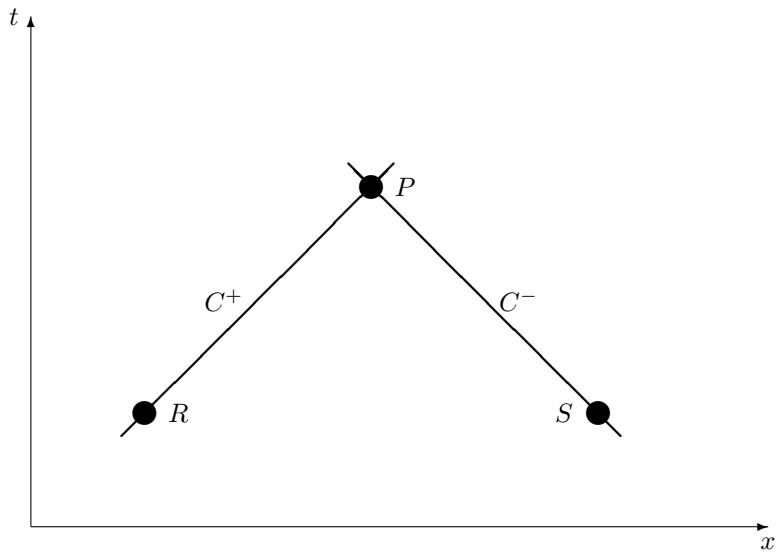

Figure 5.1: Characteristic lines.

The C^+ and C^- characteristics will now appear as lines along a given $\frac{dx}{dt}$, where the speed of sound may or may not be constant, as shown in Fig. (5.1). After this treatment of the equations, all the variables are still transient, including pipe area and pressure.

The characteristics can now be discretized using a finite difference scheme. To simplify, Eq. (5.2) is used to substitute p with ρ. The bulk modulus of elasticity and the speed of sound are considered constant. Integrating the C^+ characteristic

from R to P, yields

$$-K\left[\frac{1}{\rho_P}-\frac{1}{\rho_R}\right]+a\left[V_P-V_R\right]+ag\sin\alpha\left[t_P-t_R\right] + \frac{afV_P\left|V_R\right|}{2D}\left[t_P-t_R\right]=0 \tag{5.17a}$$

$$x_P-x_R=\left(V_R+a\right)\left[t_P-t_R\right] \tag{5.17b}$$

The friction term was integrated in parts, with the diameter assumed constant, and only the term $V\left|V\right|$ remained to be integrated.

Similarly, integrating the C^- characteristic from S to P yields,

$$-K\left[\frac{1}{\rho_P}-\frac{1}{\rho_S}\right]+a\left[V_P-V_S\right]+ag\sin\alpha\left[t_P-t_S\right] + \frac{afV_P\left|V_S\right|}{2D}\left[t_P-t_S\right]=0 \tag{5.18a}$$

$$x_P-x_S=\left(V_S-a\right)\left[t_P-t_S\right] \tag{5.18b}$$

The pressure can be found from the density by integration of Eq. (5.2),

$$p_P=p_0+K\ln\frac{\rho_P}{\rho_0} \tag{5.19}$$

which turned around can yield density from pressure,

$$\rho_P=\rho_0\cdot\exp\left[\frac{p_P-p_0}{K}\right] \tag{5.20}$$

Furthermore, by eliminating pressure, the area variation can be found by combining Eqs. (5.3) and (5.2),

$$\frac{dA}{A}=\frac{K}{a^2}\frac{d\rho}{\rho^2}-\frac{d\rho}{\rho} \tag{5.21}$$

and integrating, to get the real area, A_P,

$$A_P=A_0\frac{\rho_0}{\rho_P}\exp\left[\frac{K}{a^2}\left(\frac{1}{\rho_0}-\frac{1}{\rho_P}\right)\right] \tag{5.22}$$

It is also worth noting that the actual mass flow rate is $\dot{m}=\rho_P A_P V_P$, and will require the calculation of the real area A_P.

5.2 Steady-State Solution

To ensure the consistency of the results, both for the transient and the steady-state solutions, it would be an advantage to have the steady-state solution be abbreviated from the transient solution. To do this, we must define the steady-state condition, $\dot{m} = \rho AV = \text{const}$ is defined as

$$d\dot{m} = d(\rho AV) = 0 \tag{5.23}$$

$$\Rightarrow \frac{d\rho}{\rho} + \frac{dA}{A} + \frac{dV}{V} = 0 \tag{5.24}$$

Combining this with Eq. (5.21), yields

$$\frac{d\rho}{\rho^2} = -\frac{a^2}{K}\frac{dV}{V} \tag{5.25}$$

At the downstream side, Eqs. (5.3) and (5.25) are substituted into Eq. (5.16a), while the steady-state condition makes $dV = 0$,

$$\frac{dV^2}{dx} = 2\frac{V^2}{a^2}\frac{g\sin\alpha + fV^2/2D}{1-(V/a)^2} \tag{5.26}$$

If $(V/a)^2 \ll 1$, D is assumed constant, and define

$$s = \frac{2g\sin\alpha}{a^2} \tag{5.27}$$

integrating from R to P yields,

$$V_P = \frac{V_R}{\exp\left[-s\Delta x/2\right] - fV_R\left|V_R\right|\Delta x/\left(2Da^2\right)} \tag{5.28}$$

Substituting Eq. (5.25) into Eq. (5.26), utilizing the same assumptions and approximations, yields

$$K\frac{d\rho}{\rho^2} = -\left(g\sin\alpha + \frac{fV^2}{2D}\right)dx \tag{5.29}$$

This can now be solved, since V_P is already known from Eq. (5.28). Integration yields

$$\frac{1}{\rho_P} = \frac{1}{\rho_R} + \left(g \sin\alpha + \frac{f V_P \left| V_R \right|}{2D} \right) \frac{\Delta x}{K} \tag{5.30}$$

For the upstream values, the same procedure is applied, but Eq. (5.16b) is used instead of Eq. (5.16a) to yield,

$$V_P = \frac{V_S}{\exp\left[s\Delta x/2 \right] + f V_S \left| V_S \right| \Delta x / \left(2Da^2 \right)} \tag{5.31}$$

and

$$\frac{1}{\rho_P} = \frac{1}{\rho_S} - \left(g \sin\alpha + \frac{f V_P \left| V_S \right|}{2D} \right) \frac{\Delta x}{K} \tag{5.32}$$

This is everything needed to calculate velocity and density (and thereby pressure), given any initial conditions, at either side of the system.

5.3 Interpolations

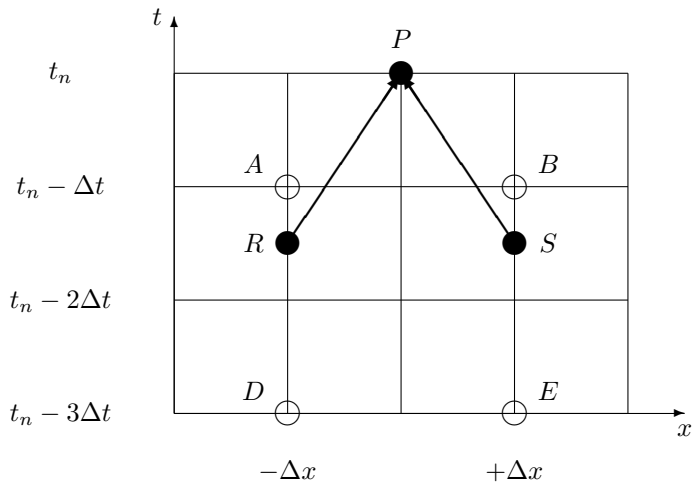

Figure 5.2: Interpolations.

The method of characteristics is different from many other transient methods, in that it does not instantly make it possible to include the time steps in the relations. Instead, their relation is given by Eqs. (5.17b) and (5.18b).

This gives two alternative approaches to calculate the next time step, namely to keep the relation between Δx and Δt constant, or to have a variable Δt and Δx. The latter of these approaches produces the nicest results, as they have constant Δt and Δx. However, this requires interpolations to find values for the velocity and density at points R and S.

Fig. (5.2) shows the grid that now forms. P lies at (x_m, t_n), and A, B, D and E are previously calculated at earlier time steps. Interpolating this, yields

$$\frac{V_A - V_R}{t_A - t_R} = \frac{V_A - V_D}{t_A - t_D} \tag{5.33}$$

$$(V_A - V_R)(t_A - t_D) = (V_A - V_D)(t_A - t_R) \tag{5.34}$$

Here, Fig. (5.2) shows that,

$$t_R = t_P - \frac{\Delta x}{V_R + a} \tag{5.35}$$

$$t_P - t_D = 3\Delta t \tag{5.36}$$

$$t_A - t_D = 2\Delta t \tag{5.37}$$

Furthermore, introducing $\Theta = \frac{\Delta x}{2\Delta t}$ and inserting this into Eq. (5.34), yields

$$(V_A - V_R)(V_R + a) = \left(\frac{3}{2}(V_R + a) - \theta\right)(V_A - V_D) \tag{5.38}$$

this can be rearranged into a second-order equation;

$$V_R^2 - \left(\frac{3}{2}V_A - \frac{1}{2}V_D - a\right)V_R - (V_A - V_D)\left(\frac{3}{2}a - \theta\right) - aV_D = 0 \tag{5.39}$$

V_R and ρ_R can then be found by

$$V_R = V_X + \sqrt{V_X^2 + (V_A - V_D)\left(\frac{3}{2}a - \theta\right) + aV_D} \tag{5.40}$$

$$\rho_R = \rho_D + (\rho_A - \rho_D)\left(\frac{3}{2} - \frac{\theta}{V_R + a}\right) \tag{5.41}$$

where $V_X = \frac{3}{4}V_A - \frac{1}{4}V_D + \frac{1}{2}a$. Likewise, the interpolated values for V_S and ρ_S can be found by:[§]

$$V_S = V_Y - \sqrt{V_Y^2 - (V_B - V_E)\left(\frac{3}{2}a - \theta\right) - aV_E} \tag{5.42}$$

$$\rho_S = \rho_E + (\rho_B - \rho_E)\left(\frac{3}{2} + \frac{\theta}{V_R - a}\right) \tag{5.43}$$

where $V_Y = \frac{3}{4}V_B - \frac{1}{4}V_E - \frac{1}{2}a$.

5.4 The Numerical Method

The grid is now fixed, thanks to the interpolations in Sec. 5.3. Now, there are six equations that all have to be solved in turn to find V_P and ρ_P. They are Equations (5.40) to (5.43) and Eqs. (5.17a) and (5.18a). The latter two can be rearranged to yield

$$\boxed{\frac{1}{\rho_P} = C_P + B_P V_P} \tag{5.44}$$

$$\boxed{\frac{1}{\rho_P} = C_M - B_M V_P} \tag{5.45}$$

or

$$\boxed{\rho_P = \frac{B_P + B_M}{C_P B_M + C_M B_P}} \tag{5.46}$$

$$\boxed{V_P = \frac{C_M - C_P}{B_M + B_P}} \tag{5.47}$$

where

[§]These equations differ somewhat from those in Chapter 4 in Wylie & Streeter (1993).[7] This is due to some small errors there.

$$\boxed{C_P = \frac{1}{\rho_R} - BV_R + \frac{Bg\Delta x \sin\alpha}{V_R + a}} \tag{5.48}$$

$$\boxed{B_P = B + \frac{R\,|V_R|}{V_R + a}} \tag{5.49}$$

$$\boxed{C_M = \frac{1}{\rho_S} + BV_S + \frac{Bg\Delta x \sin\alpha}{V_S - a}} \tag{5.50}$$

$$\boxed{B_M = B + \frac{R\,|V_S|}{V_S - a}} \tag{5.51}$$

here, $B = \frac{a}{K}$ and $R = \frac{af\Delta x}{2DK}$. These equations can now be used as a basis for a computer program to solve the equations.

5.5 Stability

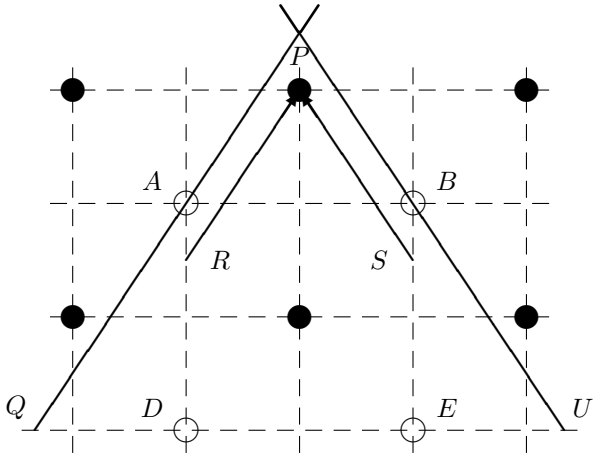

Figure 5.3: Stability

The convective terms VV_x in the characteristic equations (Eqs. (5.16a) and (5.16b)) makes the characteristic lines appear as curved in the xt plane. The interpolations of V_R and V_S is required because of this curved nature, and it is also what can ultimately make the numeric solution unstable.

An easy way to explain this is shown in Fig. (5.3). For the interpolated velocities and pressures to be correct, $t_R \leq t_A$ and $t_S \leq t_B$. Otherwise, they

would not be interpolations between known values, but rather extrapolations into the unknown. They would therefore easily lead to errors, and ultimately instabilities. Following this thought yields,

$$t_R \leq t_A \tag{5.52}$$

$$t_P - \frac{\Delta x}{V_R + a} \leq t_A \tag{5.53}$$

$$\Delta x \leq \Delta t \left(V_R + a\right) \tag{5.54}$$

and

$$t_S \leq t_B \tag{5.55}$$

$$t_P + \frac{\Delta x}{V_S - a} \leq t_B \tag{5.56}$$

$$\Delta x \leq \Delta t \left(-V_S + a\right) \tag{5.57}$$

The velocities V_R and V_S can be either positive or negative, so the absolute value should be used. Furthermore, since the same grid is wanted all over the x axis, a common, maximum value for the velocity should be used. This yields the Courant condition from Wylie & Streeter (1993):[7]

$$\Delta t \left(|V| + a\right) \leq \Delta x \tag{5.58}$$

or

$$\theta \geq \frac{1}{2} \left(|V| + a\right) \tag{5.59}$$

The value of V can now be found in two ways; either by finding the maximum velocity through all time steps in the pipe, or by finding the maximum velocity at each time step. The latter of these can be used in the implementation of variable time steps at start-up or shut-down of the flow, where the maximum velocity can vary strongly from time step to time step. This is discussed further in Sec. 6.5.

5.6 Interpolation Errors

Interpolation errors arise when the relation between Δx and Δt wander too far away from the critically stable relation given in Sec. 5.5. Fig. (5.4) shows how these errors transmit through the xt plane, and exert a dampening effect on the

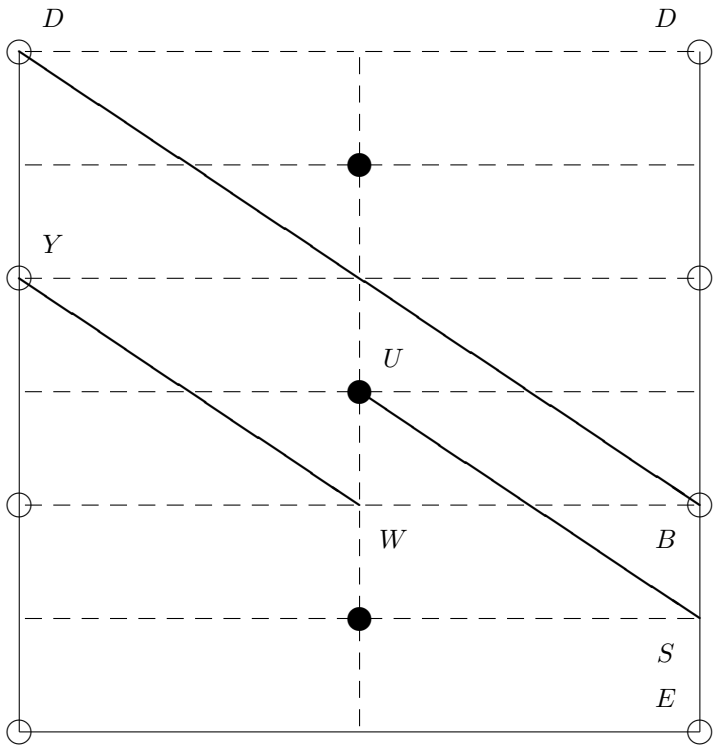

Figure 5.4: Interpolation error, xt plane

results. Here, an extreme situation with 50 % interpolation is described (i.e. $\theta = a \approx 2\theta_{stable}$).

If a sudden change in pressure occurs at point B, the theoretical propagation of this information is shown as the line BD. However, since S is an interpolation of B and E, 50 % of this disturbance will reach point U ahead of time, and 50 % of this, i.e. 25 % of the original disturbance, will in turn reach Y well ahead of time. The solution to get rid of this is quite simple, yet time consuming; to keep θ as close as possible to the stability criteria, Eq. (5.59), so that the interpolation points R and S are as close as possible to $t - \Delta t$.

6 The MATLAB Programs

There are numerous ways to numerically solve the equations given in Sec. 5. A large number of programming languages can be applied, all with their individual strengths and weaknesses. However, MATLAB points itself out as the best choice. This is mainly because of its powerful array of built-in functions, including easy plotting of results, and the fact that it has no need for compilation prior to running programs.

Other languages are however quicker, like FORTRAN or C++, but since ease of analysis has a higher priority than extreme accuracy in this treatment, the choice of language stands firm.

6.1 `storeinitdata.m`

This program is just a way of simplifying the procedure of changing grid sizes and other constants for the whole system, and ensures that the same constants are used as background for all calculations. Data like gravitational acceleration, $g = 9.806\ 65\ \mathrm{m/s^2}$ and atmospheric pressure $p_0 = 101.3$ kPa are stored in the file `initdata.mat`. Many of these constants stem from SI Chemical Data (2002),[9] while others stem from the likes of Tabs. (4.1) and (4.2).

6.2 `blowout.m`

The simplest of all the programs, `blowout.m`, with full code in App. A.2, calculates the flow in the well (and in the riser) at blowout conditions. It uses the equations in Sec. 5.2 to calculate PI, p and V. This is done in two turns, upstream and downstream, since the boundary conditions $p_{\mathrm{head}} = p_{\mathrm{seabed}}$ and Q_0 are in the middle of the system.

The program can calculate up to four (or any number with some recoding) different cases with e.g. varying diameters or flow rates at one time. It uses the data stored in `initdata.mat`, and returns the results as various plots and the file `blowout.mat`.

6.3 `steadystate.m`

This is a more fiddly family of programs, as the two boundary conditions PI and p_{top} are on opposite ends of the system. The solution to this is to guess some initial value of the velocity V_{top}, and then use Eqs. (5.31) and (5.32) to calculate downstream all the way to the bottom of the well. Then, V_{well} can be found from the newly calculated ρ_{well} using Eq. (4.32).

The last step is then to use Eqs. (5.28) and (5.30), or the downstream equations to calculate new values of V_{top} and ρ_{top}. This is then repeated for as long

as the absolute value of the difference between $\rho_{\text{top,actual}}$ and $\rho_{\text{top,calculated}}$ is less than a defined accuracy factor, or that the number of iterations reaches a limit.

This back-and-forth iteration makes the program quite slow compared to the straight-forward calculation in `blowout.m`, but it is hard to make it any quicker. Anyway, the data are stored in a `.mat` file corresponding to its own filename.

