Chair of Petroleum and Geothermal Energy Recovery

Master's Thesis

Downhole Dynamometer Sensor Data Evaluation

Georg Ignaz Frauscher, BSc

November 2020

MONTANUNIVERSITÄT LEOBEN
www.unileoben.ac.at

EIDESSTATTLICHE ERKLÄRUNG

Ich erkläre an Eides statt, dass ich diese Arbeit selbständig verfasst, andere als die angegebenen Quellen und Hilfsmittel nicht benutzt, und mich auch sonst keiner unerlaubten Hilfsmittel bedient habe.

Ich erkläre, dass ich die Richtlinien des Senats der Montanuniversität Leoben zu "Gute wissenschaftliche Praxis" gelesen, verstanden und befolgt habe.

Weiters erkläre ich, dass die elektronische und gedruckte Version der eingereichten wissenschaftlichen Abschlussarbeit formal und inhaltlich identisch sind.

Datum 09.11.2020

Georg Frauscher

Unterschrift Verfasser/in
Georg Ignaz Frauscher

Danksagung / Acknowledgement

Hiermit möchte ich mich herzlich bei Dipl.-Ing. Dipl.-Ing. Dr. mont. Clemens Langbauer für die tatkräftige Unterstützung bei der Erstellung dieser Diplomarbeit bedanken.

Ein weiterer Dank gilt meinem MatLab Nachhilfe Lehrer, Aram Khachatouri, der bei der Erstellung der MatLab Programme große Hilfe geleistet hat.

Außerdem möchte ich mich meinen Eltern, meiner Familie und meiner Freundin danken für die unaufhörliche Ermutigung, Unterstützung und Aufmerksamkeit.

Kurzfassung

Ziel dieser Arbeit ist es, die Downhole Dynamometer Sensor Technologie theoretisch und anhand eines Feldbeispiels in einer Ölquelle im Wiener Becken zu erläutern. Mithilfe der Programmierplattform MatLab werden Dateien generiert, die eine Interpretation und Auswertung der Ergebnisse ermöglichen.

Der erste Teil der Arbeit bietet einen allgemeinen Überblick über Sucker Rod Pumping Systeme und herkömmliche Dynamometer sowie die daraus resultierenden Dynamometerkarten und Mustererkennungsmethoden. Der zweite Teil konzentriert sich auf die Downhole Dynamometer Sensoren, die ab 2014 von der Montanuniversität Leoben in Zusammenarbeit mit der lilatech GmbH entwickelt wurden und die direkte Messung von Bohrlochdaten ermöglichen. Ein Tutorial zur Verwendung der generierten MatLab-Dateien wird bereitgestellt und die Ergebnisse des Feldbeispiels für das Wiener Becken werden damit ausgewertet.

5 Downhole Dynamometer Sensoren wurden an verschiedenen Punkten entlang des Sucker Rod Strings der Ölquelle im Wiener Becken installiert, und es wurden Messungen für ungefähr 3 Wochen durchgeführt, was zu 471 Datensätzen führte. Die erstellte MatLab-Datei „DataEvaluationDDS“ ermöglicht eine Interpretation jedes einzelnen Datensatzes, während die Dateien „DataEvaluationDDSTotals“ und „DataEvaluationDDSComparison“ eine Analyse und einen Vergleich über den gesamten Aufzeichnungszeitraum für jeden der 5 Sensoren ermöglichen. Die wichtigsten Ergebnisse werden bei der Untersuchung der resultierenden Load versus Position Diagramme ermittelt, die auf eine zunehmende Abpumpsituation im Laufe der Zeit hinweisen. Eine Temperaturanalyse zeigt, dass die aus dem Bohrloch gepumpte Flüssigkeit während der Hebeperiode um 17°C abkühlt und eine Bewertung der Bewegung zeigt bestimmte zeitliche Verschiebungen zwischen jedem Downhole Dynamometer Sensor, die durch Trägheitseffekte des Sucker Rod Strings und dessen hohe Dynamik verursacht werden. Sichtbare Schwankungen in einer Lastanalyse werden durch erhöhte Reibung verursacht und die Ergebnisse eines Vergleichs der Batteriespannung aller 5 Sensoren über den gesamten Aufzeichnungszeitraum zeigen, dass die Aufzeichnungszeiten für zukünftige Feldtests verlängert werden könnten.

Die Neuheit dieser Arbeit ist die Erstellung der MatLab-Dateien, die so programmiert sind, dass sie für zukünftige Feldbeispiele für Downhole Dynamometer Sensoren geeignet sind und für eine vollständige Auswertung der aufgezeichneten Bohrlochdaten in einem Sucker Rod Pumping System verwendet werden können.

Abstract

The objective of this thesis is to explain the downhole dynamometer sensor technology in theory and also by means of a field example in a Vienna Basin oil well. The programming platform MatLab is used to generate files, which enable an interpretation and evaluation of the results.

The first part of the thesis provides a general overview of sucker rod pumping systems and conventional dynamometers along with their resulting dynamometer cards and pattern recognition methods. The second part focuses on the downhole dynamometer sensors, which were developed starting in 2014 by the Montanuniversität Leoben in cooperation with lilatech GmbH and enable direct measuring of downhole data. A tutorial on how to use the generated MatLab files is provided and the results of the Vienna Basin field example are evaluated with them.

5 downhole dynamometer sensors were installed at different points of interest along the sucker rod string of the Vienna Basin oil well and measurements were taken for approximately 3 weeks, resulting in 471 datasets. The created MatLab file “DataEvaluationDDS” allows an interpretation of each individual dataset, while the files “DataEvaluationDDSTotals” and “DataEvaluationDDSComparison” enable an analysis and comparison over the entire recording period for each of the 5 sensors. The most significant findings are identified when examining the resulting load versus position diagrams, which indicate an increasing pump-off situation as time progresses. A temperature analysis shows that the liquid produced from the well lost around 17°C during the lifting period and an evaluation of the displacement presents certain temporal shifts between each downhole dynamometer sensor, which are caused by inertia effects of the rod string and its highly dynamic behaviour. Visible fluctuations in a load analysis are caused by increased friction and the results of a comparison of the battery voltage of all 5 sensors over the entire recording period indicate, that the recording times for future field test could be expanded.

The novelty of this thesis is the creation of the MatLab files, which are programmed in a manner to work for future downhole dynamometer field examples and can be used for a complete evaluation of recorded downhole data in a sucker rod pumping system.

Table of Content

Page

1 Introduction and Problem Definition

Nowadays, a large quantity of the world's oil supply is extracted by using a sucker rod pumping system, mostly due to its mechanically simple construction, excellent low volume lifting capacity and a very common and easy operating principle.

Overall, sucker rod pumps show great results in many different environments, yet there are still a number of possible failures that may occur, largely because of the lack of reliable downhole data.

Conventional dynamometers – like an electrical dynamometer, a horseshoe load cell dynamometer or a polished rod transducer dynamometer - are commonly used in the oil and gas industry to produce the surface dynamometer card. The pump card is then generated from the surface card by applying the so-called wave equation, but the interpretation of the results is still open to misunderstandings and prone to errors.

In order to maximize the efficiency of these pumps and limit excessive downtime, gathering and interpreting downhole data will play an ever-increasing role in the future of the oil & gas production process.

This is one of the main reasons why the Montanuniversität Leoben in cooperation with lilatech GmbH started in 2014 to develop a technology to directly record downhole data, the so-called downhole dynamometer sensor (DDS).

A downhole dynamometer sensor is a data logging device equipped with electronics, which enables the measurement of a variety of physically relevant downhole quantities. It is positioned along the sucker rod string and once the rod string is pulled, the recorded downhole data can be analyzed and used to identify the reasons behind failures and help to solve these problems.

In the first part of this thesis, the sucker rod pumping system, conventional dynamometers, and their resulting plots, as well the history, components and working principle of the Downhole Dynamometer Sensors will be explained in detail.

The second part of this thesis will concentrate on the evaluation of a field example with 470 real-life datasets recorded from 5 downhole dynamometer sensors located at different depths in an oil well in the Vienna Basin. The recorded data consist of information about temperature, load, battery voltage and acceleration and the objective of the thesis is to analyze and evaluate the results of the field test by using the computer program MatLab.

2 Sucker Rod Pumping System

Sucker rod pumps, as shown in Figure 1, are the oldest and most commonly used artificial lift system for onshore oil wells in the oil and gas industry. Their working principle is relatively simple and the pumps tend to last for a substantial amount of time. Furthermore, they are also inexpensive and easy to operate.

Figure 1: Picture of a sucker rod pumping system (Aurelio, 2019)

2.1 Selection Criteria and Advantages/Disadvantages of Sucker Rod Pumps

The selection of this type of artificial lift system depends on several factors, as shown in table 1, and should be considered very carefully. Sucker rod pumping systems have a long and proven history of being a reliable choice for most onshore oil wells. Their applications also cover wells with very low downhole pressure, slim holes, multiple completions, high temperatures, viscous oils, and many others (Hein Jr., 1996). Operating personnel is also usually quite familiar with sucker rod pumps and, therefore, are able to handle them more efficiently than other types of artificial lift.

Table 1 shows a comparison of different artificial lift systems:

	Sucker Rod Pumping System	**ESP**	**PCP**	**Gas Lift**	**Hydraulic Jet Pump**
Maximum Operating Rate [BFPD]	6.000	64.000	4.500	50.000	20.000
Maximum Operating Depth [ft]	16.000	15.000	6.000	18.000	15.000
Fluid Gravity [°API]	> 8	> 10	< 40	> 15	> 8
Efficiency [%]	45 - 60	35 -60	50 - 75	10 - 30	10 - 30
Temperature [°C]	Up to 290	Up to 220	Up to 220	Up to 180 - 200	Up to 260 - 320
Noise	moderate	Very low	low	low	low
Hole Deviation	0 to 20°	0 to 2° maximum deviation	Leads to wear and load issues	0 to 50°	0 to 20°
Gas Handling	Use of gas anchor recommended	Up to 40% free gas at pump suction	Free gas leads to gas handling issues	Free gas reduces the amount of injection gas → excellent gas handling	Good gas handling if downhole gas separation below pump intake
Offshore usability	Poor but possible	Good	Poor	Excellent (most common)	Good

Table 1: Comparison of artificial lift systems (Hofstätter, 2019)

As shown in table 1, there are obviously some limitations to the sucker rod pumping systems like the operating rate and also the depth. Other factors to consider are the size of the

casing, tubing & the downhole pump, the strength and size of the rods and the speed with which they can be reciprocated.

In general, the advantages of sucker rod pumps are (Brown, 1982):

- Relatively simple system design
- Units can easily be changed to other wells with minimum cost
- Efficient, simple, and easy for field personnel to operate
- Applicable to slim holes and multiple completions
- Can pump a well down to very low pressure (depth and rate dependent) – excellent low volume lifting capacity
- System usually is naturally vented for gas separation and fluid level soundings
- Good flexibility - can match displacement rate to well capability as the well declines over time.
- Can lift high-temperature and viscous oils
- Can use gas or electricity as power source
- Corrosion and scale treatments are quite easy to perform
- Applicable to pump off control if electrified
- Available in multiple sizes
- Hollow sucker rods are available for slim hole completions and for ease of inhibitor treatment.
- Possibility to have pumps with double valving that pump on both upstroke and downstroke

Disadvantages on the other hand are (Brown, 1982):

- Crooked holes may present a friction problem
- High solids production may be troublesome
- Gassy wells usually show a lower volumetric efficiency
- Depth limitations, primarily because of rod capability
- Obtrusive in urban locations
- Heavy and bulky in offshore operations
- Susceptible to paraffin problems
- Tubing cannot be internally coated for corrosion
- H_2S limits depth at which a large-volume pump can be set
- Limitation of downhole pump design in smaller diameter casings

Another major disadvantage is the inability to directly measure downhole data.

2.2 Working Principle of Sucker Rod Pumps

The general working principle of a sucker rod pump is presented in Figure 2 and all steps are explained in detail:

- **1:** Traveling and standing valve are both closed = bottom of the stroke.
- **1→2**: Expansion step: The plunger picks up the fluid load by moving up. The tubing un-stretches and goes with the plunger due to the transfer of the fluid load from the tubing to the rods → gas expansion and excessive slippage may occur at this point.
- **2→3:** Intake: The standing valve opens in order to take in new fluid → the full fluid load is now carried by the rods.
- **3:** The standing valve closes and the plunger ceases to vacate the chamber = top of the stroke.
- **3→4:** Compression step: The plunger is moving down again → the fluid load changes back from the rods to the tubing → the tubing stretches and the traveling valve is opened at point 4 once the pressure in the chamber is higher than the pump discharge pressure.
- **4:** Discharge: The cycle starts again as the plunger travels through the fluid.

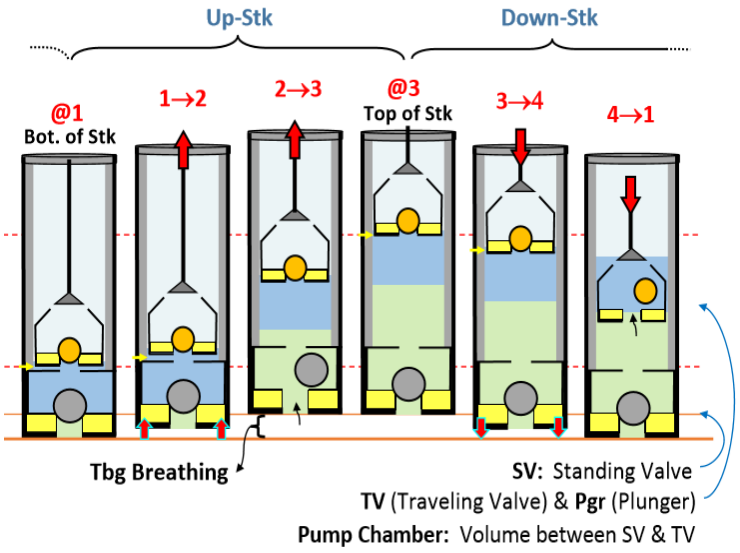


Figure 2: Working principle of a sucker rod pump (Downhole Diagnostic, 2020)

2.3 Components of Sucker Rod Pumping System

A sucker rod pump, as shown in Figure 3, consists of many components, some underground and others above. The most critical ones to the system are:

- Sucker rod string: Several 24- or 30-feet long sucker rods linked by couplings to transmits the mechanical energy from gearbox to the subsurface pump.
- Subsurface pump: Elevates the fluid from the reservoir to the surface. 2 main types are shown in Figure 4: rod (insert) pumps and tubing pumps. Rod pumps are inserted in the production tubing while a tubing pumps is coupled with the production-tubing string.
- Prime mover: Usually an electric motor that is part of the surface pumping unit.
- Gear reducer/gearbox: Used to achieve optimal and sustainable pumping speeds.
- Pump jack: Used to transform circular motion into a vertical reciprocating movement. Different types available: Beam-balanced conventional unit, conventional crank-balanced unit, air-balanced unit, Mark II unit, Reverse Mark II unit, hydraulic pumping unit, low profile pumping unit, etc.
- Subsurface equipment: Tubing, tubing-anchor catchers, sinker bars, tubing rotators, rod centralizers, etc.
- Surface equipment: Polished rod, stuffing boxes, surface valves, etc.

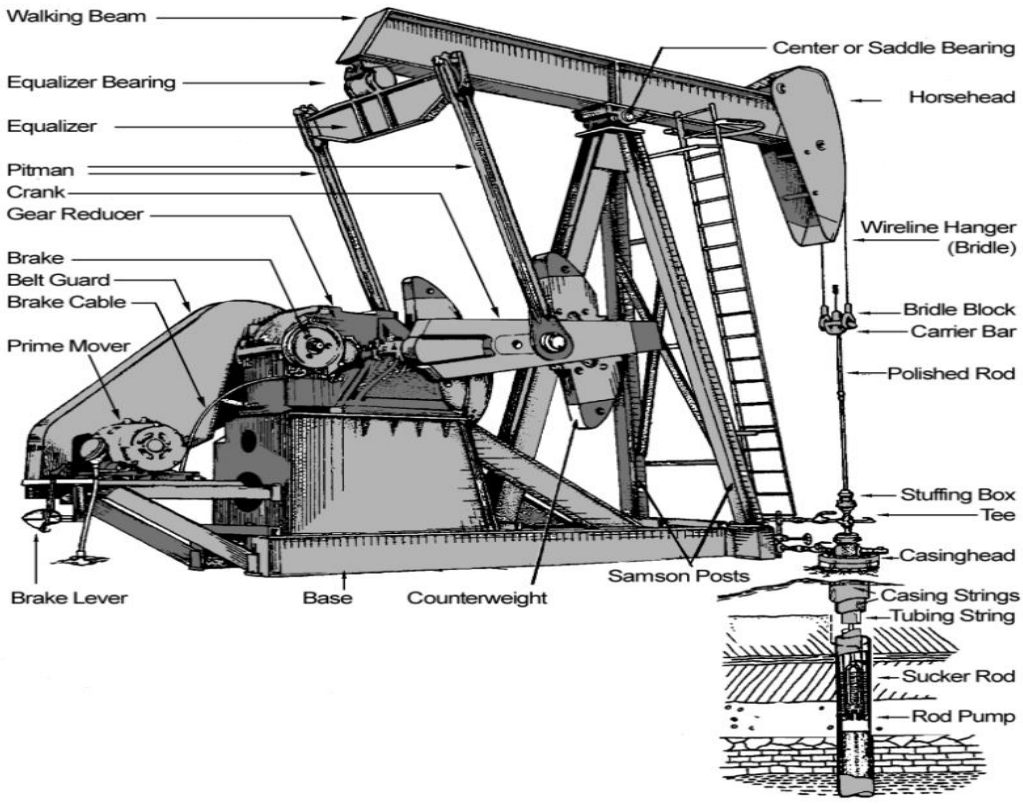


Figure 3: Schematic of a sucker rod pumping system (Ahba et al., 2014)

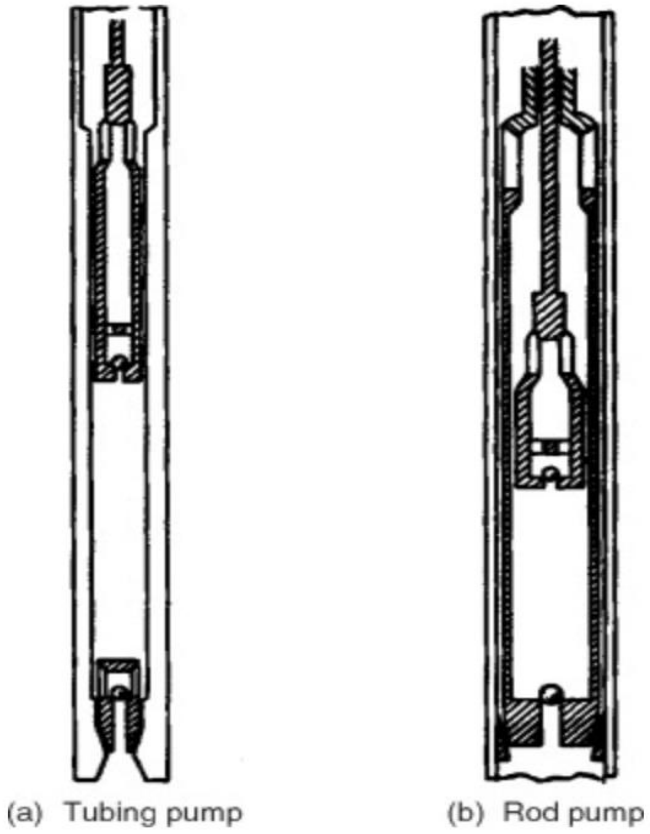


Figure 4: Tubing/rod subsurface pump (Ahba et al., 2014)

2.4 Common Failures of Sucker Rod Pumping Systems

While the sucker rod artificial lift system has been successfully used worldwide over a number of years, there are still a variety of possible failures that may occur under certain conditions like:

- Tensile or fatigue failures (Figure 5)
- Design and operational failures (Figure 6)
- Mechanical failures
- Bent rod failures (Figure 7)
- Surface damage failures (Figure 8)
- Connection failures
- Corrosion-fatigue failures (Figure 9)
- Manufacturing defects

Figure 5: Tensile or fatigue failures (Norris, 2007)

Figure 6: Design and operating failures (Norris, 2007)

Figure 7: Bent rod failures (Norris, 2007)

Figure 8: Surface damage failures (Norris, 2007)

Figure 9: CO_2 corrosion on couplings (Norris, 2007)

Although the specific reasons for these types of failures may vary quite significantly and not all of them will be avoided by the use of downhole dynamometer sensors, the recorded downhole data may be very helpful in identifying the causes of these deficiencies and also to execute troubleshooting in problem wells. It will also aide in improving the design of rod strings in the future.

3 Dynamometer Cards

A dynamometer is a diagnostic instrument for sucker rod pumped wells, that is used as a valuable tool to analyze the pumping performance of a well.

Conventionally, the loads occurring in the rod string are measured at the surface and are used to generate a load versus position diagram. This resulting plot is generally known as the so-called dynamometer card. A difference must be made between 2 types of dynamometer cards, which are shown in Figure 10:

- Surface dynamometer card
- Downhole dynamometer card (pump card)

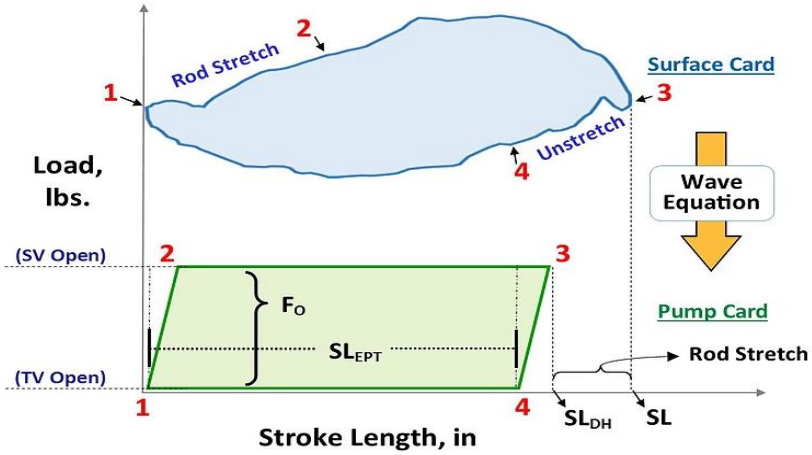


Figure 10: Example of a surface (top) and downhole or pump (bottom) dynamometer card (Downhole Diagnostic, 2020)

The interpretation of these dynamometer cards is a difficult, yet important part of improving the profitability of sucker rod pumps and a correct analysis can lead to (Downhole Diagnostic, 2020):

- Determination of the loads occurring on the pumping unit structure and in the rod string.
- Calculation of the torsional loading on the speed reducer and on the prime mover.
- Determination of the power required to drive the pumping unit.
- After discovering the counterbalance effect, the degree of the unit's counterbalancing can be identified.
- Diagnosis of the conditions and possible malfunctions of the downhole pump and its valves.
- Detection of possible downhole problems or uncertainties
- Information about the pump fillage, pump intake pressure and pump displacement

These are just a few reasons, why it is of critical importance to correctly understand both the surface and the pump card and how to analyse them!

3.1 Surface Dynamometer Card

The surface dynamometer card is a plot of the polished rod load versus its position, where the polished rod load is the sum of all loads downhole. It is therefore influenced by:

- The design of the rod string
- The pump setting depth
- The fluid load on the pump
- The forces of acceleration and viscous friction between rods and fluid
- The buoyant rod weights
- The mechanical friction of the rods on the tubing

While the surface card is a useful tool to present the rod stretch or contraction, all these influencing factors make the interpretation quite difficult, with the main reason for difficulties being the rod vibrations.

To get a better understanding of how surface dynamometer cards are analysed, 5 different examples are shown in Figure 11:

A) An ideal surface card including rod stretch and contraction

B) Another ideal card that is rotated clockwise due to acceleration forces

C) 3 surface cards with rod vibrations. These vibrations result in damped oscillation in the rod string and waves traveling from one end of the string to the other.

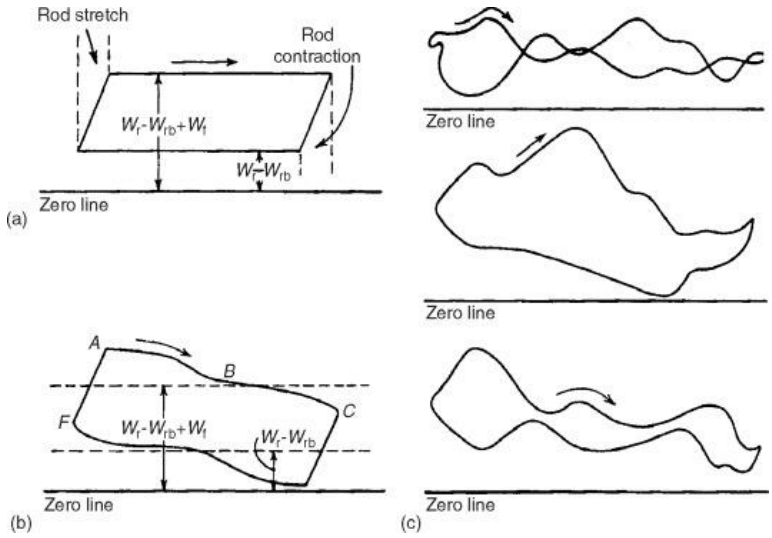


Figure 11: Examples of surface dynamometer cards (Nind, 1964)

3.2 Types of Conventional Dynamometers

Nowadays, mostly electrical dynamometers, as shown in Figure 12, are used in the oil & gas industry.

Figure 12: Electrical downhole dynamometer (Langbauer, 2019)

Electrical downhole dynamometers are able to measure the polished rod load versus the time function and also the displacement versus the time and then those 2 measurements are combined to receive a polished rod load versus displacement graph.

Other conventional oilfield dynamometers that are used in the oil & gas industry are:

- Horseshoe load cells
- Quick-install polished rod transducer (PRT) strain gauges

Horseshoe load cell dynamometer

Figure 13: Installation of a horseshoe dynamometer (Downhole Diagnostic, 2020)

Before the horseshoe (or also called donut) load cells can be installed, the rod string should be stacked out on the wellhead in order to provide enough space for the cells to be inserted (as shown in Figure 8 – the rod load is supported by resting the stack-off “suitcase” on the stuffing box). The load cells are then installed between the clamp of the polished rod and the bridle. Afterwards the well resumes pumping and the true load on the polished rod is measured directly.

Compared to the horseshoe dynamometer, which is used primarily for temporary installations, the donut load cell dynamometer is used in connection with pump-off controllers for continuous dynamometer acquisition.

Horseshoe (or donut) load cell dynamometers provide very accurate solutions, because of the direct measurement of the polished rod, yet there are still several issues associated with these types of dynamometers, especially concerning the installation:

- During the installation, the pumping from the well must be stopped, which may cause a fluid level build-up and a change of the downhole conditions.
- A good functioning brake is a must due to the stacking out of the rod string on the wellhead

- The stacking out also increases the injury hazard to the operator and/or failure of the equipment
- The installation generally takes quite some time to complete and after the data have been recorded, the original condition of the well must be restored
- Raising of the rod string may change the spacing of the pump, which leads to a reduction of the compression rate of the pump and increases the possibility of gas interference
- Raising of the rod string can also cause the plunger to move into an area of the barrel that is not swept and for a pump that has been downhole for a longer period, this increases the amount of scale deposits.

Polished rod transducer dynamometer (PRT)

Figure 14: Installation of a PRT dynamometer (Downhole Diagnostic, 2020)

Compared to the horseshoe load cell dynamometer, the installation of the PRT dynamometer is rather simple and quick and there is no need to stop pumping either.

The measurement of the polished rod load is not done directly, but a clamp-on strain gauge is placed onto the rod to determine the minute changes in the diameter along the stroke

length. The fluid load is carried by the rods on the upstroke and adds additional stress to the rod string, which causes axial stress and ultimately a change in the radial diameter. A relation between the radial strain caused by axial stress is provided by Hooke's Law and it is used to back-calculate the axial load that causes the diameter change (→ fluid load). The surface dynamometer is then constructed after the fluid load and the calculated buoyant rod weight are added together. Finally, the pump card is obtained from the surface card by applying the aforementioned wave equation.

In general horseshoe load cell dynamometers provide more accurate information, but PRT's are much more commonly used, since there are fewer issued concerning their installation.

Nevertheless, there are still a few concerns with PRT dynamometers:

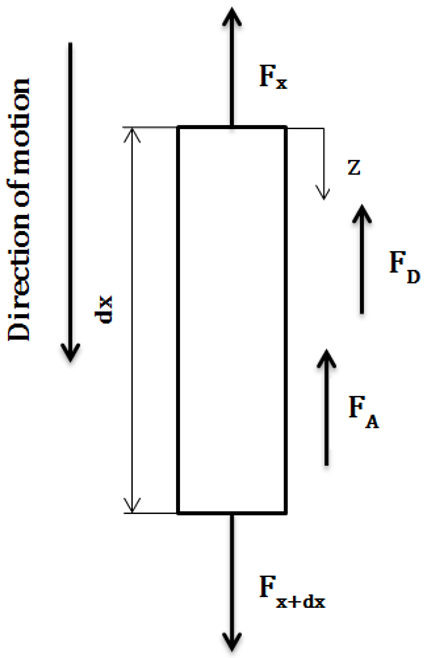
- PRT data are (as mentioned above) less accurate and more qualitative than the data collected from a horseshoe dynamometer
- Polished rod bending, caused by the misalignment of the wellhead and the rod, leads to additional changes in the rod diameter that are "wrongly" interpreted by the PRT as load changes downhole
- Temperature changes on the installed strain gauge lead to an expansion or contraction of the gauge and cause a temperature/load drift

3.3 Wave Equation

While surface dynamometer cards are a valuable tool by itself, it is of much more significance to the petroleum engineer to know what is happening downhole. Therefore, a mathematical equation is used to derive the downhole card from the surface card - the so-called wave equation.

This solution to derive the downhole card from the surface card was established by Dr. Sam Gibbs and it is based on Newtonian dynamics and Hooke's law of elasticity. It illustrates the elastic behaviour of the rod string in its dynamic motion. A downhole friction factor is also assumed and in the end, the results generated from the surface can be transferred down to the pump, which leads to a load versus position plot of the pump's plunger – the so-called downhole dynamometer card or pump card.

To fully understand the wave equation, it is necessary to understand the forces acting on a vertical rod string, which is shown in Figure 15 along with the respective equations. The variable a represents the velocity of sound in the rod, m is the mass, r is the radius, c is the damping coefficient, z represents the displacement at x, E is the modulus of elasticity, ϱ is the density of the rod and x and dx are the variables for the position and the changes in position, respectively. F_D represents the damping force opposing the movement and F_X is the tension force on the upward pull, while F_{X+dx} is the tension force representing the pull from below on the rod element. The variable t represents the time, A is the cross-sectional area of the rod element, μ is the friction coefficient, ε the strain and F_A is the gravitational force.



$$F_A = m\ddot{z} = A\varrho\frac{\partial z^2}{\partial^2 t}dx$$

$$F_D = 2r\pi\mu\frac{1}{rt-r}\frac{\partial z}{\partial t}dx$$

$$F_X = AE\varepsilon = AE\frac{\partial z}{\partial x}$$

$$F_{X+dx} = F_X + \frac{\partial}{\partial x}(F_X)dx + .. = AE\frac{\partial z}{\partial x} + AE\frac{\partial z^2}{\partial^2 x}dx$$

$$\sum_i^i F_i = 0 \rightarrow \frac{\partial z^2}{\partial^2 t} = a^2\frac{\partial z^2}{\partial^2 x} - c\frac{\partial z}{\partial t}$$

Figure 15: Forces on the vertical rod string (Gibbs, 2012)

Some important factors have to be considered for the aforementioned wave equation:

- It is only valid for vertical wells
- Coulomb friction, viscous effects and wax cannot be considered
- The definition of the bottom boundary condition is difficult and limited to simple cases
- Damping coefficients can only be selected artificially, yet have an important influence on the result
- The solution is only one-dimensional, which means that radial movements to the beam axis cannot be determined → buckling cannot be seen directly

The forces acting on a deviated rod string, which are shown in Figure 16, are different to the vertical ones and, therefore, also the wave equation is changing:

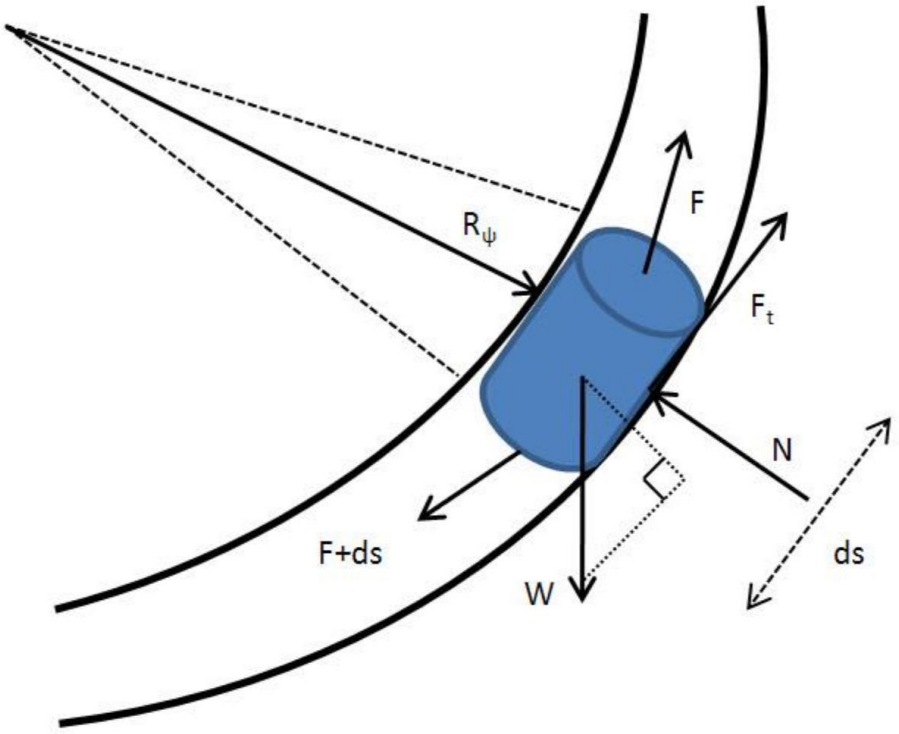


Figure 16: Forces acting on a deviated rod string (Gibbs, 2012)

$$-F + F + ds + \gamma gA\cos\varphi\, ds - A\gamma\frac{\partial^2 z}{\partial t^2}ds - \nu\frac{\partial z}{\partial t}ds - n\mu ds = 0$$

$\rho gA\cos\varphi\, ds$ represents the gravity force acting tangential to the rod direction, $A\rho\frac{\partial z^2}{\partial^2 t}ds$ is the acceleration force, $\nu\frac{\partial z}{\partial t}ds$ represents the viscous friction and the Coulomb friction can be seen in the equation as $n\mu ds$. The variables γ, A, z, t and μ represent the rod material density, cross-sectional area of the rod element, displacement, time and friction coefficient, while ds is the variable for the changes in position. F and F+ds represent the axial forces on

the upward and downward pull and v is the viscous damping coefficient. φ represents the angle of inclination of the rod string and g is the gravity constant.

$$\frac{dM}{ds} + A\gamma \frac{\partial^2 w}{\partial t^2} ds + F_n - \gamma g A \sin\varphi \, ds + S sin\varphi$$

$\frac{dM}{ds}$ represents the bending force, $A\gamma \frac{\partial^2 w}{\partial t^2} ds$ is the acceleration force and F_n demonstrates the friction. $\gamma g A \sin\varphi \, ds$ is the gravity force normal to rod direction and $Fsin\varphi$ demonstrates the normal force. The variables M and w represent the bending moment and the transverse displacement and S is the length measured along the curved rod.

In the end the wave equation for a deviated rod string is as followed, where R is the radius of curvature and a is the sound velocity of the rod:

$$\frac{\partial^2 z}{\partial s^2} - \frac{1}{a^2}\frac{\partial^2 z}{\partial t^2} + \frac{\rho g cos\varphi}{E} - \frac{\nu}{AE}\frac{\partial z}{\partial t} - \frac{\nu}{R}\frac{\partial z}{\partial s} - \frac{\mu \rho g sin\varphi}{E} = 0$$

By using the wave equation, the acquired surface data are mathematically filtered in order to reflect the downhole pump performance each stroke. Everything happening above the pump's plunger, especially the effects of the rod string, are filtered out and the end result is the downhole dynamometer card.

3.4 Downhole Dynamometer Card

The downhole dynamometer card or also called pump card was created first in 1936, when Walton E. Gilbert published his work “An Oil-Well Pump Dynagraph”. Compared to surface dynamometer cards, the value of the downhole dynamometer cards is of more significance, since it is much more important for a production engineer to know what is happening downhole.

Some important points should be considered for the analysis of downhole dynamometer cards:

- Rod stretch and contraction as well as everything else occurring above the pump’s plunger are NOT displayed on the pump card
- Pump card describes just the load on the plunger
- The shape of the card illustrates the way the plunger picks up, holds, and releases the fluid for each stroke
- The shape of the card is also highly dependent on the pressure changes inside the barrel of the pump regarding the movement of the plunger
- Slow load losses during downstroke → Indication that fluid load is gradually released, possibly caused by tubing breathing or gas compression
- Sudden load loss during downstroke → Indication of fluid pounding
- Slow & gradual load pick up during upstroke → Indication of gas expansion, fluid slippage and/or tubing movement

Additional factors influencing the pump card are:

- Tubing movement
- Gas interference
- Excessive fluid slippage
- Incomplete pump fillage

The dimensions of the pump card provide information as well:

- The area represents the work performed by the pump
- The height is a function of the fluid load acting on the plunger and the fluid load is a function of the hydrostatic pressure and the diameter of the plunger

- The length of the pump card represents the plunger stroke length

In order to get a better understanding of how downhole dynamometer cards may look and how to analyse them, Figures 17 and 18 show different pump cards:

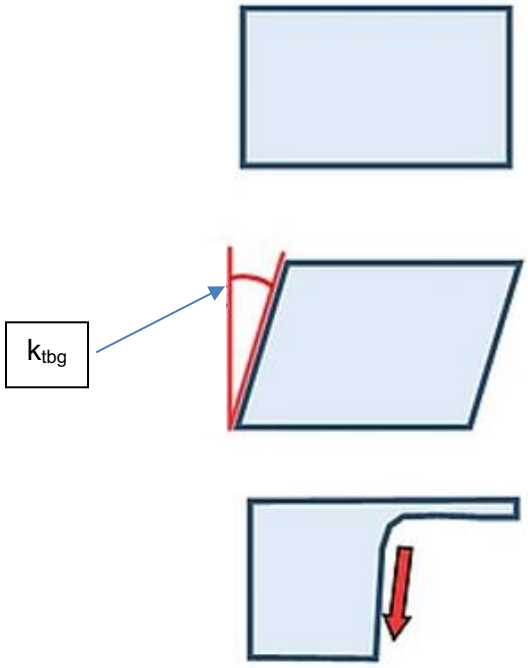


Figure 17: Examples of various downhole dynamometer cards (1) (Downhole Diagnostic, 2020)

Analysis of the pump card examples (From the top to the bottom):

- **Ideal Card:** Fully anchored tubing, 100% liquid fillage & pump in good condition
- **Slanted:** Unanchored tubing indicated by the card being slanted at the point k_{tbg} (Tubing Spring Constant)
- **Fluid Pound:** Sudden impact load. Inefficient and very damaging to pump, rods, tubing and gear box. The impact load causes rod buckling & rod-on-tubing slap

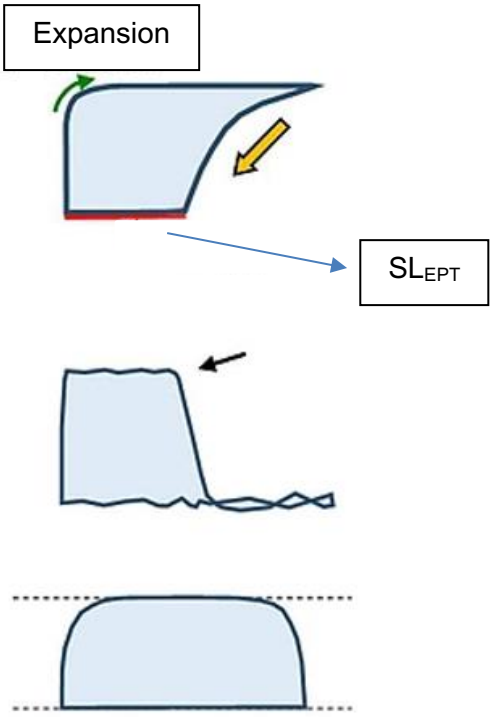


Figure 18: Examples of various downhole dynamometer cards (2) (Downhole Diagnostic, 2020)

- **Gas Interference:** A more gradual load transfer as gas compresses (pneumatic cushioning). Greatly reduces the pumping efficiency and indicates the well is not pumped off
- **Hole in Barrel:** This is a very rare incident, where the bottom of the plunger passes the hole and the hydrostatic pressure is equalized across the plunger causing the fluid load to be lost
- **Worn Pump:** slow to pick up & quick to release the fluid load, due to leaking traveling valve or (severe) plunger/barrel wear

Further information, that can be derived from downhole dynamometer cards also includes:

- Excessive friction, which may be caused by the presence of solids between the barrel and the plunger
- Plunger sticking
- Pump tagging on top or bottom
- Inconsistent valve action, which may be caused by solids or a worn ball and seat

3.5 Pattern Recognition Methods of Downhole Dynamometer Cards

Several theoretical and practical works have been conducted on the topic of using artificial neural networks (ANN), convolutional neural networks (CNN) or Backpropagation Neural Network (BPNN) models to create an automated system for pattern recognition of pump dynamometer cards.

In theory, this would lead to the possibility to anticipate pump problems beforehand and would allow the production engineer to take corrective and preventive measures against these problems.

The desired result is either an increase in the oil production or the reduction of the energy that is needed, while maintaining the same oil production *(*A. M. Felippe de Souza, 2009).

In general, these works can be split up into 3 main groups (Sayed Ali Sharaf, 2018):

1.) **Rule-based methods:**

In order to implement rule-based methods, a number of so-called descriptors for the dynamometer cards must be chosen. Some examples for possible descriptors are:

- Border
- Region
- Centroid
- Curvature
- K-curvature

Afterwards, similarity is calculated by using mathematical tools like the Euclidean distance or the Pearson correlation

2.) **Machine learning with manual feature extraction**

"Machine learning is based on algorithms that can learn patterns from data. *Manual feature engineering is required such as dividing dynamometer cards and then extracting moment invariants for pattern recognition using a support vector machine (SVM)." (*Sayed Ali Sharaf, 2018)

3.) **Deep learning with automatic feature extraction**

Deep learning methods use artificial neural networks (ANN) and convolutional neural networks (CNN) to create an automatic extraction process of dynamometer card features. "*Convolutional neural networks, a class of deep, feed-forward artificial neural networks, use a*

*variation of multilayer perceptron designed to require minimal preprocessing." (*Sayed Ali Sharaf, 2018)

Groups 1 and 2 usually require very extensive research and testing performed by engineers over a large time period and many results still show either limited applications or low performance. Therefore, it is still necessary, that the downhole dynamometer cards are visually inspected by a trained engineer.

