High dimensional Design Optimization of Wave Energy Converters with Computational Fluid Dynamics

Christina Peuker

Born 23rd May 1989 in Bergisch Gladbach

February 20, 2014

Master's Thesis Mathematics

Advisor: Prof. Dr. Michael Griebel

Institute for Numerical Simulation

Mathematisch-Naturwissenschaftliche Fakultät der

Rheinischen Friedrich-Wilhelms-Universität Bonn

Contents

1 Introduction

The oceans bear a vast reliable natural energy source, which is still waiting to be exploited — the waves. Numerous types of Wave Energy Converters (WEC) have been proposed in the last century to take advantage of the wave energy. A few of these devices have already been tested in laboratories and also in field tests around the globe. A huge problem scientists face when trying to examine a model in a real world setting, are the cost and effort necessary to build a prototype. A challenging wave climate makes it hard to construct a device and in some cases even destroys it during construction — e.g. the project *OSPREY (Ocean Swell Powered Renewable Energy)*, which was supposed to be built in 1995 off the coast of Scotland — or after only a few years of testing, such as the WEC built in 1985 by Kvaemer on the Norwegian island Toftestallen, which was destroyed in a storm in 1988 [Graw, 1995].

A solution to a better understanding of these devices, prior to the actual construction, is the aid of *Computational Fluid Dynamics (CFD)*. This makes it possible to simulate both smaller laboratory experiments, as well as large scale projects, without ever actually building a wave energy plant.

The aim of this thesis is to employ numerical optimization techniques to maximize the energy output of a wave energy converter. The computationally expensive CFD simulation is treated as a *blackbox*, whose output or *response* is unknown. For that purpose, the energy converter design is parametrized resulting in a high dimensional parameter space. By simulating a large number of the possible parameter combinations an exploration of the high dimensional parameter space can be achieved to determine an optimal shape of the device.

High dimensional design optimization with computationally expensive blackbox functions is an active research field and not many publications address all three of the major difficulties of those so–called HEB (**h**igh dimensional, **e**xpensive, **b**lackbox) problems. Typical optimization methods for expensive blackbox functions are usually limited to lower dimensional problems [Shan and Wang, 2010].

1.1 Motivation

The world wide available resources of wave power are estimated to be around two TW [Drew et al., 2009] and only a small fraction of it is being captured today. The use of renewable energy sources is a current research topic and the demand is likely to increase over the next years. Wave power in particular has a number of advantages. For example, it has the highest energy density among the renewable energy sources [Clément et al., 2002] and waves can travel large distances with almost no loss in energy. Additionally, the wave climate changes due to seasonal temperature changes and this tends to produce more energy in the colder seasons, when there is also a higher energy demand. On top of that, wave power is available about 90% of the time as opposed to only 20% to 30% for solar or wind power [Drew et al., 2009]. Finally, the construction of wave energy converters has a very small environmental impact, comparable to offshore wind energy. Since no chemicals of any kind are involved, there is no direct pollution and also no harm to sea life, which can even swim inside the wave power plant unharmed. At coasts with already existing breakwater constructions, a device can even be built into a breakwater caisson, like at *Sakata Port*, Japan. This cuts down the construction cost and the environmental impact during construction. Depending on the size and position of a WEC, the visual impact is also very small.

Nevertheless, there are also some downsides of wave power usage. The wave climate can change dramatically on a daily basis and only seasonal averages are predictable. This influences the efficiency, since most WEC work best in a specific wave climate. Also, sudden weather changes put a power plant at risk of minor damages or even complete destruction by suddenly occurring large waves, like OSPREY and the device on Toftestallen. While some swimming or otherwise moving WEC are built to “duck” under such waves and avoid any damages, fixed constructions are at the mercy of the elements.

These considerations have to be taken into account when designing a wave power plant. So far, most devices are not competitive with other technologies and more research is necessary to cut down the cost of energy production.

The goal of this thesis is to optimize the energy output of an Oscillating Water Column device for a given wave climate and water height within specified bounds.

1.2 Solution strategy

The approach to determine the optimal design is to parametrize the wave energy converter and investigate its behavior in a CFD simulation. The output of the simulation is used to define an objective function depending on the flow velocity field, which models the energy output of the device. This objective function is to be optimized on a prescribed parameter space. Since the objective function depends on a computer simulation and cannot be expressed explicitly by an algebraic equation, not much information about its properties, especially the derivatives, is known. This is called a *blackbox* function and owing to the lack of gradient information, most gradient–based methods fail at the optimization task.

Instead, the function is sampled within the domain and a metamodel is constructed from the calculated solution to find the *global* maximum. In this manner, the response of the computationally *expensive* fluid simulation is modeled by a cheap surrogate model. This model can be chosen to be smooth enough to define its gradient and hence can be optimized by gradient–based approaches as well.

The task of choosing an appropriate set of sample points, called the *Design of Experiments* (DoE) is very important and crucial for the fidelity of the model. A large set of well selected points results in an accurate approximation, but also increases the computational cost of the optimization process. In this thesis, four different points sets are considered and compared. The found optimum is then compared to five selected reference devices, which have already been actually build and tested.

The CFD simulation consists of a numerical wave tank, which is used to numerically generate specified waves, which arrive at the discretized OWC construction. As mentioned before, each simulation run is expensive and it is not feasible to conduct a large number of them. Therefore, two cases are considered to explore the parameter space. The first case only examines two parameters and only one solitary wave induced by a falling water column is simulated. This ensures a short runtime and a larger set of points can be evaluated more quickly.

The second case inspects a set of six parameters in a fully equipped wave tank, i.e. the waves are generated by a moving paddle, called *wave maker*. These simulations run longer with regard to total simulation time and are more expensive due to the moving wave maker and larger size of the wave tank. Therefore, smaller sets of sample points are considered than in the two–dimensional case.

The CFD simulations for this thesis are performed with the Navier–Stokes solver *NaSt3DGPF* developed at the *Institute for Numerical Simulation* at the University of Bonn.

1.3 Own contribution

Following, the achievements in the scope of this thesis are summarized.

- Characterization and parametrization of Oscillating Water Column devices with respect to real world reference constructions.
- Establishing a model to express the optimization problem with the help of a high dimensional, bounded and constrained blackbox function.
- Defining two test cases with an appropriate computational complexity, that are therefore feasible to compute with the given resources.
- Optimization of the objective function within the specified parameter space, resulting in a significant improvement compared to the reference wave power devices.

1.4 Overview

This thesis is structured as follows. In **chapter 2** an introduction to theory of water waves and Oscillating Water Column devices is given. First of all, the formation and mathematical description of water waves is discussed, followed by the working principle of wave energy converters.

Chapter 3 defines the parametrization of the considered OWC and states the problem formulation including bounds and constraints on the objective function.

A overview of global optimization and the methods employed in this thesis is given in **chapter 4**. Both direct search and metamodel methods are presented with a focus on the latter, how the sample points are chosen and the surrogate model is constructed.

Chapter 5 discusses the Navier–Stokes solver *NaSt3DGPF* developed at the *Institute for Numerical Simulation*, which is used for the simulation of a numerical wave tank. This is utilized to generate waves and analyze the WEC design.

The results are presented in **chapter 6**. First, the numerical generation of the desired average wave is discussed. Then, two cases of optimization with different sets of parameters are investigated. The results show considerable improvement compared to the reference designs.

Finally, a conclusion and outlook is given in **chapter 7**.

2 Wave power and Wave Energy Converters

Waves occur naturally in different forms and are the product of a variety of influences. Understanding of how waves arise in nature and which type of waves should be captured by the wave energy converter is essential for the numerical wave generation.

This chapter gives an overview of water waves, their characteristics and how they can be described mathematically by wave theory. Also, an outline of wave energy converters, how they work and the current state of the art in research and development is given.

2.1 Water waves

Water waves are the deflection of the water surface, i.e. the free surface separating water and air, for example in the ocean, in lakes, rivers or even in puddles. They are induced by different influences, such as gravity, wind or earthquakes. Tides, for example, are the result of the gravity forces of moon and sun and have a very long period of 12 to 24 hours. In contrast, ordinary gravity waves have a period ranging from one to 30 seconds. Hence, we need to know which waves are of interest for wave energy converters.

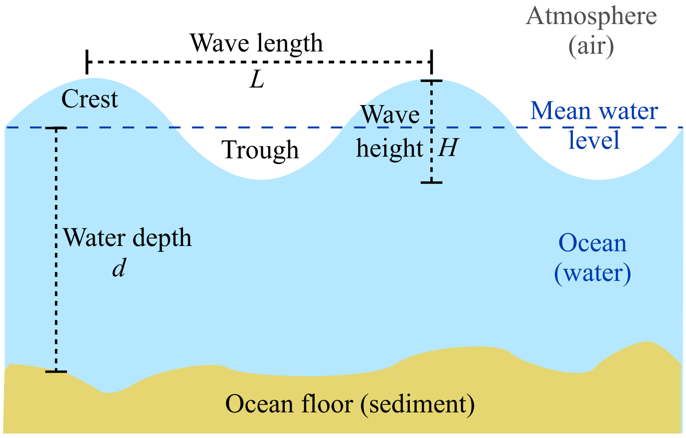


Fig. 2.1.1: Characteristics of waves.

Figure 2.1.1 depicts the wave characteristics and the notation, which is used in the following. The highest part of wave is called *crest*, the lowest part *trough*. An ideal progressive wave — like a sine wave — shows a regular sequence of crests and troughs. The horizontal distance between two successive wave crests (or troughs) is known as the *wave length* L, the vertical distance between crest and trough as *wave height* H. The *amplitude* A is the distance from the mean water level to the highest part of the crest. The *wave period* T is the time needed for a successive crest to pass a fixed point in space. The

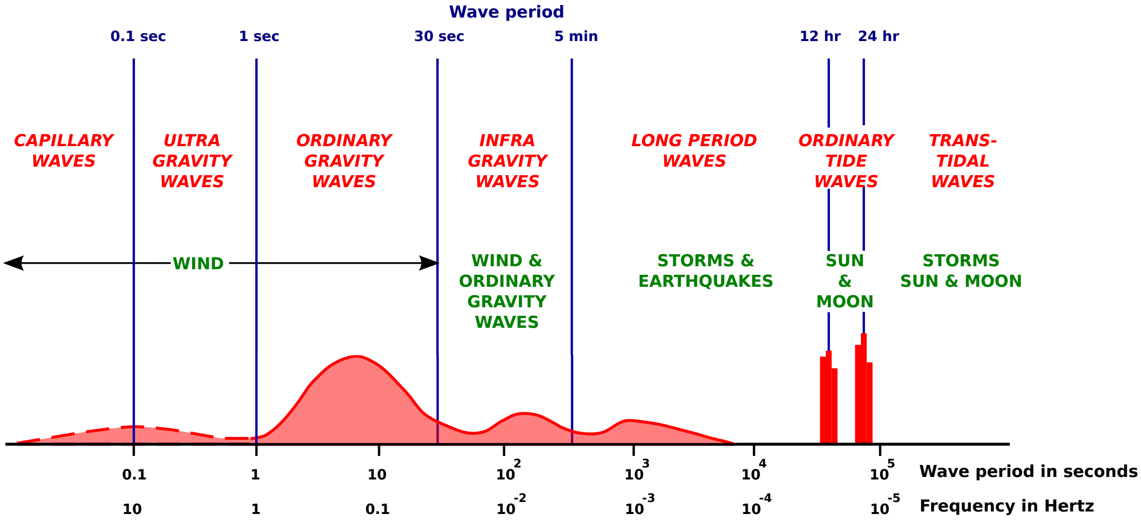


Fig. 2.1.2: Different types of waves, classified by frequency/period. Causing forces are shown in green. The curve indicates the relative amplitude [Munk, 1950].

wave velocity or *celerity* C is the rate of propagation, i.e.

$$C = \frac{L}{T}.$$

The *wave steepness* s describes the ratio of wave height to its length, i.e.

$$s = \frac{H}{L}.$$

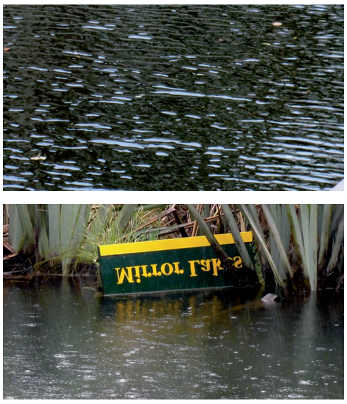


Fig. 2.1.3: Ripples (capillary waves) on the surface of a river due to wind (above) and a lake due to rain (below).

Figure 2.1.2 depicts a classification of ocean waves according to the wave period, as proposed by [Munk, 1950]. The classification and corresponding acting forces are also summarized in table 2.1.1.

Capillary waves, also called *ripples*, have a period of less than 0.1 seconds and a wave length of up to a few centimeters. They are caused by wind or droplets, such as rain drops, and their dynamics are dominated by surface tension. These waves have a great effect on the appearance of the water surface, as they cause what Munk refers to as “the inherent roughness of the surface”, c.f. figure 2.1.3. Most surfaces display this roughness caused by wind or rain. Some waters are even famous for their (almost) absence of capillary waves, such as the Mirror Lakes in New Zealand or California, which can be almost perfectly still due to lack of wind (and rain) in the morning or evening and therefore reflect the surrounding landscape. Figure 2.1.3 shows a part of the Mirror Lakes in New Zealand, which — unlike in the picture — perfectly mirrors the words “Mirror Lakes” at times without any disturbances of the water surface.

Gravity waves are induced by wind as well, but are also influenced by gravity as the restoring force. *Ultra–gravity* waves, also called *gravity–capillary waves*, have a period of about 0.1 to 1 second and are influenced by both surface tension and gravity. Capillary and gravity waves are also referred to as wind waves, or wind–generated waves. Ordinary gravity waves can be much larger than ripples; they reach heights over 30 meters and lengths of up to one kilometer. Once the local wind, which generates the wave, subsides, the wave propagates freely until it hits the coast. The propagated waves are called *swell* and can travel large distances without loosing much of their energy or height. Several waves can interfere with each other, creating a vast variety of waves in the oceans. Sometimes they cancel each other out or superpose producing one huge wave. *Infra–gravity waves* are the product of both wind and gravity and are highly influenced by the bottom topography of the entire continental shelf [Munk, 1950].

Long period waves with a period of 5 min to 12 h are caused by earthquakes resulting in tsunamis or storms producing storm surges.

Ordinary tidal waves are generated by the gravitational attraction of moon and sun. They have a fixed period of about 12 or 24 hours for semi–diurnal or diurnal waves, respectively. Such waves can be observed for example in the north sea.

Finally, trans–tidal waves with a period larger than 24 hours caused by storms or the gravitational influence of sun and moon.

Table 2.1.1: Classification of ocean surface waves [Affholder and Valiron, 2001].

Classification	Period range	Depth classification	Restoring forces	Generation forces
Capillary waves	≤ 0.1 s	Deep water	Surface tension	Winds
Ultra–gravity waves	$0.1 - 1$ s	Deep water	Surface tension & gravity	Winds
Ordinary gravity waves	$1 - 30$ s	Deep to shallow water	Gravity	Winds
Infra–gravity waves	30 s - 5 min	Deep to shallow water	Gravity & Coriolis effect	Winds & gravity
Long period waves	5 min - 12 h	Intermediate	Gravity & Coriolis effect	Storms & earthquakes
Ordinary tidal waves	12h or 24 h	Shallow water	Gravity & Coriolis effect	Gravitational attraction of moon and sun
Trans–tidal waves	≥ 24 h	Shallow water	Gravity & Coriolis effect	Storms, sun & moon

2.2 Gravity waves

Ordinary gravity waves bear a vast amount of the total estimated energy of the oceans and seas, and they propagate over large distances of the water, thus arriving steadily at the coasts. Therefore, they are of great interest for the use of wave energy converters. The energy E_w of a wave is proportional to the square of its amplitude A and the wave period T, i.e. $E_w \propto A^2T$. For example, waves with relatively long periods of about seven to ten seconds and high amplitudes of about two meters have an estimated energy of about 40 to 70 kW per meter width of incoming wave [Clément et al., 2002]. There are also devices that capture the energy of tidal waves, but these might have a greater (negative) impact on the coastal environment and might not be the best choice. Most wave energy converters are designed to capture ordinary gravity waves.

As described above, these are generated by wind with three critical factors for the shape and height of the wave: wind *velocity*, *duration* of the wind blowing in the same direction and the *fetch* — the length of water over which the corresponding wind blows in one direction. In other words, the waves are influenced by the amount of energy transferred from the wind to the water [Affholder and Valiron, 2001]. The water depth can also play a role, as described in more detail below.

The physical process of the creation of wind–generated waves is still not completely understood, but Phillips [Phillips, 1957] and Miles [Miles, 1957] developed two theories, also known as the *Miles–Phillips Mechanism*, which helped to a better understanding of the procedure. According to Phillips' theory, turbulent winds blowing over a perfectly still water surface create a resonant interaction between water and air. That means, a wind blowing over a body of water at rest, which blows not only horizontally, but also has some vertical components, creates tiny motions in the water surface. The wind pushes up disturbances in the water surface, causing pressure differences in front of and behind (in terms of wind direction) the wave crest. These pressure differences cause a fluctuation of the water surface and form capillary waves. These are vital for the generation of any wind induced wave. This mechanism is linear in time.

The second part of mechanism is explained by Miles' theory. Once the water surface contains capillary waves, the water surface appears to be more rough, which also allows the horizontally blowing wind to influence the wave climate. The wind grasps the rough water surface and pushes up already existing bumps. This leads to more roughness of the surface, which in turn lets the wind get an even better grip of the water. This process leads to an exponential growth of the waves in time.

Obviously, this procedure cannot go on forever. This is where the restoring force comes into play, which is gravity for the larger waves. It pulls the water back down at the same rate the wind lifts it up and thus prevents the wind from generating higher waves. The stronger the wind, i.e. the higher the wind velocity, the larger its influces becomes compared to gravitational forces and higher waves will be produced.

The wave height is also limited by the wave period and water depth. Once the particles at the water surface move faster than the wave velocity, the wave breaks. Whether a breaking wave is spilling, plunging, collapsing or surging (which are the four types of breaking waves) it loses energy due to turbulences or friction. This can occur on the open

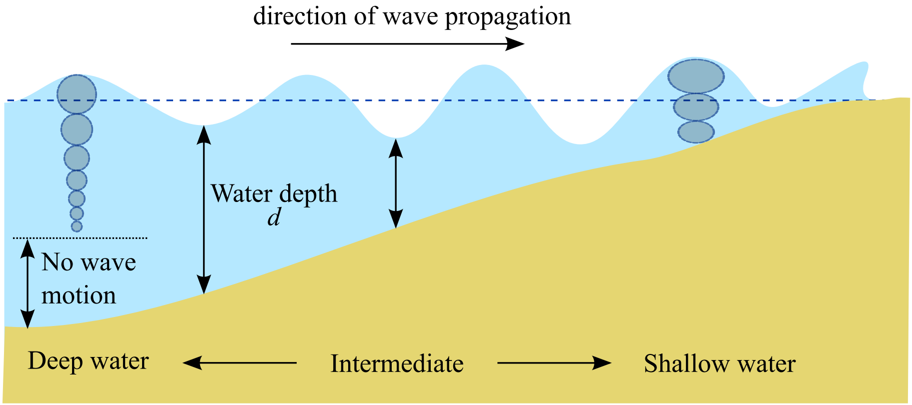


Fig. 2.2.1: Influence of water depth on waves. The division into deep and shallow water is with regard to the relative water depth d/L. The ellipses represent the movement of water particles (adapted from [Keith A. Sverdrup and Duxbury, 2000]).

sea or at the coast, where all waves eventually break. Here, the water depth is an essential influence on the wave height and shape.