6.4 `transient.m`

The main simulator in many ways, the `transient.m` family of programs use the full extent of the Method of Characteristics. The calculation is transient, and restrictions like change in diameter or valve opening can easily be incorporated. This makes testing different cases quite easy.

To save memory and hard drive space, only the last four iterations are kept in memory at one time, and also only a finite number of data points are stored to the hard drive in a `.mat` file corresponding to its own filename. These data only include time series along four points along the x axis; the bottom of the well, $m = 1$, the wellhead at the well side, $m = 2M$, the wellhead at the riser side, $m = 2M + 1$, and the top, $m = 3M + 1$.

6.5 The Grid

The grid is set up so that each of the three stretches of pipe (the horizontal and vertical well sections an the riser) all have their own Δx, but the same number of nodes M. This way, the bend in the well is always in position $M + 1$, the wellhead is at $2M + 1$, and the top valve at $3M + 1$.

This changing Δx can make the system unstable or inaccurate, but this is no real problem here. As shown in Sec. 5.6, one should try keep Δx as close as possible to the stability criteria, but still low enough to be stable. The stability criteria repeated from Eq. (5.58) is

$$\Delta t \left(|V| + a \right) \leq \Delta x \tag{6.1}$$

which means that the smallest Δx in the pack defines the stability. This is the horizontal well bit can be assumed as both stable and without interpolation errors, whereas the rest of the grid might encounter the latter. Since H_{well} (vertical length of well) and H_{riser} are both 1,500 m long, i.e. 1.5 times the length of L_{well} (horizontal length of well), some interpolation errors can be expected, by this is dealt with by just increasing the number of nodes.

All cases these programs are intended for, all have values of V and a in the same spectrum. It would therefore be practical to make $\Delta t = t_{\text{end}}/N$ a function of M. For a low-Mach number flow, $a + |V| \approx a$ with an added 20 or so m/s to

compensate for any changes in a will be quite sufficient. But still, much of this is just trial and error. For instance, for a case with a diameter of 400 mm, it turns out that a relation $\Delta x = 1,250\Delta t$ yields a stable simulation, even though the free-stream speed of sound in the oil is 1,315 m/s, as derived in Sec. 4.8. This is because of the effect of compressibility and pipe diameter on the speed of sound, and is discussed further in Sec. 7.3.1.

Every new value of ρ and V at a point P, located at (m, n), in which m is the node in x direction, and n is the iteration number, is calculated from values at positions $(m-1, n-1)$, $(m+1, n-1)$, $(m-1, n-3)$, and $(m+1, n-3)$ or points A, B, D, and E, respectively. This is shown in Fig. (5.2).

7 Results

A number of scenarios has been calculated, using the method developed in Sec. 5 and the programs described in Sec. 6. They range from simulating a full blowout, to a steady-state solution of the whole system put together, and finally, to various transient solutions of the start-up of the system.

7.1 Blowout

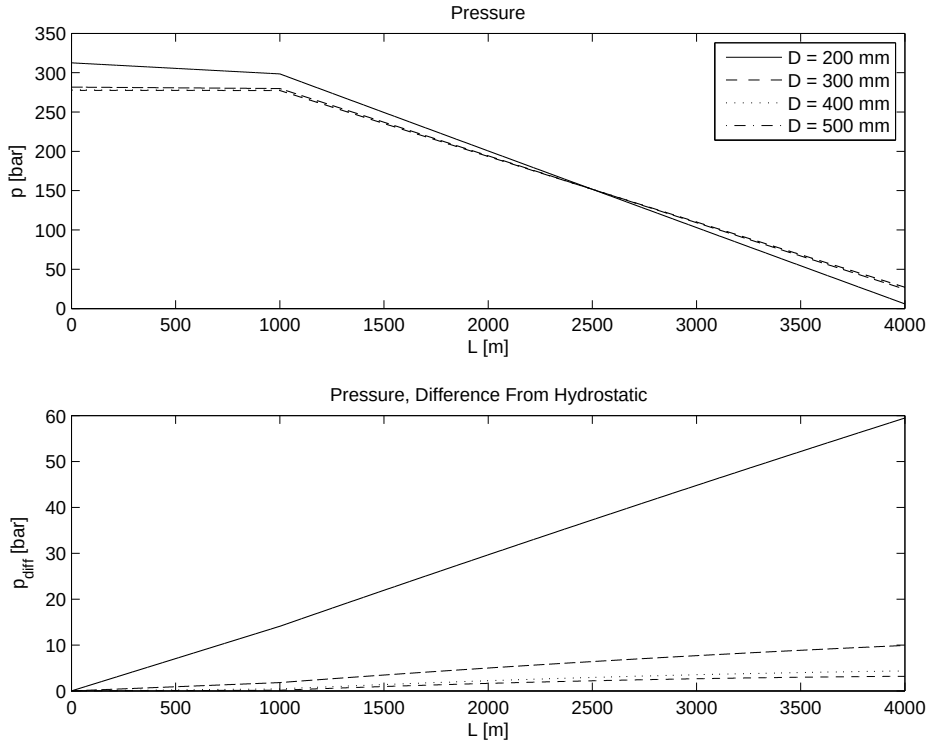

Figure 7.1: Steady-state pressures at various volumetric flux and well diameter.

The object of simulating a full blowout situation of the Macondo well, is to find a reasonable value of PI, the productivity index. We need to know this to be able to simulate the well at conditions other than full blowout. The well itself is simulated as two stretches of pipe, one horizontal and one vertical with lengths of 1,000 m and 1,500 m, respectively. In addition comes the 1,500 m

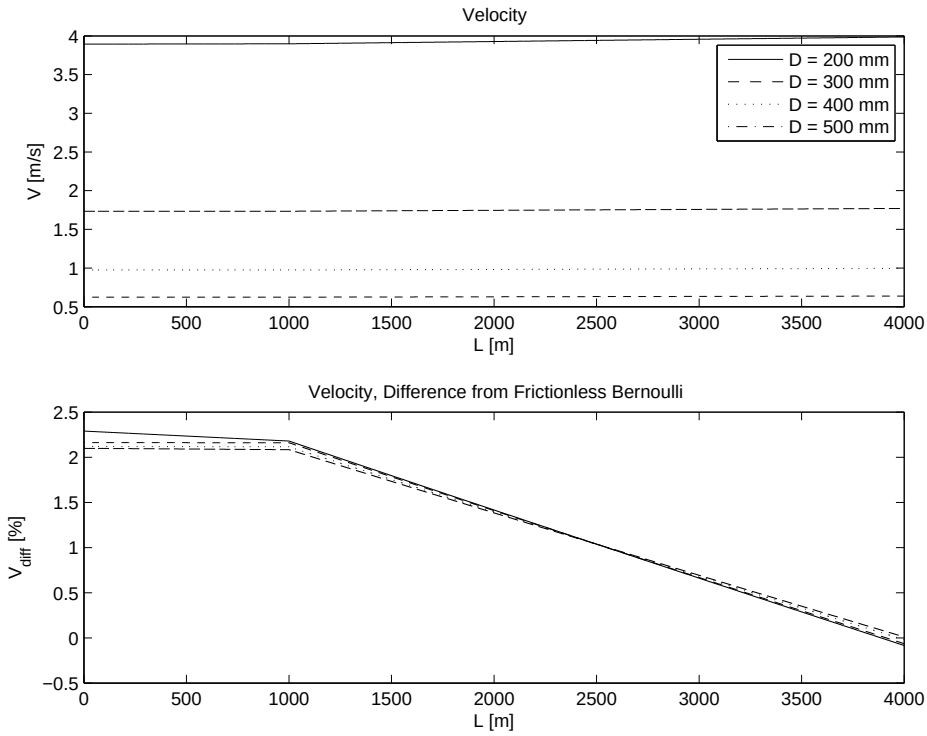

Figure 7.2: Steady-state velocities at various volumetric flux and well diameter.

long vertical riser to the collection and treatment vessel at sea level. For the Macondo well, there were many different estimates of the blowout flux, ranging from $5 \cdot 10^3$ bbl/day to about $1.0 \cdot 10^5$ bbl/day. Crone (2010)[3] calculates the total leakage to $6.8 \cdot 10^4$ bbl/day (0.12 m^3/s), based on video footage from the well. We now want to find a fitting diameter of the well.

These flows are calculated with four different well (and riser) diameters; 200, 300, 400 and 500 mm, using the MATLAB program `blowout.m` (See Sec. 6.2). The grid was set at 6000 points along the x axis, 2000 points in each pipe section, and yielded p and V shown in Figures (7.1) and (7.2). Calculation time was a mere number of seconds on a normal laptop computer. The angles in the graph stem from the change from horizontal to vertical well at 1000 m and the change in diameter from the well to the riser at 2500 m.

In Fig. (7.1), the pressure has a steep increase when decreasing diameter below 300 mm. This is caused by the low diameter's effect on the friction factor. This

Case	D [mm]	PI $[\mathrm{m}^3/\mathrm{bar}\cdot\mathrm{s}]$	p_{well}	p_{head} [bara]	p_{top}	V_{well}	V_{head} [m/s]	V_{top}
1	200	$-12.53\cdot10^{-3}$	313.6	151.8	6.0	3.89	3.94	3.99
2	300	$5.99\cdot10^{-3}$	281.7	151.8	24.7	1.73	1.75	1.77
3	400	$5.14\cdot10^{-3}$	278.2	151.8	26.7	.98	.99	1.0
4	500	$4.99\cdot10^{-3}$	277.5	151.8	27.2	.62	.63	.64

Table 7.1: Blowout at $6.8 \cdot 10^4$ bbl/day, different well and riser diameters

low diameter also makes the well useless and unproductive, given the data in Tab. (7.1). A negative PI, as in the two first cases, has no root in reality, as the oil must see a negative pressure gradient through the well to actually cause a leakage.

The 200 mm diameter can therefore be discarded, and instead focus is drawn on the latter three. For the 300 to 500 mm well bores, there is only about 4.2 bar difference in p_{well}, and therefore also only a 16 % difference in PI from small to large.

p_{top} also varies very little for all cases, but it is too soon to determine what riser diameter to choose, based on these results, keeping in mind that this should be optimized to collect the oil as easily as possible. However, it shows one key component, and that is the pressure reserves available on sea level to make the sought-after hydrodynamic seal on the sea bed. There is between 24.7 and 27.2 bar of excess pressure at sea level, that can be released with a valve to make the capture dome stick to the sea floor.

In further treatment, a well diameter of 300 mm is chosen. This is simply because it is the smaller one of the three that produce quite similar results. The choice between the three higher diameters will not be that important, as friction loss is more or less the same in both cases.

7.2 Steady-State

The next step towards simulating start-up, is to check what happens when varying the top pressure and riser diameter. This was done at two riser diameters, 300 and 400 mm, using the MATLAB program `steadystate.m` with 6000 nodes in the x direction. The well diameter was 300 mm in both cases.

Tab. (7.2) shows numerical values for key pressures along the piping, along with Q_0, given in standard m^3/s, and the top velocity. The top row in each of the two tables is the situation at undisturbed blowout, while the following four are the results when reducing the pressure in four steps; 20, 15, 10, and 5 bara.

There are, as in the blowout scenario, no big differences between the pressures

$D_{\text{riser}} = 300$ mm						
p_{well}	p_{head} [bara]	p_{top}	V_{well}	V_{head} [m³/s]	V_{top}	Q_0 [m/s]
281.7	151.8	24.7	1.73	1.75	1.77	.125
278.9	147.9	20.0	1.97	1.99	2.01	.142
276.0	143.6	15.0	2.21	2.23	2.26	.160
273.2	139.8	10.0	2.44	2.47	2.49	.176
270.6	135.3	5.0	2.66	2.69	2.72	.192
$D_{\text{riser}} = 400$ mm						
p_{well}	p_{head} [bara]	p_{top}	V_{well}	V_{head} [m³/s]	V_{top}	Q_0 [m/s]
278.2	151.8	26.7	0.98	.99	1.0	.125
277.1	145.3	20.0	2.12	2.14	1.22	.153
273.9	140.5	15.0	2.39	2.41	1.37	.172
270.8	135.7	10.0	2.64	2.67	1.52	.191
267.8	132.9	5.0	2.89	2.93	1.66	.209

Table 7.2: Variation of p_{top} at $D_{\text{riser}} = 300$ mm and 400 mm, with $D_{\text{well}} = 300$ mm.

at the two riser diameters. Nor is Q_0 significantly different in the two sets. In fact, the only significant difference is seen in the velocities, but this is only natural when varying the diameter.

Fig. (7.3) shows another representation of the scenarios, in which the section from 0 to 1000 m is the horizontal section of the well, 1000 to 2500 m is the vertical well section, and 2500 to 4000 m is the riser. The blowout scenarios for the respective diameters are used as a basis. The graphs then represent the difference in pressure along the system when the top pressure is reduced.

This mainly shows that a reduction in p_{top} of 1 bar only leads to about 0.85 bar reduction in p_{head} at $D_{\text{riser}} = 300$ mm, while at $D_{\text{riser}} = 400$ mm, it changes by 0.97 bar at the wellhead for every bar at sea level. This makes a riser diameter of 400 mm the best choice of the two, since this reduces more pressure on sea floor bar for bar than the 300 mm case. The top pressure is also higher in this case, so there is more of a margin for the pressure reduction.

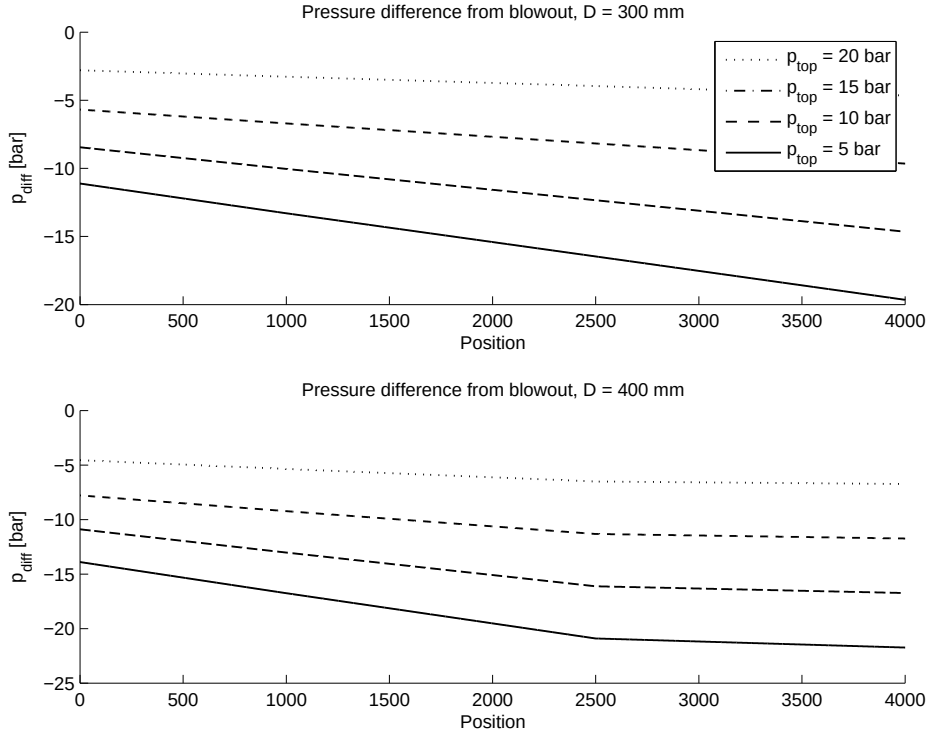

Figure 7.3: Steady-state results of a reduction in p_{top} through the system, relative to the blowout scenarios.

7.3 Variability of The Speed of Sound and Other Secondary Variables

In Sec. 5, concerning the application of the Method of Characteristics to pipe flow, a few additional variables to V and p were mentioned. Among those were the speed of sound a, given by Eq. (4.27), the real area A_P, given by Eq. (5.22), and the density ρ, given by Eq. (5.46). The 20 bar, 300 mm steady-state case can be used as a basis for seeing how changing pressures and velocities make these variables behave.

Fig. (7.4) shows how this plays out through a steady-state simulation with p and D as mentioned earlier. To fit both a, A and ρ into the same figure, every series has been divided by their maximum value.

The series vary in the intervals $1,205 \text{ m/s} < a < 1,216 \text{ m/s}$, $0.707 \text{ m}^2 < A < 0.709 \text{ m}^2$ and $841 \text{ kg/m}^3 < \rho < 856 \text{ kg/m}^3$. By comparison, their values at

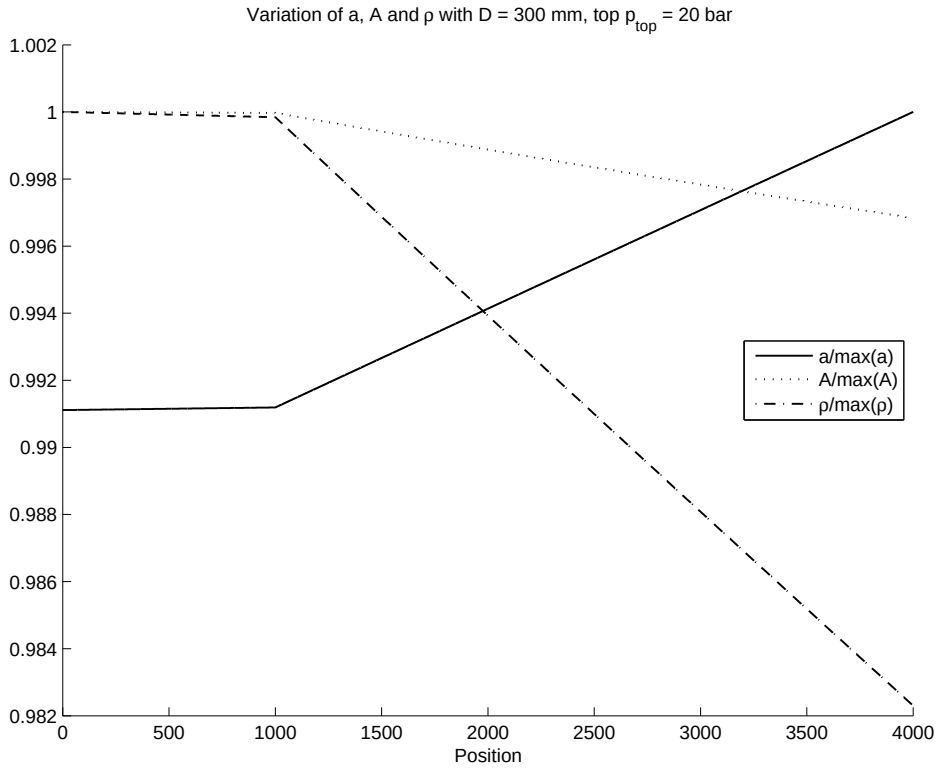

Figure 7.4: Variation of a, A and ρ through a steady-state system. Series are divided by their maximum values.

standard conditions are $a_0 = 1,315$ m/s, $A_0 = 0.0707$ m^2 and $\rho_0 = 840$ kg/m^3.

7.3.1 The Speed of Sound, a

a is given by

$$a = \sqrt{\frac{K/\rho}{1 + (K/E)\,(D/e)}} \tag{7.1}$$

which means it is not linear, but rather an exponential function of mainly ρ, as this is the only size that varies through a single stretch of pipe. Also, the standard value of a is the free-stream speed of sound of the fluid, and assumes

$(K/E)\,(D/e) \ll 1$ (which for $D = 300$ mm is 0.17 and for $D = 400$ mm is 0.22), so that what remains is just $a_0 = \sqrt{K/\rho_0}$. By correcting for this, the new and more easily comparable a becomes 1,217 m/s for 300 mm diameter and 1,190 m/s for 400 mm diameter, or respectively about 7.5 and 9.5 % lower than the free-stream speed of sound, which is quite significant.