Group 3 on the other hand improves the accuracy of the pattern recognition process, because of the automated feature extraction and the use of neural networks. However, in order to use these deep learning methods, the neural networks must be trained from scratch, which could be a very time-consuming effort.

In general, machine learning methods involve 2 major components that have a significant effect on the prediction process (Abdalla, 2018):

- Data acquisition
- Pattern recognition (developed models)

Figure 19 shows the data acquisition and signal processing, where data are acquired by field instruments and afterwards the a surface dynamometer card, which was acquired by multiple sensors placed on surface equipment, is transformed into a set of tension and displacement values by a computational solution for the model. (A. M. Felippe de Souza, 2009)

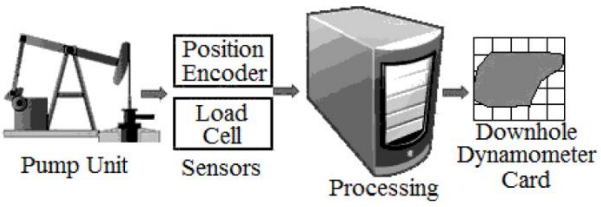


Figure 19: Data acquisition and signal processing (A. M. Felippe de Souza, 2009)

The next step is the pattern recognition, which is shown in Figure 20. It consists of 2 main stages: feature extraction and the classifier.

- *"The **feature extraction** does a transformation of the vector X from an observation space of dimension 'm', which is the downhole dynamometer cards data, into the characteristic space of dimension 'q', where q < m, in order to simplify the classification task." (*A. M. Felippe de Souza, 2009)

- *"The **classifier** consists of an ANN that associates the vector of characteristics Y of dimension 'q' into one of the classes of the decision space which has dimension 'r'." (*A. M. Felippe de Souza, 2009)

The goal of the pattern recognition is to create a characteristic extractor that should be able to eliminate redundant information and enable a linear/nonlinear transformation into the observation space.

To simplify the design of the characteristic extractor, the following factors should be implemented:

- Histograms can be used to extract characteristics, because the pump dynamometer cards are not subjected to rotation.
- Fourier descriptors can be used because the downhole cards have a closed and periodic borderline
- Translation and scale effects are eliminated by the data normalization process
- There is no overlap of other signals, which could lead to interference.

The classifier is then designed as a feed forward artificial neural network with multiple layers and also a supervised training device. (A. M. Felippe de Souza, 2009)

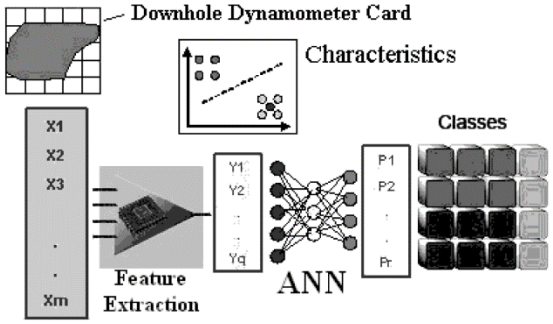


Figure 20: Classifier and characteristic extractor for pattern recognition (A. M. Felippe de Souza, 2009)

An example of a successful pattern recognition model is described in the thesis “Automatic Well Failure Analysis for the Sucker Rod Pumping System”, written by Ramez Abdallah in 2018. A descriptive model based on elliptical Fourier descriptors was developed to reflect the characters of dynamometer cards, which was trained by field data from 4469 dynamometer cards (Abdalla, 2018). The results and conclusion of the thesis state, that the Backpropagation Neural Network (BPNN) model is the superior choice and this pattern recognition model is capable *of predicting the rod pump condition with reasonable accuracy* (Abdalla, 2018).

In the end, the results of various works using pattern recognition methods lead to the conclusion, that “*the generation of artificial standards for training neural networks in order to analyze the pump mechanic system in oil elevation is feasible.” (*A. M. Felippe de Souza, 2009)

4 Downhole Dynamometer Sensors

A major disadvantage of conventional dynamometers concerning the performance evaluation of sucker rod pumping systems is the lack of downhole data.

Furthermore, even predictive and diagnostic software packages, that are used for design and analysing purposes, struggle to solve complex problems, mostly because they are also missing important downhole parameters like friction coefficients (Langbauer et al., 2019).

In order to resolve these issues, the Montanuniversität Leoben in cooperation with lilatech GmbH started in 2014 to develop a technology to directly record downhole data, the so-called downhole dynamometer sensor (DDS).

A downhole dynamometer sensor is a data logging device equipped with electronics, that are able to measure a number of physically important and relevant quantities (Langbauer et al.,2019). The idea behind this innovation is, to analyse the recorded data, once the rod string is pulled, in order to identify the reasons behind failures and solve these problems.

This data may then be implemented in a database to enhance the aforementioned predictive and diagnostic tools. Since the downhole dynamometer sensor can also measure real data, even in complex wells, the goal in the long run is to use them as standard equipment in the sucker rod string.

4.1 History of Downhole Dynamometer Sensors

While it seems, that there is still a long way to go until downhole dynamometer sensors become the norm in the oil & gas industry, the idea of developing a device to measure downhole data directly has long been on the minds of engineers.

In 1986, Glenn Albert, the founder of Albert Engineering, developed the first electronic downhole dynamometer (Glenn Albert, 1995). His company then teamed up with the Sandia National Laboratory to conduct a study, using 5 of these downhole dynamometer tools (DDT). The goal of this study was to find ways to minimize sucker rod failures and also to prove the reliability of the aforementioned wave equation.

The downhole dynamometer built by Albert Engineering was a 12-inch long cylindrical steel probe (Glenn Albert, 1995), that consisted of a strain gauges, pressure & temperature gauges and an accelerometer, that was capable of measuring load, acceleration, temperature and pressure at the deployed position in the sucker rod string. 2 integration steps of the acceleration then lead to the position and ultimately to a load versus position downhole dynagraph.

The study was conducted on 6 different vertical wells, which were hand-picked, because of their downhole or pumping conditions and all measurements were taken while the wells were

pumping. It is important to note, that nowadays, the complexity of rod pumped wells is far more challenging than simply vertical ones.

Nevertheless, the results of this study proved, that the wave equation is an effective solution and if solved with appropriate damping, it provides an accurate and reliable way to compute downhole cards (Glenn Albert, 1995).

Other benefits named by Sandia National Laboratory included an easy installation and removal and if used properly, operators should be able to reduce and eliminate premature failure of surface and/or downhole equipment.

The downhole dynamometer tools (DDT) developed by Glenn Albert can also be viewed as direct ancestors to the downhole dynamometer sensors (DDS) developed in 2014.

4.2 DDS Value Chain

The value chain for the downhole dynamometer can be seen in Figure 21 and involves 4 phases:

- Record
- Process
- Interpret
- Implement

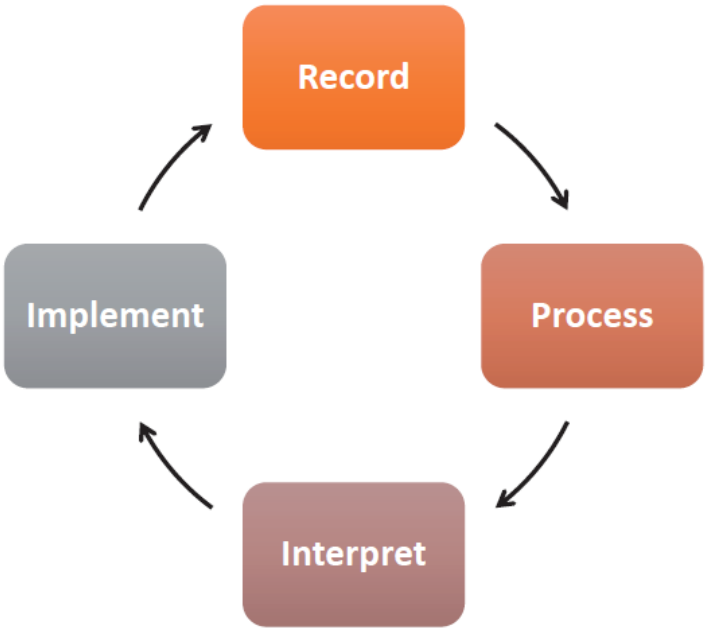


Figure 21: DDS Value Chain (Langbauer et al., 2019)

During the recording stage, the installation depth of the downhole dynamometer sensors, as well as the sampling interval and frequency are chosen along with any additional surface measurements, that are considered necessary.

Afterwards, the processing phase consists of copying and converting the data from the DDS using the aforementioned calibration.

In the 3rd step of the DDS value chain, the interpretation of the collected data includes the following analyses (Langbauer et al., 2019):

- Stroke identification
- Static and flowing gradient analysis → temperature analysis
- Load analysis
- Stress analysis
- Time & depth behavior
- Friction analysis
- Buckling investigation
- Motion analysis
- Evaluation of production rate
- Analysis of volumetric efficiency
- Load transfer of the standing and traveling valve
- Analysis of the power consumption
- Interpretation of the pump card

The last step of the DDS value chain is the implementation of important findings. All useful discoveries and possible optimizations are outlined in a summary. A predictive software may be used in conjunction with the measured values in order to improve the operation mode of the pump and enhance the sucker rod string. Last but not least, once the necessary adjustments are implemented, the cycle of the DDS Value chain repeats itself.

It is important to note that, while a downhole dynamometer sensor provides the engineer with a great understanding of the downhole environment, it is still recommended to obtain additional information like (Langbauer et al., 2019):

- Rod string composition

- Tubing & downhole pump designation
- Tubing pressure
- Casing pressure
- Surface dynamometer measurement
- Surface production data
- Fluid composition

The DDS together with this additional data will provide the engineer with a full understanding of the status of the sucker rod pump and help eliminate many uncertainties.

4.3 Components and Dimensions of Downhole Dynamometer Sensors

A downhole dynamometer sensor is positioned along the sucker rod string and on the outside, it consists of 3 different parts, which are shown in Figure 22:

- Sensor housing
- Accumulator housing
- 7/8'' sucker rod pin connections (pin couplings)

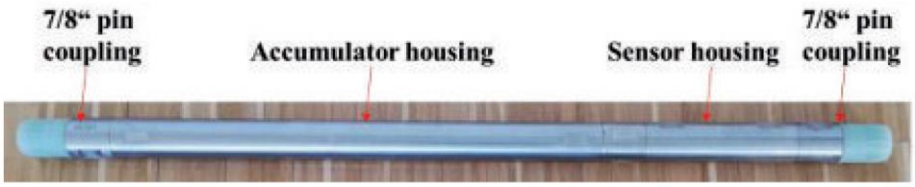


Figure 22: Downhole dynamometer sensor (DDS) (Langbauer et al., 2019)

A downhole dynamometer sensor consists of the sensor housing, which has a diameter of 43mm and the length of the device, excluding the 7/8” couplings on each side, is 795mm. A DDS weighs around 7kg and the maximum load it can take is 20kN. Maximum ambient pressure and ambient temperature are 15MPa and 60°C, respectively and the whole device is EX-certified.

Table 2 provides an overview about the specifications of a downhole dynamometer sensors (Langbauer et al., 2019):

Measurement Duration Capacity	50 days (at continuous sampling rate of 25 Hz)
Data Storage Capacity	4 GB
Data Recovery	Via micro USB plug
Measurement Cycle Modes	1.) Continuous measurements 2.) Sleep & wake up option 3.) Interval measurements
Diameter	43mm
Length	795mm (excluding pin couplings)
Weight	7kg
Maximum Load	20kN
Maximum Ambient Pressure	15MPa
Maximum Ambient Temperature	60°C
Battery Pack	5 Rechargeable batteries with protective circuit against overcharging – 4,5Ah each

Table 2: DDS specifications (Langbauer et al., 2019)

Inside the housing, the downhole dynamometer sensor is comprised of (Langbauer et al. 2019):

- 3 temperature compensated, high precision strain gauges to measure forces of up to 20 kN with a resolution of about 10 N
- Three-dimensional accelerometer that measures the acceleration in x-, y- and z-direction
- Temperature sensor that is designed to withstand the difficult downhole environments and measure the downhole temperatures
- Flash drive to internally store data, which later can be transmitted to a PC for the analysis of the collected downhole data
- Low-power 32-bit Microcontroller that is used for a quick and efficient processing of the data and it also provides the data with a timestamp
- Auxiliary electronics

Another important part is a 24-bit analog-to-digital converter. It is combined with an instrumentation amplifier, which is used to digitize even very small analog signals. Supremely accurate, temperature compensated reference voltages are also applied to accurately measure these signals (Langbauer et al., 2019).

4.4 DAQ Software for Data Acquisition

In order to acquire the aforementioned 3-dimensional acceleration data and also the load signals with the downhole dynamometer sensors, the development of a firmware and user interface had to be completed for the configuration of the so-called DAQ software (Langbauer et al., 2019).

DAQ (or also called DAS) stands for “Data Acquisition System”. It is usually controlled by a software and used to transform analog waveforms into digital numerical values for processing. The system components commonly involve:

- Multiple sensors used for the conversion of physical parameters into electrical signals
- Signal conditioning circuitry in order to transform sensor signals into a certain form, which can then be converted to digital values.
- Analog-to-digital converters for transformation of conditioned sensor signals to digital numerical values.

Important functional components of this firmware include:

- Power consumption
- Cache Management Unit
- Trigger generation & timeout monitoring
- Data conversion

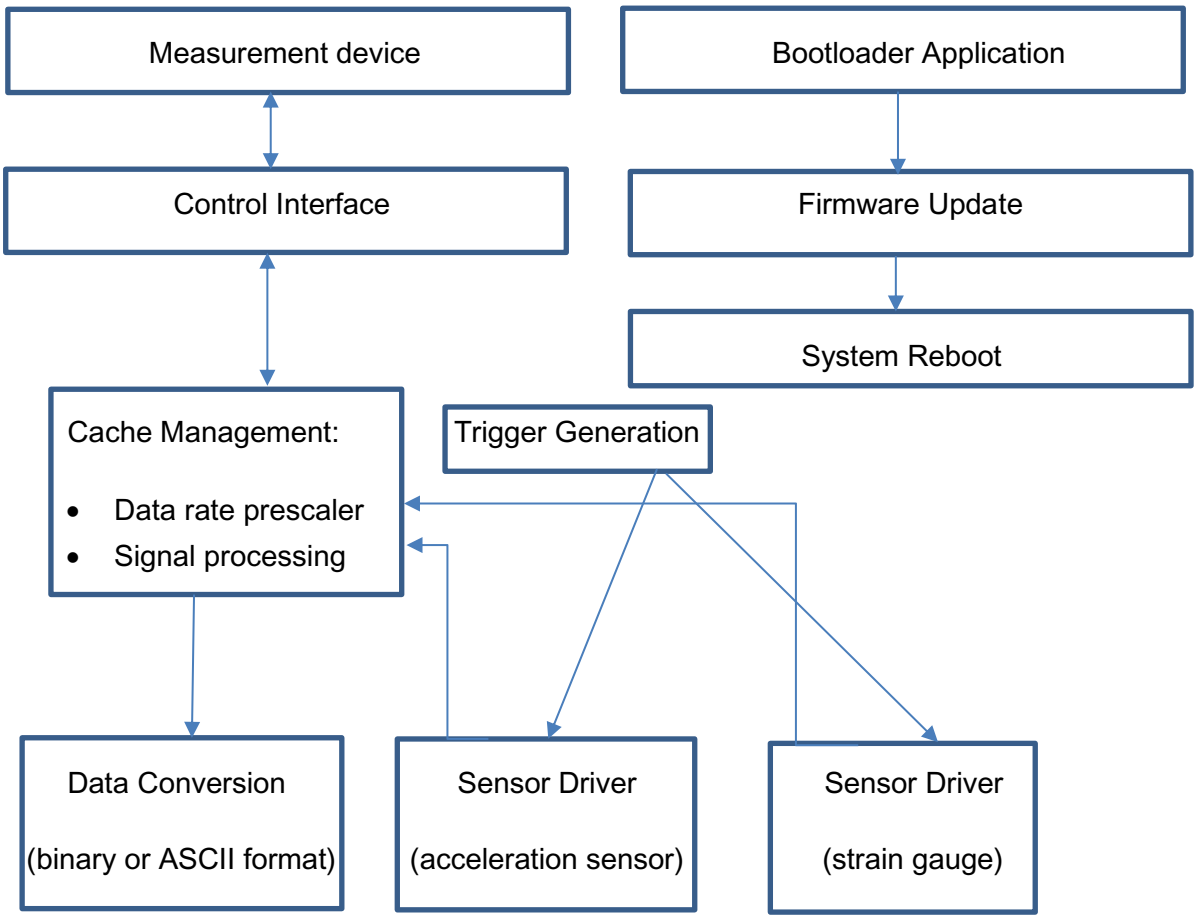
Table 3 shows an overview of important details of these components (Langbauer et al., 2019):

Functional Component	**Important Details**
Power Consumption	In order to be as effective as possible, the goal is to achieve the longest possible running time with a low energy consumption. Therefore, the processor is paused and the acceleration sensor, as well as the supply of the strain gauge bridge are switched off in between the sample period. The start of the software and recording is done either by the alarm of an integrated real-time clock or by USB communication. Using the real-time clock alarm has the benefit, that the system remains switched off until it is automatically started up again by the configurated start time. While the system is off, power consumption is tremendously low, since only the real-time alarm clock remains switched on.
Cache Management Unit	The management of the cache unit follows a first-in-first-out ring storage strategy. Storage blocks are used to structure the cache memory and all measurement values are saved in binary form. A pre-scaler is used to regulate the maximum data rate of the measurement signal and low-pass filtering is employed to avoid aliasing.
Trigger Generation	The use of a 16-bit timer ensures a standard time basis for all measurement channels. A hardware pre-scaler connects the system clock with the clock input of the timer. *The time hardware can automatically trigger interrupts with the same frequency as the fastest measurement channel[8].* The Interrupt Service Routine is carried out with the highest priority in order to achieve a continuous addition of measurement signals.
Data Conversion	The DAQ software works with FAT12, FAT16 and FAT32 data systems and cache memory measurements can be stored in 2 formats: • Binary (used for high sampling rates and fast recording of data) • Conversion to ASCII format Important measurement settings like units, sampling rates, number of channels, data type, etc. are stored in a header as well.

Table 3: Details of functional components of the DAQ Software

Another essential part of the DAQ software is to provide a uniform and precise time basis to implement a steady online and offline communication between the downhole dynamometer sensor and a computer (Langbauer et al., 2019). The result should lead to the same time basis for data stored on the internal flash drive of the sensors as well as for the data on the USB interface.

The following block diagram shows an overview of the proceedings of the firmware:



Important configuration steps concerning the sampling rate, measurement range and resolution are accomplished through interface functions, which *allow to direct each initialized measurement channel to the internal flash memory and/or the communication channels* (Langbauer et al., 2019). New functionalities may also be added by means of a software update process, making the DAQ software a very important part of the process to analyze data gathered from the downhole dynamometer sensors.

5 MatLab Programs for DDS Data Evaluation

In order to carry out the downhole dynamometer data evaluation, the programming platform MatLab was used to generate various files to process and interpret recorded data from a field example.

In general, MatLab is a proprietary multi-paradigm programming language that directly expresses matrix and array mathematics. It offers a numerical computing environment that can be used for (Mathworks.com, 2020):

- Developing and implementing algorithms
- Data analysis
- Creation of models and applications
- Plotting of functions and data
- Matrix manipulations

Simulink, a sub-program of MatLab that provides a graphical multi-domain simulation and model-based design for embedded and dynamic systems, was also used for the data evaluation of the downhole dynamometer sensors.

5.1 Explanation of Datasets

Over the duration of the field testing in the Vienna Basin oil well, the 5 sensors provided a total of 471 datasets. The name of a dataset always consists of a number in the beginning, which represents the individual sensor (1, 2, 3, 4 or 6), followed by the letters “DAQ” for the DAQ Software, which was explained in an earlier chapter. The last digits stand for the hour of the recording. It should be noted again that the recordings were taken in 1-hour intervals, so for example, the dataset 1DAQ2 represents the second hour of recording of downhole dynamometer 1, but since there is 1 hour of “sleep” between each recording, it could be interpreted as the actual 3rd hour of the field test.

- **DDS1:** 1DAQ1 to 1DAQ91 = 91 hours of recording
- **DDS2:** 2DAQ1 to 2DAQ48 = 48 hours of recording
- **DDS3:** 3DAQ1 to 3DAQ83 = 83 hours of recording
- **DDS4:** 4DAQ1 to 4DAQ131 = 131 hours of recording
- **DDS6:** 6DAQ1 to 6DAQ119 = 119 hours of recording

Once the datasets have been implemented in MatLab, they can be found in the workspace of the program under the name “ConvertedData”. After clicking on “Data”, the measured data of the downhole dynamometer sensors can be found. As shown in Figure 23, the measured data consist of 11 rows. 8 of these rows (row 3 to 10) each provide 180000 datapoints, representing a total measurement of 3600 seconds with 0,02 seconds of time increment between each datapoint.

Variables - ConvertedData.Data.MeasuredData

ConvertedData | ConvertedData.Data | ConvertedData.Data.MeasuredData

ConvertedData.Data.MeasuredData

Fields	Name	Data	Total_Samples	Property
1	'Root'	[]	0	*1x1 struct*
2	'Messdaten'	[]	0	[]
3	'Messdaten/Acceleration-X'	*180000x1 double*	180000	*1x12 struct*
4	'Messdaten/Acceleration-Y'	*180000x1 double*	180000	*1x12 struct*
5	'Messdaten/Acceleration-Z'	*180000x1 double*	180000	*1x12 struct*
6	'Messdaten/Battery Voltage'	*180000x1 double*	180000	*1x12 struct*
7	'Messdaten/Analog 1'	*180000x1 double*	180000	*1x12 struct*
8	'Messdaten/Analog 2'	*180000x1 double*	180000	*1x12 struct*
9	'Messdaten/Analog 3'	*180000x1 double*	180000	*1x12 struct*
10	'Messdaten/Analog 4'	*180000x1 double*	180000	*1x12 struct*
11	'Kalibrierungsdaten'	[]	0	[]

Figure 23: Measured data of a DDS dataset

The column “Property” also provides important information about each row, such as the start time of the recording, the aforementioned time increment of 0,02 seconds (which is the same for every dataset) and also the unit of the measurement. An example can be seen in Figure 24:

Variables - ConvertedData.Data.MeasuredData(3).Property

ConvertedData | ConvertedData.Data | ConvertedData.Data.Meas

ConvertedData.Data.MeasuredData(3).Property

Fields	Name	Value
1	'wf_start_time'	'25-Aug-2013 12:00:00'
2	'wf_start_offset'	0
3	'wf_increment'	0.0200
4	'wf_samples'	33
5	'NI_ChannelName'	'Acceleration-X'
6	'NI_ExpXDimension'	't'
7	'wf_xname'	'Zeit'
8	'wf_xunit_string'	's'
9	'NI_UnitDescription'	'g'
10	'unit_string'	'g'
11	'description'	'KanalID=0'
12	'groupname'	'Messdaten'
13		

Figure 24: Properties of a dataset

The unit of each measurement is of particular importance and an overview is provided in table 4:

Name in Dataset	Measurement of	Unit
‘Messdaten/Acceleration-X’	Acceleration in x-direction	g-force (g)
‘Messdaten/Acceleration-Y’	Acceleration in y-direction	g-force (g)
‘Messdaten/Acceleration-Z’	Acceleration in z-direction	g-force (g)
‘Messdaten/Battery Voltage’	Battery Voltage of DDS	Millivolt (mV)
‘Messdaten/Analog1’	Strain Gauge 1 for load measurement	Millivolt (mV)
‘Messdaten/Analog2’	Strain Gauge 2 for load measurement	Millivolt (mV)
‘Messdaten/Analog3’	Strain Gauge 3 for load measurement	Millivolt (mV)
‘Messdaten/Analog4’	Temperature	Millivolt (mV)

Table 4: Units of measured data

5.2 Calibration of Sensors and Datasets

An important part before the installation of the DDS is the calibration of sensors as it is recommended to be done before and after every run. The very first calibration of a DDS happens immediately after the device is manufactured, but the difficult downhole environment may lead to material creeping and therefore recalibration is always necessary.

In general, calibration of a downhole dynamometer sensor is done for:

- Temperature
- Load – axial load, fluid pressure and total head are also weighed in
- Acceleration

Measurements can be done continuously or be programmed to turn on and off at specified intervals. It is also advisable to finish the programming and locking of the DDS device prior to the installation in order to avoid delays during it.

Once the data from the calibrated downhole dynamometer sensors have been implemented into MatLab, it is necessary to also complete a calibration for each individual measurement. For this step, 3 different MatLab programs were used:

1. **Calibration of acceleration:** the acceleration data in x-, y- and z-direction is calibrated by a program called “AccelerationCalibration”, which executes a linear coordinate transformation for the measured data. The new coordinates are linear functions of the original ones and a matrix multiplication is used to multiplicate the old coordinate vectors. The origin of the new coordinate system is equal to the old one. There are small differences in the calibration for each sensor, therefore “if-functions” were implemented to differ between DDS1, DDS2, DDS3, DDS4 and DDS6. In addition to the calibration, the unit was also change from “g” to m/s^2 by multiplying the data by 9,81.

2. **Calibration of temperature:** the temperature calibration (file name = “TemperatureCalibration”) is quite straight forward. There are 4 values in the original unit of millivolt for each downhole dynamometer sensor, which correspond to 4 values in the desired unit of Celsius.

3. **Calibration of strain gauges:** the strain gauge calibration (file name = “DMSCalibration”) carries out a coordinate transformation as well by using pressure values. The program also implements the ambient pressure values around the sensors and subtracts from laboratory values. Each sensor again corresponds to different values and through the whole calibration process, the unit was also changed to Newton. In total 3 strain gauges are available: DMS1, DMS2 & DMS3

“AccelerationCalibration” MatLab file:

```
function [y]=AccelerationCalibration(Nummer,wert)


wert=wert';

if Nummer == 1
   A=[0.999 -.05 -0.036 0 0 0 0 0 0; 0 0 0 0.999 -.05 -.036 0 0 0;0 0 0 0 0 0 .999 -.05 -.036;
      -.047 -.999 -.047 0 0 0 0 0 0;0 0 0 -.047 -.999 -.019 0 0 0;0 0 0 0 0 0 -.47 -.999 -.019;
      -.02 -.049 .999 0 0 0 0 0 0;0 0 0 -.02 -.049 .999 0 0 0;0 0 0 0 0 0 -.02 -.049 .999];
   B=[1 0 0 0 1 0 0 0 1]';
end

if Nummer == 2
   A=[0.999 -.005 -0.048 0 0 0 0 0 0; 0 0 0 0.999 -.0005 -.048 0 0 0;0 0 0 0 0 0 .999 -.005 -.048;
      -.029 -.998 -.048 0 0 0 0 0 0;0 0 0 -.029 -.998 -.048 0 0 0;0 0 0 0 0 0 -.29 -.998 -.048;
      .01 0 0.99 0 0 0 0 0 0;0 0 0 .01 0 .99 0 0 0;0 0 0 0 0 0 .01 0 .99];
   B=[1 0 0 0 1 0 0 0 1]';
end

if Nummer == 3
   A=[0.999 .01 -0.034 0 0 0 0 0 0; 0 0 0 0.999 -.01 -.034 0 0 0;0 0 0 0 0 0 .999 .01 -.034;
      .02 .9997 .015 0 0 0 0 0 0;0 0 0 .02 .9997 .015 0 0 0;0 0 0 0 0 0 .02 -.9997 -.015;
      .015 .019 .9997 0 0 0 0 0 0;0 0 0 .015 0.019 .9997 0 0 0;0 0 0 0 0 0 .015 0.019 .9997];
   B=[1 0 0 0 1 0 0 0 1]';
end

if Nummer == 4
   A=[0.9997 .015 -0.02 0 0 0 0 0 0; 0 0 0 0.9997 .015 -.02 0 0 0;0 0 0 0 0 0 .9997 .015 -.02;
      .019 .999 .029 0 0 0 0 0 0;0 0 0 .019 .999 .029 0 0 0;0 0 0 0 0 0 .019 .999 .029;
      0 0 1 0 0 0 0 0 0;0 0 0 0 0 1 0 0 0;0 0 0 0 0 0 0 0 1];
   B=[1 0 0 0 1 0 0 0 1]';
end

if Nummer == 5
   A=[0.999 .043 0.014 0 0 0 0 0 0; 0 0 0 0.999 .043 .014 0 0 0;0 0 0 0 0 0 .999 .043 .014;
      .005 .99998 .005 0 0 0 0 0 0;0 0 0 .005 .99998 .005 0 0 0;0 0 0 0 0 0 .005 .99998 .005;
      .005 .005 .99998 0 0 0 0 0 0;0 0 0 .005 .005 .99998 0 0 0;0 0 0 0 0 0 .005 .005 .99998];
   B=[1 0 0 0 1 0 0 0 1]';
end

if Nummer == 6
   A=[0.9996 0 -0.029 0 0 0 0 0 0; 0 0 0 0.9996 0 -0.029 0 0 0;0 0 0 0 0 0 0.9996 0 -0.029;
      -.019 -.9996 -.019 0 0 0 0 0 0;0 0 0 -.019 -.9996 -.019 0 0 0;0 0 0 0 0 0 -.019 -.9996 -.019;
      .005 .005 .99998 0 0 0 0 0 0;0 0 0 .005 .005 .99998 0 0 0;0 0 0 0 0 0 .005 .005 .99998];
   B=[1 0 0 0 1 0 0 0 1]';
end

x=A\B;

K=[x(1) x(2) x(3);x(4) x(5) x(6); x(7) x(8) x(9)];

y=K*wert;
```

“TemperatureCalibration” MatLab file:

```
1   function [a] = TemperatureCalibration(Nummer)
2   %Temperature in Celsius
3   y=[27 36.5 45 55];
4
5   %Calibration for area 1
6   if Nummer==1
7      x=[732.5 753.5 772 793]; %values in mV
8   end
9
10  %Calibration for area 2
11  if Nummer==2
12     x=[582 597 612 628.5];%values in mV
13  end
14
15  %Calibration for area 3
16  if Nummer==3
17     x=[583.5 599 614.5 631]; %values in mV
18  end
19
20  %Calibration for area 4
21  if Nummer==4
22     x=[576.5 592 607 623.5]; %values in mV
23  end
24
25  %Calibration for area 6
26  if Nummer==6
27     x=[577 592.5 608 624.5]; %values in mV
28  end
29
30  a = polyfit(x,y,1);
```

“DMSCalibration” MatLab file:

```
function [a] = DMSCalibration(Nummer)

%Nummer=1;

%Messgerät #4
druck=[0,2.1,4.1,6,8,10,12.1,14,16];
DMS1=[-434,-422,-404,-388,-372,-355,-338,-319,-303];
DMS2=[-266,-254,-236,-219,-202,-185,-168,-148.5,-132];
DMS3=[42,51,69,86,102,119,136,155,171];

p_gerade(1,:)=polyfit(druck,DMS1,1);    %mV - Änderung je MPa - Druckänderung
p_gerade(2,:)=polyfit(druck,DMS2,1);
p_gerade(3,:)=polyfit(druck,DMS3,1);

pt=0.4;
Dichte=1020; %Workover Flüssigkeit während Behandlung
TVD=863.1;
pa=pt+Dichte*9.81*TVD*10^-6;
Fa=22.2^2*pi/4*pa;

if Nummer==1
    TVD=4;
    p=pt+Dichte*9.81*TVD*10^-6;
    wert=30400-Fa;
    reduction(1)=p_gerade(1,1)*p;
    reduction(2)=p_gerade(2,1)*p;
    reduction(3)=p_gerade(3,1)*p;
    F=26.42^2*pi/4*p;
elseif Nummer==2
    TVD=19.9;
    p=pt+Dichte*9.81*TVD*10^-6;
    wert=29833-Fa;
    reduction(1)=p_gerade(1,1)*p;
    reduction(2)=p_gerade(2,1)*p;
    reduction(3)=p_gerade(3,1)*p;
    F=26.42^2*pi/4*p;
elseif Nummer==3
    TVD=485;
    p=pt+Dichte*9.81*TVD*10^-6;
    wert=13515-Fa;
    reduction(1)=p_gerade(1,1)*p;
    reduction(2)=p_gerade(2,1)*p;
    reduction(3)=p_gerade(3,1)*p;
    F=26.42^2*pi/4*p;
elseif Nummer==4
    TVD=861.3;
    p=pt+Dichte*9.81*TVD*10^-6;
    wert=898-Fa;
    reduction(1)=p_gerade(1,1)*p;
    reduction(2)=p_gerade(2,1)*p;
    reduction(3)=p_gerade(3,1)*p;
    F=26.42^2*pi/4*p;
elseif Nummer==6
   TVD=877.1;
   p=pt+Dichte*9.81*TVD*10^-6;
    %Fa=22.2^2*pi/4*p
    %Fb=2*0.3*p*43^2*pi/4
```

```
58      %Fd=p*(43^2-22.2^2)*pi/4
59      wert=358-Fa;
60      reduction(1)=p_gerade(1,1)*p;
61      reduction(2)=p_gerade(2,1)*p;
62      reduction(3)=p_gerade(3,1)*p;
63      F=26.42^2*pi/4*p;
64    end
65
66    ye=[wert];
67    yls=[150000 100000 50000 0 -50000 -100000 -150000];
68
69    %Kalibrierung 1
70    if Nummer==1
71       %xe=[-226.5;-242;-285.5];
72       xe=[-221;-237;-280.5];
73       xls=[-975 -686.8 -403 -96 161.5 449.3 737.5; -1001.5 -690.3 -380.3 -66 247 530 814; -1061.5 -
74    756.5 -449.5 -162 156.3 446 732.5];
75       xls(1,:)=xls(1,:)-reduction(1);
76       xls(2,:)=xls(2,:)-reduction(2);
77       xls(3,:)=xls(3,:)-reduction(3);
78       x=[xls(1,1) xls(1,2) xls(1,3) xls(1,4) xe(1) xls(1,5) xls(1,6) xls(1,7);xls(2,1) xls(2,2) xls(2,3) xls(2,4)
79    xe(2) xls(2,5) xls(2,6) xls(2,7);xls(3,1) xls(3,2) xls(3,3) xls(3,4) xe(3) xls(3,5) xls(3,6) xls(3,7)];
80       yls=yls-F;
81    end
82    %Kalibrierung 2
83    if Nummer==2
84       xe=[-294;138;40]
85       xls=[492 188 -116.5 -445 -730 -1026 -1324;923 617.5 314.5 10 -306.5 -609.5 -922;789 480.5 177 -
86    133 -444.5 -737 -1030];
87       xls(1,:)=xls(1,:)+reduction(1);
88       xls(2,:)=xls(2,:)+reduction(2);
89       xls(3,:)=xls(3,:)+reduction(3);
90       x=[xls(1,1) xls(1,2) xls(1,3) xls(1,4) xe(1) xls(1,5) xls(1,6) xls(1,7);xls(2,1) xls(2,2) xls(2,3) xls(2,4)
91    xe(2) xls(2,5) xls(2,6) xls(2,7);xls(3,1) xls(3,2) xls(3,3) xls(3,4) xe(3) xls(3,5) xls(3,6) xls(3,7)];
92       yls=yls+F;
93    end
94    %Kalibrierung 3
95    if Nummer==3
96       xe=[92.5;490;-33.5];
97       xls=[831 520 210 -84 -401 -706 -1013.5; 1173 863.8 606.3 270 -64.3 -359.5 -650; 744 438.3 110.3 -
98    184 -497 -796.5 -1094.5];
99       %    xls=[831 520 210 -70 -401 -706 -1013.5; 1173 863.8 606.3 265 -64.3 -359.5 -650; 744 438.3
100   110.3 -160 -497 -796.5 -1094.5];
101      xls(1,:)=xls(1,:)+reduction(1);
102       xls(2,:)=xls(2,:)+reduction(2);
103       xls(3,:)=xls(3,:)+reduction(3);
104      x=[xls(1,1) xls(1,2) xls(1,3) xls(1,4) xe(1) xls(1,5) xls(1,6) xls(1,7);xls(2,1) xls(2,2) xls(2,3) xls(2,4)
105   xe(2) xls(2,5) xls(2,6) xls(2,7);xls(3,1) xls(3,2) xls(3,3) xls(3,4) xe(3) xls(3,5) xls(3,6) xls(3,7)];
106      yls=yls-F;
107   end
108
109   %Kalibrierung 4
110   if Nummer==4
111      xe=[-398.5;-221;110];
112      xls=[492 180.8 -131.8 -434 -770 -1071 -1366; 663 350 39 -266 -581.3 -883.3 -1180; 925 621.3
113   319.3 42 -285 -576.5 -869];
114      xls(1,:)=xls(1,:)+reduction(1);
115      xls(2,:)=xls(2,:)+reduction(2);
116      xls(3,:)=xls(3,:)+reduction(3);
117      x=[xls(1,1) xls(1,2) xls(1,3) xls(1,4) xe(1) xls(1,5) xls(1,6) xls(1,7);xls(2,1) xls(2,2) xls(2,3) xls(2,4)
```

```
xe(2) xls(2,5) xls(2,6) xls(2,7);xls(3,1) xls(3,2) xls(3,3) xls(3,4) xe(3) xls(3,5) xls(3,6) xls(3,7)];
    yls=yls+F;
end

%Kalibrierung 6
if Nummer==6
    %xe=[62;34.5;97.5];
    xe=[62.5;35;98];
    xls=[-763.5 -458.8 -149.5 142 452 749 1041;-799 -490 -185 129 379 668 954;-756 -451 -143.8 168
471.3 759 1050.5];
    xls(1,:)=xls(1,:)-reduction(1);
    xls(2,:)=xls(2,:)-reduction(2);
    xls(3,:)=xls(3,:)-reduction(3);
    x=[xls(1,1) xls(1,2) xls(1,3) xls(1,4) xe(1) xls(1,5) xls(1,6) xls(1,7);xls(2,1) xls(2,2) xls(2,3) xls(2,4)
xe(2) xls(2,5) xls(2,6) xls(2,7);xls(3,1)...
        xls(3,2) xls(3,3) xls(3,4) xe(3) xls(3,5) xls(3,6) xls(3,7)];
   % x=[-763.5 -458.8 -149.5 60 142 452 749 1041;0 0 0 0 0 0 0 0;-756 -451 -143.8 96 168 471.3 759
1050.5];
    yls=yls-F;
end

y=[yls(1) yls(2) yls(3) yls(4) wert yls(5) yls(6) yls(7)];

for i=1:3
   a(i,:)=polyfit_exact_and_ls(xe(i,:),ye,xls(i,:),yls,1);
   %a(i,:)=polyfit(x(i,:),y,1);
    f(i,:)=polyval(a(i,:),x(i,:));
end

if Nummer==1
    %Nr 1_1
    %a(1,:)=[-186.6 -14335];
    %a(2,:)=[-161.3 -11320];
    %a(3,:)=[-171.4 -21159];
    %Nr 1_2
    %a(1,:)=[-176.6 -12117];
    %a(2,:)=[-164.4 -12071];
    %a(3,:)=[-166.1 -19974];

    a(1,:)=[-181.6 -13226];
    a(2,:)=[-162.85 -11695.5];
    a(3,:)=[-168.75 -20566.5];

elseif Nummer==2
    a(1,:)=[164.7 74755];
    a(2,:)=[162 3973];
    a(3,:)=[164.1 19772];

elseif Nummer==3
    %Nr 3_1
    %a(1,:)=[155.2 -4339];
    %a(2,:)=[165.7 -71175];
    %a(3,:)=[168.2 15650];
    %Nr 3_2
    %a(1,:)=[163.2 -5078];
    %a(2,:)=[163.8 -70254];
    %a(3,:)=[162.3 15455];

    a(1,:)=[159.2 -4708.5];
```

```
177     a(2,:)=[164.75 -70714.5];
178     a(3,:)=[165.25 15552.5];
179
180 elseif Nummer==4
181     %Nr 4
182     a(1,:)=[160.2 61244];
183     a(2,:)=[162.6 33321];
184     a(3,:)=[167.1 -20976];
185
186 elseif Nummer==6
187     %Nr 6_1
188     %a(1,:)=[-162.7 7029];
189     %a(2,:)=[-172.5 2898];
190     %a(3,:)=[-158.7 12408];
191     %Nr 6_2
192     %a(1,:)=[-166.1 7242];
193     %a(2,:)=[-172.5 2898];
194     %a(3,:)=[-165.4 13068];
195
196     a(1,:)=[-164.4 7135.5];
197     a(2,:)=[-172.5 2898];
198     a(3,:)=[-162.05 12738];
199 end
200
201 %plot(x(1,:),y,'+r',x(2,:),y,'+y',x(3,:),y,'+g',x(1,:),f(1,:),'red',x(2,:),f(2,:),'yellow',
202 %x(3,:),f(3,:),'green'), grid on, xlabel('Voltage (mV)'),
203 %ylabel('Force [N]'), hold on legend('DMS1','DMS2','DMS3')
204
205 function [a] = polyfit_exact_and_ls(xe, ye, xls, yls, n)
206 %function to force a polynomial exactly through some points of a data set,
207 %and approximately through others
208 %
209 % input:
210 %  xe, ye points to fit exactly (xe, ye are same size)
211 %  xls, yls  points to least squares fit (xls, yls are same size)
212 %  n  degree of of polynomial (n+1 should be <= total points)
213
214 % output:
215 %      a   n+1 polynomial coefficients ordered from high to low terms in a row array
216 %  WARNINGS - 1. This seems to work okay on small data sets but probably not robust in any way
217 %             2. NO ERROR CHECKING HERE
218 %             3. I doubt there is any theoretical basis for this to work, but the results on
219 %                my tests are good
220
221 ps = size(xe);
222 npe=prod(ps); %number of exact points
223 xe = reshape(xe, [npe,1]); %turn into columns
224 ye = reshape(ye, [npe,1]); %turn into columns
225
226 ps = size(xls);
227 npls=prod(ps);%number of ls points
228 xls = reshape(xls, [npls,1]); %turn into columns
229 yls = reshape(yls, [npls,1]); %turn into columns
230
231 x = [xe;xls];
232 V(length(x), n+1)=0; %preallocate V
233
234 for k = n+1:-1:1
```

```
235       V(:,k)=x.^(k-1); %create Vandermonde matrix
236     end
237
238     % calcs use Schur complement analysis of Vandermonde matrix;
239     %
240     % below based on symbolic solution of exact equations of
241     % "npe" variables in terms of "n+1-npe" other variables;
242     % Something at least similar to a "least squares" analysis
243     %   performed on remaining eqs for "n+1-npe" unknowns
244     %   Normal eqs formed with these remaining eqs, and
245     %      "n+1-npe" are determined;
246     %   remaining npe unknowns are calculated based on other solution;
247     % total unknowns is n+1
248     %
249     % get submatrices for calculations - NOTE A^(-1) must exist
250     A = V(1:npe,1:npe);
251     B = V(1:npe, npe+1:n+1);
252     C = V(npe+1:length(x),1:npe);
253     D = V(npe+1:length(x), npe+1:n+1);
254     % create normals equations on reduced unknown columns, then solve
255     D = D-C*(A\B);
256     b = D'*(yls - C*(A\ye));
257     D=D'*D; ahigh=D\b;
258     % use ls and solve exact equations
259     alow = A\(ye-B*ahigh);
260     % combine solutions, transpose, and reverse for use in polyval
261     a = fliplr([alow; ahigh]')
```

5.3 Explanation of Main Files "DataEvaluationDDS", "DataEvaluationDDSTotals" and "DataEvaluationDDSComparison"

Once the MatLab files for the calibrations have been set up, they are all implemented into 1 main file, where the majority of the data evaluation for a single dataset takes place. This file, called "DataEvaluationDDS", has been constructed in a way, so that it functions for all the 471 available data sets. Afterwards, the file "DataEvaluationDDSTotals" is programmed to combine the results of all single datasets for each sensor and "DataEvaluationDDSComparison" then allows to compare the total results of all 5 sensor to each other. All files are programmed in a manner, that they should also work for any future field tests as long as the measured data are provided in the same way. This subchapter is dedicated to explaining these 3 main files and all external files, that are used in combination with them, in detail.