As the waves reach shallow water, the friction of the bottom changes the wave. In deep water, the water particles show a circular motion, i.e. a particle on the water surface moves in a circle during a wave cycle. With less water depth, this movement transforms into an elliptical motion, since the wave motion reaches the bottom, c.f. figure 2.2.1. The characterization of deep and shallow water is with respect to the relative water depth $\frac{d}{L}$, where d is the water depth (with respect to the mean water level) and L the wave length, c.f. figure 2.1.1, which also influences the wave velocity. Deep water has a relative water depth of $\frac{d}{L} \geq \frac{1}{2}$, in shallow water it is $\frac{d}{L} \leq \frac{1}{20}$, and in the intermediate zone $\frac{1}{20} \leq \frac{d}{L} \leq \frac{1}{2}$.

When a wave reaches shallow water, it decreases in both height and velocity. A part of its energy is lost due to turbulences while breaking or by friction at the bottom, which both slows it down. Sometimes even the direction of wave propagation can change. This can be observed at beaches, for example, where the wave crests are usually almost parallel to the coastline. This phenomenon can be explained by the influence of the water depth.

2.3 Wave theory

There is a variety of water wave theories developed to describe the behavior of waves as explained above. The earliest attempts date back to Newton in 1687, later works include theories by Lagrange, Laplace, Poisson and Cauchy. In the middle of the 19th century, British scientists made considerable contributions, among them Airy, whose linear wave theory, known as *Airy theory* is still used in many applications today. This set the foundation for the works of Stokes, who derived a nonlinear wave theory. A very thorough account of the historical development of water wave theory can be found in [Craik, 2004].

The following gives a short overview over two popular theories, namely *Airy* theory describing a linear waves for shallow and deep water and *Stokes* theory describing a nonlinear waves in non–shallow water.

2.3.1 Airy wave theory

Airy wave theory [Airy, 1841], also known as *linear wave theory*, describes both deep and shallow water waves. For the deep water waves, it gives a good approximation of waves with small ratios of the wave height to water depth $H/d \ll 1$; for shallow water waves the same is true for a small ratio of wave height to wave length, called *steepness*, $H/L \ll 1$. Thus, it is limited to waves with a small amplitude. It is a linearized description of the propagation of gravity waves over a horizontal bottom and assumes a uniform mean water depth, i.e. the mean water level does not change during wave propagation. The water flow is assumed to be inviscid, incompressible and irrotational. Also, surface tension and Coriolis effect are neglected, thus gravity is the only influencing force.

A potential flow approach is used to characterize the motion of the waves. The free surface elevation $\eta(x,t)$ of a wave component, i.e. the elevation from the mean water level, is a sine function, where x is the horizontal position x and t the time. The x–axis is represented by the mean water level, which is assumed to be constant. The function η takes the form

$$\eta(x,t) = A\cos(kx - \omega t), \tag{2.3.1}$$

where A is the wave amplitude — since we consider a sinusoidal wave, the amplitude is equal to half the wave height, i.e. $H = 2a$ — k is the wave number in radian, which is related to the wave length L by

$$k = \frac{2\pi}{L}.$$

ω is the angular frequency

$$\omega = \frac{2\pi}{T}.$$

Hence, the wave velocity C can also be expressed in terms of k and ω by

$$C = \frac{L}{T} = \frac{\omega}{k}.$$

Note, that a one–dimensional model is considered, with x being the direction of wave propagation and $y = \eta(x,t)$ the surface elevation, hence $y = A$ at the top of a crest and $y = -A$ at the bottom of a trough.

As depicted in figure 2.2.1, the fluid underneath the surface moves according to the free surface motion. The particles in a propagating wave show an orbital motion, which are closed curves according to Airy theory. These orbits shrink with depth below the surface. The particles do not move in the direction of wave propagation, but rather oscillate around their average position. Thus, instead of mass, energy is transported in the direction of wave propagation.

The flow problem formulation is derived from potential theory, i.e. the general theory of solutions to Laplace's equation. As an irrotational flow is assumed, the flow velocity $\boldsymbol{u} = (u_x, u_y) \in \mathbb{R}^2$ of the fluid — which should not be confused with the wave velocity C

— satisfies $\nabla \times \boldsymbol{u} = 0$ and can be represented by the gradient of a scalar function Φ, i.e.

$$\boldsymbol{u} = \nabla\Phi,$$

which is called *velocity potential* of $\boldsymbol{u}$. Since the flow is also supposed to be incompressible, the fluid density ρ is constant and the continuity equation

$$\frac{\partial\rho}{\partial t} + \nabla \cdot (\rho\boldsymbol{u}) = 0$$

reduces to

$$\nabla \cdot \boldsymbol{u} = 0.$$

Thus, the velocity potential satisfies the Laplace equation

$$\nabla \cdot (\nabla\Phi) = \Delta\Phi = 0. \tag{2.3.2}$$

As the assumptions were made for the water phase, this equation holds for $-d \leq y \leq 0$.

Three boundary conditions close the system of equations. The first one, known as *kinematic boundary condition* is derived from the fluid particle motion, and states, that particles close to the surface stay close to the surface over time, following the motion of the surface. In general, it can be expressed as

$$\frac{\partial\eta}{\partial t} + \boldsymbol{u}\frac{\partial\eta}{\partial x} = \frac{\partial\Phi}{\partial y} \quad \text{at } y = \eta. \tag{2.3.3}$$

In the case of Airy theory, this equation is linearized and reduces to

$$\frac{\partial\eta}{\partial t} = \frac{\partial\Phi}{\partial y} \quad \text{at } y = \eta. \tag{2.3.4}$$

Additionally, the bottom is supposed to be impermeable, which is expressed by the *bed kinematic boundary condition*

$$\frac{\partial\eta}{\partial t} = 0 \quad \text{at } y = -d. \tag{2.3.5}$$

The third boundary condition is derived from Bernoulli's equation for unsteady potential flow

$$\frac{\partial\Phi}{\partial t} + \frac{1}{2}\boldsymbol{u}^2 + \frac{p}{\rho} + g\eta = 0 \quad \text{at } y = \eta, \tag{2.3.6}$$

with gravity denoted by g and pressure by p, which in turn is derived by integrating Euler's momentum equation. In linear theory, the pressure at the water surface is assumed to be constant and equal to zero without loss of generality, since this does not alter the flow. Also, the nonlinear term in $\boldsymbol{u}$ is dropped and thus yields the *dynamic boundary condition*

$$\frac{\partial\Phi}{\partial t} + g\eta = 0 \quad \text{at } y = \eta. \tag{2.3.7}$$

Given the surface elevation is of the form (2.3.1), the potential flow can be derived from the solution of the system of equations (2.3.2), (2.3.4) and (2.3.5), which is

$$\Phi(x,y) = \frac{Ag}{\omega}\frac{\cosh(k(y+d))}{\cosh(kd)}\sin(kx-\omega t). \tag{2.3.8}$$

The last boundary condition, equation (2.3.7), is used to derive the dispersion relation

$$\omega^2 = gk\tanh(kd).$$

Hence, ω and k are dependent and cannot be chosen independently.

2.3.2 Stokes wave theory

Similar to the linear theory developed by Airy, Stokes derived a nonlinear theory for periodic waves in non–shallow water [Stokes, 1847]. The underlying assumptions are the same as in Airy theory, i.e. the fluid is assumed to be inviscid, incompressible, irrotational, of constant density and the only influencing force to be considered is gravity. Hence, the same system of equations (2.3.2), (2.3.3), (2.3.5) and (2.3.6) is studied, but not linearized. Instead, a Fourier series is considered.

The elevation function $\eta(x,t)$ and the velocity potential $\Phi(x,y)$ are considered to be periodic with spatial period equal to the wavelength L, i.e.

$$\eta(x+L) = \eta(x) \quad \text{and} \quad \Phi(x+L,y) = \Phi(x,y).$$

Also, the velocity potential is assumed to be an odd function in space and therefore the surface elevation must an even function. These considerations lead to a similar form of η as in equation, namely a Fourier series von even components

$$\eta(x,t) = \sum_{n=1}^{\infty}\eta_n\cos(knx-\omega nt), \tag{2.3.9}$$

where H_n may depend on A and d and have to be calculated using the boundary conditions. Likewise, the potential velocity takes the form

$$\Phi(x,y) = \sum_{n=1}^{\infty}\phi_n\frac{\cosh(kn(y+d))}{\cosh(knd)}\sin(knx-\omega nt) \tag{2.3.10}$$

and the dispersion relation

$$\omega^2 = \sum_{n=1}^{\infty}\Omega_n gk\tanh(knd).$$

It is not feasible to calculate all terms in the Fourier series, hence an approximation up to order N is used in Stokes theory, i.e. the series is truncated for $n > N$. For $N = 1$, this yields the linear approximation of Airy theory, for $N = 2$ a second order approximation, and so forth. Most studies use approximations up to fifth order.

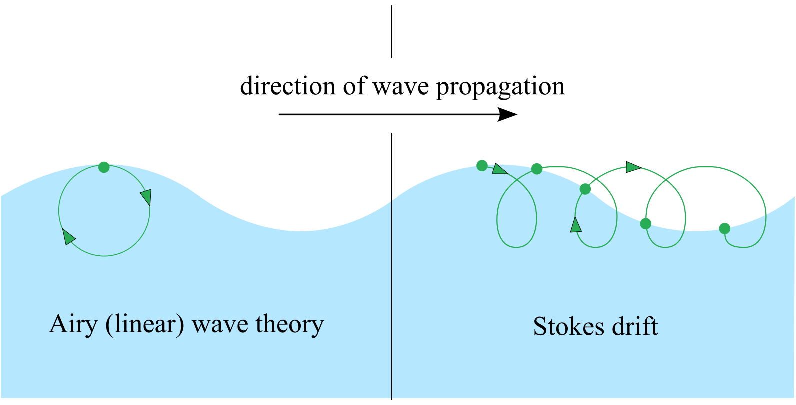


Fig. 2.3.1: Movement of water particles according to Airy and Stokes wave theory, respectively. The green point tracks the motion of a particle over time.

Another aspect, which changes in higher order approximations is the particle movement. In linear theory, they are assumed to move in circles or ellipses around a fixed point, which is their average position in space over time, i.e. a particle starting at position (x, y) will always return to position (x, y) after each time period T. However, taking higher order terms into account, the particles additionally exhibit a movement in the direction of wave propagation, i.e. a particle starting in (x, y) will move some distance X in the direction of wave propagation, Y in vertical direction, and end up at position $(x + X, y + Y)$ after one period T, c.f. figure 2.3.1. This phenomenon is known as *Stokes drift* and can be observed in nature, for example when objects are washed ashore. Hence, besides the transport of energy in a wave, there is also a transport of mass. The exact magnitude of X and Y can be calculated by a Lagrange representation of the flow velocity.

Other wave theories include the *Boussinesq approximation* for nonlinear waves in shallow water, with arbitrary direction of wave propagation, i.e waves can travel in opposing direction simultaneously. The *Korteweg–de Vries (KdV) equation* is a nonlinear equation of third order, which defines long shallow water waves. The latter is easier to solve than the Boussinesq equation. One type of solutions of the KdV equation are known as *Cnoidal waves*. Further information can be found in [Affholder and Valiron, 2001, Dean and Dalrymple, 1990]

Knowing how waves are formed and how to describe them mathematically, is very important for the physically correct numerical generation of waves, see section 5.4.

2.3.3 Wave climate

The theory described so far assumes a very basic model of a wave and is essentially two–dimensional. However, actual ocean waves are far more complicated, less predictable and of course three–dimensional.

One of the most challenging tasks is to estimate the average wave climate for a given location. This is usually described by the *significant wave height* $H_s = H_{1/3}$ and *significant wave period* $T_s = T_{1/3}$, which are determined by the mean height and period, respectively, of the highest third of the waves. This definition stems from the original kind of wave measurement, namely the visual judgment, which usually overlooks smaller waves and focuses on the larger components.

There are various methods to measure waves, visually or with the help of instruments. [Affholder and Valiron, 2001] categorizes the available tools into three groups: surface type, sub–surface or underwater type and above surface or aerial type. The first class includes step–resistance gauges, which are usually installed at piers or other shoreline structures. The gauge consists of a sealed pipe with several electrical contact points, which measure the presence or absence of water along the vertical pipe. Since these kind of measuring devices are sometimes impractical due to size, cost or visual obstruction, underwater instruments are a good alternative. These can detect pressure fluctuations under the water surface or measure the vertical and horizontal movement of buoys fixed at the seabed to capture the wave motion. The latter is a very common tool in the North Sea. Up–to–date reports and forecasts can be found at the Bundesamt für Seeschifffart und Hydrographie (BSH)[1], which has six buoys in different locations.

Finally, aerial type wave measuring tools include satellites, aircrafts and radars, which determine the wave climate from a remote location. The main difference to the first two approaches is, that the waves can be measured over a greater area and therefore are usually space averaged and almost immediate, as opposed to the time averaged point measurements of gauges or buoys.

Several large data collections (atlases) of wave energy statistics have been developed over the last decades. Among them the *WERATLAS*, which covers European deep waters and is an essential tool for wave energy planning in Europe [Falcão, 2010]. Since the water depth has a large influence on the wave propagation, there is hardly any comprehensive data available for near–shore areas. One exception is the *ONDATLAS* [Pontes et al., 2005], which contains detailed statistics of the Portuguese coast. Data for specific sites, where a Wave Energy Converter is planned, usually have to be collected prior to designing the actual device.

Even with given statistics for a given construction site, it is still hard to predict the actual wave climate for the near future, except for seasonal fluctuations and a rough average. Nevertheless, the wave amplitude, period and direction are highly irregular and sudden extreme weather conditions, like storms or hurricanes, put any construction at risk to be at least partially destroyed. Hence, it is crucial to estimate the expected load on a device, while neither under– nor overestimating it. The latter might lead to unnecessarily high production costs.

[1] available online at `www.bsh.de/de/Meeresdaten/Beobachtungen/Seegang`

2.3.4 Wave simulation

For a controlled simulation of waves in a lab, so–called *experimental wave tanks* (EWC) are used. Inside the wave tank, different waves can be reproduced and small–scale tests can be performed. This method is particularly of interest for the testing of wave energy converters.

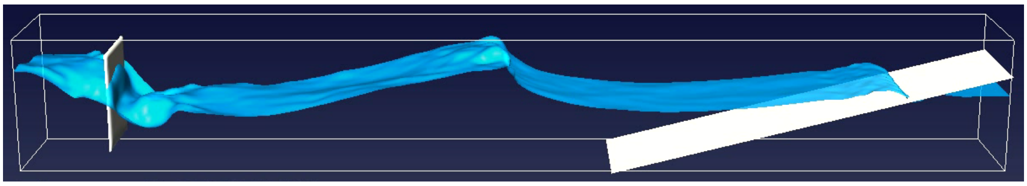


Fig. 2.3.2: Wave tank with a piston wave maker [Peuker, 2011].

An experimental wave tank can be thought of as a huge aquarium, with a *wave maker*, consisting of a moving object, for example a paddle or flap, on the one side, and a kind of *beach* on the other side to prevent the waves from being reflected off the back wall of the wave tank, c.f. figure 2.3.2. This "beach" can take different forms. For example, a literal beach, i.e. a slope in the bottom of the tank, slows the waves down through friction and wave breaking. Another solution can be the use of a lower wall at the end of tank, such that the waves can overflow, in which case the overspilling water has to be transported back into the wave tank. Some wave tanks also include a sediment bed to examine the interaction between sediment and waves, e.g. [Hudson et al., 2001].

2.4 Wave Energy Converter

A variety of wave energy converters have been proposed to capture the power of the ocean. The devices not only differ in construction type (e.g. floating vs. fixed at the seabed), but also in the type of energy conversion (water or air driven turbine vs. flexible hydraulic elements). Also, the devices are designed for a particular location: on–shore, near–shore or off–shore with a specific wave climate and water depth, while keeping their effect on the environment as low as possible. A good overview of different WEC types can be found in [Graw, 1995]; current state of art and an overview of former and current installments, both experimental and commercial, can be found in [Clément et al., 2002, Falcão, 2010].

2.4.1 Types of WEC

There are various ways to classify WEC devices, for example by type of installment, i.e. *fixed* or *floating*. The former includes Oscillating Water Column (OWC) devices, c.f. section 2.4.2, such as **Pico** on Pico Island, Azores, or **LIMPET** by Voith on Islay, Scotland; and over–topping devices like **TAPCHAN** (Tapered Channel Wave Power Device) by Norwave located in Norway, which collects incoming waves in a basin and lets the water flow back into the ocean, driving an hydroelectric turbine with the flow.

Floating designs also include OWC, like the **OE Buoy** by Ocean Energy Ltd. in Ireland or the **Mighty Whale** by JAMSTEC in Japan, and over–topping mechanisms, such as the **Wave Dragon** in Denmark. Other floating devices use the wave motion through hydraulic elements, which employ either the wave heave (i.e. vertical translation) or the rotational component, such as the **Pelamis** by the Scottish Pelamis Wave Power Ltd. The latter has been successfully tested in full scale and has now, amongst other projects, several wave farms in development.

In the course of this thesis, the focus lies on fixed OWC devices, as they can be modeled and examined even without the turbine and have a quite basic construction.

2.4.2 Oscillating Water Column

Figure 2.4.1 depicts the simple composition of an OWC device: the construction consists of a partly submerged building with two openings. One is under water to capture the incoming waves and the other is above the water for a turbine, usually at the back or on top of the building, and lets the air flow through the turbine. As the waves rush into the device, the water level rises inside the chamber, which compresses the enclosed air and therefore pushes the air through the turbine. Once the waves retreat, the water level falls, letting the air flow back into the chamber and passing the other way through the turbine. This oscillating movement of the water level — or water column — gave these devices their name.

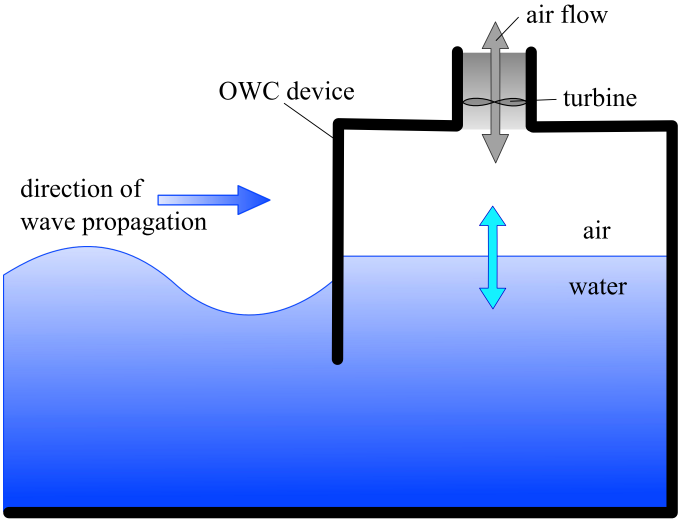


Fig. 2.4.1: Operation mode of an *Oscillating–Water–Column* device. The enclosed water column induces an air flow, which drives the turbine when flowing in and out of the chamber [Peuker, 2011].

One aspect that needs special attention is the choice of turbine, as the air flow is bidirectional. Several possible solutions have been proposed to solve this problem, for example the *Wells turbine*, which always rotates in the same direction, independently of the air flow direction. Another bidirectional turbine is the *Denniss–Auld turbine*, or its advancement the *airWAVE* by Oceanlinx.

The use of air turbines instead of directly water driven turbines ensures longer durability due to less corrosion and other technical factors.

This type of WEC can be installed onshore, near shore or offshore and it can be either installed fixed into the ground or a wall, or it can float on the water. A wide variety of these devices have been proposed and some of them have already been tested in small scale wave tanks, fewer even in actual open sea tests.

Following, five OWC devices, are described, which are depicted in figure 2.4.2. These particular designs were chosen for comparison reasons and are also modeled in the course of the thesis, c.f. section 3.2. Figure 2.4.2 also shows the computer simulation model of each device. Note, that only one chamber is modeled, even if some of them are composed of several chambers.

All of these wave energy converters are on–shore or near–shore devices, which are fixed at the seabed. They share certain characteristics, such as comparable height and width, and are built in similar wave climates and water depths.