The speed of sound also decreases with increasing pressure, in this system by about 0.9 % from top to bottom. Not very much, and had it only been up to the change in pressure decreasing the sound speed, then a could have been calculated as a constant. However, the diameter changes in many of these simulations, so a variable a will be required for accurate results.

7.3.2 The Real Area, A

Similarly, A, given by

$$A_P = A_0 \frac{\rho_0}{\rho_P} \exp\left[\frac{K}{a^2}\left(\frac{1}{\rho_0} - \frac{1}{\rho_P}\right)\right] \tag{7.2}$$

is the variable with the least change from top to bottom of the three, only changing by about 0.3 % from max to min. It could therefore have been replaced by the constant A_0 without too grave consequences. However, the real area is not a part of the numerical system, and only comes into play as a correction factor when calculating the flow rate $Q_0 = M/\rho_0 = \rho/\rho_0 V A$, in which M is the mass flow rate. So, as a correctional factor, is plays an important role even though it varies very little.

7.3.3 The Density, ρ

The density ρ, given by

$$\rho_P = \rho_0 \cdot \exp\left[\frac{p_P - p_0}{K}\right] \tag{7.3}$$

varies the most out of these three series. This is no surprise, as it is directly connected to only the pressure and vice versa. In fact, for the sake of convenience, it replaces pressure in the numerical system, and together with V, it is one of two main variables in the calculation. All values of p in the results are therefore just a conversion of ρ, and all initial and boundary values given as pressures must also be converted to densities before calculation.

It would perhaps be "right" to present results in densities rather than pressures, but of course a result given in pressure is easier to read. Anyway, the point is that a variable ρ is essential to the calculation itself, and must therefore stay so.

7.4 Transient Start-Up

The transient startup procedure is done with using a well diameter of 300 mm and a riser diameter of 400 mm, as chosen in Secs. 7.1 and 7.2. Initial conditions are blowout in the well, with data from case 2 in Sec. 7.1, and stagnant fluid in the riser.

One important limitation is that the flow rate Q_0 cannot be reduced below that of an undisturbed blowout. It must be assumed that all measures have been made to stop or slow the blowout, so that Q_0 at blowout is at its practically allowed low-point.

Another limitation connected to this, is that the pressure at the bottom of the riser, or at the wellhead, cannot rise above the hydrostatic pressure at sea bed. This would only make it impossible to lower the end bit onto the leakage point properly, so it is dealt with in the program by setting the wellhead flow speed equal to its initial condition for as long as $p_{\text{dome}} \geq p_{\text{head}}$.

The physical way to deal with this assumption, is that the end bit of the riser has one or several one-way valves that release excess fluid when pressure on the inside is higher than on the outside. This will then allow for a smoother start-up of the riser.

7.4.1 Single Linear Opening of Top Valve

The initial test consists of four simulations, each with a slight difference in the behavior of the top valve. The valve opening is in this case simulated as a linear decrease in p_{top}, over the period t_{valve}. This is simulated at two pressures; 25 and 20 bara, and two valvetimes; 10 and 20 s.

The whole system is simulated over a time period of 40 s, with 6,000 nodes in x direction, and 100,000 nodes in time, using variations of `transient.m`, discussed in Sec. 6.4. Calculation time for each case was approximately 30 minutes.

Fig. (7.5) shows in its bottom plot how pressure is reduced at the top valve. It is easy to spot the linear reduction in pressure, the inclined sections that stretch from 0 to 10 or 20 s. The top plot shows what happens at the wellhead. There are no changes in pressure here, until the pressure in the capture dome decreases below the initial value of p_{head}. The pressure then drops rapidly for a couple of seconds, until the pressure wave returns from the well. In all cases, the pressure seems to flat out from about 5 s after the valve is completely open, and towards 40 s, it reaches near-steady-state pressure, as seen in Tab. (7.3).

In fact, the transient results seem to over-exceed the steady-state results, with slightly higher V and Q_0 (≈ 1 %) after 40 s than is seen in the steady-state case. The reason for this is unknown, but might be due to different programming structure for the two cases. For instance, the steady-state version only uses the speed of sound from one side at a time to calculate the next step, while the

$t_{\text{valve}} = 10$ s, $p_{\text{valve}} = 20$ bara							
t	p_{well}	p_{head}	p_{top}	V_{well}	V_{head}	V_{top}	$Q_{0,\text{top}}$
[s]		[bara]			[m/s]		[m³/s]
0	281.7	151.8	28.0	1.73	1.74	.0	.0
10	279.1	145.1	20.0	1.99	2.30	1.38	.173
20	277.3	145.3	20.0	2.14	2.17	1.24	.155
30	277.3	145.3	20.0	2.14	2.17	1.23	.155
40	277.3	145.3	20.0	2.14	2.17	1.23	.155
∞	277.1	145.3	20.0	2.12	2.14	1.22	.153
$t_{\text{valve}} = 20$ s, $p_{\text{valve}} = 20$ bara							
t	p_{well}	p_{head}	p_{top}	V_{well}	V_{head}	V_{top}	$Q_{0,\text{top}}$
[s]		[bara]			[m/s]		[m³/s]
0	281.7	151.8	28.0	1.73	1.75	.0	.0
10	281.9	151.4	24.0	1.75	1.82	1.12	.141
20	277.9	145.5	20.0	2.09	2.18	1.30	.163
30	277.3	145.3	20.0	2.14	2.17	1.24	.155
40	277.3	145.3	20.0	2.14	2.17	1.23	.155
∞	277.1	145.3	20.0	2.12	2.14	1.22	.153
$t_{\text{valve}} = 10$ s, $p_{\text{valve}} = 15$ bara							
t	p_{well}	p_{head}	p_{top}	V_{well}	V_{head}	V_{top}	$Q_{0,\text{top}}$
[s]		[bara]			[m/s]		[m³/s]
0	281.7	151.8	28.0	1.73	1.75	.0	.0
10	275.0	141.2	15.0	2.34	2.50	1.57	.199
20	274.1	140.5	15.0	2.41	2.44	1.39	.175
30	274.1	140.5	15.0	2.41	2.44	1.39	.175
40	274.1	140.5	15.0	2.41	2.44	1.39	.175
∞	273.9	140.5	15.0	2.38	2.41	1.37	.172
$t_{\text{valve}} = 20$ s, $p_{\text{valve}} = 15$ bara							
t	p_{well}	p_{head}	p_{top}	V_{well}	V_{head}	V_{top}	$Q_{0,\text{top}}$
[s]		[bara]			[m/s]		[m³/s]
0	281.7	151.8	28.0	1.73	1.75	.0	.0
10	280.8	146.4	21.5	1.84	2.24	1.37	.172
20	275.0	140.8	15.0	2.33	2.47	1.49	.187
30	274.1	140.5	15.0	2.41	2.44	1.39	.175
40	274.1	140.5	15.0	2.41	2.44	1.39	.175
∞	273.9	140.5	15.0	2.38	2.41	1.37	.172

Table 7.3: Results of single linear opening of top valve.

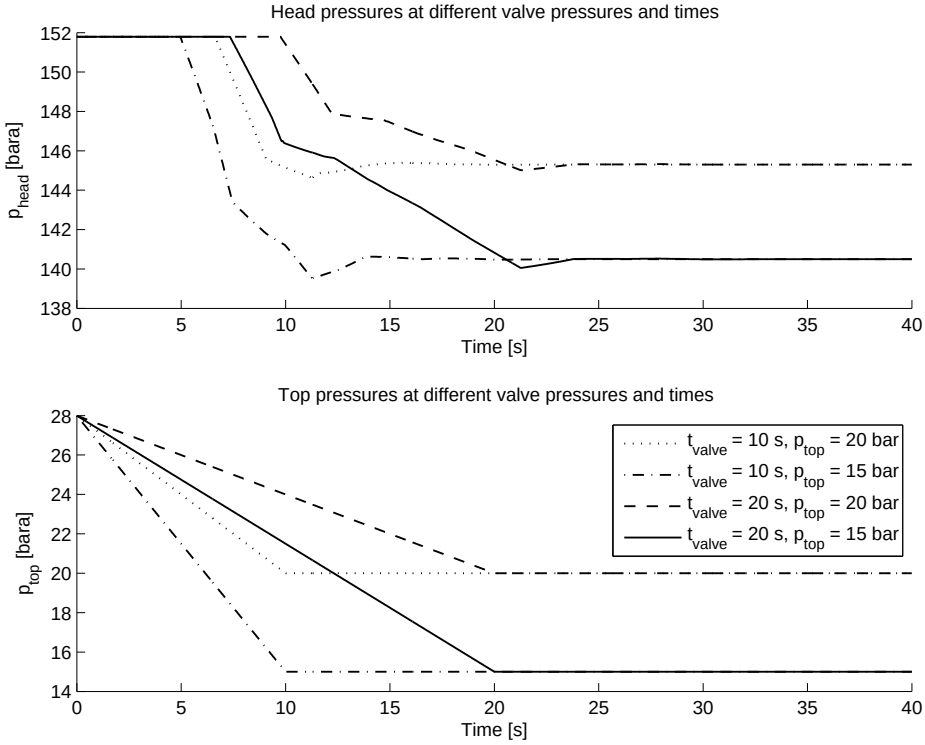

Figure 7.5: Pressures at transient start-up with linear reduction in p_{top}.

transient one uses both left and right values at the last time step to calculate the new values.

7.4.2 Velocity Peaks and Pressure Change

The velocity is shown in Fig. (7.6). The significant drop in velocity from wellhead to top is caused by the change in diameter. The flow Q_0 stays constant, as can be seen in Tab. (7.3).

Some peaks in the velocities can also be seen in all cases. The 10 s cases both have only one, which comes as a consequence of both the stop in valve opening and the start-up of the well. In the 20 s cases, however, these two come in separate waves. First, after around 10 s, the riser pressure drops below the wellhead pressure, and the two systems become one. Then, after 20 s, the valve stops opening, and riser velocity peaks, with wellhead velocity following an little

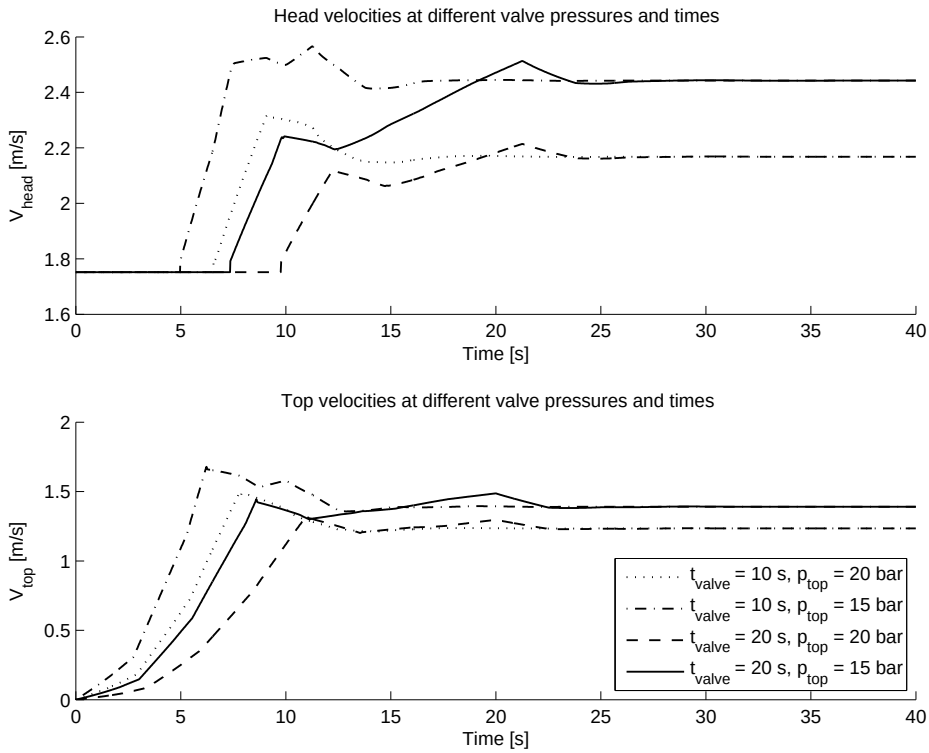

Figure 7.6: Velocities at transient start-up with linear reduction in p_{top}.

later.

Relative to the velocity after the peak, the velocity peaks at around 10 s seem to be the larger ones in all cases. For the 10 s, 15 bar case, for instance, the head velocity reaches a peak of 2.57 m/s, or 5.06 % higher than the end velocity. This happens after 11.3 s, or 1.3 s after the top valve is fully closed, just the time the pressure information needs to get from the valve to the wellhead (1,500 m @ $a \approx 1200$ m/s).

By comparison, for the same end pressure, but 20 s opening time, the velocity peaks at 2.51 m/s. That's just 2.90 % higher than the end velocity. The smallest wave occurs in the 20 bar, 20 s case, in which the peak velocity at wellhead is 2.21 m/s, or 2.12 % higher than after 40 s.

Fig. (7.7) shows the wave in the 10 s, 15 bar case. A high-velocity wave can be seen traveling from the top to the bottom. However, very little of this wave gets reflected back up at the well, since the standard and highly reflective

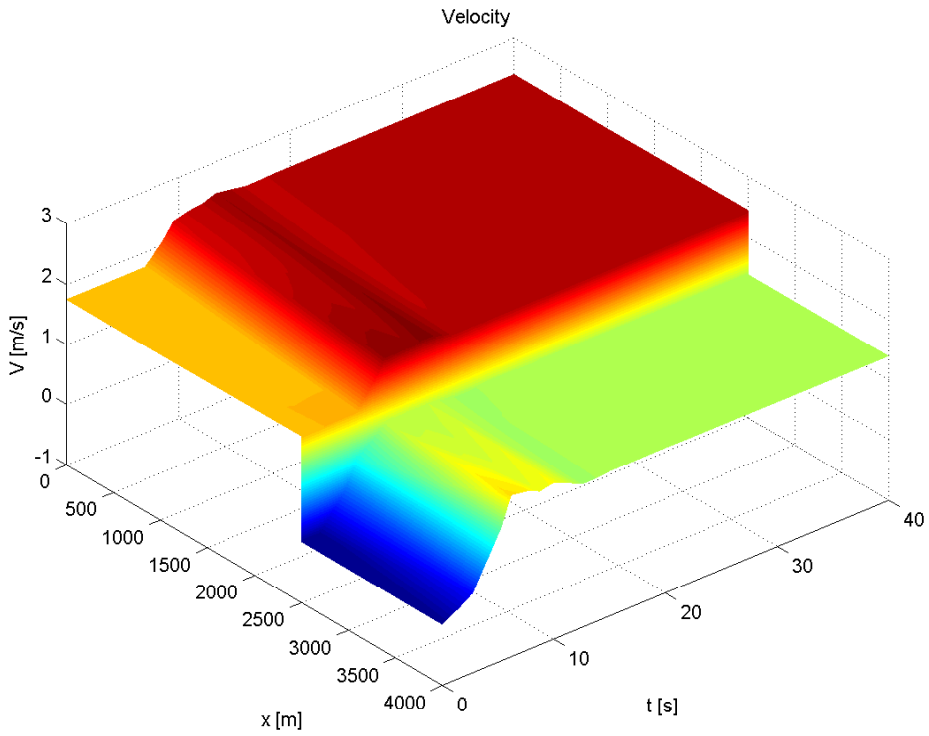

Figure 7.7: Surface plot of V for the 10 s, 15 bar transient start-up with linear reduction in p_{top}.

boundary condition in Eq. (5.45) has been replaced by Eq. (4.32), derived from the definition of the productivity index.

This plot also shows how the flow in the well is unaffected by the riser until the riser pressure drops below that of the sea bed. Also, the sharp edge in the velocity, caused by the diameter change between the well and the riser, is clearly seen. The simulation of this is very simplified – perhaps too much so. In a real scenario, a diameter change like this would both slow the flow somewhat, and also cause some reflections of pressure waves.

The reason why these peaks in velocity can be observed, is a change in the pressure gradient which after all is the driving force of the fluid. Both the connection of the well to the riser and the stop in valve opening causes a change in $\frac{\partial p}{\partial t}$, which in turn causes a change in the pressure gradient $\frac{\partial p}{\partial x}$.

The velocity peaks are therefore just a symptom of a non-optimal use of the

top valve. This behavior is unwanted, mainly because this is an operation taking place 1,500 m below sea level, with this valve being the only way to control it. The valve should therefore be tuned to act as predictable as possible, to ensure good system control.

7.4.3 Effect of Grid Variation

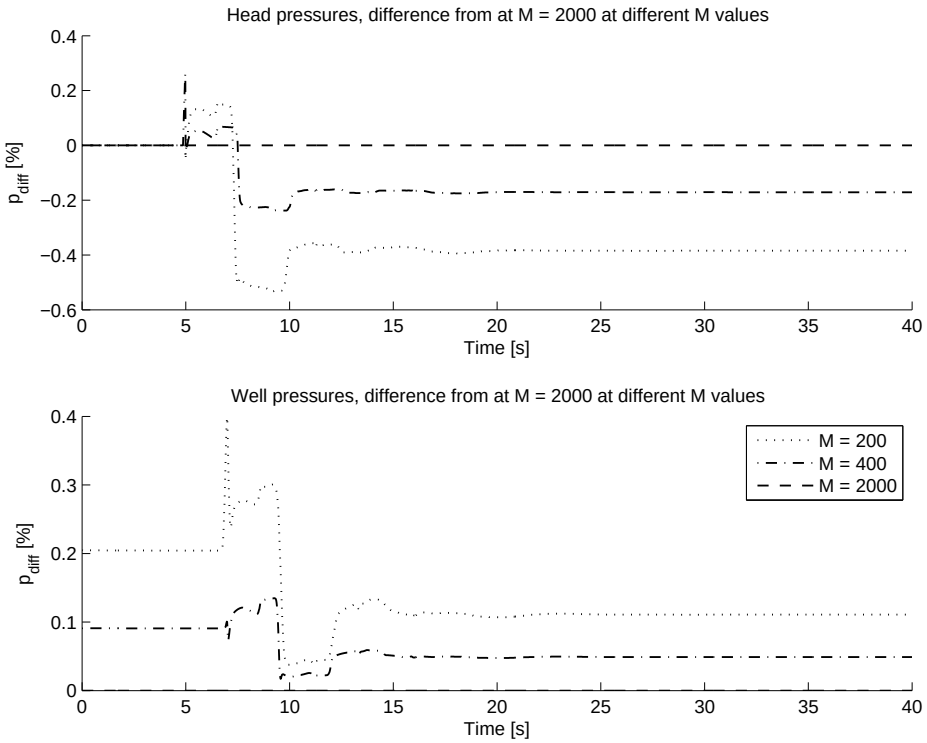

Figure 7.8: Pressures at transient start-up with linear reduction in p_{top}, comparison between different M values.

To see how much a fine grid plays in on the results, a test is done based on the data in the last section. Three tests are done, with three different values of M, the number of x nodes in each pipe stretch. The total number of x nodes is therefore 600, 1200 and 6000 in the respective cases. The relation between the number of x nodes and the number of time steps is also equal, so that there are 10,000, 20,000 and 100,000 time steps in the respective cases.