1. **"DataEvaluationDDS"**

The MatLab file "DataEvaluationDDS", which is shown from page 47 to 51, is used to analyse single datasets and the following results and diagrams are produced by it:

- Acceleration in x-, y- and z-direction
- Battery voltage
- 3 individual strain gauge values
- Strain gauges mean value
- Maximum frequency and period duration
- Single-sided amplitude spectrum plots
- Temperature
- Displacement in x-, y- and z-direction and their averages
- Load versus position diagrams

For this tutorial, the dataset 4DAQ40 will be used and as explained before, this means the 40^{th} hour of recording of the downhole dynamometer sensor number 4 and subsequently the 80^{th} hour after the start of the field test.

The first step is to insert the desired dataset in row 6 of the program, where it says "filename". The file is programmed in a way to read the hour and the position of the file (row 11 and 12), which is important for the calibration later on. Afterwards the measured data of the chosen dataset will be automatically implemented as an array of time and data values

(row 15). From row 22 to 28, a number of important variables, like the amount of datapoints, the time and the time increments between datapoints, are defined by accessing the values of the measured data.

In the next step, the calibration of the acceleration data is performed by accessing the aforementioned “AccelerationCalibration” file in row 32. The calibrated data are then also multiplied by the variable g, which has been defined in row 5 as 9,81, in order to change the unit of the acceleration from “g” to m/s^2. The variables “acx”, “acy” & “acz”, which represent the acceleration in x-, y- and z-direction, are created to save the calibrated acceleration data. The MatLab files for the calibration of the temperature and the strain gauges are also defined in row 39 and 40, but they will only be invoked later on.

Subsequently, a for-loop is used to plot all the given data values. It starts with the calibrated acceleration data from row 51 to 60. The calibration step was already done earlier, so it is only necessary to transfer the data from “acx”, “acy” & “acz” into the loop. From row 63 to 66, the measured values for the battery voltage of the sensor are saved.

From row 69 to 91, the calibration of the strain gauges and a fast Fourier transform are performed. The file for the fast Fourier transform, which is called “FFT”, is an external one and it can be seen on page 52. This additional program is necessary in order to receive values for the maximum frequency and the maximum amplitude, which are needed for the calculation of the period duration.

The results for the frequency (in Hertz) and period duration (in seconds) of the dataset 4DAQ40 are shown in Figure 25. It is important, that all 3 values for the maximum frequency and the period duration are equal, because it proves that all 3 strain gauges (DMS1, DMS2 & DMS3) are functioning properly.

```
fMax = 0.047302
PeriodDuration = 21.140645
fMax = 0.047302
PeriodDuration = 21.140645
fMax = 0.047302
PeriodDuration = 21.140645
```

Figure 25: Results of maximum frequency and period duration for dataset 4DAQ40

Back in the MatLab File “DataEvaluationDDS”, the temperature calibration is performed in row 94 to 98 to change the unit to Celsius.

From row 100 to 106, the for-loop closes by plotting all the calibrated values and a single-sided amplitude spectrum plot for all 3 strain gauges as well. The resulting plots of 4DAQ40 will be shown and explained in the next subchapter.

In row 109 an additional external file, called “DMSMeanValue” is accessed, which calculates the mean value for all 3 strain gauges. This external MatLab file is shown on page 53. It is

important to note here, that the strain gauge number 2 (DMS2) for the downhole dynamometer sensor 6 (DDS6) was not working properly, therefore the mean value calculation for sensor 6 was altered, as seen in row 5 of the file.

After the calibration and plotting of the acceleration data, the MatLab sub-program Simulink is used to perform 2 integration steps, which result in the displacement data “sx”, “sy” & “sz”. The Simulink program, which is shown in Figure 26, first calculates a mean value for each pump cycle and then uses a lowpass filter to eliminate any noise and data outliers. This step was performed for the calibrated acceleration data in x-, y- and z-direction.

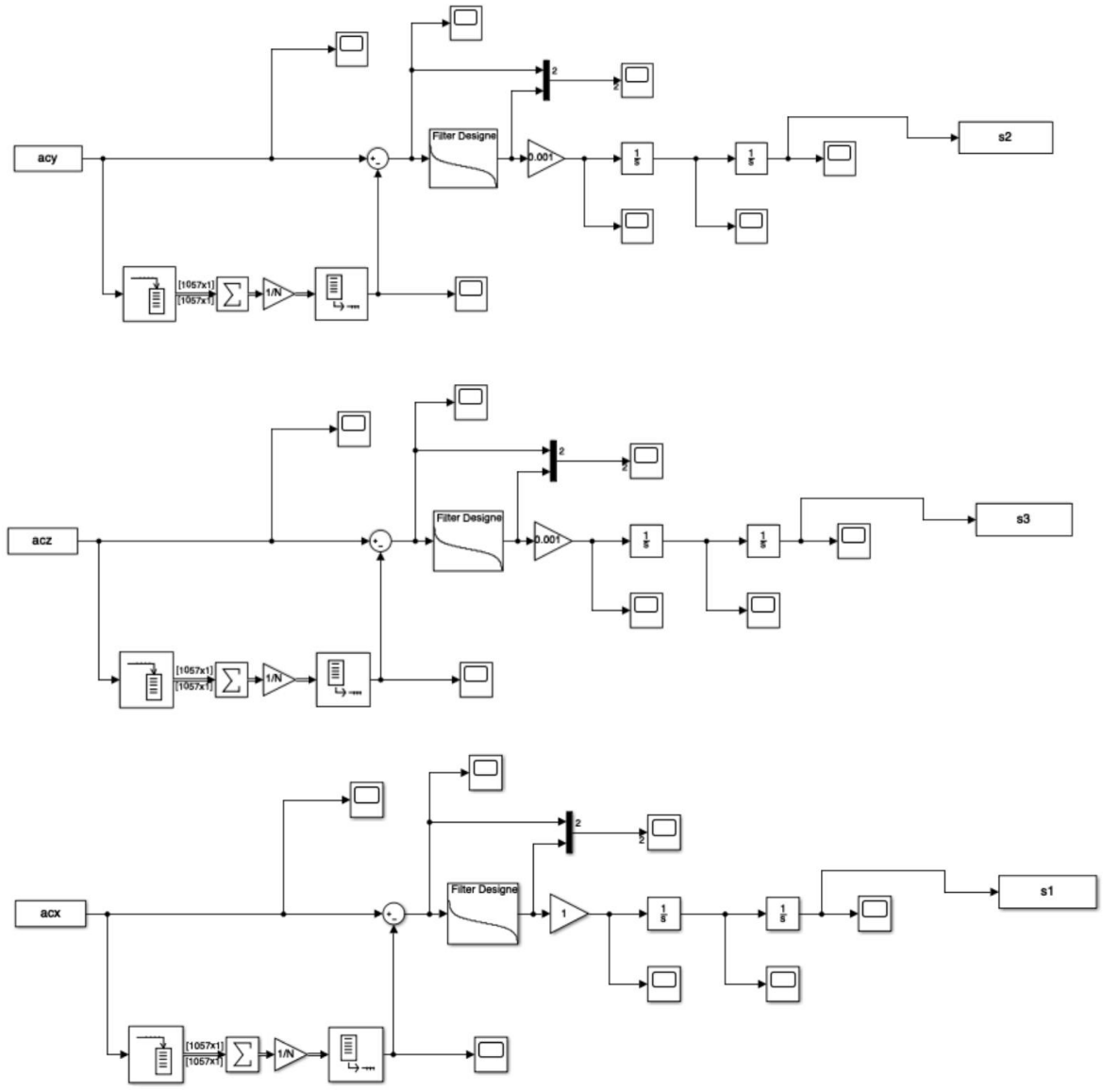


Figure 26: Simulink filter program “Signal_Dem_sep_filt”

The main file automatically accesses the Simulink file in row 123, where the integration steps are performed, and then plots the results of the displacement in all three directions. The displacement values are also averaged per period duration, which is programmed by putting the results of each period in a matrix and then using the function “mean” to calculate a mean cycle.

The last step of the file “DataEvaluationDDS” then uses the information from the integration to generate load versus position plots. Since the system needs some time to stabilize, the values for the first plot of the diagram are taken from the 10th period duration until period duration 120. It is possible to choose either the average displacement values in x-direction or the actual ones . Plot 2 is programmed to only show the load versus position diagram for one single period duration.

“DataEvaluationDDS” MatLab file:

```
clear;
clc;
close all;

g = 9.81; % given unit of acceleration (g=9,81m/s^2)
fileName = '4DAQ40.mat'; % insert different datasets at this point
PeriodDurations = []; % Value of Period Duration in s

[mat,tok,ext] = regexp(fileName, 'Q(.*)\.mat', 'match', 'tokens', 'tokenExtents');
hour = cell2mat(tok{1,1});
hour = str2num(hour); % hour is read from fileName
nr = str2num(fileName(1)); % DDS position is read from fileName (important for calibration)

load(strcat('data/',fileName));
data = cell(8);  % array of [time-values; data-values]
labels = {'Messdaten/Acceleration-X','Messdaten/Acceleration-Y',...
    'Messdaten/Acceleration-Z','Messdaten/Battery Voltage',...
    'Messdaten/Analog 1','Messdaten/Analog 2',...
    'Messdaten/Analog 3','Messdaten/Analog 4'};

% Definition of variables
for i = 3 : 10
    l = length(ConvertedData.Data.MeasuredData(i).Data);
    dt = ConvertedData.Data.MeasuredData(i).Property(3).Value;
    t = linspace(0,l*dt,l);
    da = ConvertedData.Data.MeasuredData(i).Data;
    data{i-2} = [t' da];
end

% Calibration of Acceleration Data
uncalibrated_acc_data = [data{1}(:,2) data{2}(:,2) data{3}(:,2)];
calibrated_acc_data = AccelerationCalibration(nr,uncalibrated_acc_data)*g; % multiplied by 9,81 -->
unit = m^2/s
acx = [data{1}(:,1) calibrated_acc_data(1,:)'];
acy = [data{2}(:,1) calibrated_acc_data(2,:)'];
acz = [data{3}(:,1) calibrated_acc_data(3,:)'];

%definitions needed to use external files for  DMS & Temperature calibration
dms_coeff = DMSCalibration(nr);
```

```
T_coeff = TemperatureCalibration(nr);
% Plot of all given data values
for i = 1:length(data)
    di = ConvertedData.Data.MeasuredData(i+2).Property(6).Value;
    unitX = ConvertedData.Data.MeasuredData(i+2).Property(8).Value;
    unitY = ConvertedData.Data.MeasuredData(i+2).Property(10).Value;
    d = data{i};
    x = d(:,1);
    y = d(:,2);

     %Transfer of calibrated acceleration data already multiplied by 9,81
    if i == 1 || i == 2 || i == 3
        unitY = 'a[m/s^2]';
        if i==1
            y = acx(:,2);
        elseif i==2
            y = acy(:,2);
        elseif i==3
            y = acz(:,2);
        end
    end

       % Batery Voltage
    if i == 4
         BattVoltage=[x y];
         unitY = 'Battery Voltage in mV';
    end

     % use of file DMSCalibration to convert data values to Newton (Load)
    if i == 5 || i == 6 || i == 7

        % Plot of single-sided Amplitude Spectrum after FFT
        [f,amp] = FFT(y);
        figure
        plot(f,amp,'.-');
        xlim([0 0.5]);
        title('Single-Sided Amplitude Spectrum of y(t)')
        xlabel('Frequency (Hz)')
        ylabel('|Y(f)|')

        % Idendification of Frequency and Calculation of Period Duration
        [m,index] = max(amp);
        fMax = f(index);
        fprintf('fMax = %.6f \n', fMax);
        fprintf('PeriodDuration = %.6f \n', 1/fMax);
        PeriodDurations = [PeriodDurations (1/fMax)];

        currentDMSCoeff = dms_coeff(i-4,:);
        y = polyval(currentDMSCoeff, y);
        DMSValues(i-4,:) = y;
        unitY = 'Load in N';
    end

     % use of file TemperatureCalibration to plot Temperature in Celsius
    if i == 8
        y = polyval(T_coeff,y);
        unitY = 'Temperature in °C';
        Temperature=[x y];
    end
```

```
99
100     figure;
101     plot(x,y);
102     title(labels{i});
103     xlabel(strcat(di,'[',unitX,']'));
104     ylabel(strcat(labels{i},'[',unitY,']'));
105     axis tight;
106 end
107
108 % use of file DMSMeanValue to plot the mean value of DMS1, DMS2 and DMS3
109 DMS_mean = DMSMeanValue(DMSValues,nr);
110 t=0:0.02:0.02*size(DMSValues,2)-0.02;
111 figure;
112 plot(t,DMS_mean,'b');
113 title('Mean DMS');
114 xlabel('t [s]');
115 ylabel('Load in N');
116 DMSmeanValue=[t' DMS_mean'];
117
118 Fs=1/dt;
119 N = round(1/fMax*Fs,0);  % length of one cycle
120
121 % x direction
122 acx;
123 sim('Signal_Dem_sep_filt');
124 sx=s1';
125 sx=sx([1:length(sx)-1]);
126 sx = sx - min(sx);
127 x = [0:dt:length(sx)*dt-dt];
128 figure;
129 plot(x,sx);
130 title('displacement-X');
131 xlabel('t [s]');
132 ylabel('displacement-X [m]');
133 sx=[x' sx'];
134
135 % x average
136 start = ((0)*PeriodDurations(1))/dt;
137 AmountPeriods = floor((3600 - start) / PeriodDurations(1));
138
139 for i = 1:1:AmountPeriods
140     startIndex = ((i-1)*PeriodDurations(1))/dt+1;
141     endIndex = (i*PeriodDurations(1))/dt;
142     averagedisplacement =  sx(startIndex:1:endIndex,2);
143     layers(:,i) = averagedisplacement;
144 end
145
146 AvgDispl = mean(layers,2);
147 time = linspace(0,PeriodDurations(1),size(layers,1));
148 averagePeriod = (AvgDispl-min(AvgDispl));
149
150 PeriodSequence = repmat(averagePeriod',1,AmountPeriods)';
151 time = linspace(0,AmountPeriods*PeriodDurations(1),size(PeriodSequence,1));
152 figure();
153 plot(time, PeriodSequence);
154 title('Average displacement in x')
155 xlabel('t [s]');
156 ylabel('avg displacement-x [m]');
157 AverageDisplacementX=[time' PeriodSequence];
```

```
158
159 % y direction
160 acy;
161 %sim('Signal_Dem_sep_filt');
162 sy=s2';
163 sy=sy([1:length(sy)-1]);
164 sy = sy - min(sy);
165 figure;
166 plot(x,sy);
167 title('displcement-Y');
168 xlabel('t [s]');
169 ylabel('displacement-Y [m]');
170 sy=[x' sy'];
171
172 % y average
173 start = ((0)*PeriodDurations(1))/dt;
174 AmountPeriods = floor((3600 - start) / PeriodDurations(1));
175
176 for i = 1:1:AmountPeriods
177     startIndex = ((i-1)*PeriodDurations(1))/dt+1;
178     endIndex = (i*PeriodDurations(1))/dt;
179     averagedisplacement =  sy(startIndex:1:endIndex,2);
180     layers(:,i) = averagedisplacement;
181 end
182
183 AvgDispl = mean(layers,2);
184 %figure();
185 time = linspace(0,PeriodDurations(1),size(layers,1));
186 %plot(time, AvgDispl-min(AvgDispl));
187 %title('Average displacement in x');
188 %xlabel('t');
189 %ylabel('m');
190 averagePeriod = (AvgDispl-min(AvgDispl));
191
192 PeriodSequence = repmat(averagePeriod',1,AmountPeriods)';
193 time = linspace(0,AmountPeriods*PeriodDurations(1),size(PeriodSequence,1));
194 figure();
195 plot(time, PeriodSequence);
196 title('Average displacement in y')
197 xlabel('t [s]');
198 ylabel('avg displacement-y [m]');
199
200 % z direction
201 acz;
202 %sim('Signal_Dem_sep_filt');
203 sz=s3';
204 sz=sz([1:length(sz)-1]);
205 sz = sz - min(sz);
206 figure;
207 plot(x,sz);
208 title('displacement-Z');
209 xlabel('t [s]');
210 ylabel('displacement-Z [m]');
211 sz=[x' sz'];
212
213 % z average
214 start = ((0)*PeriodDurations(1))/dt;
215 AmountPeriods = floor((3600 - start) / PeriodDurations(1));
216
```

```
for i = 1:1:AmountPeriods
    startIndex = ((i-1)*PeriodDurations(1))/dt+1;
    endIndex = (i*PeriodDurations(1))/dt;
    averagedisplacement =  sz(startIndex:1:endIndex,2);
    layers(:,i) = averagedisplacement;
end

AvgDispl = mean(layers,2);
%figure();
time = linspace(0,PeriodDurations(1),size(layers,1));
%plot(time, AvgDispl-min(AvgDispl));
%title('Average displacement in x');
%xlabel('t');
%ylabel('m');
averagePeriod = (AvgDispl-min(AvgDispl));

PeriodSequence = repmat(averagePeriod',1,AmountPeriods)';
time = linspace(0,AmountPeriods*PeriodDurations(1),size(PeriodSequence,1));
figure();
plot(time, PeriodSequence);
title('Average displacement in z')
xlabel('t [s]');
ylabel('avg displacement-z [m]');

% Load vs. Position Diagrams
position = sx(:,2)'; % values from displacement in x-direction are taken
averageposition = AverageDisplacementX(:,2)'; % values from average displacement in x-direction

% Load vs. Position Diagram 1(values from 10 Period Durations to 100)
%position1 = averageposition(100000:1:150000);
%load1 = DMS_mean(100000:1:150000);
startIndex = ((10)*PeriodDurations(1))/dt;
endIndex = (120*PeriodDurations(1))/dt;
position1 = averageposition(startIndex:1:endIndex);
load1 = DMS_mean(startIndex:1:endIndex);

LoadVSPosition1 = [position1' load1'];
figure();
plot(position1, load1);
title('Load vs Position 1');
xlabel('Position [m]');
ylabel('Load [N]');

% Load vs. Position Diagram 2(1 Period Duration)
startIndex = ((10)*PeriodDurations(1))/dt;
endIndex = (11*PeriodDurations(1))/dt;
position2 = position(startIndex:1:endIndex);
%position2 = position(7000:1:(140+1*PeriodDurations(1,1))/dt);
position2 = position2 - min(position2);
%load2 = DMS_mean(10000:1:(200+1*PeriodDurations(1,1))/dt);
load2 = DMS_mean(startIndex:1:endIndex);
LoadVSPosition2 = [position2' load2'];
figure();
plot(position2, load2);
title('Load vs Position 2');
xlabel('Position [m]');
ylabel('Load [N]');
```

“FFT” MatLab file:

```
1   function [f,amp]=FFT(dms)
2   Fs = 50;                % Sampling frequency
3   T = 1/Fs;               % Sample time
4   L = length(dms);            % Length of signal
5   t = (0:L-1)*T;           % Time vector
6
7
8   NFFT = 2^nextpow2(L);
9   Y = fft(dms,NFFT)/L;
10  f = Fs/2*linspace(0,1,NFFT/2+1);
11
12  amp=2*abs(Y(1:NFFT/2+1));
13
14  i=1;
15  while f(i)<0.02
16      i=i+1;
17  end
18  amp(1:i)=[];
19  f(1:i)=[];
20  [C,I]=max(amp);
21  delta=1/(NFFT/2+1)*Fs/2*(I+i);
22  T=1/delta;
23  f=f';
24  end
```

DMSMeanValue MatLab file:

```
1  function  [k] = DMSMeanValue(f, Nummer)
2  k=mean(f);
3
4  if Nummer==6
5     k=(1.1*f(1,:)+1.9*f(3,:))/3; % mean value calculation for
6     % DDS postion 6 (DMS1 is valued times 1.1 and DMS 3 times 1.9)
7  end
```

2. “DataEvaluationDDSTotals”

The MatLab file “DataEvaluationDDSTotals”, which is shown from page 55 to 58, is used to analyse the totals datasets for each sensor over the entire recording time. The results from the individual datasets from each sensor are added together in chronological order and the following results and diagrams emerge:

- Acceleration in x-, y- and z-direction over entire recording period for each sensor
- Battery voltage over entire recording period for each sensor
- Temperature over entire recording period for each sensor
- Strain gauges mean value over entire recording period for each sensor
- Displacement in x-, y- and z-direction over entire recording period for each sensor

In row 2, the desired sensor data (1DAQdata, 2DAQdata, 3DAQdata, 4DAQdata, 6DAQdata) is implemented and in row 8, the starting point of the sensor data is chosen (variable z). This makes it possible to also skip certain datasets, for example starting the evaluation of the sensor totals at the 12th recording hour, when the pump was activated. From row 13 to 27, the individual datasets of the sensor are arranged in chronological order according to the hour of recording.

The variables for the collection of the total data are defined in the next step. This is done for the temperature, acceleration in x-, y- and z-direction, battery voltage, DMS mean value and also the displacement in x-, y- and z-direction.

Afterwards, a for-loop is created to merge all the individual datasets of each sensor together into the aforementioned total variables. In row 43, “DataEvaluationDDSTotals” accesses the MatLab file “ProcessOneFile”. This is basically a trimmed-down version of “DataEvaluationDDS”, that helps to generate a faster and more efficient calculation of the total values. “ProcessOneFile” can be seen on pages 59 and 60. In row 45, the programming for the total values of the temperature starts. It is important to note, that certain values are “bridged” in order to get more realistic results. This had to be implemented, because the values for each individual dataset show a certain start up period, which ultimately leads to a very convoluted plot for the total values. In this case, the last 10 minutes of the temperature of each dataset are programmed to also be the first 10 minutes of temperature values for the next dataset. If it is preferred to plot the results without bridging, the percentage signs in row 45 and 46 can be removed and percentage signs should be added to row 49 to 57.

Back in the “DataEvaluationDDSTotals” file, the total results for the acceleration in x-, y- and z-direction and the battery voltage are collected. For both the acceleration and the battery voltage, only the first minute of each dataset is bridged, since the individual datasets show a much shorter start-up period compared to the temperature values. As for the temperature

though, the acceleration and battery totals may also be plotted without bridging by removing the percentage signs and adding them to the rows where the bridging is programmed. The DMS mean value totals are plotted without bridging, since the start-up period for these values can be neglected.

Last but not least, the total displacement values in x-, y- and z-direction are assembled and again, the first minute of each dataset is bridged, but it can also be plotted without bridging if desired.

The last part of "DataEvaluationDDSTotals" directs the plotting of all total results for the chosen sensor (DDS1, DDS2, DDS3, DDS4 or DDS6). Compared to the plots of "DataEvaluationDDS", the unit of time was changed from seconds to hours. Examples and an explanation of the total plots will be provided in the next subchapter.

"DataEvaluationDDSTotals" MatLab file:

```
1   if exist('ordnername','var') == 0
2       ordnername = '6DAQData';
3   end
4
5   a = dir(ordnername);
6
7   j = 1;
8   z = 1 % starting dataset
9
10  clear fileNames;
11  clear hours;
12  % Arranging of correct order of datasets
13  for i = 3:1:size(a)
14      fileName = a(i).name;
15      [mat,tok,ext] = regexp(fileName, 'Q(.*)\.mat', 'match', 'tokens', 'tokenExtents');
16      hour = cell2mat(tok{1,1});
17      hour = str2num(hour); % hour is read from fileName
18      hours(j,2) = hour;
19      hours(j,1) = j;
20      fileNames{j} = fileName;
21      j = j + 1;
22  end
23  hours = sortrows(hours,2);
24  clear sortedFileNames;
25  for i = 1:1:size(hours,1)
26      sortedFileNames{i} = fileNames{hours(i,1)};
27  end
28
29  % Definition of variables for collection of total data
30  TemperatureTotal=[];
31  AcxTotal=[];
32  AcyTotal=[];
33  AczTotal=[];
34  BattVoltageTotal=[];
35  DMSmeanValueTotal=[];
36  SxTotal=[];
37  SyTotal=[];
38  SzTotal=[];
```

```
39
40  for j = z:1:length(sortedFileNames)
41      currentFileName = sortedFileNames{j}
42      name = currentFileName;
43      ProcessOneFile;
44
45  % Temperature Totals without bridging of start up period
46  %TemperatureTotal = [TemperatureTotal;Temperature(:,2)];
47
48  % Temperature Totals with bridging of first 10 minutes for each dataset except the first one
49      if j == z
50          TemperatureTotal = [TemperatureTotal;Temperature(:,2)];
51      else
52          TemperatureTotal = [...
53          TemperatureTotal;...
54          TemperatureTotal([length(TemperatureTotal)-30000+1:length(TemperatureTotal)]);...
55          Temperature([30000:1:length(Temperature)],2)...
56          ];
57      end
58
59  % Acceleration Totals with bridging of first minute for each dataset except the first one
60      if j == z
61          AcxTotal = [AcxTotal;acx(:,2)];
62      else
63          AcxTotal = [...
64          AcxTotal;...
65          AcxTotal([length(AcxTotal)-3000+1:length(AcxTotal)]);...
66          acx([3000:1:length(acx)],2)...
67          ];
68      end
69
70      if j == z
71          AcyTotal = [AcyTotal;acy(:,2)];
72      else
73          AcyTotal = [...
74          AcyTotal;...
75          AcyTotal([length(AcyTotal)-3000+1:length(AcyTotal)]);...
76          acy([3000:1:length(acy)],2)...
77          ];
78      end
79
80      if j == z
81          AczTotal = [AczTotal;acz(:,2)];
82      else
83          AczTotal = [...
84          AczTotal;...
85          AczTotal([length(AczTotal)-3000+1:length(AczTotal)]);...
86          acz([3000:1:length(acz)],2)...
87          ];
88      end
89
90      % Acceleration Totals without bridging
91      %AcxTotal = [AcxTotal;acx(:,2)];
92      %AcyTotal = [AcyTotal;acy(:,2)];
93      %AczTotal = [AczTotal;acz(:,2)];
94
95      % Battery Voltage Totals without bridging
96      %BattVoltageTotal = [BattVoltageTotal;BattVoltage(:,2)];
97
```

```
98        % Battery Voltage Totals with bridging of the first minute
99       if j == z
100         BattVoltageTotal = [BattVoltageTotal;BattVoltage(:,2)];
101      else
102         BattVoltageTotal = [...
103         BattVoltageTotal;...
104         BattVoltageTotal([length(BattVoltageTotal)-3000+1:length(BattVoltageTotal)]);...
105         BattVoltage([3000:1:length(BattVoltage)],2)...
106         ];
107      end
108
109      % DMS Mean Values Totals without bridging
110      DMSmeanValueTotal = [DMSmeanValueTotal;DMSmeanValue(:,2)];
111
112       % DMS Mean Values Totals with bridging of the first minute
113      %if j == z
114        %  DMSmeanValueTotal = [DMSmeanValueTotal;DMSmeanValue(:,2)];
115      % else
116        % DMSmeanValueTotal = [...
117        % DMSmeanValueTotal;...
118        % DMSmeanValueTotal([length(DMSmeanValueTotal)-3000+1:length(DMSmeanValueTotal)]);...
119        % DMSmeanValue([3000:1:length(DMSmeanValue)],2)...
120        % ];
121      % end
122
123      % Displacement Totals without bridging
124      %SxTotal = [SxTotal;sx(:,2)];
125      %SyTotal = [SyTotal;sy(:,2)];
126      %SzTotal = [SzTotal;sz(:,2)];
127
128   % Displacement Totals with bridging of first minute for each dataset except the first one
129      if j == z
130         SxTotal = [SxTotal;sx(:,2)];
131      else
132         SxTotal = [...
133         SxTotal;...
134         SxTotal([length(SxTotal)-3000+1:length(SxTotal)]);...
135         sx([3000:1:length(sx)],2)...
136         ];
137      end
138
139      if j == z
140         SyTotal = [SyTotal;sy(:,2)];
141      else
142         SyTotal = [...
143         SyTotal;...
144         SyTotal([length(SyTotal)-3000+1:length(SyTotal)]);...
145         sy([3000:1:length(sy)],2)...
146         ];
147      end
148
149      if j == z
150         SzTotal = [SzTotal;sz(:,2)];
151      else
152         SzTotal = [...
153         SzTotal;...
154         SzTotal([length(SzTotal)-3000+1:length(SzTotal)]);...
155         sz([3000:1:length(sz)],2)...
156         ];
```

```
157         end
158
159     end
160
161     % Plotting of Results
162     figure(1);
163     plot([1:1:length(TemperatureTotal)]*0.02 / 3600,TemperatureTotal);
164     title('Temperature Totals');
165     xlabel('t [h]');
166     ylabel('Temperature [°C]');
167     figure(2);
168     plot([1:1:length(AcxTotal)]*0.02 / 3600,AcxTotal);
169     title('Acceleration X Totals');
170     xlabel('t [h]');
171     ylabel('Acceleration-X [m/s^2]');
172     figure(3);
173     plot([1:1:length(AcyTotal)]*0.02 / 3600,AcyTotal);
174     title('Acceleration Y Totals');
175     xlabel('t [h]');
176     ylabel('Acceleration-Y [m/s^2]');
177     figure(4);
178     plot([1:1:length(AczTotal)]*0.02 / 3600,AczTotal);
179     title('Acceleration Z Totals');
180     xlabel('t [h]');
181     ylabel('Acceleration-Z [m/s^2]');
182     figure(5);
183     plot([1:1:length(BattVoltageTotal)]*0.02 / 3600,BattVoltageTotal);
184     title('Battery Voltage Totals');
185     xlabel('t [h]');
186     ylabel('Battery Voltage [mV]');
187     figure(6);
188     plot([1:1:length(DMSmeanValueTotal)]*0.02 / 3600,DMSmeanValueTotal);
189     title('DMS Mean Value Totals');
190     xlabel('t [h]');
191     ylabel('Load [N]');
192     figure(7);
193     plot([1:1:length(SxTotal)]*0.02 / 3600,SxTotal);
194     title('displacement-X Totals');
195     xlabel('t [h]');
196     ylabel('displacement-X [m]');
197     figure(8);
198     plot([1:1:length(SyTotal)]*0.02 / 3600,SyTotal);
199     title('displacement-Y Totals');
200     xlabel('t [h]');
201     ylabel('displacement-Y [m]');
202     figure(9);
203     plot([1:1:length(SzTotal)]*0.02 / 3600,SzTotal);
204     title('displacement-Z Totals');
205     xlabel('t [h]');
206     ylabel('displacement-Z [m]');
```

“ProcessOneFile” MatLab file:

```
clear DMSValues;
    g = 9.81; % given unit of acceleration (g=9,81m/s^2)
    PeriodDurations = []; % Value of Period Duration in s


    [mat,tok,ext] = regexp(name, 'Q(.*)\.mat', 'match', 'tokens', 'tokenExtents');
    hour = cell2mat(tok{1,1});
    hour = str2num(hour); % hour is read from name

    nr = str2num(name(1)); % DDS position is read from name (important for calibration)

    load(strcat(ordnername,'/',name));
    data = cell(8);  % array of [time-values; data-values]

    % Definition of variables
    for i = 3 : 10
        l = length(ConvertedData.Data.MeasuredData(i).Data);
        dt = ConvertedData.Data.MeasuredData(i).Property(3).Value;
        t = linspace(0,l*dt,l);
        da = ConvertedData.Data.MeasuredData(i).Data;
        data{i-2} = [t' da];
    end

    % Calibration of Acceleration Data
    uncalibrated_acc_data = [data{1}(:,2) data{2}(:,2) data{3}(:,2)];
    calibrated_acc_data = AccelerationCalibration(nr,uncalibrated_acc_data)*g; % multiplied by 9,81 --
> unit = m^2/s
    acx = [data{1}(:,1) calibrated_acc_data(1,:)'];
    acy = [data{2}(:,1) calibrated_acc_data(2,:)'];
    acz = [data{3}(:,1) calibrated_acc_data(3,:)'];

    %definitions needed to use external files for  DMS & Temperature calibration
    dms_coeff = DMSCalibration(nr);
    T_coeff = TemperatureCalibration(nr);


    % Plot of all given data values
    for i = 1:length(data)
        d = data{i};
        x = d(:,1);
        y = d(:,2);

         %Transfer of calibrated acceleration data already multiplied by 9,81
        if i == 1 || i == 2 || i == 3
            if i==1
                y = acx(:,2);
            elseif i==2
                y = acy(:,2);
            elseif i==3
                y = acz(:,2);
            end
        end

          % Batery Voltage
        if i == 4
            BattVoltage=[x y];
```

```
        end

         % use of file DMSCalibration to convert data values to Newton (Load)
        if i == 5 || i == 6 || i == 7
            % Plot of single-sided Amplitude Spectrum after FFT
            [f,amp] = FFT(y);
            % Idendification of Frequency and Calculation of Period Duration
            [m,index] = max(amp);
            fMax = f(index);
            fprintf('fMax = %.6f \n', fMax);
            fprintf('PeriodDuration = %.6f \n', 1/fMax);
            PeriodDurations = [PeriodDurations (1/fMax)];
            currentDMSCoeff = dms_coeff(i-4,:);
            y = polyval(currentDMSCoeff, y);
            DMSValues(i-4,:) = y;
        end

         % use of file TemperatureCalibration to plot Temperature in Celsius
        if i == 8
            y = polyval(T_coeff,y);
            Temperature=[x y];
        end
    end


    % use of file DMSMeanValue to plot the mean value of DMS1, DMS2 and DMS3
    DMS_mean = DMSMeanValue(DMSValues,nr);
    t=0:0.02:0.02*size(DMSValues,2)-0.02;
    DMSmeanValue=[t' DMS_mean'];

    Fs=1/dt;
    N = round(1/fMax*Fs,0);  % length of one cycle

    % Use of Simulink Filter Program for 2 Integration steps --> distance s[m]

    % x direction
    acx;
    sim('Signal_Dem_sep_filt');
    sx=s1';
    sx=sx([1:length(sx)-1]);
    sx = sx - min(sx);
    x = [0:dt:length(sx)*dt-dt];
    sx=[x' sx'];

    % y direction
    acy;
    %sim('Signal_Dem_sep_filt');
    sy=s2';
    sy=sy([1:length(sy)-1]);
    sy = sy - min(sy);
    sy=[x' sy'];

    % z direction
    acz;
    %sim('Signal_Dem_sep_filt');
    sz=s3';
    sz=sz([1:length(sz)-1]);
    sz = sz - min(sz);
    sz=[x' sz'];
```

3. **“DataEvaluationDDSComparison”**

The MatLab file “DataEvaluationDDSComparison”, which is shown from page 61 to 65 enables to compare the total results of each downhole dynamometer sensor by plotting the outcomes for DDS1, DDS2, DDS3, DDS4 and DDS6 in one plot. The following diagrams result:

- Comparison of all acceleration plots in x-, y- and z-direction over entire recording period
- Comparison of all battery voltage plots over entire recording period
- Comparison of all temperature plots over entire recording period
- Comparison of all strain gauges mean value plots over entire recording period
- Comparison of all displacement plots in x-, y- and z-direction over entire recording period

From row 4 to 23, the datasets desired for comparison are defined. Exactly like for the total values, this is done for the temperature, acceleration in x-, y- and z-direction, battery voltage, DMS mean value and the displacement in x-, y- and z-direction.

Staring in row 25, another for-loop is programmed that collects the totals values from the previous MatLab file “DataEvaluationDDSTotals” for each of the 5 sensors.

The last step in “DataEvaluationDDSComparison” is dedicated to plotting the total results for each quantity (temperature, acceleration, DMS mean value, battery voltage, displacement) off all 5 sensor in one plot to generate a better overview for analysing purposes.

The function “legend”, for example in row 93, assigns different colours to the total values of the individual sensor in order to differentiate between them.

“DataEvaluationDDSComparison” MatLab file:

```
1   clear;clc;
2
3   % Definition of datasets
4   directories = {'1DAQdata','2DAQdata','3DAQdata','4DAQdata','6DAQdata'};
5   tempComparison = [];
6   sxComparison = [];
7   syComparison = [];
8   szComparison = [];
9   DMSmeanValueComparison = [];
10  BattVoltageComparison = [];
11  AcxComparison = [];
12  AcyComparison = [];
13  AczComparison = [];
14
15  daqTemp1=[];daqTemp2=[];daqTemp3=[];daqTemp4=[];daqTemp6=[];
```

```
daqsx1=[];daqsx2=[];daqsx3=[];daqsx4=[];daqsx6=[];
daqsy1=[];daqsy2=[];daqsy3=[];daqsy4=[];daqsy6=[];
daqsz1=[];daqsz2=[];daqsz3=[];daqsz4=[];daqsz6=[];
daqDMS1=[];daqDMS2=[];daqDMS3=[];daqDMS4=[];daqDMS6=[];
daqBatt1=[];daqBatt2=[];daqBatt3=[];daqBat4=[];daqBatt6=[];
daqacx1=[];daqacx2=[];daqacx3=[];daqacx4=[];daqacx6=[];
daqacy1=[];daqacy2=[];daqacy3=[];daqacy4=[];daqacy6=[];
daqacz1=[];daqacz2=[];daqacz3=[];daqacz4=[];daqacz6=[];

% for loop to compare total values
for k = 1:length(directories)
    ordnername = directories{k}
    DataEvaluationDDSTotals;
    if k==1
        daqTemp1 = TemperatureTotal;
        daqsx1 = SxTotal;
        daqsy1 = SyTotal;
        daqsz1 = SzTotal;
        daqDMS1 = DMSmeanValueTotal;
        daqBatt1 = BattVoltageTotal;
        daqacx1 = AcxTotal;
        daqacy1 = AcyTotal;
        daqacz1 = AczTotal;
    elseif k==2
        daqTemp2 = TemperatureTotal;
        daqsx2 = SxTotal;
        daqsy2 = SyTotal;
        daqsz2 = SzTotal;
        daqDMS2 = DMSmeanValueTotal;
        daqBatt2 = BattVoltageTotal;
        daqacx2 = AcxTotal;
        daqacy2 = AcyTotal;
        daqacz2 = AczTotal;
    elseif k==3
        daqTemp3 = TemperatureTotal;
        daqsx3 = SxTotal;
        daqsy3 = SyTotal;
        daqsz3 = SzTotal;
        daqDMS3 = DMSmeanValueTotal;
        daqBatt3 = BattVoltageTotal;
        daqacx3 = AcxTotal;
        daqacy3 = AcyTotal;
        daqacz3 = AczTotal;
    elseif k==4
        daqTemp4 = TemperatureTotal;
        daqsx4 = SxTotal;
        daqsy4 = SyTotal;
        daqsz4 = SzTotal;
        daqDMS4 = DMSmeanValueTotal;
        daqBatt4 = BattVoltageTotal;
        daqacx4 = AcxTotal;
        daqacy4 = AcyTotal;
        daqacz4 = AczTotal;
    elseif k==5
        daqTemp6 = TemperatureTotal;
        daqsx6 = SxTotal;
        daqsy6 = SyTotal;
        daqsz6 = SzTotal;
        daqDMS6 = DMSmeanValueTotal;
        daqBatt6 = BattVoltageTotal;
```

```
        daqacx6 = AcxTotal;
        daqacy6 = AcyTotal;
        daqacz6 = AczTotal;
    end
end

% comparison plots
figure;
plot([1:1:length(daqTemp1)]*0.02 / 3600,daqTemp1);
hold on;
plot([1:1:length(daqTemp2)]*0.02 / 3600,daqTemp2);
hold on;
plot([1:1:length(daqTemp3)]*0.02 / 3600,daqTemp3);
hold on;
plot([1:1:length(daqTemp4)]*0.02 / 3600,daqTemp4);
hold on;
plot([1:1:length(daqTemp6)]*0.02 / 3600,daqTemp6);
legend('1DAQ','2DAQ','3DAQ','4DAQ','6DAQ');
title('Temperature Totals Comparison');
xlabel('t [h]');
ylabel('Temperature [°C]');

figure;
plot([1:1:length(daqsx1)]*0.02 / 3600,daqsx1);
hold on;
plot([1:1:length(daqsx2)]*0.02 / 3600,daqsx2);
hold on;
plot([1:1:length(daqsx3)]*0.02 / 3600,daqsx3);
hold on;
plot([1:1:length(daqsx4)]*0.02 / 3600,daqsx4);
hold on;
plot([1:1:length(daqsx6)]*0.02 / 3600,daqsx6);
legend('1DAQ','2DAQ','3DAQ','4DAQ','6DAQ');
title('displacement-X Totals Comparison');
xlabel('t [h]');
ylabel('displacement-X [m]');

figure;
plot([1:1:length(daqsy1)]*0.02 / 3600,daqsy1);
hold on;
plot([1:1:length(daqsy2)]*0.02 / 3600,daqsy2);
hold on;
plot([1:1:length(daqsy3)]*0.02 / 3600,daqsy3);
hold on;
plot([1:1:length(daqsy4)]*0.02 / 3600,daqsy4);
hold on;
plot([1:1:length(daqsy6)]*0.02 / 3600,daqsy6);
legend('1DAQ','2DAQ','3DAQ','4DAQ','6DAQ');
title('displacement-Y Totals Comparison');
xlabel('t [h]');
ylabel('displacement-Y [m]');

figure;
plot([1:1:length(daqsz1)]*0.02 / 3600,daqsz1);
hold on;
plot([1:1:length(daqsz2)]*0.02 / 3600,daqsz2);
hold on;
plot([1:1:length(daqsz3)]*0.02 / 3600,daqsz3);
hold on;
plot([1:1:length(daqsz4)]*0.02 / 3600,daqsz4);
```

```
136  hold on;
137  plot([1:1:length(daqsz6)]*0.02 / 3600,daqsz6);
138  legend('1DAQ','2DAQ','3DAQ','4DAQ','6DAQ');
139  title('displacement-Z Totals Comparison');
140  xlabel('t [h]');
141  ylabel('displacement-Z [m]');
142
143  figure;
144  plot([1:1:length(daqDMS1)]*0.02 / 3600,daqDMS1);
145  hold on;
146  plot([1:1:length(daqDMS2)]*0.02 / 3600,daqDMS2);
147  hold on;
148  plot([1:1:length(daqDMS3)]*0.02 / 3600,daqDMS3);
149  hold on;
150  plot([1:1:length(daqDMS4)]*0.02 / 3600,daqDMS4);
151  hold on;
152  plot([1:1:length(daqDMS6)]*0.02 / 3600,daqDMS6);
153  legend('1DAQ','2DAQ','3DAQ','4DAQ','6DAQ');
154  title('DMS Mean Value Totals Comparison');
155  xlabel('t [h]');
156  ylabel('Load [N]');
157
158  figure;
159  plot([1:1:length(daqBatt1)]*0.02 / 3600,daqBatt1);
160  hold on;
161  plot([1:1:length(daqBatt2)]*0.02 / 3600,daqBatt2);
162  hold on;
163  plot([1:1:length(daqBatt3)]*0.02 / 3600,daqBatt3);
164  hold on;
165  plot([1:1:length(daqBatt4)]*0.02 / 3600,daqBatt4);
166  hold on;
167  plot([1:1:length(daqBatt6)]*0.02 / 3600,daqBatt6);
168  legend('1DAQ','2DAQ','3DAQ','4DAQ','6DAQ');
169  title('Battery Voltage Totals Comparison');
170  xlabel('t [h]');
171  ylabel('Voltage [mV]');
172
173  figure;
174  plot([1:1:length(daqacx1)]*0.02 / 3600,daqacx1);
175  hold on;
176  plot([1:1:length(daqacx2)]*0.02 / 3600,daqacx2);
177  hold on;
178  plot([1:1:length(daqacx3)]*0.02 / 3600,daqacx3);
179  hold on;
180  plot([1:1:length(daqacx4)]*0.02 / 3600,daqacx4);
181  hold on;
182  plot([1:1:length(daqacx6)]*0.02 / 3600,daqacx6);
183  legend('1DAQ','2DAQ','3DAQ','4DAQ','6DAQ');
184  title('Acceleration-X Totals Comparison');
185  xlabel('t [h]');
186  ylabel('Acceleration-X [m/s^2]');
187
188  figure;
189  plot([1:1:length(daqacy1)]*0.02 / 3600,daqacy1);
190  hold on;
191  plot([1:1:length(daqacy2)]*0.02 / 3600,daqacy2);
192  hold on;
193  plot([1:1:length(daqacy3)]*0.02 / 3600,daqacy3);
194  hold on;
195  plot([1:1:length(daqacy4)]*0.02 / 3600,daqacy4);
```

```
196 hold on;
197 plot([1:1:length(daqacy6)]*0.02 / 3600,daqacy6);
198 legend('1DAQ','2DAQ','3DAQ','4DAQ','6DAQ');
199 title('Acceleration-Y Totals Comparison');
200 xlabel('t [h]');
201 ylabel('Acceleration-Y [m/s^2]');
202
203 figure;
204 plot([1:1:length(daqacz1)]*0.02 / 3600,daqacz1);
205 hold on;
206 plot([1:1:length(daqacz2)]*0.02 / 3600,daqacz2);
207 hold on;
208 plot([1:1:length(daqacz3)]*0.02 / 3600,daqacz3);
209 hold on;
210 plot([1:1:length(daqacz4)]*0.02 / 3600,daqacz4);
211 hold on;
212 plot([1:1:length(daqacz6)]*0.02 / 3600,daqacz6);
213 legend('1DAQ','2DAQ','3DAQ','4DAQ','6DAQ');
214 title('Acceleration-Z Totals Comparison');
215 xlabel('t [h]');
216 ylabel('Acceleration-Z [m/s^2]');
```

5.4 Examples and Explanation of Resulting Plots

In total, the MatLab file “DataEvaluationDDS” will provide 20 plots for a single dataset. “DataEvaluationDDSTotals” generates 9 plots for each of the 5 sensors and “DataEvaluationDDSComparison” another 9 for a final analysis of the total results.

