Technical University of Crete
Electronic & Computer
Engineering Department

ADAPTIVE IDENTIFICATION OF A PNEUMATIC PUMP

DIPLOMA THESIS

From:

Petraki Aliki

Supervisor:

Prof. E. Christodoulou

Examining Committee:

Prof. P. Stavroulakis

Prof. I. Mpoutalis

November 2011

Acknowledgements

I would like to thank all the people that made the fulfillment of this thesis possible.

First of all, my supervisor professor Manolis Christodoulou for his guidance, cooperation and support throughout the duration of this thesis.

My Thesis committee Professor P. Stavroulakis and I. Mpoutalis for their contribution in the preparation of this thesis.

Moreover I would like to express my gratitude to Thomas Giotis PhD student of the Automation laboratory for his invaluable support and toleration and for the time he spent helping me.

I would also like to express my thanks to Professor Enrico Canutto for his guidance, his useful suggestions and contribution.

Furthermore, special thanks to my family for their love, patience and support all these years.

Finally, I couldn't forget to thank my friends for the memorable time we spent during all these years and especially my friend Christos for always being there for me.

Table of Contents

Chapter 1

Introduction

In this thesis we proceed to the adaptive identification of a pneumatic pump. Pumps are used in an incredibly great deal of different applications in our life. Since, virtually any environment where there is a need for moving, displacing or regulating the flow of fluids relies on pumps, their identification with ultimate aim their control is very important. In order to identify this nonlinear dynamical system we use the Recurrent High Order Neural Network model and robust learning algorithms.

In chapter 2 we present the theoretical background of our work, introducing some general facts about pumps. We focus on pump characteristic parameters, how pumps can be classified with a brief description of each category and we emphasize at pneumatic pumps, which are the objective of our study, their applications and how they work.

In chapter 3, the implementation of our pneumatic pump-model takes place. After the presentation our model, we describe the tool we used to implement it, which is the Matlab Simulink environment and then we end up with the implementation of the pneumatic pump-model in Matlab Simulink.

In chapter 4 we deal with the adaptive system identification using Recurrent High Order Neural Networks. After some general facts about neural networks are mentioned, we present the RHONN model and then we focus on some different identification techniques based on this model.

In chapter 5 we apply the adaptive identification scheme using robust learning algorithms presented in chapter 4 to our pneumatic pump-model implemented in chapter 3. In the end we present its results which are really successful.

Finally in chapter 6 we end up with the conclusions of our work and how it could be possibly further developed in the future.

Chapter 2

Pumps

In order to implement and identify our pneumatic pump model, some definitions about how pumps work, the way they are classified and their basic characteristics should be given. That is the subject this chapter deals with. In the end of chapter 2 we emphasize at pneumatic pumps, which are the objective of our study, their operation and some of their applications.

2.1 Introduction

A pump is a machine used to transfer fluids such as liquids, gases or slurries from one place to another. This transfer is accomplished by increasing the pressure of the fluid to the amount needed. A fluid's pressure must be increased in order to raise the fluid from one elevation to a higher. For example when is needed to move liquid from one floor of a building to a higher, or to pump liquid up a hill. Moreover pressure must be increased to move the fluid through a piping system because pipes, valves, and fittings experience frictional losses along the way. These losses vary with the viscosity and density of the fluid, the flow rate and with the geometry and material of the pipe, valves and fittings. There are also process reasons as the pressure of a fluid must often be raised to move the fluid into a pressurized vessel, such as a boiler or fractionating tower, or into a pressurized pipeline. Or, it may be necessary to overcome a vacuum in the supply vessel.

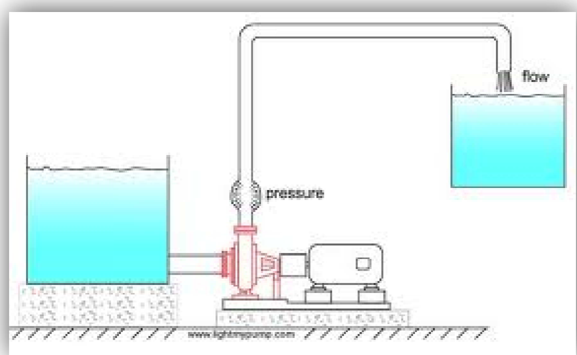


Figure 1.1 A typical pumping system [13]

A pump can be further defined as a machine that uses a lot of different energy transformations to increase the pressure of a fluid. A pump in operation converts the energy available in the engine into potential, kinetic and heat energy of the fluid it transfers. The energy input into the pump is typically the energy source used to power the driver. Most commonly, this is electricity used to power an electric motor. Alternative forms of energy used to power the driver include high-pressure steam to drive a steam turbine, fuel oil to power a diesel engine, high-pressure hydraulic fluid to power a hydraulic motor, and compressed air to drive an air motor.

[2], [4]

2.2 Early History of Pumps

''After the existence of life a perfect pump started operating. The heart! It's characterized as perfect since it works tirelessly for many decades, with variable flow depending on its needs; it is self-adjusting, quiet and self-repairing. No man has ever built a pump which is so gentle, so strong and so reliable.''

The pump is the earliest invention for the conversion of natural energy to useful work, substituting natural energy for human physical effort. It could only contend with the sail for this "title", however the last one cannot be classified as a machine.

The earliest pump we know is the shaduf or swape. It was invented by the Mesopotamians about 3000 B.C. and it is an ancient water-raising device. It was positioned right next to a riverbank, and made with a wooden lever pivoted on two posts placed in the ground upright. On one end of the lever they placed a wooden pole or branch. At the long end of this pole there was attached a bucket, and something heavy like a stone was attached to the other end to serve as a counterweight. Water was retrieved by pushing the pole down until the bucket was filled with water, at which point the counterweight would help raise the bucket back up. It became popular throughout the Middle East, and was the only form of water pump used in that region for the next two thousand years. Around 300 B.C. Ctesibius of Alexandria, often mentioned as the father of pneumatics , invented the ancestor of the modern force pump—which features a cylinder with a plunger or piston at the top that creates a vacuum and draws water upward through valves at the bottom. Vetruvius mentioned this device, as he did the *saqiya*, in his first century B.C. chronicles. Remains of this pump, which was usually made of bronze, have been found in many buildings dating from the days of the Roman Empire. The valves and plungers were particularly valuable inventions that were incorporated into other kinds of machinery, including military equipment. The Romans, for instance, used this kind of pump to hurl flammable liquids at invading Arabs.

About the same time period Archimedes described the Archimedes' screw, also called the Archimedean screw or screw pump. It is mentioned that he described the screw pump because though it is commonly attributed to Archimedes, it was actually used for centuries before him, in Egypt. The Archimedes crew is made of screw inside a hollow metal pipe. The screw is turned usually by a windmill or by manual labor. As the bottom end of the tube turns, it scoops up a volume of water which slides up in the spiral tube as the shaft is turned, until it finally pours out from the top of the tube and feeds the irrigation systems. The contact surface between the screw and the pipe does not need to be perfectly water-tight because of the relatively large amount of water being scooped at each turn with respect to the angular frequency and angular speed of the screw. Also, water leaking from the top section of the screw leaks into the previous one and so on, so a sort of mechanical equilibrium is achieved while using the machine, thus limiting a decrease in mechanical efficiency. This device was firstly used for removing water from the hold of a ship but it was mostly used for draining water out of mines or other areas of low lying water.

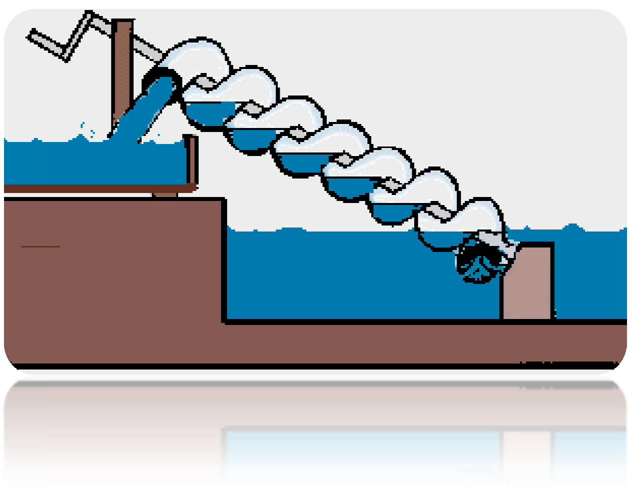


Figure 1.2 The Archimedean screw [15]

The Archimedean screw also persists into modern times. It is still being manufactured for low-head applications where the liquid is frequently laden with trash or other solids. Perhaps most interesting, however, is the fact that with all the technological development that has occurred since ancient times, including the transformation from water power through other forms of energy all the way to nuclear fission, the pump remains probably the second most common machine in use, exceeded in numbers only by the electric motor.
[4], [7], [15], [16], [17], [18]

2.3 Pump Characteristics

A pump is characterized by some basic features. These are the pump's capacity, head, efficiency and power. The fluid quantities associated with all pumps are the flow rate and the head, whereas the basic mechanical quantities involved are the power and the efficiency. Each one of them is being discussed bellow.

2.3.1 Capacity

Capacity (flow rate, discharge or Q) of a pump is the volume of fluid pumped per unit of time. It is commonly measured in cubic meters per second $\left(\mathrm{m}^3/\mathrm{s}\right)$ (in SI units) for large pumps or cubic meters per hour $\left(\mathrm{m}^3/\mathrm{h}\right)$ and litres per second $\left(\mathrm{lt}/\mathrm{s}\right)$ for smaller pumps.

The requirements of the system in which the pump is located, is what determines the required capacity of the pump. We can distinguish the following specific meanings of capacity:

- Nominal Capacity $\mathrm{Q_N}$ is the capacity for which the pump is ordered and applies to the pump with nominal speed $\mathrm{n_N}$, with nominal total head $\mathrm{H_N}$ and with pumped fluid which is indicated in the contract order (conventional liquid/gas).
- Minimum Capacity $\mathrm{Q_{min}}$ is the minimum permissible capacity in which the pump can operate continuously without being damaged.

- Maximum Capacity Q_{max} is the Maximum permissible capacity in which the pump can operate continuously without being damaged.
- Optimal Capacity Q_{opt} is capacity in the point of maximum efficiency with nominal speed n_N and with the conventional liquid/gas.

Mass Flow m_f of a pump is the product of the density of the pumped fluid ρ with the capacity and it is given by the type:

$$m_f = \rho Q.$$

[2], [7]

2.3.2 Head

The energy imparted to a fluid by a pump is measured as the head per unit weight of fluid.
It is expressed in meters (in SI) because it is the height of the column of the same fluid which contains the same amount of energy.
It is important to realize the relationship between head and pressure. Any pressure expressed in SI Pa is equivalent to a static volume of liquid expressed in m (SI) of head. Conceptually head is a specific energy term and pressure is a force applied to an area, however if for example we would like to measure the pressure at the bottom of liquid it would be equal to: $P=H\,p\,g$ where P is the static pressure in Pa, H the head in m, p the density of the liquid and g the gravitational constant.
To determine the required size of a pump for a particular application, all the components of the head for the system in which the pump is to operate must be added up to determine the pump total head.
There are four separate components of total system which are:

2.3.2.1 Static Head

Static head (h or H) is the elevation of a free surface of fluid above (or below) a reference datum which is chosen arbitrarily. In most cases, static head is normally measured from the

surface of the fluid in the supply vessel to the surface of the fluid in the vessel where the fluid is being delivered.
[2], [7]

2.3.2.2 Total static head

Total static head is the useful mechanical energy imparted by the pump per unit weight of the fluid and it is equal to the difference of head at discharge (h_d) and head at suction (h_s).

The total static head is measured from supply vessel surface to delivery vessel surface, regardless of whether the pump is located above the liquid level in the suction vessel (which is referred to as a “suction lift”), or below the liquid level in the suction vessel (“discharge head”). Figure below demonstrates an example of a pump on a suction lift, and defines static suction lift, static discharge head, and total static head. Note that for a pump in a closed loop system, the total static head is zero.

[2], [7]

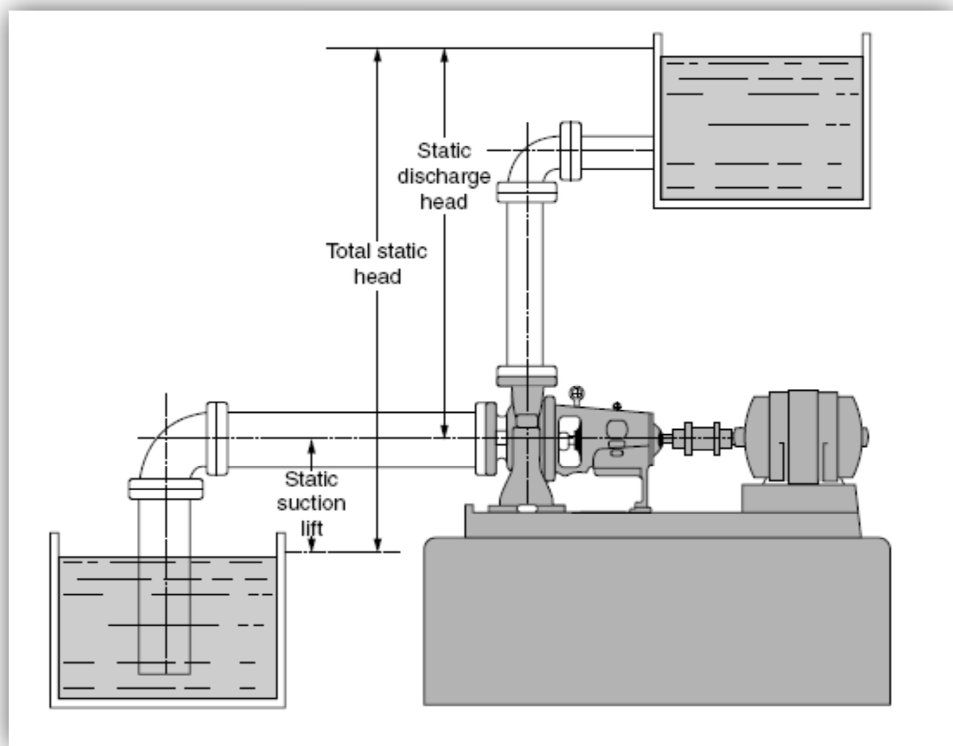


Figure 1.3 Static suction lift, static discharge head, and total static head [19]

2.3.2.3 Total head

The determination of the total head of the pump from the suction and discharge gauge indications is given by:

$$H = \frac{(p_d - p_s)}{\rho g} + \frac{(v_d^2 - v_s^2)}{2g} + z_d - z_s,$$

where p_d is the pressure at pump discharge nozzle,

p_s is the pressure at pump suction nozzle,

ρg is the product of the fluid's density with the gravitational constant,

v_d is the flow velocity at discharge nozzle,

v_s is the flow velocity at suction nozzle and

$z_d - z_s$ is the vertical distance of gauges.

[7]

2.3.2.4 Friction Head

Friction head or resistance head (or head loss) is the head necessary to overcome the friction in the pipes, valves, fittings and elbows of the system in which the pump operates. It takes a force to move the fluid against friction, like the force that is required to lift a weight. The direction of the force that is exerted is the same as the moving fluid and energy is expended. The friction head is calculated with the force required to overcome friction times the displacement (pipe length) divided by the weight of fluid displaced. Friction loss in a piping system varies as the square of the liquid's velocity (assuming fully turbulent flow). The smaller the size of the pipe, valves, and fittings for a given flow rate, the greater the friction head loss.

In designing a piping system, if smaller sizes of pipes, valves, and fittings are chosen, the cost of the piping system is reduced. However, the trade-off is that this has as result higher total pump head due to the increased friction head loss. This, in turn, usually increases pump and driver capital cost, and also increases lifetime energy costs.

In theory, friction losses that take place as liquid flows through a piping system must be calculated by means of complicated formulae, taking into account such factors as liquid density and viscosity, and pipe inside diameter and material. Luckily, the information needed

is gathered empirically, and then recorded in tables so that we can estimate these values according to the flow, the pipe size, the pipes material it is constructed out of, pipe age and any deposits, the type of valve, etc. This additional resistance to flow must be compensated for, in order to deliver the desired flow rate.
[2], [20], [21]

2.3.2.5 Pressure Head

Pressure head is the head required to overcome a pressure or vacuum in the system upstream or downstream of the pump. It is normally measured at the fluid surface in the supply and delivery vessels. This happens due to the static pressure of the fluid and it is equal to $\frac{p}{\rho g}$, where p is fluid's pressure and ρ is fluid's density. If the pressure in the supply vessel from which the pump is pumping and the pressure in the delivery vessel are identical, then there is no required pressure head adjustment to total head. Likewise, there is no pressure adjustment to total head for a closed loop system. If the supply vessel is under a vacuum or under a pressure different than that of the delivery vessel, a pressure head adjustment to total head is required. [2], [7]

2.3.2.6 Velocity Head

Velocity head (H_v) is the energy produced due to the fluid's motion at some velocity. It is equal to:

$H_v = \frac{v^2}{2g}$, where v is fluid's velocity.

It can be measured with a Pitot tube which can measure fluid's flow. The value of velocity head is different at the suction and discharge of the pump, because the size of the suction piping is usually larger than the size of the discharge piping. So in order to determine the velocity head component of total head it is necessary to calculate the change in velocity head from suction to discharge. However the change of velocity head is often less than 1% of total head many pump selectors choose to totally ignore the effect of velocity which is not always a valid assumption. [2], [7]

2.3.3 Power and Efficiency

Pump power is the power of the pump typically expressed in W or KW. It refers to the amount of energy a pump needs to be supplied in order to operate.

The output pump power $\mathrm{N_D}$ is the output of the pump handling a fluid of a given specific gravity, with a given flow and head and is calculated as follows:

$$\mathrm{N_D} = \rho\, \mathrm{g\, Q\, H},$$

Where ρ is the fluid's density $\left(\mathrm{kg/dm^3}\right)$,

g is the gravitational constant approximately $9{,}81\ \mathrm{m/s^2}$,

Q is the flow rate $\left(\mathrm{m^3/s}\right)$ and

H is the total head (m).

The input pump power N is the actual amount of power that must be supplied to the pump to obtain a particular flow and head. It is the input power to the pump or the required output power from the driver and is calculated as follows:

$$\mathrm{N} = \frac{\rho\ \mathrm{g\ Q\ H}}{\eta}, \quad (KW)$$, where η=pump efficiency.

Another equation that can also be used for the calculation of the input pump power is the following:

$$\mathrm{N} = \frac{\rho\ \mathrm{Q\ H}}{367\eta}\ (KW)$$

Where

ρ is the fluid's density $\left(\mathrm{kg/dm^3}\right)$,

Q is the flow rate $\left(\mathrm{m^3}/h\right)$,

H is the total head (m) and

η=pump efficiency.

The pump efficiency is expressed as a decimal number less than one; it is equal to the ratio of the output and input pump power and is calculated as follows:

$$\eta = \frac{N_D}{N}.$$

There are many factors that cause pumps to be less efficient. Some of them are hydraulic loses, mechanical loses, volumetric and disk friction loses.
[2], [7], [8]

2.4 Pump Classification

There are many ways to classify pumps, such as on the basis of the applications they serve, the materials of construction, the liquids they handle, their conditions of service, or the way they are oriented in space.

Here, in our system of classification, pumps are classified by general mechanical configuration.
Under this system, all pumps may be divided into two major categories:

- **Kinetic or (roto)-dynamic pumps**
  In this type of pumps, energy is continuously added to the liquid in order to increase its velocity. When velocity is increased to values greater than those that occur at the discharge, then subsequent velocity reduction within or beyond the pump produces a pressure increase. The basic two groups in which they are subdivided are centrifugal pumps and vertical-turbine pumps.

- **Positive displacement pumps**
  In this type of pumps, energy isn't added continuously but periodically to the liquid by the direct application of a force to one or more movable volumes of liquid. This causes an increase in pressure up to the value required to move the liquid through ports in the discharge line. Positive displacement pumps are subdivided into three major groups which are reciprocating, rotary and pneumatic pumps.[2], [4], [5], [24]

The classification of pumps into kinetic and positive displacement pumps and into their sub-categories is illustrated at the following figure:

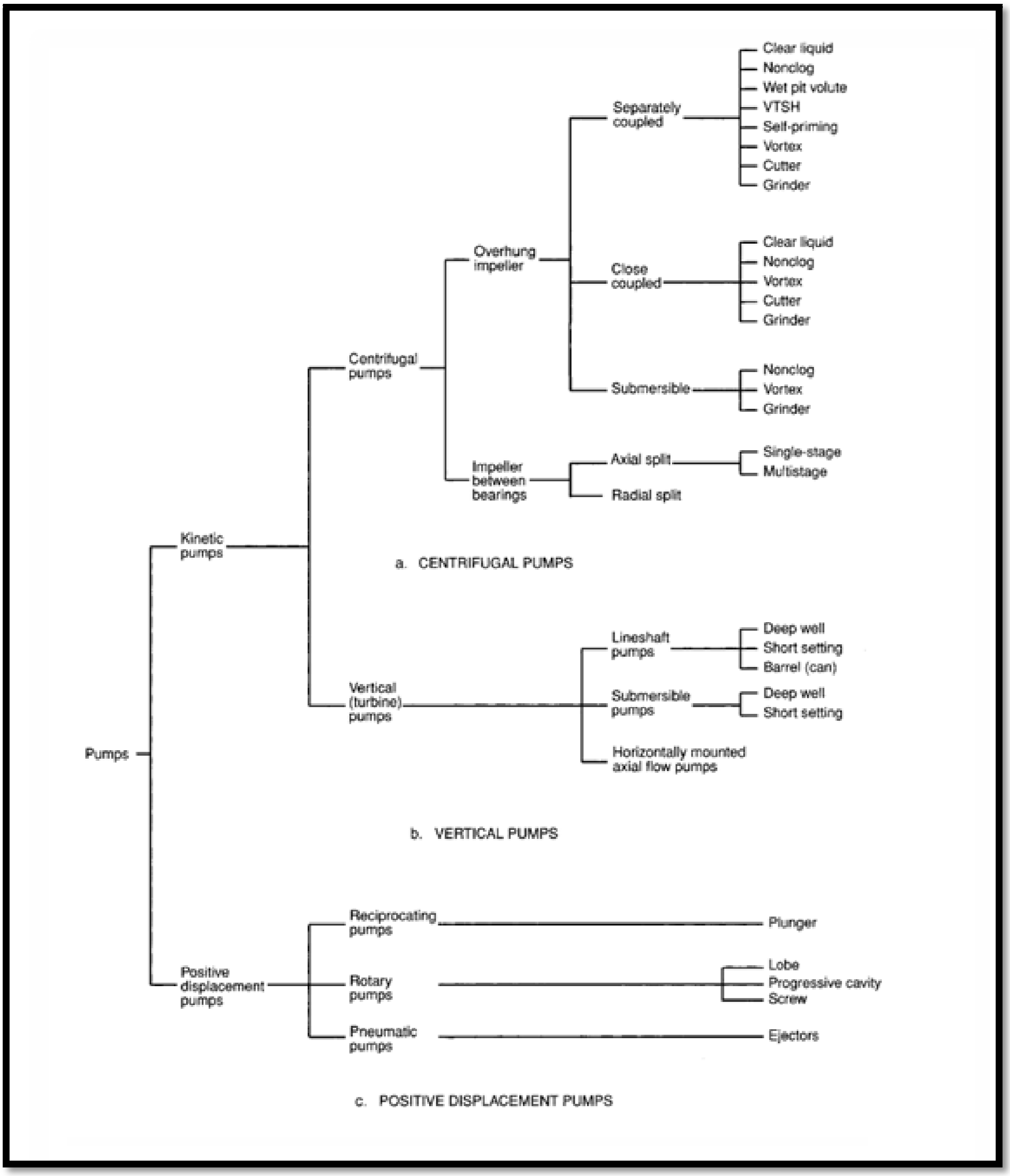


Figure 1.4 Classification of pumps. Courtesy of hydraulic institute standards [5]

2.4.1 Kinetic Pumps

Kinetic pumps impart velocity and pressure to the fluid as it moves past or through the pump impeller and, subsequently, convert some of that velocity into additional pressure. They are inexpensive and have low maintenance requirements; their size is rather small so there is no demand for large area of installation and they do not require external lubrication. However, they have low efficiencies when the flow rate is low and the pressure is high.