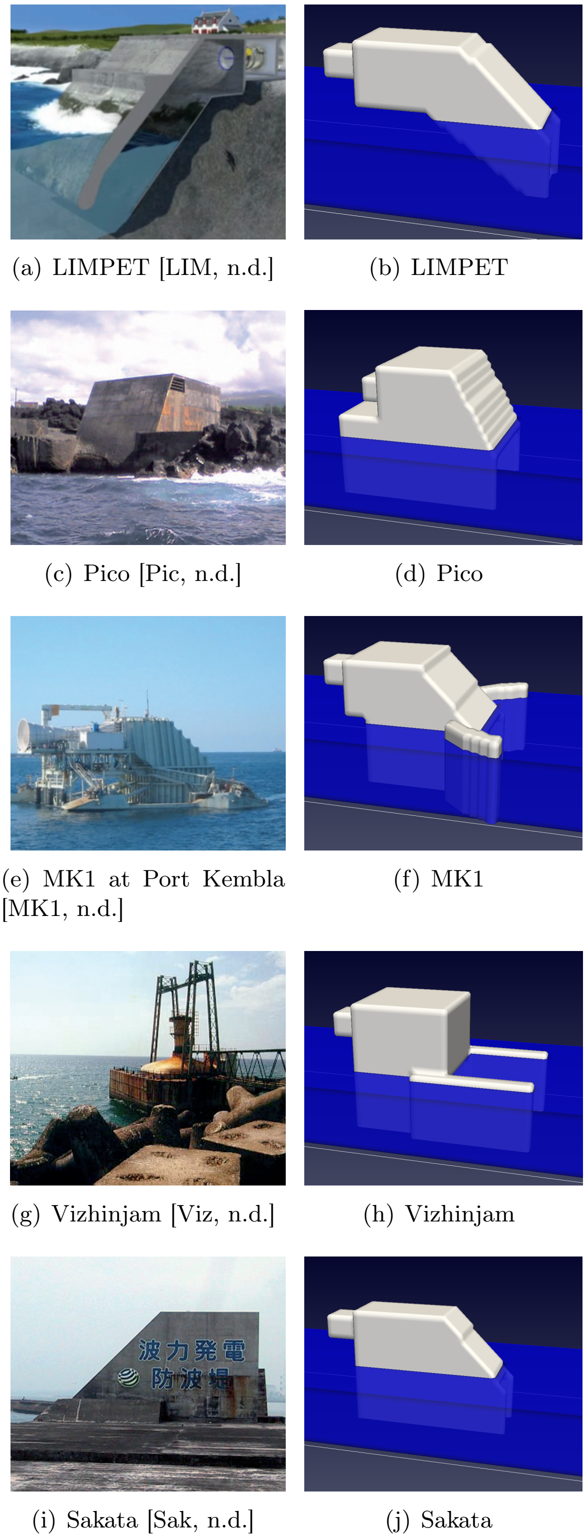
LIMPET, Islay, Scotland

The first commercial OWC device, known as LIMPET (Land Installed Marine Power Energy Transmitter) [Boake et al., 2002], was installed in 2000 on the Isle of Islay, Scotland. It was designed and built by the Irish Wavegen Ltd. (which was bought by the German company Voith in 2005) in cooperation with the Queen's University of Belfast, UK, with support of the *European Non Nuclear Energy Programme JOULE III.* The device delivered power into the British power grid until it was shut down in 2013.

A first 75 kW[2] prototype was built in 1991 and in operation until 1999. The full scale plant consists of three chambers with a width of six meters each and a height of 12.8 meters above mean water level. The chamber is angled at 40° to the horizontal, c.f. figure 2.2(a), which is supposed to give an easier path for the water resulting in less turbulence and hence a lower energy loss [Webb et al., 2005].

Although the original estimated overall efficiency — from wave to pneumatic to mechanical to electrical — was predicted to be about 48%, the actually measured efficiency was only 8%. Part of the problem was the overestimated wave climate at the construction site. Prior measurements had estimated 20 kW/m, whereas after construction a resource of only 12 kW/m was measured at the lip. This demonstrates the importance of wave climate measurements prior to construction.

[2] A model with a rated capacity of 1 kW will produce 1 kWh of electricity per hour when exposed to a specific wave climate.



(a) LIMPET [LIM, n.d.]

(b) LIMPET

(c) Pico [Pic, n.d.]

(d) Pico

(e) MK1 at Port Kembla [MK1, n.d.]

(f) MK1

(g) Vizhinjam [Viz, n.d.]

(h) Vizhinjam

(i) Sakata [Sak, n.d.]

(j) Sakata

Fig. 2.4.2: Actual OWC constructions and the corresponding OWC geometries for the values according to table 3.2.1.

European Wave Energy Pilot Plant, Pico, Portugal

Another wave energy plant was built from 1996 to 1999 on Pico Island in the Central Group of the Portuguese Azores. It was designed by the Instituto Superior Tecnico of Lisbon and is managed by the Wave Energy Centre (WavEC), a private non–for–profit organization, since 2004. The device has undergone several testing phases and subsequent improvements and is currently in danger of collapsing due to corrosion[3].

The OWC plant consists of one chamber with a width of 12 meters and a total height of 15.0 meter above mean water level. The front of the chamber is angled at 60° to the horizontal, c.f. figure 2.2(c).

The device was built as pilot project co–funded by the European Commission to evaluate and demonstrate the viability of wave energy on small islands. So far it has been very successful and has also been included in the European Marine Renewables Infrastructure Network (MARINET) as a one of the most relevant testing devices in Europe[4].

MK1, Port Kembla, Australia

The Australian company Oceanlinx (formerly Energetech Australia Pty Limited), founded in 1997 by Dr. Tom Denniss, has developed three OWC devices so far, which where built in full scale at the Australian coast at Port Kembla in New South Wales. The first one, called MK1, was built in 2005 and heavy enough — weighing 500 tonne — to sit on the seabed under its own weight. It has one chamber with 10 meters in width, which is angled at 45°. Unlike LIMPET or the Pico plant, it has parabolic *harbor walls* at each side to concentrate the incoming waves, c.f. figure 2.2(e).

Another difference is the employed turbine. LIMPET and the Pico device both use the Wells turbine, whereas Oceanlinx developed its own bidirectional air turbine, known as the Denniss–Auld turbine. Its successor, the airWAVE, uses a latching control stategy and is supposed to spin constantly regardless of change of direction of the air flow.

In 2012, Oceanlinx was awarded the first prize in the The Australian Innovation Challenge in the category Energy and Minerals for its 1 MW OWC plant greenWAVE, an advancement of the MK1.

Vizhinjam, Trivandrum, India

At Port Vizhinjam near the town Trivandrum, India, one of the oldest full scale OWC has been built in 1991. The device also includes harbor walls, which protrude straight towards the sea. It consists of one chamber, 10 meters wide and 14.5 meters above sea level. The front is a vertical wall, which is not angled, c.f. figure 2.2(g).

A new, improved model has been commissioned in 1996 and research has been continued up to a few years ago. One of the biggest problems was the fluctuations in power generation. Only during the months of monsoon (June to September), the facility was able to capture

[3] See `http://www.save--pico--powerplant.org/`

[4] See `http://www.fp7-marinet.eu/WavEC-wavec-pico-owc.html`

a maximal amount of energy from the incoming waves. During the rest of the year, the power output was low to nearly non existent. This again demonstrates the difficulties faced with fluctuations in the wave climate.

Sakata, Japan

The first breakwater OWC was built into an existing breakwater caisson at Sakata Port, Japan, in 1989 [Takahashi et al., 1992]. It was a project of the Port and Harbor Research Institute, a division of the Ministry of Transport. The device consists of five chambers with a width of 3 meters each. However, only three of the chamber were actually used afer a testing phase. The front is angled at 45° and the caisson has a height of 12.5 meters above sealevel, c.f. figure 2.2(i).

Many other OWC projects have been launched around the globe and research is growing in many countries. However, only a few projects have made the jump to a commercial start so far, while others have been shut down or are still struggling, as the OWC plant on Pico Island. Part of the problem are the high construction costs and the improvable overall wave–to–wire efficiency, which makes it less profitable. Apart from the very important turbine construction, a better understanding of the WEC design itself through computer simulations can help immensely to cut down the cost of research and development and improve the energy output.

3 Optimization Problem

In the previous chapter, theory on waves, wave power and wave energy converters was introduced. From this discussion, it became clear, that a computer simulation of an OWC device can be helpful to achieve an improved efficiency of the construction. The following introduces the mathematical model to undertake this task.

3.1 Definition of the objective function

A complete model of a wave power plant would involve many degrees of freedom, including the simulation of air and water via a CFD simulation, different wave climates and a turbine. However, such a complex model also requires a vast amount of computation time and has many unknowns leading to an exceptionally high dimensional problem.

Therefore, a few simplifications are in order. First of all, the construction is to be tested in a prescribed wave climate. An "average" wave is generated and used for all simulations. More details on the chosen wave and its parameters can be found in section 3.2.

Furthermore, the turbine is very complex and involves many unknowns. Either a very simple model can be employed or the turbine can be left out altogether. For this thesis, the latter approach is used due to lack of significant knowledge about the operation of an air turbine. This way, the different OWC designs also become more comparable, since the turbine influences the pressure inside the chamber and therefore also the air flow and final energy output.

What is left in the simplified model is a two–phase fluid simulation, wave generation, and parameters describing the geometry of the Oscillating Water Column. Thus, we know the input, namely the parametrization of the WEC, and the setup for the CFD simulation. What is left to determine is the desired output of the simulation.

A typical CFD simulation delivers information about the flow, more precisely the flow velocity $\boldsymbol{u}(\boldsymbol{x}, t, \boldsymbol{s}) \in \mathbb{R}^3$ and the pressure $\boldsymbol{p}(\boldsymbol{x}, t, \boldsymbol{s}) \in \mathbb{R}^3$ over time. These are the unknown variables of the Navier–Stokes equations, described in chapter 5. They both depend on space ($\boldsymbol{x}$), time (t) and the d–dimensional parameter domain $\boldsymbol{s} \in \mathcal{D} \subset \mathbb{R}^d$ describing the OWC design. The question is which information we can extract from these to determine the energy output of the device.

A common approach in engineering would be to determine the average flow through the turbine and match it to a characteristic curve which ensures maximal efficiency of the turbine. A turbine needs a certain amount of air speed in order to operate and produce electricity. Since the air flows in and out, there are intervals of low to zero air flow, which need to be decreased, while times with greater flow should increase.

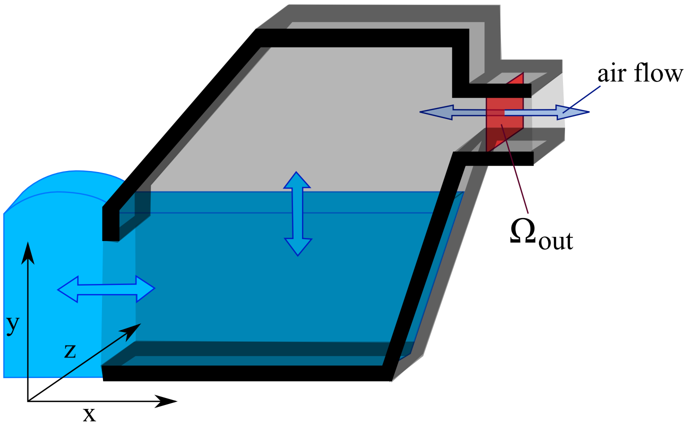


Fig. 3.1.1: Cutting plane Ω_{out} through turbine orifice to measure the average air flow.

In our case, only the flow in the orifice, where the turbine would be, can be measured. Given the air flow velocity $\boldsymbol{u}(\boldsymbol{x}, t, \boldsymbol{s}) = \boldsymbol{u}(\boldsymbol{s}) = (u(\boldsymbol{s}), v(\boldsymbol{s}), w(\boldsymbol{s}))$ from the simulation, the average air flow through a cutting plane $\Omega_{out} \subset \mathbb{R}^2$ of the opening, c.f. figure 3.1.1, can be determined by

$$\int_{\Omega_{out}\times[t_1,t_2]} |u(\boldsymbol{s})|\, dy\, dz\, dt. \tag{3.1.1}$$

If a characteristic curve $c(t)$ were known, the optimization problem would read

$$\min_{\mathcal{D}} \int_{\Omega_{out}\times[t_1,t_2]} |u(\boldsymbol{s})|\, dy\, dz\ - c(t)\, dt. \tag{3.1.2}$$

However, $c(t)$ depends on the turbine in use and we do not know the exact shape of such a curve. Therefore, this idea is dismissed for the use in this thesis. Instead, the average air flow (3.1.1) itself is to be maximized.

3.2 OWC design parametrization

The last step is to define a parametrization of the WEC geometry. To this end, consider the devices described in section 2.4.2. For comparison reasons and to get an idea of the actual size, we chose five devices, that have actually been built and have been in use for testing purposes. With each parameter the dimension d of the parameter domain $\mathcal{D} \subset \mathbb{R}^d$ and thus the complexity of the optimization problem increases. Therefore, the number of parameters should be chosen carefully. Only as many as necessary should be considered to avoid the *curse of dimensionality* as much as possible.

Some characteristics are crucial, such as the height, length and width of the construction. Others might have a smaller influence, for example the lip depth below the mean water level, which mainly depends on the wave climate (which is fixed in our scenario). Also, not all measurements of the power plants are known, leaving some dimensions to be guessed by sense of proportion.

Some measurements can be found in [Webb et al., 2005], namely the total width, chamber width, lip depth below mean water level, total height above mean water level and the turbine diameter. Taken these into account, the following eleven parameters are considered for fixed mean water level (MWL= $15.0\,m$) and inner air shaft height above mean water level ($h = 3.0\,m$):

a the height above MWL,

b the lip depth below MWL,

c the inner chamber length,

α the vertical lip angle α with associated length $d(\alpha)$, i.e.

$$d(\alpha) = \begin{cases} (a+b)\frac{\sin(\alpha)}{\cos(\alpha)} & \text{if } 0 < \alpha < 90 \\ 0 & \text{if } \alpha = 0 \end{cases},$$

β the inner vertical chamber angle β with associated length $e(\beta)$, i.e.,

$$e(\beta) = \begin{cases} (\mathrm{MWL}+h)\frac{\sin(\beta)}{\cos(\beta)} & \text{if } 0 < \beta < 90 \\ 0 & \text{if } \beta = 0 \end{cases},$$

f the chamber width and

g the inner air shaft length.

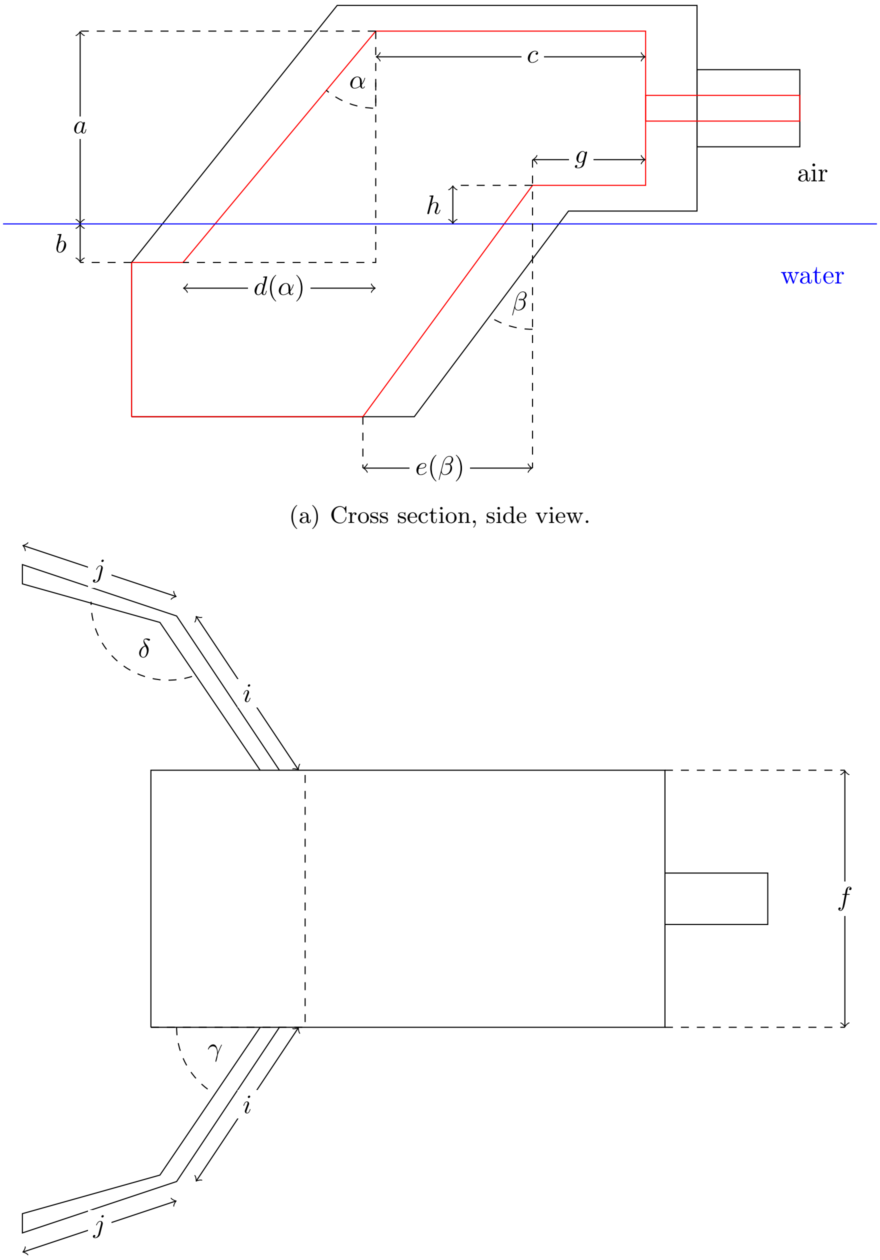
i, j lengths of the two–part harbor walls in front of the device

γ, δ angles of the two–part harbor walls

Furthermore, the position, as well as width, height and length of the air shaft are fixed for comparison reasons. The final parametrization is depicted in figure 3.2.1 and the values for the five reference OWC devices (figure 2.4.2) are summarized in table 3.2.1. Note, that the values of c, g and any measurements of the harbor walls, i.e. i, j, γ, δ, are unknown to the author and can only be guessed from pictures and descriptions.

Table 3.2.1: Parameter values for the five real OWC devices [Webb et al., 2005].

	a	b	c	α	β	f	g	i	j	γ	δ
LIMPET	12.8 m	2.3 m	15.0 m	50°	50°	6.0 m	10.0 m	0 m	0 m	0°	0°
Pico	15.0 m	2.5 m	10.0 m	30°	0°	12.0 m	−10.0 m	0 m	0 m	0°	0°
Port Kembla	12.0 m	3.0 m	10.0 m	45°	0°	10.0 m	5.0 m	8 m	8 m	45°	30°
Vizhinjam	14.5 m	1.1 m	15.0 m	0°	0°	10.0 m	0.0 m	14 m	14 m	0°	0°
Sakata	12.5 m	3.0 m	15.0 m	45°	0°	5.0 m	0.0 m	0 m	0 m	0°	0°



(a) Cross section, side view.

(b) Bird's eye view.

Fig. 3.2.1: Parametrization of the OWC geometry.