For the explanation of the “DataEvaluationDDS” plots, the dataset “4DAQ40” will again be used and for the “DataEvaluationDDSTotals” plots, the results of sensor 4 are shown.

The plots shown on the following pages are in the same order as the resulting diagrams in the MatLab programs and the x-, y- and z-directions for the acceleration and displacement plots are defined as followed:

The comparison plot and a discussion and analysis of the results for all 5 dynamometers will then be presented in chapter 6.

1. **“DataEvaluationDDS” Plots**

- **Acceleration Plots (x-, y- & z-direction)**

Plot Name (3 Plots)	Messdaten/Acceleration-X/Y/Z
Y-Axis	Acceleration in x-, y-, z-direction [m/s^2]
X-Axis	Time [s]

Table 5: Details of acceleration plots

The first 3 plots, which are shown in Figure 27, 28 & 29, present the calibrated acceleration data on the y-axis in m/s^2 versus the time on the x-axis in seconds. As seen in all 3 plots, there is a considerable amount of noise and data outliers. Without these exceptions the acceleration lies between 7 and 10,5 m/s^2 in x-direction, between 0,5 and 2 m/s^2 in y-direction and between 5 and 6,5 m/s^2 in z-direction. When zoomed-in closely, a symmetrical shape can be recognized for all three acceleration directions. An example of the zoomed-in plot in x-direction is shown on the bottom right of Figure 27.

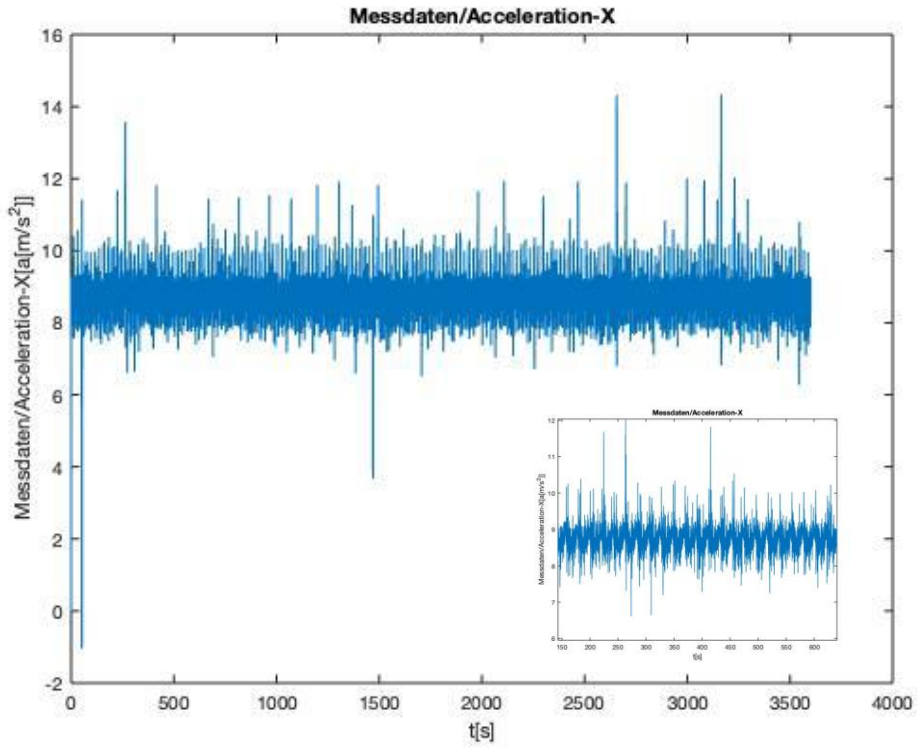


Figure 27: Acceleration plot in x-direction

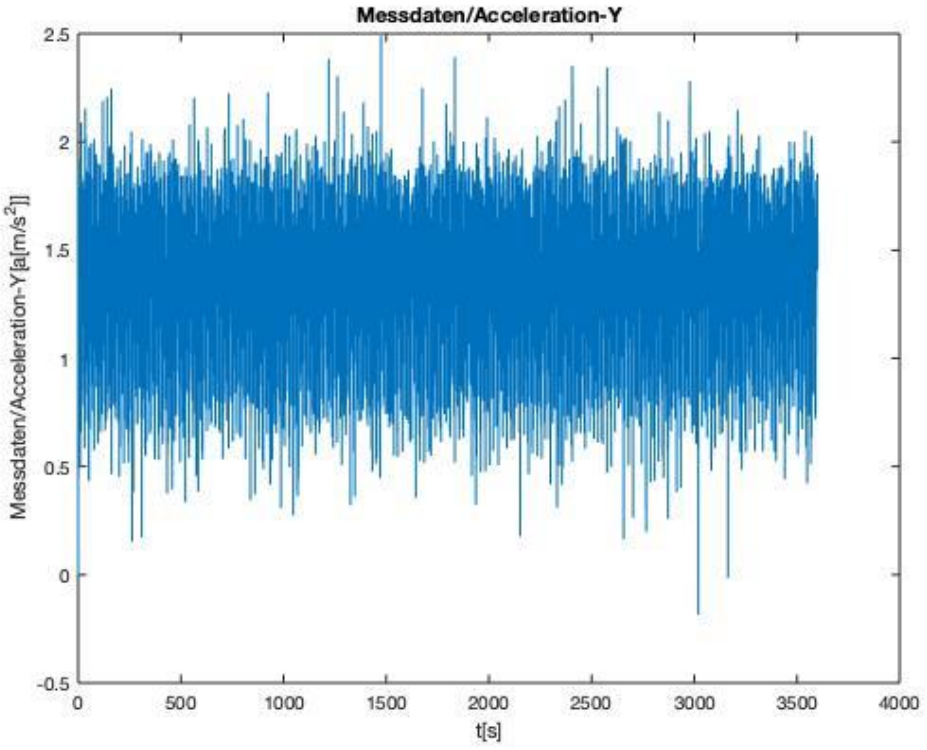


Figure 28: Acceleration plot in y-direction

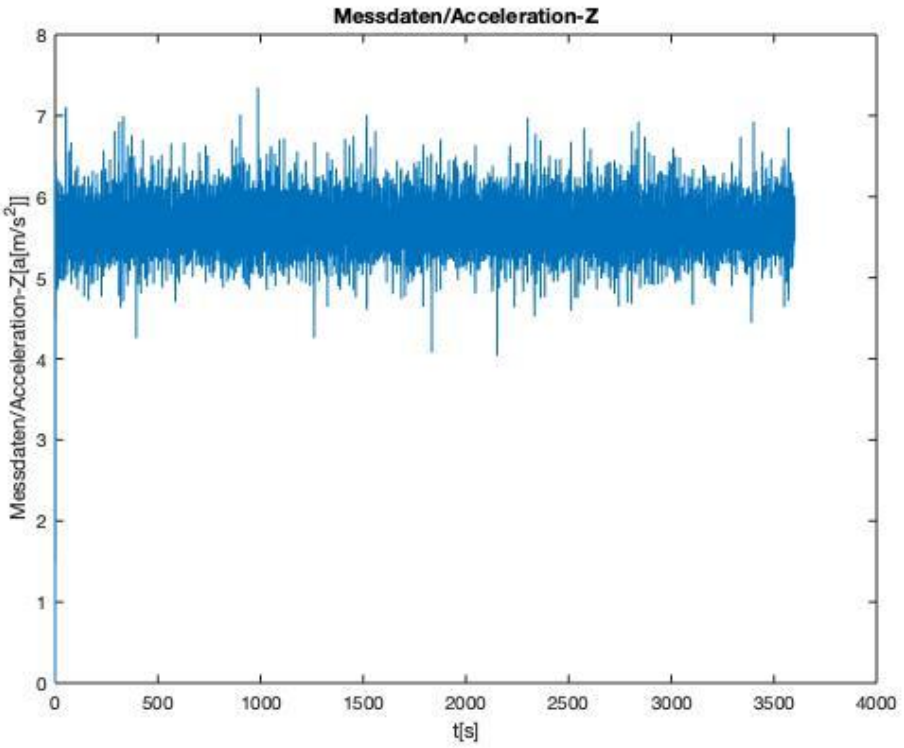


Figure 29: Acceleration plot in z-direction

The zoomed-in plot of the acceleration in x-direction for the dataset 4DAQ40 shows the upwards and downwards motion of the pump quite well. The outliers seen in this picture are the reason why it was so important to create the aforementioned Simulink file “Signal_Dem_sep_filt” for the 2 integration steps. Without filtering of the outliers, the results of the integrations, which lead to the displacement plots, would be falsified.

- **Battery Voltage Plot:**

Plot Name (1 Plot)	Messdaten/Battery Voltage
Y-Axis	Battery Voltage of the Sensor [mV]
X-Axis	Time [s]

Table 6: Details of battery voltage plot

Figure 30 shows the plot of the battery voltage of the downhole dynamometer sensor number 4. It remains at an almost constant value over the 1-hour recording period with only a few small outliers.

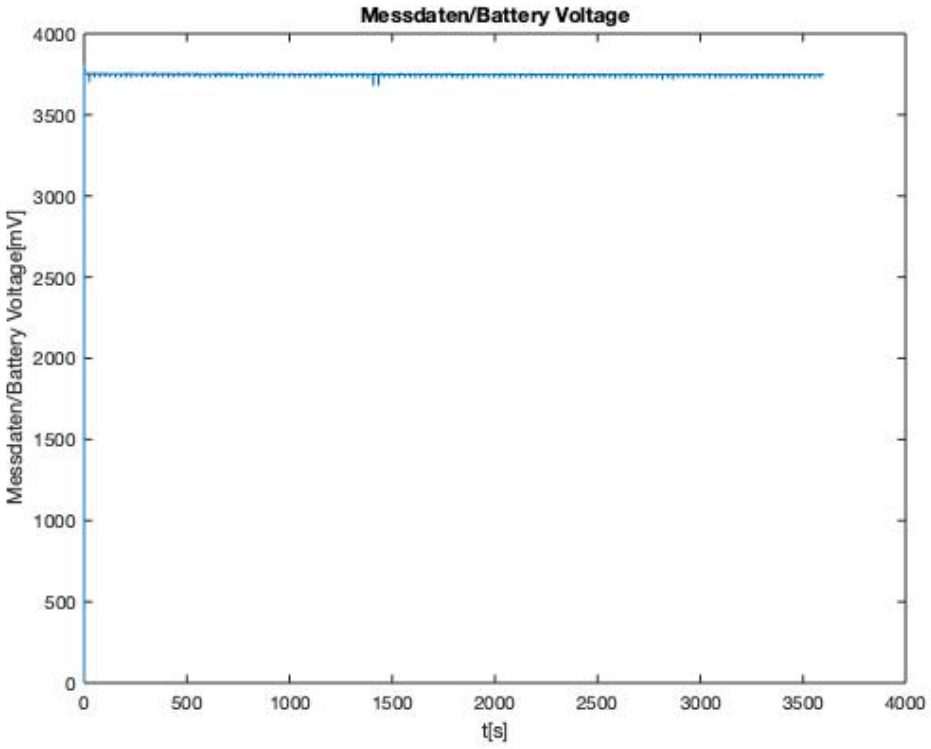


Figure 30: Plot of sensor's battery voltage

- **Single-Sided Amplitude Spectrum Plots**

Plot Name (3 Plots)	Single-Sided Amplitude Spectrum y(t)
Y-Axis	Amplitude [-]
X-Axis	Frequency [Hz]

Table 7: Details of single-sided amplitude spectrum plots

- **Strain Gauge Plots**

Plot Name (3 Plots)	Messdaten/Analog 1/2/3
Y-Axis	Load [N]
X-Axis	Time [s]

Table 8: Details of strain gauge plots

Figures 31, 32 & 33 show the load plots of the 3 strain gauges DMS1, DMS2 and DMS3 over 1 recording hour and zoomed in and also the corresponding plots of the single-sided amplitude spectrum.

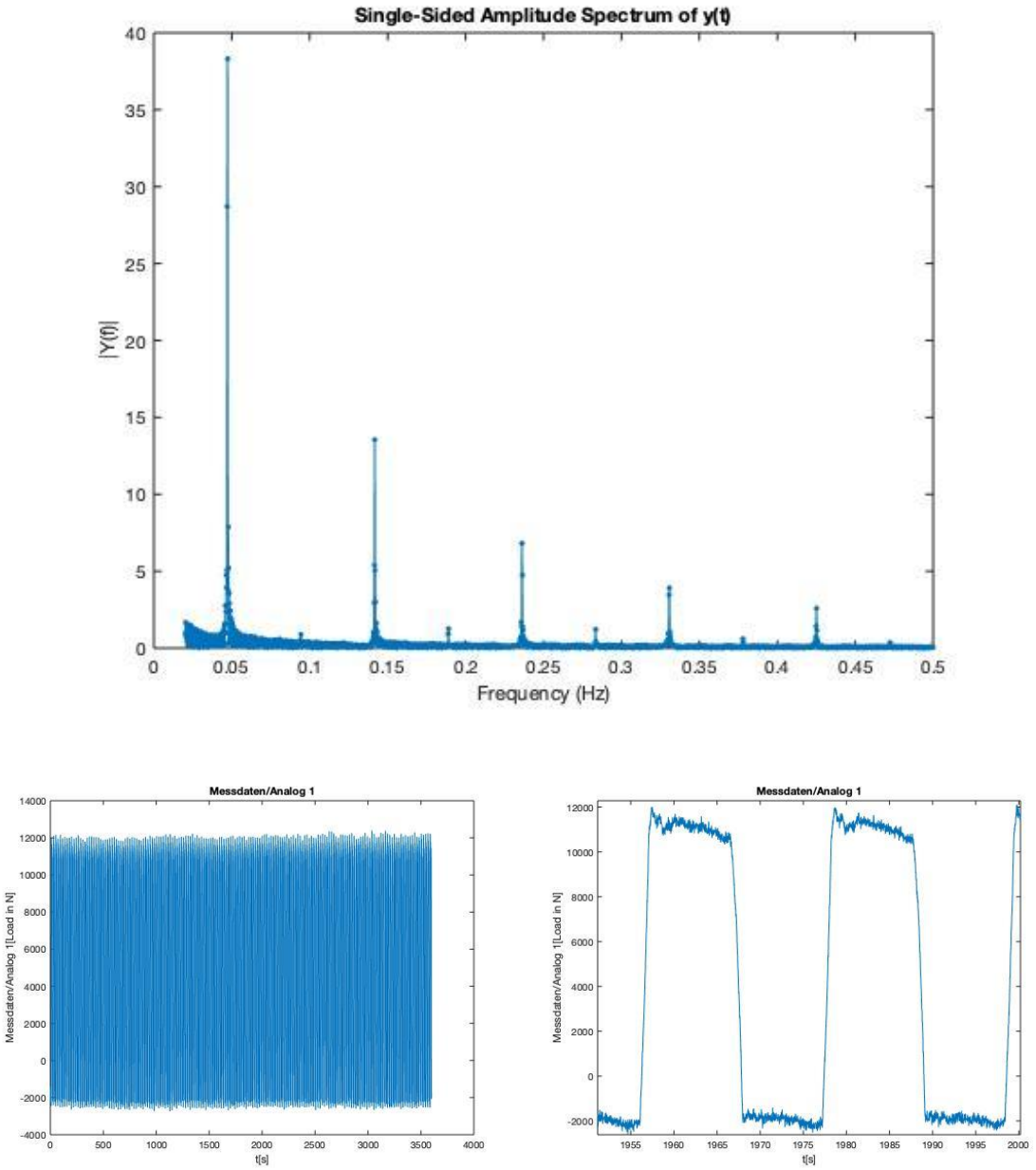


Figure 31: Single-sided amplitude spectrum & load plot of DMS1 (full & zoomed-in)

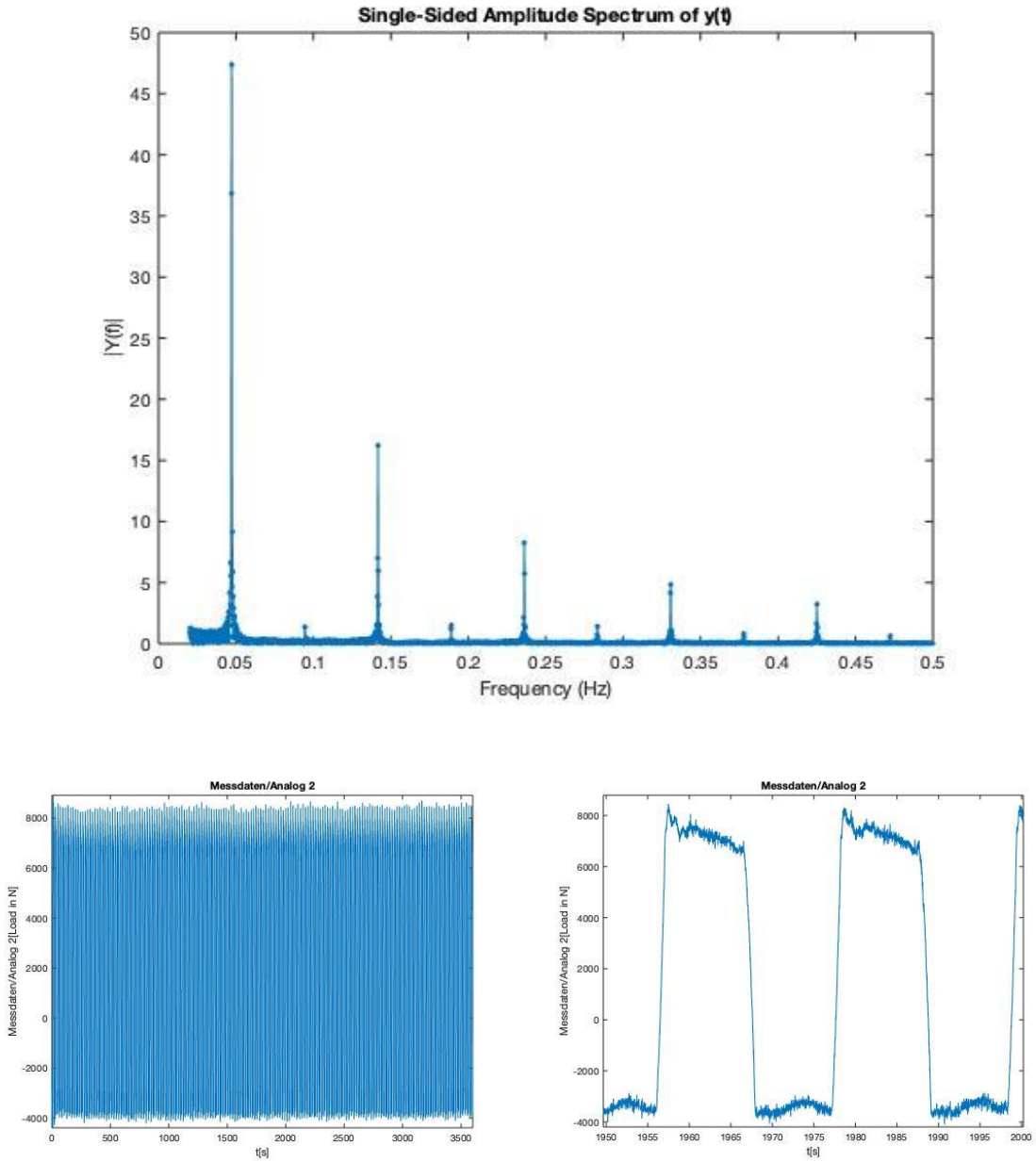


Figure 32: Single-sided amplitude spectrum & load plot of DMS2 (full & zoomed-in)

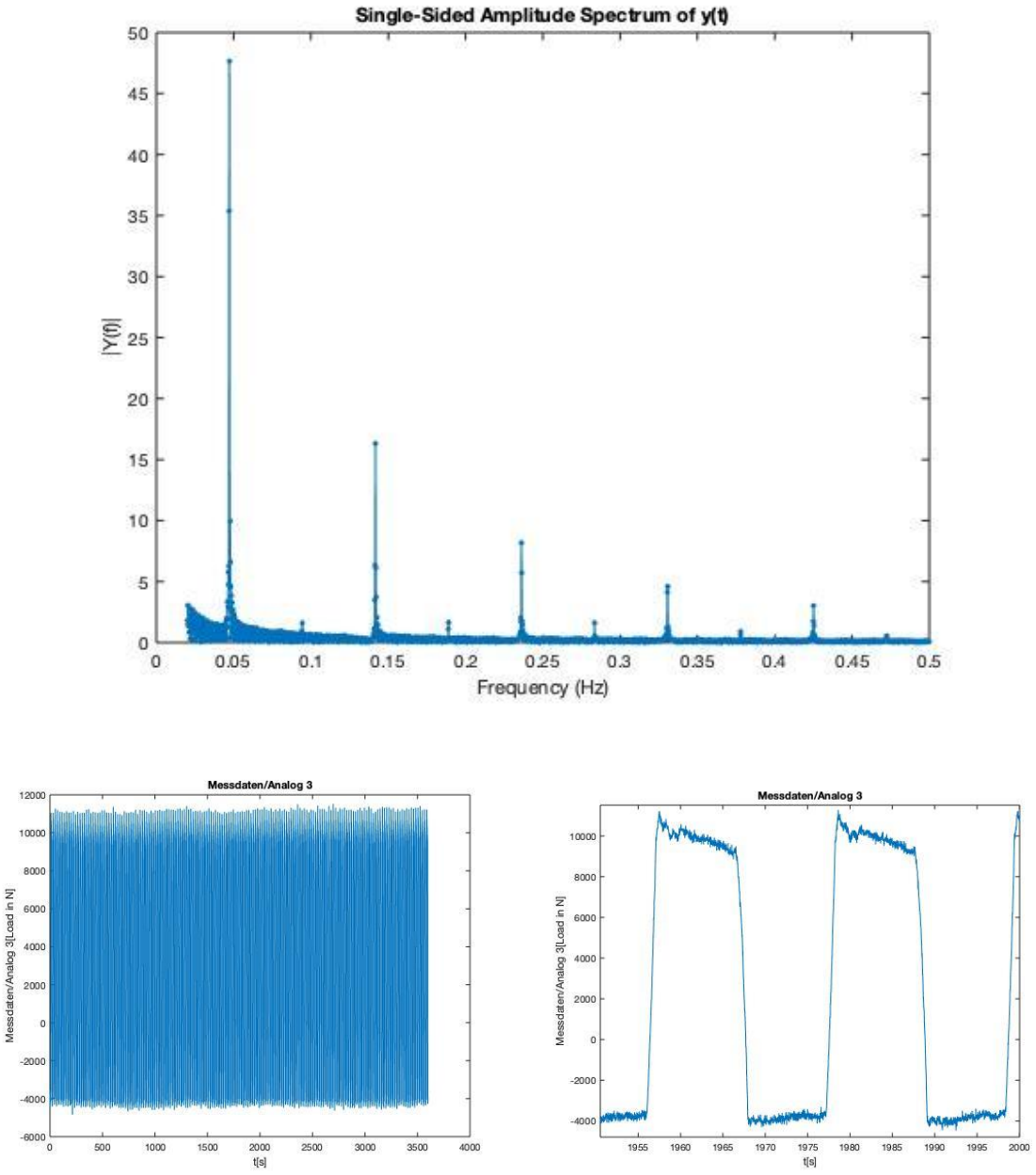


Figure 33: Single-sided amplitude spectrum & load plot of DMS3 (full & zoomed-in)

- **Temperature Plot:**

Plot Name (1 Plot)	Messdaten/Analog 4
Y-Axis	Temperature [°C]
X-Axis	Time [s]

Table 9: Details of temperature plot

The next plot in Figure 34 shows the calibrated temperature data over the duration of the 1-hour recording. The variation in the first 500 seconds could be due to the systems need for stabilization and may be neglected. Afterwards the temperature remains relatively constant between 37,3 and 37,5 degrees Celsius for the dataset 4DAQ40.

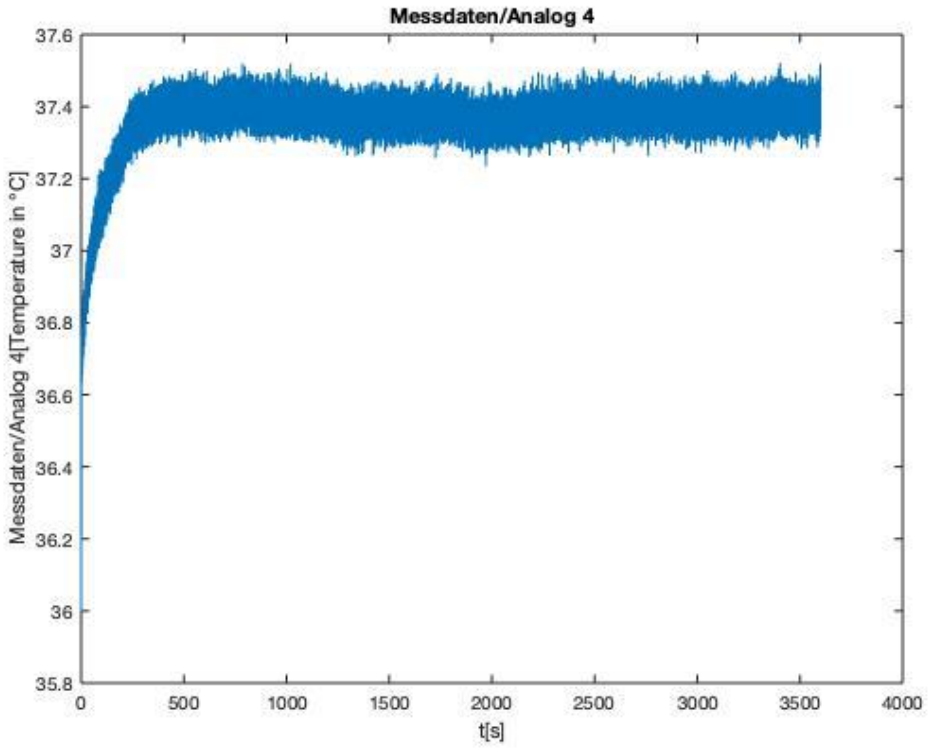


Figure 34: Temperature plot

- **Strain Gauges Mean Value Plot:**

Plot Name (1 Plot)	Mean DMS
X-Axis	Load [N]
Y-Axis	Time [s]

Table 10: Details of strain gauges mean value plot

After the temperature plot, the mean value of all 3 strain gauges will be calculated by the aforementioned external MatLab file “DMSMeanValue” and Figure 35 shows the result. It is important to zoom-in closely into the plot because it provides an excellent analysis of the

duration of one pump cycle and can be compared to the calculated value of the period duration. The symmetrical zoomed-in plot of the strain gauges mean value is pictured in Figure 36.

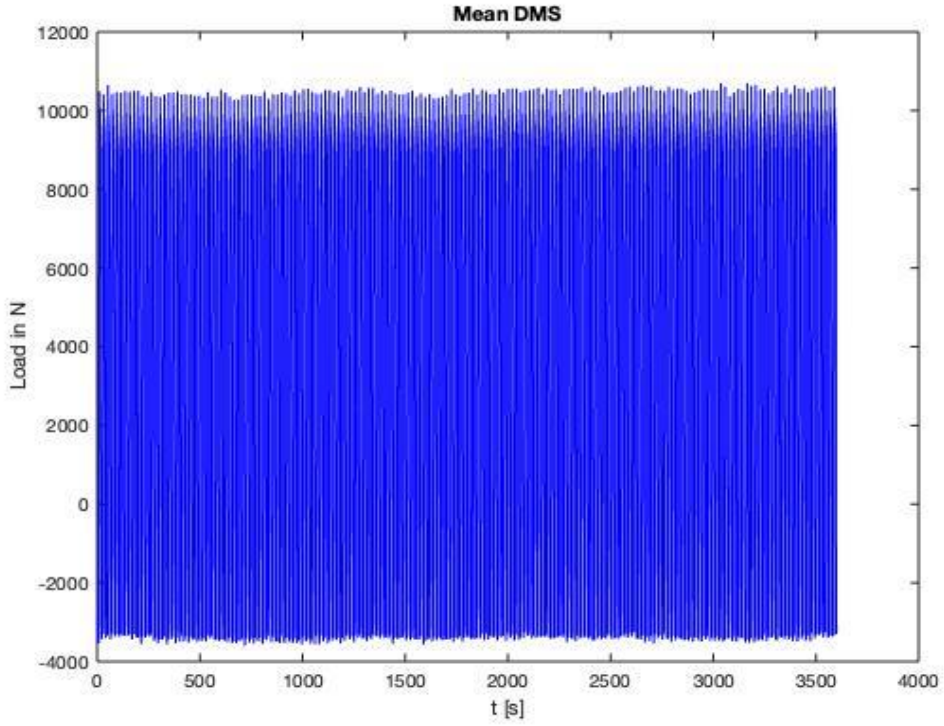


Figure 35: Strain gauges mean value plot

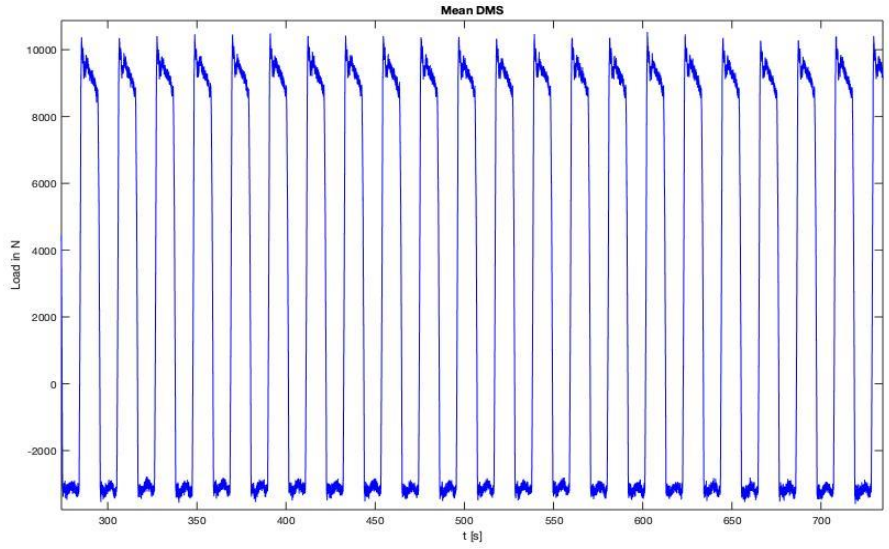


Figure 36: Zoomed-in strain gauges mean value plot

- **Displacement Plots (x-, y- & z-direction):**

Plot Name (6 Plots)	Displacement-X/Y/Z & Average displacement in x/y/z
Y-Axis	Displacement [m]
X-Axis	Time [s]

Table 11: Details of displacement plots

Afterwards the Simulink file “Signal_Dem_sep_filt”, which filters the noise and data outliers of the acceleration data, performs the 2 integration steps, which then leads to the displacement data in x-, y- and z-direction.

One main difference between displacements plots in x-, y- and z-direction is the magnitude of the displacement. In Figure 37 it can be seen that the displacement in x-direction is in meters, while the one in y-direction is in 10^{-3} meters (Figure 38) and in z-direction (Figure 39), the values on the y-axis are in 10^{-4} meters.

In the end, the resulting displacement values are also averaged per period, which can be seen in Figures 40, 41 & 42.

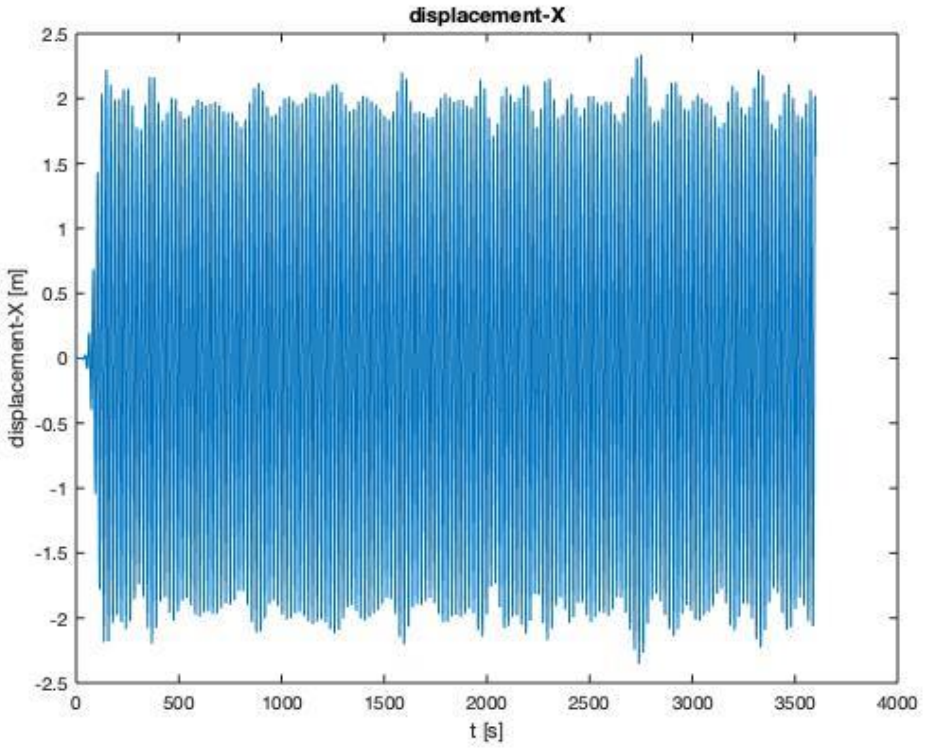


Figure 37: Displacement plot in x-direction

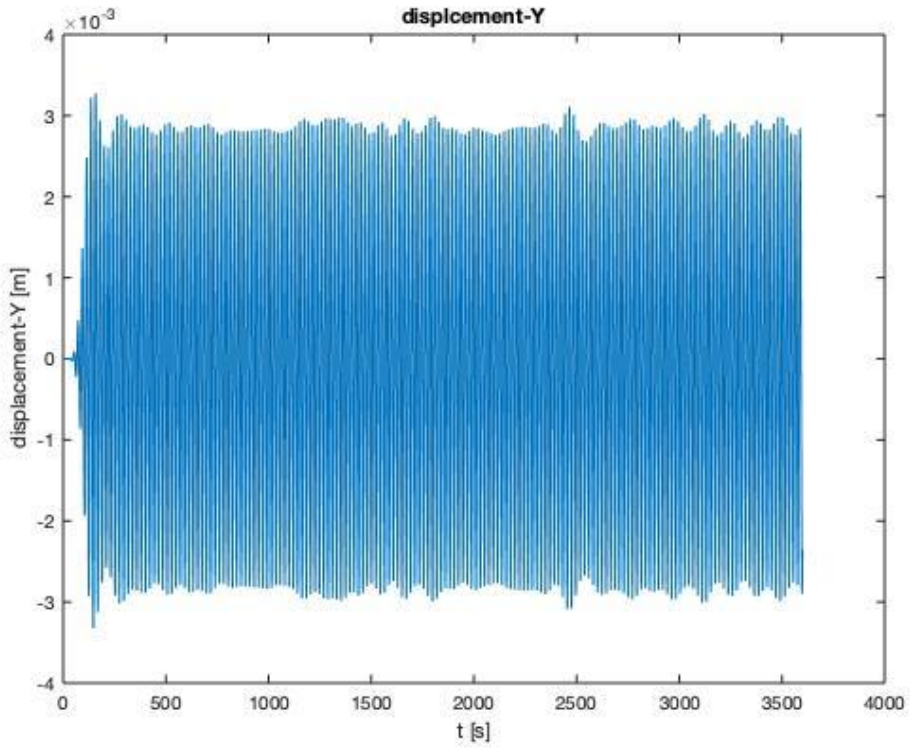


Figure 38: Displacement plot in y-direction

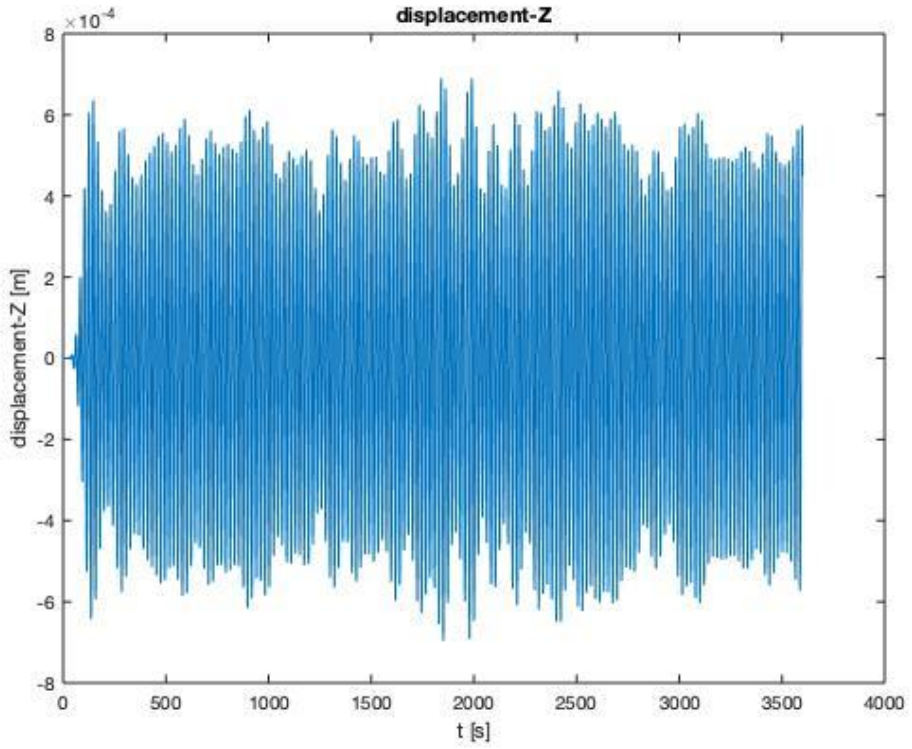


Figure 39: Displacement plot in z-direction

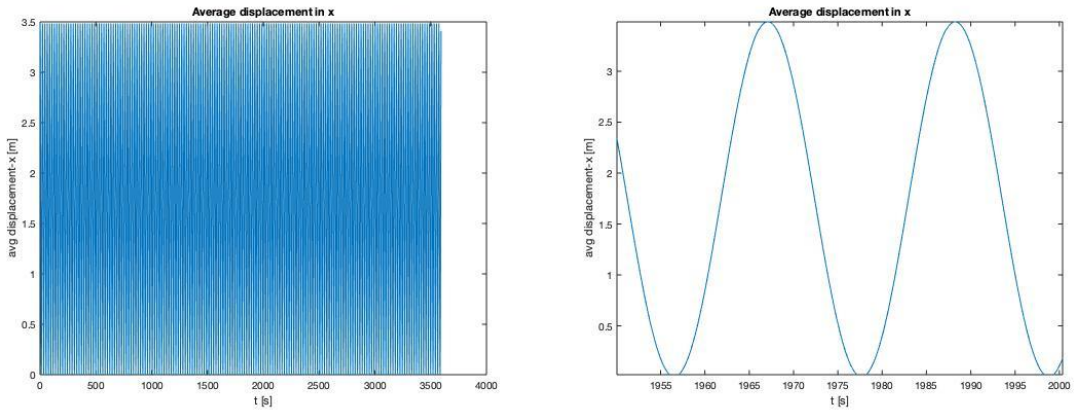


Figure 40: Average displacement plot in x-direction (full & zoomed-in)

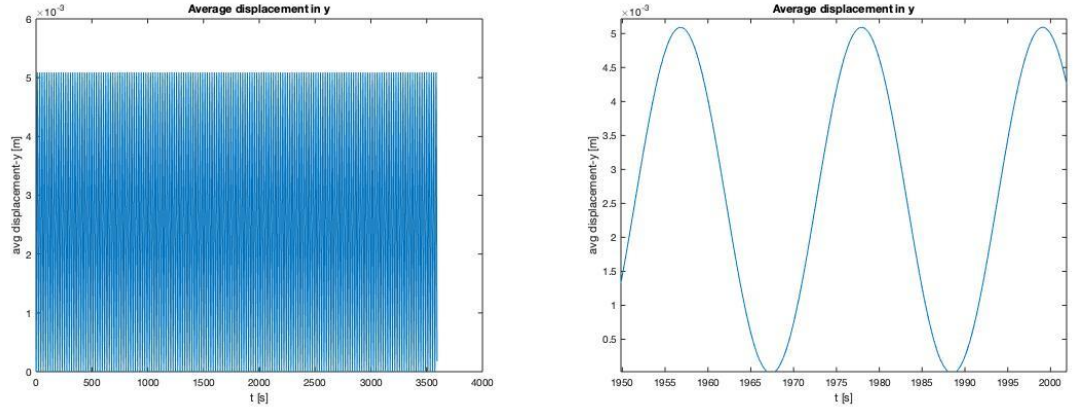


Figure 41: Average displacement plot in y-direction (full & zoomed-in)

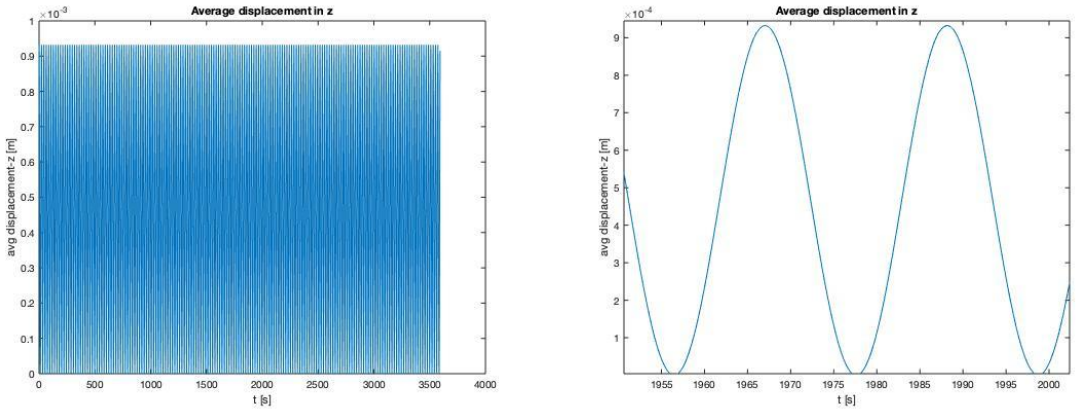


Figure 42: Average displacement plot in z-direction (full & zoomed-in)

- **Load vs. Position Plots**

Plot Name (2 Plots)	Load vs. Position
Y-Axis	Load [N]
X-Axis	Position [m]

Table 12: Details of load versus position plots

The final 2 plots present the load versus position diagrams that can be used to evaluate the performance of the pump. For those diagrams, the values of the displacement in x-direction and the strain gauge mean values are used.