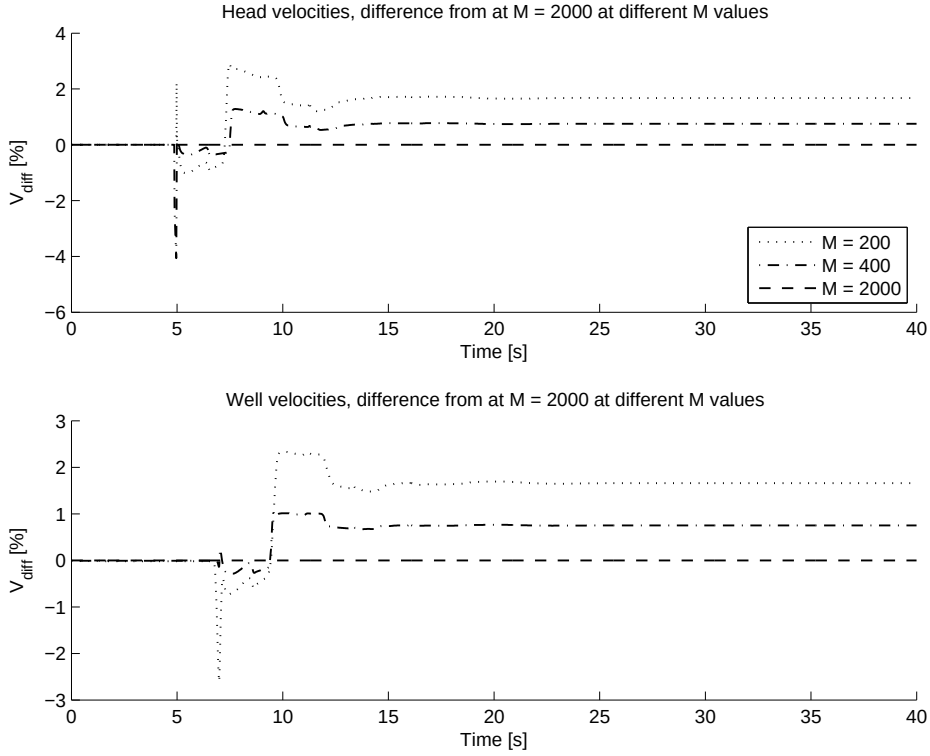

Figure 7.9: Velocities at transient start-up with linear reduction in p_{top}, comparison between different M values.

Looking at Figs. (7.8) and (7.9), it turns out that there are no very large instabilities along the time axis at a fixed point along the x axis, although there are some fluctuations between 5 and 15 s to within ± 4 % for velocity and $\pm .6$ % for pressure. There is also a steady-state variation in the series as they approach $t = 40$ s. This is true for both the velocity and the pressure.

The values at $M = 200$ seem to diverge a little more than twice as far from the $M = 2000$ values, compared to what the $M = 400$ values. This implies that there is still a slight margin in getting the results correct, but that they are converging strongly to the correct with $M = 2000$. After all, the number of nodes in x direction is only doubled from the first case to the second, but quintupled from the second to the third, with less error reduction in the second increase.

The grid can therefore be said to be quite accurate at $M = 2000$, at least for this simplified simulation. It can also be noted that the computer upon which

the simulations are done does not have enough memory to calculate any larger grid without some major recoding of the programs.

7.4.4 Double Linear Opening of Top Valve

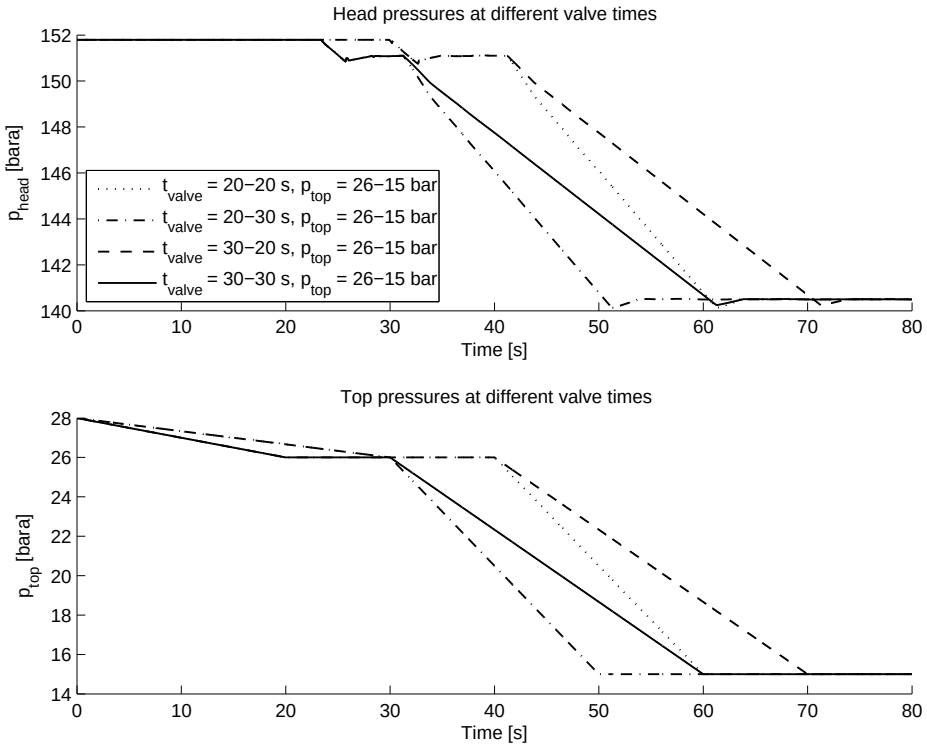

Figure 7.10: Pressures at transient start-up with double linear reduction in p_{top}.

Results so far have shown that a pressure surge occurs when the pressure at the bottom of the riser reaches the wellhead pressure. The best way to reduce this, seems to be a smaller reduction in top pressure, over a longer time. However, start-up to a high top pressure might just mean that the connection between the riser and the wellhead just isn't tight, so it isn't practical to reduce the valve to a higher end pressure.

A solution to this can be to decrease the top pressure in two turns. First, to start the riser flow to just below the balanced blowout flow rate from Tab. (7.1) and make this steady, and then to make a tight hydrodynamic seal. This will

$t_{\text{valve},1} = 20$ s, $t_{\text{valve},2} = 20$ s							
t	p_{well}	p_{head}	p_{top}	V_{well}	V_{head}	V_{top}	$Q_{0,\text{top}}$
[s]		[bara]			[m/s]		[m^3/s]
0	281.7	151.8	28.0	1.73	1.75	.0	.0
10	281.9	151.8	27.0	1.75	1.75	.11	.014
20	281.9	151.8	26.0	1.75	1.75	.78	.098
30	281.3	151.1	26.0	1.80	1.82	1.04	.130
40	278.4	146.1	20.5	2.05	2.16	1.30	.163
50	274.9	140.8	15.0	2.35	2.46	1.47	.185
60	274.1	140.5	15.0	2.41	2.44	1.39	.175
70	274.1	140.5	15.0	2.41	2.44	1.39	.175
80	274.1	140.5	15.0	2.41	2.44	1.39	.175
∞	273.9	140.5	15.0	2.38	2.41	1.37	.172
$t_{\text{valve},1} = 20$ s, $t_{\text{valve},2} = 30$ s							
t	p_{well}	p_{head}	p_{top}	V_{well}	V_{head}	V_{top}	$Q_{0,\text{top}}$
[s]		[bara]			[m/s]		[m^3/s]
0	281.7	151.8	28.0	1.73	1.75	.0	.0
10	281.9	151.8	27.3	1.75	1.75	.02	.003
20	281.9	151.8	26.7	1.75	1.75	.39	.049
30	281.9	151.8	26.0	1.75	1.79	1.03	.129
40	281.3	151.1	26.0	1.80	1.82	1.04	.131
50	278.4	146.1	20.5	2.05	2.16	1.30	.163
60	274.9	140.8	15.0	2.35	2.46	1.47	.185
70	274.1	140.5	15.0	2.41	2.44	1.39	.175
80	274.1	140.5	15.0	2.41	2.44	1.39	.175
∞	273.9	140.5	15.0	2.38	2.41	1.37	.172

Table 7.4: Results of double linear opening of top valve, part 1.

$t_{\text{valve},1} = 30$ s, $t_{\text{valve},2} = 20$ s							
t [s]	p_{well}	p_{head} [bara]	p_{top}	V_{well}	V_{head} [m/s]	V_{top}	$Q_{0,\text{top}}$ [m³/s]
0	281.7	151.8	28.0	1.73	1.75	.0	.0
10	281.9	151.8	27.0	1.75	1.75	.11	.014
20	281.9	151.8	26.0	1.75	1.75	.78	.098
30	281.3	151.1	26.0	1.80	1.82	1.04	.130
40	279.4	147.7	22.3	1.97	2.05	1.21	.152
50	277.0	144.2	18.7	2.17	2.26	1.33	.167
60	274.6	140.7	15.0	2.37	2.46	1.44	.182
70	274.1	140.5	15.0	2.41	2.44	1.39	.175
80	274.1	140.5	15.0	2.41	2.44	1.39	.175
∞	273.9	140.5	15.0	2.38	2.41	1.37	.172
$t_{\text{valve},1} = 30$ s, $t_{\text{valve},2} = 30$ s							
t [s]	p_{well}	p_{head} [bara]	p_{top}	V_{well}	V_{head} [m/s]	V_{top}	$Q_{0,\text{top}}$ [m³/s]
0	281.7	151.8	28.0	1.73	1.75	.0	.0
10	281.9	151.8	27.3	1.75	1.75	.02	.003
20	281.9	151.8	26.7	1.75	1.75	.39	.049
30	281.9	151.8	26.0	1.75	1.79	1.03	.129
40	281.3	151.1	26.0	1.80	1.82	1.04	.131
50	279.4	147.7	22.3	1.97	2.05	1.21	.152
60	277.0	144.2	18.7	2.17	2.26	1.33	.167
70	274.6	140.7	15.0	2.37	2.46	1.44	.182
80	274.1	140.5	15.0	2.41	2.44	1.39	.175
∞	273.9	140.5	15.0	2.38	2.41	1.37	.172

Table 7.5: Results of double linear opening of top valve, part 2.

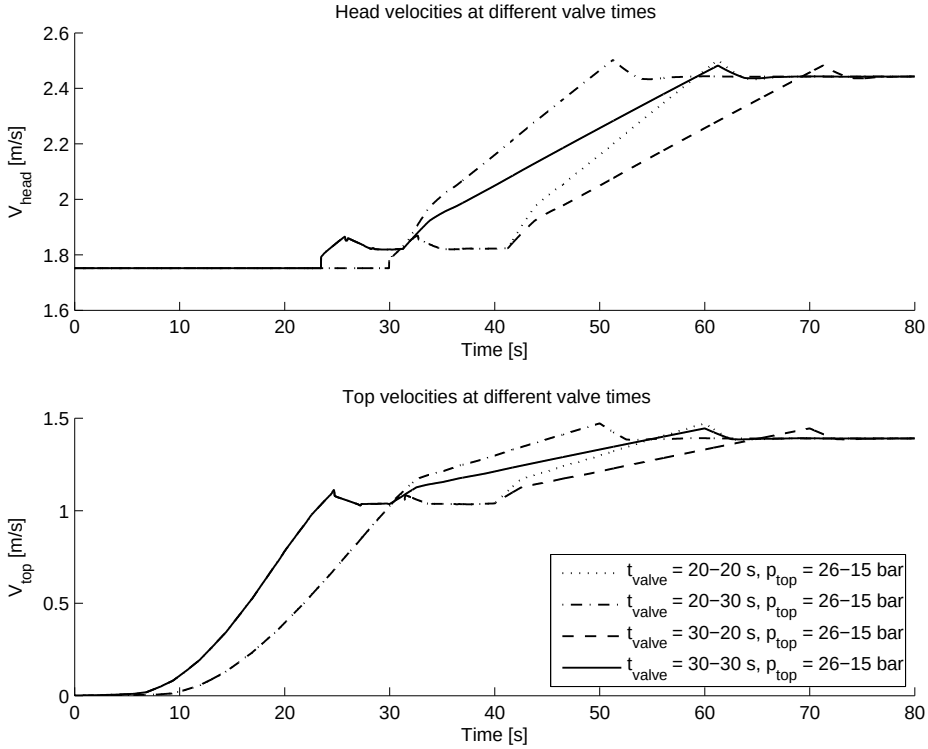

Figure 7.11: Velocities at transient start-up with double linear reduction in p_{top}.

hopefully make the whole start-up go much more controlled, and remove much of the unpredictable pressure behavior.

If the valve first opens a small amount to an end pressure of 26 bara over 20 to 30 s, then stops for 10 s to let the flow settle, and then resumes to open slowly for another 20 to 30 s to the desired top pressure (in this case 15 bara), the high-velocity waves might just disappear. Four tests are done, to see how the duration of each valve opening affects the flow.

The simulations are again done with 6,000 nodes in x direction ($M = 2000$), but this time over a time period of 80 s. This leads to the need of more time steps, 200,000 all in all. The four calculations took about one hour each, leading to a total calculation time of 4 hours.

The result of this very slow opening of the top valve to just below the p_{top} at blowout, is shown in Figs. (7.10) and (7.11) and in Tabs. (7.4) and (7.5).

Still, a small wave of higher velocity occurs at both points where the valve

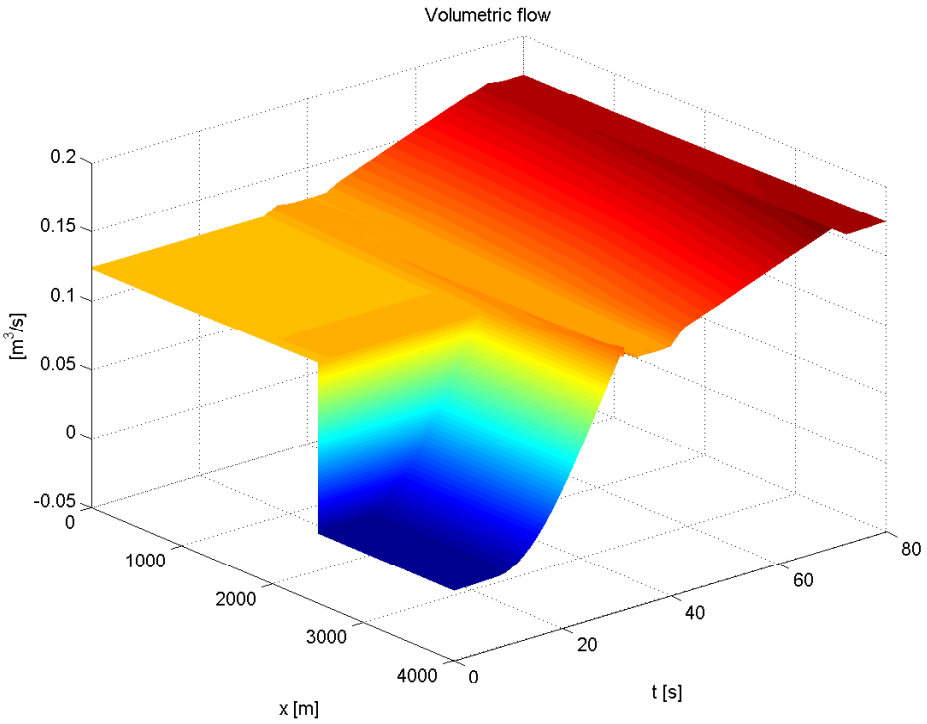

Figure 7.12: Velocities at transient start-up with double linear reduction in p_{top}.

stops opening. Even when both valve operations take 30 s each, a velocity peak appears, unnoticeably lower than at the 20 s opening times. This can also be seen in Fig. (7.12), where at both instances the waves travel to the bottom for then to disappear.

A pattern emerges, and it seems that the opening procedure of linearly lowering the pressure might not be the best way to open the valve. A non-linear opening procedure is therefore proposed.

7.4.5 Nonlinear Opening of Top Valve

The problem with the linear opening of the top valve all seem to come at the end of each opening cycle, no matter how carefully it is opened. To address this problem, then, might be to make the valve opening model nonlinear, and a tanh function is proposed. This will hopefully cause less change in the pressure gradients, an make the start-up more controlled.

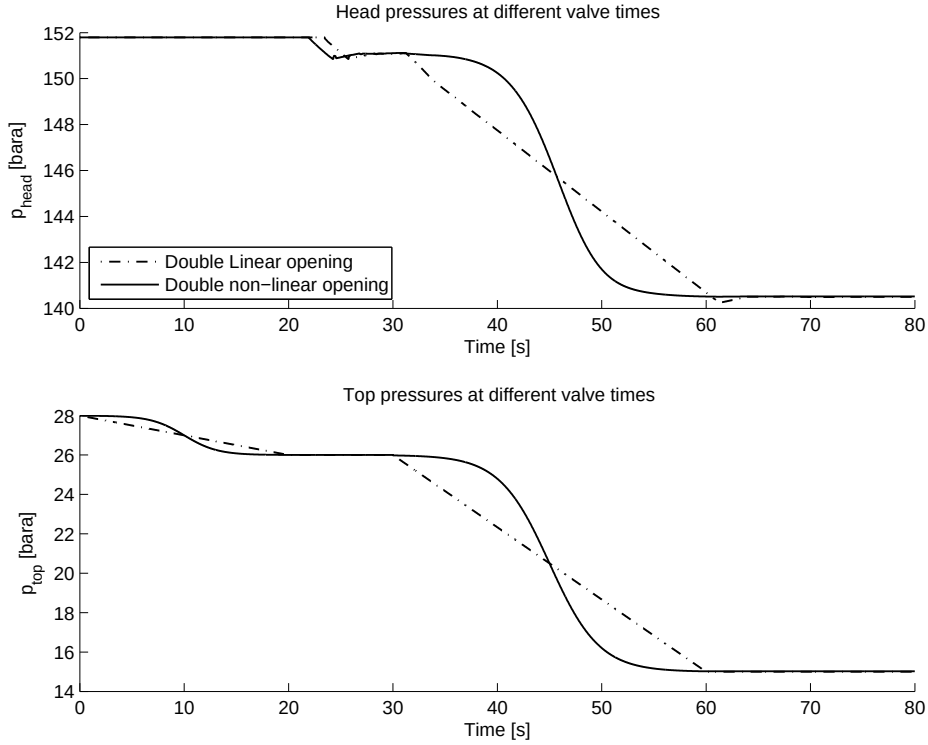

Figure 7.13: Pressures at transient start-up with double nonlinear reduction in p_{top}.

To implement this, the following relation is proposed to handle the pressure drop during valve opening,

$$p\left(t\right) = p_0 - \frac{p_{\text{valve}}}{2} \cdot \left[\tanh\left(2\pi \cdot \frac{t}{t_{\text{valve}}} - \pi\right) + 1\right] \tag{7.4}$$

in which p_0 is the pressure when the closing process starts. This is implemented in the MATLAB program `transient.m`, and only one simulation is done, in which the pressure is first dropped to 26 bara over 20 s, and then dropped to 15 bara over 30 s, similar to one of the cases in Sec. 7.4.4. The simulated time is 80 s, with a grid consisting of 6,000 nodes in x direction and 200,000 timesteps. Simulation time was about one hour.