Table 3.2.1 describes the values for the five reference OWC, denoted by $\mathcal{D}_{ref} \subset \mathcal{D}$, and defines the bounds for the parameter domain $\mathcal{D}$. Hence,

$$\begin{aligned}
\mathcal{D} = \{s_1, \ldots, s_11 | lb_i \leq x_i \leq ub_i, i = 1, \ldots, 11\} \\
= \{12.5 \leq a \leq 15.0 \\
3.0^* \leq b \leq 4.0^* \\
10.0 \leq c \leq 15.0 \\
0.0 \leq \alpha \leq 50.0 \\
0.0 \leq \beta \leq 50.0 \\
5.0 \leq f \leq 12.0 \\
-10.0 \leq g \leq 10.0 \\
0.0 \leq i \leq 14.0 \\
0.0 \leq j \leq 14.0 \\
0.0 \leq \gamma \leq 45.0 \\
0.0 \leq \delta \leq 30.0\}
\end{aligned}$$

Note, that due to a different wave average than at the constructions sites, the values of b were changed to ensure that the lip remains under water at all times during the simulation. Furthermore, due to computational complexity and limited resources, two cases will be considered in Chapter 6 with a subset if these parameters. First, a two–dimensional test case with varying height a and angle α and finally a six–dimensional example with fixed width and without the harbor walls.

Now we can also define the volume of an OWC device using the parameters above. Since the mean water level MWL= $15.0\,m$ and the inner air shaft height $h = 3.0\,m$ above mean water level is fixed, we get the volume $V(\boldsymbol{s})$ of an arbitrary wave energy plan parametrized by $\boldsymbol{s}$ and the average volume of the five reference OWC V_{ref} defined by

$$V(\boldsymbol{s}) = f((a + 15)c + 0.5(a + b)d(\alpha) + (15 - b)d(\alpha) - 19g - (15 + h)e(\beta)) \qquad (3.2.1)$$

$$V_{ref} = \frac{1}{5} \sum_{\boldsymbol{s} \in \mathcal{D}_{ref}} V(\boldsymbol{s}). \qquad (3.2.2)$$

Finally, the wave climate defined by significant wave height and period plus water depth at the OWC lip as given in [Webb et al., 2005] for each of the reference constructions, can be used to define an average target wave. The height is about 2.5 meters and the wave period around 7.4 seconds. The mean water level is fixed at 15 meters.

3.3 Constraints

Next, we need to consider any constraints on the input variables. First of all, the parameter domain $\mathcal{D}$ should be bounded in order to find a global optimum of the objective function, i.e.

$$\mathcal{D} = \{s_1, \dots, s_d | lb_i \leq s_i \leq ub_i, i = 1, \dots, d\}.$$

Furthermore, the volume of the OWC device should be bounded. Otherwise, the largest designs will be favored and the simulations become (at least partially) redundant. Defining a reference average volume V_{ref} (see equation (3.2.2) below) and the volume $V(\boldsymbol{s})$ (equation (3.2.1)) of the parametrized OWC for each simulation, we get the condition $|V(\boldsymbol{s}) - V_{ref}| \leq \delta$ for some $\delta > 0$. This constraint can be added to the problem formulation by a penalization term.

In order to find an optimum *inside* the domain, another penalization term is added at the boundary. For this purpose, we define the d–dimensional flattened Gauss function $f : [-1, 1]^d \to \mathbb{R}^d$ by

$$f(\boldsymbol{x}) = 1 - \exp(-\sum_{i=1}^{d} x_i^{30}).$$

Then, the boundary penalization can be expressed in terms of $f(t(\boldsymbol{s}))$, where $t : \mathcal{D} \to [-1, 1]^d$ is the transformation from $\mathcal{D}$ to the unit–hypercube.

Hence, the objective function $F : \mathcal{D} \to \mathbb{R}$ including the two penalization terms is defined by

$$F(\boldsymbol{s}) = \int_{\Omega_{out} \times [t_1, t_2]} |u(\boldsymbol{s})| \, dy \, dz \, dt. - \lambda_1 |V(\boldsymbol{s}) - V_{ref}| - \lambda_2 f(t(\boldsymbol{s})) \tag{3.3.1}$$

for some constants $\lambda_1, \lambda_2 > 0$.

Additionally, there needs to be a constraint on the angles α and β to ensure there is an opening at the entrance of the device. If β is large and α is small, the two angled walls cross and leave no opening. Therefore, the constraint $c + d(\alpha) > e(\beta) + g + 4.0$ is applied to ensure a horizontal distance of the end of the lip to the inner angled wall of at least 4.0 meters. This is constraint is enforced in two ways. First of all, the values of $F(\boldsymbol{s})$ are set to $-\infty$ on the subset $\{\boldsymbol{s} \in \mathcal{D} | c_{\boldsymbol{s}} + d(\alpha_{\boldsymbol{s}}) > e(\beta_{\boldsymbol{s}}) + g_{\boldsymbol{s}} + 4.0\}$. Secondly, the objective function is not sampled on this subset, c.f. section 4.4.1.

3.4 Problem formulation

Taking all the considerations above into account, we can formulate the optimization problem as a blackbox optimization problem. In this scenario, the Navier–Stokes solver, defining $\boldsymbol{u}(\boldsymbol{s}) = \mathrm{NaSt}(\boldsymbol{s})$, is the blackbox, i.e. an unknown function to be evaluated, c.f. figure 3.4.1. The full problem then reads

Find the maximum $s^* \in \mathcal{D}$ of $F : \mathcal{D} \to \mathbb{R}$ defined by

$$s^* = \arg\max_{\mathcal{D}} F(\boldsymbol{s}) = \int_{\Omega_{out}\times[t_1,t_2]} |u(\boldsymbol{s})|\, dy\, dz\, dt. - \lambda_1 |V(\boldsymbol{s}) - V_{ref}| - \lambda_2 f(t(\boldsymbol{s})) \tag{3.4.1}$$

subject to

$$\boldsymbol{u}(\boldsymbol{s}) = \mathrm{NaSt}(\boldsymbol{s}) \tag{3.4.2a}$$

$$\mathcal{D} = \{s_1, \ldots, s_d | lb_i \leq x_i \leq ub_i, i = 1, \ldots, d\} \tag{3.4.2b}$$

$$c_{\boldsymbol{s}} + d(\alpha_{\boldsymbol{s}}) > e(\beta_{\boldsymbol{s}}) + g_{\boldsymbol{s}} + 4.0. \tag{3.4.2c}$$

Hence, we have a

- high dimensional,
- bounded,
- constrained,
- expensive blackbox

optimization problem.

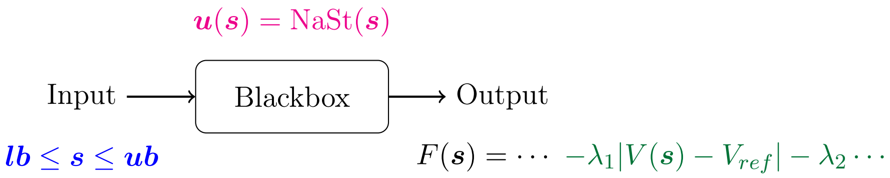

$\boldsymbol{u}(\boldsymbol{s}) = \mathrm{NaSt}(\boldsymbol{s})$
Input
Blackbox
Output
$\boldsymbol{lb} \leq \boldsymbol{s} \leq \boldsymbol{ub}$
$F(\boldsymbol{s}) = \cdots -\lambda_1 |V(\boldsymbol{s}) - V_{ref}| - \lambda_2 \cdots$

Fig. 3.4.1: Blackbox optimization.

4 Global Optimization

The following chapter describes general *global* optimization techniques to solve problems as described in the previous chapter, the difficulties observed in this field and different approaches to deal with these.

In general, global optimization refers to the minimization (or maximization, which is equivalent to the minimization of $-F$) of a function $F : \mathbb{X} \to \mathbb{Y} \subseteq \mathbb{R}$, called *objective function*. Here, $\mathbb{X} \subset \mathbb{R}^d$, often called *feasible region*, is some bounded set. Note, that $\mathbb{Y}$ can also be higher dimensional, making it a *multiobjective* optimization problem and some of the methods mentioned in this chapter can also be used in this case. However, we are going to concentrate on the single–objective scenario $\mathbb{Y} \subseteq \mathbb{R}$.

4.1 Difficulties in global optimization

An element $\boldsymbol{x}^* \in \mathbb{X}$ is said to be a *global* minimum of the objective function F, if and only if

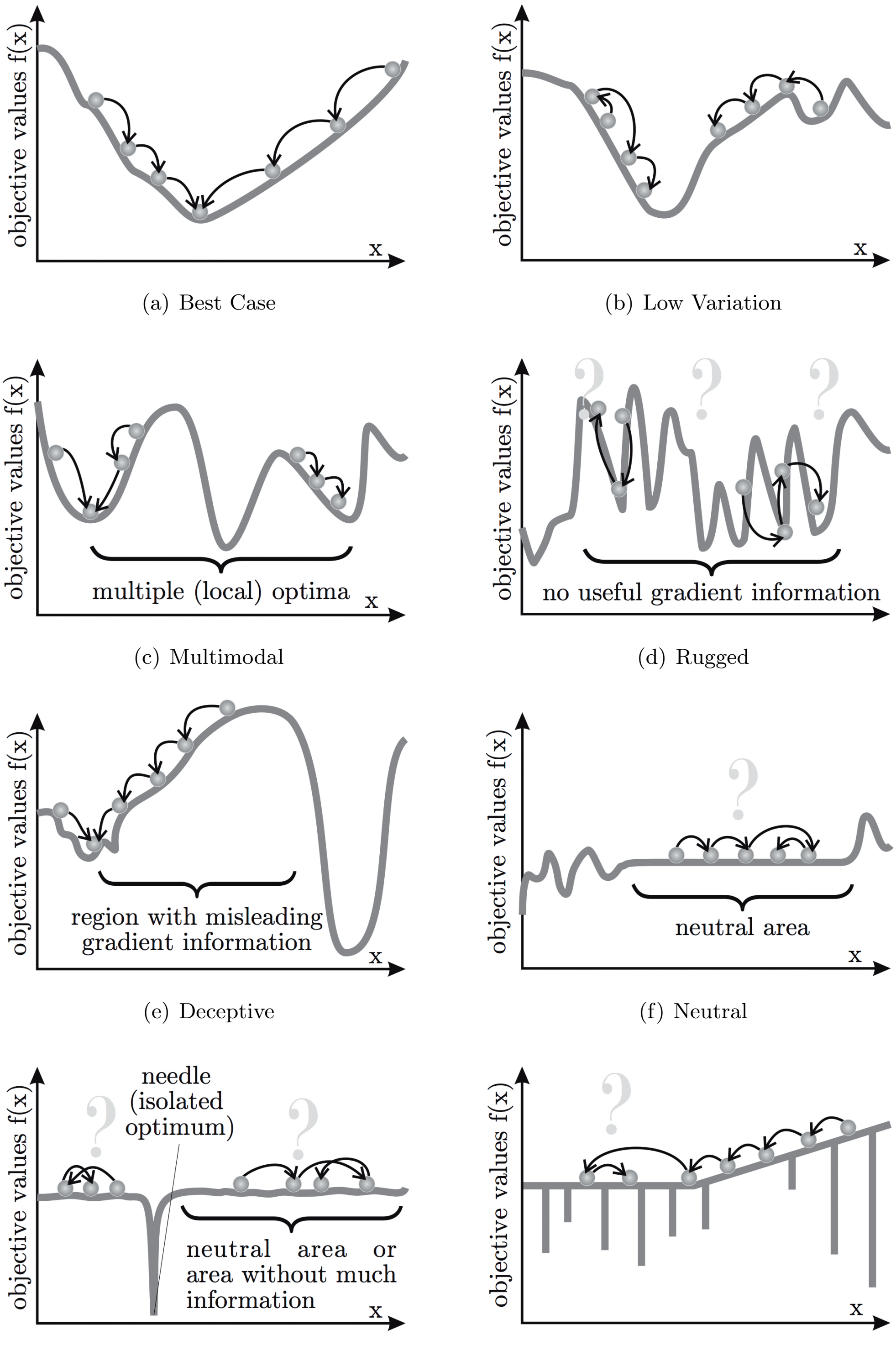
$$F(\boldsymbol{x}^*) \leq F(\boldsymbol{x}) \forall \boldsymbol{x} \in \mathbb{X}.$$

However, this is not always possible and sometimes only *local* minima can be found, i.e. elements $\boldsymbol{x}_l^* \in \mathbb{X}$ such that there exists an $\epsilon > 0$ with

$$F(\boldsymbol{x}_l^*) \leq F(\boldsymbol{x}) \forall \boldsymbol{x} \in \mathbb{X}, \|\boldsymbol{x} - \boldsymbol{x}_l^*\| < \epsilon.$$

Many nonlinear functions observe multiple local and/or global minima and maxima in its domain $\mathbb{X}$, which can be hard to detect. Many classical algorithms focus on local optimization and tend to "get caught" in local minima. For this reason a global method should have a good balance between local and global optimization.

Figure 4.1.1 demonstrates a number of possible scenarios faced when minimizing a function. The bubbles represent the possible candidates for a minimum found by a typical (search) algorithm [Weise, 2009]. The best case scenario (figure 4.1(a)) shows a *convex* function, which has only a single global optimum by its nature, that can be found by basically any optimization method. There are methods to transform a non–convex problem into a convex one by a *convexification*, which are however not discussed here. Figure 4.1(b) shows an example of a non–convex function with low variation in the neighborhood of the local optimum. Even though an algorithm might find the local minimum first, the global one can still be attained with a moderate number of function evaluations/iterations. This task becomes more difficult with multiple (local) minima and more variation, c.f. figure 4.1(c). This situation is even more complicated in a scenario as in figure 4.1(d), where


objective values f(x)
x
multiple (local) optima
no useful gradient information
region with misleading gradient information
neutral area
needle (isolated optimum)
neutral area or area without much information
?

(a) Best Case

(b) Low Variation

(c) Multimodal

(d) Rugged

(e) Deceptive

(f) Neutral

(g) Neelde–In–A–Haystack

(h) Nightmare

Fig. 4.1.1: Different possible properties of an objective function (minimization) [Weise, 2009].

the information of the gradient leads to jumps between different local optima. Similarly, the gradient in figure 4.1(e) is misleading and leads to a local minimum instead of the global optimum. Even less helpful is the gradient in flat areas as depicted in figure 4.1(f) and figure 4.1(g). Additionally, the only (global) minimum might *isolated*, i.e. in rapidly decreasing in a very small interval, which can be easily overlooked, c.f. figure 4.1(g). Finally, the possible worst case scenario is a combination of all these problems and can be seen in figure 4.1(h).

The “difficulty” of a problem correlates to the number of function evaluations necessary to find the optimum, i.e. its computational complexity [Weise, 2009]. This is especially important if the function to be evaluated is computationally expensive. Therefore, the goal is to find a reliable method with as little function evaluation as possible. As can be seen from the examples above, it is almost impossible to find an algorithm that works well for *any* kind of problem. Thus, it can be helpful to know as much as possible about the objective function and choose the optimization techniques accordingly.

4.2 Blackbox optimization

However, there are cases, where essentially no information about the function is given. For example, if the function cannot be represented explicitly by some formula and can be treated as a blackbox. This is the case, where the objective function depends on the output of a computer simulations — like the Navier–Stokes solver employed in this thesis. Merely heuristics can by applied to predict how the function might behave to choose an appropriate optimization algorithm.

It should be noted, however, that there exist other approaches, which aim to derive the blackbox function numerically and thus can use the classical framework of gradient based methods. Nevertheless, this is not the focus of this thesis, as we are interested in a metamodel to describe the behavior of the unknown function.

4.3 Derivative–free optimization

Since not much is known about the objective function, classical optimization algorithms, that use information such as the derivative to find the optimum, are not feasible. However, there exist a vast amount of derivative–free optimization algorithms. A good overview can be found in [Rios and Sahinidis, 2012], where both local and global methods are discussed along with different implementations. Following, some global methods are discussed.

4.3.1 Deterministic search algortihms

An advantage of deterministic algortihms is that no multiple runs have to be performed, as the outcome will always be the same.

Lipschitzian optimization is concerned with the global optimization by constructing a piecewise linear underestimator of the function. A function $f : D \subset \mathbb{R}^d \to \mathbb{R}$ is called

Lipschitz continuous, if there exists a constant $L > 0$, such that

$$|f(x) - f(y)| \leq L\|x - y\| \text{ for } x, y \in D.$$

Shubert's algorithm

Assuming the Lipschitz constant is known, [Shubert, 1972] considered the one–dimensional minimization problem

$$\Phi = \min_{x \in D} \ f(x),$$

where f is Lipschitz continuous and $D = [a, b] \subset \mathbb{R}$ is a closed interval. Substituting the interval bounds into the definition of Lipschitz–contiunity results in the two conditions

$$f(x) \geq f(a) - L(x - a), \tag{4.3.1a}$$

$$f(x) \geq f(b) + L(x - b) \tag{4.3.1b}$$

for $f(x)$ with $x \in [a, b]$. This defines the underestimator, which has one extreme point

$$M_0 = \max_{x \in [a,b]} \ \min_{x_k = a,b} \ f(x_k) + L|x - x_k|,$$

c.f. figure 4.1(a). The new point divides the interval $[a, b]$ into two subintervals, which define a new piecewise underestimator. The minimum of this new estimator defines the next point of the iteration and so on. The algorithm is summarized in algorithm 1 and illustrated in figure 4.3.1.

Algorithm 1 Shubert's algorithm

```
Require: Function $f$, interval bounds $a, b$, Lipschitz constant $L > 0$, $\epsilon$
$F_0 = \min_{x_k=a,b} \ f(x_k) + L|x - x_k|$
$M_0 = \max_{x \in [a,b]} \ F_0$
$x_1 = \arg\max_{x \in [a,b]} \ F_0$
$\Phi_0 = f(x_0)$
$n = 0$
while $M_n - \Phi_n \geq \epsilon$ do
    $F_{n+1}(x) = \min_{k=0,\ldots,n} \ f(x_k) + L|x - x_k|$
    $M_{n+1} = \max_{x \in [a,b]} \ F_{n+1}$
    $x_{n+1} = \arg\max_{x \in [a,b]} \ F_{n+1}$
    $\Phi_{n+1} = \max_{k=0,\ldots,n} \ f(x_k)$
    $n = n + 1$
end while
```

Some advantages of the algorithm include the use of very few parameters, i.e. only the Lipschitz constant L has to be known. This is also gives a bound on the error $\Phi - \Phi_n$, since $M_n \geq \Phi \geq \Phi_n$ and therefore the difference

$$\Delta_n = M_n - \Phi_n$$

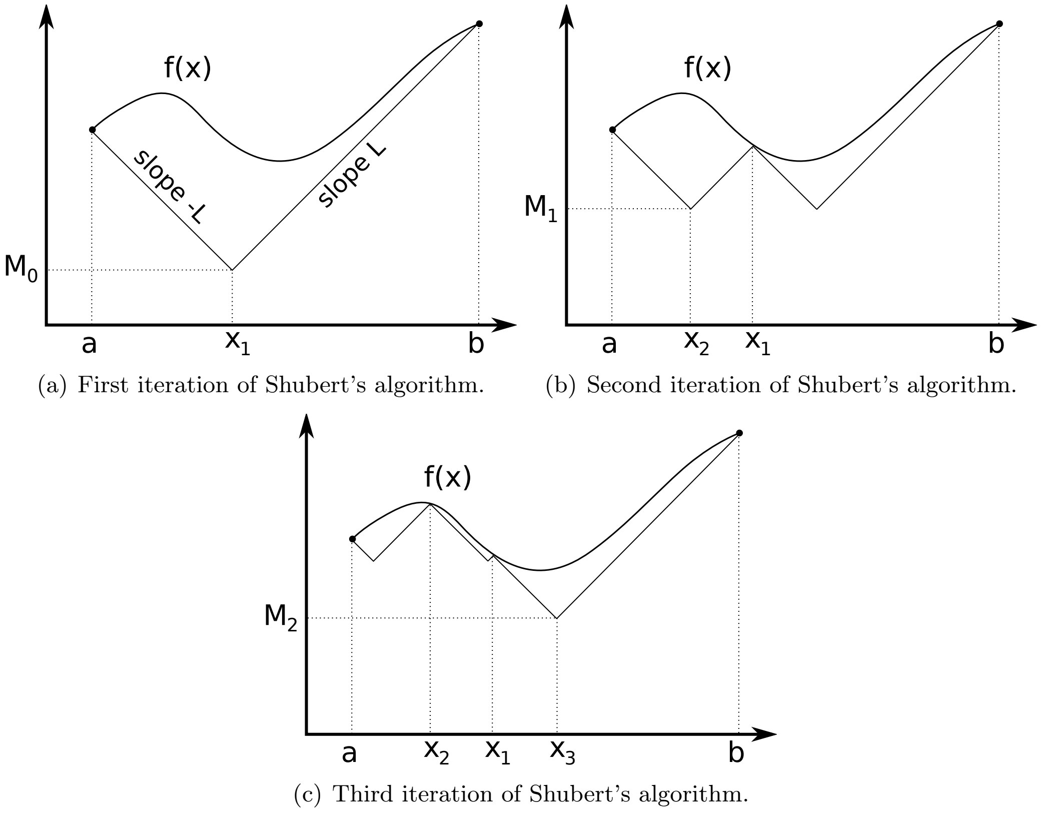
is an upper bound to the error and can be used as stopping criterion.