The two major groups of kinetic pumps are centrifugal (or volute) and vertical (or turbine) pumps. [5], [7], [8]

2.4.1.1 Centrifugal pumps

A centrifugal pump consists of an impeller mounted on a rotating shaft and a pump casing that encloses the impeller. As the impeller rotates the liquid moves toward the discharge side of the pump into the casing surrounding the impeller. This movement has as result the creation of a void or reduced pressure area at the impeller inlet.

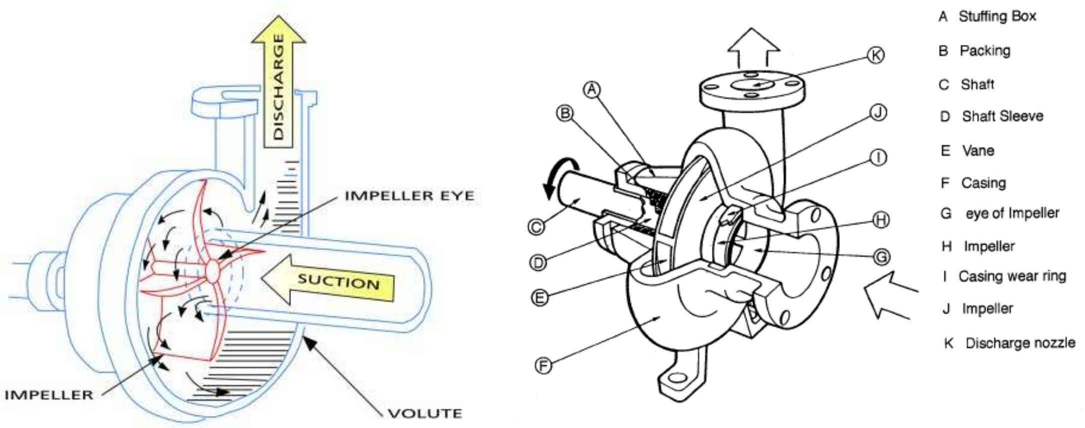


Figure 1.5a, 1.5 b centrifugal pump parts

The difference between this pressure area and the higher pressure of the casing inlet leads to force additional liquid into the impeller to fill the void. Once it reaches the rotating impeller, the liquid entering the pump moves along the impeller vanes, increasing in velocity as it progresses. As the liquid leaves the impeller vane, the liquid is at its maximum velocity.

The vanes of the rotating impeller impart a radial and rotary motion to the liquid, forcing it to the outer periphery of the pump casing, where an expansion of cross-sectional area occurs. This cross-sectional area of the flow passages increases as the liquid moves through the casing resulting in a diffusion process causing the liquid's velocity to decrease. The decreased kinetic energy is transformed into increased potential energy, causing the pressure of the liquid to increase as the velocity decreases. The fluid is then discharged from the centrifugal pump through the discharge connection.

[2], [11], [12]

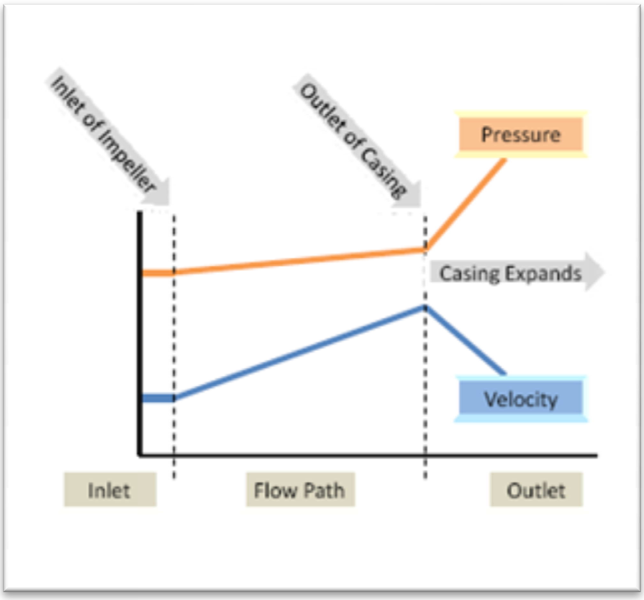
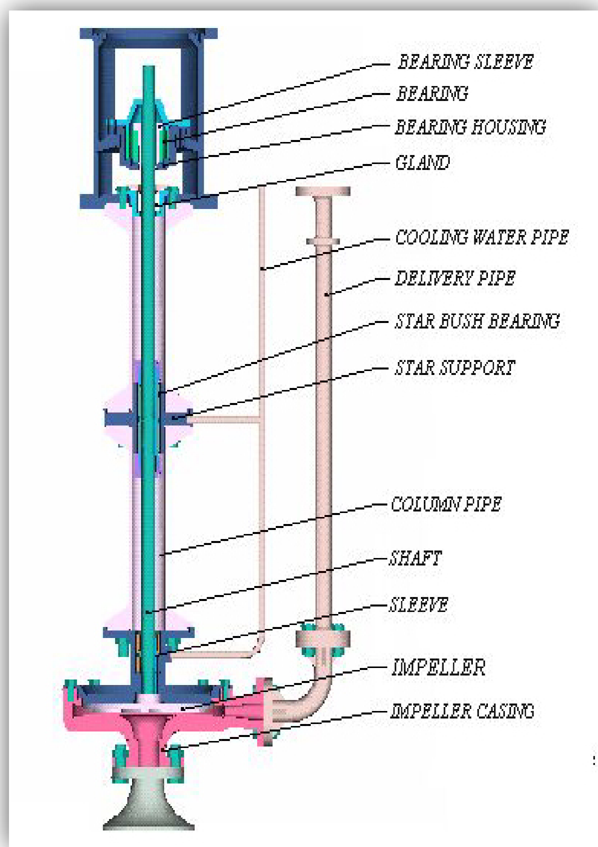


Figure 1.6 pressure vs. velocity through a centrifugal pump [12]

Centrifugal pumps have low maintenance expense and they are characterized by simplicity. They don't have a great deal of moving parts and they have no check valves associated with the pumps. Centrifugal pumps produce minimal pressure pulsations, they do not have rubbing contact with the pump rotor, and are not subject to the fatigue loading of bearings and seals that the periodic aspect of many positive displacement pumps produce. A basic problem of centrifugal pumps is that they develop cavitation. This happens when vapors of the liquid being pumped are present on the suction side of the pump and can cause serious damage to the pump. [2]

2.4.1.2 Vertical Pumps

Vertical pumps are equipped with an axial diffuser or discharge bowl that performs the same basic functions as the volute. Vertical pumps were originally developed for well pumping.



The bore size of the well limits the outside diameter of the pump and so controls the overall pump design. These pumps are very versatile and are often used for installations not related to well pumping.

The basic components of which a typical vertical pump is consisted are the bowl assembly, the column, the discharge head and the driver. Vertical pumps can be sub-divided into many categories. The most common are line shaft pumps, submersible, and horizontally mounted axial-flow pumps.

ump [22]

[4], [5]

2.4.2 Positive Displacement Pumps

In positive displacement pumps, the fluid flows into a contained space and a moving element such as a piston forces the fluid out of the cylinder increasing the pressure. This contained space could be a cylinder, plunger, or rotor.

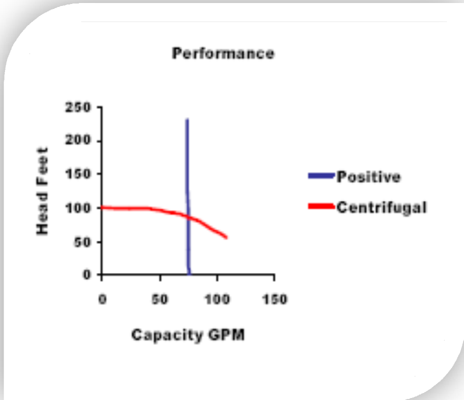
[3], [5]

2.4.2.1 Positive Displacement and Centrifugal Pumps Comparison and Characteristics

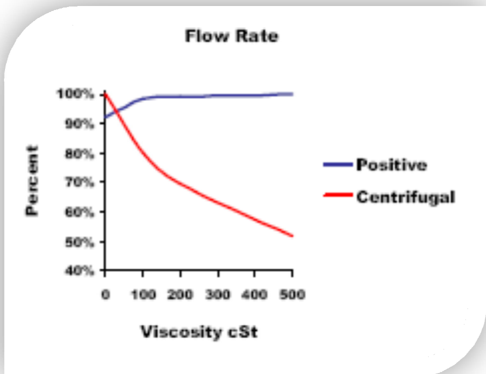
The choice between a centrifugal and a positive displacement pump depends on the behavior of these two types. The characteristics of each pump (flow, head, efficiency, power) although they are of equal importance; the emphasis placed on certain of these quantities is different for different pumps. The output of a pump running at a given speed is the flow rate delivered by it and the head developed. Thus, a plot of head and flow rate at a given speed forms the fundamental performance characteristic of a pump. In order to achieve this performance, a power input is required which involves efficiency of energy transfer. Thus, it is useful to plot also the efficiency η against Q.

Hence, after studying carefully the following characteristic curves of the two types of pumps, it can be concluded when the use of positive displacement pumps is more common.

The Q-H characteristic curve of the positive displacement pump is a straight line perpendicular to the axis of flow; therefore the capacity is constant and independent of the head and pressure, whereas the capacity of a centrifugal pump is varying and depends on pressure and head.



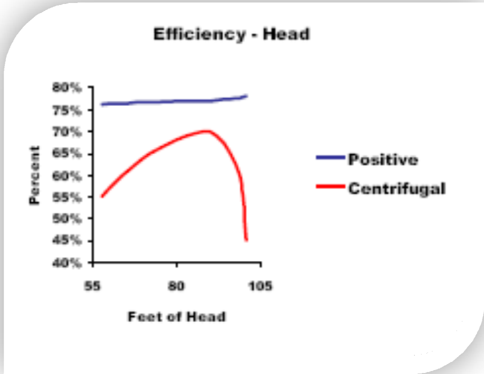
cteristics of a
ntrifugal pump



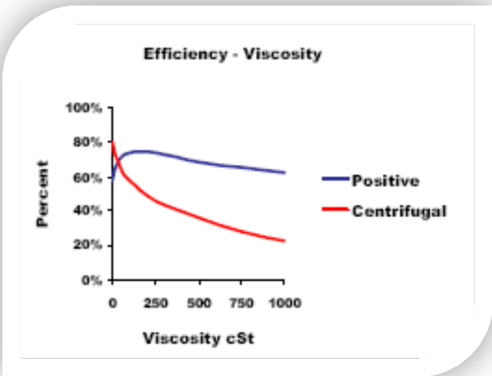
This chart portrays how the capacity of the pump is affected by viscosity. There is an increase in positive displacement pump's flow as the viscosity rises, whereas in centrifugal pumps, the increase of the viscosity implies the reduction of the pump's flow rate. This is due to the fact that the higher viscosity liquids fill the clearances of the pump causing a higher volumetric efficiency.

istics of
ıtrifugal

The H-η chart on the right illustrates how pressure changes influence pump's efficiency. The curve of the positive displacement pump indicates that pump's efficiency is slightly affected by changes of pressure. However pressure changes have a great impact on the efficiency of the centrifugal pump.



ead characteristics
ıt and a centrifugal



characteristics
a centrifugal

In centrifugal pumps the rise of the viscosity implies the rapid drop of the pump's efficiency. This happens because of the frictional losses within the pump. In a positive displacement pump efficiency often increases with the rise of the viscosity.

The data presented above are adopted from [23] and are the actual data of a specific application. They are presented as an example of the performance behavior and in other applications these charts would have taken different curves and efficiency values. The centrifugal pump was selected at its best efficient point and the positive displacement pump was picked to match the flow, viscosity and pressure.

According to the above mentioned points, positive displacement pumps are used in case of high viscosity, when there are variations in pressure because they maintain their flow rate and obviously in case variations in viscosity exist since centrifugal pumps become very inefficient at even modest viscosity. Moreover, they are the best choice in high pressure applications and they have the ability to handle shear sensitive liquids better than centrifugal pumps, since pumps shear liquids more when the speed is increased and centrifugal pumps are high speed pumps. [23], [2] The three basic types of positive displacement pumps are:

- Reciprocating pumps
- Rotary pumps
- Pneumatic pumps

2.4.2.2 Reciprocating Pumps

In a reciprocating pump, a piston or plunger moves up and down. During the suction stroke, a volume of liquid is drawn into the cylinder and is discharged under positive pressure through the outlet valves on the discharge stroke. The discharge from a reciprocating pump is pulsating and changes only when the speed of the pump is modified. This is due to the fact that the intake is always a constant volume. Reciprocating pumps must always be operated with over-pressure protection because they can develop very high pressures and a relief valve or bursting disc must be fitted.

Types of reciprocating pumps include plunger pumps, piston pumps, metering pumps and diaphragm pumps. Simplex pumps are reciprocating pumps with a single piston, plunger, or diaphragm. Names for multiple-cylinder pumps are duplex, with two cylinders, triplex, with three cylinders, quadruplex, with four cylinders, quintuplex, with five cylinders and multiplex, with many cylinders.

[3], [5], [9]

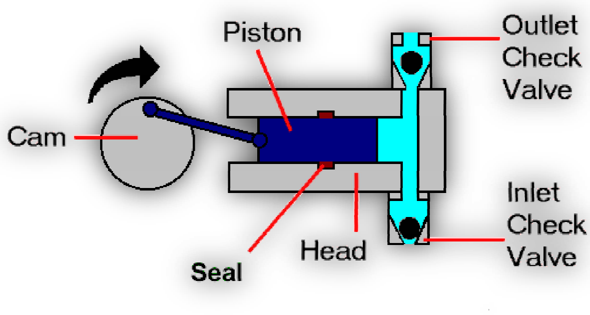


Figure 1.12 A simple piston reciprocating pump's part [25]

2.4.2.3 Rotary pumps

A rotary pump has a rotary displacement element, such as gears, screws, vanes, or lobes. Each compartment between the dividing elements will hold a determined volume of fluid. As the first compartment fills with liquid, the fluid in the last compartment flows into the discharge piping. They can handle almost any liquid that does not contain hard and abrasive solids, including viscous liquids. They are also simple in design and efficient in handling flow conditions that are usually considered too low for economic application of centrifugals. Types of rotary pumps include lobe pumps, progressing cavity pumps and screw pumps.

[5]

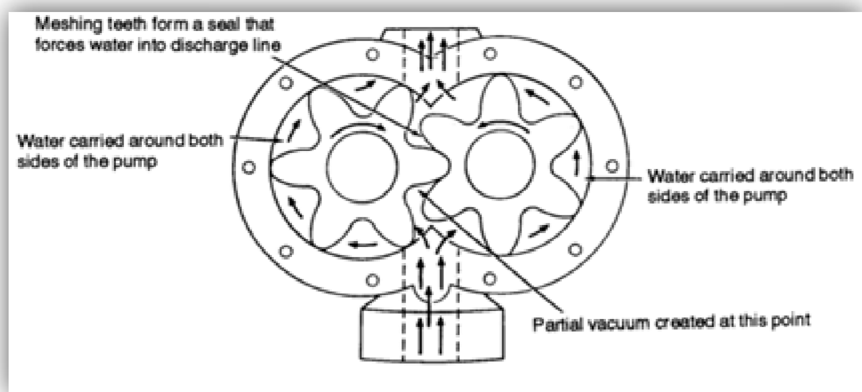


Figure 1.13 How a rotary pump works [26]

2.4.4 Pneumatic Pumps

Pneumatics is the science which deals with the study and application of use of pressurized gas to effect mechanical motion. The term pneumatics is derived from the Greek word pneuma which signifies breath or air. Though the science of air was known to man for centuries, it wasn't used in industry before the beginning of the Second World War. This was the age when the present day concept of automation started provoking man to use compressed air in production plants. Nowadays, pneumatic systems, air operated tools and accessories are used in every aspect of industrial life, where factories are commonly plumbed with compressed air or compressed inert gases. Compressed air is air which is kept under a certain pressure, usually greater than that of the atmosphere. In Europe, 10 percent of all electricity used by industry is used to produce compressed air, amounting to 80 terawatt hour consumption per year. An inert gas is a non-reactive gas used during chemical synthesis, chemical analysis, or preservation of reactive materials. Inert gases are selected for specific settings for which they are functionally inert since the cost of the gas and the cost of purifying the gas are usually a consideration. Neon and argon are the most common inert gases for use in chemistry and archival settings.

Pneumatics also has applications in dentistry, construction, mining, medicine, entertainment and other areas.

[27], [28], [29], [30]

Pneumatic pumping and generally the use of pneumatic systems have many advantages and that's why they are so extensively used. One of the most important is the simplicity and flexibility of design, since a pneumatic system can be easily designed using standard cylinders and other components. The wide availability and the compressibility of the air, as well as the fact that compressed air can be easily transferred in pressure vessels, containers and long pipes and can be stored, allowing the use of machines when electrical power is lost, are some of the reasons that make the application of pneumatics in industries more advantageous. By the same token, pneumatic systems are very safe because they do not pose any risk of fire or explosion since a probable leak would not cause contamination in the way that hydraulic systems leaking oil would. The initial cost of such a system is very low because of the simplicity of their design and the inexpensive materials they are composed of, but the long term operating cost of a pumping system could be very high because of the amount of energy needed for the gas compression. Compared to hydraulic systems, pneumatics has better operational advantages but hydraulic systems are indispensable in terms of power requirement and accuracy of the operations.

[27], [30]

Pneumatic pumps are the pumps that use compressed air or other pressurized gas in order to move fluids from one place to another. Pneumatic pumps are classified as positive displacement pumps because it is their most common use however there are also pneumatic centrifugal pumps. The two most common and interesting applications of the pneumatic pumps are the pneumatic ejector and the air lift pump.

[5]

2.4.4.1 Pneumatic ejector

Pneumatic ejector is a pump used to raise sewage or sludge. The liquid is displaced by compressed air from a gravity-fed pressure vessel through a check valve into the discharge line. It pumps low flow rates of wastewater at high heads and it was developed in 1870s by I. Shone. The flow rates in which it is used amount to approximately 0,04 m^3/sec and it can operate at discharge heads up to 100 m.

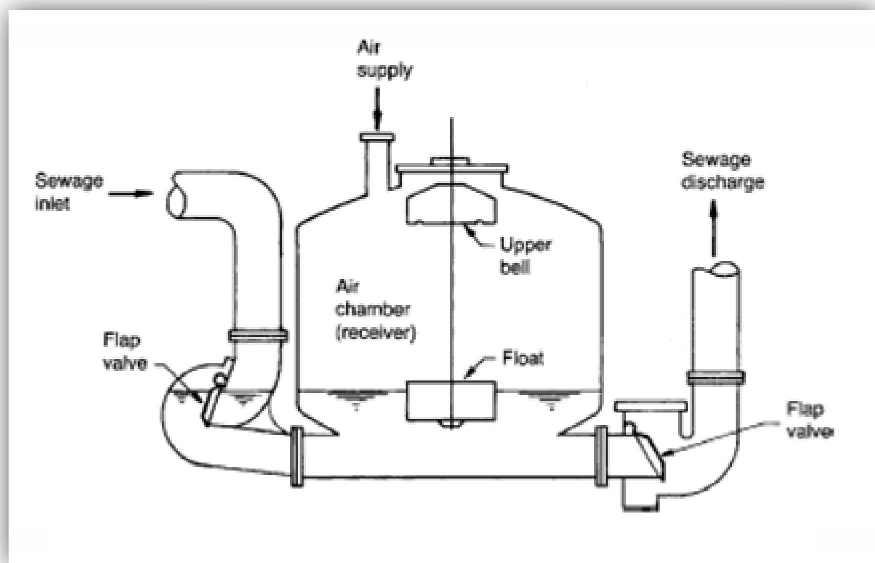


Figure 1.14 A pneumatic ejector. Adopted from Yeomans Chicago Corp [5]

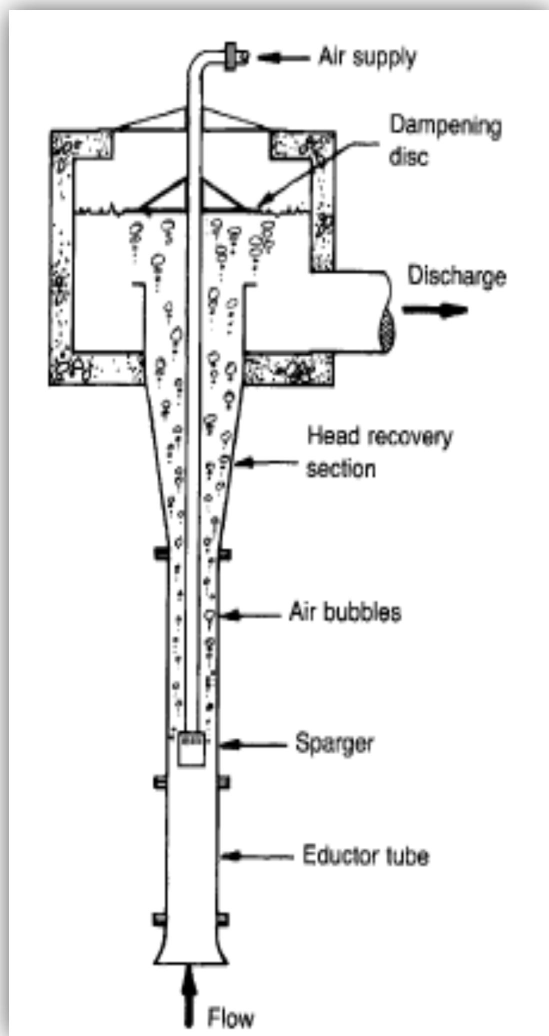
The liquid is admitted into an air chamber through a flap valve and a pressure vessel is allowed to fill by gravity until a predetermined level is reached. The compressed air is supplied from a plant air system or from close located compressors. Controls are then operated to admit compressed air to the vessel. The high pressure moves the liquid into the force main and when the chamber has been emptied, the controls close the air supply valve and vent the air in the tank to the atmosphere, which allows the next cycle to begin. Because of exposure to air the pneumatic ejector and air supply system are in danger of freezing, so low temperatures should be avoided.