Results are shown in Figs. (7.13) and (7.14), where they are also compared to

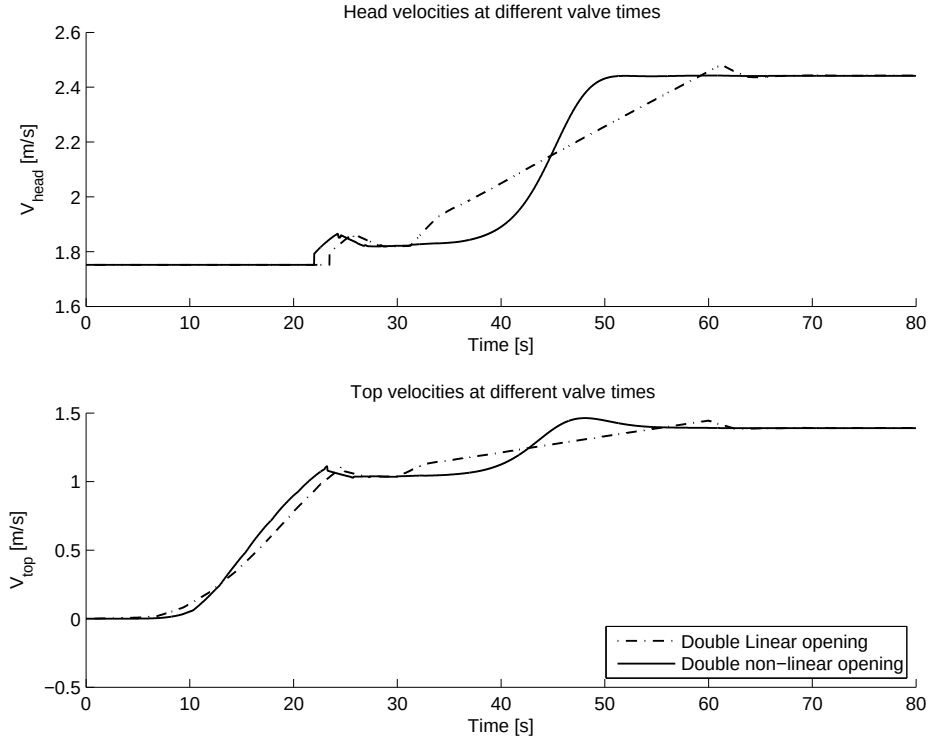

Figure 7.14: Velocities at transient start-up with double nonlinear reduction in p_{top}.

the corresponding linear case from Sec. 7.4.4. It seems at least the last opening of the valve turned out as hoped for. There are no dips in end pressure at ≈ 60 s, and the velocity also handles quite predictable. However, there is a small rise in top velocity between 45 and 50 s, but a rise in velocity at this end is not that critical.

All in all, then, it seems as though a hyperbolic curve is a good way to reduce the pressure, once the two systems are merged.

The first opening, on the other hand, still behaves a bit uncontrolled. The reason for this may be spotted in Fig. (7.15), where it can be seen that the riser velocity is still on the rise at the point where it gets connected to the well. For the well, this leads to a sudden exposure to the pressure gradient in the riser, which in turn causes a sudden pressure dip in the well of approximately 1 bar.

Some tuning of the first valve is therefore proposed, to find out if a nonlinear

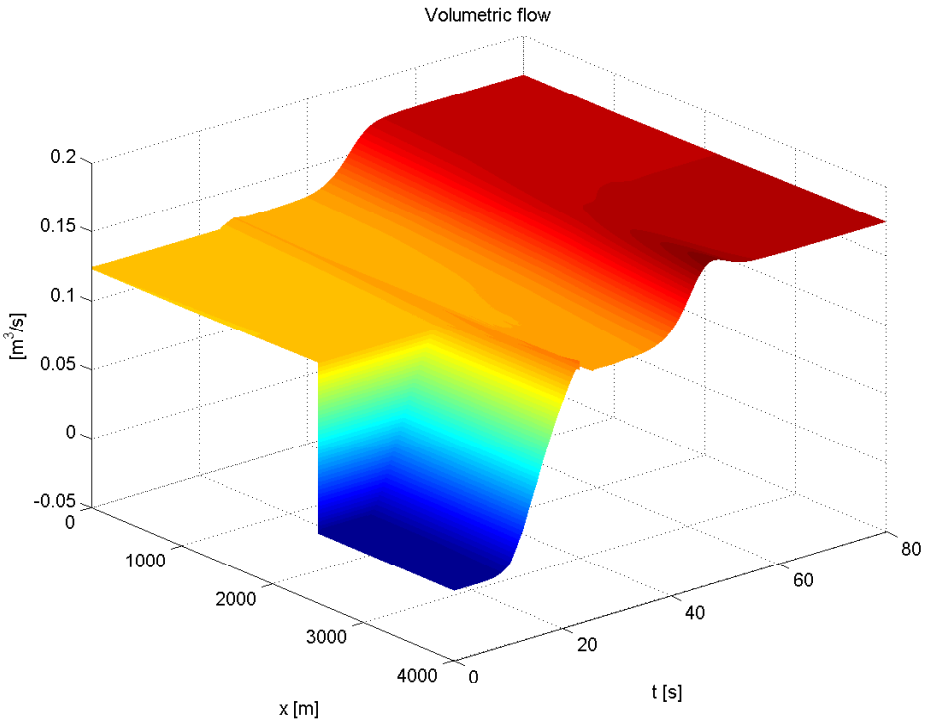

Figure 7.15: Velocities at transient start-up with double nonlinear reduction in p_{top}.

function may still make the first opening procedure smooth.

7.4.6 Nonlinear Opening – Fine Tuning of the 1st Opening

It turns out that the first opening of the valve is still not quite optimal, and it seems that even the small difference in p_{head} of 0.7 bar between the steady-state case of 26.7 bar in Tab. (7.1) to the 26 bar case in Sec. 7.4.5 is enough to still cause a pressure wave upon connection to between the riser capture dome and the wellhead.

A fine-tuning of the intermediate pressure between the two valve openings is therefore in order. In this simulation, only the first opening procedure is simulated, with valve opening time of 30 s and simulation time of 60 s, with 150,000 time steps. Five different intermediate pressures are tested, ranging from 26.5 to 26.9 bar.

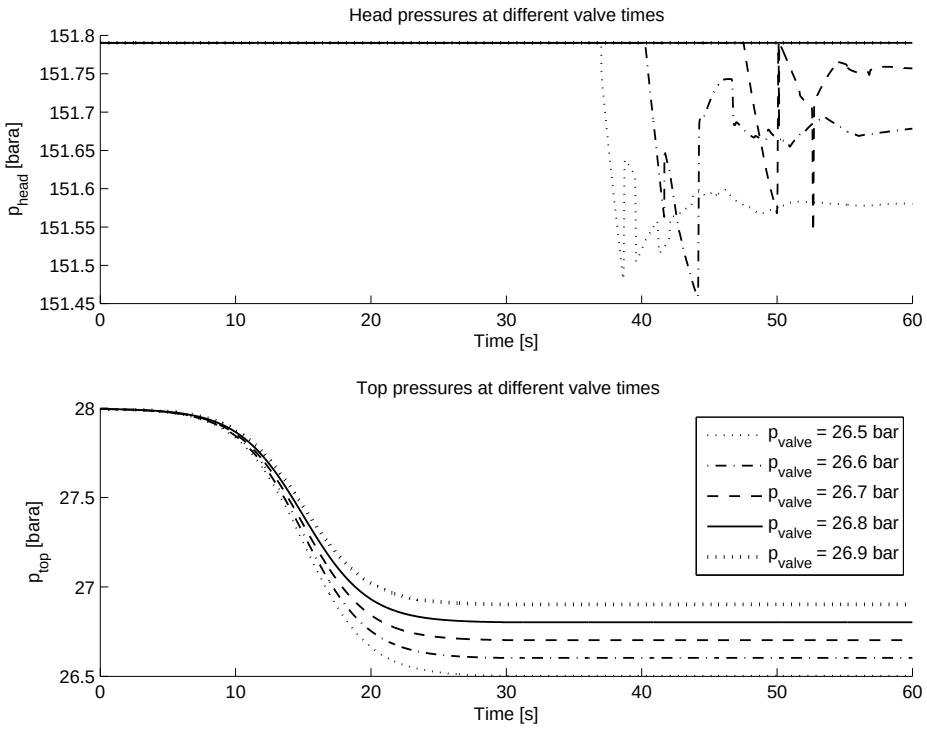

Figure 7.16: Pressures when adjusting the first nonlinear reduction in p_{top}.

The results shown in Figs. (7.16) and (7.17) all seem a bit unstable at the head. This is only natural when zooming in on a small pressure reduction like this. However, the top graphs lie a bit, as the pressure does not fluctuate by more than 0.3 bar, and the velocity does not fluctuate by more than 0.06 m/s, so it is not that critical. This may also be a symptom of getting close to the accuracy of the current grid, so further attempts to make the start-up smoother are most likely impractical.

The main lesson learned from this, however, is that the fluctuations are very much reduced by lowering the pressure first to the balanced blowout pressure from Tab. (7.1), and that a calculation of this can prove useful in finding the optimal opening procedure. This is backed mainly by that the lowering in pressure to the actual value of the balanced top pressure of 26.7 bar, gives the smallest fluctuations of all cases, as seen in the figures.

It may also be seen that some of the curves – especially the top velocity do

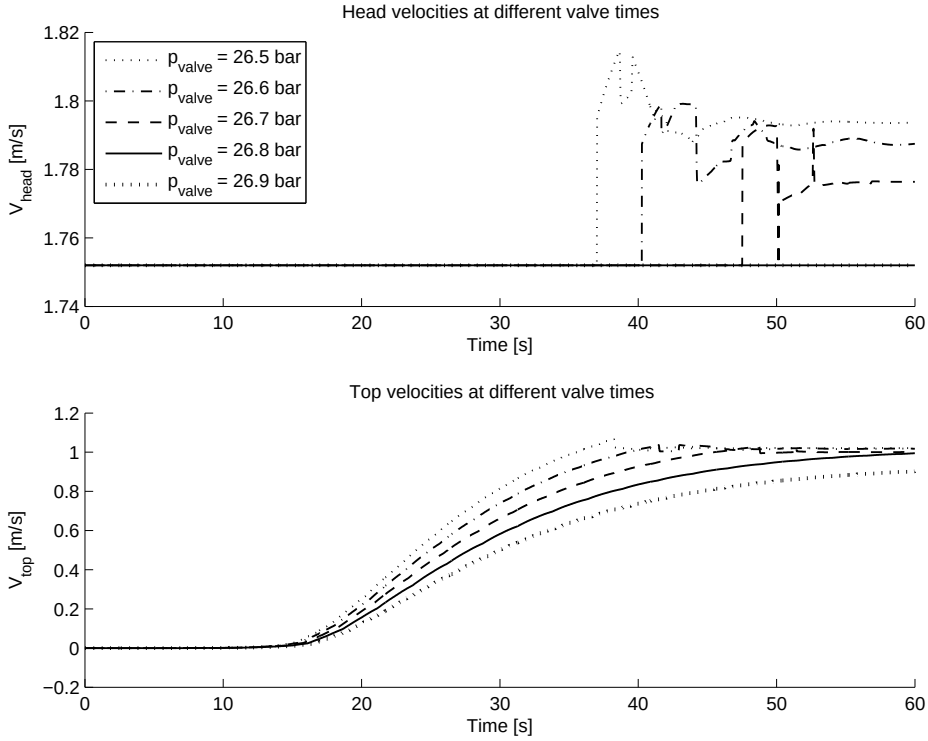

Figure 7.17: Velocities when adjusting the first nonlinear reduction in p_{top}.

not seem to converge entirely within the time limit portrayed here. However, none of them seem to be low enough to start the well, even when extending the simulation in time (not shown).

8 Conclusion

With focus on the start-up procedure, both steady-state and transient analyses have been made of the flow in a well at blowout and a vertical riser intended to contain and collect the spilled oil. All of this has been done as a preliminary study of whether the method would work or not, with the main focus on finding a smooth way to start the system up.

Firstly, it is worth noting that many factors here are extremely rough estimates based available data from the «Deepwater Horizon»/Macondo spill. The productivity index PI is among these, based on an estimated reservoir pressure p_{res}, well diameter D_{well}, depth and roughness ϵ and oil density ρ_{oil}, speed of sound a, and flow rate Q. This is in turn based on API gravity and temperature T from the top hat #4 data, and the bulk modulus of elasticity K calculated from these. Many joints leading up to the result, in other words.

This of course makes the simulation unmatchable with the real spill case in the Gulf of Mexico in 2010, but this was never the intention. The goal was only to make a simple model based on a semi-real well, so a simulation with some sort of root in reality was possible. Anyway, for a real application, the operator of a leaking oil well will have all the data needed to make a good simulation of the well.

Furthermore, the hydrodynamic seal between the sea floor and the collection dome is not thoroughly simulated. It is only assumed friction-free and to have the same diameter as the rest of the riser. Also, the joint between the two pipe diameters is non-reflective and friction-free. However, this also takes away some of the dampening in the system, so that a smooth start-up may actually be made harder by it.

The simulations in Sec. 7.1 showed that a well diameter of 300 mm was the most probable, while Sec. 5.2 showed that a riser diameter of 400 mm was the optimal choice, with lower loss than at 300 mm, but with roughly the same loss as at 500 mm.

The transient analyses showed that a non-linear, slow opening of the top valve in two steps was the best bet for a smooth start-up. The first of these should be opened to the pressure where the pressure at the capture dome equals the pressure in the wellhead.

8.0.7 Further Work

The simulations are done on a very simplified model of the system. It should be improved, to include all equipment actually needed at sea floor, as well as a better model of the leaking well itself.

The most important unknown factor in this report has been the seal between the capture dome and the sea floor. This should be properly simulated – and for

real-world application, it should also undergo scale testing.

Furthermore, an improved system model should be simulated as a multiphase flow. This will be much more realistic, as both water and gas will flow together with the oil in an actual spill.

References

[1] Bob Graham and William K. Reilly. Deep water – the Gulf oil disaster and the future of offshore drilling – report to the President. Washington, D.C, 2011.

[2] Kystverket. Ett år siden «Full City»- havariet, `http://www.kystverket.no/Default.aspx?did=10092886`, 2010.

[3] Timothy J. Crone and Maya Tolstoy. Magnitude of the 2010 Gulf of Mexico Oil Leak. *Science*, October 2010.

[4] Frank White. *Fluid Mechanics*. McGraw-Hill, New York, 2003.

[5] United States Department of Energy. Deepwater Horizon Resonse, `http://www.energy.gov/open/oilspilldata.htm`, 2010.

[6] R.J. Wostl et al. Velocimeter measures bulk moduli. *Oil and Gas Journal*, December 1970.

[7] E. Benjamin Wylie and Victor L. Streeter. *Fluid Transients in Systems*. Prentice Hall, Englewood Cliffs, 1993.

[8] Norwegian Piping AS. Pipe schedule & weights, `http://www.norwegianpiping.com/bilder/filer/PipeSchedulerev2.pdf`, 2011.

[9] Gordon Aylward and Tristan Finlay. *SI chemical data*. Wiley, Chichester, 2002.

A MATLAB Code

The MATLAB codes are sorted by family, so that any plotting program or initial data functions etc. are sorted under the main programs.

A.1 storeinitdata.m

```
% 0————————————————————————————————————————————————————————————————————————0
% | Program storeinitdata.m                                                    |
% | Used to store various constants used by the other MATLAB programs.         |
% | Saves results in:                                                          |
% |     initdata.mat                                                           |
% |                                                                            |
% | Coded by:                                                                  |
% |     Stud. Techn. Eivind Vrålstad,                                          |
% |     Institutt for Energi— og prosessteknikk, NTNU, 2011                    |
% 0————————————————————————————————————————————————————————————————————————0

clear all
clc

% General constants
g = 9.80665;

% Grid
M = 2000;

% Oil
oilrho_0 = 840;
oilp_0 = 101.3E3;
oila_0 = 1315;
oilK_0 = 1.45E9;
oilmu_0 = .07;



% Sea water
searho_0 = 1025;
seaa_0 = 1535;
seaK_0 = 2.42E9;
seamu_0 = 1E—3;



% Riser
riserH = 1500;
```

```
% Well
wellL = 1000;
wellH = 1500;



% Steel
steelE = 207.0E9;    % Pa
steeleps = 2E-3;     % m
steelmbyL = 123.3;   % kg/m
steele = 12.7E-3;    % m



% Saves results in initdata.mat
save('initdata', ...
    'oilrho_0','oilK_0','oila_0','oilmu_0','oilp_0', ...
    'searho_0','seaK_0','seaa_0','seamu_0', ...
    'riserH', ...
    'wellH','wellL', ...
    'steelE','steeleps','steelmbyL','steele', ...
    'M','g')

disp('Fluid data stored')
```