However, the Lipschitz constant, which balances the algorithm between local and global search, is not uniquely defined and if not much information about the objective function is given, it can be arbitrarily large. This can influence the speed of convergence severely. Also, the computational complexity grows exponentially with dimension and hence the method is not feasible in high dimensions.

DIRECT algorithm

An extension to Shubert's algorithm is the *DIRECT* method [Jones et al., 1993], where the name can be taken literal, as it is a *direct* search algorithm. Also, the initials stand for DIViding RECTangles, which describes how the method works. Since one major drawback of Shubert's algorithm is the lack of knowledge about the Lipschitz constant L, DIRECT uses *all* possible values of L, which also leads to a better balance between global and local search. The method was also designed for use in higher dimensions, i.e. the objective function lives in a d–dimensional domain $D \subset \mathbb{R}^d$, which consists of a hyper–rectangle $D = [A_i, B_i]_{i=1}^d$. For simplicity, the domain is normalized to the unit hypercube $\bar{D} = \{x \in \mathbb{R}^d : 0 \leq x_i \leq 1\}$.

The main idea is to sample the objective function at the center of hyper–rectangles. This solves the problem of computational complexity of Shubert's algorithm in the initialization



(a) First iteration of Shubert's algorithm.

(b) Second iteration of Shubert's algorithm.

(c) Third iteration of Shubert's algorithm.

Fig. 4.3.1: First three iteration steps of Shubert's algorithm, c.f. algorithm 1.

step, as only *one* point of the initial domain D has to be sampled, instead of all 2^d corner points. When dividing the rectangle, it has to be ensured, that no previously calculated function evaluations are lost, i.e. they should still be center points of some hyper–rectangle. Hence, the domain is divided into three subrectangles instead of two.

The first step of the algorithm is to evaluate the objective function at the center point $c_1 = (0.5, \ldots, 0.5) \in \mathbb{R}^d$ of the unit hypercube. Then the domain is to be divided into three equally sized hyper–rectangles. To this end, the function is sampled at the points $c_1 + \delta \boldsymbol{e}_i, \quad i = 1, \ldots, d$, where $\delta = 1/3$ is one third of the side length of the hyper cube and $\boldsymbol{e}_i$ is the i–th unit vector. To decide which dimension is to divided first, the best function values

$$w_i = \min(f(c_1 + \delta \boldsymbol{e}_i), f(c_1 - \delta \boldsymbol{e}_i)), \quad i = 1, \ldots, d$$

in each dimension are calculated. The dimension with the smallest value w_i is then divided into thirds. Next, the other dimensions are divided accordingly, leaving the new points at the center of new hyper–rectangles, c.f. figure 4.3.2. This way the best function values are left in the largest space, which leads to a preferred search in these rectangles leading to a slight emphasis on the local search. Each hyper–rectangle is only partitioned along its long dimensions. For example, considering a flat hyper–rectangle with measurements $5 \times 6 \times 1$, would be divided in its longer dimensions x_1 and x_2, but not in the third dimension x_3. This ensures, that the rectangles shrink after each iterations, which is essential for the convergence of the algorithm.

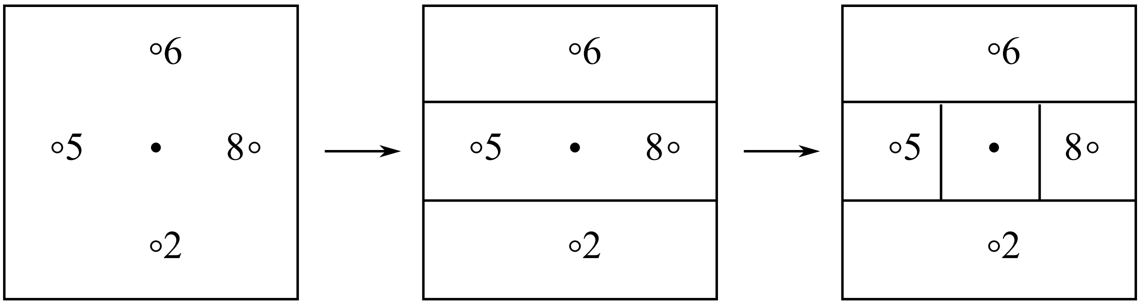


Fig. 4.3.2: Division of rectangles in one DIRECT iteration [Jones et al., 1993].

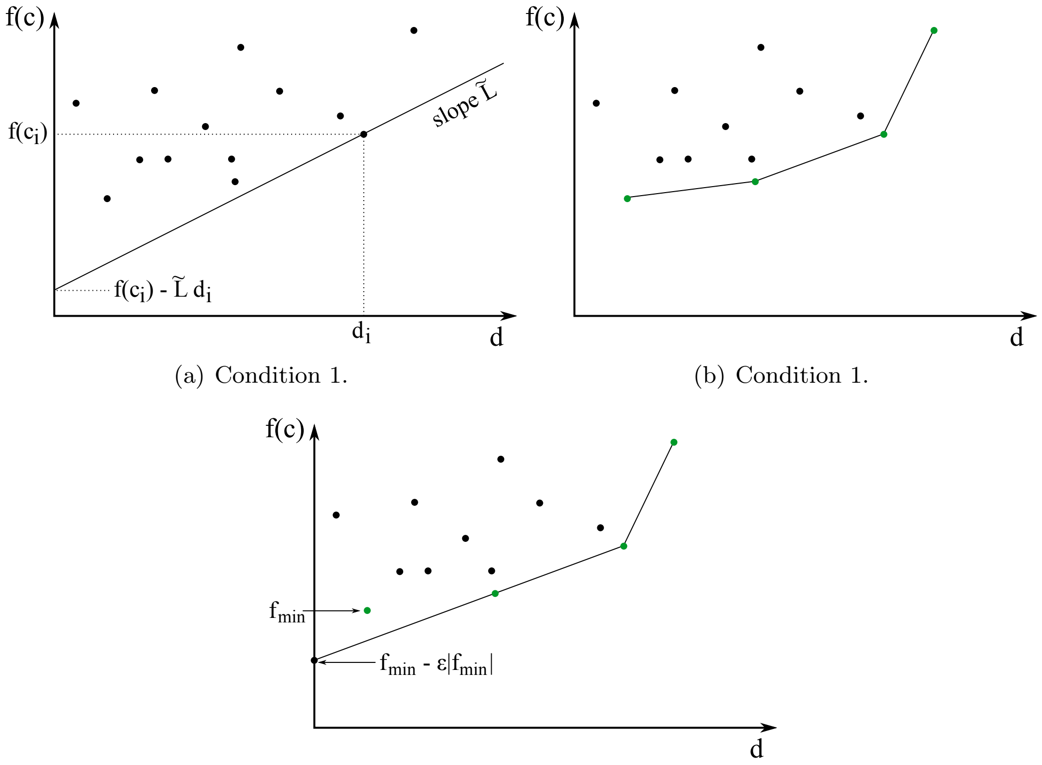
This process in then iterated, but no for all hyper–rectangles. Instead, the so–called *potentially optimal* hyper–rectangles are identified and further divided in each iteration. The definition of these rectangles is motivated by the following idea (for simplicity assume $d = 1$). Suppose the domain has already been partitioned into subintervals $[a_i, b_i], i = 1, \ldots, m$. Figure 4.3(a) depicts each division by a dot with the coordinates $(d_i, f(c_i))$, where $d_i = (b_i - a_i)/2$ is the distance of the inteval's center point c_i to its vertices. Since the points c_i always remain in the center of the interval, the value of d_i captures how well the domain has been searched in the interval with respect to global search. The value of the evaluated function values $f(c_i)$ on the other hand, i.e. the vertical coordinate, represents the local search. Any line with slope $L > 0$ through the point $(d_i, f(c_i))$ will define a lower bound for the corresponding interval at the vertical intercept. Like in Shubert's algorithm, the Lipschitz constant L determines the balance between local and global search. If its value is too high, the global search will dominate, if it is too low, the search will be very local. To overcome these difficulties, all possible values of $L > 0$, that define a lower bound of an interval are selected, c.f. figure 4.3(b). The corresponding intervals are the potentially optimal hyper–rectangles. In the general d–dimensional case, the definition is the following. A hyper–rectangle is said to be potentially optimal, if there exists a constant $\tilde{L} > 0$, such that

$$f(c_j) - \tilde{L} \cdot d_j \leq f(c_i) - \tilde{L} \cdot d_i, \quad \forall i = 1, \ldots, d, \tag{4.3.2a}$$

$$f(c_j) - \tilde{L} \cdot d_j \leq f_{min} - \epsilon |f_{min}|, \tag{4.3.2b}$$

where c_j is the center point of the rectangle j and d_j is a measure for the hyper–rectangles (e.g. the distance of c_j to the edges of j). The parameter ϵ ensures that $f(c_j)$ is greater than the current optimum. This ensures, that no unnecessary function evaluations are performed in a local search, when there is not a big improvement to be expected. Therefore, some smaller rectangles might not be selected as potentially optimal. Note, that $\tilde{L}$ is *not* the actual Lipschitz constant, but rather represents the rate of change.

Since the algorithm does not require to know a Lipschitz constant of the objective function, a different stopping criterion than in Shubert's method has to be used. In the



(a) Condition 1.

(b) Condition 1.

(c) Condition 2.

Fig. 4.3.3: Illustration of condition (4.3.2a) and (4.3.2b) for potentially optimal hyper–rectangles [Jones et al., 1993].

original version of DIRECT, a limit of the number of iterations has been proposed. Another criterion could be the maximal number of function samples, if the objective function is expensive to evaluate. Algorithm 2 summarizes the DIRECT method in d dimensions.

Algorithm 2 DIRECT algorithm [Jones et al., 1993]

```
Require: Function $f$, lower & upper bounds $\{A_i, B_i\}_{i=1}^d$, $maxits$, $maxevals$
Normalization: $\bar{D} = \{x \in \mathbb{R}^d : 0 \leq x_i \leq 1\}$
Initialization: $c_1 = (0.5, \ldots, 0.5), f_{min} = f(c_1), i = 0, m = 1$
while $it \leq maxits$ and $m \leq maxevals$ do
    Determine $S$ of all potentially optimal rectangles (according to (4.3.2a),(4.3.2b))
    for $j \in S$ do
        Determine longest side of rectangle $j$
        Evaluate $f$ at center points of new rectangles
        Divide $j$ into smaller rectangles:
    Identify set $I$ of dimensions with maximum side length $l_{max}$
    Set $\delta = 1/3 l_{max}$
    Evaluate $f(c_j \pm \delta \boldsymbol{e}_i) \quad \forall i \in I$
    Calculate $w_i = \min(f(c_j + \delta \boldsymbol{e}_i), f(c_j - \delta \boldsymbol{e}_i)) \quad \forall i \in I$
    Divide $j$ into thirds (dimesion with lowest value $w_i$ first)
        Update $f_{min}$
        $m = m + 2^d$
    end for
    $it = it + 1$
end while
```

4.3.2 Model–based search algorithms

Another approach is to construct a *surrogate–* or *metamodel* to get an approximation of the objective function, which gives information about the general behavior of the function and also permits to determine an (approximate) gradient or even higher derivatives. This model should capture the function as accurate as possible in order to produce valuable data concerning its local and global optima. To this end, the function has to be sampled within the given domain, covering as much of it as possible to give a high–fidelity model.

Both direct search methods as described above and the model–based approach have been considered for the use in this thesis. However, the latter was chosen, because it can be easily implemented to run fully parallel, whereas a direct search depends on the values calculated in the previous iteration steps. More details on model–based algorithms can be found in section 4.4.

4.3.3 Stochastic search algorithms

In contrast to deterministic search algorithms, stochastic methods involve a non–deterministic approach at some point in the algorithm. There is a vast amount of literature on this type of optimization methods, which include *Hit–and–run algorithms*, *simulated annealing*, *genetic algorithms* and *particle swarm algorithms* [Rios and Sahinidis, 2012].

4.4 Metamodeling

In order to optimize the energy output of the OWC device, we employ the methods of *Metamodeling*, which is very popular for engineering problems, where a process with multiple input variables and objectives has to be optimized.

A *metamodel* (or *response surface model* or *surrogate model*) $\hat{Y}$ is an approximation of the unknown response function $y = F(\mathrm{NaSt}(\boldsymbol{s}))$, where the error of the approximation is assumed to be due to model error as opposed to experimental noise, as we consider a deterministic computer simulation. Given the sample points $S \in \mathbb{R}^{d \times k}$ and the corresponding responses $Y \in \mathbb{R}^k$, the response surface can either be the result of an interpolation problem, i.e. functions that pass through the given points have to be found. Or the model can be non–interpolating, which means that it is obtained via a minimization of the deviations of the model $\hat{Y}$ from the unknown objective function y. In this case, low–order polynomials obtained by least square regression are a common choice. For interpolation response surfaces, radial basis functions or *Kriging*, which is described in more detail in section 4.4.2, are often used.

For this process, both the input parameters, called *factors* or *variables*, and the respective (computer) experiment results, called *responses*, are needed. The steps are usually as follows.

First, the *design variables* or *parameters*

$$x_i, i = 1, \ldots, d$$

have to be selected and if necessary, bounds and constraints need to be chosen. In the case of the OWC device, these are the height, width, depth, and so on, as described in section 3.2. This yields the *design space* or *parameter space*, which often consists of a hyper–rectangle

$$\mathcal{D} = \{\boldsymbol{x} \in \mathbb{R}^d | lb_i \leq x_i \leq ub_i, i = 1, \ldots, d\}$$

with appropriate lower and upper bounds lb_i and ub_i, respectively. For simplicity, this rectangle can be normalized to the d–dimensional unit hyper–cube $\hat{\mathcal{D}}$, such that

$$\hat{\mathcal{D}} = \{\boldsymbol{x} \in \mathbb{R}^d | -1 \leq \hat{x}_i \leq 1 \, i = 1, \ldots, d\}.$$

The variables and bounds need to be chosen carefully, keeping in mind what needs to be achieved. It is advisable to choose only as many input parameters as necessary, since each factor increases the complexity of the problem. If in doubt which factors are important and which might be neglectable, a screening process with less trial runs could be a good first step before running a complete experiment series.

Once the parameters are chosen, the *sample points* $s_1, \ldots, s_k \in \hat{\mathcal{D}}$ — also called *design sites* — can be selected. These are the points in the parameter space $\hat{\mathcal{D}}$, for which an experiment/simulation is run, i.e. a response is determined. This task is not quite simple, as it needs to be appropiate for the task at hand. One needs to keep in mind how the expected response might behave (e.g. linear or nonlinear) and how well the parameter space has to be covered in order to estimate the response at untried points in the design space. These questions are incorporated in the so–called *Design of Experiments* (DoE). There are several approaches to select the sample points appropriatly, which are covered in more detail in section 4.4.1.

At the given sample points $s_1, \ldots, s_k \in \hat{\mathcal{D}}$, the response needs to be calculated. Since the subsequent optimization process does not depend on the evaluated function, this is called *blackbox* optimization. This means, that the objective function — which in our case depends on the Navier Stokes solver with the complete setup including the wave generation and the wave energy converter, which we denote by NaSt(), c.f. chapter 5 — does not have to be represented by some algebraic formula, but instead might be quite abstract. It is simply treated as a blackbox, which determines the desired responses $Y_i, i = 1, \ldots, k$ for given input variables, i.e.

$$Y_i = F(\mathrm{NaSt}(s_i)),$$

c.f. section 3.4.

The previous step is the most time and cost consuming, as each conducted experiment — in our case each simulation run — can be very expensive. This is why the process of DoE is very important.

Given the sample points and respective responses $(s_i, Y_i), i = 1, \ldots, k$, the optimization process begins. As mentioned before, a screening process and possible model reduction might be helpful to identify the important factors and reduce the design complexity. Following this step, a new, refined DoE and therefore additional blackbox evaluations might be useful for a more refined model.

The next step is to determine an appropriate *meta– or surrogate model.* This can be for example a response surface model calculated by Kriging, e.g. with the quite popular *Design and Analysis of Computer Experiments* (DACE) method [Sacks et al., 1989]. See section 4.4.2 for more details.

Finally, the calculated metamodel can be employed to estimate the optimum of the objective blackbox function. The steps of the optimization algorithm is summarized in algorithm 3.

Another approach to determine the optimum, is to simply select the optimal point among the calculated responses $(s_i, Y_i), i = 1, \ldots, k$, without any further calculations.

Obviously, all point evaluations to calculate the responses $(s_i, Y_i), i = 1, \ldots, k$ can run in parallel once the sample points are determined, which yields a perfectly scaling parallel algorithm with respect to point evalutation. But other adaptive algorithms, where the choice of new points depends on already calculated responses, have also shown to be be quite useful.

Algorithm 3 Design Optimization

```
Require: design variables/parameters $x_i, i = 1, \dots, d$, bounds $x_i^{lower} \leq x_i \leq x_i^{upper}, i = 1, \dots, d$ for each parameter, number of desired sample points $K, L$, blackbox function $\mathbf{f}$
Normalize parameter space $\mathcal{D}$ to $\hat{\mathcal{D}}[-1,1]^d$
Initialize $k := 0, l := 0, j := 0$
while $k \leq K$ do
    Determine sample points $s_1, \dots, s_{k_j} \in \hat{\mathcal{D}}, k_j \leq K - k$
    Calculate responses $Y_i = \mathbf{f}(s_i), i = 1, \dots, k_j$
    Set $k = k + k_j, j = j + 1$
end while
Postprocessing: determine metamodel and find optimum of metamodel
Screening process & parameter space reduction (optinal)
while $l \leq L$ do
    Determine sample points $s_1, \dots, s_{l_j} \in \hat{\mathcal{D}}, l_j \leq L - l$
    Calculate responses $Y_i = \mathbf{f}(s_i), i = 1, \dots, l_j$
    Set $l = l + l_j, j = j + 1$
end while
Postprocessing: determine refined metamodel and find optimum of metamodel (optional)
```

4.4.1 Design of Experiments (DoE)

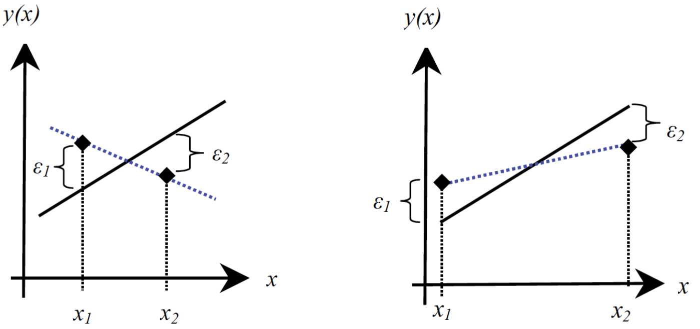
As mentioned before, the choice of sample points $s_i \in \hat{\mathcal{D}}, i = 1, \dots, k$, called the *design* of the experiment, is essential to obtain an accurate model while conducting only few simulation runs. Therefore, the DoE step is very important to maintain both the precision and the efficiency of the method. Following, a short overview of common experimental designs is given. Given the development of DoE, these can be categorized into *classical* and *modern* designs, where the difference lies in the conducted experiments — lab/field experiments vs. computer experiments [Giunta et al., 2003]. In both classical and modern designs, it is essential to choose as few sample points as possible to cut the cost of the experiment.