As mentioned in the previous sub-chapter, plot 1, which is shown in Figure 43, uses the data starting from the 10th period duration until period duration 150, which results in an overlap of all pump cycles during this time period. It is possible to use either the average displacement values in x-direction (left) or the actual ones (right).

Plot 2, shown in Figure 44, only plots the load versus the position for a single period duration.

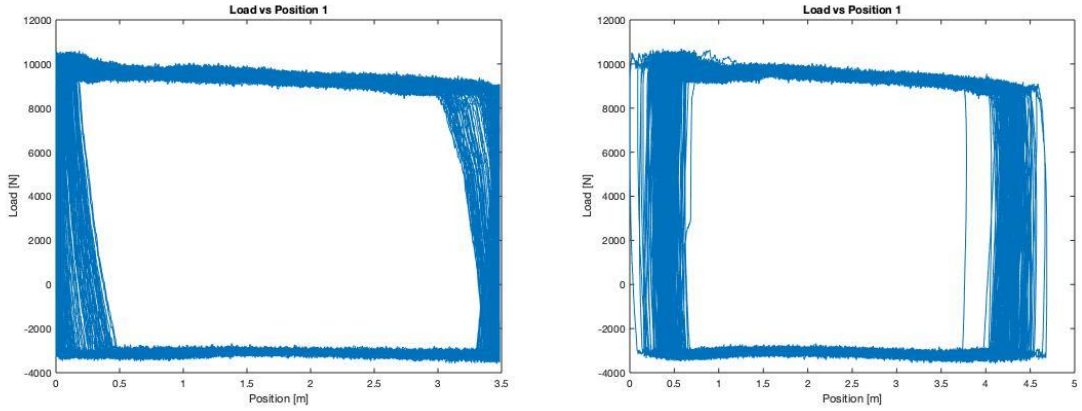


Figure 43: Load vs. position diagram with values from period duration 10 to period duration 150

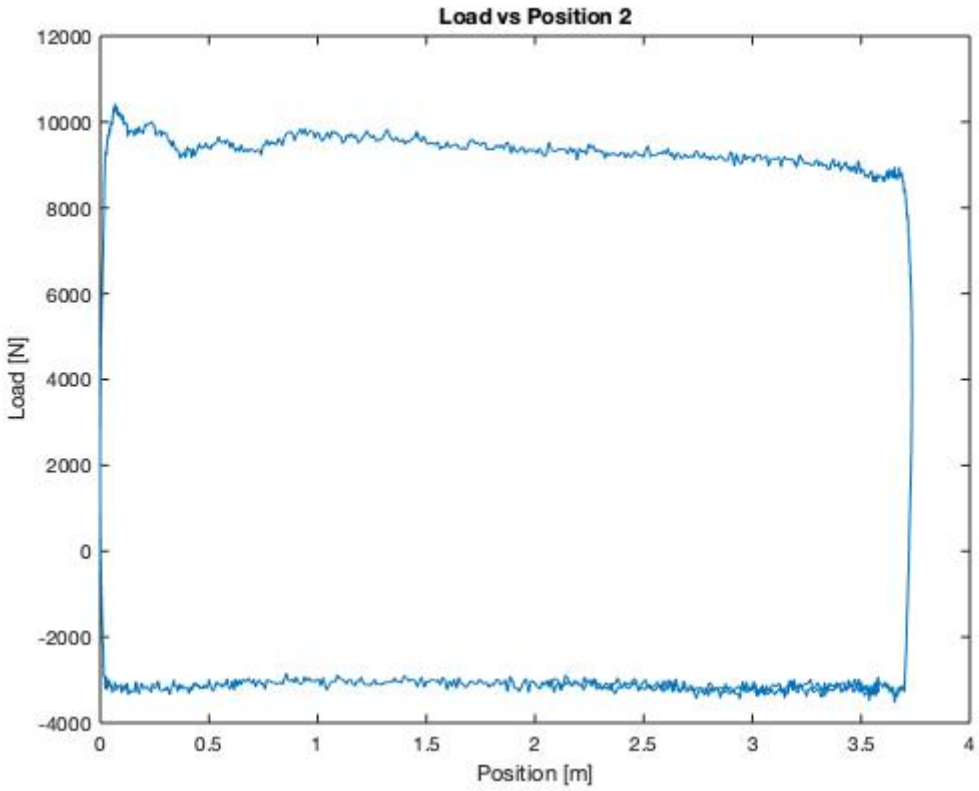


Figure 44: Load vs. position diagram over one period duration

2. "DataEvaluationDDSTotals" Plots (Sensor 4 used as example)

- **Acceleration Plots (x-, y- & z-direction)**

Plot Name (3 Plots)	Acceleration X/Y/Z Totals
Y-Axis	Acceleration in x-, y-, z-direction [m/s2]
X-Axis	Time [h]

Table 13: Details of acceleration plots over total recording period

Since the acceleration plots of "DataEvaluationDDSTotals" span over the entire recording period of each sensor, the unit of the time on the x-axis was changed to hours. One important analysis that can already be made by looking at the 3 plots in Figures 45, 46 and 47, is that the pump was not running for the first 12 recording hours.

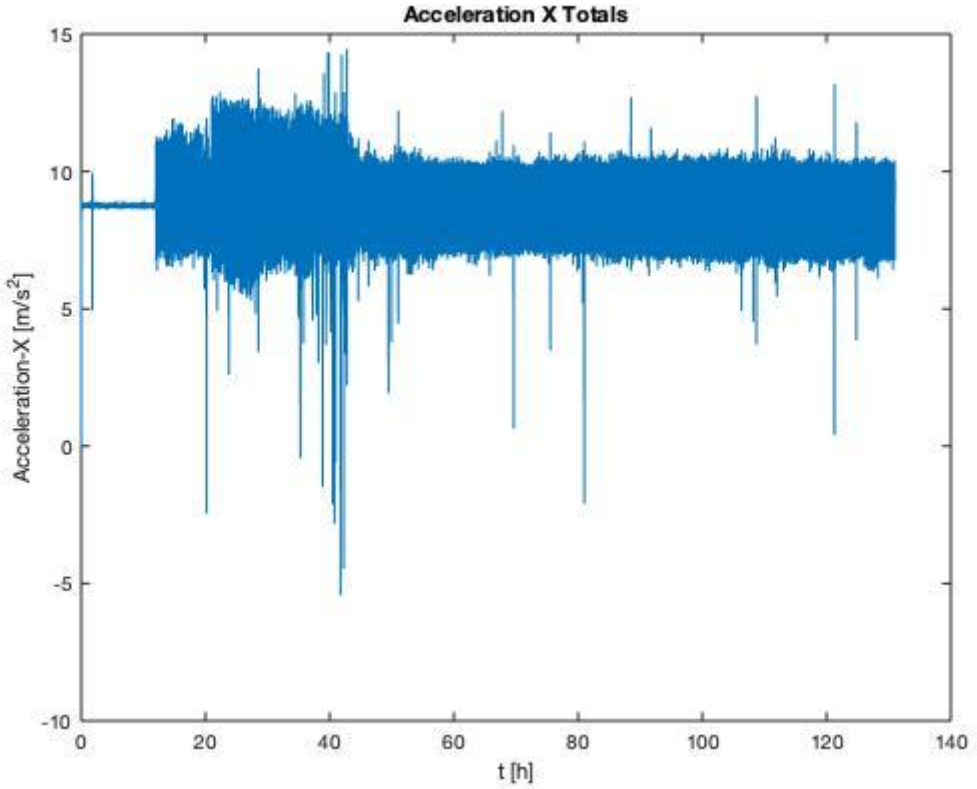


Figure 45: Acceleration plot in x-direction over total recording period

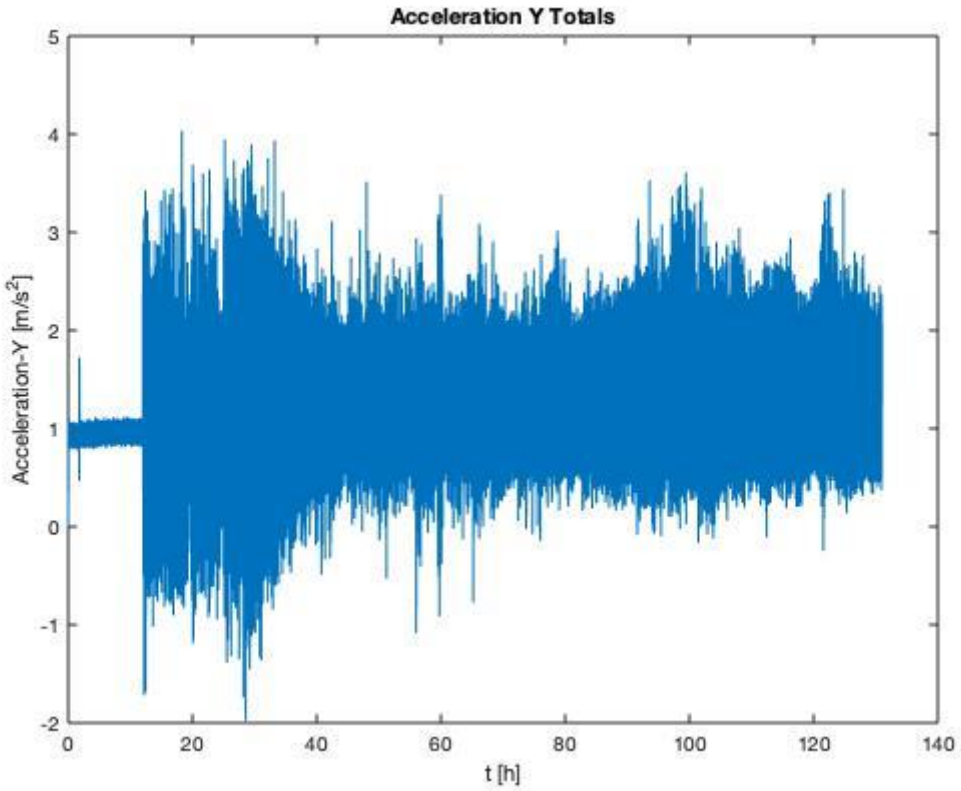


Figure 46: Acceleration plot in y-direction over total recording period

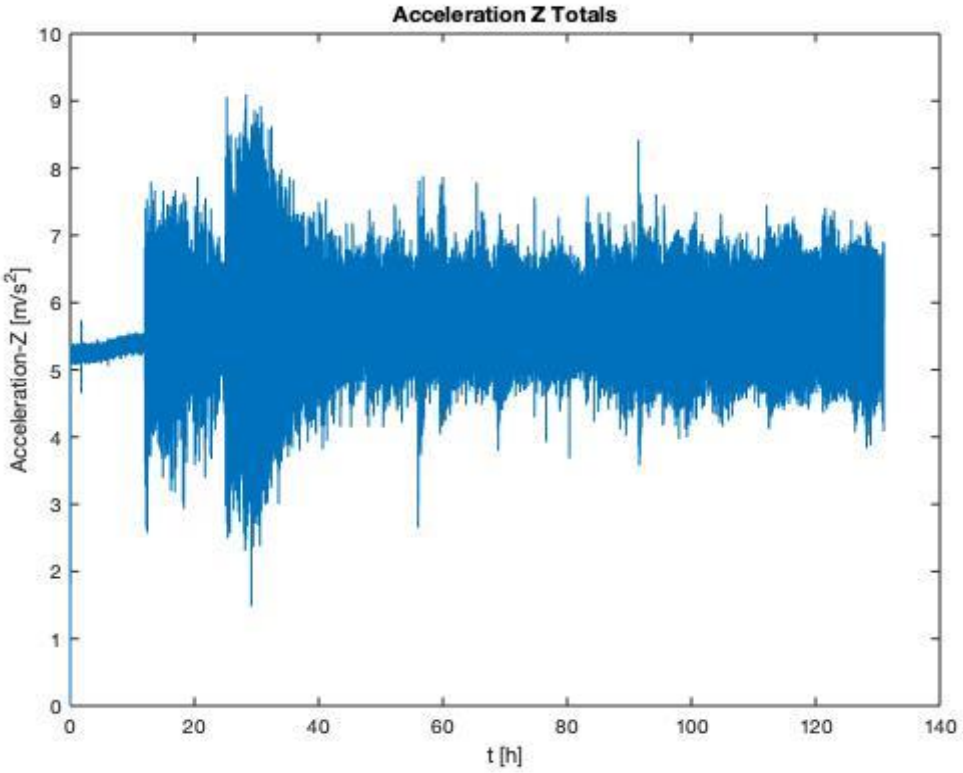


Figure 47: Acceleration plot in y-direction over total recording period

- **Battery Voltage Plot:**

Plot Name (1 Plot)	Battery Voltage Totals
Y-Axis	Battery Voltage of the Sensor [mV]
X-Axis	Time [h]

Table 14: Details of battery voltage plot over total recording period

As expected, the battery voltage over the entire recording period of sensor 4, shown in Figure 48, gradually declines, but even after 131 hours of recording, the voltage still has a solid value of around 3200 milli Volt.

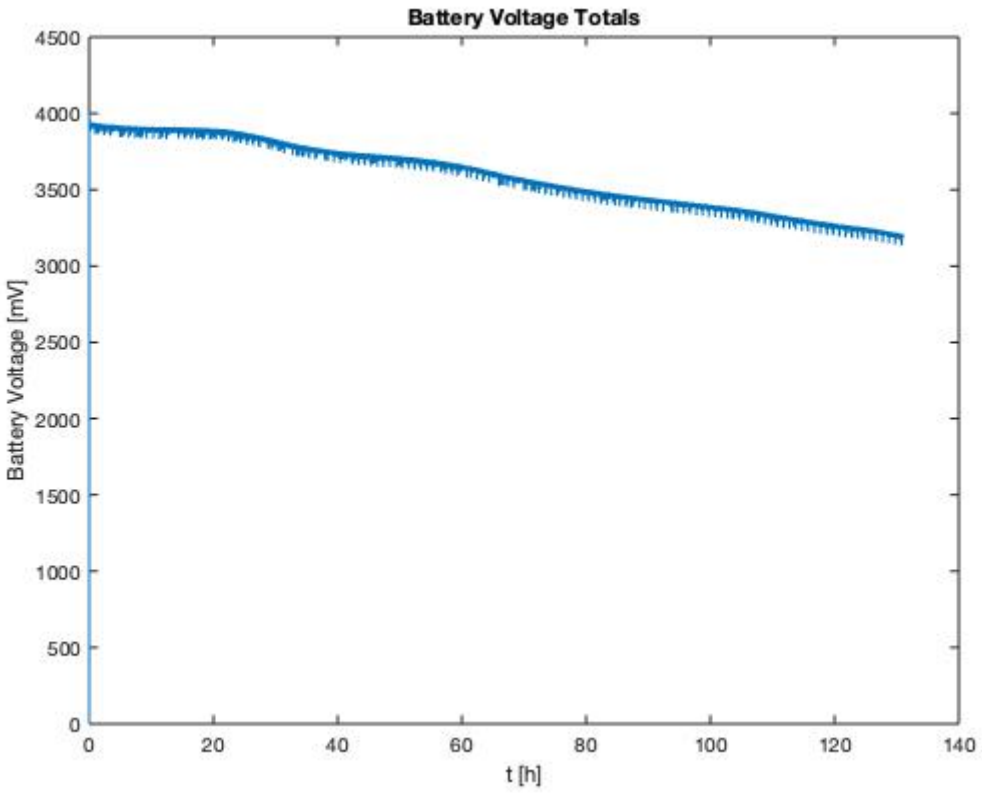


Figure 48: Battery voltage plot over total recording period

- **Temperature Plot:**

Plot Name (1 Plot)	Temperature Totals
Y-Axis	Temperature [°C]
X-Axis	Time [h]

Table 15: Details of temperature plot over total recording period

The plot of the temperature totals over the whole recording period is also affected by the pump downtime in the beginning of the recording and the 2 outliers at the end might be caused by pressure releases from the casing.

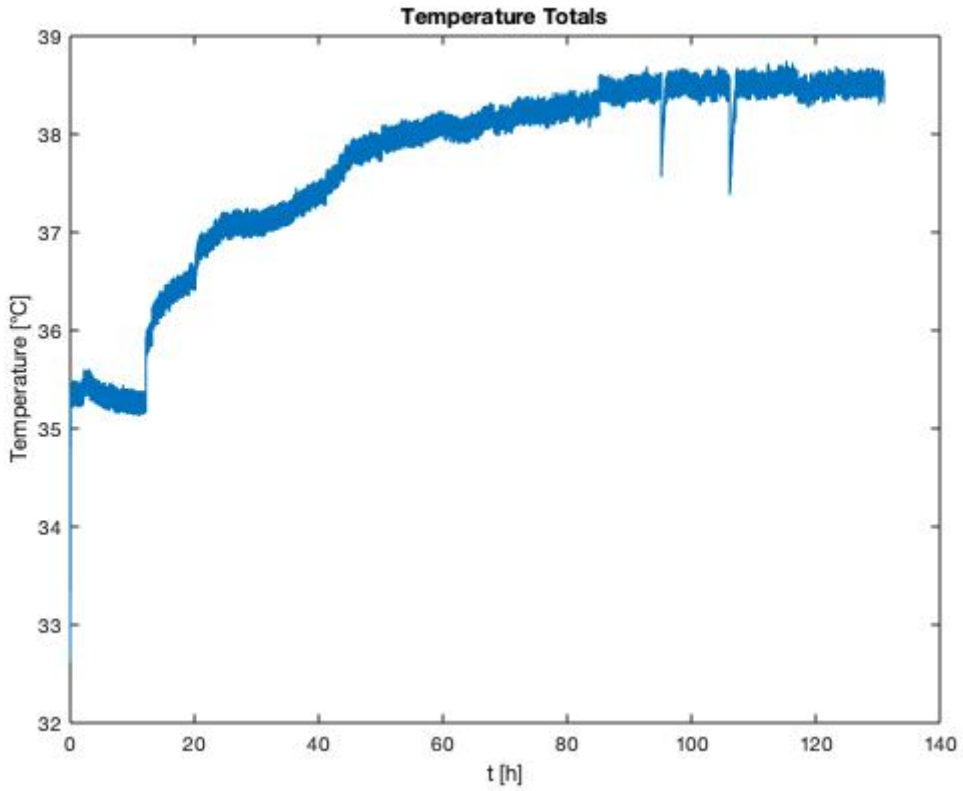


Figure 49: Temperature plot over total recording period

- **Displacement Plots (x-, y- & z-direction):**

Plot Name (3 Plots)	Displacement-X/Y/Z Totals
Y-Axis	Displacement [m]
X-Axis	Time [h]

Table 16: Details of displacement plots over total recording period

The displacement plots in x-, y- and z-direction provide a great opportunity to identify the maximum displacement values. For example, in Figure 50, it can be seen that the maximum displacement in x-direction for downhole dynamometer sensor is just under 5 meters. Figure 51 and 52 then show the displacement totals in y- and z-direction.

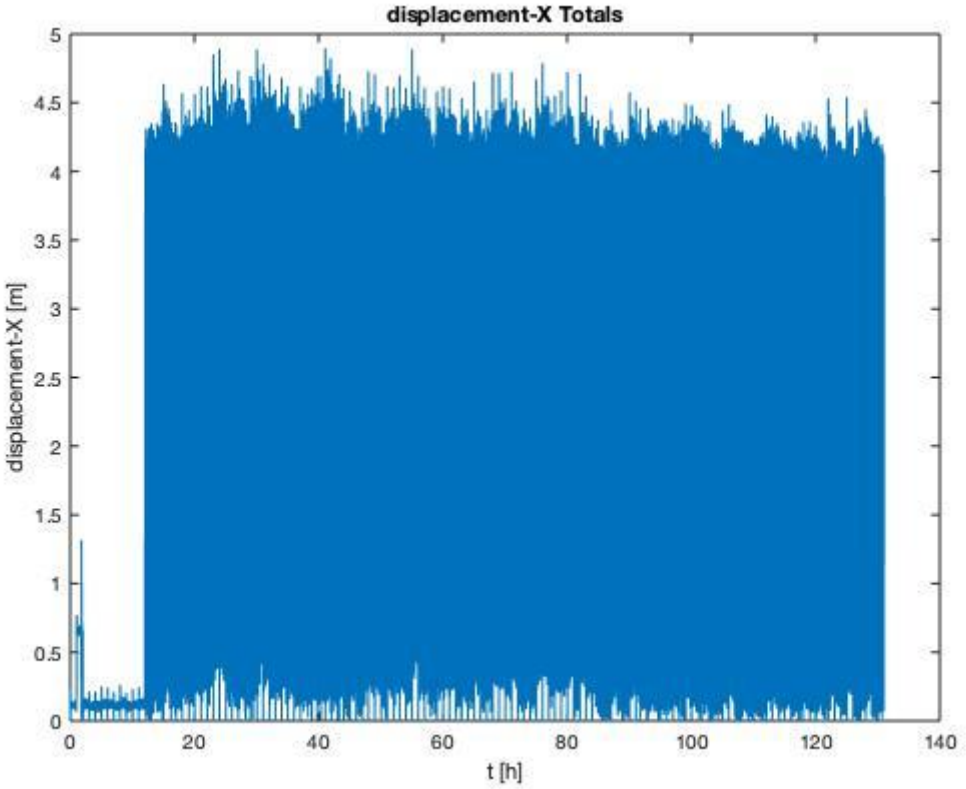


Figure 50: Displacement plot in x-direction over total recording period

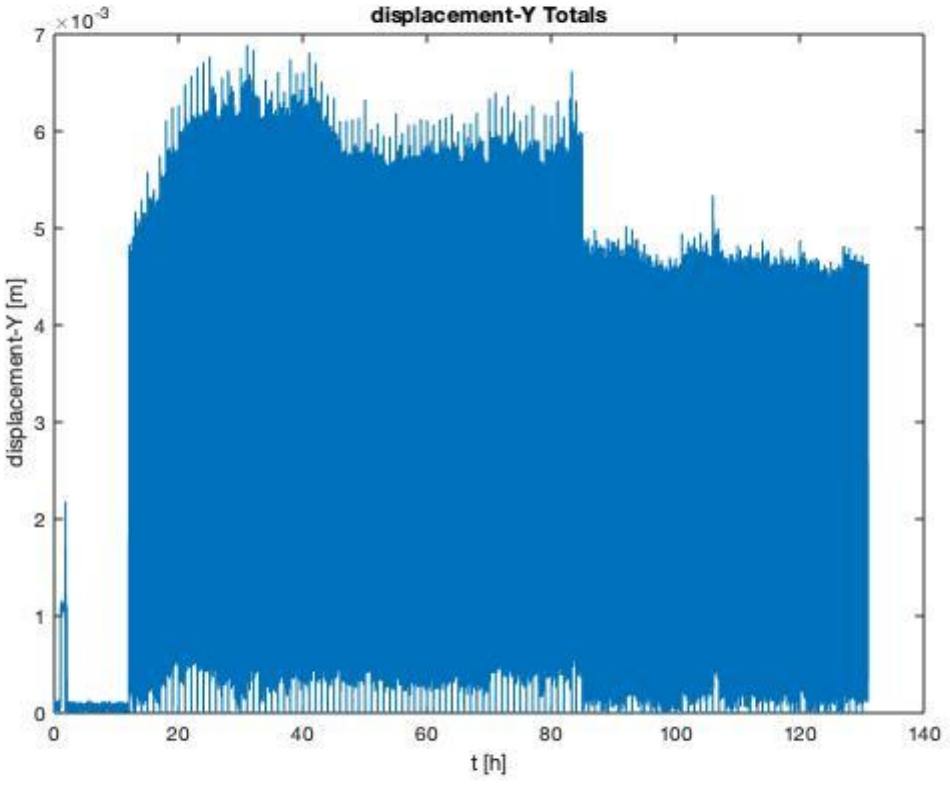


Figure 51: Displacement plot in y-direction over total recording period

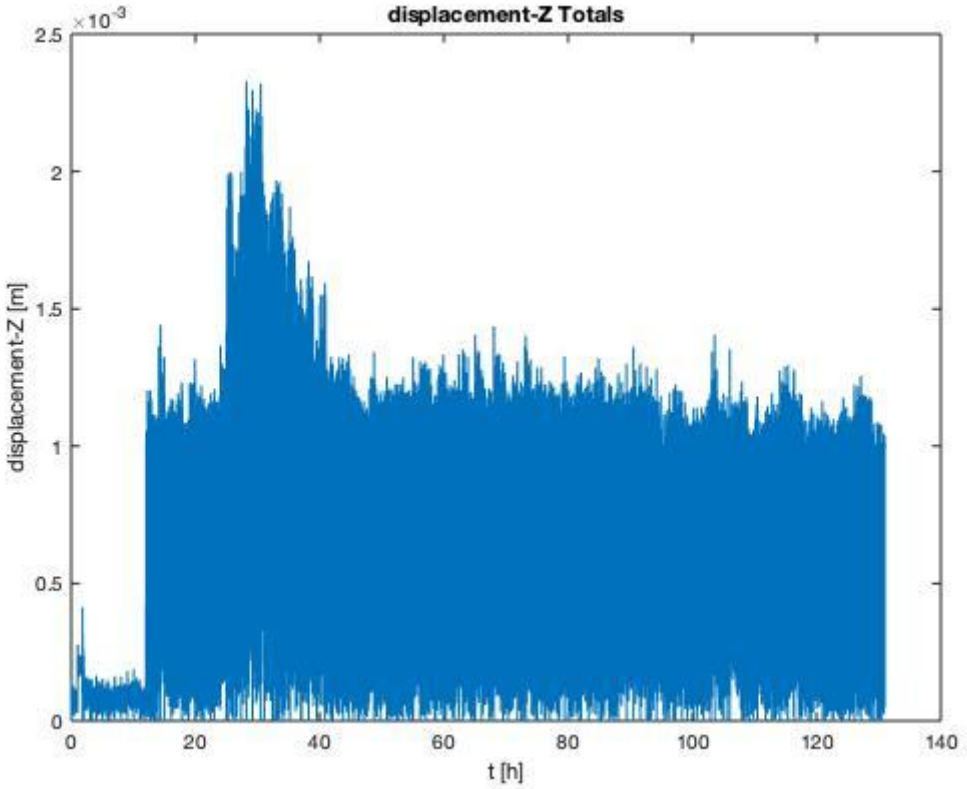


Figure 52: Displacement plot in z-direction over total recording period

- **Strain Gauges Mean Value Plot:**

Plot Name (1 Plot)	DMS Mean Value Totals
X-Axis	Load [N]
Y-Axis	Time [h]

Table 17: Details of strain gauges mean value plot

Figure 53 shows the plot of the strain gauges mean value over the whole recording period. Again, it can be seen, that the pump was not turned on in the beginning.

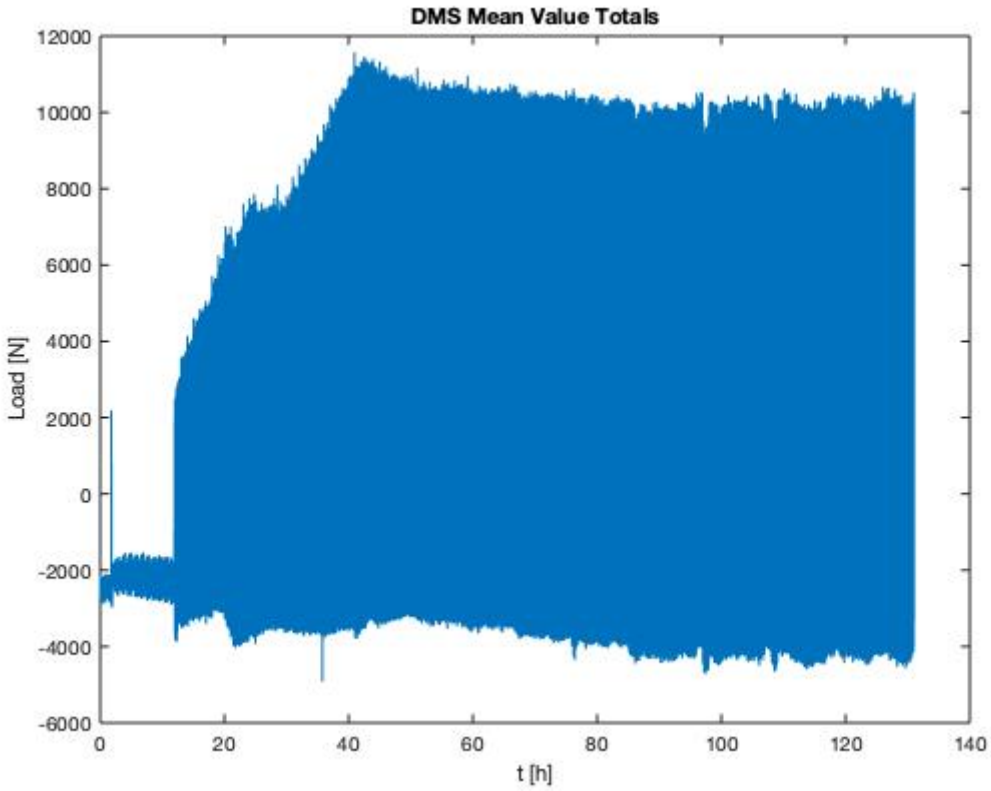


Figure 53: DMS mean value plot over total recording period

6 Data Evaluation of Downhole Dynamometer Field Example

In order to test the effectiveness of downhole dynamometer sensors, an oil well in the Vienna Basin was chosen as the location of a field example. 5 sensors were installed at different points of interest along the sucker rod string and the resulting data sets were then processed and evaluated by using aforementioned MatLab files.

6.1 Details of Vienna Basin Oil Well

General information about the well, its production, the pump jack and also the downhole pump are listed in table 18 (Langbauer et al., 2019):

Depth of reservoir layer	955m true vertical depth
Kick-off point	480m true vertical depth
End of build	800m true vertical depth
Inclination at end of build	28°
Pump jack designation	C-320D-256-144
Surface stroke length	3,7m
Type of downhole pump	Tubing pump
Downhole pump designation	25-225-TH-18-4
Plunger size	2,25in
Pump strokes per minute (before field test)	2,77
Oil production per day	1,2m³
Water production per day	6,6m³
Water Cut	66%
Solution gas ratio	23 sm³/m³ of gas per m³ of oil
Prime Mover Driver	Variable speed drive
Type of engine	Asynchronous electrical engine
Tubing string size	2 7/8”
Tubing string anchor depth	884m true vertical depth (906m measured depth)

Table 18: General information about Vienna Basin oil well

Furthermore, a rod rotator and a system for corrosion inhibition were implemented in the well. No meaningful issues with precipitations of paraffins were present and the additional installation of 7/8" sucker rods was done specifically for this field test.

6.2 Positioning of Downhole Dynamometer Sensors

In this field example 5 sensors (DDS1, 2, 3, 4 & 6) were installed at different depths along the sucker rod string, which is shown in Figure 54. To ensure an easy handling, pony rods were used to screw the DDS into the rod string.

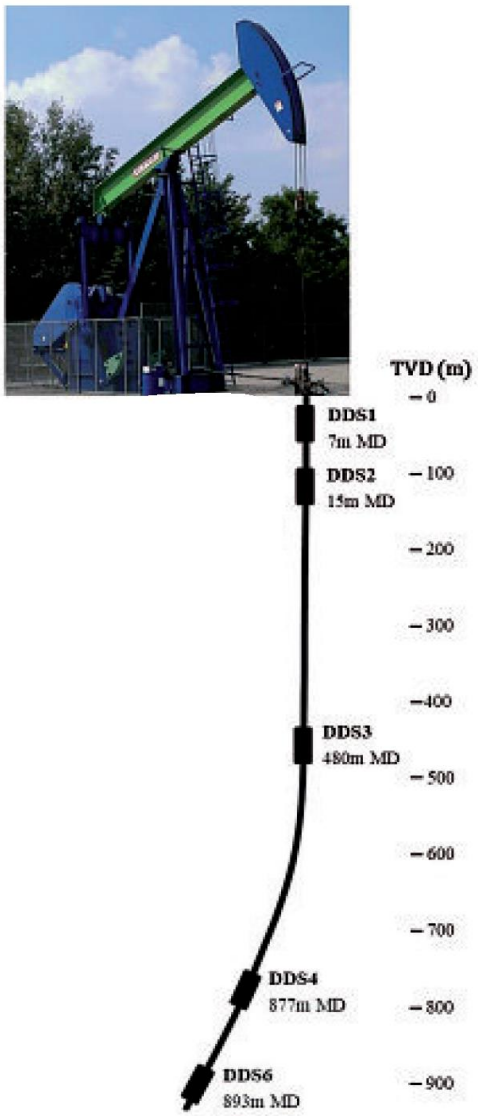


Figure 54: Positioning of downhole dynamometer sensors in Vienna Basin field example (Langbauer et al., 2019)

The positions of all 5 dynamometers were chosen for a particular reason and table 19 gives an overview about the installation depth and the logic behind the choice (Langbauer et al., 2019):

Sensor Number	Installation Depth (MD)	Installation Position	Reason
DDS1	7m	Positioned below the polished rod and near the top of the rod string	Recording of surface load in the sucker rod string
DDS2	15m	Positioned below the polished rod and near the top of the rod string	Backup sensor for DDS1
DDS3	480m	Positioned at kick-off point	Identification of behavior & friction differences between the vertical and deviated rod string sections
DDS4	877m	Two 24” rods above the pump plunger	Identification of buckling
DDS6	893m	Directly on top of the pump plunger	Identification of buckling

Table 19: Positioning details of downhole dynamometer sensors

The exact angle in which the sensors are positioned along the sucker rod string can be checked by comparing the acceleration results in x-, y- and z-direction at a time, when the pump has not been activated yet. Table 20 shows the results at recording hour 10 for each sensor, which indicate that sensors 1 and 2 are installed vertically, while sensors 4 and 6 are positioned in the deviated section.

Sensor Number	Acceleration in x-direction [m/s^2]	Acceleration in y-direction [m/s^2]	Acceleration in z-direction [m/s^2]
DDS1	9,6	-0,5	0,3
DDS2	10,2	-0,1	-0,1
DDS3	10,1	-0,2	0,1
DDS4	8,8	1	5,4
DDS6	8,7	-5	0,5

Table 20: Acceleration values of all 5 sensors at recording hour 10

6.3 Details of Testing Sequence

Overall, the duration of the entire field trial lasted for approximately 3 weeks. Before the installation of the sensors though, some standard procedures had to be implemented in order to help simplify the data evaluation (Langbauer et al., 2019):

1. Since there is a limit on the accumulator capacity and the data storage of downhole dynamometer sensors, the recordings were taken in 1-hour intervals → data were measured for 1 hour at a sampling frequency of 50Hz. Afterwards, the DDS were programmed to “sleep” for 1 hour.

2. The different positioning of the 5 downhole dynamometer sensors leads to a difference in temperature, which results in a deviation of the individual timestamps of each DDS → prior to the installation, the sensor times were adjusted and equalized with a computer. Once the field test was completed and the sensors were removed, they were placed in a tray and turned around by 180°. This accomplishes a reading on the acceleration sensor, which can then be used to rescale the timestamps.

3. After the stabilization of the production data, a check of the standing and travelling valve was completed by a gradual stop of the pump jack - once during each the upstroke and downstroke. The system was then maintained in these locations for some time → this check helps to not only identify the average density of the fluid in the tubing, but also provides an evaluation of the buoyant weight of the rod string.

4. The value of the strokes per minute (SPM) was changed several times since it has a major influence on the dynamic load behavior of the sucker rod string. The values were randomly chosen from a range of 2,9 to 3,3 SPM in order to avoid aliasing. A higher SPM (3,3) was chosen in the beginning in order to remove workover fluids from the annulus. Afterwards, the strokes per minute were reduced until they remained constant at a value of 3,1 after the first week. The reason for these changes was to increase the amount of information from the system, but in general, it is possible to operate the downhole dynamometer sensors at any SPM. In this case, the well’s poor inflow performance relationship additionally restricted the pumping speed.

6.4 Results / Interpretation

This subchapter is dedicated to discussing and interpreting the results of the downhole dynamometer sensor field example by using the aforementioned MatLab files. The following quantities will be analysed in detail:

- Temperature
- Battery voltage
- Acceleration in x-, y- and z-direction for an analysis of the acceleration
- Displacement in x-, y- and z-direction for a motion analysis
- Strain gauges mean value for a minimum and maximum load evaluation
- Load vs. position diagrams for an evaluation of the pump efficiency and load transfer

6.4.1 Temperature Analysis

To fully understand the temperature analysis, it should be noted, that the temperature sensor is located inside the downhole dynamometer sensor and determines its housing temperature, which then adjusts to the temperature of the produced fluid (Langbauer et al., 2019). Figure 55 shows the results of the temperature for each of the 5 downhole dynamometer sensors over the entire recording time:

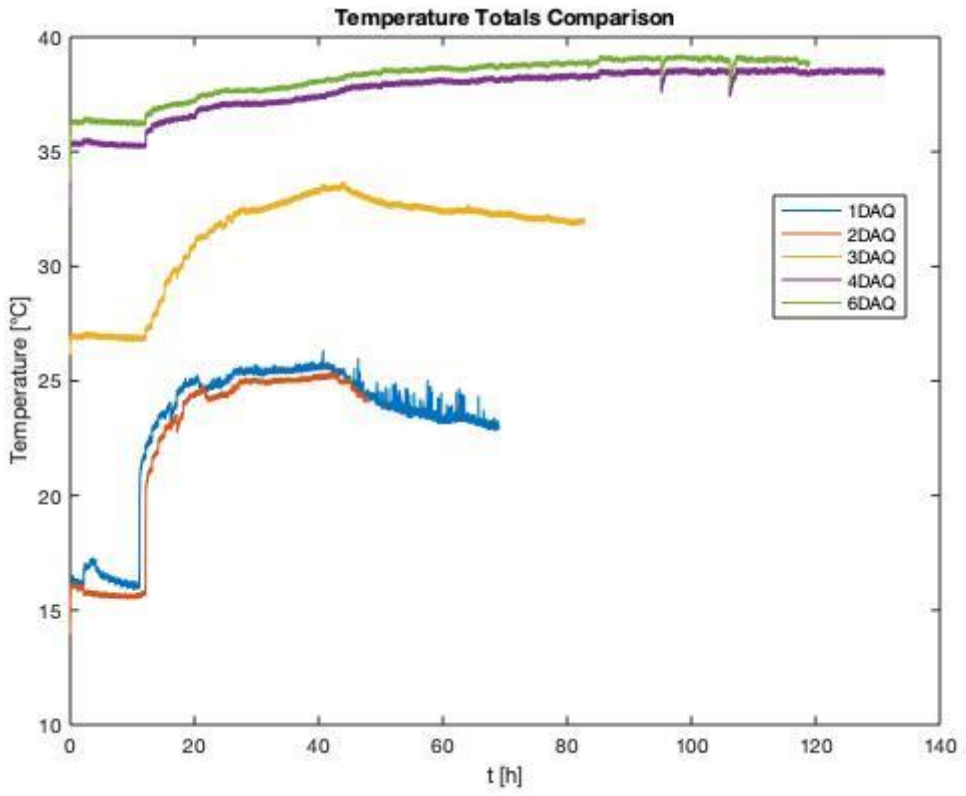


Figure 55: Comparison of temperature totals

In Figure 55, it can be observed, that after the installation of the 5 downhole dynamometer sensors, the pump was not activated until the 12th hour of recording. This leads to a lower temperature course in the beginning for all sensors followed by a considerable spike once the pump was turned on.