A.2 blowout.m

```
% 0-------------------------------------------------------------------------------0
% | Program blowout.m                                                           |
% | Used to find properties of flow in the system at blowout conditions.        |
% | Uses data in:                                                               |
% |     initdata.mat                                                            |
% | Saves results in:                                                           |
% |     blowout.mat                                                             |
% | Saves figures in:                                                           |
% |     p_p_diff_blowout.eps                                                    |
% |     V_V_diff_blowout.eps                                                    |
% |                                                                             |
% | Coded by:                                                                   |
% |     Stud. Techn. Eivind Vrålstad,                                           |
% |     Institutt for Energi- og prosessteknikk, NTNU, 2011                     |
% 0-------------------------------------------------------------------------------0

clear all
clc

tol = 1E-14;
maxit = 1000;

load('initdata')



% 0-------------------------------------------------------------------------------0
% | Properties of water                                                         |
% 0-------------------------------------------------------------------------------0




% 0-------------------------------------------------------------------------------0
% | Properties of oil                                                           |
% 0-------------------------------------------------------------------------------0

oilp_0 = 101.3E3;



% 0-------------------------------------------------------------------------------0
% | Various flow properties                                                     |
% 0-------------------------------------------------------------------------------0

Q_bbl = 68000; % bbl/day
Q_0 = bbl_to_m3(Q_bbl)/3600/24; % m^3/s
m = Q_0 * oilrho_0; % kg/s
```

```
p_r = searho_0*g*(wellH+riserH) + oilp_0;    % Reservoir pressure [Pa]
p_h = searho_0*g*riserH + oilp_0;            % Wellhead pressure [Pa]


%0-------------------------------------------------------------------------0
%| Defining the grid                                                      |
%0-------------------------------------------------------------------------0

M3 = M*3+1;

dx1 = wellL/M;
x1 = (0:dx1:wellL)';

dx2 = wellH/M;
x2 = ((wellL+dx2):dx2:(wellL+wellH))';

dx3 = riserH/M;
x3 = ((wellL+wellH+dx3):dx3:(wellL+wellH+riserH))';

x = [x1;x2;x3];



%0-------------------------------------------------------------------------0
%| Setting constants that vary through the pipe                           |
%0-------------------------------------------------------------------------0

sinalpha = [zeros(M,1);ones(2*M+1,1)];
D = [.2 .3 .4 .5];
A_0 = pi/4*D.^2;
H = [zeros(M+1,1);(1:M)'*dx2;(M+1:2*M)'*dx3];



%0-------------------------------------------------------------------------0
%| Setting initial values                                                 |
%0-------------------------------------------------------------------------0

p_P = p_h;
rho_P = oilrho_0*exp((p_P-oilp_0)/oilK_0);
A_P = A_0*oilrho_0./rho_P.*exp(oilK_0/oila_0^2*(1/oilrho_0-1./rho_P));
Q_P = Q_0*oilrho_0./rho_P;
V_P = Q_P./A_P;


%0-------------------------------------------------------------------------0
%| Defining the grid                                                      |
%0-------------------------------------------------------------------------0

V = zeros(M3,4);
rho = zeros(M3,4);
```

```
p = zeros(M3,4);
V(2*M,:) = V_P;
p(2*M,:) = p_P;
rho(2*M,:) = rho_P;



%0————————————————————————————————————————————————————————————————————————0
%| Running sequence from sea bed to bottom of well (upstream)                  |
%0————————————————————————————————————————————————————————————————————————0

for m = 2*M-1:-1:1
        rho_S = rho_P;
        V_S = V_P;
        a_S = sqrt(oilK_0./rho_S./(1+oilK_0/steelE*D/steele));
        s = 2*g./a_S.^2;
        Re_D = abs(rho_S.*V_S.*D/oilmu_0);
        f = (-1.8 * log10(6.9./Re_D ...
            + (steeleps./D/3.7).^1.11)).^-2;

        V_P = V_S ...
            ./(exp(s*(H(m+1,:)-H(m,:))/2) ...
            + f.*V_S.*abs(V_S)*(x(m+1,:)-x(m,:))./(2*D.*a_S.^2));
        V(m,:) = V_P;
        rho_P = 1./(1./rho_S ...
            - (g*(H(m+1,:)-H(m,:)) ...
            + f.*V_P.*abs(V_S)/2./D.*(x(m+1,:)-x(m,:))) ...
            / oilK_0);
        rho(m,:) = rho_P;
        p(m,:) = oilp_0 + oilK_0*reallog(rho_P/oilrho_0);
end


V_P = V(2*M,:);
p_P = p(2*M,:);
rho_P = rho(2*M,:);

%0————————————————————————————————————————————————————————————————————————0
%| Running sequence from sea bed to top of riser (downstream)                  |
%0————————————————————————————————————————————————————————————————————————0

for m = 2*M+1:M3
    rho_R = rho_P;
    V_R = V_P;
    a_R = sqrt(oilK_0./rho_R./(1+oilK_0/steelE*D/steele));
    s = 2*g./a_R.^2;
    Re_D = abs(rho_R.*V_R.*D/oilmu_0);
    f = (-1.8 * log10(6.9./Re_D ...
        + (steeleps./D/3.7).^1.11)).^-2;

    V_P = V_R ...
        ./(exp(-s*(H(m,:)-H(m-1,:))/2) ...
```

```
        - f.*V_R.*abs(V_R)*(x(m,:)-x(m-1,:))./(2*D.*a_R.^2));
    V(m,:) = V_P;
    rho_P = 1./(1./rho_R ...
        + (g*(H(m,:)-H(m-1,:)) ...
        + f.*V_P.*abs(V_R)/2./D.*(x(m,:)-x(m-1,:))) ...
        / oilK_0);
    rho(m,:) = rho_P;
    p(m,:) = oilp_0 + oilK_0*reallog(rho_P/oilrho_0);
end



%0-------------------------------------------------------------------------------0
%| Finding p_w and PI for the four cases;                                        |
%| 1. Q1 D1                                                                      |
%| 2. Q1 D2                                                                      |
%| 3. Q2 D1                                                                      |
%| 4. Q2 D2                                                                      |
%0-------------------------------------------------------------------------------0

p_w = p(1,:);            % Well pressure [Pa]
PI = Q_0./(p_r-p_w);       % Productivity index [m^3/Pa s]
F_b = p(M3,:)*pi/4.*D.^2;



%0-------------------------------------------------------------------------------0
%| Displaying results on-screen                                                  |
%0-------------------------------------------------------------------------------0


disp('Case 0')
disp(' ')
disp('V_h = 0 m/s')
disp(['p_w = ' num2str(p_r/1E5) ' bar'])
disp(['p_t = ' num2str((searho_0-oilrho_0)*g*(riserH)/1E5) ' bar'])
disp('Q_0 = 0 m^3/s')
disp('PI = 0 m^3/bar s')
disp('Q_US = 0 bbl/d')
disp('PI_US = 0 bbl/psi d')
% disp(['F_buoy = ' num2str((searho_0-oilrho_0)*g*(riserH)*pi/4*riserD.^2/1E3) ' kN'])
disp(' ')
disp(' ')


for i=1:4
    disp(['Case ' num2str(i)])
    disp(' ')
    disp(['p_well = ' num2str(p(1,i)/1E5) ' bar'])
    disp(['p_head = ' num2str(p(2*M,i)/1E5) ' bar'])
    disp(['p_top = ' num2str(p(M3,i)/1E5) ' bar'])
    disp(['V_well = ' num2str(V(1,i)) ' m/s'])
```

```
    disp(['V_head = ' num2str(V(2*M,i)) ' m/s'])
    disp(['V_top = ' num2str(V(M3,i)) ' m/s'])
    disp(' ')
    disp(['D = ' num2str(D(i)) ' m'])
    disp(['Q_0 = ' num2str(Q_0) ' m^3/s'])
    disp(['PI = ' num2str(PI(i)*1E5) ' m^3/bar s'])
    disp(['Q_US = ' num2str(Q_bbl) ' bbl/d'])
    disp(['F_buoy = ' num2str(F_b(i)/1E3) ' kN'])
    disp(' ')
    disp(' ')
end


p_hydrostat = ones(M3,1)*p_w ...
    - oilrho_0*g*(H*ones(1,4));

V_bernoulli = sqrt(ones(M3,1)*(Q_0./A_0).^2 ...
    - 2*g.*H*ones(1,4) ...
    + ones(M3,1)*p_w*2/oilrho_0 ...
    - 2*p_hydrostat/oilrho_0);

p_diff_h = (p_hydrostat-p)/1E5;
V_diff_b = (V_bernoulli-V)./V*100;

x_blowout = x;
V_blowout = V;
rho_blowout = rho;
p_blowout = p;
Q_blowout_0 = Q_0;



%0-------------------------------------------------------------------------0
%| Saving results in blowout.mat                                          |
%0-------------------------------------------------------------------------0

save('blowout','x_blowout','V_blowout','rho_blowout','p_blowout','Q_blowout_0','PI')



%0-------------------------------------------------------------------------0
%| Plotting pressure results and saving to p_p_diff_blowout.eps           |
%0-------------------------------------------------------------------------0

figure(1)
subplot(2,1,1); plot(x,p(:,1)/1E5,'k', ...
    x,p(:,2)/1E5,'k--', ...
    x,p(:,3)/1E5,'k:', ...
    x,p(:,4)/1E5,'k-.')
title('Pressure')
xlabel('L [m]')
ylabel('p [bar]')
```

```
legend(['D = ' num2str(D(1,1)*1E3) ' mm'], ...
    ['D = ' num2str(D(1,2)*1E3) ' mm'], ...
    ['D = ' num2str(D(1,3)*1E3) ' mm'], ...
    ['D = ' num2str(D(1,4)*1E3) ' mm'])

subplot(2,1,2); plot(x,p_diff_h(:,1),'k', ...
    x,p_diff_h(:,2),'k--', ...
    x,p_diff_h(:,3),'k:', ...
    x,p_diff_h(:,4),'k-.')
title('Pressure, Difference From Hydrostatic')
xlabel('L [m]')
ylabel('p_{diff} [bar]')
print -deps plots/p_p_diff_blowout



%0-------------------------------------------------------------------------------0
%| Plotting velocity results and saving to V_V_diff_blowout.eps                  |
%0-------------------------------------------------------------------------------0

figure(2)
subplot(2,1,1); plot(x,V(:,1),'k', ...
    x,V(:,2),'k--', ...
    x,V(:,3),'k:', ...
    x,V(:,4),'k-.')
title('Velocity')
xlabel('L [m]')
ylabel('V [m/s]')
legend(['D = ' num2str(D(1,1)*1E3) ' mm'], ...
    ['D = ' num2str(D(1,2)*1E3) ' mm'], ...
    ['D = ' num2str(D(1,3)*1E3) ' mm'], ...
    ['D = ' num2str(D(1,4)*1E3) ' mm'])


subplot(2,1,2); plot(x,V_diff_b(:,1),'k', ...
    x,V_diff_b(:,2),'k--', ...
    x,V_diff_b(:,3),'k:', ...
    x,V_diff_b(:,4),'k-.')
title('Velocity, Difference from Frictionless Bernoulli')
xlabel('L [m]')
ylabel('V_{diff} [%]')
print -deps plots/V_V_diff_blowout
```

A.3 steadystate.m

```
% 0-------------------------------------------------------------------------0
% | Program ssD400.m                                                        |
% | Used to find properties of flow in the system at steady-state           |
% | conditions and D = 400 mm.                                              |
% | Uses data from:                                                         |
% |     blowout.mat                                                         |
% |     initdata.mat                                                        |
% | Saves figures in:                                                       |
% |     V_V_diff_ss_D400.eps                                                |
% |     p_p_diff_ss_D400.eps                                                |
% |                                                                         |
% | Coded by:                                                               |
% |     Stud. Techn. Eivind Vrålstad,                                       |
% |     Institutt for Energi- og prosessteknikk, NTNU, 2011                 |
% 0-------------------------------------------------------------------------0

clear all
clc

tol = 1E-5;
maxit = 50;



% 0-------------------------------------------------------------------------0
% | Loading data from blowout.m, stored in blowout.mat                      |
% 0-------------------------------------------------------------------------0

load('../blowout')
load('../initdata')



% Properties of oil

oilp_0 = 101.3E3;


% Properties of well

wellD = .3;             % m

% Properties of riser

riserD = .4;


% Flow properties
```

```
p_r = searho_0*g*(wellH+riserH) + oilp_0;    % Reservoir pressure [Pa]


% Setting the grid

M3 = M*3+1;

dx1 = wellL/M;
x1 = (0:dx1:wellL)';

dx2 = wellH/M;
x2 = (wellL+dx2:dx2:(wellL+wellH))';

dx3 = riserH/M;
x3 = ((wellL+wellH)+dx3:dx3:(wellL+wellH+riserH))';

x = [x1;x2;x3];


% Setting constants that have different values through the pipe

sinalpha = [zeros(M,1);ones(2*M+1,1)];
D = [ones(2*M,1)*wellD;ones(M+1,1)*riserD];
A_0 = pi/4*D.^2;
H = sinalpha.*(x-x(M+1,1));


p_t = [20 15 10 5]*1E5;
p = zeros(M3,4);
V = zeros(M3,4);
rho = zeros(M3,4);


for i = 1:4
    % Setting boundary values

    p_P = p_t(1,i);              % Top pressure [Pa]
    rho_P = oilrho_0*exp((p_P-oilp_0)/oilK_0);    % Top density [kg/m^3]
    V_P = 5;                % First guess velocity [m/s]

    V_new = zeros(M3,1);
    rho_new = zeros(M3,1);
    p_new = zeros(M3,1);
    V_new(M3) = V_P;
    p_new(M3) = p_P;
    rho_new(M3) = rho_P;

    err = inf;
    it = 0;
    while err>=tol && it<maxit
        for m = M3-1:-1:1
```

```
    rho_S = rho_new(m+1);
    V_S = V_new(m+1)*A_0(m+1)/A_0(m);
    a_S = sqrt(oilK_0/rho_S/(1+oilK_0/steelE*D(m+1)/steele));
    s = 2*g/a_S^2;
    Re_D = abs(rho_S*V_S*D(m)/oilmu_0);
    f = (-1.8 * log10(6.9/Re_D ...
        + (steeleps/D(m)/3.7)^1.11))^-2;

    V_P = V_S ...
        / (exp(s*(H(m+1)-H(m))/2) ...
        + f*V_S*abs(V_S)*(x(m+1)-x(m))/(2*D(m)*a_S^2));
    V_new(m) = V_P;
    rho_P = 1/(1/rho_S ...
        - (g*(H(m+1)-H(m)) ...
        + f*V_P*abs(V_S)/2/D(m)*(x(m+1)-x(m))) ...
        /oilK_0);
    rho_new(m) = rho_P;
    p_new(m) = oilp_0 + oilK_0*reallog(rho_P/oilrho_0);
end

% Reassessing the velocity V_w, at the bottom of the well

Q_w = PI(1,2) * (p_r - p_new(1,1));
rho_w = rho_new(1,1);
A_w = A_0(1,:)*oilrho_0./rho_w.*exp(oilK_0/oila_0^2*(1/oilrho_0-1./rho_w));
V_w = Q_w/A_w*oilrho_0/rho_new(1,1);

V_P = mean([V_w,V_new(1,1)]);
rho_P = rho_new(1,1);

V_new(1,1) = V_P;


% Calculating back from bottom, with the new velocity

for m = 2:M3
    rho_R = rho_P;
    V_R = V_P*A_0(m-1)/A_0(m);
    a_R = sqrt(oilK_0/rho_R/(1+oilK_0/steelE*D(m-1)/steele));
    s = 2*g/a_R^2;
    Re_D = abs(rho_R*V_R*D(m)/oilmu_0);
    f = (-1.8 * log10(6.9/Re_D ...
        + (steeleps/D(m)/3.7)^1.11))^-2;

    V_P = V_R ...
        / (exp(-s*(H(m)-H(m-1))/2) ...
        - f*V_R*abs(V_R)*(x(m)-x(m-1))/(2*D(m)*a_R^2));
    V_new(m) = V_P;
    rho_P = 1/(1/rho_R ...
        + (g*(H(m)-H(m-1)) ...
        + f*V_P*abs(V_R)/2/D(m)*(x(m)-x(m-1))) ...
        / oilK_0);
```

```
            rho_new(m) = rho_P;
            p_new(m) = oilp_0 + oilK_0*reallog(rho_P/oilrho_0);
        end

        p_P = p_t(i);
        V_P = V_new(M3);
        rho_P = oilrho_0*exp((p_P-oilp_0)/oilK_0);

        err = norm((p_new(M3,1)-p_t));

        it = it + 1;

        p_new(M3) = p_P;
        rho_new(M3) = rho_P;
    end

    p(:,i) = p_new;
    V(:,i) = V_new;
    rho(:,i) = rho_new;
    disp(['Case ' num2str(i) ' done'])
end


p_blowout = [p_blowout(1:2*M,wellD*10-1); ...
    p_blowout(2*M+1:M3,riserD*10-1)];
V_blowout = [V_blowout(1:2*M,wellD*10-1); ...
    V_blowout(2*M+1:M3,riserD*10-1)];
rho_blowout = [rho_blowout(1:2*M,wellD*10-1); ...
    rho_blowout(2*M+1:M3,riserD*10-1)];

V_ss_D400 = [V_blowout V];
rho_ss_D400 = [rho_blowout rho];
p_ss_D400 = [p_blowout p];
a_ss_D400 = sqrt(oilK_0./rho_ss_D400./(1+oilK_0/steelE.*D*ones(1,5)/steele));
A_ss_D400 = A_0*ones(1,5).*oilrho_0./rho_ss_D400.*exp(oilK_0./a_ss_D400.^2.*(1/oilrho_
Q_ss_D400 = A_ss_D400.*V_ss_D400;
Q_ss_D400_0 = rho_ss_D400(1,:)/oilrho_0.*Q_ss_D400(1,:);

% Making matrices with the key data
p_ss_D400_well = [p_blowout(1) p(1,:)];
p_ss_D400_head = [p_blowout(2*M) p(2*M,:)];
p_ss_D400_top = [p_blowout(M3) p(M3,:)];
Q_ss_D400_well = [Q_blowout_0*oilrho_0./rho_blowout(1) Q_ss_D400(1,:)];
Q_ss_D400_head = [Q_blowout_0*oilrho_0./rho_blowout(2*M) Q_ss_D400(2*M,:)];
Q_ss_D400_top = [Q_blowout_0*oilrho_0./rho_blowout(M3) Q_ss_D400(M3,:)];

save('ssD400','V_ss_D400','Q_ss_D400', ...
    'rho_ss_D400','p_ss_D400', ...
    'p_ss_D400_well','p_ss_D400_head','p_ss_D400_top', ...
    'Q_ss_D400_well','Q_ss_D400_head','Q_ss_D400_top')
```

```
for i=1:5
    disp(['Case ' num2str(i)])
    disp(' ')
    disp(['D_well = ' num2str(wellD) ' m'])
    disp(['D_riser = ' num2str(riserD) ' m'])
    disp(['p_well = ' num2str(p_ss_D400(1,i)/1E5) ' bar'])
    disp(['p_head = ' num2str(p_ss_D400(2*M,i)/1E5) ' bar'])
    disp(['p_top = ' num2str(p_ss_D400(M3,i)/1E5) ' bar'])
    disp(['Q_0 = ' num2str(Q_ss_D400_0(i)) ' m^3/s'])
    disp(['V_top = ' num2str(V_ss_D400(M3,i)) ' m/s'])
    disp(' ')
    disp(' ')
end


p_w = p_ss_D400(1,:);
p_hydrostat = ones(M3,1)*p_w ...
    - oilrho_0*g*H*ones(1,5);

V_bernoulli = sqrt((ones(M3,1)*Q_ss_D400_0./(A_0*ones(1,5))).^2 ...
    - 2*g*H*ones(1,5) ...
    + ones(M3,1)*p_w*2/oilrho_0 ...
    - 2*p_hydrostat/oilrho_0);

p_diff = (p_hydrostat-p_ss_D400)/1E5;
V_diff = (V_bernoulli-V_ss_D400)./V_ss_D400*100;


figure(3)
clf
subplot(2,1,1)
hold on
plot(x,p_ss_D400(:,1)/1E5,'k','LineWidth',2)
plot(x,p_ss_D400(:,2)/1E5,'k:','LineWidth',1)
plot(x,p_ss_D400(:,3)/1E5,'k-.','LineWidth',1)
plot(x,p_ss_D400(:,4)/1E5,'k--','LineWidth',1)
plot(x,p_ss_D400(:,5)/1E5,'k','LineWidth',1)
hold off
title('Pressure')
xlabel('L [m]')
ylabel('p [bar]')
legend(['p_{top} = ' num2str(p_ss_D400(M3,1)/1E5) ' bar'], ...
    ['p_{top} = ' num2str(p_ss_D400(M3,2)/1E5) ' bar'], ...
    ['p_{top} = ' num2str(p_ss_D400(M3,3)/1E5) ' bar'], ...
    ['p_{top} = ' num2str(p_ss_D400(M3,4)/1E5) ' bar'], ...
    ['p_{top} = ' num2str(p_ss_D400(M3,5)/1E5) ' bar'])

subplot(2,1,2)
hold on
```

```
plot(x,p_diff(:,1),'k','LineWidth',2)
plot(x,p_diff(:,2),'k:','LineWidth',1)
plot(x,p_diff(:,3),'k-.','LineWidth',1)
plot(x,p_diff(:,4),'k--','LineWidth',1)
plot(x,p_diff(:,5),'k','LineWidth',1)
hold off
title('Pressure, Difference from Hydrostatic')
xlabel('L [m]')
ylabel('p_{diff} [bar]')

print -deps ../plots/V_V_diff_ss_D400


figure(4)
clf
subplot(2,1,1)
hold on
plot(x,V_ss_D400(:,1),'k','LineWidth',2)
plot(x,V_ss_D400(:,2),'k:','LineWidth',1)
plot(x,V_ss_D400(:,3),'k-.','LineWidth',1)
plot(x,V_ss_D400(:,4),'k--','LineWidth',1)
plot(x,V_ss_D400(:,5),'k','LineWidth',1)
hold off
title('Velocity')
xlabel('L [m]')
ylabel('V [m/s]')
legend(['p_t = ' num2str(p_ss_D400(M3,1)/1E5) ' bar'], ...
    ['p_t = ' num2str(p_ss_D400(M3,2)/1E5) ' bar'], ...
    ['p_t = ' num2str(p_ss_D400(M3,3)/1E5) ' bar'], ...
    ['p_t = ' num2str(p_ss_D400(M3,4)/1E5) ' bar'], ...
    ['p_t = ' num2str(p_ss_D400(M3,5)/1E5) ' bar'])

subplot(2,1,2)
hold on
plot(x,V_diff(:,1),'k','LineWidth',2)
plot(x,V_diff(:,2),'k:','LineWidth',1)
plot(x,V_diff(:,3),'k-.','LineWidth',1)
plot(x,V_diff(:,4),'k--','LineWidth',1)
plot(x,V_diff(:,5),'k','LineWidth',1)
hold off
title('Velocity, Difference from Bernoulli')
xlabel('L [m]')
ylabel('V_{diff} [%]')

print -deps ../plots/p_p_diff_ss_D400
```