4.4.1.1 Classical DoE

DoE was initially developed for field or lab experiments, where the measured response Y_i is assumed to contain a random error term ϵ due to measurement error for example, i.e.

$$Y_i = \mathbf{f}(s_i) + \epsilon_i \,, i = 1, \dots, k.$$

As a result of this error term, the calculated response may vary even if measured at the same sample point. Also, to minimize the effect of the random error, the sample points are usually chosen at the boundary of the design space. This ensures, that at least the right trend of the actual response can be approximated. A good illustration of this can be found in [Giunta et al., 2003], c.f. figure 4.4.1. It shows, that the effect of random error terms ϵ_1 and ϵ_2 can be lead to a wrong trend in the estimated model. In this case, it causes a negative slope in the estimated model, whereas the actual linear model has a positive slope. This effect can be reduced by moving the sample points to the boundary, as it is now easier to distinguish the influence of the response trend from the influence of the error terms. Therefore, classical designs have many boundary points and only few interior points, which are often repeated, i.e. more than one response is calculated to minimize the random error.



(a) Due to random error, the estimated linear model (dashed) shows a different trend than the true model (solid).

(b) Moving the sample points to the boundary reduces the effect of the random error.

Fig. 4.4.1: Effect of random error in classical DoE [Giunta et al., 2003].

A few common classical designs are summarized to give an overview and a better understanding of how and why we choose our final designs.

Factorial Designs Factorial designs are very popular in engineering and statistics, where factors are tested at different discrete *levels* $\boldsymbol{l} = (l_1, ..., l_d) \in \mathbb{N}^d$ and the combination of these should be tested as well, i.e. the correlation between the different factors. A *full factorial* design consists of all possible combination of these, whereas a *fractional factorial* design only incorporates some combinations. Hence, a full factorial design is simply a regular grid with mesh size $\boldsymbol{h_{l_k} = 2^{-l_k}}$ for the respective level $l_k, k = 1, \ldots, d$ in each dimension, i.e.

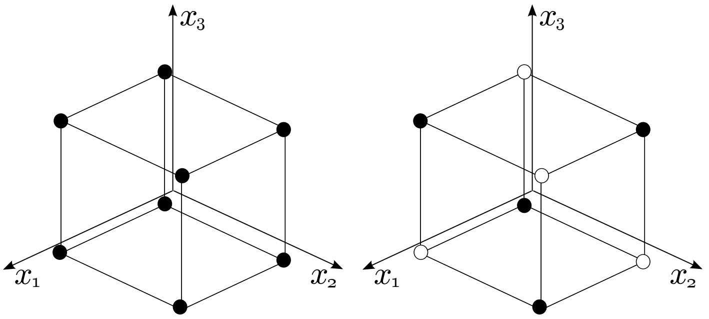
$$S_{FracFac} \subset S_{FullFac} = \{s_{i,l} \in \hat{\mathcal{D}} | x_{i_k,l_k} = \pm i_k \cdot h_{l_k} = \pm i_k \cdot 2^{-l_k}, i_k = 0, \ldots, 2^{l_k}\}$$

and

$$|S_{FullFac}| = \boldsymbol{l}^d, |S_{FracFac}| = \boldsymbol{l}^{d-p}.$$

For example, a full factorial of three factors with levels $\boldsymbol{l} = (2, 2, 2)$, called a 2^3 design, is commonly visualized as a cube, as seen in figure 4.2(a), where the vertices each represent a sample point, i.e. there are 2^3 sample points. This design evaluates the *main effects* and interactions between the factors. The term *effect* refers to the influence of a factor, i.e. how the response changes when the value of the input parameter varies. The influence of a single factor is called *main effect.* Increasing the number of levels will result in evaluation of higher order effects. In general, the full factorial design with the same number of levels L for each of the d factors consists of L^d sample points, i.e. the complexity increases exponentially with the number of factors. Hence, for a large number of factors, it might be too expensive to evaluate at all sample points.

An obvious solution to this problem is to evaluate only at some points of the full grid, resulting in a fractional factorial design, the most common being 2^{d-p} designs, where the fraction is equal to $1/2^p$, c.f. figure 4.2(b) for $p = 1$. One drawback of this, is of course the loss of accuracy. As some effects are not tested, the results are biased, which needs to be kept in mind when evaluating the model. Never the less, this design proves to be quite handy and is often used for a first *screening* step, where the most important factors are identified.



(a) 2^3 full factorial. (b) 2^{3-1} fractional factorial.

Fig. 4.4.2: Factorial designs for three factors with two levels each.

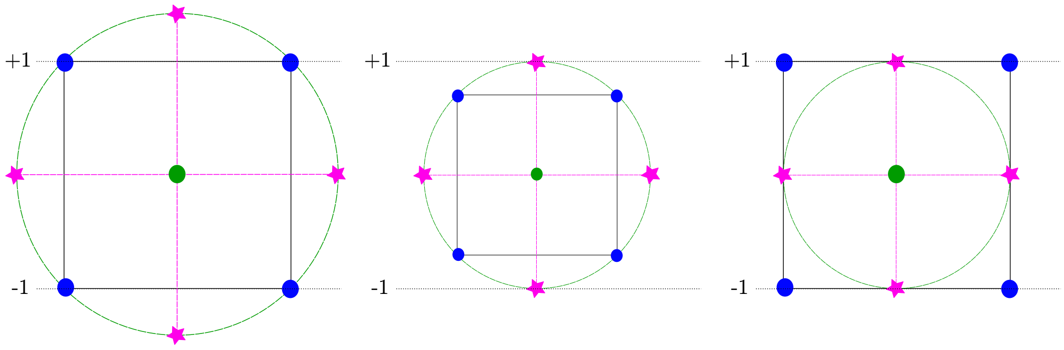
Central Composite Designs Quadratic effects can be estimated by the 3^d full factorial or an appropriate fractional factorial designs. However, the number of sample points can get out of hand quite quickly. A common second–order design, which is constructed to reduce the number of design points in this case, is the *Central Composite Design (CCD)* introduced by Box and Wilson [Box and Wilson, 1951]. It consists of a two–level factorial design (full or fractional) and additional axial points, called the *star points*, plus a certain number n of center points, i.e. the center point is evaluated more than once to reduce the impact of the error term ϵ. If the factorial design points are located at the vertices of a unit hyper–cube, the star points are located at $\pm\alpha$ for each factor, where $|\alpha| \geq 1$ depends on the desired properties of the chosen design. Hence

$$S_{CC} \subset \{s = (s_1, \dots s_d) \in \hat{\mathcal{D}} | s_j \in \{0, \pm 1, \pm\alpha\}, j = 1, \dots, d\}$$

and

$$|S_{CC}| = 2^{d-p} + 2d + n.$$

Figure 4.4.3 shows the three common designs and their relationship. The so–called *Circumscribed* (CCC) design is the original Central Composite Design, where the star points constitute new extremes for each factor; in figure 4.3(a) $\alpha = \sqrt{2}$. The *Inscribed* (CCI) design limits the star points by the original extremes and is therefore a scaled down version of the CCC design by the factor α. Finally, the *Face Centered* (CCF) design sets the star points at the faces of the factorial design space, i.e. $\alpha = \pm 1$. Note that the CCC investigates the largest design space, whereas the CCI the smallest space. The CCF explores the same design space as the underlying 2^d factorial design, but evaluates more sample points.



(a) CCC (b) CCI (c) CCF

Fig. 4.4.3: Different Central Composite Design for two factors. It is composed of the 2^2 factorial design points, additional star points and center points.

In general, α can be chosen, such that the experiment design is *rotatable*, i.e. the design can be rotated around the center point without changing the prediction variance at any point in the design space. Also, the value of α may help to achieve *orthogonality* of the design, i.e. that each factor can be evaluated independtly.

Box–Behnken Unlike CCD, *Box–Behnken* designs [Box and Behnken, 1960] do not incorporate a factorial design, but include the center points of the faces of the parameter space, i.e. the star points of the CCF design, and the center point. It is rotatable, but usually not orthogonal. It requires three levels for each factor and evaluates quadratic effects.

Placket–Burman Placket and Burman investigated a design [Plackett and Burman, 1946], which — similar to factorial designs — includes factors with a certain number of levels, but with a smaller number of experiments and such that the variance of the dependent variables would be minimized. These designs assume negligible interactions between the factors and focuses on a few important main effects. In general, main effects are heavily *confounded* (or *aliased*) with two–factor interactions, i.e. the response of the main effect is influenced by higher order interactions.

The method of finding the desired design works for a number of sample points, which are a multiple of four. However, if the number of points is a power of two, the design coincides with a fractional factorial design. Hence, the Placket–Burman designs usually consider a number, which is a multiple of four, but not a power of two, e.g. 12, 20, 24, 28, and so on.

4.4.1.2 Modern DoE

In contrast to classical DoE, the modern designs do not have to account for random error due to measurement or other external influences, since the conducted experiments are *deterministic* computer experiments. Therefore the repeated evaluation of a sample points does not make much sense, as the outcome will be (exactly) the same. Instead, the error term in equation (4.4.1.1) accounts for model error. Also, the sample points must not lie mostly at boundary to minimize the influence of the error term. Instead, it is assumed that the true trend of the response is unknown and therefore most points are placed in the interior of the design space in order to search the parameter space most thoroughly, i.e. one is looking for a *space filling* design. Also, the design should be *extendable*, so that additional design sites can be added at any point of the optimization algorithm. Of course, also the number of necessary experiments should be as low as possible to save valuable computing time. Hence, the design should also show a good computational *complexity.*

Random selection (Monte Carlo) One of the most obvious ways to select points in a domain (other than a regular grid, i.e. a full factorial design) is by *random sampling.* This Monte Carlo approach works quite well for a large number of points, but as one would like to keep the number of sampling points as low as possible, this might not be the smartest choice. Let N denote the number of sample points. Then the set of design sites is defined by

$$S_{MC} = \{s_i \in \hat{\mathcal{D}} | s_i (\text{pseudo–random}) \text{ number with normal distribution in } \hat{\mathcal{D}}, i = 1, \ldots, N\}.$$

The biggest problem is, that often large portions of the design space remain unexplored and therefore no information of the response is given in these areas. Hence, the Monte Carlo method is neither the best choice in terms of complexity, nor in terms of design space coverage. But is possible to add new points to the design at any time, i.e. it can be augmented, c.f. figure 4.4.5 for a random selection of 10 points and additional 90 points, i.e. a total of 100 points.

Quasi Random selection (Quasi Monte Carlo) This approach selects the points according to a Quasi Monte Carlo method [Morokoff and Caflisch, 1995], i.e. the points are generated according to a given low discrepancy sequence, which appears to be random, but is actually deterministic. However, this method can also be randomized. Popular sequences are *Van der Corput*, *Halton* or *Sobol* sequences.

The discrepancy refers to a measure of how much the distribution of the sample points differs from a uniform distribution. The discrepancy D_N of a sequence of N points in the unit hypercube $\hat{\mathcal{D}}$ is defined by

$$D_N = \sup_{Q \subset \hat{\mathcal{D}}} \left| \frac{\#\text{points in } Q}{N} - m(Q) \right|,$$

where Q is a rectangle in $\hat{\mathcal{D}}$ with sides parallel to the coordinate axes and with volume $m(Q)$. The discrepancy contributes to the error bound of Monte Carlo methods and should be therefore bounded from above. Low discrepancy sequences have a particularly low upper bound.

The Van der Corput sequence is defined by reversing the base representation of the set of natural numbers $\mathbb{N}$. More precisely, if $n \in \mathbb{N}$ is (uniquely) represented in base b by

$$n = \sum_{j=0}^{m} a_j(n) b^j,$$

where $a_i \in \{0, \ldots, b-1\}$, then its corresponding value in the Van der Corput sequence is

$$VC(n) = \sum_{j=0}^{m} a_j(n) b^{-j-1}.$$

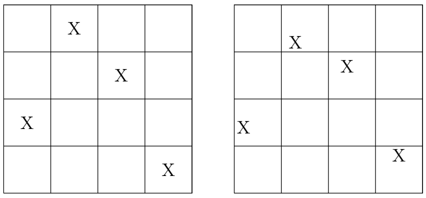
This sequence is defined for the interval $[0, 1]$ in one dimension. It can be extended to the interval $[-1, 1]$ and several methods to extend the sequence to higher dimesions, while keeping the low discrepancy, have been proposed.

One of them is the Halton sequence, which is a generalization of the one–dimensional Van der Corput sequence, where a different prime base is used in each dimension. However, this approach leads to a high correlation between the points in successive higher dimensions. Therefore, a sort of rearranged version of the Halton sequence, called *Faure* sequence, has been proposed. It uses the same base for each dimension, but rearranges the corresponding elements of the Van der Corput sequence in each dimension.

The Sobol sequence is another generalization of the Halton sequence. Similar to the Faure sequence, it uses the same base, namely base 2, for all dimensions, but the rearranging is more complicated. For details refer to [Galanti and Jung, 1997].

The set of Quasi Monte Carlo points ensure a good coverage of the design space due to the low discrepancy. The design also shows a better complexity than the random selection of points, but can be augmented just as easily. Figure 4.4.5 shows the first 10 and the next 90 to give a total of 100 points of the Sobol sequence. Note how the configuration resembles a rotated regular grid as the number of points gets larger.

Latin Hypercube Similar to the first designs, the *Latin Hypercube sampling* first proposed by [McKay et al., 1979] involves randomly selected points. But in this case, the points are not selected independently. Instead, each of the d one–dimensional subspace of the design space is divided into N equidistant intervals, where N is the number of sampling points and d the number of factors, i.e. the design space is divided into N^d bins. The points are then selected, such that each projection onto a one–dimensional subspace contains exactly one point in each bin. These points can either lie in the center of each bin (c.f. figure 4.4(a)) or at a random point inside the bin (c.f. figure 4.4(b)). This yields a uniform distribution for each one–dimensional projection. The complexity and design space coverage is similar to the Quasi Monte Carlo approach, but the addition of new points can be a challenge, as all points are chosen simultaneously. However, this is not a problem when a larger set of points is to be selected in each iteration of an algorithm. In figure 4.4.5, the first 10 points are randomly selected within $10^2 = 100$ bins (10 bins for each dimension) and to refine the design to give a total of 100 points, the domain is divided into $110^2 = 12100$ bins to get a Latin Hypercube sampling of the additional 90 points, where bins containing one of the previous 10 sampled points are rejected.

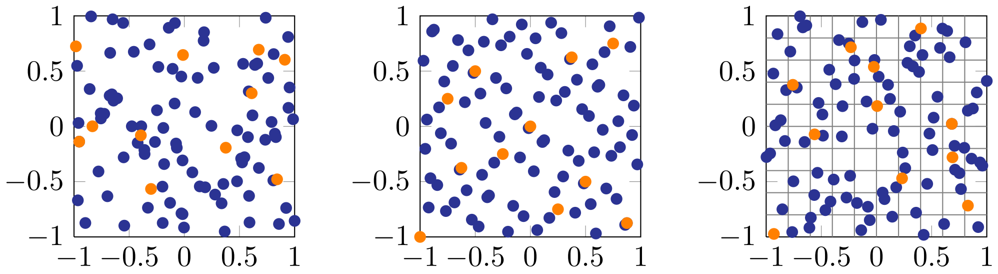


(a) centered (b) random

Fig. 4.4.4: Latin hypercube designs.

Orthogonal designs A generalization to Latin Hypercube sampling is *orthogonal sampling*, which requires, that the space should be covered evenly. This design is comprised of equally probable subspaces, which are sampled with the same density. A t-dimensional projection should show a uniform distribution, where t is called the strength of the design. As a special case, Latin Hypercube designs are orthogonal with strength 1. This is not easy to implement for $t \geq 2$, as sometimes orthogonality can only be achieved for a certain number of points and subspaces.

Sparse Grids In particular with regard to computational complexity in high dimensional space, *sparse grids* [Bungartz and Griebel, 2004] come to mind. These grids naturally cover



(a) Monte Carlo (b) Quasi Monte Carlo (c) Latin Hypercube

Fig. 4.4.5: Different random designs with refinement. Number of total sample points $N = 10, 100$ in the unit hypercube $[-1, 1]^2$. In the Latin Hypercube design, the bins for the sampling of 10 points are shown.

the whole design space and can be (adaptively) refined if necessary. They also show a good complexity. Sparse grid points are not very common in response surface methods, however there have been some works on this topic; see for example [Beena and Ganguli, 2010], where this approach has been employed to optimize various objective functions commonly used for testing global optimization algorithms.

Sparse grids are a special discretization method based on hierarchical basis functions, c.f. section 4.4.1.2 for more details. There are several ways to define a sparse grid, depending on the norm used to define the index set. The maximum– or L_2–norm grid S_{SGmax} with boundary points is defined as

$$S_{SGmax} = \bigcup_{|\boldsymbol{l}|_1 \leq n} (S^{l_1}_{\text{SGmax}} \times \cdots \times S^{l_d}_{\text{SGmax}}) \tag{4.4.1}$$

with the nested sets $S^{l}_{\text{SGmax}} \subset S^{l+1}_{\text{SGmax}}$ of level l defined by

$$S^{l}_{\text{SGmax}} = \{s_{l,i} \in \hat{\mathcal{D}} | s_{l,i} = \frac{i-1}{m_l - 1} forj = 1, \ldots, m_l, m_l = 2^l + 1\}. \tag{4.4.2}$$

Excluding the boundary points, it is defined as

$$S_{SGNB} = \bigcup_{|\boldsymbol{l}|_1 \leq n} (S^{l_1}_{\text{SGNB}} \times \cdots \times S^{l_d}_{\text{SGNB}}) \tag{4.4.3}$$

with the nested sets $S^{l}_{\text{SGNB}} \subset S^{l+1}_{\text{SGNB}}$ defined by

$$S^{l}_{\text{SGNB}} = \{s_{l,i} \in \hat{\mathcal{D}} | s_{l,i} = \frac{i}{m_l + 1} forj = 1, \ldots, m_l, m_l = 2^l - 1\}. \tag{4.4.4}$$

In this thesis, the latter approach is used due its lower complexity. For example, the first level of S_{SGmax} consists of 3^d points, i.e. 729 for $d = 6$ or 177147 for $d = 11$, whereas S_{SGNB} has only one.

Other sparse grids based on one–dimensional quadrature rules to construct the grid for numerical integration include the *Clenshaw–Curtis* rule, the *Gauss–Patterson* rule or *Chebychev* integration, c.f. [Gerstner and Griebel, 1998].

Select By Hand In addition to all those designs, it is possible to add some or even all sample points by hand. For example, if one wishes to determine a certain response for comparison reasons. In our case, these points are the parameter points of the five OWC constructions discussed in section 3.2, i.e. the points given in table 3.2.1.

Note, that for all designs, points in the subset $\{\boldsymbol{s} \in \mathcal{D} | c_{\boldsymbol{S}} + d(\alpha_{\boldsymbol{S}}) > e(\beta_{\boldsymbol{S}}) + g_{\boldsymbol{S}} + 4.0\}$ are neglected and not sampled, as described in section 3.3.

Four sets of sample points are selected for comparison and also combined for a more refined metamodel. These four selected designs are Monte Carlo, Quasi Monte Carlo, Latin Hypercube and Sparse Grid points.

Sparse Grids

Sparse grids, also known as hyperbolic cross points, discrete blending or Boolean interpolation, were originally designed for the interpolation of high dimensional functions, which suffer the *curse of dimensionality* — a term coined by [Bellman, 1961]. It refers to exponential dependence of classical discretization techniques on the dimension d, i.e. the number of grid points and therefore (expensive) function evaluations grows asymptotically $\mathcal{O}(2^{nd})$, c.f. the full factorial grid.