[5], [10]

2.4.4.2 Air-lift pumps

Air lift pumps are pneumatic devices used for pumping sludge, contaminated liquid, large particles, sugar beets, hot or corrosive fluids, raw wastewater and sandy or dirty water.

The pump consists of a simple tube immersed in a sump or a wet well. A high volume of low-pressure compressed air is forced into the bottom of the tube submerged in the liquid to be pumped. This mixture of air and liquid is lighter than the surrounding liquid and therefore rises up the tube. The air-liquid mixture hits at the dampening disc and overflows into the open discharge channel. The reduced density of a column of an air–liquid mixture is used to raise the liquid in an air lift pump and the liquid is not pressurized. This is the reason why air lift pump is not regarded by many people as a pump but as a 'water-lifting device'.



ted from Walker

Such pumps are easy to maintain and operate. The pump itself is almost indestructible and, except for the splashing and the daily need for cleaning, requires virtually no maintenance. They are inexpensive and their construction is very simple; there are not any moving parts and sealing problems, the risk of blockage is small and they are not sensitive to temperature.

However, air lift pumps are relatively inefficient (usually 30 to 50%) and allow very little system flexibility. Process control is difficult because it is also not simple to regulate flow and the pump age changes erratically with small variations in the air delivered. Moreover compressed air is expensive; the use of compressors reduces the overall efficiency even more and air blowers do require maintenance. For all except very low heads, the air lift pump requires large submergence.

[1], [4], [5]

2.4.4.3 Pneumatic pump types

Pneumatic pumps are usually made of three main types: membrane-diaphragm pumps, bellows pumps and piston pumps.

In pneumatic bellows pumps, pumping action is performed by two bellows mounted on an alternatively moving shaft, in which one side of the bellows is in contact with the gas being pumped and the other side is in contact with the compressed air.

In pneumatic piston pumps the pumping action is performed by a piston connected to an alternatively moving shaft linked to a pneumatic engine.

A pneumatic diaphragm pump contains a single diaphragm or double diaphragms connected to a reciprocating shaft in which one side of the diaphragm is in contact with the gas being pumped and the other side is in contact with the compressed air.

Membrane pneumatic pumps are cheap however piston pneumatic pumps are more reliable and are more commonly used.

[31], [32]

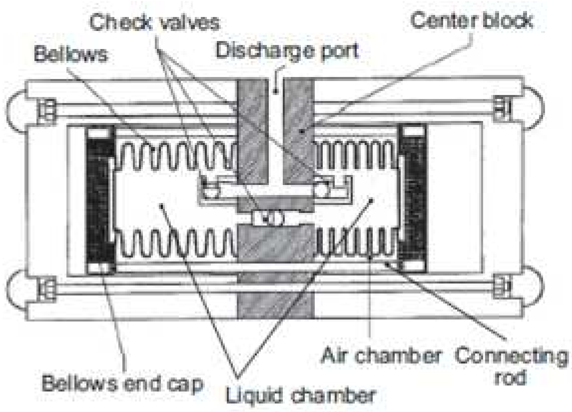


Figure 1.16 An air operated bellows pump configuration [32]

2.4.4.4 A pneumatic piston pump commercial example

A typical example of a pneumatically operated piston pump, used for commercial purposes, adopted from [33] is the one following.

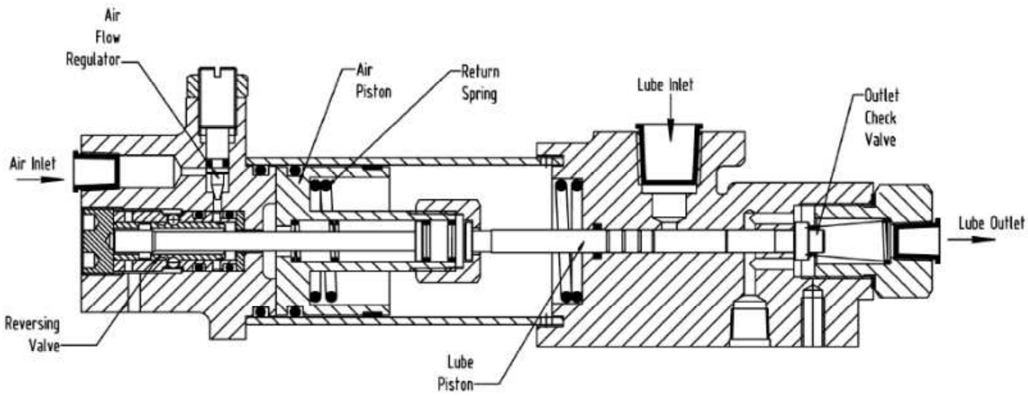


Figure 1.17 A pneumatic operated pump used in progressive lubricating systems [35]

This pump is intended for use in progressive lubricating systems. The air is supplied to the air inlet via a 2/2 solenoid valve. This valve when activated the pump starts its operation and if it is de-activated it stops the operation of the pump.

When the air is fed into the pump, the air piston starts its forward stroke, pushing the lube piston forward and ejecting grease through the outlet check valve and into the main lubrication line. As the piston nears the end of the forward stroke, it actuates the reversing valve, which dumps the air pressure through a vent hole on the underside of the pump. The piston is forced back by the return spring. In doing so, the metering chamber once more fills with grease. The outlet check valve prevents grease being sucked back from the outlet. The air piston returns to the original position, it closes the vent port, enabling the cycle to begin again. Outlet pressure is determined by the inlet air pressure and the fixed piston ratio. The speed of pump operation may be regulated by means of an air flow regulator.

[33], [34], [35]

Chapter 3

Presentation of our model and Implementation in Matlab Simulink®

Since some basic facts about pumps were introduced in the previous chapter, we are now able to deal with the presentation of our model and its implementation in Matlab Simulink. Firstly, we give some basic facts about differential and ordinary differential equations and what second order models are. Afterwards, we present our model, the two equations that characterize it and the values of the pump's parameters. There is a brief description about the tool we use to implement our model which the Matlab Simulink® environment, how we use ordinary differential equations in MATLAB Simulink® and what is the way we chose a Solver Type for the solution of ODE's. Then an example is given and what follows is the implementation of our model in Matlab simulink.

3.1 Introduction

Our goal is to create a mathematical model of our system which describes the function of a pneumatic pump. In this procedure the processing mechanism of the system has to be understood and be described in the form of some mathematical equations. These equations constitute the mathematical model of the system. The mathematical model of the system should describe accurately the input/output behavior of the system and be simple enough. Simplicity is very important because it makes the control task easier to understand and implement and it is more reliable for practical purposes.

Our model is a second order model and has two states. The first one is the line pressure and the other one is the pump stroke. The pressure under which the pump operates is the line pressure and the length of the pump stroke is the principal contributor to how much flow and pressure the pump produces. Each one of these two states is described by a first order ordinary differential equation.

In order to fully understand our model, some definitions should be given.

3.2 Ordinary differential equations and second order models

A differential equation (or DE) is an equation that contains an unknown function and one or more derivatives of it. An ordinary differential equation (or ODE) is an equation that involves functions of only one independent variable and one or more derivatives of this variable. The order of a differential equation is the order of the highest derivative that occurs in the equation. An example of a first order equation is

$\frac{dy}{dx} = ay$, where α is a constant.

Another notation used for the derivatives is $y' = \frac{dy}{dx}$.

Solving the differential equation means finding a function (or every such function) that satisfies the differential equation.

Second order models arise from systems that are modeled with two differential equations- two states. Second-order state determined systems are systems that two state variables describe them. There are many physical second-order models that contain two independent energy storage elements. These elements exchange stored energy, and may contain additional dissipative elements. Physical second-order models are often used to represent the exchange of energy between mass and stiffness elements in mechanical systems, between capacitors and inductors in electrical systems and between fluid inheritance and capacitance elements in hydraulic systems. Moreover second-order system models are frequently used to represent the exchange of energy between two independent energy storage elements in different energy domains coupled through a two-port element. Energy exchange for example, occurs between a mechanical mass and a fluid capacitance (tank) through a piston or between an electrical inductance and mechanical inertia as might occur in an electric motor.

Second-order models are used in the preliminary stages of design so as to establish the parameters of the energy storage and dissipation elements which are required in order to accomplish a satisfactory response. Second-order system responses depend on the dissipative elements in the system. Some systems are oscillatory and are characterized by

decaying, growing, or continuous oscillations, whereas some other second order systems' responses do not show oscillations.
[36], [37]

3.3 Model Presentation

As mentioned above, the model has two states the line pressure and the pump stroke.
Line pressure $P_l(t)$ is described by the following differential equation:

- $\dot{P}_l(t)=\dfrac{\beta}{V_l(Pl)}\left(K_l(\omega)x-Q(t)-Q_0-l(s)Q_\alpha(s,\Delta P_\alpha)\right) \quad , 0<\mathrm{P}_1\leq\mathrm{P}_{\max}$

  where β is the compressibility and

  $V_l=V_{l0}\left(1+\alpha_v P_l\right)$, V_{l0} is the line volume; α_v is the volume variation rate;

  $K_l(\omega)=k_l\omega$, k_l is the pump gain; ω [rad/s] is pump angular rate and Q_0 is the Lubricating flow, which is known and constant.

  $Q(t)$ describes the demanded flow. Q is series of trapezoidal profiles with slew rate $\mathrm{Q}_{\max}$ (increasing and decreasing) not greater than the table value which amounts to 2 dm^3/s, and max value lower than max pump flow in (1).

  $$l(s)=\begin{matrix}1, s<0\\0, s\geq 0\end{matrix}$$

  S and Q_a are presented below.

The pump stroke $x(t)$, $0\leq x\leq x_{\max}$ whose rate is proportional to pilot cylinder flow is described by the following differential equation:

- $\dot{x}(t)=\left(Q_a\left(s,\Delta P_\alpha(s)\right)-leakage\right)/A_\alpha$ [$m^3/s/m^2$]

$$Q_a\left(s,\Delta P_a\right)=\operatorname{sgn}\left(\Delta P_a(s)\right)s\left(\mu_1+\mu_2|s|\right)\sqrt{\left|\Delta P_a(s)\right|}$$

$s = \frac{-K_a x + \varphi I}{K_c + K_a}$ [m], where I is the driving current provided by a solenoid.

It should be remarked that fluid leakage is neglected (disturbance).

$s\left(\mu_1 + \mu_2 |s|\right)$ describes the aperture of a valve. However the second order term of the aperture μ_2 may be neglected μ_2 =0 though not negligible.

The pump stroke equations have an intrinsic feedback, whose error is the pilot valve stroke s [m].

When the error is zero (steady state), then $x = \frac{\varphi I}{K_a}$.

Multiplying the pump stroke x times the pump gain, the steady-state pump flow range is obtained since:

$$Q_l = \frac{\varphi I}{K_a} k_l \omega \cong 0 \div 2.1 \times 10^{-3}\ m^3/s = 0 \div Q_{l,\max} \quad (1)$$

Assuming the feedback as above, only one measurement is available: the line pressure.

$$y = P_l + e$$

The pressure drop ΔP a depends on all variables through a static equation switching with s

$$\Delta P_a(t) = \gamma_l P_l + \left(F_u - Kx(t)\right)/A_a - P_0 \text{ , } s \geq 0$$

$$\Delta P_a(t) = P_l - \gamma_l P_l - \left(F_u - Kx(t)\right)/A_a \text{ , } s < 0$$

The bias F_u and P_l must be such that the pressure drop is no negative.

$$F_u \geq P_0 A_a + K x_{\max} \cong 400N$$

$P_l \geq F_u / A_a / \left(1 - \gamma_l\right) \cong$ 1.0 MPa=P_{min}

Therefore the initial value of the pressure line must be greater than P_{min}. The initial value of the pump stroke should be zero: no flow.

It should also be remarked that starting from no flow, the line pressure decreases because of the lubricating flow: Q_0.

In the following table there are the values of the parameters used to describe our model.

No.	Type	Symbol	Unit	Value	
0	Line volume	V_{l0}	dm^3	0.25	
0bis	Compressibility	β	MPa	1200	Uncertainty 20%
0ter	Volume variation rate	α_v	1/ MPa	0.04	
1	Max pressure	P_{max}	MPa	25	=250 bar
2	Load flow slew rate	$\dot{Q}_{max}$	dm^3/s	2	$\pm$10% uncertain
3	Pump gain	k_l	m^2	0.6×10^{-3}	
4	Pump angular rate	ω	rad/s	200	
5	Lubricating flow	Q_0	dm^3/s	0.13	$\pm$10%
6	Pump current range	I	A	$0\div0.8$	
7	Piston area	A_a	dm^2	0.05	
8	Control time unit	T	ms	5	
9	Max pump stroke	x_{max}	mm	24	
10	Flow aperture coefficient	μ_1	$m^2/s/\sqrt{Pa}$	78×10^{-6}	Uncertain 20%
10bis	Idem	μ_2	$m^2/s/\sqrt{Pa}$	78×10^{-6}	Idem
11	Solenoid gain	ϕ	V_s	28	
12	Pilot valve spring stiffness	K_c	N/m	27300	
13	Feedback spring stiffness	K_α	N/m	1300	
14	Bias spring stiffness	K	N/m	8300	
15	Bias force	F_u	N	400	
16	Line pressure fraction	γ_l		0.2	
17	Bias pressure	P_0	MPa	0.4	
18	Min line pressure	P_{min}	MPa	1	

3.4 Model Implementation

3.4.1 MATLAB Simulink

The tool used to implement the model presented is MATLAB Simulink. It is an environment developed by MathWorks used for multidomain simulation and Model-Based Design for dynamic and embedded systems. It can be used in many areas to design, simulate, implement, and test a variety of time-varying systems, including communications, controls, signal processing, video processing, and image processing. Simulink provides an interactive graphical user interface environment and a customizable set of block libraries. It offers tight integration with the rest of the MATLAB environment and can either drive MATLAB or be scripted from it. Simulink can be used to explore the behavior of a wide range of real-world dynamic systems, including electrical circuits, shock absorbers, braking systems, and many other electrical, mechanical, and thermodynamic systems. Dynamic systems are systems whose outputs change over time and the way they are represented is by a set of differential equations in time. Simulating such a system with Simulink requires a user to create a block diagram using the Simulink model editor that graphically depicts time-dependent mathematical relationships among the system's inputs, states, and outputs and then command Simulink to simulate the system represented by the model from a specified start time to a specified stop time.

3.4.1.1 Matlab Simulink for ODE's

A mathematical model of a dynamic system is described by a set of equations. The mathematical equations described by a block diagram model are known as algebraic, differential, and/or difference equations. Simulink block diagrams use Integrator blocks to indicate integration and a chain of operator blocks connected to the integrator block to represent the method for computing the state's derivative. The chain of blocks connected to the Integrator's is the graphical counterpart to an ordinary differential equation (ODE). Integrating the states requires the use of numerical methods called ODE solvers. These various methods trade computational accuracy for computational workload. Simulink comes with computerized implementations of the most common ODE integration methods and allows a user to determine which it uses to integrate states represented by Integrator blocks when simulating a system. The Simulink product provides an extensive library of solvers,

each of which determines the time of the next simulation step and applies a numerical method to solve the set of ordinary differential equations that represent the model. In the process of solving this initial value problem, the solver also satisfies the accuracy requirements specified by the user.
[38]

3.4.1.2 Choosing a Solver Type

The Simulink library of solvers is divided into two types of solvers which are fixed-step and variable-step. The other categories they are subdivided into are discrete or continuous, explicit or implicit, one-step or multistep, and single-order or variable-order solvers.
The one used in our implementation is the ode45 solver. It is an explicit continuous variable-step solver. The variable-step solvers in the Simulink product dynamically vary the step size during the simulation. Each of these solvers increases or reduces the step size using its local error control to achieve the tolerances that you specify. Computing the step size at each time step adds to the computational overhead but can reduce the total number of steps, and the simulation time required to maintain a specified level of accuracy. The explicit variable-step solvers are designed for non stiff problems.
The ode45 solver is a one-step method of medium accuracy and the numerical method applied is the Runge-Kutta, Dormand-Prince pair. In general, the ode45 solver is the best to apply as a first try for most problems. For this reason, ode45 is the default solver for models with continuous states. This solver is a fifth-order method that performs a fourth-order estimate of the error. This solver also uses a fourth-order "free" interpolant, which allows for event location and smoother plots.
[38]

3.4.1.3 Example of how we solve differential equations in Simulink

Using Simulink to solve differential equations is very quick and easy. A simple example is the following:
We want to model the differential equation:

$$x'(t) = 5x(t) + u(t),$$

where u(t) is a square wave with an amplitude of 1 and a frequency of 1rad/sec.

What is needed is an Integrator block, a Gain block, a Sum block and a Signal Generator to generate the signal wave u (t).
The solution can be approximated in Simulink by using an integrator to integrate the first order derivative, x', to produce x.
The above differential equation is solved in Simulink as portrayed bellow:

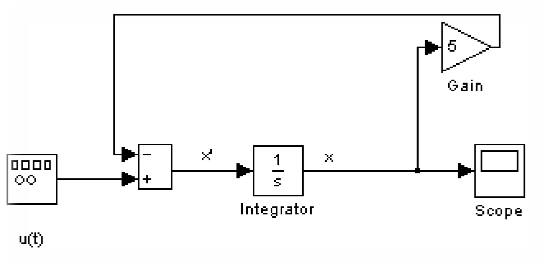


Figure 3.1 A Matlab Simulink example

3.4.2 Implementation of our model in Matlab Simulink

The pneumatic pump model presented above was implemented in Matlab Simulink environment in the way illustrated bellow:

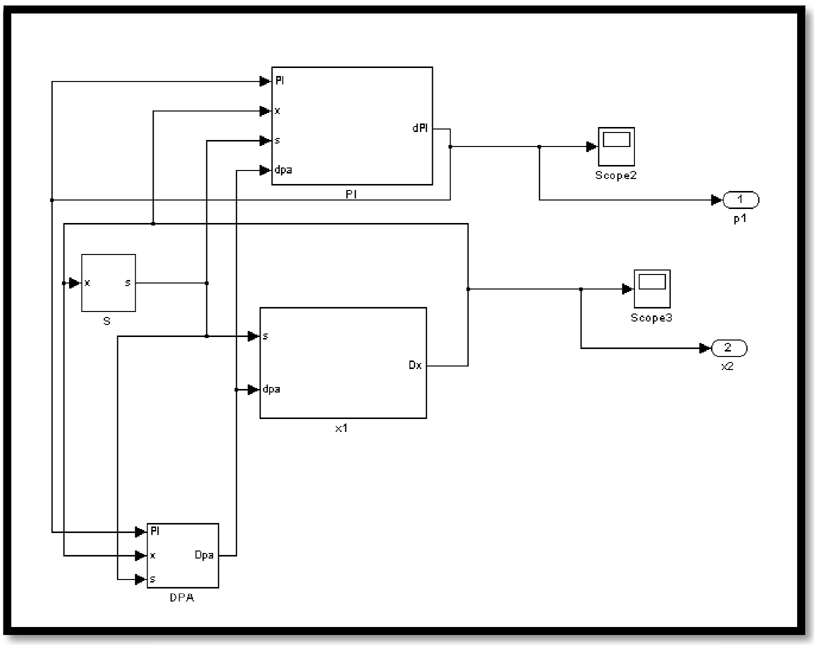


Figure 3.2 Pneumatic Pump model in Matlab Simulink

As mentioned above, our model consists of two ordinary differential equations (odes). These odes are:

$$\dot{P}_l(t)=\frac{\beta}{V_l(Pl)}\left(K_l(\omega)x-Q(t)-Q_0-l(s)Q_\alpha(s,\Delta P_\alpha)\right)$$

$$\dot{x}(t)=\left(Q_a\left(s,\Delta P_\alpha(s)\right)-leakage\right)\Big/A_\alpha$$

Using the matlab Simulink environment the first one is implemented as:

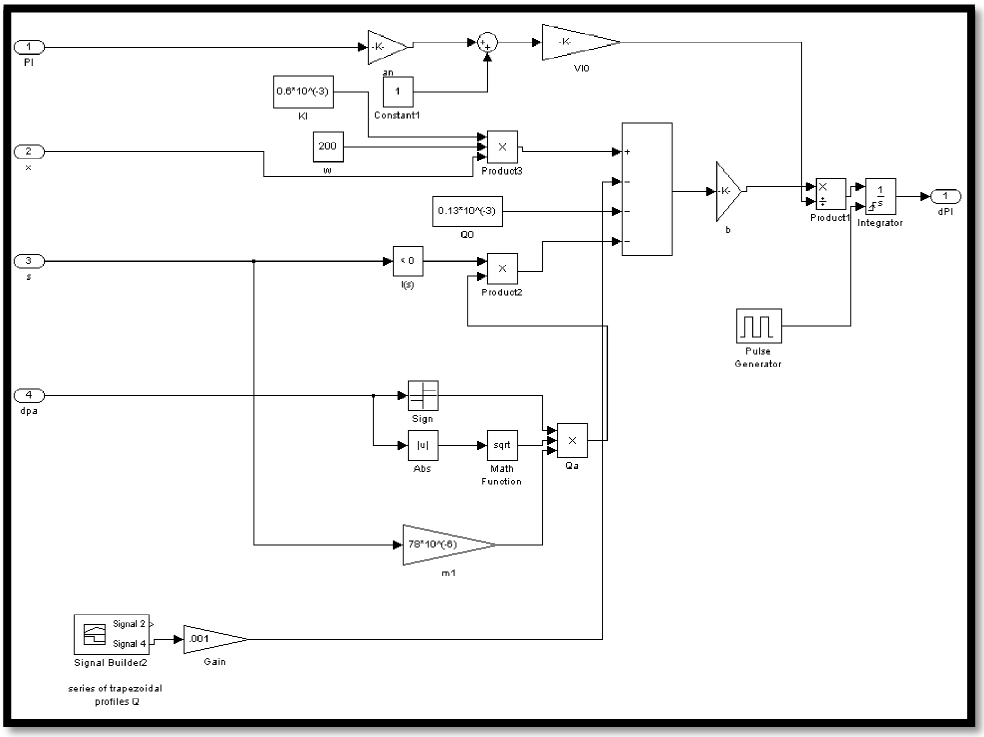


Figure 3.3 Line Pressure P_l implemented in Matlab Simulink

whereas the second one is:

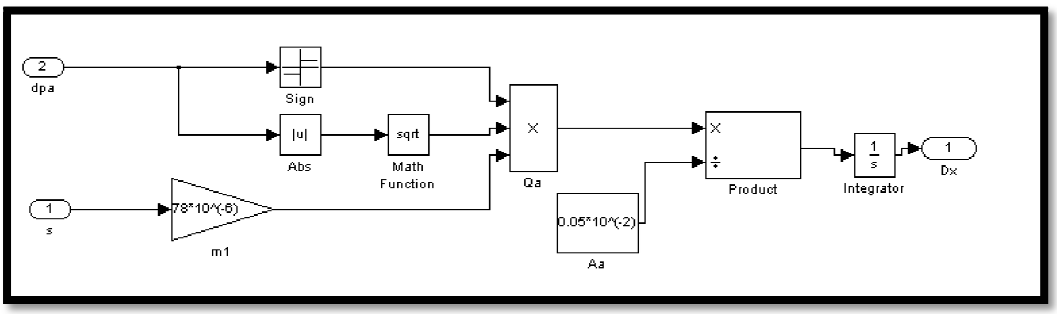


Figure 3.4 Pump stroke x implemented in Matlab Simulink

Where s and ΔP_a are implemented as:

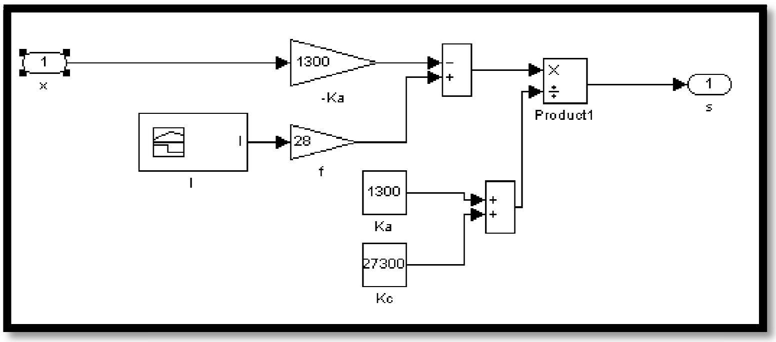


Figure 3.5 Pilot valve stroke s implemented in Matlab Simulink

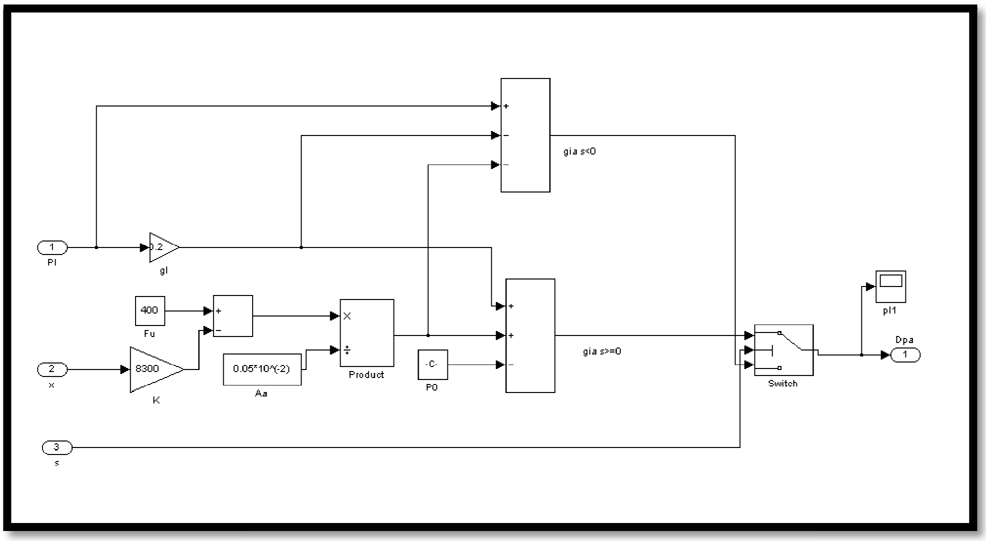


Figure 3.5 Pressure drop ΔPa implemented in Matlab Simulink

3.4.3 Simulation results of the implementation of our model

The results of the implementation for the model for the first 40 seconds are the following:

In this implementation the current increases from 0 to 0.8 for the first 15 seconds, it remains stable for 10 seconds and then it falls back to 9 for the next 15 seconds. We can notice that pump stroke increases, remains stable and then falls in proportion to the current. In regard with pressure line, it initially decreases due to the lubrication flow. During the upstroke it starts rising, it keeps increasing when being at max stroke and during the down-stroke it starts falling.

Pump stroke x is represented by the following graph:

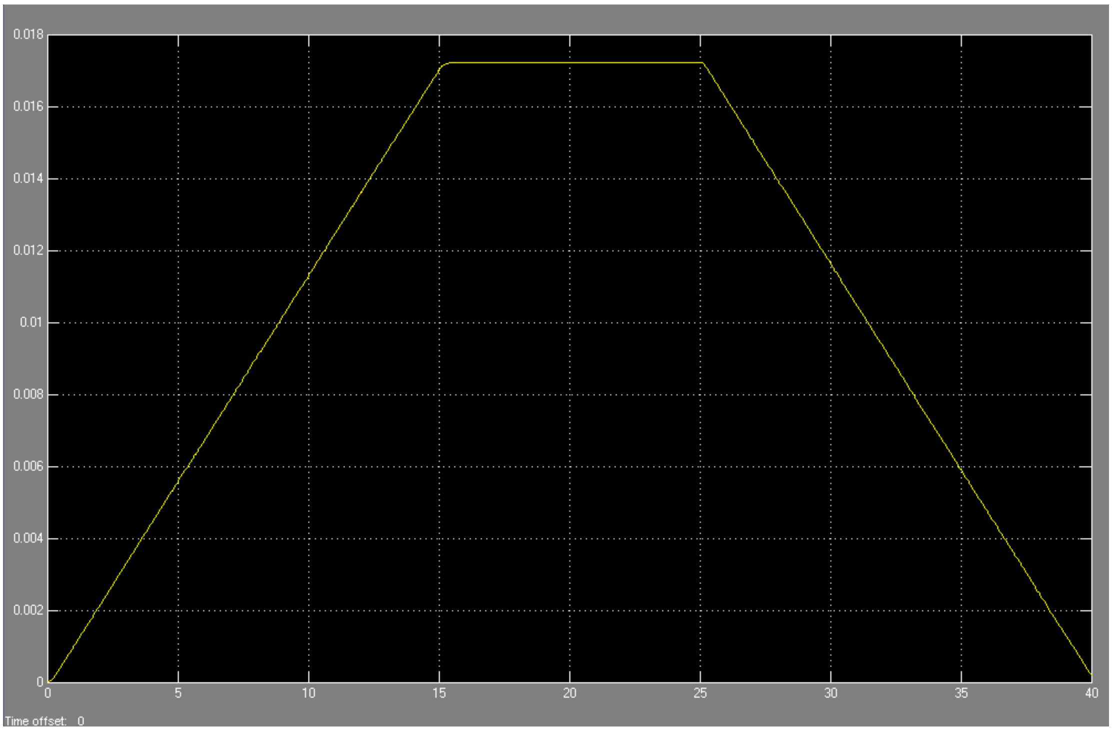


Figure 3.6 Pump stroke x

The graph bellow shows the line pressure:

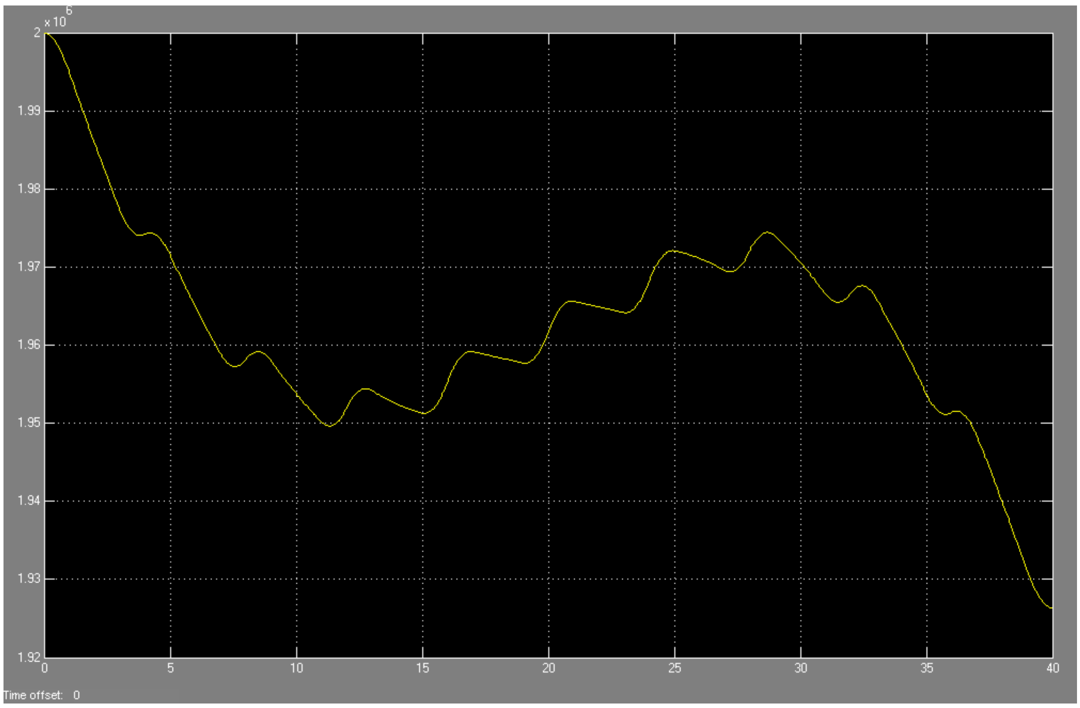


Figure 3.7 Line Pressure P_l

Pump stroke is initially 0 since there is no flow whereas line pressure must have a value greater than $P_{\min}$, which is $1\ MPa = 10^6\ Pa$ that is why it is initialized starting from 2 $MPa = 2 \cdot 10^6\ Pa$.

In steady state where $x = \dfrac{\varphi I}{K_a} = 17,2\ \ mm$, the steady-state pump flow range is obtained, since $Q_{l,\max} = 2.1 \times 10^{-3}\, m^3/s$, whereas the line pressure keeps rising in proportion with the demanded flow. When there is no current applied, the pump stroke is 0 mm and the line pressure keeps falling.

Chapter 4

System Identification using Recurrent High Order Neural Networks

In the previous chapter we examined the pneumatic pump model that we want to identify in this thesis. Now we can proceed to the presentation of the adaptive system identification scheme that we use for this process. We firstly introduce some basic facts about neural networks and how they can be used for control. Then, we present the RHONN model, three different identification techniques with or without modeling error and finally we use the RHONN model and system identification for the development of indirect adaptive control.

4.1 Introduction

The ever increasing technological advances have led to complex systems that include uncertain, and possibly unknown, nonlinearities, operating in highly uncertain environments, resulting in situations where regulatory and corrective influences should be exerted without complete knowledge of basic causes and effects. However the need for design and control decisions still exists, but the treatment of complex processes, attempts at complete understanding at a basic level may consume so much time and so large quantity of resources as to impede us in more immediate goals of control. For this reason, several "intelligent" techniques have been developed with the artificial neural networks being one of the most important tools that serve in this direction.

4.2 Artificial neural networks

Artificial neural networks are systems inspired by the study of biological neural networks. The main aim of a neural network is similar with a mathematical function, mapping an input to a desired output. They are used because the way they solve problems is much different than this followed by conventional computers. This is due to the fact that the information is processed the same way as in the human brain.

The biological nervous system is composed of millions of neurons which are simple electrically excitable cells that receive inputs from other neurons and send outputs to other neurons and cells. A typical neuron consists of a central cell body-also called soma, dendrites and axons. The dendrites are special nodes and fibers connected to the soma and structured as a tree branch. Their purpose is to send and receive electrochemical signals from other neurons. The neuron sends out spikes of electrical activity through the axon, a long thin straight structure that splits into thousands of branches at the end of which exists an electrochemical junction known as a synapse, through which neurons send signals to each other and to non-neuronal cells such as those in muscles or glands.

[40], [41], [42], [43], [44]

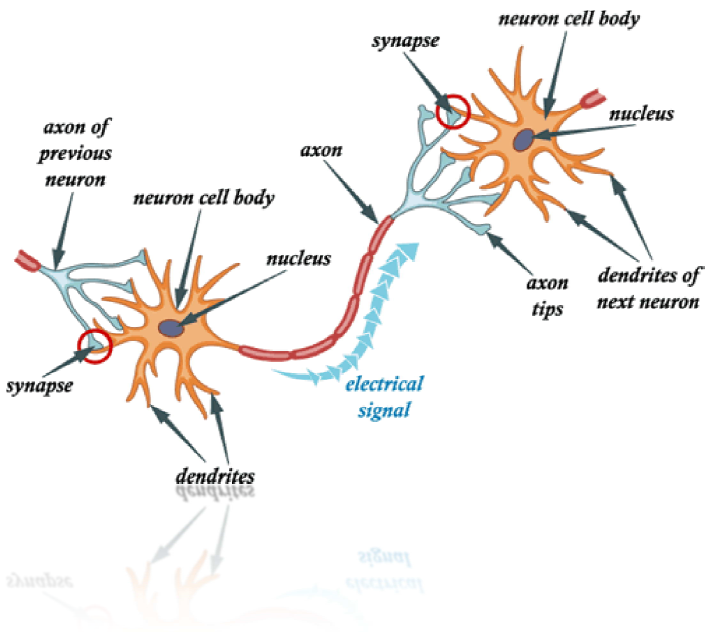


Figure 4.1 A typical neuron [39]

The first artificial neuron was produced in 1943 by the neurophysiologist Warren McCulloch and the logician Walter Pits. The artificial neural network consists of artificial neurons in correspondence with the biological neurons presented above, which are the fundamental processing elements of the neural network. In artificial neural networks a neuron is also called a node. Each node receives a set of numerical inputs (representing the one or more dendrites) from different sources, either from other neurons or from the environment, performs a calculation based on these inputs and produces a single output value. All inputs come in simultaneously or remain activated for the same amount of time until the computation of the output.

A model of a neuron is the one following. There is a set of connecting links between the input values and the neuron which represent the synapses. Each one of them is characterized by a weight which shows the strength of the synapse. The synaptic weight includes both positive and negative values,, therefore providing excitory or inhibitory influences to each input. An input signal of x_m connected to neuron k is multiplied by the synaptic weight x_{km}.

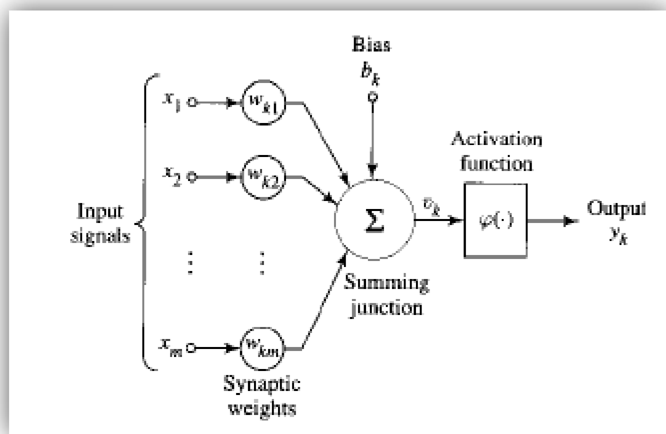


Figure 4.2 Non linear model of a neuron [40]

All the weighted inputs are added at the summing junction and the sum is passed through a non linear function known as activation function which is responsible for limiting the amplitude of the output of the neuron. The activation or transfer function is usually sigmoid, linear or threshold. The bias b_k raises or decreases the network input of the activation function in respect to whether it is positive or not. The output produced is directed either to the environment, or fed as input to other neurons in the network. When the output of an element in a dynamic system partially influences the input which is applied to same element then it is said that feedback exists.

There are many different types of neural networks that control systems use, but the decision of which type to use and what training procedure to follow depends on the intended application. The ways in which the neurons are structured, and the learning rules are used to train the network are intricately linked to each other and it is essential to study both of them.

[40], [41], [42], [43], [44]

4.2.1 Neural network architectures

The neurons are generally structured in three ways: single-layer feedforward networks, multilayer feedforward networks and recurrent networks.

In the single-layer feedforward networks there is an input layer of nodes which projects onto an output layer of neurons but not vice versa. This means that the information always move one direction and it never goes backwards. If the neural network was portrayed as a directed graph, the feedforward networks would be acyclic graphs.

In multilayer feedforward networks there one or more hidden layers. It consists of an input layer, an output layer of neurons and between them it is composed of one or more layers of neurons called hidden layers. The output signals of the second layer are the input signals of the third layer and so on is formed the whole network. The last layer outputs are the overall response of the network. The feedforward multilayer neural networks are the neural networks that are the most commonly used. The information is not fed back during operation but there is feedback information available during training. They can handle relatively complex tasks such as the known problem of exclusive or.

[40], [45], [43]

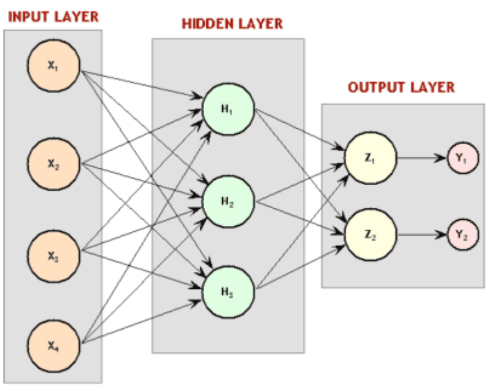


Figure 4.3 Multilayer feedforward neural networks' structure [46]

In recurrent neural networks the layer structure still exists, but it is different because recurrent neural networks consist of at least one feedback loop. The feedback loops have a great impact on the network performance and on its learning capability. They also involve the use of branches which is composed of unit delay elements and in case nonlinear units are contained, it has the outcome of nonlinear dynamical behavior.

[40], [43]

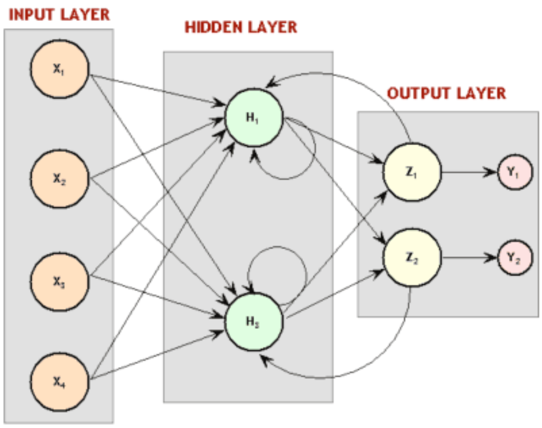


Figure 4.4 Recurrent neural networks' structure [46]

4.2.2 Learning process

What's the most important at a neural network is its ability to learn from its environment and improve its performance. Learning means, given a task to solve and a class of functions F, finding the optimal task solution using a set of observations. The optimal solution is the one that has the least cost. This means that a cost function should be determined to measure how far away from an optimal solution to the problem, a particular solution is. After the neural network is simulated by an environment, the learning procedure takes place through an interactive process of adjustments of the synaptic weights and bias level. The type of learning depends on the manner in which the free parameters of the neural network change, based on the above procedure. A set of rules used to solve a learning problem compose a learning algorithm. There are great deals of learning methods with the only difference the way in which the synaptic weights are adjusted.

The manner in which a neural network relates to its environment is also important and there are some techniques that refer to a model of the environment in which the neural network operates. They are called learning paradigms and they are basically classified into supervised

and unsupervised learning. Unsupervised learning is the learning where there are no labeled examples- patterns of the function to be learned from the network. There are given some data and the cost function and we want to minimize it but it can be any function of the data and the output of the network. In supervised learning the environment is unknown to the neural network but there are some input-output examples giving some knowledge about the environment. This means that we are given example pairs and we search the function that matches with these patterns. When the cost used is the mean-squared error and it is minimized using gradient descent for the class of multilayer perceptrons neural networks the backpropagation algorithm for training neural networks is obtained. Versions of this algorithm are most commonly used for training neural networks. However it takes a long time to converge so it is generally a slow and time consuming process.
[40], [43], [47]

4.2.3 Use of neural networks for adaptive control

The most significant problem in generalizing the application of neural networks in control is the fact that the very interesting simulation results that are provided, lack theoretical verification. Crucial properties like stability, convergence and robustness of the overall system must be developed and/or verified. The main reason for the existence of the above mentioned problem is the mathematical difficulties associated with nonlinear systems controlled by highly nonlinear neural network controllers. The problem of controlling an unknown nonlinear dynamical system has been attacked from various angles using both direct and indirect adaptive control structures and employing different neural network models.

Adaptive controllers do not need any knowledge of the system parameters and they are adapted to parameter uncertainties by using performance error information on-line, in contrast with other control methods such as PID, pole placement, optimal, robust, or nonlinear control that the system parameters must be known. A typical adaptive control system consists of a plant, a controller with parameters, and an adaptive law to update the controller parameters to achieve some desired system performance.