A.3.1 plotSteadyState.m

```
% 0-------------------------------------------------------------------------0
% | Program plotsteadystate.m                                               |
% | Used to plot properties of flow in the system at steady-state           |
% | conditions.                                                             |
% | Uses data from:                                                         |
% |     ssD300.mat                                                          |
% |     ssD400.mat                                                          |
% |     initdata.mat                                                        |
% | Saves figures in:                                                       |
% |     p_diff_steadystate.eps                                              |
% |     Q_0_steadystate.eps                                                 |
% |     a_A_rho_steadystate.eps                                             |
% |                                                                         |
% | Coded by:                                                               |
% |     Stud. Techn. Eivind Vrålstad,                                       |
% |     Institutt for Energi- og prosessteknikk, NTNU, 2011                 |
% 0-------------------------------------------------------------------------0

clc
clear all

load('../initdata')
load('ssD300')
load('ssD400')


M3 = M*3+1;

dx1 = wellL/M;
x1 = (0:dx1:wellL)';

dx2 = wellH/M;
x2 = (wellL+dx2:dx2:(wellL+wellH))';

dx3 = riserH/M;
x3 = ((wellL+wellH)+dx3:dx3:(wellL+wellH+riserH))';

x = [x1;x2;x3];


D300 = [ones(2*M,1)*.3;ones(M+1,1)*.3];
D400 = [ones(2*M,1)*.3;ones(M+1,1)*.4];


p_D300 = (p_ss_D300-p_ss_D300(:,1)*ones(1,5))*1E-5;
p_D400 = (p_ss_D400-p_ss_D400(:,1)*ones(1,5))*1E-5;

Q_D300 = Q_ss_D300;
```

```
Q_D400 = Q_ss_D400;

a_ss_D300 = sqrt(oilK_0./rho_ss_D300./(1+oilK_0/steelE.*D300*ones(1,5)/steele));
a_ss_D400 = sqrt(oilK_0./rho_ss_D400./(1+oilK_0/steelE.*D400*ones(1,5)/steele));

A_0_ss_D300 = pi/4*D300.^2;
A_0_ss_D400 = pi/4*D400.^2;

A_ss_D300 = A_0_ss_D300*ones(1,5).*oilrho_0./rho_ss_D300.*exp(oilK_0./a_ss_D300.^2.*(1
A_ss_D400 = A_0_ss_D400*ones(1,5).*oilrho_0./rho_ss_D400.*exp(oilK_0./a_ss_D400.^2.*(1

Q_0_ss_D300 = rho_ss_D300(1,:)/oilrho_0.*V_ss_D300(1,:).*A_ss_D300(1,:);
Q_0_ss_D400 = rho_ss_D400(1,:)/oilrho_0.*V_ss_D400(1,:).*A_ss_D400(1,:);

disp('Steady-state')
disp(' ')
disp(['a_max = ' num2str(max(a_ss_D300(:,2)))])
disp(['a_min = ' num2str(min(a_ss_D300(:,2)))])
disp(['a_0 = ' num2str(oila_0)])
disp(['A_max = ' num2str(max(A_ss_D300(:,2)))])
disp(['A_min = ' num2str(min(A_ss_D300(:,2)))])
disp(['A_0 = ' num2str(A_0_ss_D300(1,1))])
disp(['rho_max = ' num2str(max(rho_ss_D300(:,2)))])
disp(['rho_min = ' num2str(min(rho_ss_D300(:,2)))])
disp(['rho_0 = ' num2str(oilrho_0)])

i = 0;
for p_top = 20:-5:5
    i = i + 1;
    disp(' ')
    disp(' ')
    disp(['Case ' num2str(i) ', p_top = ' num2str(p_top) ' bar, D_riser = 300 mm'])
    disp(' ')
    disp(['p_well = ' num2str(p_ss_D300(1,i+1)*1E-5) ' bar'])
    disp(['p_head = ' num2str(p_ss_D300(2*M,i+1)*1E-5) ' bar'])
    disp(['p_top = ' num2str(p_ss_D300(M3,i+1)*1E-5) ' bar'])
    disp(' ')
    disp(['V_well = ' num2str(V_ss_D300(1,i+1)) ' m/s'])
    disp(['V_head = ' num2str(V_ss_D300(2*M,i+1)) ' m/s'])
    disp(['V_top = ' num2str(V_ss_D300(M3,i+1)) ' m/s'])
    disp(' ')
    disp(['Q_0 = ' num2str(Q_0_ss_D300(i+1)) ' m^3/s'])
end


i = 0;
for p_top = 20:-5:5
    i = i + 1;
    disp(' ')
    disp(' ')
    disp(['Case ' num2str(i) ', p_top = ' num2str(p_top) ' bar, D_riser = 400 mm'])
    disp(' ')
```

```
    disp(['p_well = ' num2str(p_ss_D400(1,i+1)*1E-5) ' bar'])
    disp(['p_head = ' num2str(p_ss_D400(2*M,i+1)*1E-5) ' bar'])
    disp(['p_top = ' num2str(p_ss_D400(M3,i+1)*1E-5) ' bar'])
    disp(' ')
    disp(['V_well = ' num2str(V_ss_D400(1,i+1)) ' m/s'])
    disp(['V_head = ' num2str(V_ss_D400(2*M,i+1)) ' m/s'])
    disp(['V_top = ' num2str(V_ss_D400(M3,i+1)) ' m/s'])
    disp(' ')
    disp(['Q_0 = ' num2str(Q_0_ss_D400(i+1)) ' m^3/s'])
end


figure(1)
clf
subplot(2,1,1)
hold on
plot(x,p_D300(:,2)','k:','LineWidth',1)
plot(x,p_D300(:,3),'k-.','LineWidth',1)
plot(x,p_D300(:,4),'k--','LineWidth',1)
plot(x,p_D300(:,5),'k','LineWidth',1)
hold off
title('Pressure difference from blowout, D = 300 mm')
xlabel('Position')
ylabel('p_{diff} [bar]')
legend( ...
    ['p_{top} = '    num2str(p_ss_D300_top(2)*1e-5) ' bar'], ...
    ['p_{top} = '    num2str(p_ss_D300_top(3)*1e-5) ' bar'], ...
    ['p_{top} = '    num2str(p_ss_D300_top(4)*1e-5) ' bar'], ...
    ['p_{top} = '    num2str(p_ss_D300_top(5)*1e-5) ' bar'])

subplot(2,1,2)
hold on
plot(x,p_D400(:,2)','k:','LineWidth',1)
plot(x,p_D400(:,3),'k-.','LineWidth',1)
plot(x,p_D400(:,4),'k--','LineWidth',1)
plot(x,p_D400(:,5),'k','LineWidth',1)
hold off
title('Pressure difference from blowout, D = 400 mm')
xlabel('Position')
ylabel('p_{diff} [bar]')

print -deps ../plots/p_diff_steadystate

figure(2)
clf
subplot(2,1,1)
hold on
plot(x,Q_D300(:,1),'k','LineWidth',2)
plot(x,Q_D300(:,2)','k:','LineWidth',1)
plot(x,Q_D300(:,3),'k-.','LineWidth',1)
plot(x,Q_D300(:,4),'k--','LineWidth',1)
plot(x,Q_D300(:,5),'k','LineWidth',1)
```

```
hold off
title('Volumetric flow, D = 300 mm')
xlabel('Position')
ylabel('Q [m^3/s]')
legend( ...
    ['Q_{0,blowout} = '    num2str(Q_0_ss_D300(1)) ' m/s'], ...
    ['Q_{0,20} = '    num2str(Q_0_ss_D300(2)) ' m/s'], ...
    ['Q_{0,15} = '    num2str(Q_0_ss_D300(3)) ' m/s'], ...
    ['Q_{0,10} = '    num2str(Q_0_ss_D300(4)) ' m/s'], ...
    ['Q_{0,5} = '    num2str(Q_0_ss_D300(5)) ' m/s'], ...
    'location','EastOutside')

subplot(2,1,2)
hold on
plot(x,Q_D400(:,1),'k','LineWidth',2)
plot(x,Q_D400(:,2)','k:','LineWidth',1)
plot(x,Q_D400(:,3),'k-.','LineWidth',1)
plot(x,Q_D400(:,4),'k--','LineWidth',1)
plot(x,Q_D400(:,5),'k','LineWidth',1)
hold off
title('Volumetric flow, D = 400 mm')
xlabel('Position')
ylabel('Q [m^3/s]')
legend( ...
    ['Q_{0,blowout} = '    num2str(Q_0_ss_D400(1)) ' m/s'], ...
    ['Q_{0,20} = '    num2str(Q_0_ss_D400(2)) ' m/s'], ...
    ['Q_{0,15} = '    num2str(Q_0_ss_D400(3)) ' m/s'], ...
    ['Q_{0,10} = '    num2str(Q_0_ss_D400(4)) ' m/s'], ...
    ['Q_{0,5} = '    num2str(Q_0_ss_D400(5)) ' m/s'], ...
    'location','EastOutside')

print -deps ../plots/Q_0_steadystate

figure(3)
clf
hold on
plot(x,a_ss_D300(:,2)/max(a_ss_D300(:,2)),'k','LineWidth',1)
plot(x,A_ss_D300(:,2)/max(A_ss_D300(:,2)),'k:','LineWidth',1)
plot(x,rho_ss_D300(:,2)/max(rho_ss_D300(:,2)),'k-.','LineWidth',1)
%plot(x,p_ss_D300(:,2)/max(p_ss_D300(:,2)),'k--','LineWidth',1)
hold off
title('Variation of a, A and \rho with D = 300 mm, top p_{top} = 20 bar')
xlabel('Position')
legend( ...
    'a/max(a)', ...
    'A/max(A)', ...
    '\rho/max(\rho)', ...
    'location','East')

print -deps ../plots/a_A_rho_steadystate
```

A.4 transient.m

```
% 0-------------------------------------------------------------------------0
% | Program transient.m                                                     |
% | Used to find properties of flow in the transient system.                |
% | Uses data from:                                                         |
% |     blowout.mat                                                         |
% |     initdata.mat                                                        |
% |     ss(...).mat                                                         |
% | Saves figures in:                                                       |
% |     V_surf_(...).eps                                                    |
% |     Q_0_surf_(...).eps                                                  |
% |     p_V_(...).eps                                                       |
% |                                                                         |
% | Coded by:                                                               |
% |     Stud. Techn. Eivind Vrålstad,                                       |
% |     Institutt for Energi- og prosessteknikk, NTNU, 2011                 |
% 0-------------------------------------------------------------------------0

clear all
clc
tic

%O-----------------------------------------------------------------------O
%| Global constants                                                      |
%O-----------------------------------------------------------------------O

global g PI theta sinalpha D tol maxit steptime M M3 N ...
    oila_0 oilmu_0 oilK_0 oilT oilp_0 oilrho_0 ...
    seaT searho_0 ...
    riserH ...
    x dx dx1 dx2 dx3 dt H p_init rho_init ...
    steeleps steelE steele


tol = 1E-20;
maxit = 1000;

load('../blowout.mat')
load('../initdata.mat')
load('../ssD400/ssD400.mat')



%O-----------------------------------------------------------------------O
%| Riser properties                                                      |
%O-----------------------------------------------------------------------O

riserD = .4;            %Diameter [m]
```

```
%O--------------------------------------------------------------------------O
%| Well properties                                                         |
%O--------------------------------------------------------------------------O

wellD = .3;             %Diameter [m]



%O--------------------------------------------------------------------------O
%| Fluid properties                                                        |
%O--------------------------------------------------------------------------O

seaT = 283.15;          %Temperature of sea water [K]

oilT = 283.15;          %Temperature of oil [K]
oilp_0 = 101.325E3;     %Reference pressure [Pa]

p_atm = oilp_0;



% O------------------------------------------------------------------------O
% | Defining the grid                                                     |
% | dx  . .  |  . .                                                       |
% | x  o o o | o o o                                                      |
% | m  1 2 3 | ... M                                                      |
% O------------------------------------------------------------------------O

M3 = 3*M+1;

dx1 = wellL/M;
x1 = (1:M+1)'*dx1;

dx2 = wellH/M;
x2 = wellL+(2:M+1)'*dx2;

dx3 = riserH/M;
x3 = wellL+wellH+(2:M+1)'*dx3;



N_dt = 50*M;            %Number of time steps
t_0 = 0;                %Starting time [s]
t_end = 40;             %End time [s]
N = N_dt+1;             %Number of nodes in time
dt = (t_end-t_0)/N_dt;  %Time step [s]


x = [x1;x2;x3];
t = 0:dt:t_end;
```

```
%0————————————————————————————————————————————————————————————————————————0
%| Setting constants that vary through the pipe                                |
%0————————————————————————————————————————————————————————————————————————0

sinalpha = [zeros(M,1);ones(2*M,1)];
D = [ones(2*M,1)*wellD;ones(M+1,1)*riserD];
A_0 = pi/4*D.^2;
H = wellH+riserH-[zeros(M+1,1); (1:M)'*dx2; wellH+(1:M)'*dx3];
dx = [ones(M,1)*dx1; ones(M,1)*dx2; ones(M,1)*dx3];
theta = dx/(2*dt);


if dt*oila_0≤max([dx1 dx2 dx3])
    disp(['dt*a = ' num2str(dt*oila_0) ', dx = ' num2str(max([dx1 dx2 dx3])) '. Grid OK'])
else
    disp(['dt*a = ' num2str(dt*oila_0) ', dx = ' num2str(max([dx1 dx2 dx3])) '. Grid unstabl
    break
end

%O————————————————————————————————————————————————————————————————————————O
%| Initial values                                                              |
%O————————————————————————————————————————————————————————————————————————O



[rho_riser_init p_riser_init] = rho_p_init(H((2*M+1):M3));

V_init = [V_blowout(1:2*M,wellD*10-1);zeros(M+1,1)];
rho_init = [rho_blowout(1:2*M,wellD*10-1);rho_riser_init];
p_init = [p_blowout(1:2*M,wellD*10-1);p_riser_init];


steptime = toc;
disp(['Initial values calculated. Time: ', num2str(steptime), ' s.'])

PI = PI(wellD*10-1);



%O————————————————————————————————————————————————————————————————————————O
%| Valve properties                                                            |
%O————————————————————————————————————————————————————————————————————————O

valvetime = 10;
valvep_end = 15E5;
dp = (p_init(M3,1)-valvep_end)/valvetime*dt;
valvep = p_init(M3,1):-dp:valvep_end;
```

```
%O--------------------------------------------------------------------------O
%| Initiating solver                                                       |
%O--------------------------------------------------------------------------O

p_new = p_init;
rho_new = rho_init;
V_new = V_init;


% Allocating space for the matrices stroing the last 4 loops' values of
% variables
rho_curr = rho_init*ones(1,4);
p_curr = p_init*ones(1,4);
V_curr = V_init*ones(1,4);
t_curr = zeros(M3,4);
x_curr = x*ones(1,4);
a_curr = sqrt(oilK_0./rho_curr./(1+oilK_0/steelE*D*ones(1,4)/steele));
A_curr = A_0*ones(1,4).*oilrho_0./rho_curr.*exp(oilK_0./a_curr.^2.*(1/oilrho_0-1./rho_
Q_0_curr = A_curr.*V_curr;


% Allocating space for the result matrices
V_res = zeros(M3,(N-1)*50/M+1);
p_res = zeros(M3,(N-1)*50/M+1);
rho_res = zeros(M3,(N-1)*50/M+1);
t_res = zeros(M3,(N-1)*50/M+1);
x_res = x*ones(1,(N-1)*50/M+1);
Q_0_res = zeros(M3,(N-1)*50/M+1);

V_res(:,1) = V_init;
rho_res(:,1) = rho_init;
p_res(:,1) = p_init;
Q_0_res(:,1) = Q_0_curr(:,1);


Re_new = D.*abs(V_curr(:,1)).*rho_curr(:,1)/oilmu_0;
Re_new(Re_new<1E3) = 2E3;
f = (-1.8.*real(log10(6.9./Re_new + (steeleps./D/3.7).^1.11))).^-2;
for n=2:N
    tic



    % A,D: Left, B,E: Right
    V_A = V_curr(1:M3-1,1);
    V_B = V_curr(2:M3,1);
    V_D = V_curr(1:M3-1,3);
    V_E = V_curr(2:M3,3);

    % A,D: Left, B,E: Right
    rho_A = rho_curr(1:M3-1,1);
```

```
rho_B = rho_curr(2:M3,1);
rho_D = rho_curr(1:M3-1,3);
rho_E = rho_curr(2:M3,3);

a_R = a_curr(1:M3-1,1);
a_S = a_curr(2:M3,1);

% Interpolating left (R) values for V and rho at t-dt
V_X = V_A*3/4 ...
    - V_D/4 ...
    - a_R/2;

V_R = (V_X ...
    + sqrt(V_X.^2 ...
    + (3/2*a_R - theta) ...
    .* (V_A - V_D) ...
    + a_R.*V_D)).*A_curr(1:M3-1,1)./A_curr(2:M3,1);

rho_R = 1./(1./rho_D ...
    - (1./rho_A - 1./rho_D) ...
    .* (theta./(V_R + a_R) - 3/2));

% Interpolating right (S) values for V and rho at t-dt
V_Y = 3/4*V_B ...
    - V_E/4 ...
    + a_S/2;

V_S = (V_Y ...
    - sqrt(V_Y.^2 ...
    - (3/2*a_S - theta) ...
    .* (V_B - V_E) ...
    - a_S.*V_E)).*A_curr(2:M3,1)./A_curr(1:M3-1,1);

rho_S = 1./(1./rho_E ...
    + (1./rho_B - 1./rho_E) ...
    .* (theta./(V_S - a_S) + 3/2));


err = inf;
it = 0;
while err>tol && it<maxit
    rho_old = rho_new;

    % Calculating intermidiate variables
    B_R = a_R/oilK_0;
    B_S = a_S/oilK_0;
    R_R = dx.*a_R.*f(1:M3-1)./(2*D(1:M3-1)*oilK_0);
    R_S = dx.*a_S.*f(2:M3)./(2*D(2:M3)*oilK_0);

    % Calculating left (+) variables
    C_P = 1./rho_R ...
        - B_R.*abs(V_R) ...
```

```
    + g*dx.*sinalpha.*B_R ...%dx(1:M3-1,1).*sinalpha(1:M3-1,1)
    ./ (V_R+a_R);

B_P = B_R ...
    + R_R.*abs(V_R) ...
    ./ (V_R+a_R);

% Calculating right (-) variables
C_M = 1./rho_S ...
    + B_S.*abs(V_S) ...
    + g*dx.*sinalpha.*B_S ...
    ./ (V_S-a_S);

B_M = B_S ...
    - R_S.*abs(V_S) ...
    ./ (V_S-a_S);

% Calculating V-values
V_P = (C_M(2:M3-1) ...
    - C_P(1:M3-2)) ...
    ./ (B_M(2:M3-1) ...
    + B_P(1:M3-2));


% Valve opening
if t_curr(1,1) < valvetime-dt
    p_M3 = valvep(n);
    rho_M3 = oilrho_0*exp((p_M3-oilp_0)/oilK_0);
else
    rho_M3 = rho_curr(M3,2);
end


%Calculating end values for V
V_M3 = 1/B_P(M3-1)*(1/rho_M3-C_P(M3-1));


rho_1_new = rho_curr(1,2);
p_1_new = p_curr(1,2);
p_R = searho_0*g*H(1)+oilp_0;
err2 = inf;
it2 = 0;
while err2>tol && it2<=maxit
    rho_1_old = rho_1_new;
    p_1_old = p_1_new;

    p_1_new = (9*p_1_old+abs(p_R - A_0(1)/B_M(1)/PI*(C_M(1)-1/rho_1_old)))/10;
    rho_1_new = oilrho_0*exp((p_1_new-oilp_0)/oilK_0);


    if it2 < 3;
        err2 = inf;
```

```
    else
        err2 = norm((rho_1_new-rho_1_old).^2);
    end

    it2 = it2+1;
end


V_1 = -1/B_M(1)*(1/rho_1_new-C_M(1));
rho_1 = rho_1_new;
p_1 = p_1_new;

%Inserting new end values for V
V_new = [V_1; V_P; V_M3];


% Calculating rho-values
rho_P = (B_P(1:M3-2) ...
    + B_M(2:M3-1)) ...
    ./ (C_P(1:M3-2).*B_M(2:M3-1) ...
    + C_M(2:M3-1).*B_P(1:M3-2));


rho_new = [rho_1; rho_P; rho_M3];

% Making shure that head velocity and density never falls below
% initial, and making an end of the riser at 2M+1 if it does
if V_new(2*M) < V_init(2*M,1) || rho_new(2*M) > rho_init(2*M,1)
    V_new(2*M) = V_init(2*M,1);
    rho_new(2*M) = rho_init(2*M);
    rho_new(2*M+1) = rho_init(2*M+1);
    V_new(2*M+1) = (C_M(2*M+2)-1/rho_new(2*M+1))./B_M(2*M+2);
end

%Calculating new values for p at n
p_new = oilp_0+oilK_0*reallog(rho_new/oilrho_0);


% Calculating new values for a and f
a = sqrt(oilK_0./rho_new./(1+oilK_0/steelE*D/steele));
Re_new = D.*abs(V_new).*rho_new/oilmu_0;
Re_new(Re_new<1E3) = 2E3;
f = (-1.8.*real(log10(6.9./Re_new + (steeleps./D/3.7).^1.11))).^-2;

%Calculating error
err = norm((rho_old-rho_new).^2);
if it>maxit
    disp(['Warning: V does not converge in ' num2str(maxit) ' iterations in n = ' nu
    break
elseif it<3
    err = inf;
    it = it+1;
```

```
        else
            it = it+1;
        end
    end


    % Storing current data, pushing old data out
    rho_curr = [rho_new rho_curr(:,1:3)];
    p_curr = [p_new p_curr(:,1:3)];
    V_curr = [V_new V_curr(:,1:3)];
    t_curr = [t_curr(:,1)+dt t_curr(:,1:3)];
    a_curr = [sqrt(oilK_0./rho_curr(:,1)./(1+oilK_0/steelE*D/steele)) a_curr(:,1:3)];
    A_curr = [A_0.*oilrho_0./rho_curr(:,1).*exp(oilK_0./a_curr(:,1).^2.*(1/oilrho_0-1.
    Q_0_curr = A_curr.*V_curr;

    %Combining and inserting into the answer matrices, each 10th time step
    if (n-1)*50/M == floor((n-1)*50/M)
        V_res(:,(n-1)*50/M+1) = V_new;
        rho_res(:,(n-1)*50/M+1) = rho_new;
        p_res(:,(n-1)*50/M+1) = p_new;
        t_res(:,(n-1)*50/M+1) = t_curr(:,1);
        x_res(:,(n-1)*50/M+1) = x_curr(:,1);
        Q_0_res(:,(n-1)*50/M+1) = Q_0_curr(:,1);
    end


    steptime(n) = toc;
    clc
    disp(['Time step ', num2str(n), ' of ', num2str(N), ', calculation time: ', num2st

end

clc
disp(['Total calculation time: ' num2str(sum(steptime)) ' s'])




x_trVt10s15bar = x_res(:,1);
t_trVt10s15bar = t_res(1,:);
V_trVt10s15bar = V_res([1 2*M 2*M+1 M3],:);
p_trVt10s15bar = p_res([1 2*M 2*M+1 M3],:);
Q_0_trVt10s15bar = Q_0_res([1 2*M 2*M+1 M3],:);
save('trVt10s15bar','x_trVt10s15bar','t_trVt10s15bar','V_trVt10s15bar','p_trVt10s15bar


figure(1)
clf
surf(x_res,t_res,V_res,'FaceColor','interp','EdgeColor','none','FaceLighting','phong')
title('Velocity')
xlabel('x [m]')
```

```
ylabel('t [s]')
zlabel('V [m/s]')
axis([0 max(x) 0 t_end])
view(50,50)
print -depsc2 ../plots/V_surf_trVt10s15bar


figure(2)
clf
surf(x_res,t_res,Q_0_res,'FaceColor','interp','EdgeColor','none','FaceLighting','phong')
title('Volumetric flow')
xlabel('x [m]')
ylabel('t [s]')
zlabel('[m^3/s]')
axis([0 max(x) 0 t_end])
view(50,30)
print -depsc2 ../plots/Q_0_surf_trVt10s15bar

figure(3)
clf
surf(x_res,t_res,p_res*1E-5,'FaceColor','interp','EdgeColor','none','FaceLighting','phong')
title('Pressure')
xlabel('x [m]')
ylabel('t [s]')
zlabel('[bar]')
axis([0 max(x) 0 t_end])
view(50,50)


figure(4)
clf
subplot(2,1,1)
hold on
plot(t_res(1,:),Q_0_res(1,:),'k:','LineWidth',1)
plot(t_res(1,:),Q_0_res(2*M,:),'k--','LineWidth',1)
plot(t_res(1,:),Q_0_res(M3,:),'k','LineWidth',1)
hold off
title('Volumetric flow Q_0')
xlabel('t [s]')
ylabel('Q [m^3/s]')
legend('Well', ...
    'Wellhead', ...
    'Top')

subplot(2,1,2)
hold on
plot(t_res(1,:),V_res(1,:)*1E-5,'k:','LineWidth',1)
plot(t_res(1,:),V_res(2*M,:)*1E-5,'k--','LineWidth',1)
plot(t_res(1,:),V_res(M3,:)*1E-5,'k','LineWidth',1)
hold off
title('Velocity')
xlabel('t [s]')
```

```
ylabel('V [m/s]')
print -depsc2 ../plots/p_V_trVt10s15bar
```