As long as the pump remained off, the temperature value was just about constant, except for sensor number one (DDS1), which, as mentioned before, was located just below the polished rod and near the top of the rod string. This initial spike in temperature, which can be seen on the bottom left in Figure 55, can be explained by an influence of the sun, since the weather during the field example was quite hot (Langbauer et al., 2019).

Once the pump started producing and lifting warm fluid to the surface, a temperature increase of around 10° C from 16° C to 26° C happens for DDS1 and DDS2, which are located at a measured depth of 7 and 15 meters, respectively. Interestingly, the temperature of DDS1 is a little higher than the one of DDS2, even though it is placed at a shallower depth. This is caused by friction of the stuffing box, which subsequently heats the polished rod that is directly linked to sensor number one (Langbauer et al., 2019).

Downhole dynamometer sensor 3 also shows a significant spike once the pump went into production and overall, the temperature was more than 10° C higher than DDS1 and DDS2. As expected though, the temperature for each sensor depends strongly on the depth of installation, meaning that a deeper installation depth leads to a higher the initial temperature.

Results for DDS 4 and DDS 6 show a temperature of above 35°C before the activation of the pump. Since the plunger of the pump is not installed at reservoir depth, there is also a small temperature increase of around 3° C.

Back in Figure 55, another notable development begins around the recording hour 40, when the temperatures for downhole dynamometer sensors 1, 2 and 3 gradually commences to decline, while values for DDS4 and DDS6 still continue to slightly increase. The reason for this behavior is a falling liquid level in the annulus combined with the production of reservoir fluid, which has a lower heat capacity than the pure workover fluid (Langbauer et al., 2019).

In the end the liquid reaches the surface with a temperature slightly above 22° C. The maximum temperature reached at DDS6 was 39° C, meaning the liquid lost around 17° C during the lifting period.

6.4.2 Battery Voltage Analysis

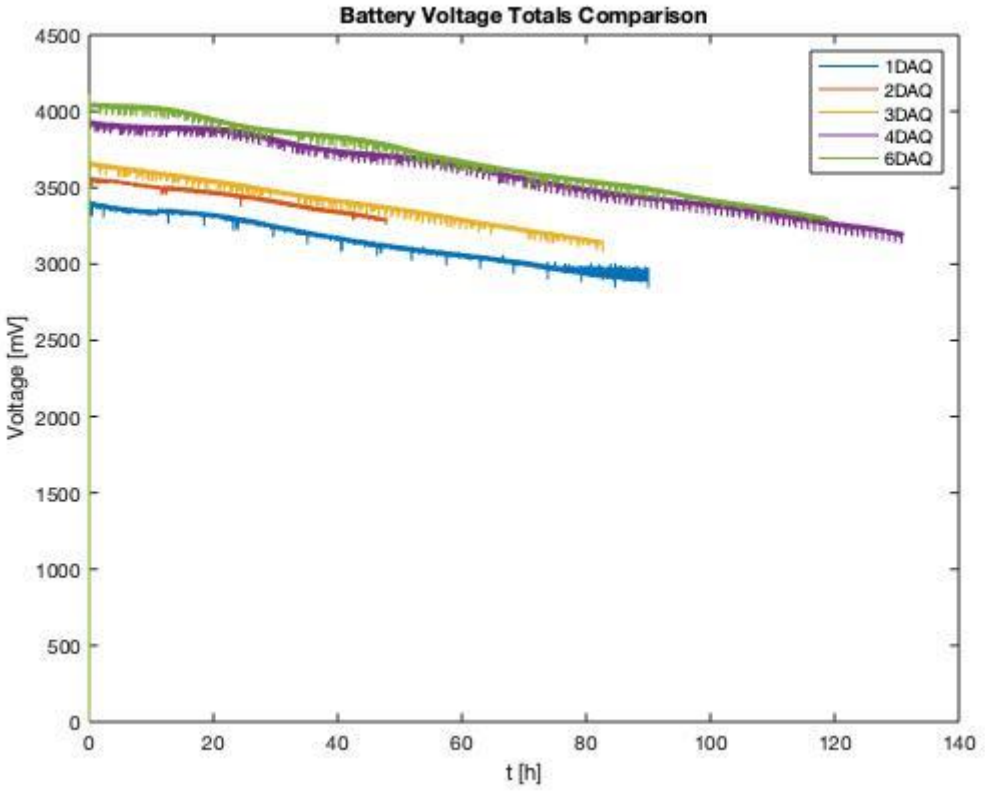


Figure 56: Battery voltage comparison over total recording period

Figure 56 shows the behavior of the battery voltage of each of the 5 downhole dynamometer sensors over their entire recording period.

DDS1 starts with an initial value of around 3400mV, while DDS2 and 3 both have starting values over 3500mV. Sensors number 4 and 6 begin at or slightly beneath 4000mV.

As expected, the values for the battery voltage gradually decline for each sensor over time. Though, it is interesting to see, that even after 131 hours of recording, sensor number 4 still has a battery voltage of around 3500mV.

Overall, the results of the battery voltage indicate, that the recording times for future field test could be expanded and/or be done consecutively instead of taking them in 1-hour intervals.

6.4.3 Acceleration Analysis

Comparison plots of the acceleration in x-, y- and z-direction for each of the 5 downhole dynamometer sensors over the entire recording period, as seen in Figures 57, 59 and 62, show quite convoluted results.

Still, it is possible to derive some information from each of the plots and if more details are desired, it is always possible to have a look at the total acceleration plots for each of the sensors individually.

In all 3 comparison plots, it can be observed again, that the pump was not activated until the 12th recording hour. Afterwards a spike in acceleration happens for all 5 sensors. While there is quite a lot of data outliers and possible noise, especially in x-direction, a certain symmetry may be recognized for each of the sensors and in all 3 directions.

The positioning of the sensors must also be considered, as sensors 4 and 6 are installed in the deviated part of the rod string and therefore their x-, y- and z-direction slightly differ from the other 3 sensors.

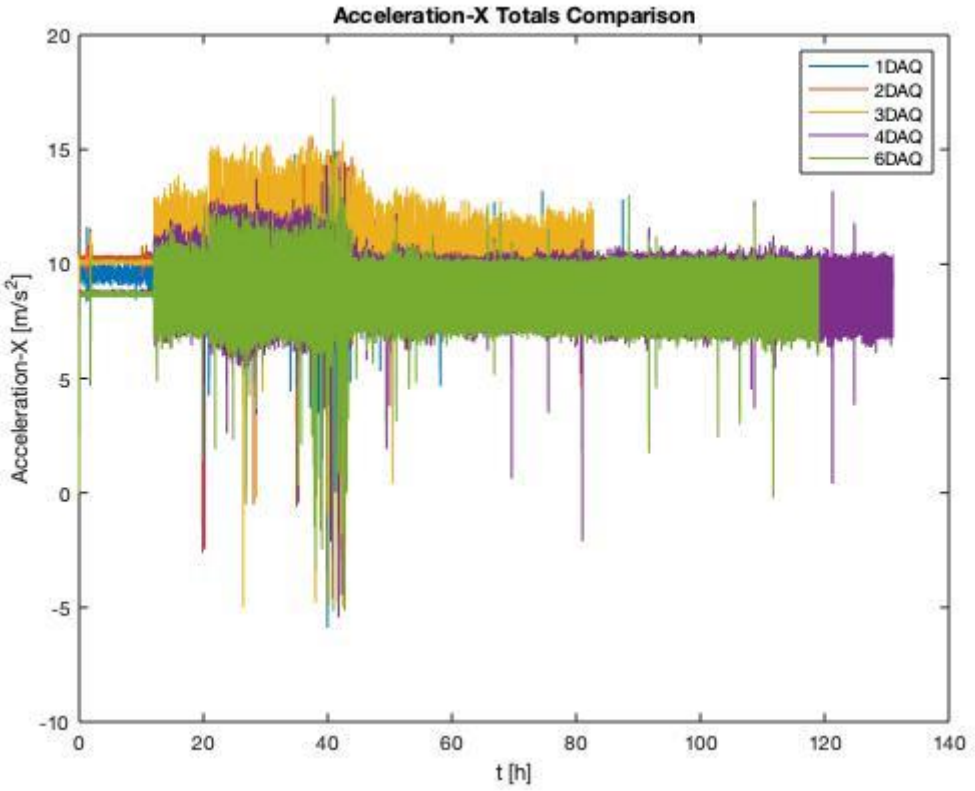


Figure 57: Acceleration in x-direction comparison over total recording period

In x-direction the acceleration values, without the aforementioned data outliers, lie between around 5 to 15m/s^2. DDS3 produces the highest value, while all other sensors stay below 13m/s^2.

As an example of an individual total acceleration plot, the results in x-direction for sensor number 6 are shown in Figure 58.

From the start of the pump until around recording hour 45, the values hover between 6 and 13m/s^2. Afterwards, more symmetric results between 8 and 11m/s^2 can be observed. This small downtick in acceleration could have been caused by the gradual decline in strokes per minute that was implemented over the course of the field test, which can be seen in the frequency analysis as well.

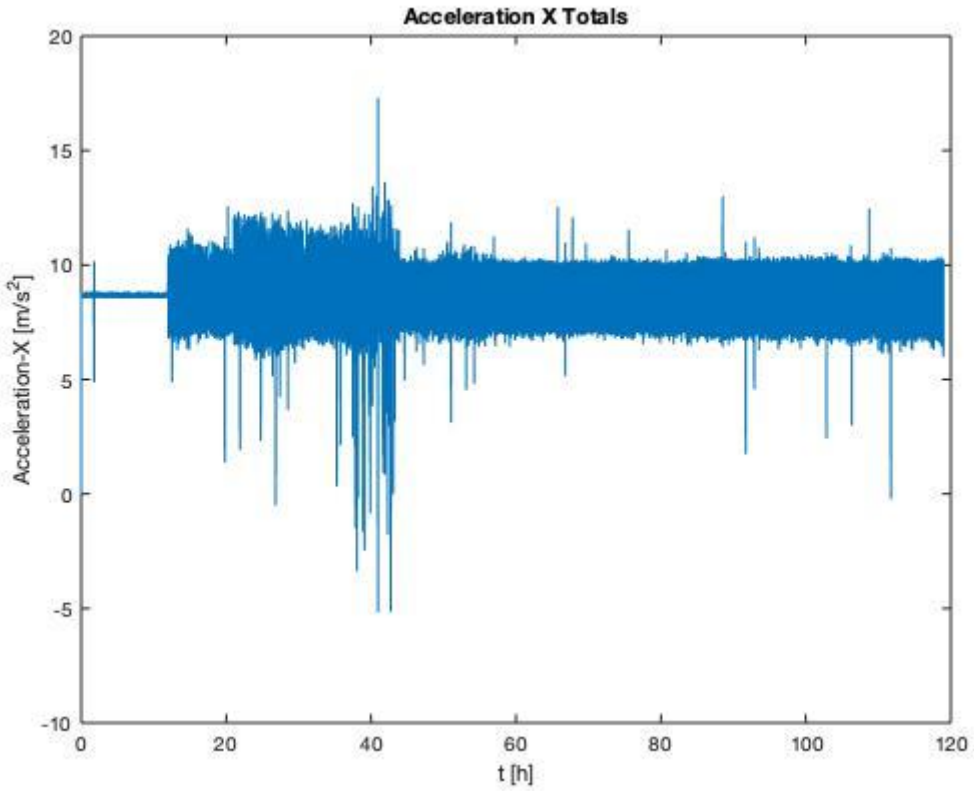


Figure 58: DDS6 acceleration in x-direction over total recording period

In y-direction, DDS3 displays the widest spread of results as the acceleration values lie between -12m/s^2 and +11m/s^2.

DDS1, which can be seen in detail in Figure 60, demonstrates two short term spikes. 1 from the activation from the pump until recording hour 20 and then again from recording hour 35 to 50. Otherwise the values linger quite constantly between 0 and -1m/s^2.

DDS6, which can be seen in Figure 61, present a similar spike from hour 12 to hour 20, but as shown in Figure 59, the range of acceleration values in y-direction is much broader compared to sensor number 1. This could be explained by the positioning differences between the 2 sensors, as sensor 1 is installed vertically below the polished rod and sensor 6 is placed on top of the plunger in the deviated section of the rod string.

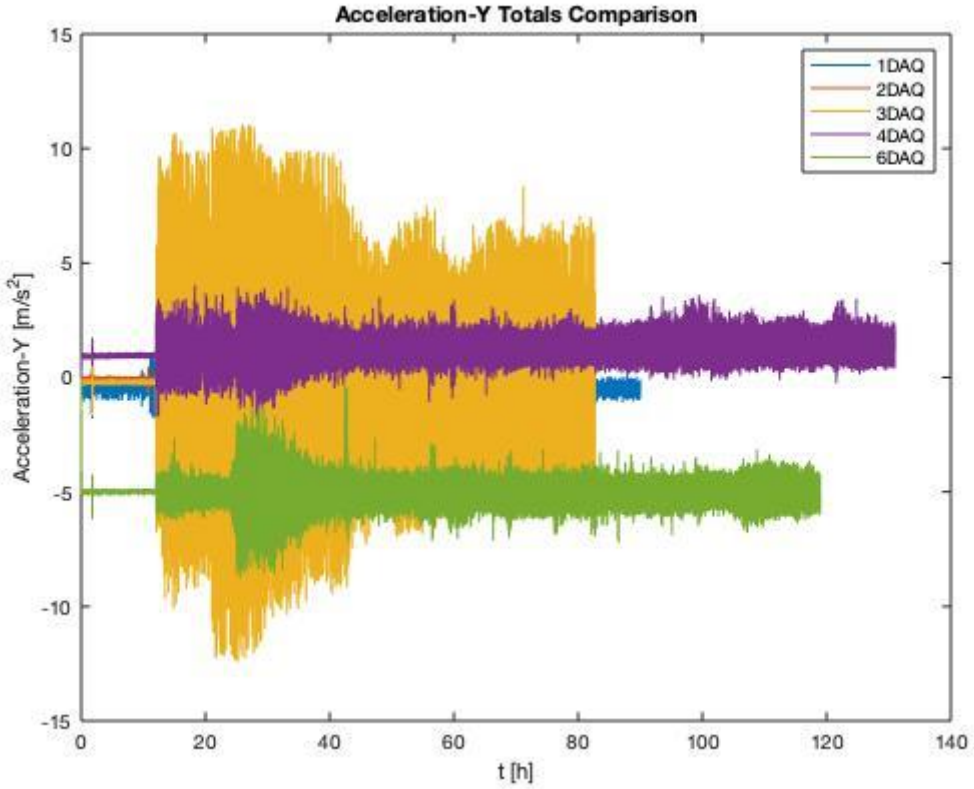


Figure 59: Acceleration in y-direction comparison over total recording period

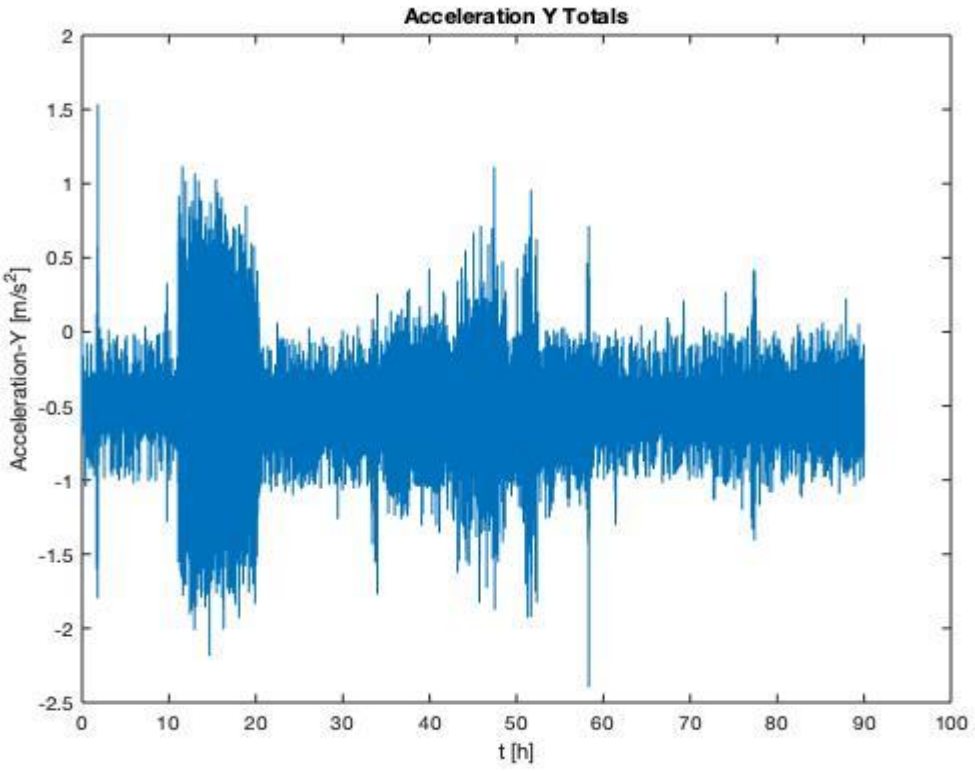


Figure 60: DDS1 acceleration in y-direction over total recording period

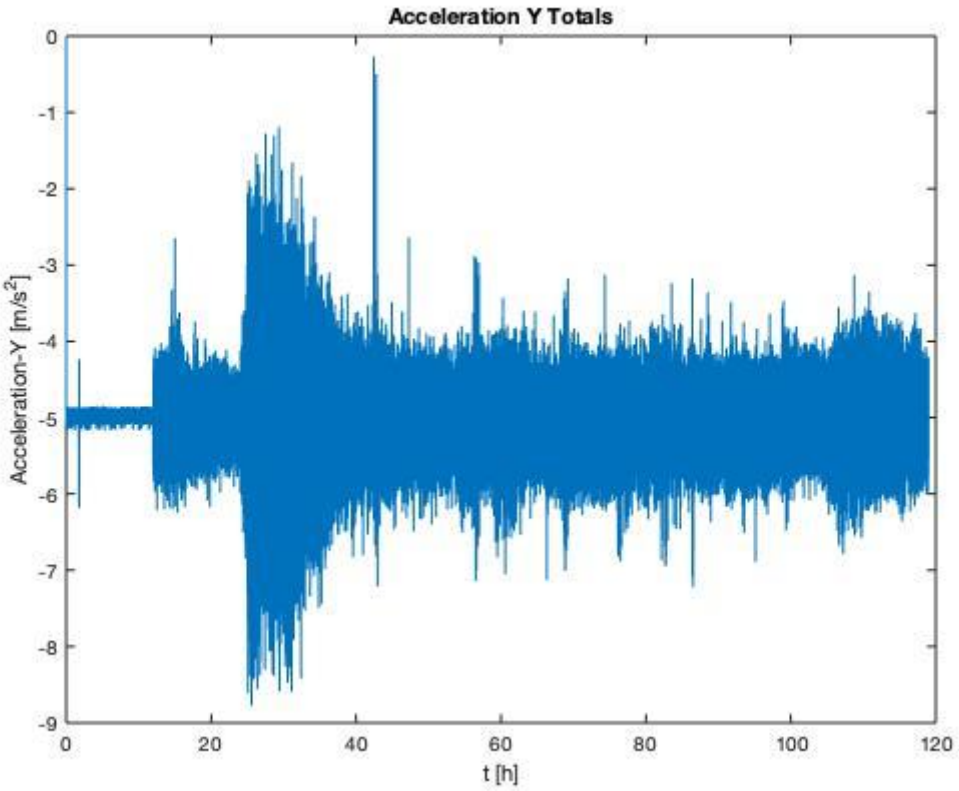


Figure 61: DDS6 acceleration in y-direction over total recording period

In z-direction the comparison plot reveals, that DDS4 reaches a maximum level of around 9m/s^2, while DDS6, after an initial spike between hours 20 and 30, hovers between -1 and +2m/s^2.

Once again, sensor number 3, which is shown in Figure 63, portrays the broadest range as the acceleration values lie between -7 and +7m/s^2. A reasoning for this behavior may again be found in the positioning of the sensors since DDS3 is placed right at the kick-off point, where it is subjected to a lot of motion between the vertical and deviated rod string sections.

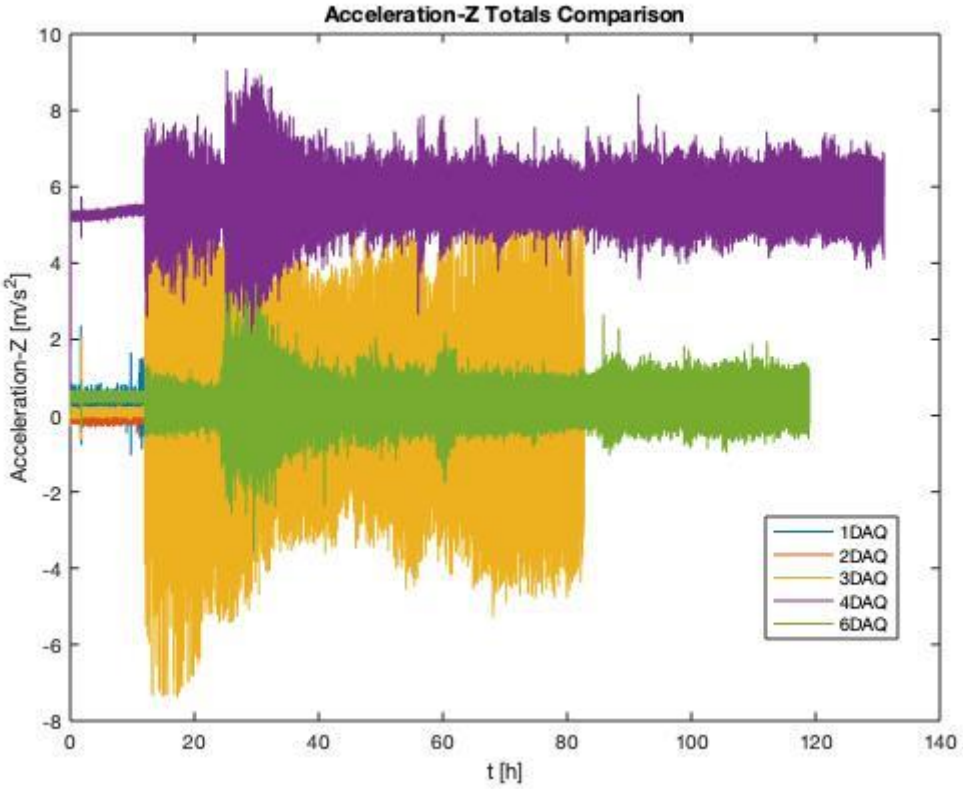


Figure 62: Acceleration in z-direction comparison over total recording period

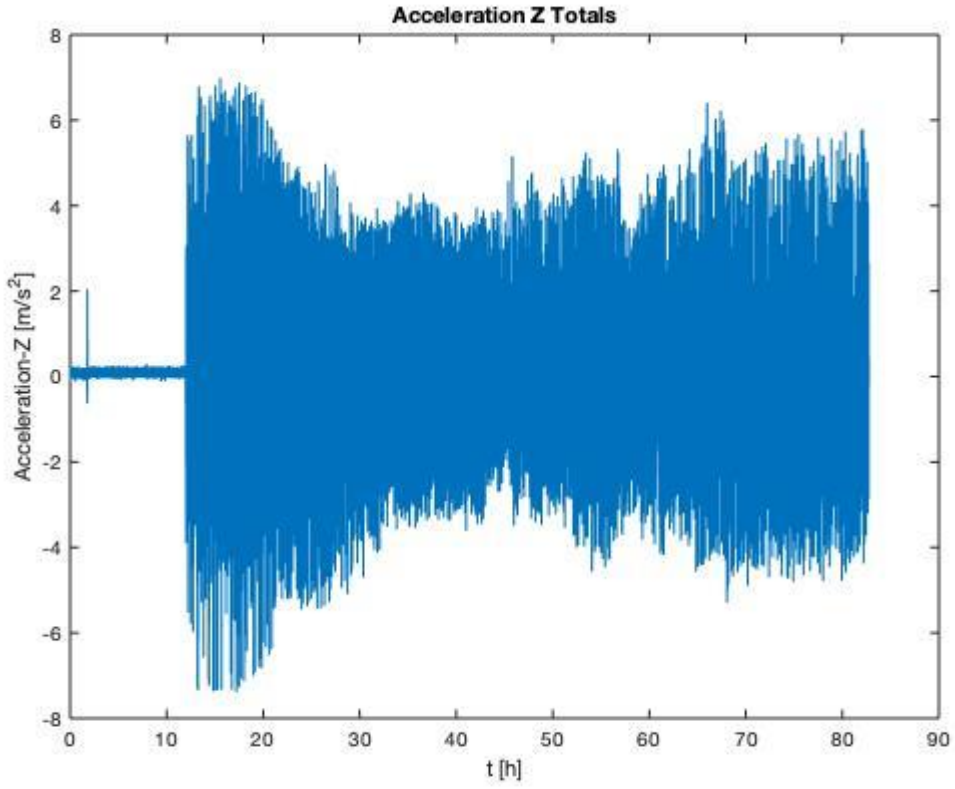


Figure 63: DDS3 acceleration in z-direction over total recording period

6.4.4 Motion Analysis

The results for the displacement in x-, y- and z-direction are used to analyse the motion and elasticity of the sucker rod string, which has a big influence on the system dynamics. In general, the stroke lengths reached at the plumber depend on (Langbauer et al., 2019):

- Overall composition of the rod string
- Depth of the system
- Strokes per minute

The importance of the displacement in x-direction, which is shown in Figure 64, is higher compared to the results in y- and z-direction, because the unit for the results in x-direction is meters, while the magnitude in y- and z-direction is 10^{-3} meters, which can be seen in Figures 65 and 66.

The highest values in y- and z-direction come from DDS4 and DDS6, which may be caused by their positioning in the deviated rod string section.

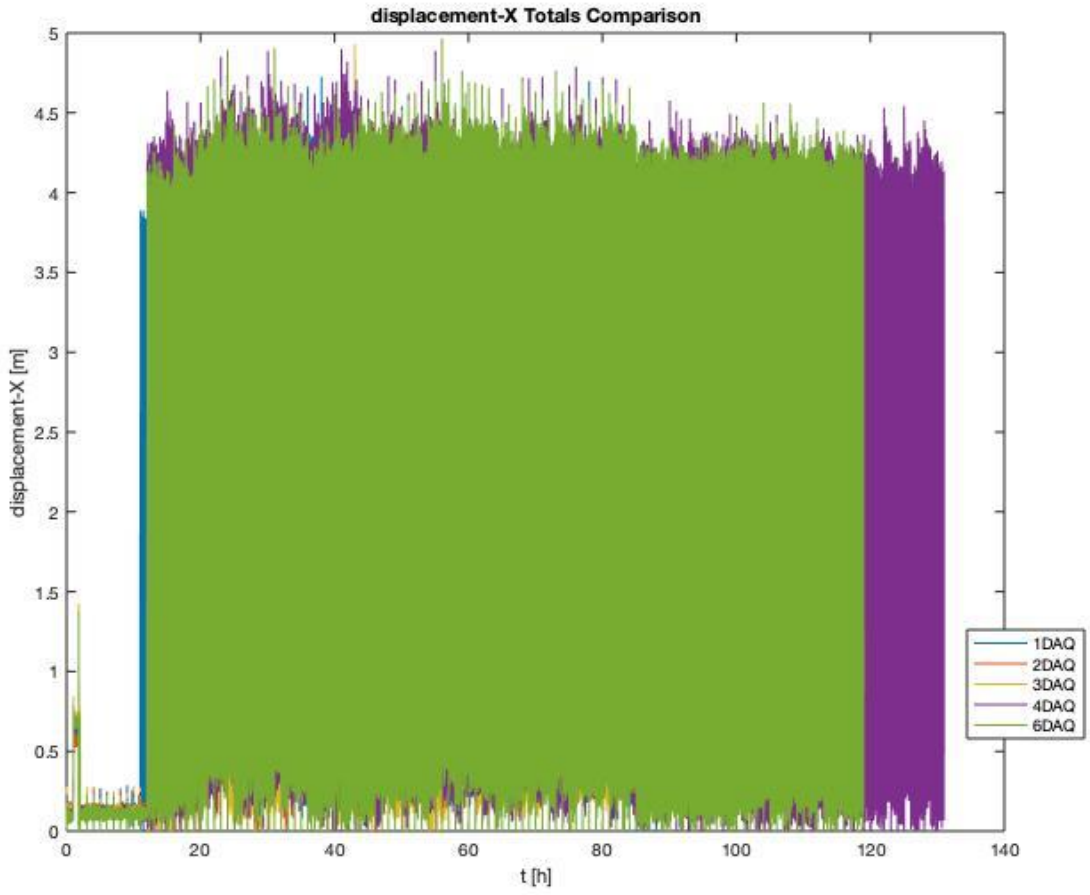


Figure 64: Comparison of displacement in x-direction over total recording period

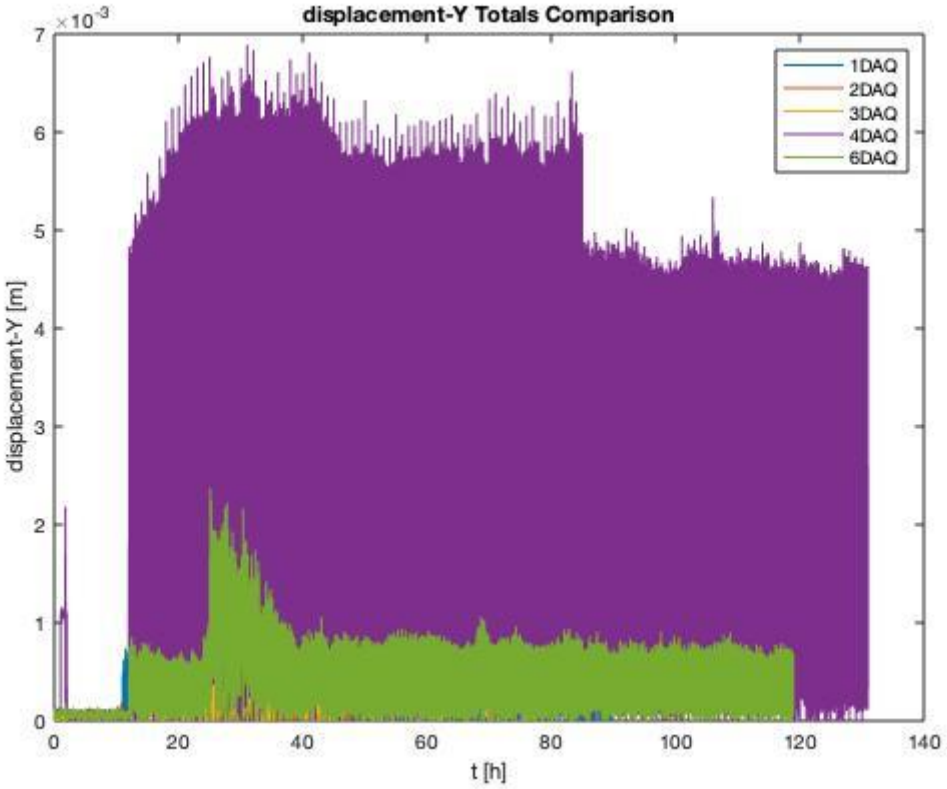


Figure 65: Comparison of displacement in y-direction over total recording period

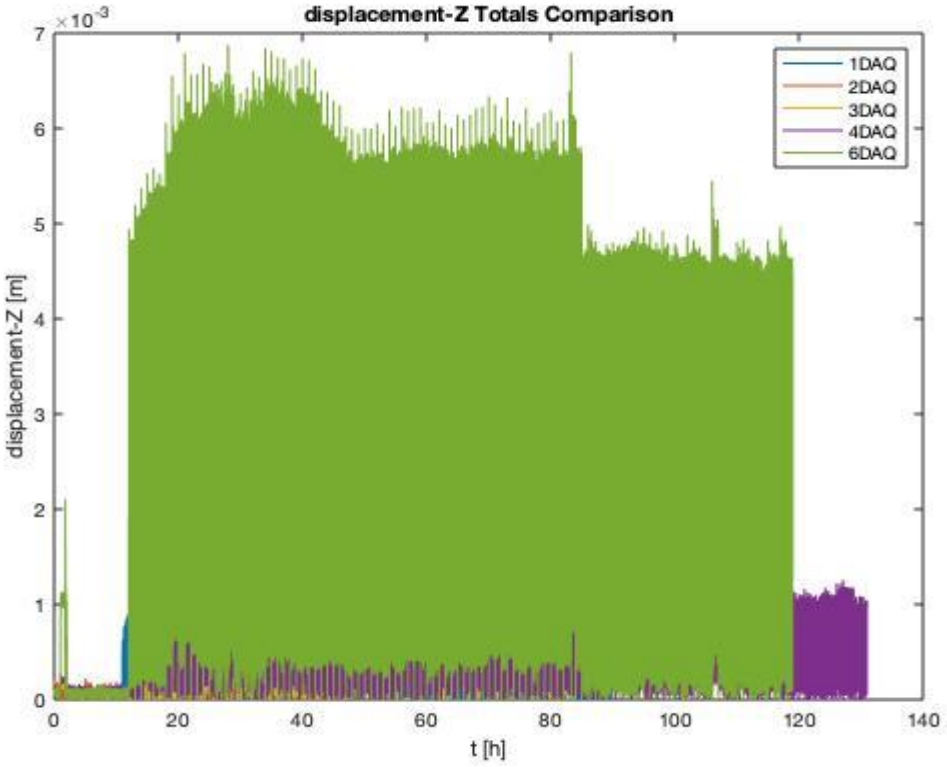


Figure 66: Comparison of displacement in z-direction over total recording period

Since the comparison plot in x-direction is quite convoluted, a more detailed look at the total results for each of the 5 downhole dynamometer sensors is necessary:

- DDS1

When disregarding the data outliers both on the bottom and the top of the diagram, the average value for displacement in x-direction is between 3,5 and 4 meters for sensor number 1. The spike in the beginning at recording hour 3 shows the installation of the horsehead.

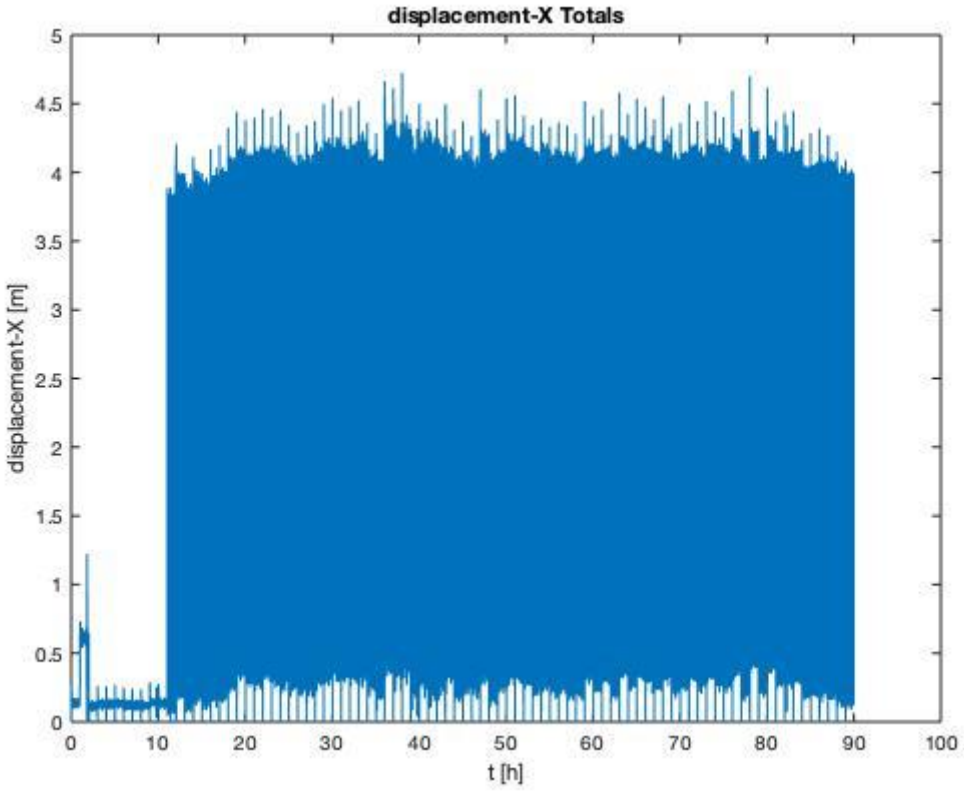


Figure 67: DDS1 displacement in x-direction over total recording period

- DDS2

Sensor number 2 is installed only 8 meters below sensor number 1 and as mentioned before, functions as a backup sensor for DDS1. Therefore, the total results for the displacement in x-directions should mirror the results of DDS1. In Figure 68, the same spike in the beginning of the recording for the horsehead installation can be observed and the average displacement value is also between 3,5 and 4 meters just like for downhole dynamometer sensor 1.

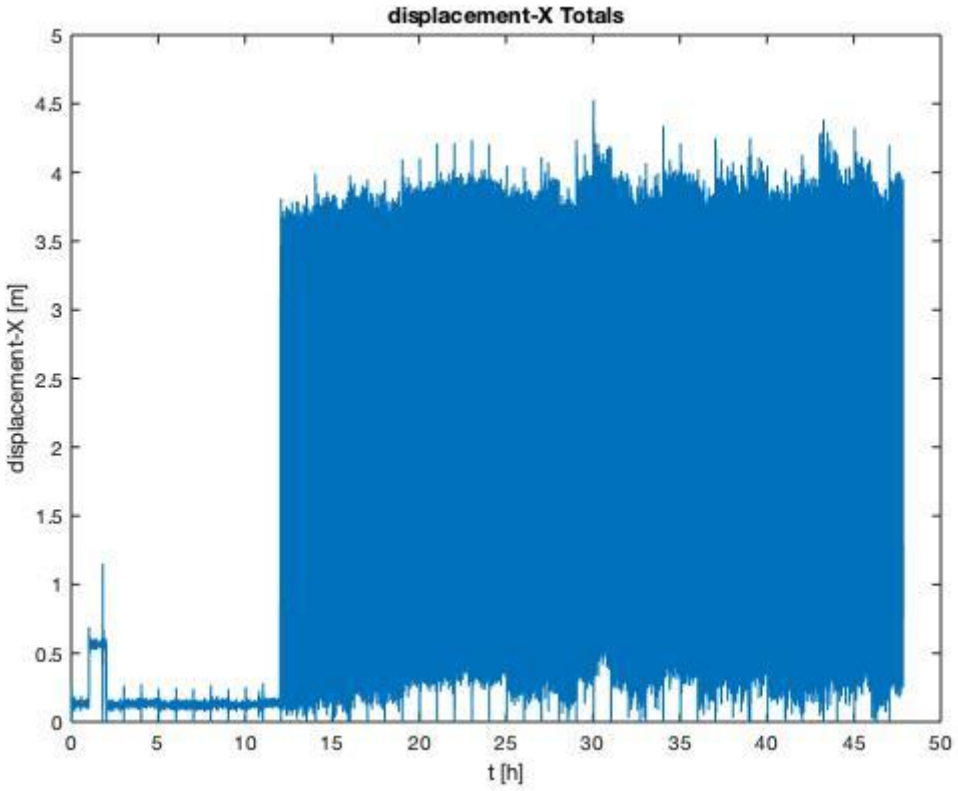


Figure 68: DDS2 displacement in x-direction over total recording period

- DDS3

An analysis of the total results for sensor number 3, which is shown in Figure 69, indicate a higher maximum displacement value as one data outlier almost reaches 5 meters. When neglecting the outliers again, the average value is closer to 4 meters compared to DDS1 and DDS2.

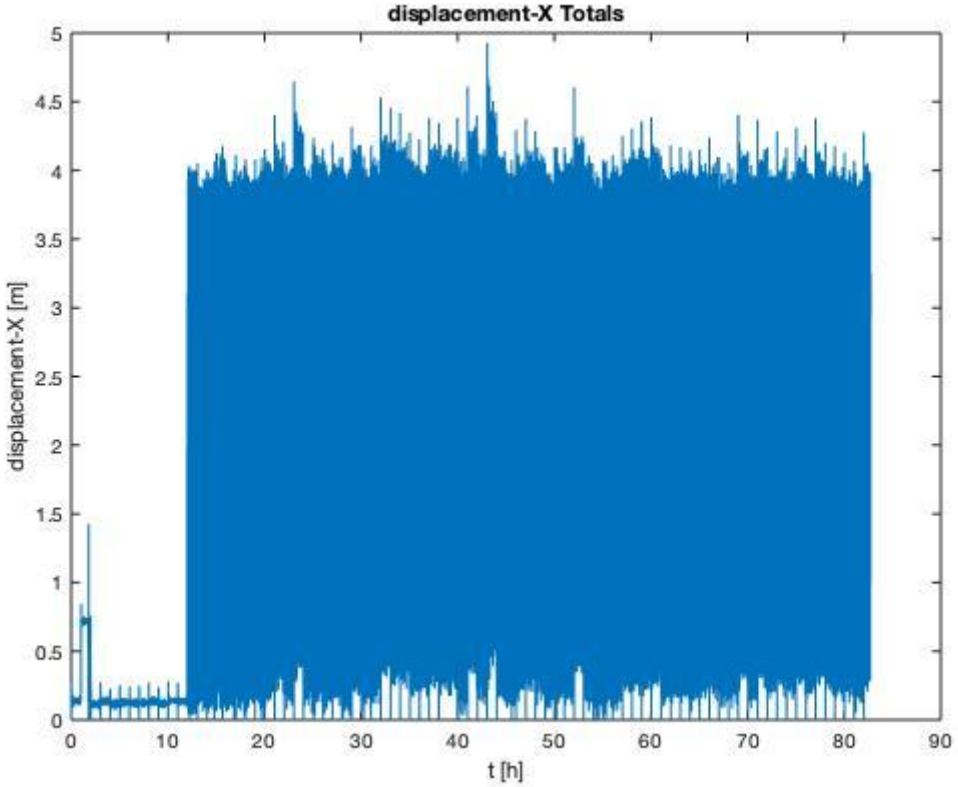


Figure 69: DDS3 displacement in x-direction over total recording period

- DDS4

The measurements of sensor number 4, which can be seen in Figure 70, show several data outliers that come close to 5 meters of displacement. The average value is also higher compared to the previous 3 sensors and lies between 4 and 4,5 meters.