Plant is the process to be controlled. It has a certain number of inputs that are processed in order to produce a number of outputs. These outputs are the dependent variables we want to control and represent the measured output response of the plant.

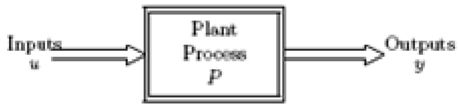


Figure 4.5 Plant representation [48]

[48]

It can be single-input single-output or multi-input multi-output, continuous or discrete, linear or nonlinear.

In adaptive control, the plant parameters are unknown and we often use discrete samples of the plant inputs and outputs for the neural network training. An adaptive controller is formed by combining an online parameter estimator-also known as adaptive law, which provides estimates of unknown parameters at each instant, with a control law that is motivated from the known parameter case. The way the parameter adaptive law is combined with the control law gives rise to two different approaches, the direct and the indirect adaptive control. In indirect adaptive control, the plant parameters are estimated on-line and then they are used to calculate the controller parameters. In direct adaptive control the plant model is parameterized in terms of the controller parameters that are estimated directly without intermediate calculations involving plant parameter estimates.

The use of neural networks for indirect control involves two steps: system identification and control design. System identification is the procedure where a neural network model of the plant we want to control is produced, whereas control design is the procedure where the neural network plant model is used to design or train the controller.

[47], [48], [49]

4.3 System identification

System identification is of a great importance in order to understand and predict the behavior of the system, but also to obtain an effective control law. Neural networks support identification of systems with any number of input and output signals.

They can be trained to identify nonlinear dynamical systems via forward and inverse modeling.

Forward modeling is the procedure of training a neural network to represent the forward dynamics of the plant.

The neural network model is placed in parallel with the plant as shown below. The training signal of the network is the prediction error, which is the error between the plant output and the neural network output.

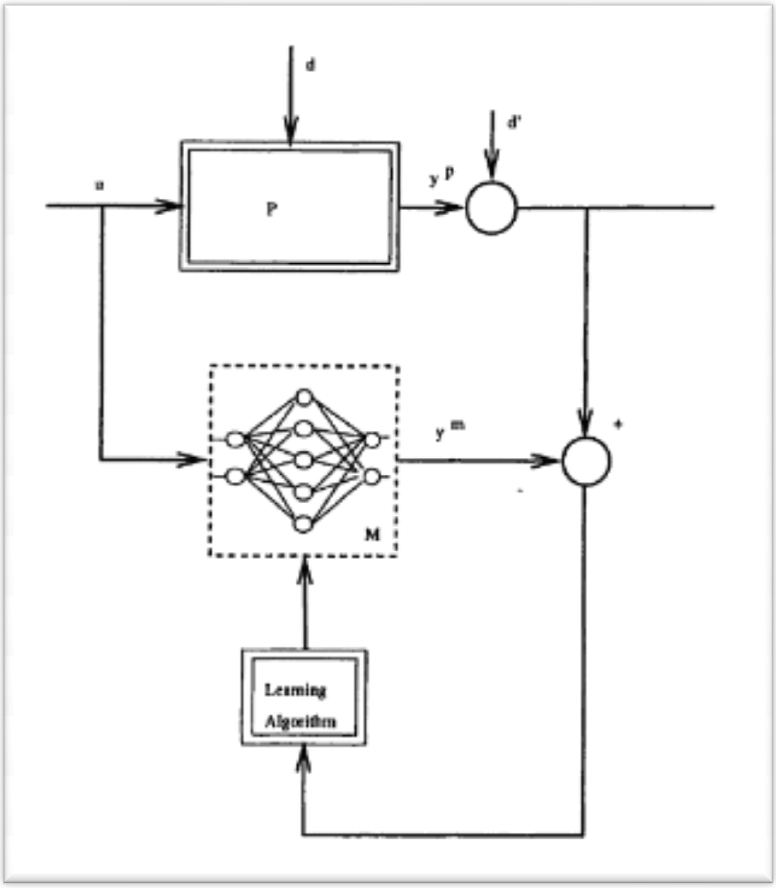


Figure 4.6 Forward modeling identification [50]

We assume that the plant is governed by the nonlinear difference equation that follows:

$$y^p(k+1) = f\left(y^p(k),...,y^p(k-n+1);u(k),...,u(k-m+1)\right)$$

The plant output y^p at time k+1 depends on the past n output values, in the sense defined by the nonlinear map f. The plant output also depends on the past m values of the input u.

An obvious approach for system modeling is to choose the input-output structure of the neural network to be the same as that of the system, as shown at the nonlinear difference equation that follows:

$$y^m(k+1) = f_{apr}\left(y^p(k),\ldots,y^p(k-n+1);u(k),\ldots,u(k-m+1)\right)$$

Where y^m is the network's output, f_{apr} represents the nonlinear input output map of the network which is the approximation of f.

It is obvious that the input to the network includes the past values of the real system output, hence, the system has no feedback. Assuming that after a certain training period the network gives a good representation of the plant, that is $y^m \approx y^p$, then for subsequent post-training purposes the network output together with its delay values can be fed back and used as part of the network input. In this way, the network can be used independently of the plant. Such a network model is described by:

$$y^m(k+1) = f_{apr}\left(y^m(k),\ldots,y^m(k-n+1);u(k),\ldots,u(k-m+1)\right)$$

[47], [50]

Inverse modeling is the procedure of training a neural network to represent the inverse dynamics of the plant. In inverse modeling the plant's input is the desired neural network's output and the plant's output is used as input to the neural network. The neural network's output is compared with the actual input of the plant and their error difference is to be minimized and is used to train the neural network. The desired output of the neural network is the current input to the plant.

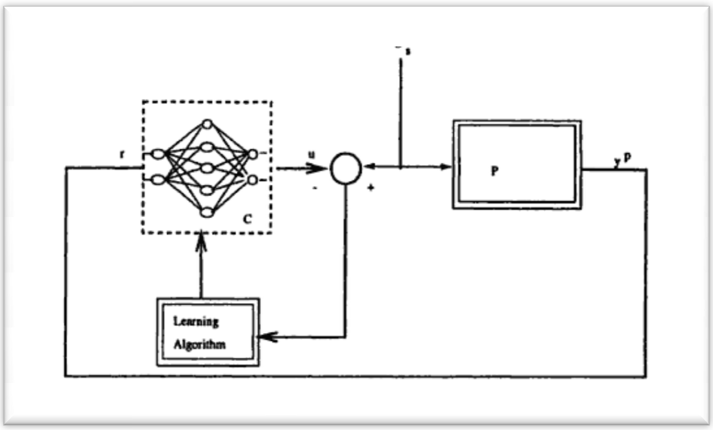


Figure 4.7 Inverse modeling identification [50]

During inverse modeling, the assumption made is that the neural network can approximate the inverse of the plant well. This obviously means that the inverse exists and it is unique and stable. If the inverse is not unique then the ranges of the input to the network should be given some special attention.

[47], [50]

4.4.1 System identification of dynamic systems

It is a fact that the interest towards the usage of neural networks for the modeling and identification of dynamical systems has increased. In order that neural network architecture be able to approximate the behavior of a dynamical system in some sense, it is clear that it should contain dynamic elements in the form of feedback connections. These networks, as mentioned in previous subsections, are called recurrent neural networks.

A dynamic network can also be made from a static one, if the past neural outputs are passed as inputs to the network. In order to avoid making the neural network a very complicated and highly nonlinear dynamic system, a more efficient way to introduce dynamics with the aid of feedforward multilayer neural networks was proposed in [49]. In this technique static multilayer neural networks are connected with stable linear dynamical systems, in a way which combines both linear, parallel and feedback connections and the synaptic weights are adjusted according to a gradient descent rule. However, by this way some serious issues occur. The main one is that the synaptic weights appear nonlinearly in the mathematical representation that governs their evolution, resulting to a number of significant drawbacks. The learning laws used, require a high amount of computational time and the fact that the synaptic weights appear nonlinearly has the outcome of the functional possessing many local minima so there is no way to ensure the convergence of the weights to the global minimum. Moreover, the highly nonlinear nature of the neural network architecture fails to verify basic properties like stability, convergence and robustness.

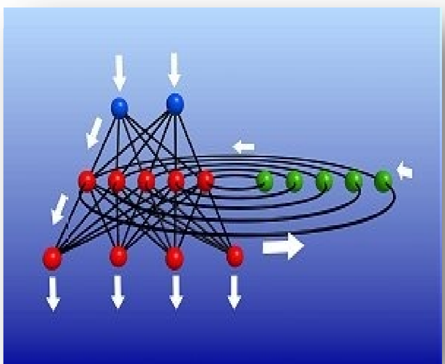
In contrast, recurrent networks possess a linear-in-the-weights property that makes the issues of proving stability and convergence feasible and their incorporation into a control loop promising. There have been proposed a lot of training methods for recurrent networks in literature, most of which rely on the gradient methodology and involve the computation of partial derivatives, or sensitivity functions. In this respect, they are extensions of the backpropagation algorithm for feedforward neural networks [68]. Although these methods have been used successfully in many empirical studies, they share some fundamental drawbacks since they require a great deal of computational time, the computation of the partial derivative is done approximately and they are unable to obtain analytical results concerning the convergence and stability properties.

In our system of view here, the RHONN model, we deal with the identification problem which consists of choosing an appropriate identification model and adjusting its parameters according to some adaptive law, such that the response of the RHONN model to an input signal (or a class of input signals), approximates the response of the real system to the same input. We combine distributed recurrent networks with high-order connections between neurons. The idea of recurrent neural networks with dynamical components distributes throughout the network in the form dynamical neurons and their application for identification of dynamical systems was proposed in [62]. Moreover high-order networks are used because of their superior storage capacity (demonstrated in [65], [52]) and their stability properties of these models for fixed-weight values (studied in [54], [61]). These networks allow higher-order interactions between neurons and they are expansions of the first-order Hopfield [59] and Cohen-Grossberg [53] models.

[47]

4.4.1.1 THE RHONN MODEL

The RHONN model uses recurrent high order neural networks. Recurrent as also mentioned before are neural networks with feedback connections. They were first designed for pattern recognition problems and they have been used as associative memories and for the solution of optimization problems. When a new input is applied to the network, the output is calculated and then it is fed back to the network in order to modify the input. These feedback connections provide the system with the dynamical behavior needed so as the neural networks can be used for the modeling and the identification of dynamical systems.



rk characterized

In the simplest case, the state history of each neuron can be described by a differential equation of the form:

$$\dot{x}_i = -a_i x_i + b_i \sum_j w_{ij} y_j$$

Where x_i is the state of the i[th] neuron, a_i, b_i are constants, w_{ij} is the synaptic weight connecting the j[th] input to the i[th] neuron. Each y_j is either an external input or the state of a neuron passed through a sigmoid function.

In a recurrent second order neural network, the input to the neuron is not only a linear combination of the components y_j but also of their product $y_j y_k$. Higher order interactions are represented by triplets $y_j y_k y_l$, quadruplets $y_j y_k y_l y_m$, etc. forming the recurrent high order neural networks.

Supposing we have a recurrent high order neural network which is composed of n neurons and m inputs. Then the state of each neuron would be described by a differential equation of the form:

$$\dot{x}_i = -a_i x_i + b_i \left[\sum_{k=1}^{L} w_{ik} \prod_{j \in I_k} y_j^{\,d_j(k)} \right]$$

Where $\{I_1, I_2, \ldots, I_L\}$ is a collection of L not-ordered subsets of $\{1, 2, \ldots, m+n\}$, w_{ik} are the adjustable synaptic weights of the neural network, a_i, b_i are real coefficients and $d_j(k)$ are non-negative integers. The state of the i[th] neuron is again represented by x_i and the input vector to each neuron $y = [y_1, y_2, \ldots, y_{m+n}]^T$ is defined by:

$$y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \\ y_{n+1} \\ \vdots \\ y_{n+m} \end{bmatrix} = \begin{bmatrix} s(x_1) \\ s(x_2) \\ \vdots \\ s(x_n) \\ u_1 \\ u_2 \\ \vdots \\ u_m \end{bmatrix}$$

where $u = [u_1, u_2, \ldots, u_m]^T$ is the external input vector to the network.

The function $s(.)$ represents the sigmoid function which is monotone-increasing, differentiable and is usually represented by sigmoids of the form:

$$s(x) = \frac{a}{1+e^{-\beta x}} - \gamma \quad ,$$

where the parameters a, β represent the bound and slope of sigmoid curvature and γ is a bias constant. The sigmoid activation functions most commonly used in neural network applications are the logistic and the hyperbolic tangent function. According to the above type for the sigmoids, in the special case where $a = \beta = 1,\ \gamma = 0$ we obtain the logistic function $s(x) = \frac{1}{1+e^{-x}}$ and by setting $a = \beta = 2, \gamma = 1$ the hyperbolic tangent function $s(x) = \frac{e^{2x}-1}{e^{2x}+1}$ is obtained.

Using the L-dimensional vector $z = \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_L \end{bmatrix} = \begin{bmatrix} \prod_{j \in I_1} y_j^{d_j(1)} \\ \prod_{j \in I_2} y_j^{d_j(2)} \\ \vdots \\ \prod_{j \in I_L} y_j^{d_j(L)} \end{bmatrix}$ **(4.1)**

the RHONN model takes the form:

$$\dot{x}_i = -a_i x_i + b_i \left[\sum_{k=1}^{L} w_{ik} z_k \right]$$

And if we define the adjustable parameter vector as

$w_i = b_i \left[w_{i1} w_{i2} \cdots w_{iL} \right]^T$, our model becomes:

$\dot{x}_i = -a_i x_i + w_i^T z$. **(4.2)**

The vectors $\{w_i : i = 1, 2, \ldots n\}$ represent the adjustable weights of the network and the coefficients $\{a_i : i = 1, 2, \ldots n\}$ are part of the underlying network architecture and are fixed during training.

In order to guarantee that each neuron x_i is bounded-input bounded-output (BIBO) stable, we shall assume that $a_i > 0,\ \forall i = 1,2,\ldots,n$. In the special case of a continuous time Hopfield model [59], we have $a_i = \frac{1}{R_i C_i}$, where $R_i > 0$ and $C_i > 0$ are the resistance and capacitance connected at the i^th^ node of the network respectively.

The above model in (4.2) can be described with vectors if we set:

$x = [x_1, x_2, \cdots, x_n]^T \in \mathbb{R}^n$

$W = [w_1, w_2, \cdots, w_n]^T \in \mathbb{R}^{L\times n}$

$A = diag\{-a_1, -a_2, \cdots, -a_n\}$, a $n \times n$ diagonal stability matrix since $a_i > 0$ $\forall i = 1,2,\ldots,n$.

The vector z is a function of both the neural network state x and the external input u.

Then the dynamic behavior of the overall network would be formed as:

$\dot{x} = Ax + W^T z$.

The reason for using high order neural networks is because if a sufficient number of connections exists in the RHONN model then it is possible to approximate any dynamical system to any degree of accuracy. This can be shown if we consider the problem of approximating a general nonlinear dynamical system with input-output behavior given by:

$$\dot{\chi} = F(\chi, u) \quad \textbf{(4.3)},$$

where $\chi \in \mathbb{R}^n$ is the system state, $u \in \mathbb{R}^m$ is the system input and $F : \mathbb{R}^{n+m} \to \mathbb{R}^n$ is a smooth vector field defined on a compact set $y \subset \mathbb{R}^{n+m}$.

The approximation problem consists of determining whether by allowing enough higher-order connections, there exist weights W, such that the RHONN model approximates the input-output behavior of an arbitrary dynamical system of the form (4.3).

We assume that F is continuous and satisfies a local Lipschitz condition such that (4.3) has a unique solution- in the sense of Caratheodory [57] and $(\chi(t), u(t)) \in Y$ for all t in some time interval $J_T = \{t : 0 \le t \le T\}$. The interval J_T is the time period over which the approximation is to be performed. Based on the above assumptions the following result is obtained.

Supposing that we have the system $\dot{\chi} = F(\chi, u)$ and the model $\dot{x} = Ax + W^T z$ and initially they are at the same state $x(0) = \chi(0)$, then for any $\varepsilon > 0$ and any finite $T > 0$, there exists an integer L and a matrix $W^* \in \mathbb{R}^{L \times n}$ such that the state $x(t)$ of the RHONN model with L high-order connections and weight values $W = W^*$ satisfies $\sup_{0 \le t \le T} \left| x(t) - \chi(t) \right| \le \varepsilon$.

From this Theorem we can conclude that if enough higher-order connections are allowed in the network then there exist weight values such that the input-output behavior of the RHONN model approximates that of an arbitrary dynamical system whose state trajectory remains in a compact set. This is strictly an existence result as it does not provide any constructive method for obtaining the optimal weights W^*.

In what follows, we consider the learning problem of adjusting the weights adaptively, such that the RHONN model identifies general dynamic systems.

[47]

4.4.1.2 Adaptive system Identification using RHONN

In this section we develop weight adjustment laws under the assumption that the unknown system is modeled exactly by RHONN architecture of the form $\dot{x} = Ax + W^T z$.

Firstly we suppose that there is no modeling error because the analysis is easier and the techniques developed in this case are of great importance for the design of weight adaptive laws in the presence of modeling errors for $t \ge 0$.

There exist unknown weight vectors w_i^*, $i = 1, 2, \ldots n$ such that each state χ_i of the unknown dynamic system $\dot{\chi} = F(\chi, u)$ satisfies:

$$\dot{\chi}_i = -a_i \chi_i + w_i^* z(\chi, u), \quad \chi_i(0) = \chi_i^0.$$

χ_i^0 is the initial i[th] state of the system. The arguments of the vector field z are omitted and the input u(t) and the state χ(t), as is standard in system identification procedures, remain bounded for all $t \geq 0$.

Based on the definition of $z(\chi,u)$ as given by (4.1),

where $z = \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_L \end{bmatrix} = \begin{bmatrix} \prod_{j\in I_1} y_j^{d_j(1)} \\ \prod_{j\in I_2} y_j^{d_j(2)} \\ \vdots \\ \prod_{j\in I_L} y_j^{d_j(L)} \end{bmatrix}$,

this implies that $z(\chi,u)$ is also bounded. With these assumptions we present two different approaches for estimating the unknown parameters w_i^* of the RHONN model. These are the filtered-regressor and the filtered-error RHONN identifiers.

[47]

4.4.1.3 Filtered Regressor RHONN

Using the lemma:

The system described by $\dot{\chi}_i = -a_i\chi_i + w_i^* z(\chi,u)$, $\chi_i(0) = \chi_i^0$,

can be expressed as:

$\chi_i = w_i^{*^T}\zeta_i + e^{-\alpha_i t}\chi_i^0, \quad \dot{\zeta}_i = -\alpha_i\zeta_i + z, \;\; \zeta_i(0) = 0$,

the dynamical system $\dot{\chi} = F(\chi,u)$ can be described as

$\chi_i = w_i^{*^T}\zeta_i + e_i, \quad i = 1,2,...,n$. **(4.4)**

Where ζ_i is a filtered version of the vector z as described by 4.1 and $e_i := e^{a_i t}\chi_i^0$ is an exponentially decaying term which appears if the system is in a nonzero initial state.

By replacing the unknown weight vector w_i^* in the 4.1, by its estimate w_i and ignoring the exponentially decaying term e_i, we obtain the RHONN model:

$x_i = w_i^T \zeta_i,\ i = 1, 2, \ldots n$.

The exponentially decaying term $e_i(t)$ can be omitted, since it does not affect the convergence properties of the scheme. The state error $e_i = x_i - \chi_i$ between the system and the model satisfies:

$e_i = \varphi_i^{\mathrm{T}} \zeta_i - e_i$, where $\varphi_i = w_i - w_i^*$ is the weight estimation error.

The problem now is to derive suitable adaptive laws for adjusting the weights w_i for $i = 1, 2, \ldots, n$. This can be achieved by using well-known optimization techniques for minimization of the quadratic cost functional:

$$J(w_1, \ldots, w_n) = \frac{1}{2} \sum_{i=1}^{n} [(w_i - w_i^*)^T \zeta_i - e_i]^2 \cdot$$

Depending on the optimization method that is employed, different weight adjustment laws can be derived. Here two methods are considered; the gradient and the least square method [45].

The gradient method yields $\dot{w}_i = -\Gamma_i \zeta_i e_i, i = 1, 2, \ldots, n$,

where Γ_i is a positive definite matrix referred to as the adaptive gain or learning rate.

With the least square method we obtain

$$\begin{cases} \dot{w}_i = -P_i \zeta_i e_i \\ \dot{P}_i = -Pi \zeta_i \zeta_i^T P \end{cases},\quad i = 1, 2, \ldots, n,$$

Where P(0) is a symmetric positive definite matrix. In this formulation, the least squares algorithm can be thought of as a gradient algorithm with a time varying learning rate.

The stability and convergence properties (which are of great importance) of these two weight adjustment laws are well-known in the adaptive control literature (for example [55], [58]).

The filtered regressor RHONN model relies on filtering the vector z. This vector is sometimes referred to as the regressor vector. The use of this filtering technique could make it possible to obtain a very simple algebraic expression for the error. This allows the application of well-known optimization procedures for designing and analyzing weight adjustment laws.

However, in this method there is a significant drawback. The filtering of the regressor requires complex configuration and heavy computational demands. Moreover, the dimension of the regressor will be larger than the dimension of the system and the employment of so many filters might be very expensive computationally.

[47]

4.4.1.4 Filtered error RHONN

In developing this identification scheme we start again from the differential equation that describes the unknown system which is:

$$\dot{\chi}_i = -a_i \chi_i + w_i^{*T} z \ , \ i = 1, 2, ..., n$$

Based on the above differential equation for the unknown system the identifier is now chosen as:

$$\dot{x}_i = -a_i x_i + w_i^T z \ , \quad i = 1, 2, ..., n \quad , \textbf{(4.5)}$$

where w_i is again the estimate of the unknown vector w_i^*.

In this case the state error: $e_i := x_i - \chi_i$ satisfies:

$$\dot{e}_i = -a_i e_i + \varphi_i^T z \ , \ i = 1, 2, ..., n \ ,$$

where $\varphi_i = w_i - w_i^*$.

The weights w_i, for $i = 1, 2, ..., n$, are adjusted according to the learning law:

$$\dot{w}_i = -\Gamma_i z e_i \ ,$$

where the adaptive gain Γ_i is a positive definite LxL matrix. In the special case that $\Gamma_i = \gamma_i \mathrm{I}$, where $\gamma_i > 0$ is a scalar, then it can be replaced by γ_i.