To fully understand this problem and find a solution, consider a function $f : U \subset \mathbb{R}^d \to \mathbb{R}$ and the corresponding interpolant u defined by

$$f(\boldsymbol{x}) \approx u(\boldsymbol{x}) = \sum_i \alpha_i \phi_i(\boldsymbol{x}),$$

with suitable basis functions $\phi_i(\boldsymbol{x})$ and coefficients α_i.

First of all, consider the problem in one dimension. For discretization on a full grid, the underlying approximation space is decomposed into subspaces. The basis of each subspace is based on the usual one–dimensional hat functions

$$\Phi(x) = \begin{cases} 1 - |x| & \text{if } x \in [-1, 1] \\ 0 & \text{else,} \end{cases} \tag{4.4.5}$$

which is used to generate a family of basis functions $\Phi_{l,i}$ defined by

$$\Phi_{l,i}(x) = \Phi\left(\frac{x - ih_l}{h_l}\right) \tag{4.4.6}$$

with support in $[x_{l,i} - h_l, x_{l,i} + h_l]$.

The hierarchical increment spaces are then defined by

$$W_l = \text{span}\{\Phi_{l,i} | i \in I_l\}$$

with the index set

$$I_l = \{i \in \mathbb{N} | 1 \leq i \leq 2^l - 1, i \text{ odd}\}.$$

The corresponding discrete function space V_l is then the direct sum of these subspaces, i.e.

$$V_l = \bigoplus_{k \leq l} W_k.$$

This yields the one–dimensional interpolant

$$f(x) \approx u(x) = \sum_{k=1}^{l} \sum_{i \in I_k} \alpha_{k,i} \Phi k, i(x),$$

which can be extended to higher dimensions via a tensor product construction. The basis functions are the product of the one–dimensional basis functions, i.e.

$$\Phi_{\boldsymbol{l},\boldsymbol{i}}(\boldsymbol{x}) = \prod_{n=1}^{d} \Phi_{l_n,i_n}(x_n)$$

with the corresponding index set

$$I_{\boldsymbol{l}} = \{\boldsymbol{i} \in \mathbb{N}^d | 1 \leq i_n \leq 2^{l_n} - 1, i_n \text{ odd}\}, 1 \leq n \leq d$$

and hierarchical subspaces

$$W_{\boldsymbol{l}} = \text{span}\{\Phi_{\boldsymbol{l},\boldsymbol{i}} | \boldsymbol{i} \in I_{\boldsymbol{l}}\}.$$

Hence, the d–dimensional interpolant of level n is defined by

$$f(\boldsymbol{x}) \approx u(\boldsymbol{x}) = \sum_{|l|_\infty \leq n} \sum_{\boldsymbol{i} \in I_{\boldsymbol{k}}} \alpha_{\boldsymbol{k},\boldsymbol{i}} \Phi_{\boldsymbol{k},\boldsymbol{i}}(\boldsymbol{x}).$$

So far, the full grid construction has been considered and the interpolant is constructed for $(2^n - 1)^d$ grid points (assuming $f|_{\partial U} = 0$). For sufficiently smooth functions, the asymptotic error decay is

$$\|f(\boldsymbol{x}) - u(\boldsymbol{x})\|_{L_2} \in \mathcal{O}(h_n^2),$$

but with $\mathcal{O}(h_n^{-d}) = \mathcal{O}(2^{nd})$ function evaluations, i.e. the computational cost increases exponentially with the dimension, which is known as the curse of dimensionality.

Sparse grids are designed to overcome this difficulty, i.e. the discretization uses less function evaluations, while maintaining the order of accuracy. The prerequisite is that the second mixed derivatives are bounded, i.e. the function belongs to the Sobolev space

$$H_2^{mix}(U) = \{f : U \to \mathbb{R} | D^{\boldsymbol{l}} f \in L_2(U), |\boldsymbol{l}|_\infty \leq 2, f|_{\partial U} = 0\},$$

which leads to a rapid decay of the coefficients

$$|\alpha_{\boldsymbol{l},\boldsymbol{i}}| = \mathcal{O}(2^{2|\boldsymbol{l}|_1}).$$

Leaving out the subspaces with little contribution to the overall interpolant leads to the sparse grid space

$$V_{0,n}^S = \bigoplus_{|\boldsymbol{l}|_1 \leq n+d-1} W_{\boldsymbol{l}}$$

and interpolant

$$f_{0,n}^S(\boldsymbol{x}) \approx u(\boldsymbol{x}) = \sum_{|l|_1 \leq n+d-1} \sum_{\boldsymbol{i} \in I_{\boldsymbol{k}}} \alpha_{\boldsymbol{k},\boldsymbol{i}} \Phi_{\boldsymbol{k},\boldsymbol{i}}(\boldsymbol{x}).$$

Thus, the number of grid points are $\mathcal{O}(h_n^{-1}(\log(h_n^{-1}))^{d-1}) = \mathcal{O}(2^n n^{d-1})$ and the accuracy

$$\|f_{0,n}^S(\boldsymbol{x}) - u(\boldsymbol{x})\|_{L_2} \in \mathcal{O}(h_n^2(\log(h_n^{-1}))^{d-1}).$$

Sparse grids were introduced by [Smolyak, 1963] for quadrature and interpolation of high dimensional functions using the tensor product approach. The method is also used for solving partial differential equations and has application in various areas such as data mining, computational finance, data compression and much more.

4.4.2 DACE

Kriging was first introduced by and named after [Krige, 1951] and later formalized by [Matheron, 1962]. After its initial use in geostatistics, [Sacks et al., 1989] used the technique for creating a metamodel for (deterministic) computer experiments, which is known as the *DACE (Design and Analysis of Computer Experiments)* model and it is now a common tool in statistics and widely used for the modeling of computer experiments.

The main idea is to interpret the deterministic response Y of a simulation as a stochastic variable, i.e.

$$\hat{Y}(x) = \sum_{j=1}^{k} \beta_j f_j(x) + Z(x),$$

where the first term is a regression model and $Z(x)$ a random variable with mean zero and covariance

$$V(w, x) = \sigma^2 \mathcal{R}(w, x),$$

between $Z(w)$ and $Z(w)$, i.e. the product of the process variance σ and a given correlation model $\mathcal{R}(w,x)$. The functions $f_j : \mathbb{R}^d \to \mathbb{R}, \quad j = 1, \ldots, k$ are chosen regression functions with regression parameters β_j.

In order to find the Kriging predictor, the design matrix $F, F_{ij} = f_j(s_i)$, the matrix of stochastic process correlation R (given at the sample points) and the vector of correlations at untried point $x \in \mathcal{D}$ are defined by

$$\begin{aligned} f(x) &= (f_1(x), \ldots, f_k(x))^T, \\ F &= (f^T(s_1), \ldots, f^T(s_d)))^T && \in \mathbb{R}^{d\times k}, \\ R &= \{R(s_i, s_j)\} && \in \mathbb{R}^{d\times d}, \\ r(x) &= (R(s_1, x), \ldots, R(s_k, x))^T && . \end{aligned}$$

Next, we want to derive the *best linear unbiased predictor* (BLUP) $\hat{Y}(x)$ of y. To this end, consider the *linear* predictor

$$\hat{Y}(x) = c(x)^T Y, \tag{4.4.8}$$

with coefficient $c = c(x) \in \mathbb{R}^k$. Assuming the predictor is *unbiased*, i.e.

$$\begin{aligned} &\mathbb{E}[Y(x) - \hat{Y}(X)] = 0 \\ \iff\ & F^T c(x) = f(x), \end{aligned} \tag{4.4.9}$$

then the error of the model is

$$\begin{aligned} \hat{Y}(x) - y(x) &\overset{(4.4.8)}{=} c^T Y - y(x) \\ &\overset{(4.4.2)}{=} c^T(F\beta + Z) - (f(x)^T\beta + z) \\ &= c^T Z - z + (F^T c - f(x))^T \beta \\ &\overset{(4.4.9)}{=} c^T Z - z, \end{aligned}$$

where Z is the error at the sample points. The mean squared error is therefore given by

$$\begin{aligned} \phi(x) &= \mathbb{E}[(\hat{Y}(x) - y(x))^2] \\ &= \mathbb{E}[(c^T Z - z)^2] \\ &= \mathbb{E}[y^2 - 2c^T Z z + c^T Z Z^T c] \\ &= \sigma^2(1 + c^T R c - 2c^T r). \end{aligned} \tag{4.4.10a}$$

In order to get the *best* predictor with respect to the mean squared error, $\phi(x)$ has to be minimized with the unbiased constraint (4.4.9). To this end, consider the Lagrangian

$$L(c, \lambda) = \sigma^2(1 + c^T R c - 2c^T r) - \lambda^T(F^T c - f).$$

Calculating the derivative

$$\frac{\partial}{\partial c} L(c, \lambda) = 2\sigma^2 (Rc - r) - \lambda^T F^T$$

yields the system of linear equations

$$\begin{pmatrix} R & F \\ F^T & 0 \end{pmatrix} \begin{pmatrix} c \\ \tilde{\lambda} \end{pmatrix} = \begin{pmatrix} r \\ f \end{pmatrix} \tag{4.4.11}$$

with $\tilde{\lambda} = -\lambda/(2\sigma^2)$.

The solution to (4.4.11) is given by

$$\tilde{\lambda} = (F^T R^{-1} F)^{-1} (F R^{-1} r - f) \tag{4.4.12}$$

$$c = R^{-1}(r - F\tilde{\lambda}), \tag{4.4.13}$$

which defines the linear predictor (4.4.8) through

$$\begin{aligned} \hat{Y} &= c^T Y \\ &= (R^{-1}(r - F\tilde{\lambda}))^T Y \\ &\overset{\text{R symm.}}{=} (r - F\tilde{\lambda})^T R^{-1} Y \\ &= r^T R^{-1} Y - (F^T R^{-1} r - f)^T \underbrace{(F^T R^{-1} F) F^T R^{-1} Y}_{\beta^*} \\ &= r^T R^{-1} Y - (F^T R^{-1} r - f)^T \beta^* \\ &= f^T \beta^* + r^T \underbrace{R^{-1}(U - F\beta^*)}_{\gamma^*} \\ &= f(x)^T \beta^* + r(x)^T \gamma^*, \end{aligned} \tag{4.4.14}$$

where $\beta^* = (F^T R^{-1} F) F^T R^{-1} Y$ is the generalized least squares estimate of β (i.e. a solution of the regression problem $F\beta \simeq Y$) and $\gamma^* = R^{-1}(U - F\beta^*)$ are fixed for a given set of sample points. The latter is computed by the residuals $R\gamma^* = Y - F\beta^*$.

Hence, once the parameters β^* and γ^* are determined for a given set $S \in \mathcal{D}$, the predictor can be easily calculated by two simple vector multiplications and one addition. Therefore a computationally cheap response surface model has been constructed.

Note, that also the gradient of the predictor (4.4.14) can be calculated via

$$\hat{Y}' = J_f(x)^T \beta^* + J_r(x)^T \gamma^*,$$

where

$$(J_f(x))_{ij} = \frac{\partial f_i}{\partial x_j}(x), \quad (J_r(x))_{ij} = \frac{\partial \mathcal{R}}{\partial x_j}(\theta, s_i, x)$$

are the Jacobian of f and r, respectively.

The error (4.4.10a) of the predictor can also be calculated explicitly by

$$\begin{aligned}
\phi(x) &= \sigma^2(1 + c^T Rc - 2c^T r) \\
&= \sigma^2(1 + c^T(Rc - 2r)) \\
&\overset{(4.4.13)}{=} \sigma^2(1 + (R^{-1}(r - F\tilde{\lambda}))^T(RR^{-1}(r - F\tilde{\lambda}) - 2r)) \\
&= \sigma^2(1 + (r - F\tilde{\lambda})^T R^{-1}(-F\tilde{\lambda} - r)) \\
&= \sigma^2(1 + (F\tilde{\lambda} - r)^T R^{-1}(F\tilde{\lambda} + r)) \\
&= \sigma^2(1 + \tilde{\lambda}^T F^T R^{-1} F\tilde{\lambda} - r^T R^{-1} r) \\
&\overset{(4.4.12)}{=} \sigma^2(1 + u^T(F^T R^{-1} F)^{-1} u - r^T R^{-1} r),
\end{aligned}$$

where $u = F^T R^{-1} r - f$ and σ^2 can be determined by

$$\sigma^2 = \frac{1}{d}(Y - F\beta^*)^T R^{-1}(Y - F\beta^*),$$

and represents the maximum likelihood estimate of the variance corresponding to the normal equations $(F^T F)\beta^* = F^T Y$ [Lophaven et al., 2002].

Finally, an appropriate regression model and correlation function have to be chosen to define the response surface. Common choices for the regression functions f are polynomials of lower order, depending on the desired accuracy of the model. An obvious choice for the sparse grid approximation are the hierarchical basis functions described in the previous section. However, to make a fair comparison between the different chosen experimental designs, the same basis functions are selected for all of them.

The correlation function $\mathcal{R}$ should be chosen according to the modeled problem, i.e. the objective function. The chosen function highly affects the accuracy of the predictor, c.f. section 6.3. Typical correlation functions are stationary, i.e. $\mathcal{R}(w, x) = \mathcal{R}(w-x)$, assuming that any non–stationary behavior can be modeled by the regression function [Sacks et al., 1989]. Furthermore, only product–type correlations of the form $\mathcal{R}(w, x) = \prod_{i=1}^{d} \mathcal{R}_i(w_i-x_i)$ are considered. For example, the exponential correlation model

$$\mathcal{R}(w, x) = \prod_{i=1}^{d} \exp(-\theta_i |w_i - x_i|^p)$$

with parameters $\theta_i > 0$ and $0 < p \leq 2$ or the linear model

$$\mathcal{R}(w, x) = \prod_{i=1}^{d} (1 - \theta_i |w_I - x_i|),$$

which yields a linear spline response $\hat{Y}$.

4.5 Implementation

The four set of sample points are produced by a set of Python scripts. Each one is saved in a separate text file. Since Sparse Grid points are the only ones depending on a level, the other experiment designs comply with the number of points of each sparse grid level. The generation of Monte Carlo sample points is straight forward with the Python build–in function `random.uniform(a,b)`, which selects an element from a uniform distribution on the interval $[a, b]$. An additional point set can be easily added the same way for each new "level".

The Python library *SOBOL* by Corrado Chisari[1] is employed for the generation of Quasi Monte Carlo points. Additional points are simply the next points of the Sobol sequence.

The augmentation of the Latin Hypercube design is slighty more complicated. For the first set of N points, the parameter space is divided into N^d bins creating a N^d matrix and points are chosen sequentially. For each selected point, the corresponding columns and rows of the bin matrix are deleted, which results in a Latin Hypercube Design. For the addition of Mpoints, the design space is divided into $(M + N)^d$ bins and all rows and columns corresponding to the already selected points are deleted before the additional points are selected in the same manner.

Finally, the Sparse Grid points are produced as described in equations (4.4.3) and (4.4.4).

The files containing the sample points are then parsed into the input files for the Navier–Stokes solver containing a parametrized version of the OWC, which in turn are utilized to start the simulation. The solver produces a number of output files containing the calculated velocity field among other flow information. These files are then converted into the VTK[2] format, which can be processed by yet another Python script calculating the the average airflow in the opening for the turbine.

The final response as well as the metamodel are calculated with MATLAB. The sample points and corresponding average air flow values are read from input files, then the additional constraints in equations (3.4.1) and (3.4.2) are applied. The response surface is determined with the MATLAB Toolbox DACE [Lophaven et al., 2002].

Since the Gauss correlation model yields a smooth response surface and the gradient can be calculated via (4.4.2), two different gradient–based optimization algorithms are applied to the metamodel. The two methods are a Trust–Region method[3] implemented in the MATLAB function `fminunc` for unconstrained optimization and a Trust–Region–Reflective algorithm[4] via the MATLAB function `fmincon`. The latter is used for constrained opti-

[1] available online at `http://people.sc.fsu.edu/~jburkardt/py_src/sobol/sobol.html`

[2] Visulization Toolkit, see `http://www.vtk.org/`

[3] See `http://www.mathworks.de/de/help/optim/ug/unconstrained-nonlinear-optimization-algorithms.html#brnpcy5`

[4] See `http://www.mathworks.de/de/help/optim/ug/constrained-nonlinear-optimization-algorithms.html#brnpeek`

mization, where the only constraints given to the function are the bounds of the parameter space.

Apart from these two methods, two different direct search algorithms are employed as well. The first one is the DIRECT search method described in section 4.3.1. A MATLAB implementation of the method by [Finkel, 2003] is used for this purpose. The method is bounded in nature, but unconstrained. Another unconstrained, but unbounded search algorithm is the Nelder–Mead–Simplex method[5] implemented in the MATLAB routine `fminsearch`.

The modularization and automatization via several scripts makes changes in the work flow very simple. For example, the information extracted from the Navier–Stokes solver can be altered easily, since all the information about the flow is saved in the output files.

[5] See `http://www.mathworks.de/de/help/optim/ug/unconstrained-nonlinear-optimization-algorithms.html#brnoxyk`

5 Navier Stokes solver

The blackbox consists of a two–phase fluid simulation with a wave generator and a wave energy converter corresponding to the input parameters. The following describes the flow solver in detail, i.e. the governing equations, discretization and employed numerical methods. Furthermore, the numerical wave generation outlined.

5.1 Modeling equations

The governing equations for the flow are the incompressible *Navier–Stokes equations*, which are to be solved in each phase. In this section, the equations and the corresponding two–phase solver including *fluid–structure interaction* with a rigid body are described.

Since we consider water and air, we have to look at a two phase flow. The two fluids do not mix, i.e. the domain $\Omega \in \mathbb{R}^3$ can be decomposed as $\Omega = \Omega_1 \cup \Omega_2 \cup \Gamma_f$, where Ω_i, $i = 1, 2$ contains the respective fluid and $\Gamma_f = \partial\Omega_1 \cap \partial\Omega_2$ denotes the free boundary separating those two. Both Ω_i and Γ_f are time–dependent. Each fluid can be described by the single–phase Navier–Stokes equation in its respective domain Ω_i.
Additionally, the fluid–structure interaction is modeled by the kinematic movement equations, which have to be integrated into the two–phase framework. This is done via the material properties of the rigid body — density and viscosity — as well as the velocity and an additional level–set function. The part of the domain occupied by the rigid body is denoted by $\mathcal{R}$.