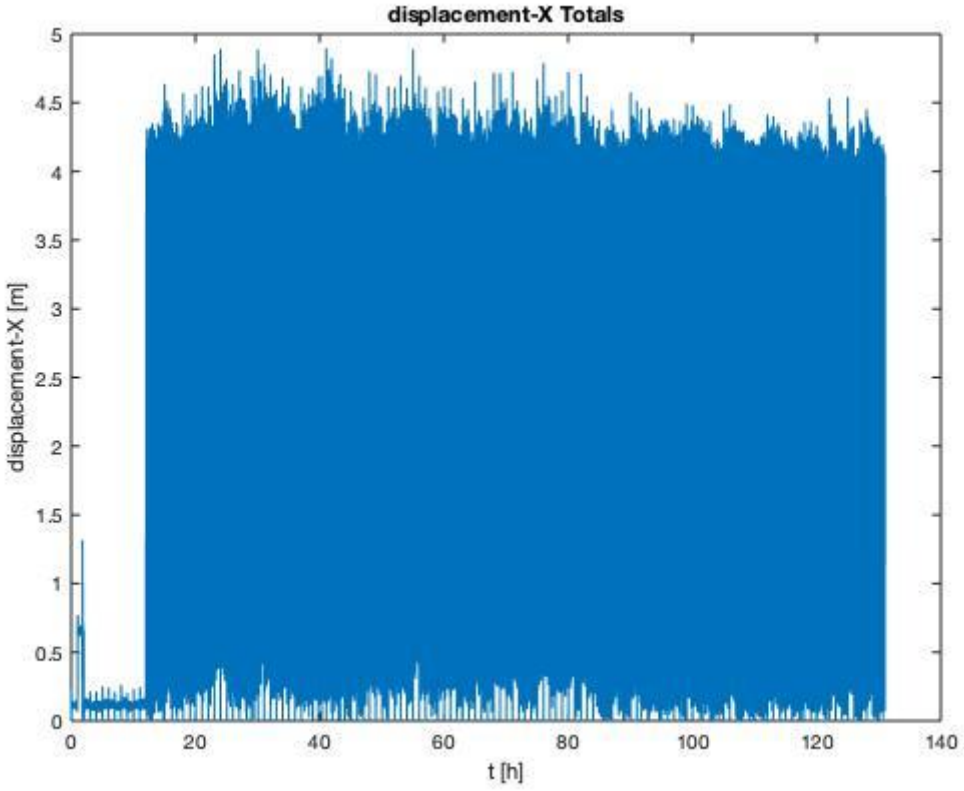


Figure 70: DDS4 displacement in x-direction over total recording period

- DDS6

The displacement plot in x-direction for sensor number 6, which is shown in Figure 71, once again shows a few data outliers close to 5 meters, but overall, the average value is closer to 4 meters again, especially after recording hour 80, when a slight dip in values can be observed.

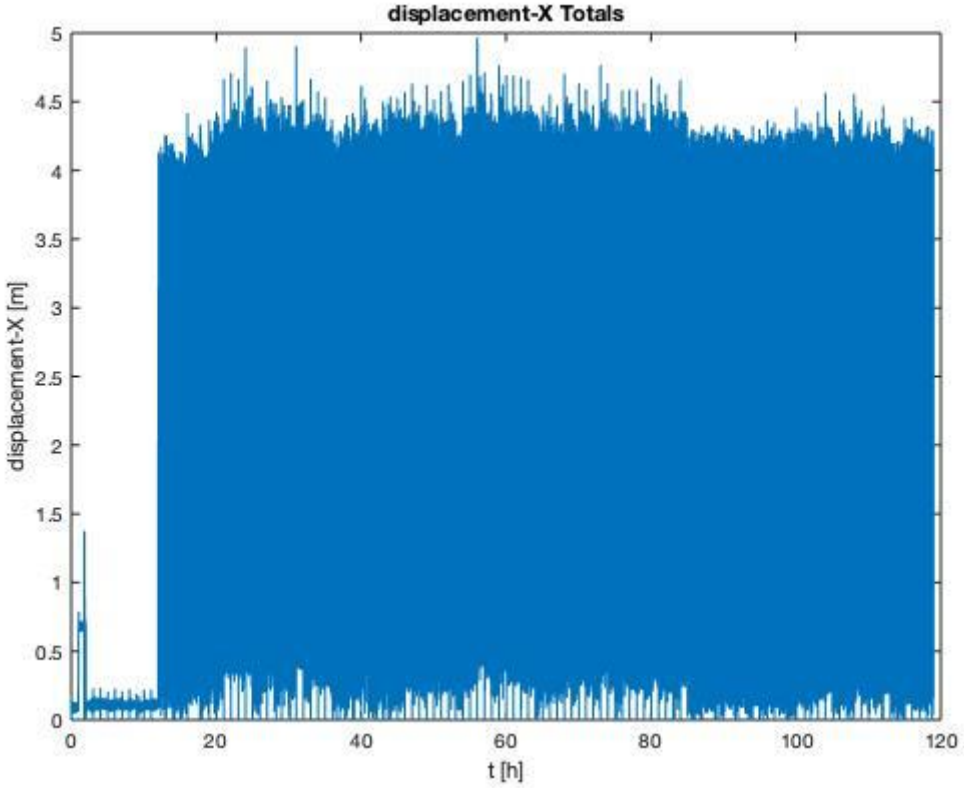


Figure 71: DDS6 displacement in x-direction over total recording period

Another interesting analysis can be derived, when zooming-in closely into the average displacement plots for an individual recording hour. For this example, the recording hour 40 was chosen at DDS2 and DDS6, which are shown in Figure 72 and 73.

A certain temporal shift in reaching the peaks, indicated by the straight red line, between the 2 sensors can be observed. On the one hand, this might be caused by inertia effects of the rod string and its highly dynamic behaviour (Langbauer et al., 2019). But on the other hand, a shift of over 5 seconds seems too large and, therefore, a small runtime error between the individual sensors should also be considered as the reason for this shift.

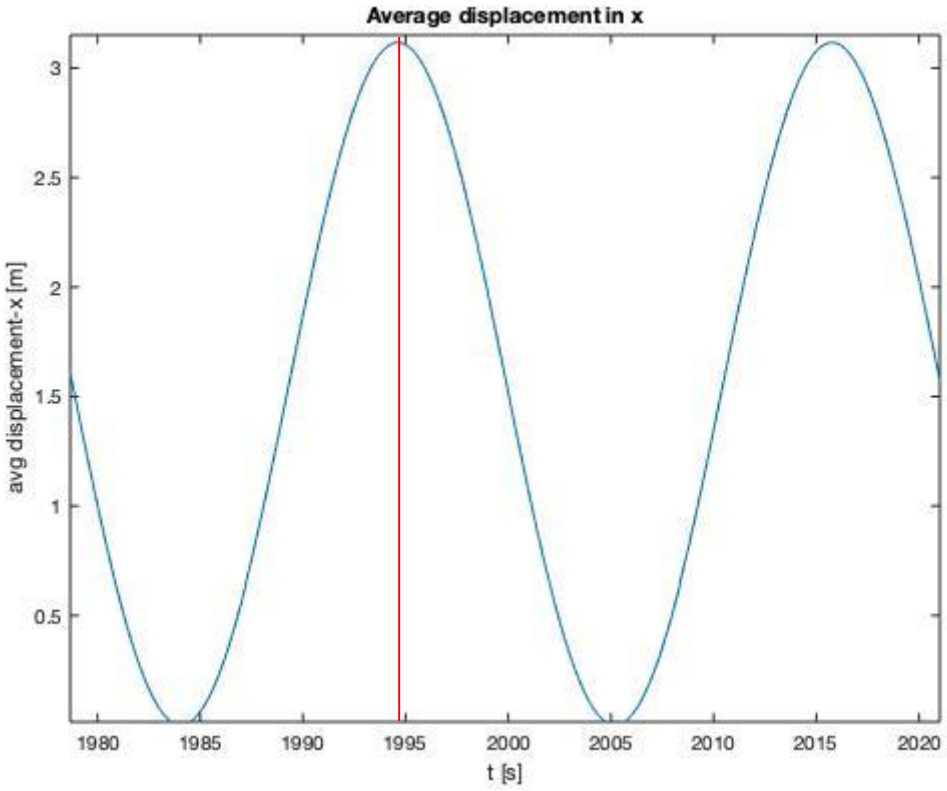


Figure 72: Zoomed-in plot of average displacement in x-direction for DDS2 at recording hour 40

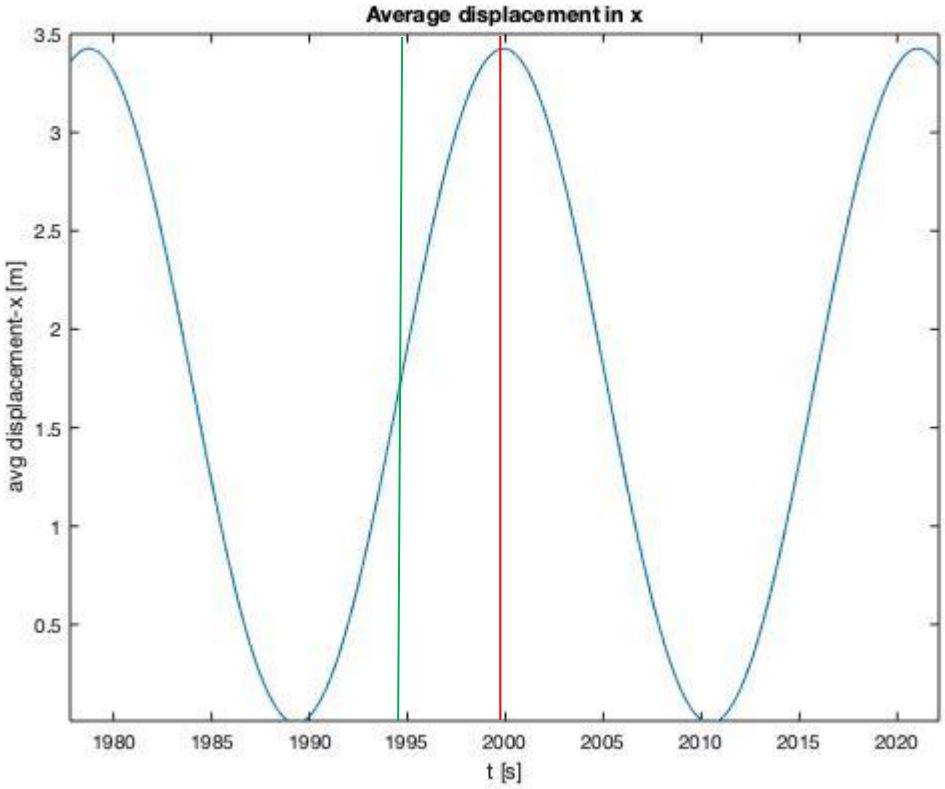


Figure 73: Zoomed-in plot of average displacement in x-direction for DDS6 at recording hour 40

6.4.5 Load Evaluation

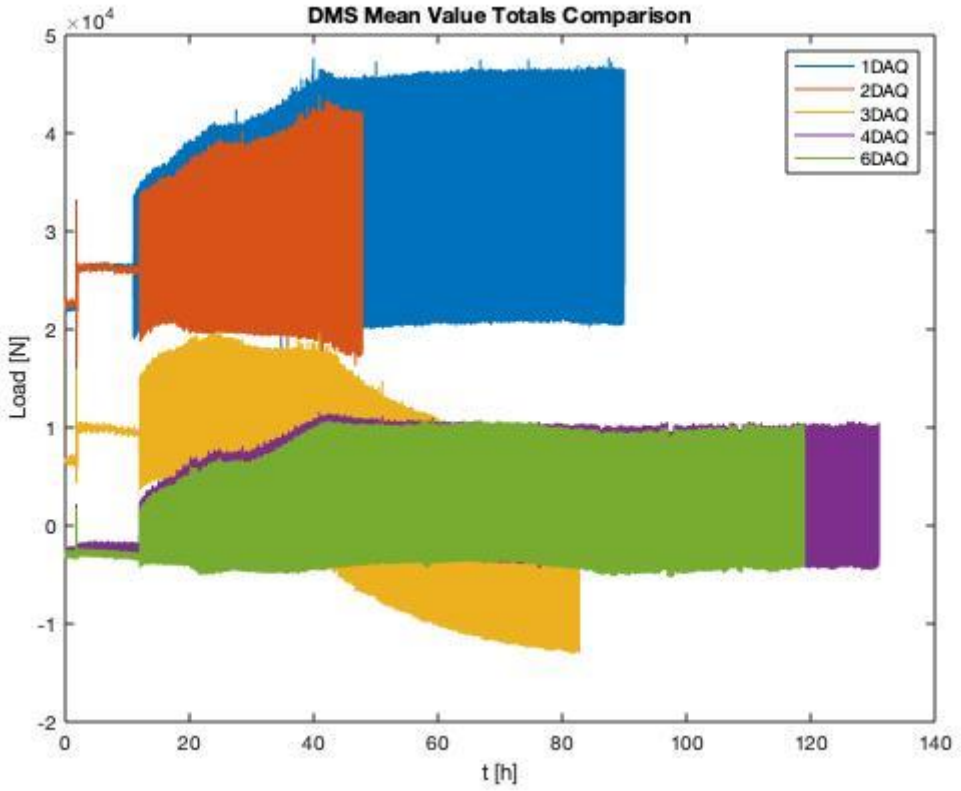


Figure 74: Comparison of strain gauges mean values over total recording period

Figure 74 presents the comparison plot of the strain gauges mean values for each of the 5 downhole dynamometer sensors. A maximum value of 45 kN can be recognized for DDS1 around recording hour 40, while the minimum value occurs at the end of the recording period of sensor number 3 and is around -12 kN.

A notable difference can be observed between the strain gauges mean values for sensors number 1 & 2 compared to sensors number 4 & 6, which could be explained by their installation depth.

When taking a more detailed look at a certain recording hour, in this case hour 40, a single up- and downstroke can be identified by zooming-in closely, which can then be used to make an evaluation about possible load fluctuations.

DDS1 & DDS2, which are shown in Figures 75 and 76, display some minor inconsistencies at the beginning of both the up- and the downstroke, which is the result of the low value of strokes per minute that was chosen for this field example (Langbauer et al., 2019).

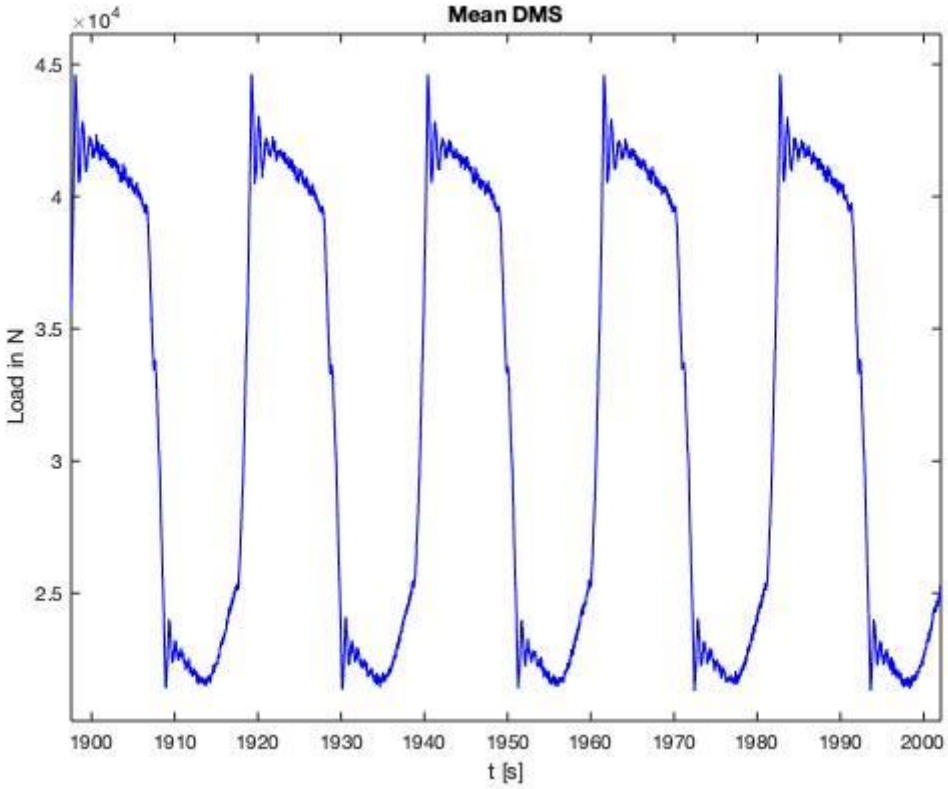


Figure 75: Zoomed-in plot of strain gauges mean value of DDS1 during recording hour 40

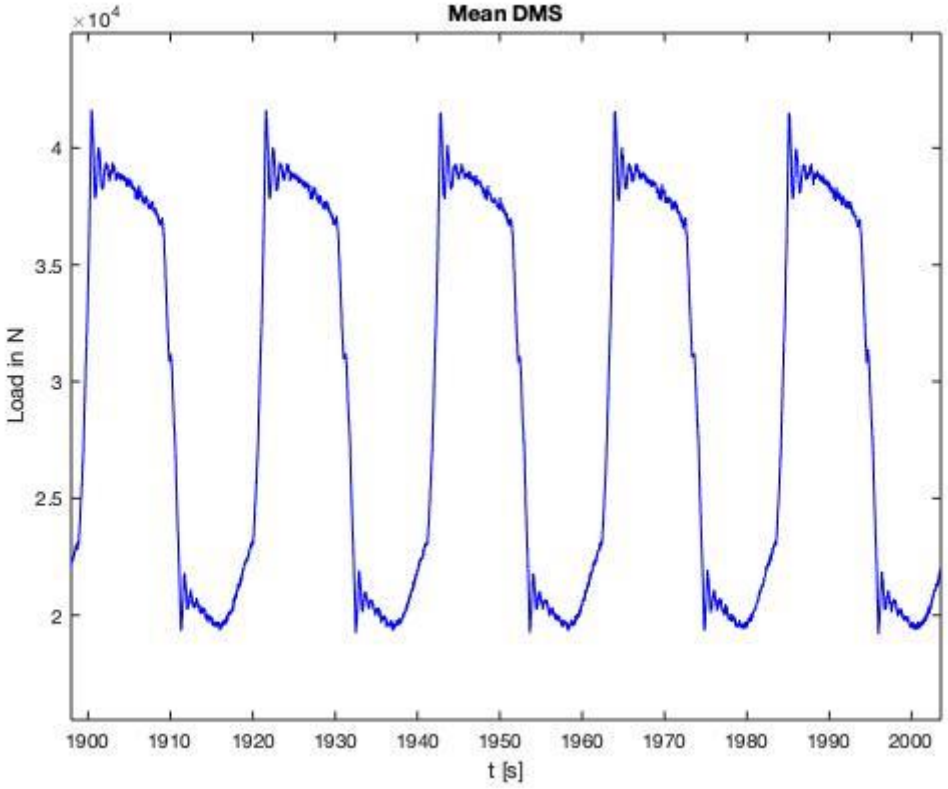


Figure 76: Zoomed-in plot of strain gauges mean value of DDS2 during recording hour 40

The results for DDS4 & DDS 6, which are shown in Figure 77 & 78, indicate a compressive load during the downstroke and concerns about possible buckling near the pump plunger may arise. Load fluctuations are also less significant compared to DDS1 & DDS2.

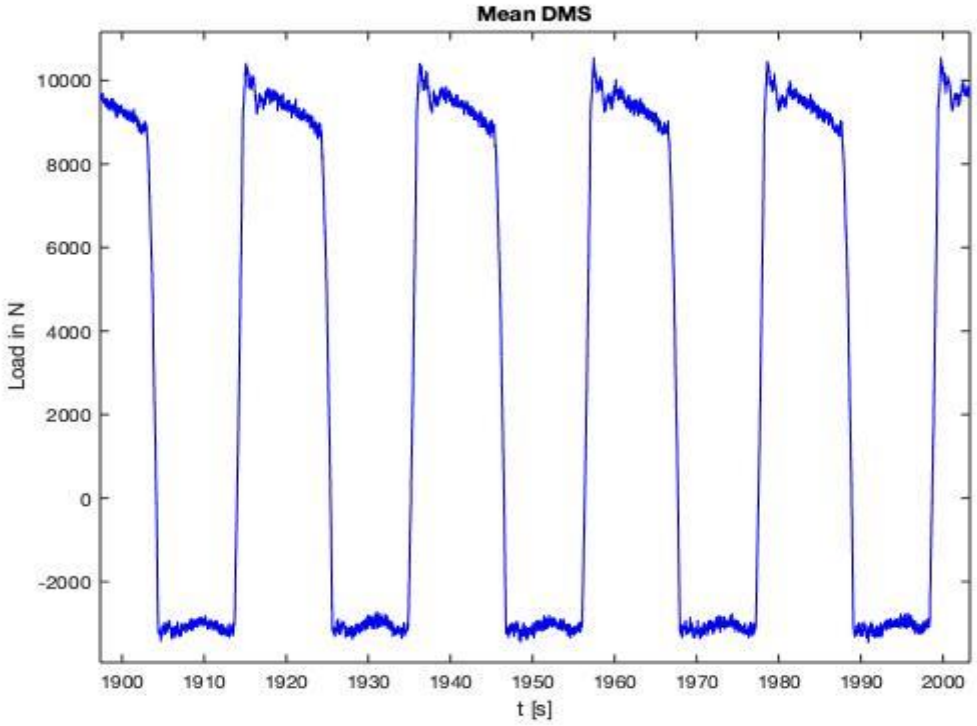


Figure 77: Zoomed-in plot of strain gauges mean value of DDS4 during recording hour 40

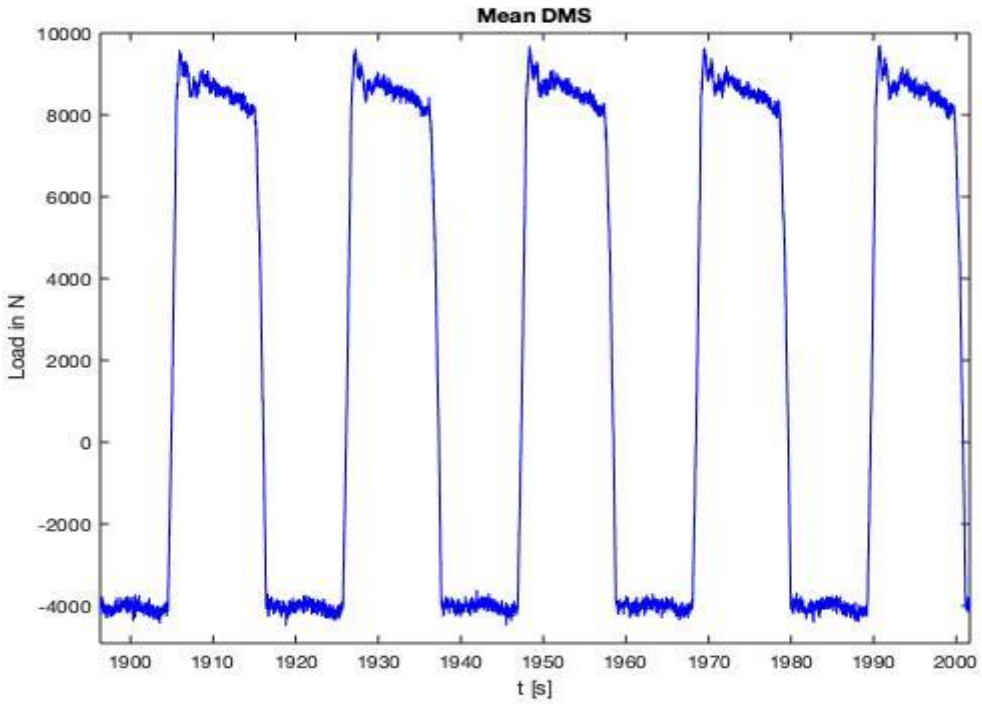


Figure 78: Zoomed-in plot of strain gauges mean value of DDS6 during recording hour 40

The results for DDS3 during recording hour 40, which can be seen in Figure 79, show some similarities to DDS1 and DDS2, since load fluctuations during up- and downstroke are clearly visible again.

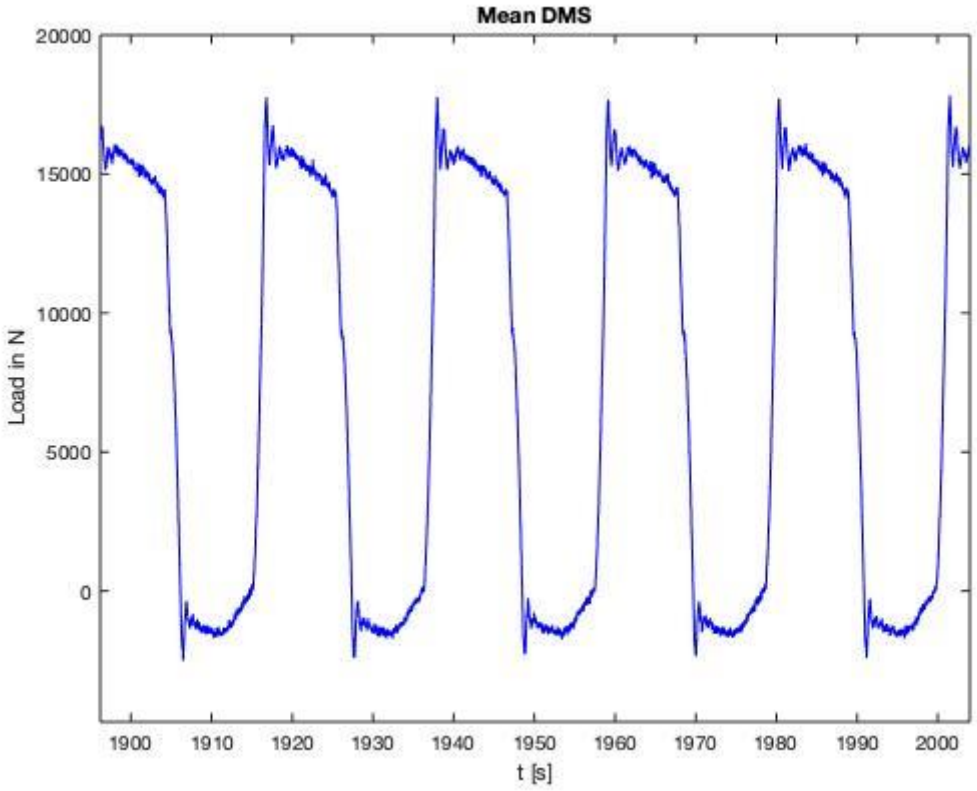


Figure 79: Zoomed-in plot of strain gauges mean value of DDS3 during recording hour 40

Furthermore, some interesting observations can be made when comparing the results of an individual sensor at different recording hours.

Figures 80, 81 & 82 present the strain gauges mean values for one stroke of DDS1 during the recording hours 30, 60 & 90. As time moves along, the peak loads as well as the load fluctuations appear to expand, which might be due to increased friction (Langbauer et al., 2019).

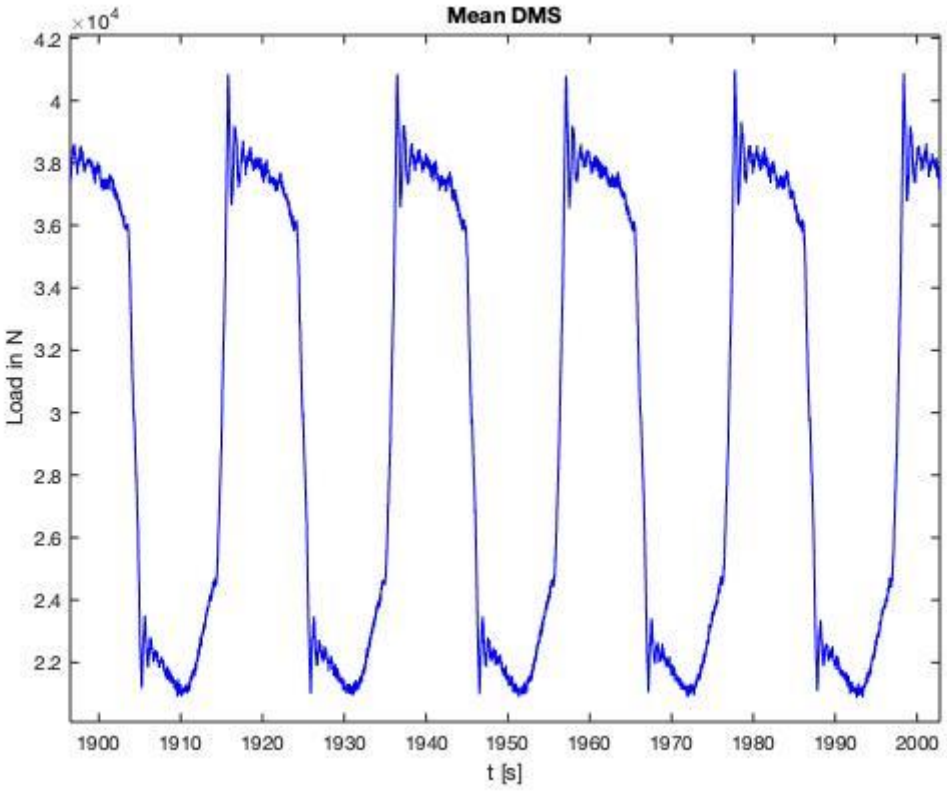


Figure 80: Zoomed-in plot of strain gauges mean value of DDS1 during recording hour 30

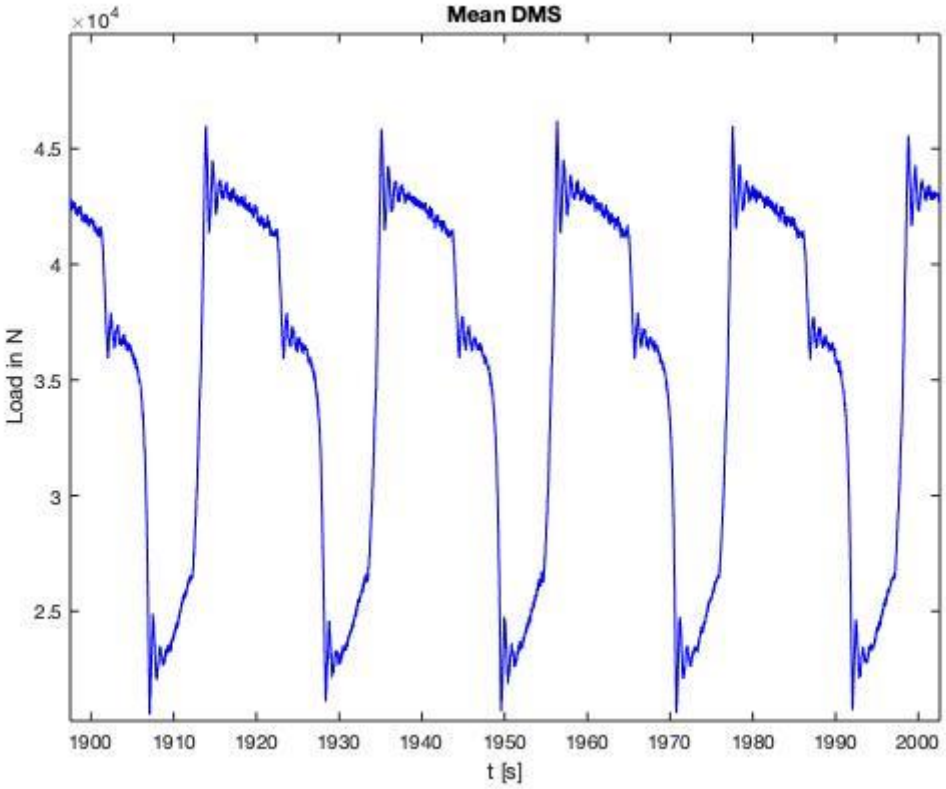


Figure 81: Zoomed-in plot of strain gauges mean value of DDS1 during recording hour 60

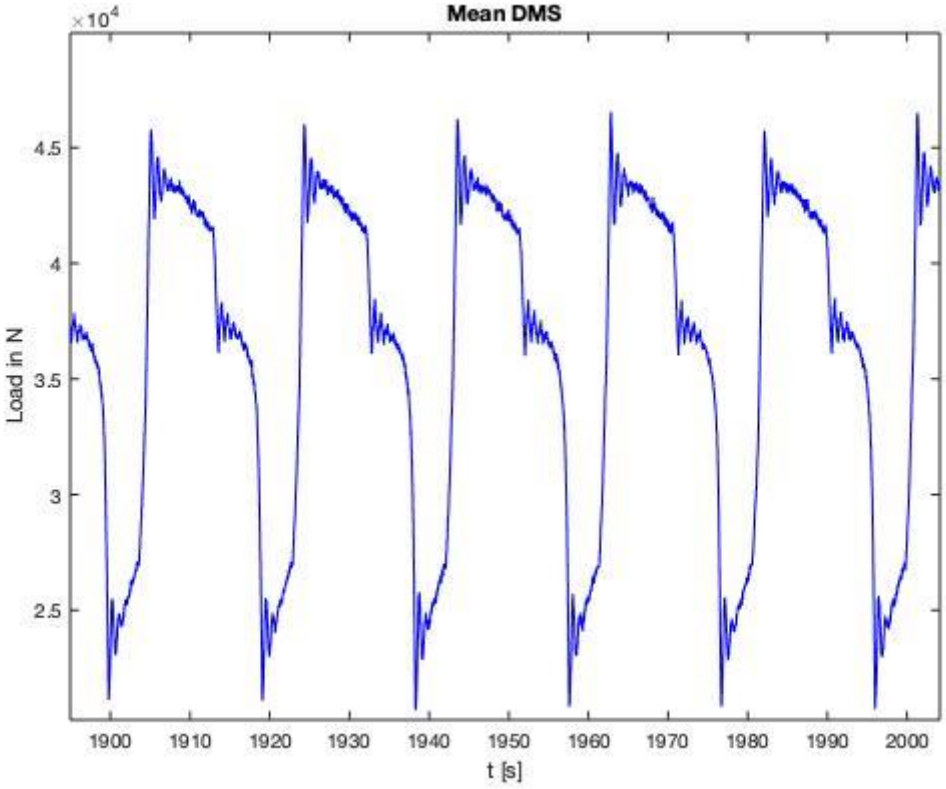


Figure 82: Zoomed-in plot of strain gauges mean value of DDS1 during recording hour 90

On the other hand, the values for DDS6 during the same recording hours, shown in Figures 83, 84 & 85, indicate a more constant load on top of the plunger, since the peak values stay between around 7.000 N and 10.000 N. Load fluctuations also do not appear to increase over time.

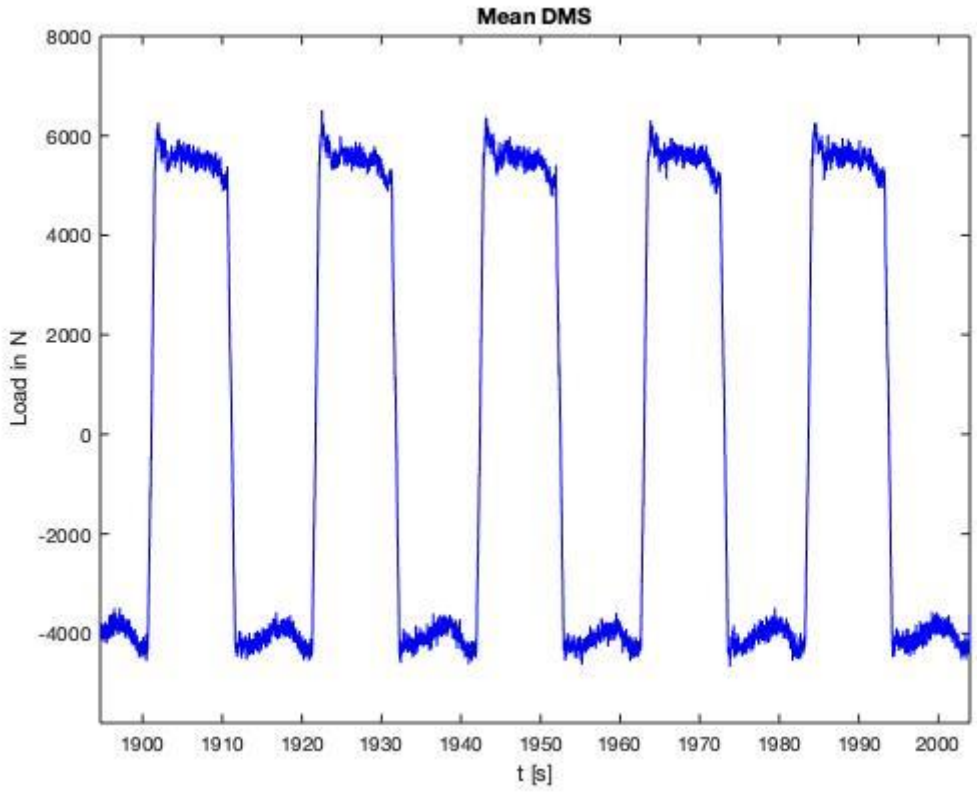


Figure 83: Zoomed-in plot of strain gauges mean value of DDS6 during recording hour 30

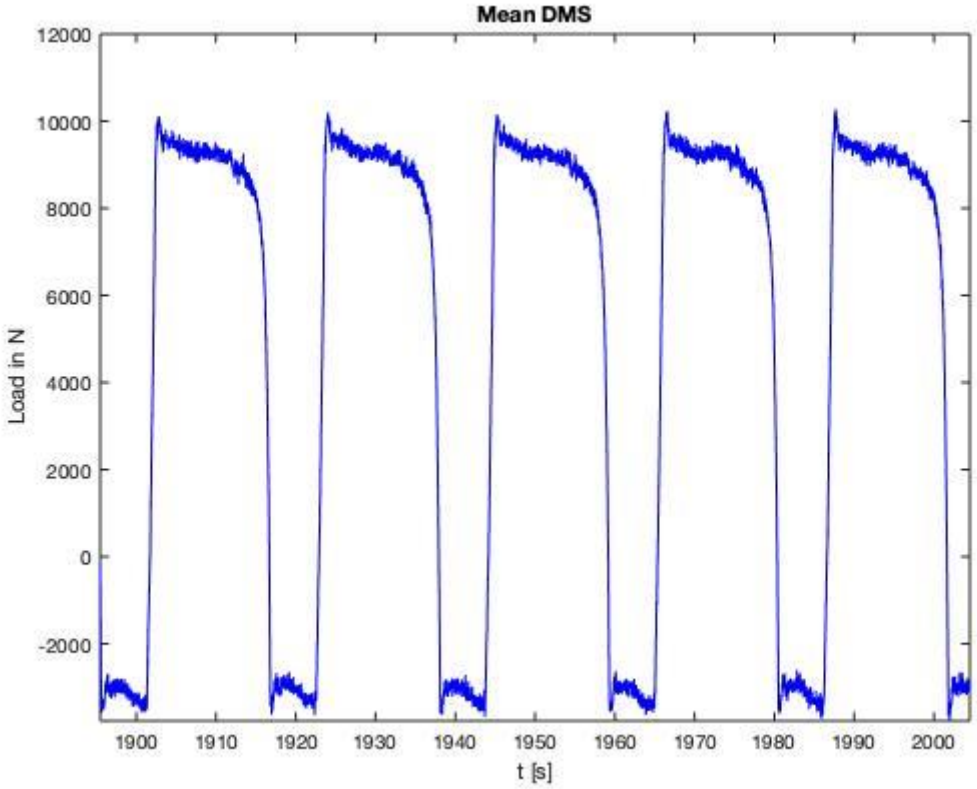


Figure 84: Zoomed-in plot of strain gauges mean value of DDS6 during recording hour 60

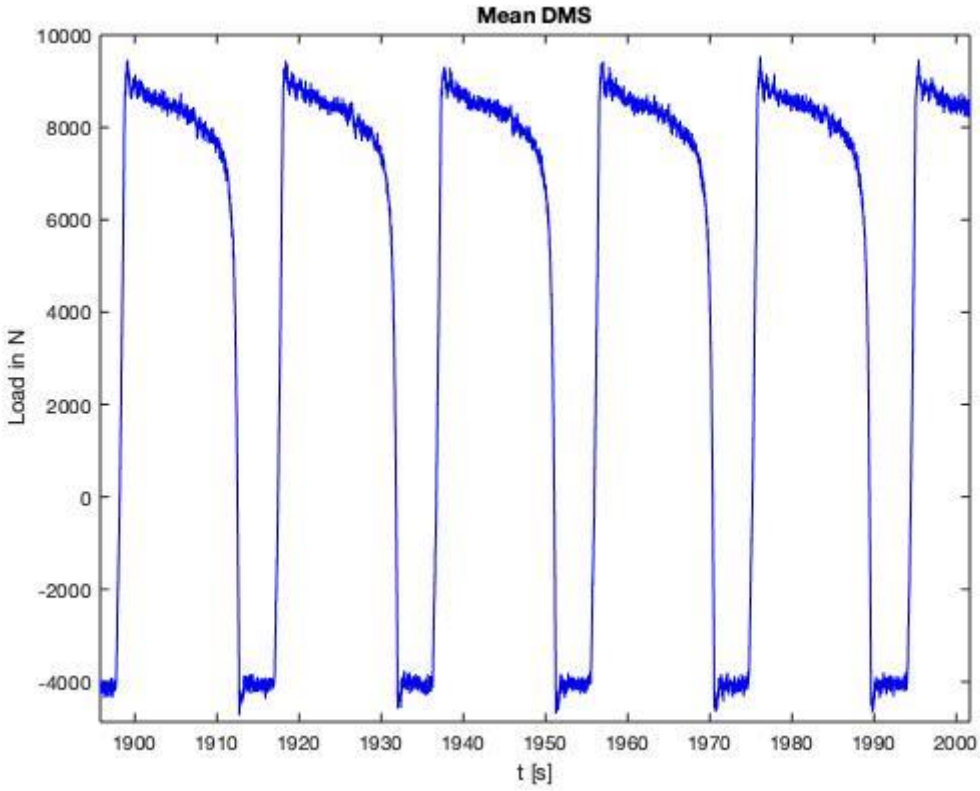


Figure 85: Zoomed-in plot of strain gauges mean value of DDS6 during recording hour 90

6.4.6 Resulting Dynamometer Cards

The results of the strain gauges mean values, along with the displacement values in x-direction, are then used to generate dynamometer cards, which are utilised for an evaluation of the pump efficiency and the load transfer.

In order to get a good overview of the behavior of the pump over the whole field-testing period, the load versus position diagrams of DDS1, DDS3 and DDS6 at recording hours 30, 60 and 80 will be compared to each other. The results of DDS1 represent the surface dynamometer cards, since the sensor is located at a depth of 7m, while the plots of DDS6, which is installed directly on top of the plunger, embody the downhole dynamometer cards. Sensor number 3 is located at the kick-off point at a depth of 480 meters. DDS2 and DDS4 are installed just a couple of meters below / above DDS1 and DDS6 and therefore their load versus position diagrams are very similar to the respective plots of sensor 1 and 6.

- Recording hour 30:

As one might expect, the shapes of the dynamometer cards change with the increasing installation depth of the sensors. The individual dynamometer cards, which are shown in Figures 86, 87 & 88, also indicate that at this point during the field test, the pump is operating under full conditions as workover fluid is still being removed.