This identification scheme has similar convergence properties as the filtered regressor RHONN model with the gradient method for adjusting the weights. This is showed by the theorem that follows:

If we consider the filtered error RHONN model given above, whose weights are adjusted according to $\dot{w}_i = -\Gamma_i z e_i$. Then for $i = 1,2,...,n$ a) e_i, $\varphi_i \in L_\infty$ b) $\lim_{t\to\infty} e_i(t) = 0$.

The filtered-error RHONN model has a simpler structure than the filtered-regressor, since less filters are required and hence, fewer computations.

[47]

4.4.1.5 Robust learning algorithm

In the previous two learning algorithms, filtered regressor and filtered error, there is made the assumption that there is no modeling error. Equivalently, it is assumed that there exist weight vectors w_i^*, for $i = 1,2,...,n$, such that each state of the unknown system $\dot{\chi} = F(\chi, u)$ satisfies:

$\dot{\chi}_i = -a_i \chi_i + w_i^{*T} z(\chi, u)$. **(4.6)**

However in many cases this assumption will be violated. This is mainly due to the fact that in the RHONN model there are an insufficient number of higher order terms. In such cases, if standard adaptive laws are used for updating the weights, then the presence of the modeling error in problems related to learning in dynamic environments may cause the adjusted weight values and consequently, the error $e_i = x_i - \chi_i$ to drift to identify. In order to avoid the parameter drift phenomenon we modify the standard weight adjustment laws. These modified weight adjustment laws are referred to as robust learning algorithms.

This identification scheme is the one we use in this thesis to identify the pneumatic pump model presented in chapter 3.

In formulating the problem it is noted that by adding and subtracting $\alpha_i \chi_i + w_i^{*T} z(\chi, u)$, the dynamic behavior of each state of the system $\dot{\chi} = F(\chi, u)$ in (4.3) can be expressed by a differential equation of the form:

$\dot{\chi}_i = -\alpha_i \chi_i + w^{*T}{}_i z(\chi, u) + v_i(t)$.**(4.7)**

The modeling error $v_i(t)$ is given by:

$$v_i(t) := F_i(\chi(t), u(t)) + a_i \chi(t) - w_i^{*T} z(\chi(t), u(t)).$$

The function $F_i(\chi,u)$ denotes the i[th] component of the vector field $F(\chi, u)$, while the unknown optimal weight vector w_i^* is defined as the value of the weight vector w_i that minimizes the L_∞-norm difference between $F_i(\chi,u)+a_i\chi$ and $w_i^T z(\chi,u)$ for all

$(\chi,u) \in \Upsilon \subset \square^{n+m}$, subject to the constraint that $|w_i| \le M_i$, where M_i is a large design constant. The region Y denotes the smallest compact subset of $\square^{n+m}$ that includes all the values that (χ,u) can take. i.e., $(\chi(t),u(t)) \in Y$ for all $t \ge 0$.

Since by assumption $\mathrm{u}(\mathrm{t})$ is uniformly bounded and the dynamical system to be identified is BIBO stable, the existence of such Y is assured. It is pointed out that in our analysis we do not require knowledge of the region Y, nor upper bounds for the modeling error $v_i(t)$.

In summary, for $i = 1, 2, ..., n$ the optimal weight vector w_i^* is defined:

$$w_i^* := \arg \min_{|w_i| \le M_i} \left\{ \sup_{(\chi,u) \in Y} \left| F_i(\chi,u) + a_i \chi - w_i^T z(\chi,u) \right| \right\}.$$

The reason for restricting w_i^* to a ball of radius M_i is firstly to avoid any numerical problems that may arise owing to having weight values that are too large, and secondly, to allow the use of the σ-modification [60], which will be developed below to handle the parameter drift problem.

The formulation developed above follows the methodology of [66] closely. Using this formulation, we now have a system of the form: $\dot{\chi}_i = -\alpha_i \chi_i + w^{*T}{}_i z(\chi, u) + v_i(t)$ (4.7) instead of the system of the form (2.6).

It should be outlined that since $\chi(t)$ and $u(t)$ are bounded, the modeling error $v_i(t)$ is also bounded, i.e., $\sup_{t\geq 0} |v_i(t)| \leq \overline{v}_i$ for some finite constant $\overline{v}_i$.

In what follows, we are based on the filtered error RHONN identifier for the development of robust learning algorithms. This doesn't mean that filtered filtered-regressor RHONN couldn't be used instead. The same underlying idea can be extended readily to the filtered-regressor RHONN.

So, the identifier is chosen as in (4.5),

i.e. $\dot{x}_i = -a_i x_i + w_i^T z$, $i = 1, 2, ..., n$, **(4.8)**

where w_i is the estimate of the unknown optimal weight vector w_i^*.

Using the equations in 4.7 and 4.8, the state error $e_i = x_i - \chi_i$ satisfies $\dot{e}_i = -a_i e_i + \varphi_i^T z - v_i$, where $\varphi_i = w_i - w_i^*$.

Owing to the presence of the modeling error v_i, the learning laws given by $\dot{w}_i = -\Gamma_i z e_i$ are modified as follows:

$$\dot{w}_i = \begin{cases} -\Gamma_i z e_i, & |w_i| \leq M_i \\ -\Gamma_i z e_i - \sigma_i \Gamma_i w_i, & |w_i| > M_i \end{cases}, \textbf{ (4.9)}$$

where σ_i is a positive constant chosen by the designer.

The above weight adjustment law is the same as $\dot{w}_i = -\Gamma_i z e_i$, if w_i belongs to a ball of radius M_i. In the case that the weights leave this ball, the weight adjustment law is modified by the addition of the leakage term $\sigma_i \Gamma_i w_i$, whose objective is to prevent the weight values from drifting to infinity.

If we use the vector notation $v := [v_1 ... v_n]^T$ and $e := [e_1 ... e_n]^T$ and consider the filtered error RHONN model given by (4.8) whose weights are adjusted according to (4.9) , then it can be proved that for $i = 1, 2, ..., n$:

(a) e_i , $\varphi_i \in L_\infty$

(b) There exist constants λ, μ such that $\int_0^t \left|e(\tau)\right|^2 d\tau \le \lambda + \mu + \int_0^t \left|v(\tau)\right|^2 d\tau$.

In simple words the above theorem states that the weight adaptive law (4.9) guarantees that e_i and φ_i remain bounded for all $i = 1, 2, ..., n$, and furthermore, the "energy" of the state error $e(t)$ is proportional to the "energy" of the modeling error $v(t)$.

In the special case that the modeling error is square integrable, i.e., $v \in L_2$, then $e(t)$ converges to zero asymptotically.

It is noted that the σ-modification causes the adaptive law (4.9) to be discontinuous; therefore standard existence and uniqueness of solutions, the trajectory behavior of $w_i(t)$ can be made "smooth" on the discontinuity hyper-surface $\left\{w_i \in \mathbb{R}^L : |w_i| = M_i\right\}$ by modifying the adaptive law (4.9) to

$$\dot{w}_i = \begin{cases} -\Gamma_i z e_i \ , \quad f\left\{|w_i| < M_i\right\} \ or \ \left\{|w_i| = M_i \ and \ w_i^T \Gamma_i z e_i > 0\right\} \\ -\Gamma_i z e_i + \dfrac{w_i^T \Gamma_i z_i e_i}{w_i^T \Gamma_i w_i} \Gamma_i w_i \ , \quad f\left|w_i\right| = M_i \ and \ \{-\sigma_i w_i^T \Gamma_i w \le w_i^T \Gamma_i z e_i \le 0\} \\ -\Gamma_i z e_i - \sigma_i \Gamma_i w_i \ , \quad if\left\{|w_i| > M_i\right\} \ or \ \left\{|w_i| = M_i\right\} \ and \ \left\{w_i^T \Gamma_i z e_i < -\sigma_i w_i^T \Gamma_i w\right\} \end{cases}$$

As shown in [82], the above adaptive law retains all the properties of (4.9) and, in addition, guarantees the existence of a unique solution, in the sense of Caratheodory [57]. The issue of existence and uniqueness of solutions in adaptive systems is treated in detail in [67].

[47]

4.5 Use of system identification for indirect adaptive control based on the RHONN model

Here, we deal with the development of indirect adaptive control techniques using recurrent high order neural networks. These techniques are used in order to control nonlinear dynamical systems, with highly uncertain and possibly unknown nonlinearities. Since the actual system is assumed to be completely unknown firstly we create an identification model, whose parameters are updated on-line such a way that the error between the actual system output and the model output is approximately zero. Then, the controller receives information from the identifier and outputs the necessary signal, which forces the plant to perform a pre-specified task.

4.5.1 Identification

In this phase, a RHONN is employed to perform "black box" identification around a known operational point. Previously, in (4.4.1) some learning laws were developed. These laws can be used herein in the building up of the identification part of the architecture. In order to increase robustness these algorithms can further enriched.

To begin, we consider affine in the control, nonlinear dynamical systems of the form: $\dot{x} = f(x) + G(x)u$.

These continuous-time non-linear systems are called affine because the control input appears linear with respect to G. In the above equation, the state $x \in \mathbb{R}^n$ is assumed to be completely measured, the control u is in $\mathbb{R}^n$ and f called the drift term is an unknown smooth vector field. G is a matrix with columns the unknown smooth controlled vector fields g_i, $i = 1,2,...n$ $G=[g_1 g_2 ... g_n]$.

In previous sections we considered non affine systems of the form: $\dot{x} = f(x,u)$.

These systems can be easily converted into affine, by passing the input through integrators [64], a procedure known as dynamic extension. Affine systems, both by nature and design, are commonly encountered in engineering and that is the main reason for using them.

According to the analysis of previous subsections, our model can be described by an affine RHONN model of the form: $\dot{\hat{x}} = A\hat{x} + BWS(x) + BWS'(x)u$, **(4.10)**

where $\hat{x} \in \mathbb{R}^n$, the inputs $u \in U \subset \mathbb{R}^n$, W is a $n \times n$ matrix of synaptic weights, A is a $n \times n$ stable matrix, $B$is a $n \times n$ matrix with elements the scalars b_i for all $i = 1, 2, ..., n$ and W_1 is a $n \times n$ diagonal matrix of synaptic weights of the form $W_1 = diag\left[w_{11} w_{21} ... w_{n1}\right]$. Finally, $S(x)$ is a n-dimensional vector and $S'(x)$ is a diagonal matrix, with elements combinations of sigmoid functions. For simplicity we could take matrix A to be diagonal.

In the case where only parametric uncertainties are present we can prove using techniques analogous to them presented before that:

Considering the identification scheme: $e = Ae + B\tilde{W}S(x) + B\tilde{W}_1 S'(x)u$

and the learning law : $\dot{w}_{ij} = -b_i p_i s(x_j) e_i$,

$\dot{w}_{i1} = -b_i p_i s'(x_j) p_i u_i e_i$, for all $i, j = 1, 2, ...n$

the properties following could be guaranteed:

$$e, \hat{x}, \tilde{W}, \tilde{W}_1 \in L_\infty,\ e \in L_2,\ \lim_{t\to\infty} e(t) = 0,\ \lim_{t\to\infty} \dot{\tilde{W}}(t) = 0,\ \lim_{t\to\infty} \dot{\tilde{W}}_1(t) = 0.$$

The robust learning algorithms developed previously can also be used herein to cover for the existence of modeling errors.

[47]

4.5.1.1 Robustness of the rhonn identifier owing to unmodeled dynamics

It is well known that the model can be of lower order than the plant. This is due to the fact that un-modeled dynamics exist. In order to include such cases where dynamic uncertainties are present, we extend our theory within the framework of singular perturbations. Now we assume that the unknown plant can be completely described by:

$$\dot{x} = Ax + BW_1^* S(x) + BW_1^* S'(x)u + F(x,W,W_1)A_0^{-1}B_0W_0u + F(x,W,W_1)z,$$

$$\mu\dot{z} = A_0 z + B_0 W_0 u, z \in \mathbb{R}^r, \quad \textbf{(4.11)}$$

where z is the state of the un-modeled dynamics and $\mu > 0$ a small singular perturbation scalar.

If we define the error between the identifier states and the real system states $e = \hat{x} - x$ then from (4.10) and (4.11) we obtain the error equation:

$$\dot{e} = Ae + B\tilde{W}S(x) + B\tilde{W}S'(x)u - F(x,W,W_1)A_0^{-1}B_0W_0u - F(x,W,W_1)z$$

$$\mu\dot{z} = A_0 z + B_0 W_0 u, \ z \in \mathbb{R}^r, \quad \textbf{(4.12)}$$

where $F(x,W,W_1), B_0W_0u, B\tilde{W}S(x), B\tilde{W}_1S'(x)$ are bounded and differentiable with respect to their arguments for every $\tilde{W}_1 \in B_{\tilde{W}_1}$ a ball in $\mathbb{R}^n$ and all $x \in B_x$ a ball in $\mathbb{R}^n$. Further, we assume that the un-modeled dynamics are asymptotically stable for all $x \in B_x$. In other words we assume that there exists a constant $\nu > 0$ such that $\operatorname{Re} \lambda\{A_0\} \le -\nu < 0$.

It should be mentioned that $\dot{z}$ is large since μ is small and hence, the un-modeled dynamics are fast. For a singular perturbation from $\mu > 0$ to $\mu = 0$ we obtain $z = -A_0^{-1}B_0W_0u$.

The existence of A_0^{-1} is assured because the un-modeled dynamics are asymptotically stable. As it is well known from singular perturbation theory, we express the state z as

$$z = h(x,\eta) + \eta, \quad \textbf{(4.13)}$$

where $h(x,\eta)$ is defined as the quasi-steady-state of z and η as its fast transient. In our case we have:

$$h(x,\eta) = -A_0^{-1}B_0W_0u.$$

Substituting (4.13) into (4.12) we obtain the singularly perturbed model as

$$\dot{e} = Ae + B\tilde{W}S(x) + B\tilde{W}_1S'(x)u - F(x,W,W_1)\eta,$$

$\mu\eta = A_0\eta - \mu\dot{h}(e,\tilde{W},\tilde{W}_1,\eta,u)$,

where we define $\dot{h}(e,\tilde{W},\tilde{W}_1,\eta,u) = \frac{\partial h}{\partial e}\dot{e} + \frac{\partial h}{\partial \tilde{W}}\dot{\tilde{W}} + \frac{\partial h}{\partial \tilde{W}_1}\dot{\tilde{W}}_1 + \frac{\partial h}{\partial u}\dot{u}$.

However, in the control case, u is a function of $e,\tilde{W},\tilde{W}_1$ therefore making $h(e,\tilde{W},\tilde{W}_1,\eta,u)$ equal to $h(e,\tilde{W},\tilde{W}_1,\eta,u) = \frac{\partial h}{\partial e}\dot{e} + \frac{\partial h}{\partial \tilde{W}}\dot{\tilde{W}} + \frac{\partial h}{\partial \tilde{W}_1}\dot{\tilde{W}}_1$.

It should be remarked that $F(x,W,W_1)A_0^{-1}B_0W_0u$, $F(x,W,W_1)z$ can be viewed as correcting terms in the input vector fields and in the drift term of $\dot{x} = Ax + BW^*S(x) + BW_1^*S'(x)u$, in the sense that the unknown system can now be described by a neural network plus the correction terms.

The successful completion of the identification phase implies that a model of the originally unknown nonlinear dynamical system has been obtained. Then we can proceed to the control phase of our algorithm. Stability of the identification scheme plus convergence of the identification error to within a small neighborhood of zero is guaranteed with the aid of Lyapunov and singular perturbations theories.

[47]

4.5.2 Indirect control

After having been identified around an operational point, the unknown nonlinear dynamical system is regulated to zero adaptively. In what follows cases that lead to modeling errors are taken into consideration, where parametric and dynamic uncertainties exist.

4.5.2.1 Parametric uncertainty

Firstly we consider that the unknown system can be modeled exactly by a dynamical neural network of the form:

$\dot{x} = Ax + BW^*S(x) + BW_1^*S'(x)u$, **(4.14)** where the matrices are as defined previously and the error between the identifier and the real system states is: $e = \hat{x} - x$.

Then from (4.10) and (4.14) the error equation can be obtained which is:

$\dot{e} = Ae + B\tilde{W}S(x) + B\tilde{W}_1 S'(x)u$, where $\tilde{W} = W - W^*$ and $\tilde{W}_1 = W_1 - W_1^*$.

What we want to accomplish is finding suitable control and learning laws to drive both e and x to zero, while all other signals in the closed loop remain bounded.

At each time instant the actual system is modeled by a RHONN of the form:

$\dot{\hat{x}} = A\hat{x} + BWS(x) + BW_1 S'(x)u$, where W and W_1 are the synaptic weight estimates, provided by the RHONN identifier.

Taking $u = -[W_1 S'(x)]^{-1} WS(x)$, we finally obtain $\dot{\hat{x}} = A\hat{x}$.

In order to derive stability of our adaptive laws, we use again the Lyapunov synthesis method. So if we take the Lyapunov function candidate,

$V(e, \hat{x}, \tilde{W}, \tilde{W}_1) = \frac{1}{2} e^T P e + \frac{1}{2} \hat{x}^T P \hat{x} + \frac{1}{2} tr\{\tilde{W}^T \tilde{W}\} + \frac{1}{2} tr\{\tilde{W}_1^T \tilde{W}_1\}$, where $P > 0$ is chosen to satisfy the Lyapunov equation: $PA + A^T P = -I$, we obtain that the learning laws :

$\dot{w}_{ij} = -b_i p_i s(x_j) e_i$,

$\dot{w}_{in+1} = -b_i s'(x_i) p_i u_i e_i$,

for all $i, j = 1, 2, ..., n$ make $\dot{V} = -\frac{1}{2}\|e\|^2 - \frac{1}{2}\|\hat{x}\|^2 \leq 0$.

Furthermore, it is trivial to verify that the learning laws above can be written in matrix form as $\dot{W} = -EBS_0$, $\dot{W}_1 = -BPS'UE$, where all matrices are defined as follows:

$P = diag[p_1, p_2, ..., p_n]$, $B = diag[b_1, b_2, ..., b_n]$, $E = diag[e_1, e_2, ..., e_n]$,

$U = diag[u_1, u_2, ..., u_n]$, $S_0 = \begin{pmatrix} s(x_1) & \dots & s(x_n) \\ \vdots & \ddots & \vdots \\ s(x_1) & \cdots & s(x_n) \end{pmatrix}$.

To apply the control law $u=-[W_1S'(x)]^{-1}WS(x)$, we have to assure the existence of $(W_1S'(x))^{-1}$. Since W_1 and $S'(x)$ are diagonal matrices and $s'(x_i)\neq 0$, $\forall t\geq 0$, $\forall i=1,2,...,n$ all we need to establish is $w_{in+1}(t)\neq 0$, $\forall i=1,2,...,n$. Hence $W_1(t)$ is confined, through the use of a projection algorithm ([63], [56], and [60]) to set $W=\{W_1:\|\tilde{W}_1\|\leq w_m\}$ where w_m a positive constant is. Furthermore, $\tilde{W}_1=W_1-W_1^*$ and W_1^* contains the initial values of W_1 that identification provides.

In particular, the standard adaptive laws are modified to:

$$\dot{W}=-EBPS_0,$$

$$\dot{W}_1=\begin{cases} -BPS'UE,\ if\ W_1\in W\ \ or\ \ \left\{\|\tilde{W}_1\|=w_m\ and\ tr\left\{-BPS'UE\tilde{W}_1\right\}\leq 0\right\} \\ -BPS'UE+tr\{BPS'UE\tilde{W}_1\}\left(\dfrac{1+\|\tilde{W}_1\|}{w_m}\right)^2\tilde{W}_1,\ if\ \|\tilde{W}_1\|=w_m\ and\ tr\left\{-BPS'UE\tilde{W}_1\right\}>0 \end{cases}$$

Therefore, if the initial weights are chosen such that $\left\|\tilde{W}\left(0\right)_{n+1}\right\|\leq w_m$, then we have that $\|\tilde{W}_1\|\leq w_m$ for all $t\geq 0$. This can be readily established by noting that whenever $\left\|\tilde{W}\left(t\right)_{n+1}\right\|=w_m$ then $\dfrac{d\left\|\tilde{W}\left(t\right)_{n+1}\right\|^2}{dt}\leq 0$, which implies that the weights W_1 are directed towards the inside or the ball $\left\{W_1:\|\tilde{W}_1\|\leq w_m\right\}$.

It can be proved that if we consider the control scheme $\dot{e}=Ae+B\tilde{W}S(x)+B\tilde{W}_1S'(x)u$, $u=-[W_1S'(x)]^{-1}WS(x)$, $\dot{\hat{x}}=A\hat{x}$ and the above learning law the following properties can be guaranteed:

$$e,\hat{x},\tilde{W},\tilde{W}_1\in L_\infty,\quad e,\hat{x}\in L_2,$$

$$\lim_{t\to\infty}\ e\left(t\right)=0,\quad \lim_{t\to\infty}\ \hat{x}\left(t\right)=0,\quad \lim_{t\to\infty}\ \dot{\tilde{W}}\left(t\right)=0,\quad \lim_{t\to\infty}\ \dot{\tilde{W}}_1\left(t\right)=0$$

The analysis presented above, implies that the projection modification guarantees boundedness of the weights, without affecting the rest of the stability properties established in the absence of projection.

[47]

4.5.2.2 Parametric plus Dynamic Uncertainties

Now, we examine a more general case where parametric and dynamic uncertainties are present. To analyze the problem, the complete singular perturbation model 3.6 is used so the control scheme is now described by the following set of nonlinear differential equations

Our actual system as we have seen is $\dot{x} = f(x) + G(x)u$.

The error system is:

$$\begin{aligned} e &= Ae + B\tilde{W}S'(x) + B\tilde{W}_1 S'(x)u - F(x, W, W_1)\eta, \\ \dot{\hat{x}} &= A\hat{x}, \\ \mu\dot{\eta} &= A_0\eta - \mu\dot{h}(e, \tilde{W}, \tilde{W}_1, \eta). \end{aligned}$$

The control law we use is $u = -\left[W_1 S'(x)\right]^{-1} WS(x)$.