A.4.1 plotTransient.m

```
% 0-------------------------------------------------------------------------0
% | Program plotTransient.m                                                 |
% | Used to plot properties of flow in the transient system.                |
% | conditions.                                                             |
% | Uses data from:                                                         |
% |     tr(#1).mat                                                          |
% |     ...                                                                 |
% |     tr(#n).mat                                                          |
% |     ssD400.mat                                                          |
% |     initdata.mat                                                        |
% | Saves figures in:                                                       |
% |     V_(...).eps                                                         |
% |     p_(...).eps                                                         |
% |                                                                         |
% | Coded by:                                                               |
% |     Stud. Techn. Eivind Vrålstad,                                       |
% |     Institutt for Energi- og prosessteknikk, NTNU, 2011                 |
% 0-------------------------------------------------------------------------0

clear all
clc

load('initdata')
load('trVt10s20bar/trVt10s20bar.mat')
load('trVt10s15bar/trVt10s15bar.mat')
load('trVt20s20bar/trVt20s20bar.mat')
load('trVt20s15bar/trVt20s15bar.mat')
load('ssD400/ssD400')

t = t_trVt10s20bar;
N = length(t);
t_step = (N-1)/ceil(max(t));


p_well = [p_trVt10s15bar(1,:)*1E-5; ...
          p_trVt10s20bar(1,:)*1E-5; ...
          p_trVt20s15bar(1,:)*1E-5; ...
          p_trVt20s20bar(1,:)*1E-5];
p_head = [p_trVt10s15bar(2,:)*1E-5; ...
          p_trVt10s20bar(2,:)*1E-5; ...
          p_trVt20s15bar(2,:)*1E-5; ...
          p_trVt20s20bar(2,:)*1E-5];
p_top  = [p_trVt10s15bar(4,:)*1E-5; ...
          p_trVt10s20bar(4,:)*1E-5; ...
          p_trVt20s15bar(4,:)*1E-5; ...
          p_trVt20s20bar(4,:)*1E-5];
```

```
V_well = [V_trVt10s15bar(1,:); ...
          V_trVt10s20bar(1,:); ...
          V_trVt20s15bar(1,:); ...
          V_trVt20s20bar(1,:)];
V_head = [V_trVt10s15bar(2,:); ...
          V_trVt10s20bar(2,:); ...
          V_trVt20s15bar(2,:); ...
          V_trVt20s20bar(2,:)];
V_top  = [V_trVt10s15bar(4,:); ...
          V_trVt10s20bar(4,:); ...
          V_trVt20s15bar(4,:); ...
          V_trVt20s20bar(4,:)];


Q_0_well = [Q_0_trVt10s15bar(1,:); ...
          Q_0_trVt10s20bar(1,:); ...
          Q_0_trVt20s15bar(1,:); ...
          Q_0_trVt20s20bar(1,:)];
Q_0_head = [Q_0_trVt10s15bar(2,:); ...
          Q_0_trVt10s20bar(2,:); ...
          Q_0_trVt20s15bar(2,:); ...
          Q_0_trVt20s20bar(2,:)];
Q_0_top  = [Q_0_trVt10s15bar(4,:); ...
          Q_0_trVt10s20bar(4,:); ...
          Q_0_trVt20s15bar(4,:); ...
          Q_0_trVt20s20bar(4,:)];

i = 0;
for valvet = [10 20]
    for valvep = [15 20]
        i = i+1;
        disp(['Case ' num2str(i)])
        disp(['t_valve = ' num2str(valvet) ' s, p_valve = ' num2str(valvep) ' bar'])



        disp(' ')
        for j = 0:10:40
            disp(['p_well, ' num2str(j) ' s = ' num2str(p_well(i,1+t_step*j)) ' bar'])
        end
        disp(' ')
        for j = 0:10:40
            disp(['p_head, ' num2str(j) ' s = ' num2str(p_head(i,1+t_step*j)) ' bar'])
        end
        disp(' ')
        for j = 0:10:40
            disp(['p_top, ' num2str(j) ' s = ' num2str(p_top(i,1+t_step*j)) ' bar'])
        end
        disp(' ')
```

```
        disp(' ')
        for j = 0:10:40
            disp(['V_well, ' num2str(j) ' s = ' num2str(V_well(i,1+t_step*j)) ' m/s'])
        end
        [V_well_max I] = max(V_well(i,:));
        disp(['max(V_well) = ' num2str(V_well_max) ' @ t = ' num2str(t(I)) ' s'])
        disp(' ')
        for j = 0:10:40
            disp(['V_head, ' num2str(j) ' s = ' num2str(V_head(i,1+t_step*j)) ' m/s'])
        end
        [V_head_max I] = max(V_head(i,:));
        disp(['max(V_head) = ' num2str(V_head_max) ' @ t = ' num2str(t(I)) ' s'])
        disp(' ')
        for j = 0:10:40
            disp(['V_top, ' num2str(j) ' s = ' num2str(V_top(i,1+t_step*j)) ' m/s'])
        end
        [V_top_max I] = max(V_top(i,:));
        disp(['max(V_top) = ' num2str(V_top_max) ' @ t = ' num2str(t(I)) ' s'])
        disp(' ')


        disp(' ')
        for j = 0:10:40
            disp(['Q_0_well, ' num2str(j) ' s = ' num2str(Q_0_well(i,1+t_step*j)) ' m^3/s'])
        end
        disp(' ')
        for j = 0:10:40
            disp(['Q_0_head, ' num2str(j) ' s = ' num2str(Q_0_head(i,1+t_step*j)) ' m^3/s'])
        end
        disp(' ')
        for j = 0:10:40
            disp(['Q_0_top, ' num2str(j) ' s = ' num2str(Q_0_top(i,1+t_step*j)) ' m^3/s'])
        end
        disp(' ')



        disp(' ')
    end
end


figure(1)
clf
subplot(2,1,1)
hold on
plot(t_trVt10s20bar(1,:),V_trVt10s20bar(2,:),'k:','LineWidth',1)
plot(t_trVt10s15bar(1,:),V_trVt10s15bar(2,:),'k-.','LineWidth',1)
plot(t_trVt20s20bar(1,:),V_trVt20s20bar(2,:),'k--','LineWidth',1)
plot(t_trVt20s15bar(1,:),V_trVt20s15bar(2,:),'k','LineWidth',1)
hold off
title('Head velocities at different valve pressures and times')
```

```
xlabel('Time [s]')
ylabel('V_{head} [m/s]')

subplot(2,1,2)
hold on
plot(t_trVt10s20bar(1,:),V_trVt10s20bar(4,:),'k:','LineWidth',1)
plot(t_trVt10s15bar(1,:),V_trVt10s15bar(4,:),'k-.','LineWidth',1)
plot(t_trVt20s20bar(1,:),V_trVt20s20bar(4,:),'k--','LineWidth',1)
plot(t_trVt20s15bar(1,:),V_trVt20s15bar(4,:),'k','LineWidth',1)
hold off
title('Top velocities at different valve pressures and times')
xlabel('Time [s]')
ylabel('V_{top} [m/s]')
legend('t_{valve} = 10 s, p_{top} = 20 bar', ...
    't_{valve} = 10 s, p_{top} = 15 bar', ...
    't_{valve} = 20 s, p_{top} = 20 bar', ...
    't_{valve} = 20 s, p_{top} = 15 bar', ...
    'location','SouthEast')

print -depsc2 ../plots/V_SingleLinearOpening

figure(2)
clf
subplot(2,1,1)
hold on
plot(t_trVt10s20bar(1,:),p_trVt10s20bar(2,:)*1E-5,'k:','LineWidth',1)
plot(t_trVt10s15bar(1,:),p_trVt10s15bar(2,:)*1E-5,'k-.','LineWidth',1)
plot(t_trVt20s20bar(1,:),p_trVt20s20bar(2,:)*1E-5,'k--','LineWidth',1)
plot(t_trVt20s15bar(1,:),p_trVt20s15bar(2,:)*1E-5,'k','LineWidth',1)
hold off
title('Head pressures at different valve pressures and times')
xlabel('Time [s]')
ylabel('p_{head} [bara]')

subplot(2,1,2)
hold on
plot(t_trVt10s20bar(1,:),p_trVt10s20bar(4,:)*1E-5,'k:','LineWidth',1)
plot(t_trVt10s15bar(1,:),p_trVt10s15bar(4,:)*1E-5,'k-.','LineWidth',1)
plot(t_trVt20s20bar(1,:),p_trVt20s20bar(4,:)*1E-5,'k--','LineWidth',1)
plot(t_trVt20s15bar(1,:),p_trVt20s15bar(4,:)*1E-5,'k','LineWidth',1)
hold off
title('Top pressures at different valve pressures and times')
xlabel('Time [s]')
ylabel('p_{top} [bara]')
legend('t_{valve} = 10 s, p_{top} = 20 bar', ...
    't_{valve} = 10 s, p_{top} = 15 bar', ...
    't_{valve} = 20 s, p_{top} = 20 bar', ...
    't_{valve} = 20 s, p_{top} = 15 bar', ...
    'location','NorthEast')

print -depsc2 ../plots/p_SingleLinearOpening
```

A.4.2 rho_p_init.m

```
function [rho p] = rho_p_init(H)

% 0-------------------------------------------------------------------------0
% | Function rho_p_init                                                     |
% | Used to find initial values for p and rho if V=0.                       |
% |                                                                         |
% | Coded by:                                                               |
% |     Stud. Techn. Eivind Vrålstad,                                       |
% |     Institutt for Energi- og prosessteknikk, NTNU, 2011                 |
% 0-------------------------------------------------------------------------0

global tol maxit searho_0 g oilrho_0 oilp_0 oilK_0

%Calculating values for rho with constant density
p_bottom = searho_0*g*H(1,1) + oilp_0;

p_new = p_bottom - g*oilrho_0*(H(1,1)-H);

err = inf;
it = 0;

while err>tol && it<maxit
    p_old = p_new;

    p_new = p_bottom - oilrho_0*g*(H(1,1)-H).*exp(1/oilK_0*(p_old-oilp_0));

    err = norm((p_old-p_new).^2);
    it = it+1;
end

p = p_new;
rho = oilrho_0*exp(1/oilK_0*(p-oilp_0));

end
```

B Full Background (Norwegian)

Since the background section was capped in the preamble, here is a repetition with the full text (in Norwegian).

Bakgrunn

Ved massive oljeutslipp på havbunnen er det av avgjørende betydning å hindre at oljen får strømme fritt ut i omgivelsene og blande seg i vannmassene, slik at den spres over store områder med havstrømmer, bølger og vind, og fører til store miljøproblemer langs kyster og strender, samt i gyte- og oppvekstområder for fisk og sjødyr. Oppsamling av slike utslipp etter at oljen har nådd opp til overflaten viser seg alltid å være problematisk; særlig i områder som er utsatt for mye vær og vind og høye bølger. Med bakgrunn i den store utblåsningen ifm. havariet av BP sin oljeplattform "Deepwater Horizon" i Mexicogulfen sommeren 2010, ble det av veileder til denne oppgaven (T.Y.) skissert en hydrodynamisk limning som i prinsippet samler utslippet på havbunnen og transporterer oljen gjennom et stigerør til et oppsamlingsfartøy på overflaten. Denne løsningen krever en del utstyr som må foreligge på forhånd i ren beredskapsøyemed, og i denne oppgaven blir det fokusert på de strømningsmekaniske forholdene som må etableres og kontrolleres i grensesnittet mellom lekkasjepunkt og stigerør mot overflaten.

Oppgaven bearbeides ut fra følgende punkter

1. En kort gjennomgang av dagens status på intervensjons- og oppsamlingsmetoder ved store oljeutslipp på havbunn.

2. Valg av en hensiktsmessig beregningsmetode for transient strømning i lange stigerør; eksempelvis karakteristikkmetoden eller tilsvarende differansemetode.

3. Gjennomføring av beregning for et typisk tilfelle av oljelekkasje på dypt vann (ca.1000m), hvor et oljefylt intervensjonsestigerør i utgangspunktet er i trykkbalanse med sjøvannet pa havbunnen, i det tilkobling mot lekkasjestedet skjer.

4. Vurdering av resultatene som oppnås, og identifisering av krav til ejektorkapasitet pa overflaten for å kontrollere utblåsningen uten utslipp i havet.

5. Utvidelse av beregningsmodellen til å inkludere også selve brønnstromningen (bare hvis tiden tillater det).