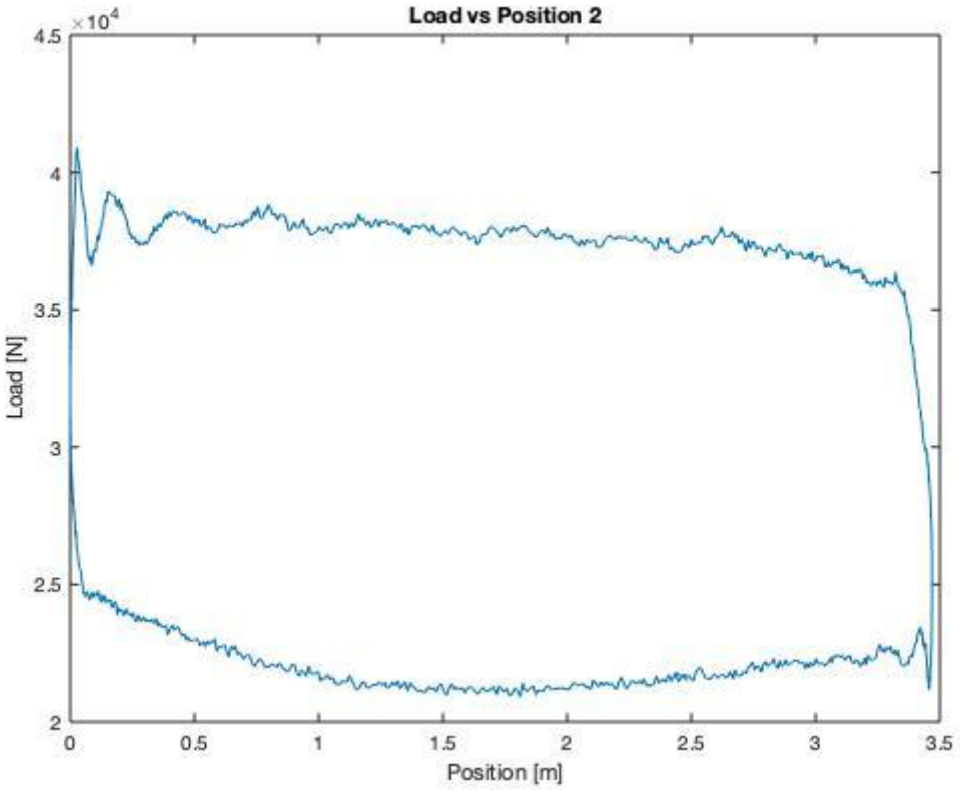


Figure 86: Load vs. position diagram of DDS1 at recording hour 30

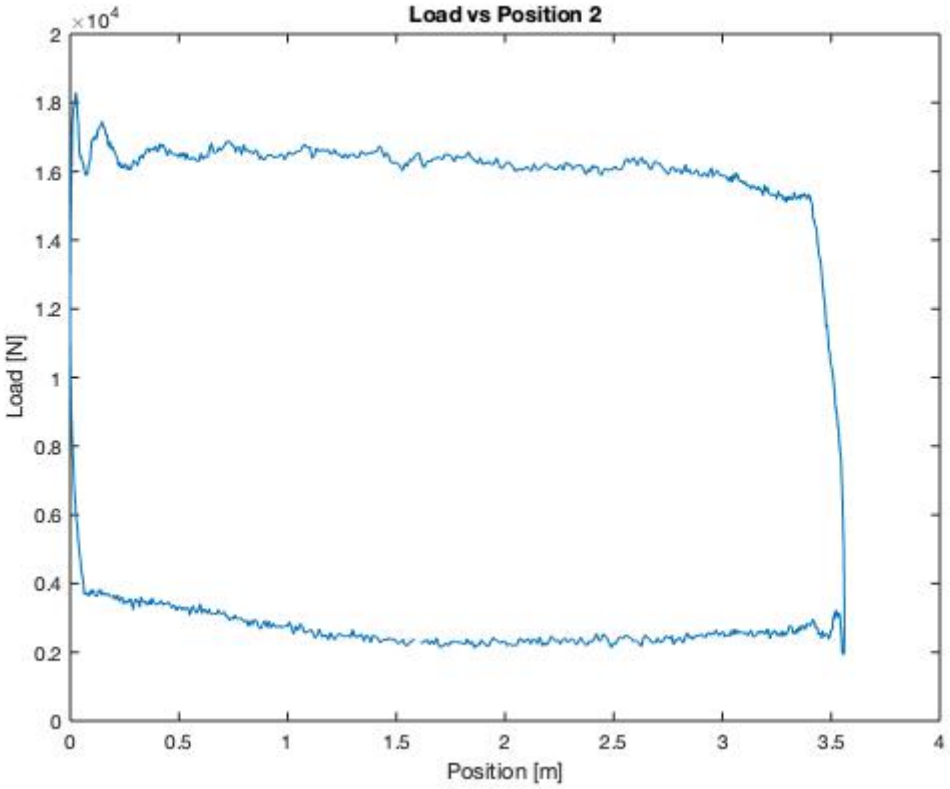


Figure 87: Load vs. position diagram of DDS3 at recording hour 30

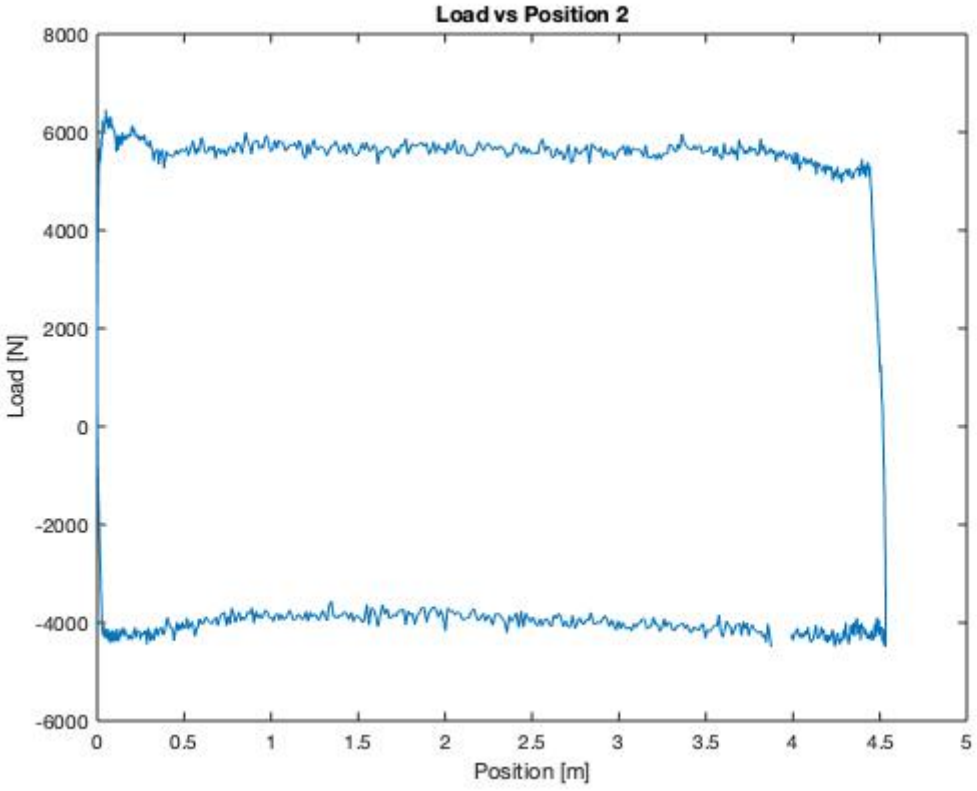


Figure 88: Load vs. position diagram of DDS6 at recording hour 30

- Recording hour 60:

Since the load versus position diagrams not only change with the installation depth of the sensors, but also over time, the results of DDS1, DDS3 and DDS6 at recording hour 60 display various changes compared to the results at hour 30.

Figures 89, 90 & 91 suggest that the pump is not operating under full conditions anymore, which is caused by an increasing pump-off over the field-testing period.

The reservoir performance in general is lacking and once the workover fluid is completely removed, the pump begins to pump-off the wellbore (Langbauer et al., 2019).

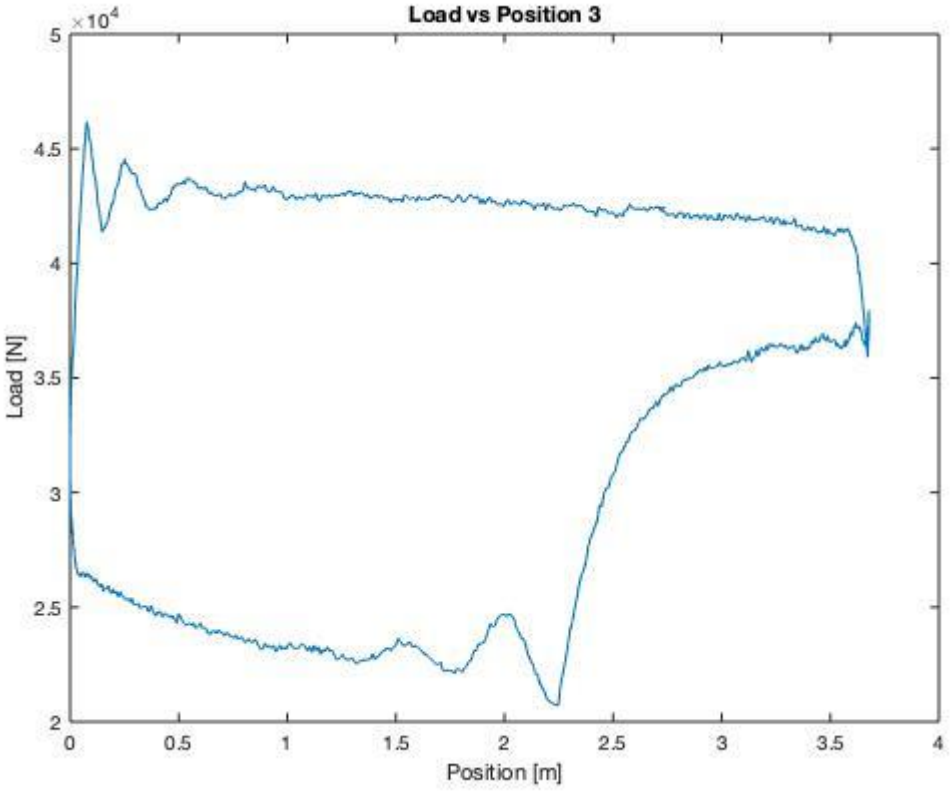


Figure 89: Load vs. position diagram of DDS1 at recording hour 60

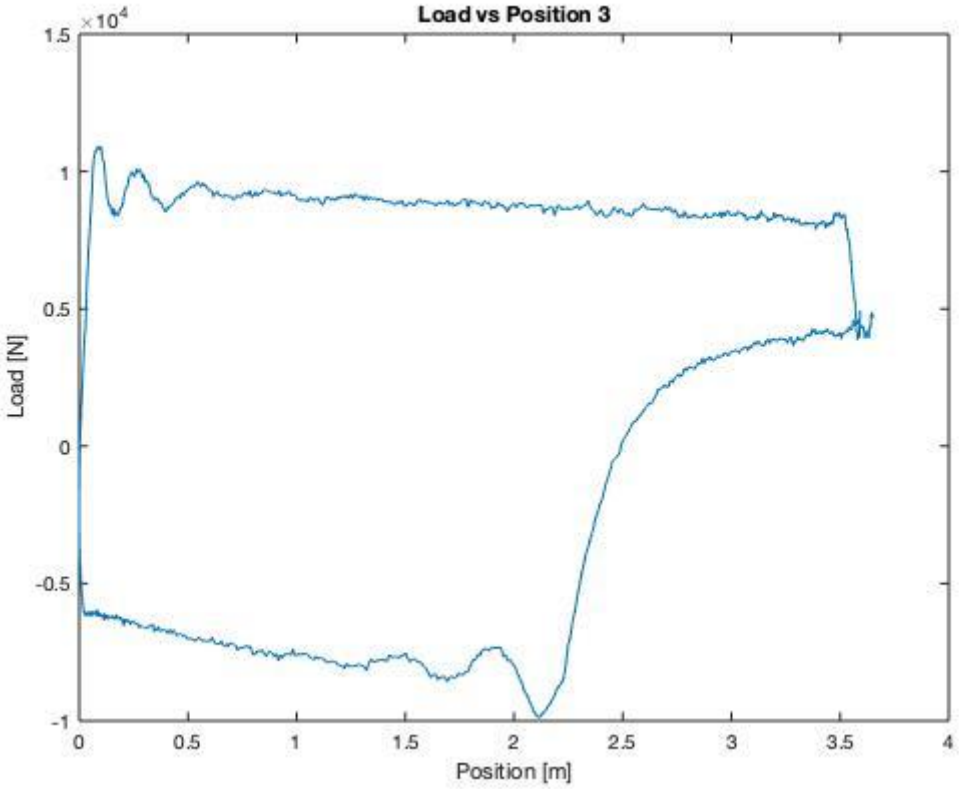


Figure 90: Load vs. position diagram of DDS3 at recording hour 60

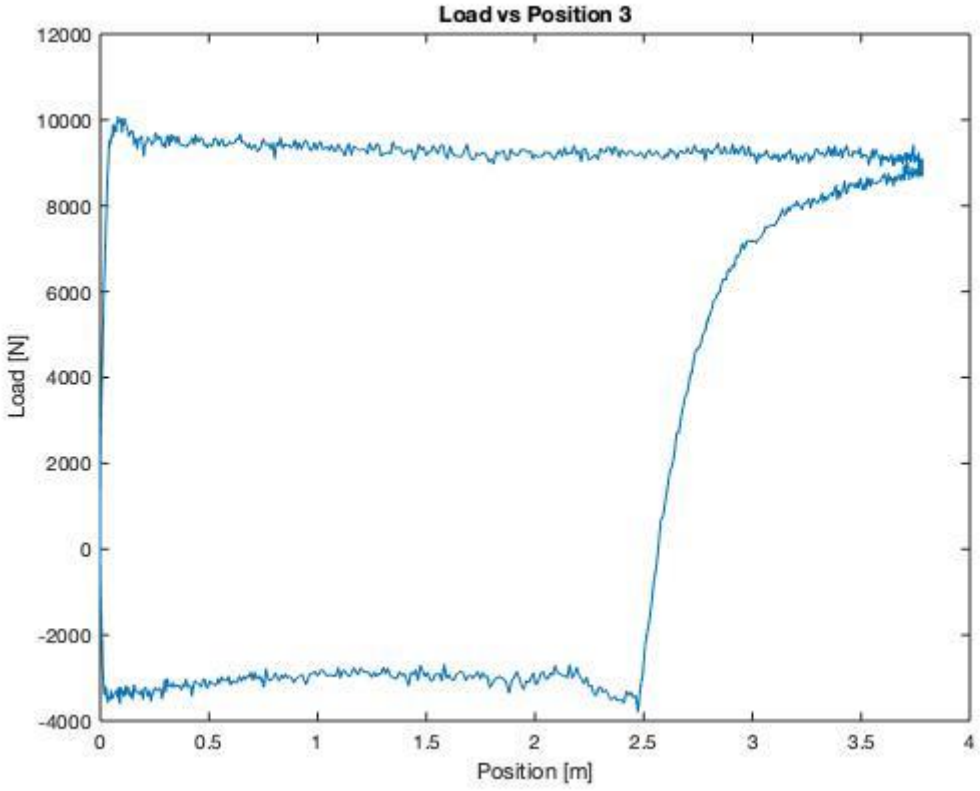


Figure 91: Load vs. position diagram of DDS6 at recording hour 60

- Recording hour 80:

As time progresses, the pump-off continues to increase, which can be seen quite clearly when placing the load versus position diagrams of sensor 6 at hour 80 and hour 60 next to each other, which is shown in Figure 92 (the green horizontal line represents the increased pump-off). Eventually though, the pump-off will reach a certain stabilization and ceases to increase.

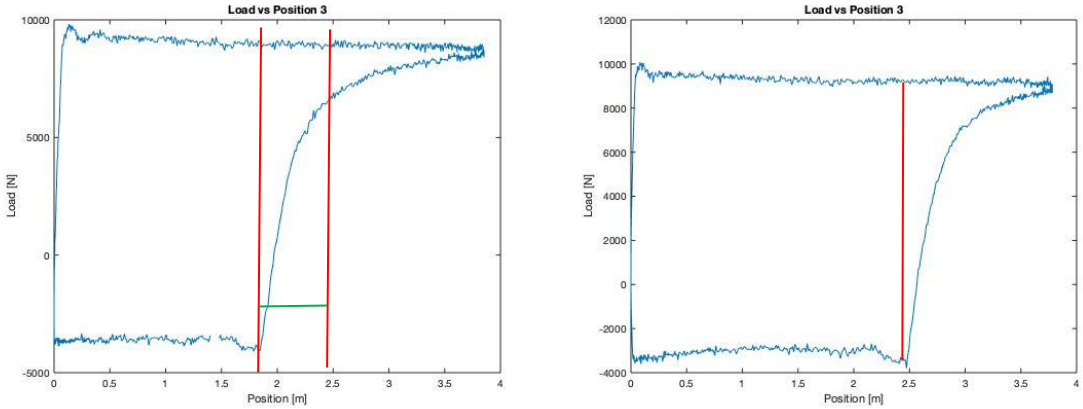


Figure 92: Load vs. position diagram of DDS6 hour 80 (left) compared to hour 60 (right)

Figures 93, 94 and 95 show the resulting load versus position plots of DDS1, DDS3 and DDS6 at recording hour 80. Compared to the previous results at hours 30 and 60, the shapes of the cards continue to change.

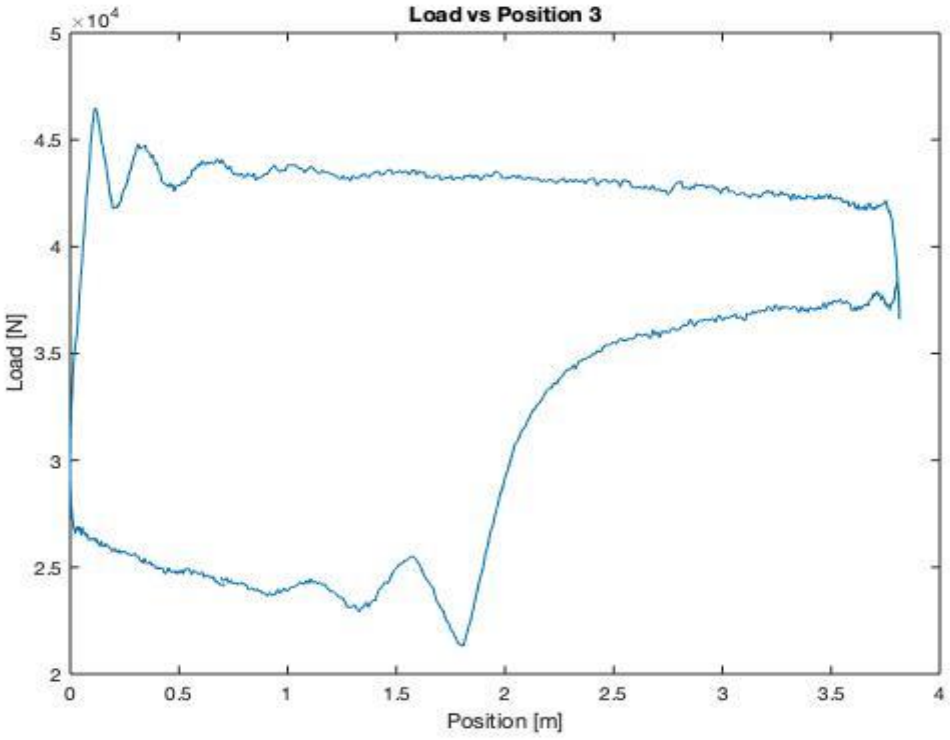


Figure 93: Load vs. position diagram of DDS1 at recording hour 80

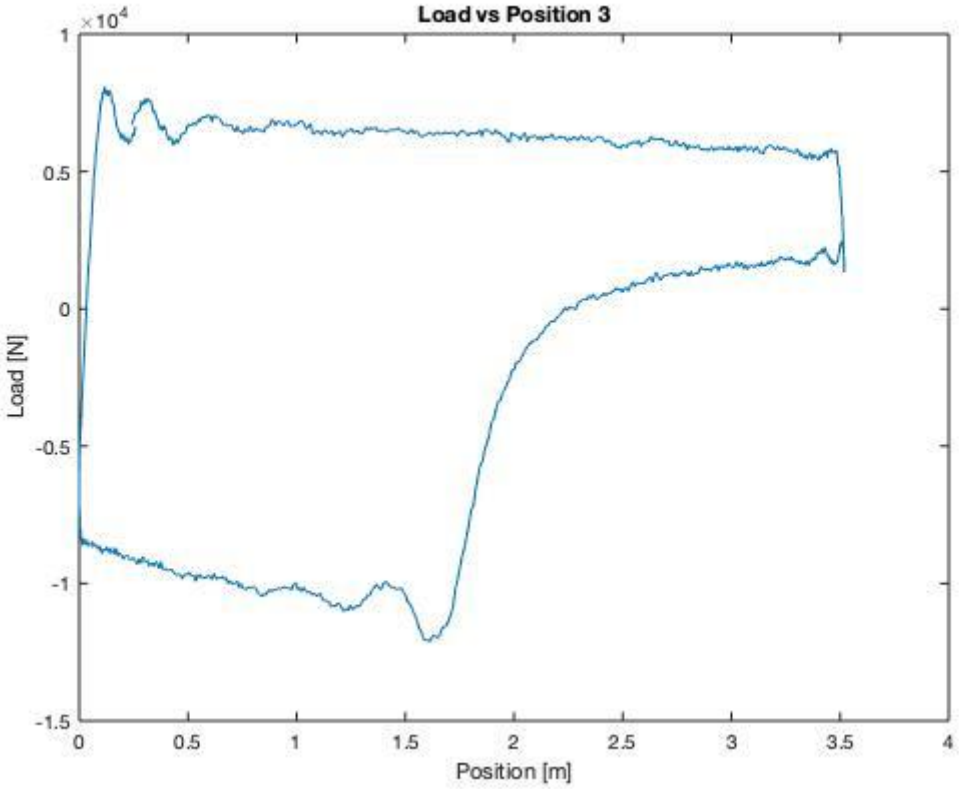


Figure 94: Load vs. position diagram of DDS3 at recording hour 80

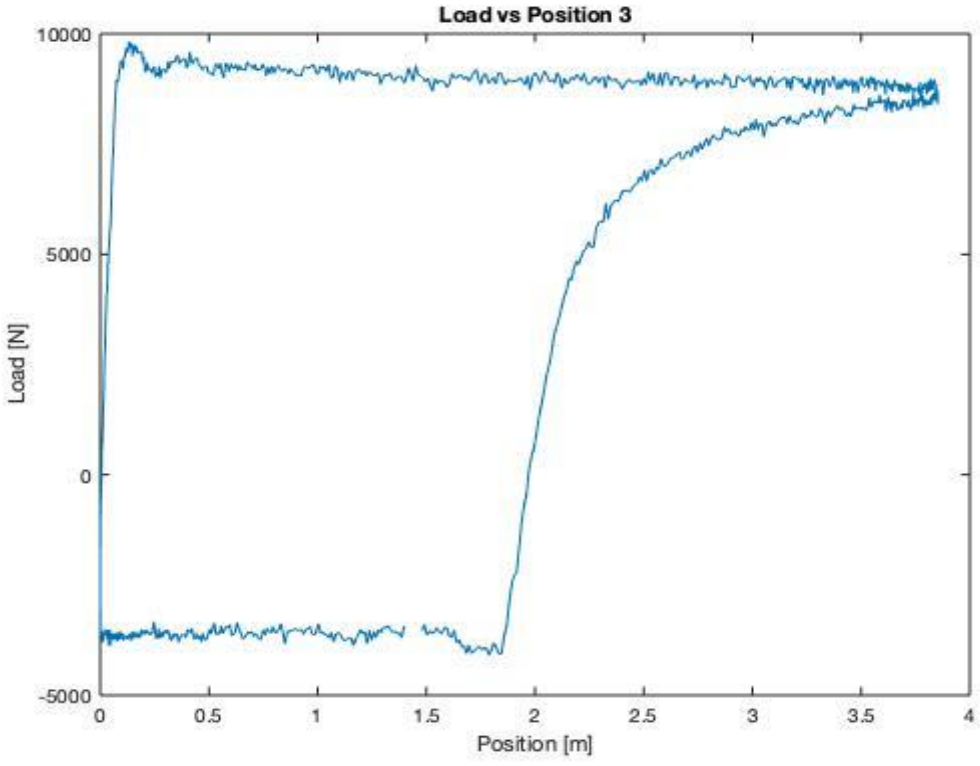


Figure 95: Load vs. position diagram of DDS6 at recording hour 80

A comparison of the surface and downhole dynamometer cards at different recording hours also provides some information about the rod string and pump friction. In general, friction, which is increased by corrosion effects, precipitations, depositions and well trajectory, negatively influences the efficiency of the system and therefore high friction forces should be avoided or mitigated by an optimization of the rod string (Langbauer et al., 2019).

The rod string friction changes over time and its value can be observed in surface dynamometer cards. Figure 96, which shows the load versus position diagrams of DDS1 at 2 different recording hours, indicates that the up- and downstroke friction reach a combined value of around 4,5 to 5 kN

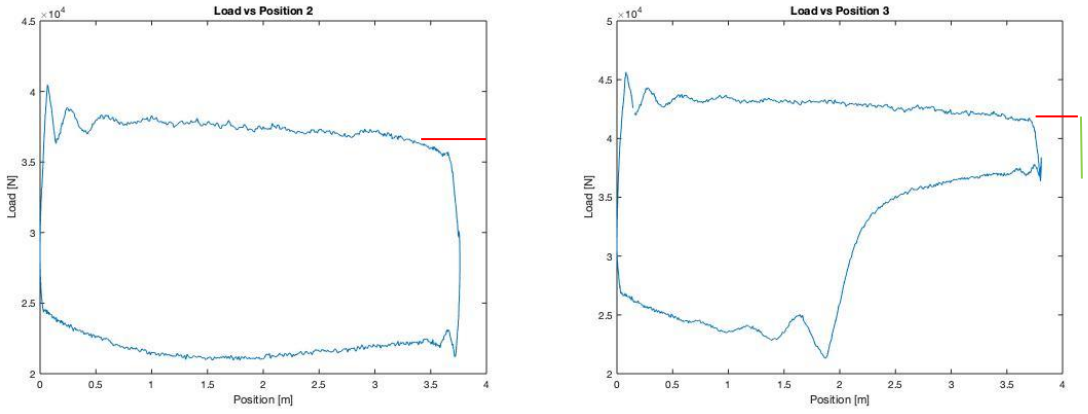


Figure 96: Load versus position diagrams of DDS1 at recording hours 25 (left) and 85 (right)

The pump friction shows a much smaller value and can be seen in Figure 97, where the load versus position plots of DDS6 are compared at recording hours 25 and 85.

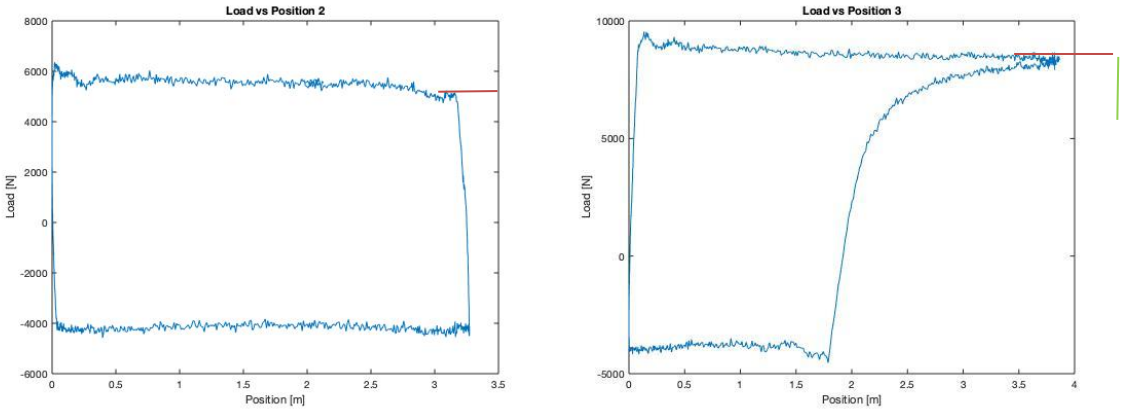


Figure 97: Load versus position diagrams of DDS6 at recording hours 25 (left) and 85 (right)

Furthermore, the downhole dynamometer sensors enable a comparison of the friction forces in the vertical section of the rod string versus the one in the deviated section. During this field example, the friction forces in the deviated portion of the rod string were much more significant.

Afterwards, the surface and plunger stroke lengths are identified:

In Figure 98, the surface stroke length of the pump jack can be established by the displacement in x-direction for DDS1 at recording hour 13, which is in the range of 3,75 meters.

Figure 99, the plot of the displacement in x-direction of DDS6 at recording hour 13, displays the stroke length of the pump plunger, which is approximately 3,5 meters.

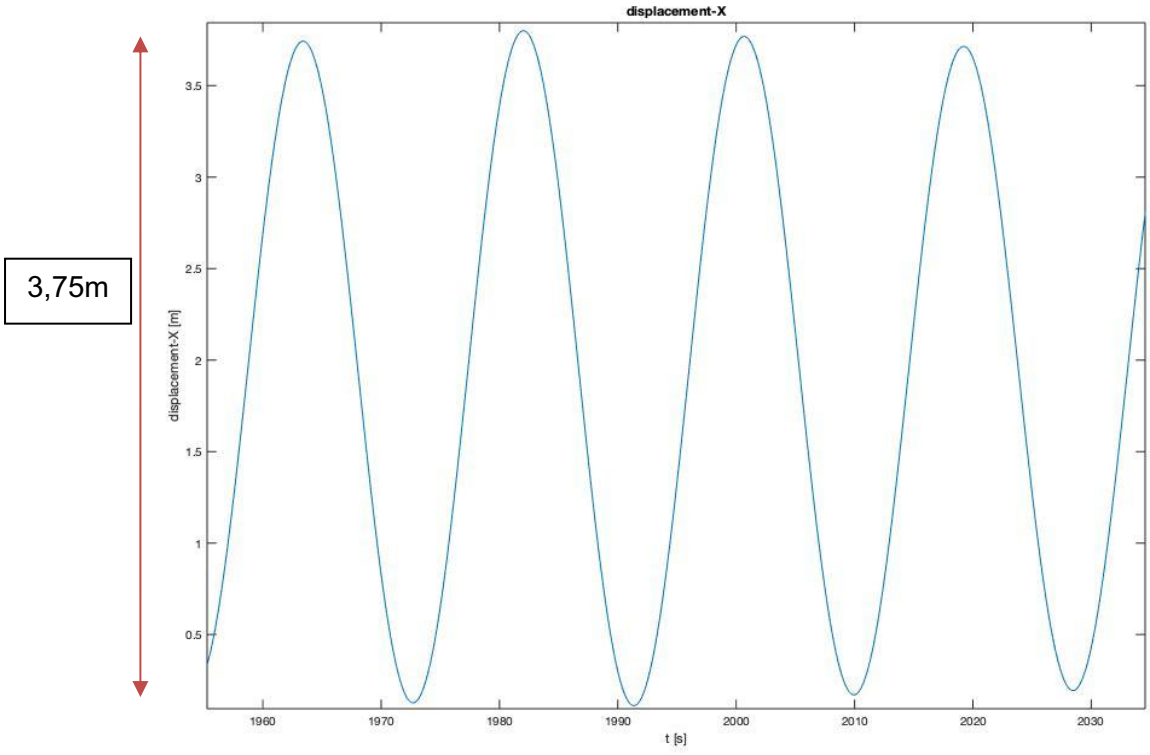


Figure 98: Displacement in x-direction of DDS1 at recording hour 13

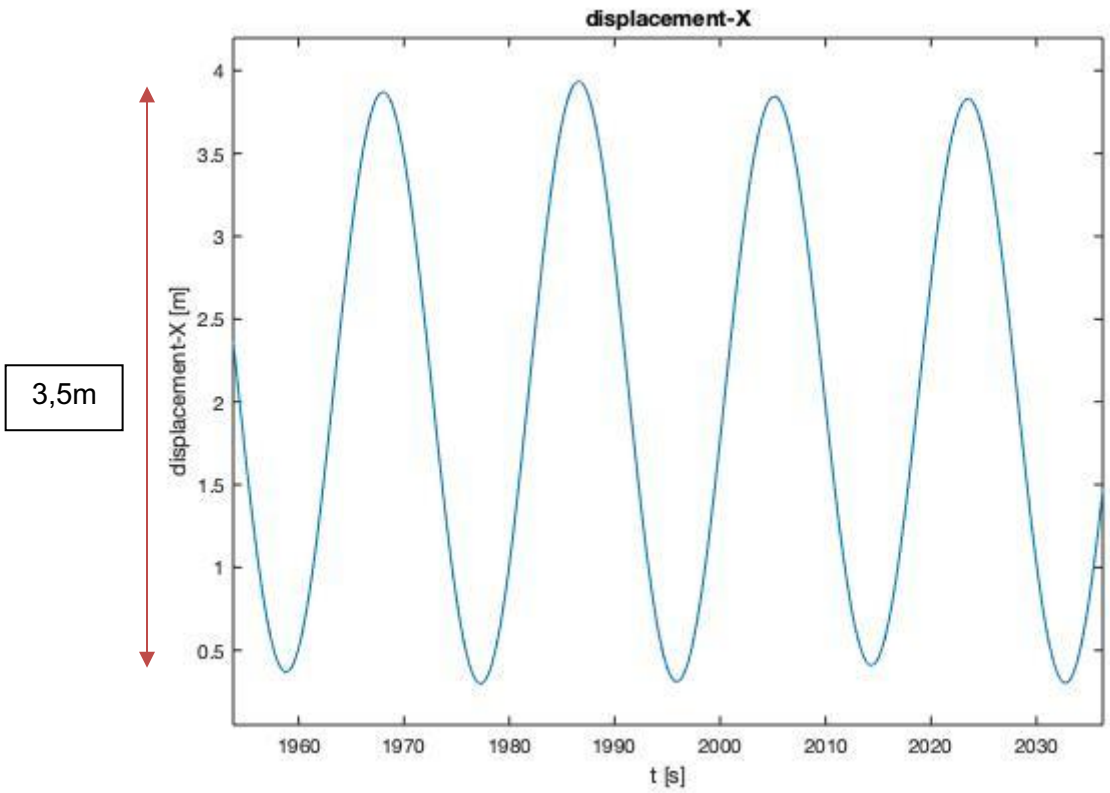


Figure 99: Displacement in x-direction of DDS6 at recording hour 13

Last but not least, the volumetric efficiency of the system can be evaluated by comparing the results of the downhole dynamometer cards over different recording hours.

In the top left load versus position diagram of Figure 100, the workover fluid removal was still in process and the pump was operating under full conditions. A volumetric efficiency of over 90% was reached, but over time, a reduction of the volumetric efficiency occurs due leakage caused by the opening and closing of the standing and traveling valve and the aforementioned pump-off. In the bottom right diagram, the efficiency of the system is already reduced to only around 25%.

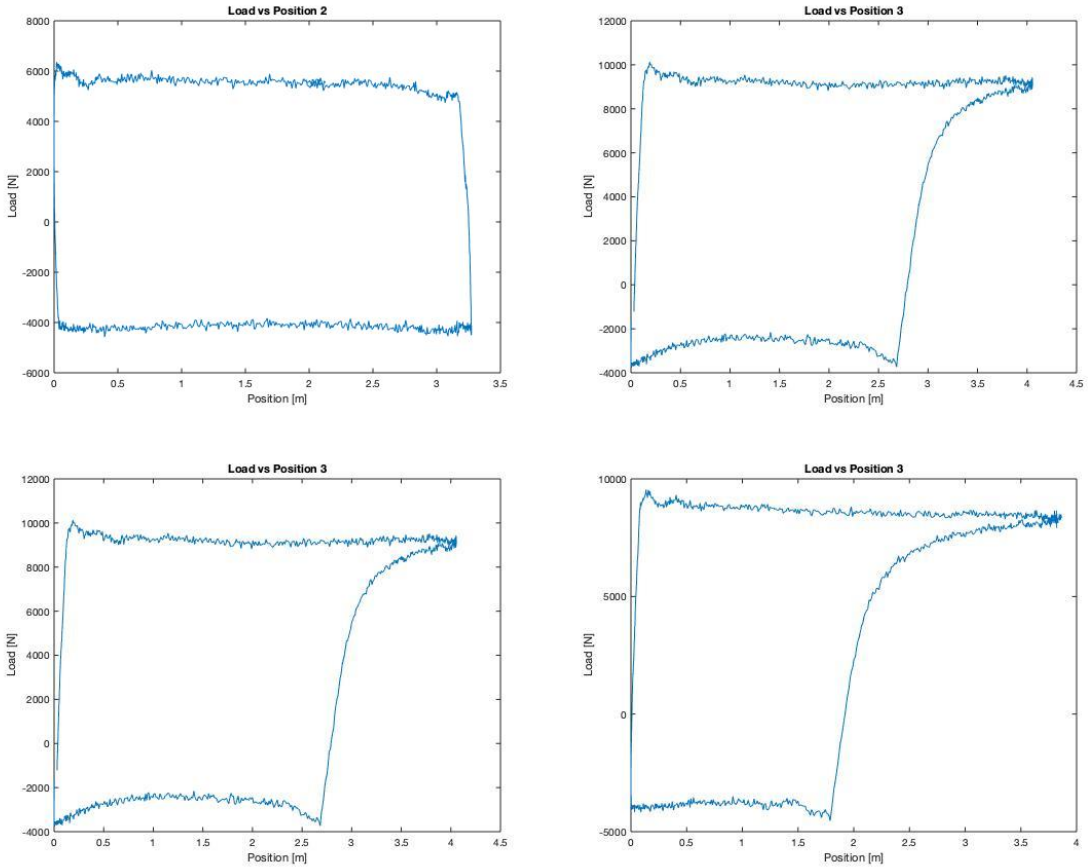


Figure 100: Load versus position diagrams of DDS6 at recording hours 25 (top left), 50 (top right), 75 (bottom left) and 85 (bottom right)

6.5 Conclusion

The following points are the most important conclusions of this thesis:

- The field example in the Vienna Basin oil well proved, that the downhole dynamometer sensors are capable of providing precise downhole information over a long recording period.
- Compared to conventional dynamometers, the DDS enable the gathering of downhole data, which can be used to update exiting predictive simulation softwares and help identify problems.
- Multiple downhole dynamometer sensors at different points of interest along the sucker rod string are necessary for a complete well evaluation.
- Analysis, interpretation and evaluation of the measured data are feasible by using the programming platform MatLab.
- The generated MatLab programs allow an analysis of single datasets and also enable an evaluation over the entire recording period of each downhole dynamometer sensor as well as a comparison of the results.
- The results of the load versus position diagrams of the field example indicate, that the increasing pump-off situation lowered the volumetric efficiency over.
- The temperature analysis demonstrates that the liquid produced from the well lost around 17°C during the lifting period.
- Load fluctuations visible in the strain gauges mean value analysis might be caused by increased friction.
- The remaining battery voltage of all 5 sensors at the end of their recording period indicates, that the recording times for future field test could be expanded.
- Since the sensors are relatively cheap concerning operation and also acquisition, it is recommended to install these devices on a more regular basis, especially in problem wells.

References

[1] Aurelio, Alberto (2019): How does a sucker rod pump work. Available online at http://www.marketingwithmiles.com/how-does-sucker-rod-pump-work/, update on 7/29/2020, checked on 3/10/2020.

[2] Hofstätter, Herbert (2019): Artificial Lift Systems Course 2019.

[3] Brown, Kermit E. (1982): Overview of Artificial Lift Systems.

[4] Downhole Diagnostics (2020): Fluid Level Shots & Dynamometer Cards. Available online at https://www.downholediagnostic.com/dynamometer, updated in 2020, checked on 2/28/2020.

[5] Hasballah Ahba, Ivan Susanto, Nibukat Zaradan, Desy Kurniawan, Lawrence Youngblood, Vadim Shapiro, Dmitriy Khots, (2014): Rod Pump Diagnostics: Maximizing Oil Extraction Performance with Advanced Control Technology.

[6] Norris/A Dover Company (2007): Sucker Rod Failure Analysis: A Special Report from Norris.

[7] Nind, T.E.W. (1964): Principles of Oil Well Production.

[8] Langbauer, Clemens (2019): Artificial Lift Systems Practical Course 2019.

[9] Gibbs, Sam Gavin (2012): Rod Pumping: Modern Methods of Design, Diagnosis and Surveillance.

[10] Sharaf, Sayed Ali (2018): Beam Pump Dynamometer Card Prediction Using Artificial Neural Networks.

[11] De Souza, A. M. Felippe (2009): Using artificial neural networks for pattern recognition of downhole dynamometer card in oil rod pump system.

[12] Abdalla, Ramez Maher Aziz Zaky (2018): Automatic Well Failure Analysis for the Sucker Rod Pumping Systems.

[13] Clemens Langbauer, Karl Diengsleder-Lambauer, Michael Lieschnegg, B. Mostobaev (2019): Sucker Rod Pump Downhole Dynamometer Sensor Technology and Field Application.

[14] Albert, Glenn (1995): Downhole Dynamometer Update, Southwestern Short Course 42nd Annual Meeting, Lubbock, Texas.

[15] Mathworks.com (2020): MatLab Product Description. Available at https://de.mathworks.com/products/matlab.html?s_tid=hp_products_matlab, updated in 2020, checked on 8/12/2020

List of Tables

List of Figures

Abbreviations

DDS	Downhole Dynamometer Sensor
BFPD	Barrels of Fluid per Day
Ft	Feet
°C	Degrees Celsius
PRT	Polished Rod Transducer Dynamometer
k_{tbg}	Tubing Spring Constant
ANN	Artificial Neural Network
CNN	Convolutional Neural Network
BPNN	Backpropagation Neural Network
DDT	Downhole Dynamometer Tools
Hz	Hertz
GB	Gigabyte
mm	Millimeter
kg	Kilogram
kN	Kilonewton
MPa	Megapascal
Ah	Ampere hour
N	Newton
DAQ / DAS	Data Acquisition System
ASCII	American Standard Code for Information Interchange
FAT	File Allocation Table
g	G-Force
mV	Millivolt
DMS	Dehnmessstreifen (strain gauge)
m	Meters
MD	Measured Depth
SPM	Strokes per Minute
h	Hour
s	Seconds
t	Time

Nomenclature

a	Velocity of sound in the rod
r	Radius of rod element
c	Damping coefficient
z	Displacement at x
E	Modulus of elasticity
ϱ	Density (in vertical rod element)
x	Position
dx	Changes in position
F_D	Damping force
F_X	Tension force on the upward pull
F_{X+dx}	Tension force from pull below
t	Time
A	cross-sectional area of rod element
μ	Friction coefficient
ε	Strain
F_A	Gravitational force
γ	Rod material density
ds	Changes in position
F	Axial force on upward pull
F+ds	Axial force on downward pull
v	Viscous damping coefficient
φ	Angle of inclination of the rod string
g	Gravity constant
F_N	Friction force
M	Bending moment
w	Transverse displacement
S	Measured length along the curved rod
R	Radius of curvature
a	Sound velocity of the rod