The adaptive law is $\dot{W} = -EBPS_0$

$$\dot{W}_1 = \begin{cases} -BPS'UE, \ if\ W_1 \in W \ \ or \ \ \left\{\left\|\tilde{W}_1\right\| = w_m \ and\ tr\left\{-BPS'UE\tilde{W}_1\right\} \le 0\right\} \\ -BPS'UE + tr\{BPS'UE\tilde{W}_1\}\left(\dfrac{1+\left\|\tilde{W}_1\right\|}{w_m}\right)^2 \tilde{W}_1, \ if\ \left\|\tilde{W}_1\right\| = w_m \ and\ tr\left\{-BPS'UE\tilde{W}_1\right\} > 0 \end{cases}.$$

It can be proved that $\dot{h}(e, \tilde{W}, \tilde{W}_1, \eta, u)$ is bounded by $\left\|\dot{h}(e, \tilde{W}, \tilde{W}_1, \eta, u)\right\| \le \rho_1 \|e\| + \rho_2 \|\eta\|$, provided that: $\left\|h_w \dot{\tilde{W}}\right\| \le k_0 \|e\|$, $\left\|h_{w1} \dot{\tilde{W}}_1\right\| \le k_1 \|e\|$, $\left\|h_e B\tilde{W}_1 S'(x)u\right\| \le k_2 \|e\|$, $\left\|h_e B\tilde{W}S(x)\right\| \le k_3 \|e\|$, $\left\|h_e F(x, W, W_1)\right\| \le \rho_2$, $\left\|h_e \mathrm{A}_e\right\| \le k_4 \|e\|$ and $\rho_1 = k_0 + k_1 + k_2 + k_3 + k_4$.

This can lead us to the result that the control scheme presented above is asymptotically stable for all $\mu \in (0, \mu_0)$, where $\mu_0 = \frac{1}{2}\left(\frac{1}{2\gamma_1\gamma_2 + \gamma_3}\right)$ and that the adaptive law mentioned guarantees the following properties:

$e, \dot{x}, \eta, \tilde{W}, \tilde{W}_1 \in L_\infty$, $e, \dot{x}, \eta \in L_2$, $\lim_{t\to\infty} e(t) = 0, \quad \lim_{t\to\infty} \dot{x}(t) = 0, \quad \lim_{t\to\infty} \eta(t) = 0$,

$\lim_{t\to\infty} \dot{\tilde{W}}(t) = 0, \quad \lim_{t\to\infty} \dot{\tilde{W}}_1(t) = 0$.

[47]

Chapter 5

Adaptive identification of a pneumatic pump implemented in Matlab Simulink

In this chapter we proceed to the adaptive identification of our pneumatic pump model. In the previous chapter we examined some different identification schemes and now we are able to use them to identify the pneumatic pump model presented in chapter three. Firstly we focus on how robust learning algorithms can be used to identify our model and how it is implemented in Matlab Simulink. Then we present the simulation results of the system identification.

5.1 Implementation of the identification using robust learning algorithms in Matlab Simulink

The identification scheme we use in this thesis was presented in (4.4.1.5) and it is the robust learning algorithms for adaptive identification based on the Recurrent High Order Neural Networks model. Our actual system is the pneumatic pump model presented in (3.3). It consists of two states the line pressure P_l and the pump stroke x and each one of these two values of our actual system are given as inputs to the RHONN identifier system. Here, our objective is to train the neural network so as it can identify the pneumatic pump model.

We have a second order neural network based on the robust training algorithms scheme. The state of the ith neuron is represented by x_i with the identifier model being a 2×1 matrix given by: $\dot{x}_i = -a_i x_i + w_i^T z$, where each i represents one of the two states of our actual system (either pressure line or pump stroke).

The weight adjustment laws are given by: $\dot{w}_i = \begin{cases} -\Gamma_i z e_i , & |w_i| \le M_i \\ -\Gamma_i z e_i - \sigma_i \Gamma_i w_i , & |w_i| > M_i \end{cases}$.

$$z = \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = \begin{bmatrix} \prod_{j \in I_1} y_j^{d_j(1)} \\ \prod_{j \in I_2} y_j^{d_j(2)} \end{bmatrix} \quad and \quad y = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} s(\chi_1) \\ s(\chi_2) \end{bmatrix}.$$

Where y is the input vector to each neuron and $S(\chi)$ is a 2×1 matrix that contains high order combinations of sigmoid functions of the state χ. The sigmoidal nonlinearity employed is: $S(x) = \frac{1}{1+e^{-x}}$.

The identification error is the difference between the identifier and the actual system and it is equal to: $e_i = x_i - \chi_i$.

Moreover, we use other two matrixes A and Γ. These are 2×2 diagonal matrixes with values:

$$A = \begin{bmatrix} 120 & 0 \\ 0 & 41 \end{bmatrix}$$ and

$$\Gamma = \begin{bmatrix} 10000 & 0 \\ 0 & 30 \end{bmatrix}$$

Γ is a positive definite matrix referred to as adaptive gain or learning rate and A is a diagonal stability matrix with the coefficients a_1, a_2 being part of the underlying network architecture and being fixed during training.

In our training process, our RHONN identification model learns to approximate the dynamical behavior of the system in each epoch. The process is consisted from the following steps:

1) Initialization of the W matrix.
2) Initialization of the diagonal matrixes A and Γ.
3) Initialization of the real system and the identifier in the same initial
condition.
4) Extraction of the training data from the pneumatic pump-model we have simulated
5) The data pass through the Log-Sigmoids to compute z.

6) Evaluation of the identifier's state.

7) Calculation of the error $e = x - \chi$.

8) Calculation of the weights (W).

9) The final weight values of W are set as initial values for W in the next iteration of the training process.

We perform these steps until a number of maximum epochs is reached. We follow this steps in order to drive the error to an acceptable low value. This means that our model 'follows' the real system, which is what actually happens, since the error converges to zero.

After the successful completition of the training, we proceed to validation process, the results of which are presented in the following section.

The identification procedure is implemented in Matlab Simulink as illustrated bellow:

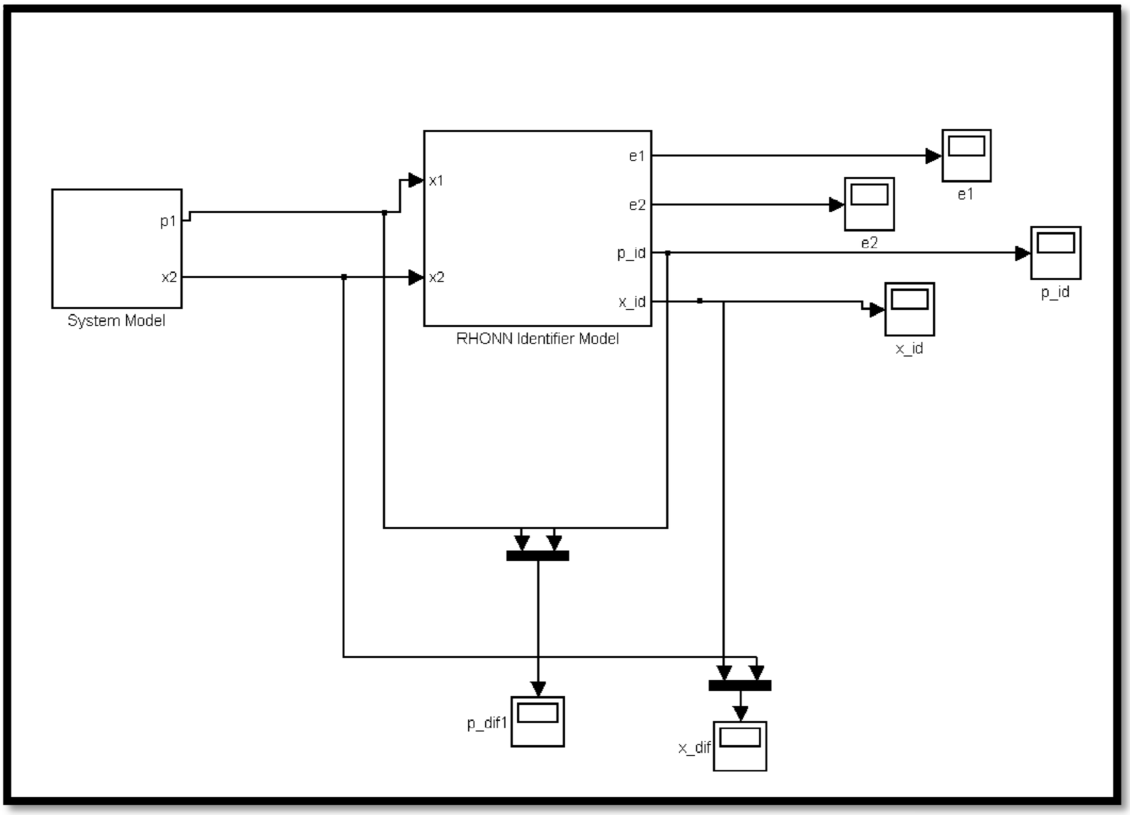


Figure 5.1

The output of our actual system is fed up as the input of the RHONN Identifier Model. The outputs we want to examine are the two errors that have to convergence to zero and the identified values of line pressure and pump stroke.

The RHONN Identifier model is implemented in Matlab Simulink the way portrayed bellow:

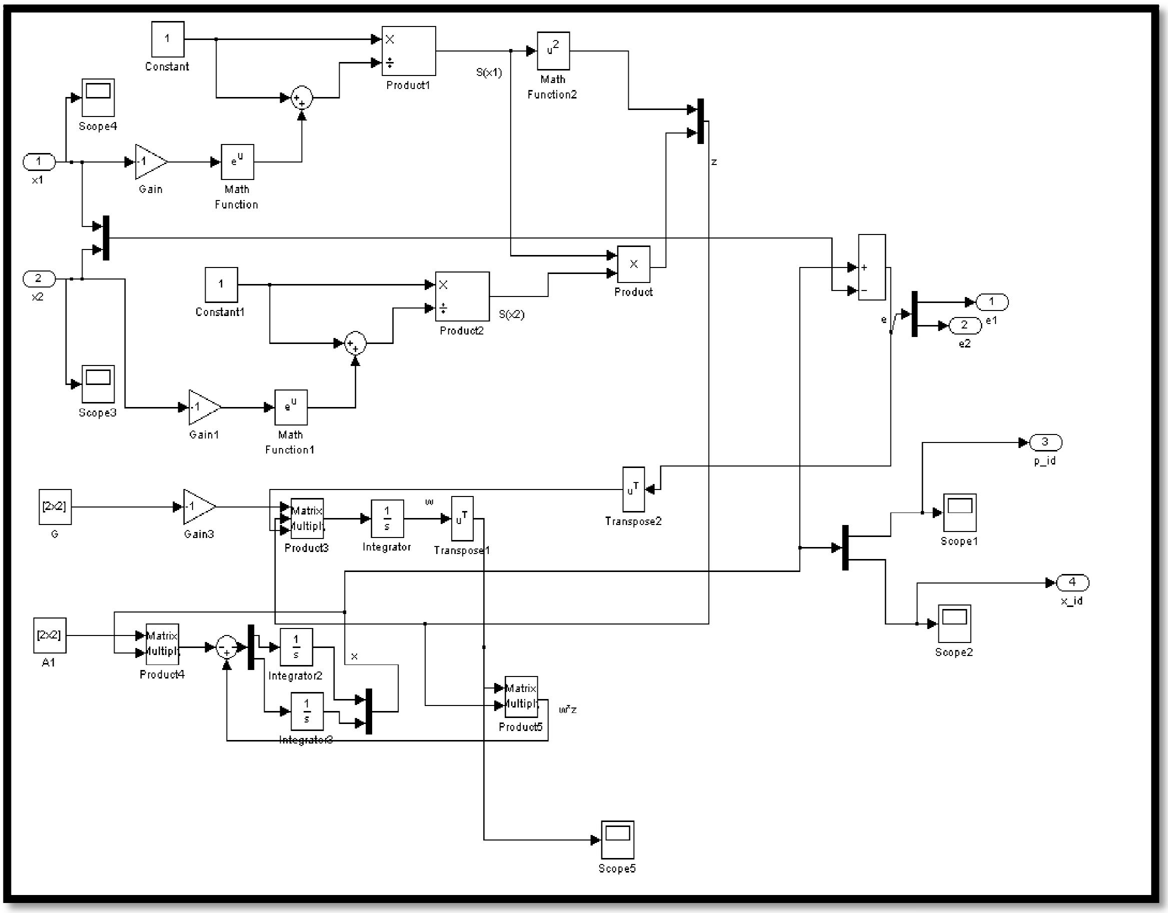


Figure 5.2 Implementation of the RHONN Identifier model

5.2 Simulation results of the adaptive identification of the pneumatic pump-model

In what follows we present the simulation results of the adaptive identification of the pneumatic pump-model. It should be mentioned that the efficiency of an identification procedure depends mainly on three factors. The error and speed of convergence, the stability in cases of abrupt input changes and the performance of the identification model after the training stops. All three factors are checked during our simulations.

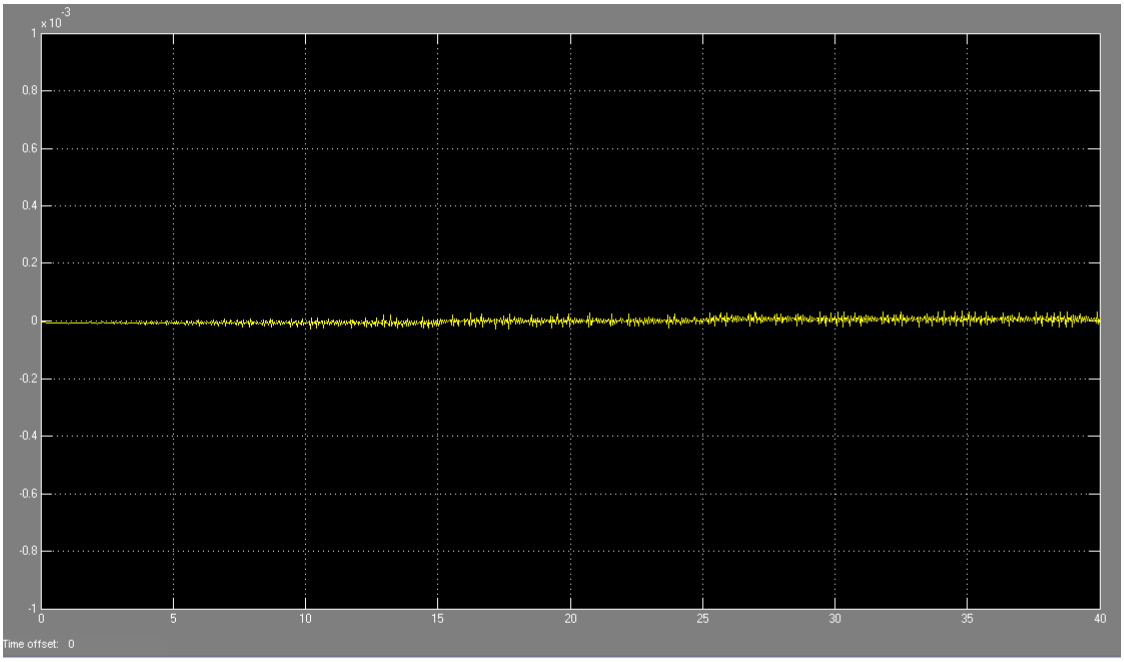


Figure 5.3 Pump stroke error

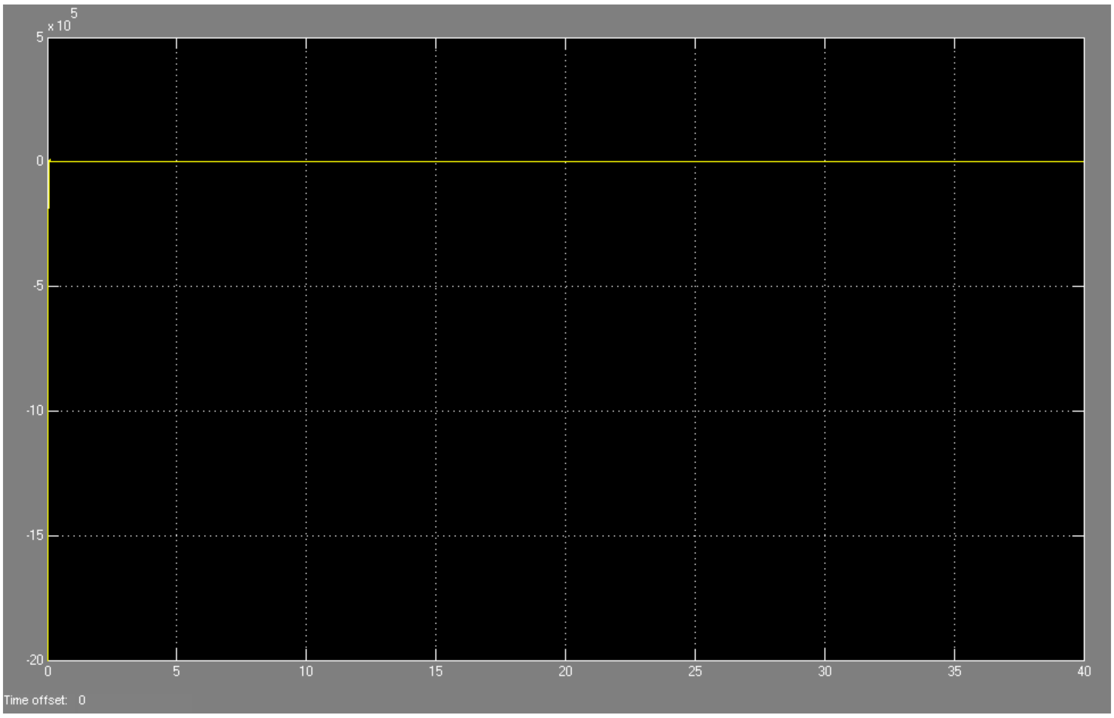


Figure 5.4 Pressure Line error

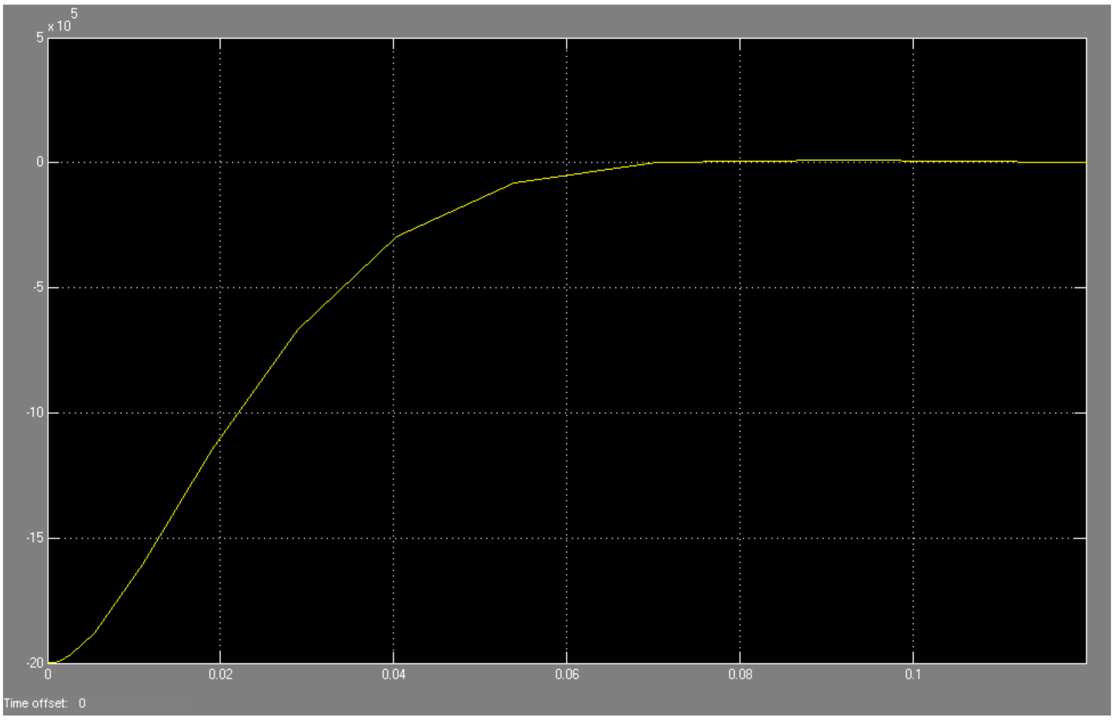


Figure 5.5 Pressure Line error with zoom for the initial 0.12 seconds

In figures 5.3 and 5.4 we can see that the approximating errors converge to zero very quickly. In figure 6.5 we can notice that line pressure error converges to zero after about 0.08 second since the initial value of line pressure is $2 \cdot 10^{6}$ Pa whereas the line pressure identifier begins from 0 Pa.

In the figures following we can see the differences between the actual system and the RHONN Identifier values. The first two figures represent the pump stroke and the other three the line pressure. The actual system's values are portrayed in pink whereas the Identification system's values are in yellow. It is very interesting how quickly both coincide.

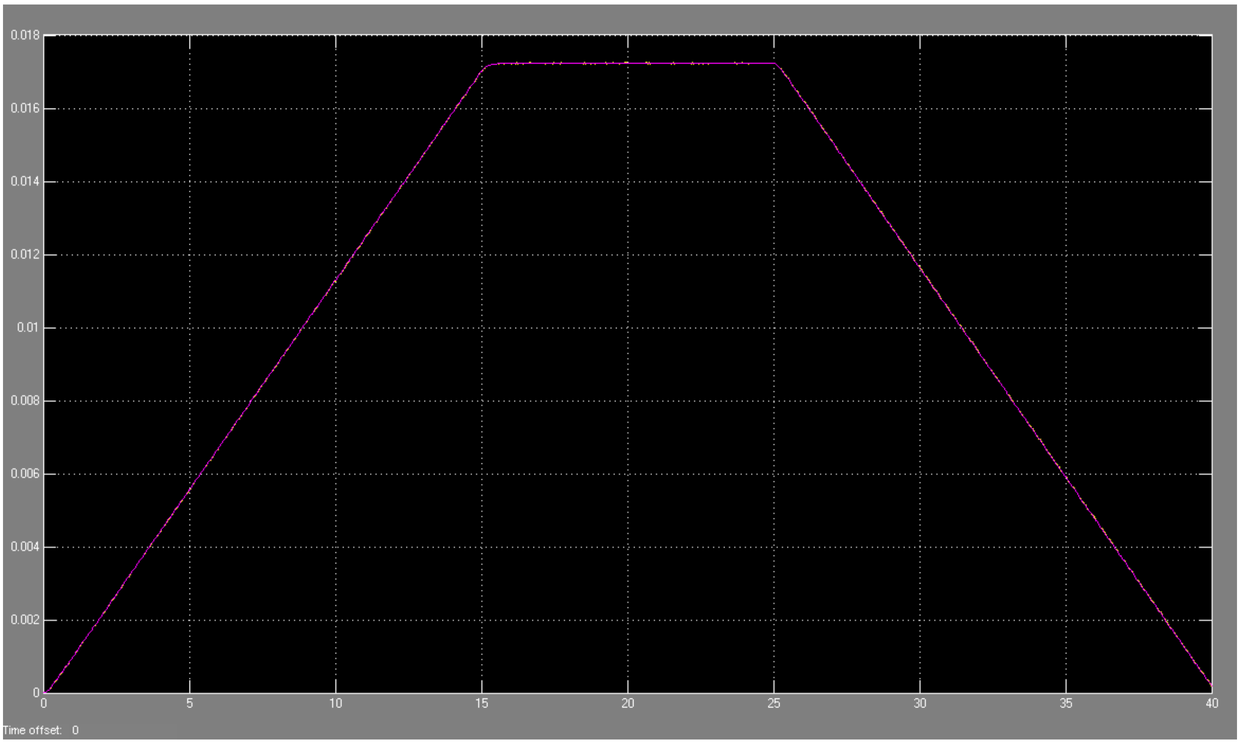


Figure 5.6 Pump Stroke both actual and identification models

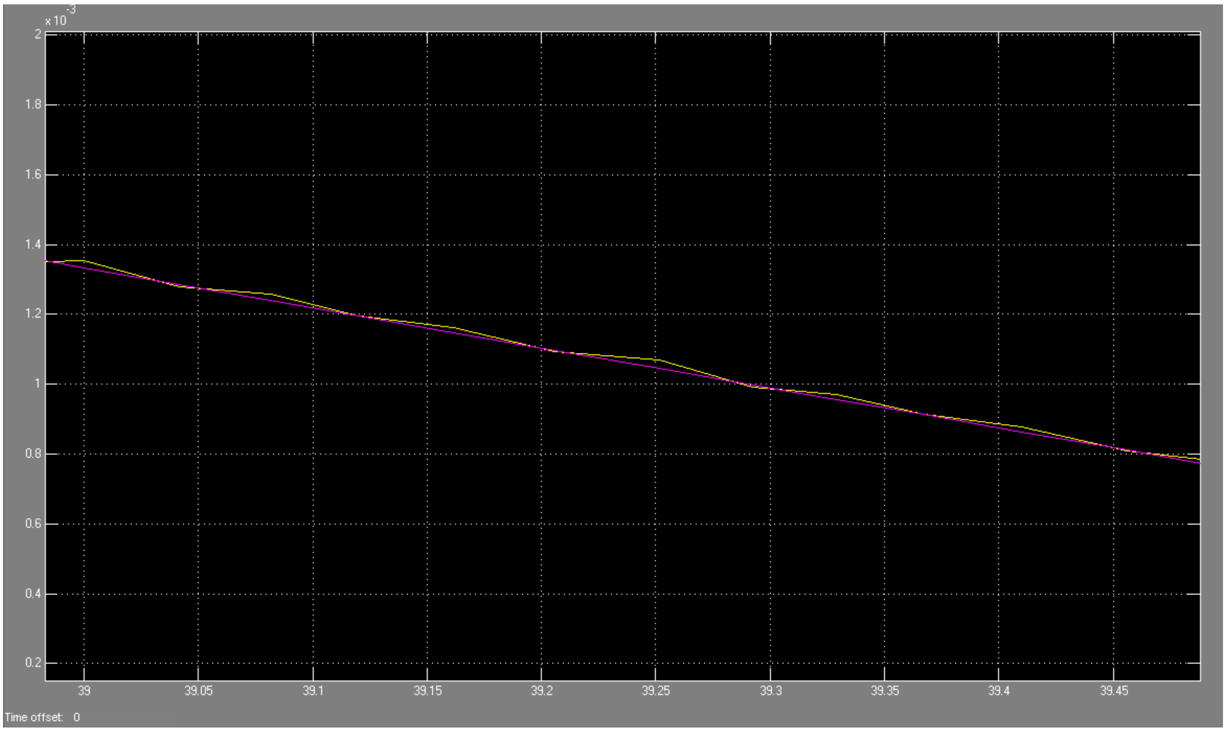


Figure 5.7 Pump Stroke both actual and identification models with zoom

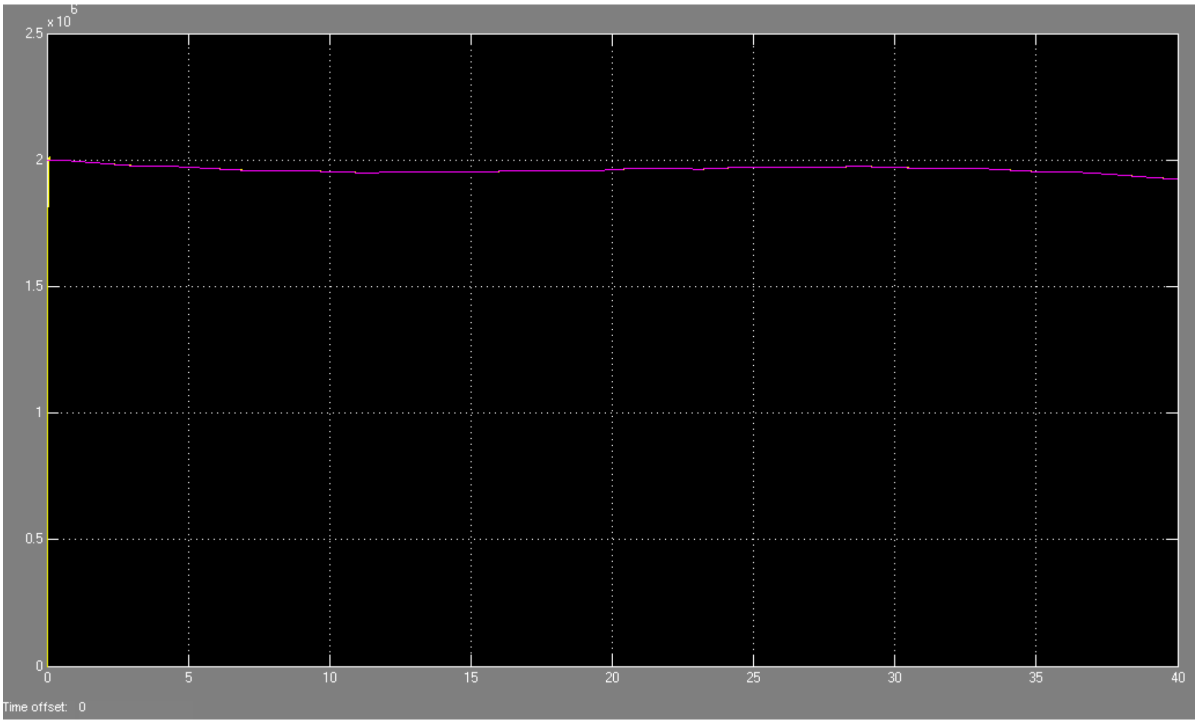


Figure 5.8 Line Pressure of both actual and identification models

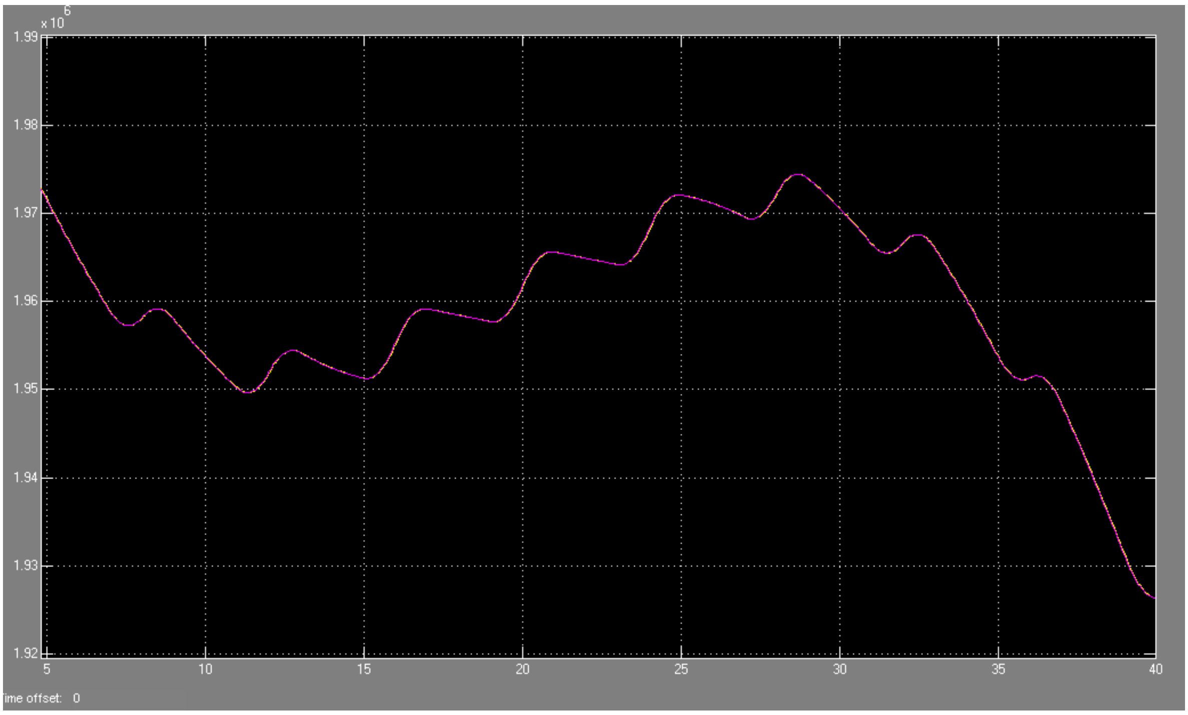


Figure 5.9 Line Pressure of both actual and identification models with zoom

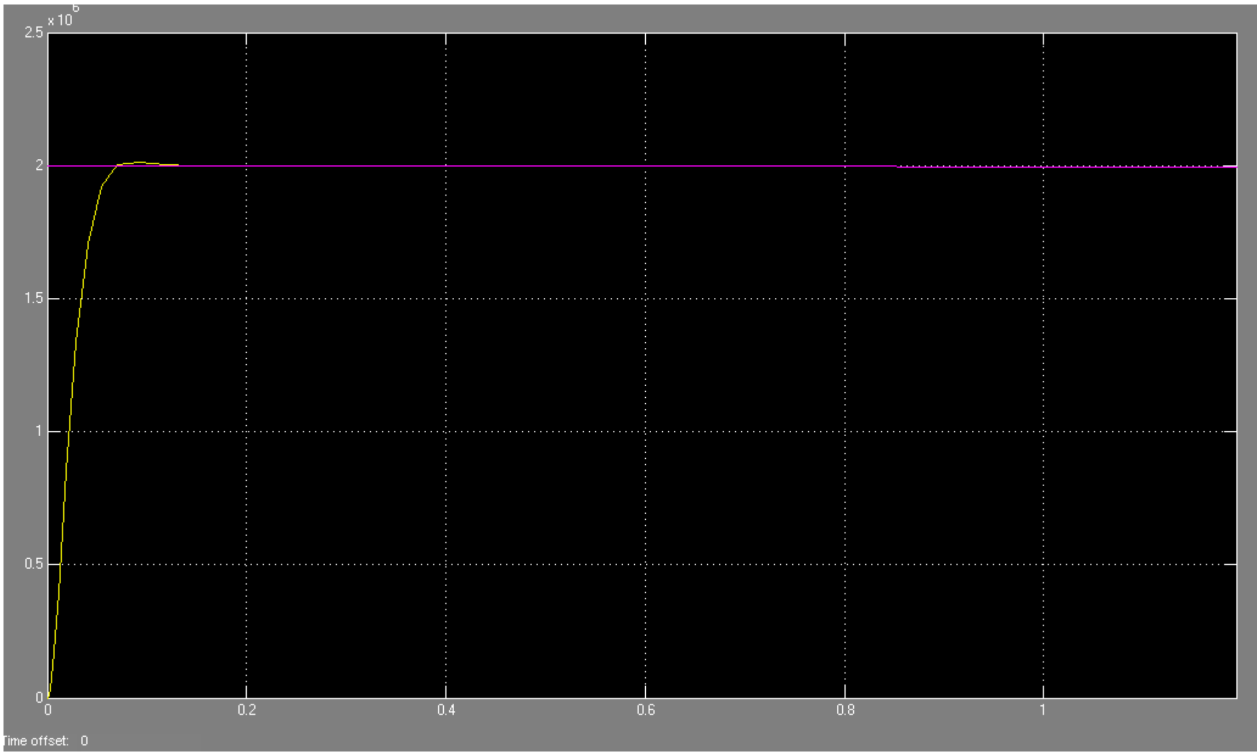


Figure 5.10 Line Pressure of both actual and identification models with zoom for the first 1.2 seconds to see exactly when these two coincide

Chapter 6

Conclusions and Future developments

In this thesis we dialed with the adaptive identification of a pneumatic pump. Firstly we presented and implemented the model and then we proceeded to the identification procedure. The identification scheme we used is for nonlinear dynamical systems and it uses robust learning algorithms when modeling errors exist. This identification scheme is based on the Recurrent High Order Neural Network model and it is very powerful since it converges to zero very quickly.

Our identification scheme, enriched further to increase robustness can be used for the design of an adaptive controller. This could possibly be the next step to our work. Recurrent high order neural networks provide a powerful mathematical tool for the calculation of the appropriate control in order to bring the system to the right dynamical behavior. This behavior would mean that we could keep our pneumatic-pump-model's line pressure between some certain values.

The control objective in our application would be to keep the line pressure equal to a constant value greater than 1MPa, which is the minimum line pressure, and less than the maximum line pressure, which is 20 MPa, whereas the demanded flow could vary. In fact in section 5.4 we have already described how system identification can be used to develop indirect adaptive control based on the Recurrent High Order Neural Network Model.

References

[1] Brian Nesbitt, ‘Handbook of Pumps and Pumping: Pumping Manual International’, Elsevier Science & Technology Books, 185617476X, December 2006

[2] Michael Volk, ‘Pump Characteristics and Applications Second Edition’, CRC Press 2005, 978-0-8247-2755-0

[3] Uno Wahren, ‘Practical Introduction to Pumping Technology’, Elsevier Science & Technology Books, 0884156869, December 1997

[4] Igor J. Karassik, Joseph P. Messina, Paul Cooper, Charles C. Heald, ‘Pump Handbook third edition’, McGraw-hill, 0-07-034032-3

[5] Garr M. Jones, Bayard E. Bosserman II, Robert L. Sanks, George Tchobanoglous, ‘Pumping Station Design Third Edition’, Butterworth-Heinemann, 0750675446, 2006

[6] Ross MacKay, ‘Practical Pumping Handbook’, Elsevier Science & Technology Books, 1856174107, April 2004

[7] Ιωάννης Μ. Μαυρουδής, ‘Αντλίες και σωληνώσεις’, Σταματούλη Α.Ε., 960-351-329-6, 2000

[8] Κωνσταντίνος Β. Ακριτίδης, ‘Αντλίες Τύποι-Επιλογή-Λειτουργία–Εγκατάσταση’, Εκδόσεις Γιαχούδη, 1985

[9] A.K. Gupta, S.K. Arora, ‘Industrial Automation and Robotics’, New Delhi Laxmi Publications, 2007

[10] John Bergendahl, ‘Treatment System Hydraulics’, American Society of Civil Engineers, 0784409196, 2008

[11] Department of Energy Fundamentals Handbook MECHANICAL SCIENCE Module 3 Pumps

[12] GATE, LLC Gibson Applied Technology & Engineering LLC, centrifugal pump selection, http://www.gatellc.com/cms-filesystem-action/gat2004-gkp-2011.001-centrifugal_pump_selection-part_1-total_head-01.25.11.pdf

[13] http://www.lightmypump.com/tutorial1.htm

[14] http://en.wikipedia.org/wiki/Centrifugal_pump

[15] http://www.school-for-champions.com/biographies/archimedes.htm

[16] http://www.bookrags.com/research/water-pump-woi/

[17] http://en.wikipedia.org/wiki/Archimedes%27_screw

[18] http://en.wikipedia.org/wiki/Shadoof#cite_note-0

[19] Courtesy of Goulds Pumps, Inc., ITT Industries, Seneca Falls, NY

[20] http://www.pumpfundamentals.com

[21] http://www.wrights-trainingsite.com/hydraulics_head.html

[22] http://www.process-pump.com/products4 html

[23] http://www.pumpschool.com/intro/pd%20vs%20centrif.pdf

[24] http://www.rpi.edu/dept/chem-eng/Biotech-Environ/PUMPS/positive.html

[25] http://www.ustudy.in/node/538

[26] http://www.globalsecurity.org/military/library/policy/army/fm/5-484/Ch4.htm

[27] http://en.wikipedia.org/wiki/Pneumatics

[28] http://en.wikipedia.org/wiki/Compressed_air

[29] http://en.wikipedia.org/wiki/Inert_gases

[30] S. R. Majumdar, ‘Pneumatic systems: principles and maintenance’, Tata McGraw-Hill Publishing Company , 0074602314, December 1995

[31] http://www.seditesa.es/index.php?option=com_content&view=article&id=81%3Abombas-neumaticas&catid=66%3Abombasneumaticas&directory=55&lang=en

[32] http://www.pumps.org/

[33] 'Bijur Delimon international', http://www.bijurdelimon.com/us /sA/france/products /product-type/pumps/pneumatic.html

[34] http://en.wikipedia.org/wiki/Solenoid_valve

[35] http://www.bijurdelimon.com/fileadmin/products/docs/bdifr/pd/en/ P_2009_11_GB_SUR881.pdf

[36] Derek Rowell, David N. Wormley, 'System dynamics: an introduction', Prentice Hall, 978-0132108089, 1997

[37] B. Wayne Bequette, 'Process control: modeling, design, and simulation', Prentice Hall, 978-0-13-353640-9, December 2002

[38] http://www.mathworks.com/products/simulink/

[39] http://www.culverco.com/sse/body/nervous.html

[40] Simon Haykin, 'Neural networks, a comprehensive foundation', Second Edition, Prentice Hall International, 0139083655, 1999

[41] Kevin L. Priddy,Paul E. Keller , 'Artificial neural networks: an introduction', SPIE Publications, 0819459879, August 2005

[42] http://en.wikipedia.org/wiki/Neuron

[43] http://en.wikipedia.org/wiki/Artificial_neural_network

[44] http://www.doc.ic.ac.uk/~nd/surprise_96/journal/vol4/cs11/report.html

[45] Tommy W. S. Chow, Siu-Yeung Cho, 'Neural networks and computing: learning algorithms and applications', London: Imperial College Press, 9781860947582, 2007

[46] http://www.information-management.com/specialreports/2008_61/10000704-1.html

[47] George A. Rovithakis, Manolis A. Christodoulou, 'Adaptive Control with Recurrent High-order Neural Networks: Theory and Industrial Applications (Advances in Industrial Control)', Springer, 2000

[48] Petros A. Ioannou, Jing Sun, 'Robust Adaptive Control', Prentice-Hall, 0134391004, 1966

[49] Kumpati S. Narendra, Kannan Parthasarathy, 'Identification and Control of Dynamical Systems Using Neural Networks', IEEE Transactions on Neural Networks, vol. 1, no. 1, March 1990

[50] George William Irwin, K.Warwick, Kenneth J. Hunt, 'Neural network applications in control', Institution of Electrical Engineers, 9780852968529, 1995

[51] http://www.scienzagiovane.unibo.it/english/artint/8-neural-networks.html

[52] P. Baldi, 'Neural networks, orientations of the hypercube and algebraic threshold functions', IEEE Transactions on Information Theory, vol. IT-34, pp. 523-530, 1988

[53] M.A Cohen, S. Grossberg, 'Absolute stability of global pattern formation and parallel memory storage by competitive neural networks ', IEEE Transactions on Systems, Man, and Cybernetics, vol. SMC-13, pp. 815-826, 1983

[54] A. Dempo, O. Farotimi, T. Kailath, 'High-order absolutely stable neural networks', IEEE Transactions on Circuits and Systems, vol. 38, no. 1

[55] G.C. Goodwin, K.S. Sin, 'Adaptive Filtering, Prediction, and Control', Prentice Hall, 1984

[56] G.C. Goodwin, D.Q. Mayne, 'A parameter estimation perspective of continues time model reference adaptive control', Automatica, vol. 23, pp. 57-70, 1987

[57] J.K Hale, 'Ordinary Differential Equations', Wiley-InterScience, 1969

[58] Y.C. Ho, 'Perturbation Analysis of Discrete Event Systems', Proc. 1st ORSA/TIMS Conf. FMS, Ann Arbor, Michigan, USA, 1984

[59] J.J. Hopfield, 'Neurons with graded response have collective computational properties like those of two-state neurons', Proc. Natl. Acad. Sci., vol. 81, pp. 3088-3092, 1984

[60] P.A. Ioannou, A. Datta 'Robust adaptive control: a unified approach', Proceedings of the IEEE, 1991

[61] Y. Kamp, M. Hasler, 'Recursive Neural Networks for Associative Memory', J. Wiley & Sons, 1990

[62] E.B. Kosmatopoulos, M Polycarpou, M.A. Christodoulou, P.A. Ioannou, 'High-order neural network structures for identification of dynamical systems', IEEE Trans. Neural Networks, vol. 6, no. 2, pp. 422-431, 1995

[63] K.S. Narendra, A.M. Annaswamy, 'Stable Adaptive Systems', Englewood Cliffs, Prentice-Hall, 1989

[64] H. Nijmeijer, and A.J. van der Schaft, 'Nonlinear Dynamical Control Systems', Springer-Verlag, 1989

[65] P. Paretto, J.J. Niez, 'Long term memory storage capacity of multiconnected neural networks', Biol. Cybern., vol. 54, pp. 53-63, 1986

[66] M.M. Polycarpou, P.A. Ioannou, 'Identificaation and control of nonlinear systems using neural network models: design and stability analysis', Tech. Rep., Univ. of Southern Cal., Los Angeles, 2001

[67] M.M. Polycarpou, P.A. Ioannou, 'On the existence and uniqueness of solutions in adaptive control systems', IEEE Trans. on Automatic Control, vol. 30, no. 3, pp. 474-479, 1994

[68] D. Rumelhart, D. Hinton, G. Williams, 'Learning internal representations by error propagation', in Parallel distributing Processing, D. Rumelhalt and F. McClelland, Eds. Vol.1 Cambridge, MA, MIT Press, 1986