Taking all of the above into consideration, we obtain the system of equations

$$\begin{aligned}
\rho(\phi)\frac{D\boldsymbol{u}}{Dt} + \nabla p &= \nabla \cdot (\mu(\phi)\boldsymbol{D}) \\
&\quad - \sigma\kappa\delta(\phi)\boldsymbol{n} + \rho(\phi)\boldsymbol{g} + \boldsymbol{f_s} && \text{in } \Omega \times [0,T], && (5.1.1a)\\
\nabla \cdot \boldsymbol{u}_i &= 0 && \text{in } \Omega_i \times [0,T], && (5.1.1b)\\
\boldsymbol{u}_i &= 0 && \text{on } \Gamma \times [0,T], && (5.1.1c)\\
\boldsymbol{u}_i &= \boldsymbol{u}_{0_i} && \text{on } \Omega_i \times \{0\}, && (5.1.1d)\\
\rho_s\frac{\partial \boldsymbol{u}_s}{\partial t} &= -\nabla p_s + \rho_s\boldsymbol{g} && \text{in } \mathcal{R} \times [0,T], && (5.1.1e)\\
\nabla \cdot \boldsymbol{u}_s &= 0 && \text{in } \mathcal{R} \times [0,T], && (5.1.1f)
\end{aligned}$$

where $\boldsymbol{u}_i : \Omega \times [0,T] \to \mathbb{R}^3$ is the velocity field — for $i = 1, 2$ the respective fluid velocity, for $i = s$ the rigid body velocity and simply u the total velocity. Also, $p_i : \Omega \times [0,T] \to \mathbb{R}$ denotes the pressure, $\rho_i \in \mathbb{R}$ the density, $\nu_i \in \mathbb{R}$ the viscosity — with the indices as for the velocity field, where applicable. The surface tension coefficient is given by σ, which is treated as a material dependent constant coefficient, and κ describes the local curvature of the fluid interface. The use of the *level–set functions* $\phi, \phi_s : \Omega \times [0,T] \to \mathbb{R}$ is described below. Also, there are the volume forces gravity $\boldsymbol{g} : \Omega \times [0,T] \to \mathbb{R}$ and $\boldsymbol{f_s} : \Omega \times [0,T] \to \mathbb{R}$, which is an additional term formally modeling the fluid–structure interaction. The deformation tensor $\boldsymbol{D}$ is defined as $\boldsymbol{D} = \nabla \boldsymbol{u} + \{\nabla \boldsymbol{u}\}^T$ and$\frac{D\boldsymbol{u}}{Dt} = \frac{d\boldsymbol{u}}{dt} + \boldsymbol{u} \cdot \nabla \boldsymbol{u}$ denotes the material derivative, c.f. [Croce et al., 2009]. Furthermore, there are two additional constraints on the free boundary, the so called *kinematic condition* $\boldsymbol{u}_1 = \boldsymbol{u}_2$ and the *dynamic condition* $(\boldsymbol{T}_1 - \boldsymbol{T}_2) \cdot \boldsymbol{n} = \sigma\kappa\boldsymbol{n}$ on Γ_f. Here $\boldsymbol{T}_i := -p_i\boldsymbol{I} + \mu_i\boldsymbol{D}_i$ denotes the stress tensor, σ the surface tension coefficient, which is treated as a material dependent constant coefficient, and local curvature κ. Using the *level–set function* $\phi : \Omega \times [0,T] \to \mathbb{R}$, we get the two–phase system

$$\begin{aligned}\rho(\phi)\frac{D\boldsymbol{u}}{Dt} + \nabla p = \nabla \cdot (\mu(\phi)\boldsymbol{D})\\ - \sigma\kappa\delta(\phi)\boldsymbol{n} + \rho(\phi)\boldsymbol{g}\text{in } \Omega,\end{aligned} \tag{5.1.2}$$

where $\frac{D\boldsymbol{u}}{Dt} = \frac{d\boldsymbol{u}}{dt} + \boldsymbol{u} \cdot \nabla \boldsymbol{u}$ denotes the material derivative. Here, density and viscosity are globally defined by

$$\begin{aligned}\rho(\phi) &= \rho_1 + (\rho_2 - \rho_1)H(\phi),\\ \mu(\phi) &= \mu_1 + (\mu_2 - \mu_1)H(\phi),\end{aligned}$$

where H describes the Heaviside function:

$$H(\phi) := \begin{cases} 0 & \text{if } \phi < 0 \\ \frac{1}{2} & \text{if } \phi = 0\ . \\ 1 & \text{if } \phi > 0 \end{cases} \tag{5.1.4}$$

We consider an open rectangular domain $\Omega \in \mathbb{R}^3$ with slip boundary condition on each wall of the Lipschitz boundary $\Gamma = \partial\Omega$, i.e.

$$(\boldsymbol{u} \cdot \boldsymbol{n})|_\Gamma = 0, \quad \partial_n(\boldsymbol{u} \cdot \boldsymbol{t})|_\Gamma = 0, \quad \partial_n(\boldsymbol{u} \cdot \boldsymbol{s})|_\Gamma = 0,$$

where $\boldsymbol{n}$ is the outer unit normal vector and $\boldsymbol{t}$ and $\boldsymbol{s}$ are linearly independent tangential vectors.

5.2 The Level Set Method

As mentioned above, the free surfaces are modeled employing the so–called *level set function* introduced by Osher and Sethian [Osher and Sethian, 1988]. Following this surface capturing method, the surface is approximated by a smooth scalar function ϕ and the surface is implicitly given by the zero level set of this function, i.e.

$$\Gamma_f(t) := \{\boldsymbol{x} \in \Omega | \phi(\boldsymbol{x}, t) = 0\} \tag{5.2.1}$$

and therefore all $\boldsymbol{x}(t)$ vanish on $\Gamma_f(t)$, i.e.

$$\phi(\boldsymbol{x}(t), t) = 0. \tag{5.2.2}$$

Derivation then yields the transport equation

$$\phi_t + \nabla\phi(\boldsymbol{x}(t), t) \cdot \boldsymbol{x}'(t) = 0. \tag{5.2.3}$$

Here, $\boldsymbol{x}'(t)$ is the velocity field $\boldsymbol{u}$ on the free surface Γ_f and thus, the initial value problem for the transport of the level set function is given by

$$\phi_t + \boldsymbol{u} \cdot \nabla\phi = 0, \tag{5.2.4a}$$

$$\phi(\boldsymbol{x}, 0) = \phi_0. \tag{5.2.4b}$$

Another property of a level set function is that it should be a signed distance function. Hence, we have the additional constraints

$$\phi(\boldsymbol{x}, t) = \begin{cases} > 0 & \text{if } \boldsymbol{x} \in \Omega_1 \\ = 0 & \text{if } \boldsymbol{x} \in \Gamma_f \\ < 0 & \text{if } \boldsymbol{x} \in \Omega_2 \end{cases} \tag{5.2.5a}$$

$$\text{and } |\nabla\phi| = 1. \tag{5.2.5b}$$

This property ensures that we have a Lipschitz–continuous function in the whole domain, which in return ensures numerical stability and accuracy. However, once the level set function is transported, the Eikonal equation (5.2.5b) might not hold any longer. This essential property should at least hold in an ϵ–neighborhood of the free boundary. We ensure this by a *reinitialization* step following the ideas of Sussman et al. [Sussman et al., 1994]. In this step, the zero level set stays unchanged, while the property of a distance function is regained. To do so, the initial value problem

$$d_\tau + \operatorname{sign}(\phi)(|\nabla d| - 1) = 0, \tag{5.2.6a}$$

$$d(\boldsymbol{x}, 0) = \phi(\boldsymbol{x}), \tag{5.2.6b}$$

with artificial time τ and sign function

$$\operatorname{sign}(x) = \begin{cases} 1 & \text{if } x > 0 \\ 0 & \text{if } x = 0 \\ -1 & \text{if } x < 0 \end{cases}$$

has to be solved. Since this is only done in an ϵ–neighborhood of the free surface, the number of necessary iterations of the reinitialization can be dramatically reduced by stopping as soon as $|\nabla d| = 1$ holds for all $|d| \le \epsilon$.

5.3 Discretization

The hyper–rectangular domain Ω is discretized by an equidistant grid in all three dimensions. For stability reasons a *staggered grid* is employed, i.e. the variables are stored at different node points, as illustrated in figure 5.3.1. The scalar variables, in our case pressure and level set function values, are stored in the cell center, whereas the vector components u, v and w of the velocity field are stored at the center points of the cell walls. In order to be able to store all values, an additional boundary layer has to be introduced, the so–called *ghost cells*. They are not part of the domain Ω and contain the values necessary to implement the boundary conditions.

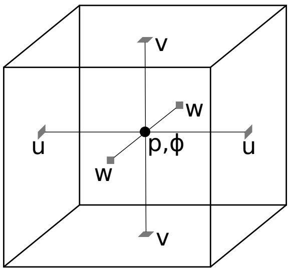


Fig. 5.3.1: Cell in a staggered grid [Peuker, 2011].

The time steps δt are adaptively controlled by a *CFL–condition* for the different forces.

The discrete level set function ϕ_h satisfies all the properties mentioned above. However, since density and viscosity are not continuous at the free surface, they have to be smoothed along Γ_f. To this end, we introduce the boundary layer thickness $\epsilon(\boldsymbol{x})$, which behaves like the mesh size and preserves the global convergence rate. Therefore, we define the global density and viscosity, respectively, by

$$\rho^\epsilon(\phi_h) = \rho_2 + (\rho_1 - \rho_2) H^\epsilon(\phi_h), \tag{5.3.1a}$$

$$\mu^\epsilon(\phi_h) = \mu_2 + (\mu_1 - \mu_2) H^\epsilon(\phi_h), \tag{5.3.1b}$$

where $H^\epsilon(\phi_h)$ is the smoothed Heaviside function. It is defined by

$$H^\epsilon(\phi_h(\boldsymbol{x})) := \begin{cases} 0 & \text{if } \phi_h(\boldsymbol{x}) < -\epsilon(\boldsymbol{x}) \\ \frac{1}{2}(1 + \frac{\phi_h(\boldsymbol{x})}{\epsilon(\boldsymbol{x})} + \frac{1}{\pi}\sin(\frac{\pi\phi_h(\boldsymbol{x})}{\epsilon(\boldsymbol{x})})) & \text{if } |\phi_h(\boldsymbol{x})| \le \epsilon(\boldsymbol{x}) \\ 1 & \text{if } \phi_h(\boldsymbol{x}) > \epsilon(\boldsymbol{x}) \end{cases} .$$

Since now we have a smooth function H^ϵ, it is differentiable and the corresponding delta functional is the derivative of this smooth Heaviside function, i.e.

$$\begin{aligned}\delta^\epsilon(\phi_h(\boldsymbol{x})) &= \partial_{\phi_h} H^\epsilon(\phi_h(\boldsymbol{x}))\\ &= \begin{cases} \frac{1}{2\epsilon(\boldsymbol{x})}(1+\cos(\frac{\pi\phi_h(\boldsymbol{x})}{\epsilon(\boldsymbol{x})})) & \text{if } |\phi_h(\boldsymbol{x})| \le \epsilon(\boldsymbol{x})\\ 0 & \text{else.}\end{cases}\end{aligned}$$

Additionally, the curvature κ and unit normal $\boldsymbol{n}$ are accordingly discretized. This means, we replace $\sigma\kappa\delta(\phi)\boldsymbol{n}$ in (5.1.1a) by $\sigma\kappa_h\delta^\epsilon(\phi_h)\boldsymbol{n}_h$ with discrete curvature

$$\kappa_h := \nabla \cdot \boldsymbol{n}_h$$

and discrete unit normal

$$\boldsymbol{n}_h := \frac{\nabla\phi_h}{|\nabla\phi_h|} = \frac{(\phi_x,\phi_y,\phi_z)_h^T}{\sqrt{(\phi_x^2+\phi_y^2+\phi_z^2)_h}}.$$

In each time step, the level set function is transported by solving

$$\partial_t\phi_h + \boldsymbol{u}_h \cdot \nabla\phi_h = 0 \tag{5.3.2}$$

and afterward density and viscosity are calculated by (5.3.1).

The rigid body level–set function is used in the same manner and the rigid body velocity can be calculated by

$$H^\epsilon(\phi_s)\boldsymbol{u}_s(\boldsymbol{x},t) = H^\epsilon(\phi_s)(\boldsymbol{v}(t) + \boldsymbol{w}(t) \times \boldsymbol{r}) \quad \text{in } \mathcal{R}+\epsilon, \tag{5.3.3}$$

where $\boldsymbol{v}$ is the translation velocity, $\boldsymbol{w}$ the angular velocity, $\boldsymbol{r}$ the vector from an arbitrary point within the rigid body to its center of mass and ϵ describes the radius, within which the structure is describes by its level–set function ϕ_s.

Following the *smoothed profile method (SPM)* [Nakayama and Yamamoto, 2005], the total velocity is represented by a smooth convex combination of the rigid body velocity $\boldsymbol{u}_s$ and the fluid velocity $\boldsymbol{u}_f$, i.e.

$$\begin{aligned}\boldsymbol{u} :&= (1-H^\epsilon(\phi_s))\boldsymbol{u}_f + H^\epsilon(\phi_s)\\ &= \boldsymbol{u}_f + H^\epsilon(\phi_s)(\boldsymbol{u}_s - \boldsymbol{u}_f).\end{aligned} \tag{5.3.4}$$

The discrete transport equation (5.3.2) is solved by a fifth order *WENO*–method for the convective part $\boldsymbol{u}\cdot\nabla\phi_h$. The time discretization is handled by a second order *Adams–Bashforth*–method. The reinitialization step (5.2.6) is discretized by a third order *Runge–Kutta*–scheme for the temporal part and a fifth order *WENO*–scheme for the spacial part.

In order to decouple the momentum (5.1.1a) and the continuity equation (5.1.1b) of the incompressible Navier–Stokes equations, we employ the *Chorin–Teman projection method* as proposed by both Chorin [Chorin, 1968] and Temam [Témam, 1969]. This allows us to calculate the pressure and the velocity field separately. The idea of this method is to decompose the impulse equation (5.1.1a) into a divergence free and a rotation free part. To this end, the divergence free approximated velocity field u^* is projected onto the space of divergence free velocities using the pressure p.

This means, we first compute the auxiliary velocity u^* by

$$\begin{aligned} u^* = u^n &+ \delta t(\boldsymbol{g} - \nabla \cdot (u^n \otimes u^n) \\ &+ \frac{1}{\rho^\epsilon(\phi^n, \phi_s^n)}(\nabla \cdot (\mu^\epsilon(\phi^n, \phi_s^n)\boldsymbol{D}^n) \\ &- (1 - H^\epsilon(\phi_s^n))\sigma\kappa(\phi^n)\delta^\epsilon(\phi^n, \phi_s^n)\nabla\phi^n)), \end{aligned} \tag{5.3.5}$$

which is subsequently transported by the level set function and reinitialized. This yields the new level set functions ϕ^{n+1}, ϕ_s^{n+1} and we get the system of equations

$$\frac{u^{*,n+1} - u^*}{\delta t} + \frac{\nabla p^{*,n+1}}{\rho(\phi^{n+1}, \phi_s^{n+1})} = 0, \tag{5.3.6a}$$

$$\nabla \cdot u^{*,n+1} = 0. \tag{5.3.6b}$$

Next, the divergence operator is applied to (5.3.6a) and under consideration of equation (5.3.6b), we get the *first pressure poisson equation*

$$\nabla \cdot \left(\frac{\nabla p^{*,n+1}}{\rho^\epsilon(\phi^{n+1})\phi_s^{n+1}} \right) = \nabla \cdot \frac{u^*}{\delta t}. \tag{5.3.7}$$

Solving this equation gives the new pressure $p^{*,n+1}$ and $u^{*,n+1}$ is calculated by the first velocity correction step

$$u^{*,n+1} = u^* - \frac{\delta t}{\rho^\epsilon(\phi^{n+1})\phi_s^{n+1}}\nabla p^{*,n+1}. \tag{5.3.8}$$

The preliminary pressure and velocity does not distinguish between fluid and structure, i.e. the rigid body is considered to be a part of the fluid. This assumption is corrected by an additional pressure within the body. The rigid body velocity is given by

$$\begin{aligned} H^\epsilon(\phi_s^{n+1})\boldsymbol{u}_s^{n+1}(\boldsymbol{x}, t) = H^\epsilon(\phi_s^{n+1})(&\boldsymbol{v}^{n+1}(t) \\ &+ \boldsymbol{w}^{n+1}(t) \times \boldsymbol{r}^{n+1}). \end{aligned} \tag{5.3.9}$$

Next, the auxiliary pressure p_s^* for the rigid body can be derived by solving the *second pressure poisson equation*

$$\nabla \cdot \left(\frac{\nabla p_s^{*,n+1}}{\rho^\epsilon(\phi^{n+1}), \phi_s^{n+1}} \right) = \nabla \cdot \frac{H^\epsilon(\phi_s^{n+1})(u_s^{n+1} - u^{*,n+1})}{\delta t}. \tag{5.3.10}$$

For this step, we assume the velocity field to be divergence free, which in turn implicitly guarantees the *impermeability condition*, i.e. the rigid body boundary is impervious. Finally, the rigid body velocity $\boldsymbol{u}_s$ can be integrated into the total velocity field $\boldsymbol{u}$ by the second velocity correction step

$$\begin{aligned} u^{n+1} &= u^{*,n+1} + H^{\epsilon}(\phi_s^{n+1})(u_s^{n+1} - u^{*,n+1}) \\ &- \frac{\delta t}{\rho^{\epsilon}(\phi^{n+1}, \phi_s^{n+1})} \nabla p_s^{*,n+1}. \end{aligned} \tag{5.3.11}$$

This step ensures that the global velocity is divergence free, as well as the stiffness of the rigid body.

Finally, the pressure can be simply calculated by

$$p^{n+1} = p^{*,n+1} + p_s^{*,n+1}. \tag{5.3.12}$$

In order to have well–defined problems (5.3.7) and (5.3.10), we need the additional boundary conditions

$$\partial_n p^{*n+1}|_{\Gamma_f} = \frac{\rho^{\epsilon}(\phi^{n+1}, \phi_s^{n+1})}{\delta t} (\boldsymbol{u}_\Gamma^* - \boldsymbol{u}_\Gamma^{n+1}) \cdot \boldsymbol{n}, \tag{5.3.13a}$$

$$\partial_n p_s^{*n+1}|_{\Gamma_f} = \frac{\rho^{\epsilon}(\phi^{n+1}, \phi_s^{n+1})}{\delta t} (\boldsymbol{u}_\Gamma^* - \boldsymbol{u}_\Gamma^{n+1}) \cdot \boldsymbol{n} \tag{5.3.13b}$$

i.e. equation (5.1.1a) is projected onto the outer unit normal. In case $u_\Gamma^* = u_\Gamma^{n+1}$, we know that $\partial_n p^{n+1}|_{\Gamma_f} = 0$, which means we have a homogeneous Neumann boundary condition.

The Poisson equations (5.3.7) and (5.3.10) are discretized using finite differences, which yields a linear system of equations $Ax = b$, which can be solved by various iterative methods, e.g. SOR or CG.

In summary, the main steps of the algorithm to solve the incompressible two–phase system are to use the Chorin–Temam projection method to calculate the pressure and the auxiliary velocity fields, transport and reinitialize the level set functions and finally compute the actual velocity field by a correction step, see algorithm 1.

The whole algorithm runs parallel using the library *MPI (Message Passing Interface)* [Forum, 1993]. To this end, the domain Ω is decomposed into disjoint subdomains $\Omega_h^1, \dots, \Omega_h^P$ and each of the $1, \dots, P$ processes solves the equations in its respective domain. Additionally, the boundary layers of each subdomain have to be communicated between the different processes at each time step.

Algorithm 4 Parallel two–phase solver

```
Require: mesh sizes $\delta x, \delta y, \delta z \in \mathbb{R}$, end time $T$, level set function $\phi_0$, domain $\Omega$
Assign local subdomains $\Omega^q$, $q \in \{1, \dots, P\}$.
Set $t := 0, n := 0$ and $h := \max(\delta x, \delta y, \delta z)$.
Initialize $u^n := u_0$, $p^n := p_0$ und $\phi^n := \phi_0$ in each subdomain $\Omega^q$, $q \in \{1, \dots, P\}$.
Set boundary layer thickness $\epsilon(\boldsymbol{x})$.
Initialize rigid body center of mass $\boldsymbol{x}_s^0$.
Set boundary values of $u^n$ on local boundaries $\partial\Omega^q \cap \partial\Omega$.
Time loop:
while $t \leq T$ do
    Compute time step size $\delta t$.
    Impulse equation:
     Comm. boundary values of $u^n$ and $\phi^n$.
     Compute aux. velocity $u^*(u^n, \phi^n, \phi_s i^n)$ as in (5.3.5) in $\Omega^q$.
     Set boundary values for $u^*$ on $\partial\Omega^q \cap \partial\Omega$.
    Set RHS of Poisson equation:
     Comm. boundary values of $u^*$.
     Set RHS $\boldsymbol{b} := \nabla \cdot u^*$ of euqation (5.3.7) in $\Omega^q$.
    Transport equation:
     Transport level set functions as in (5.3.2).
    Reinitialization:
     Reinitialize following (5.2.6).
    Pressure Poisson equation I:
     Compute $p^{*,n+1}$.
     Compute boundary values of $p^*_{it}$ for each iteration step $it$.
    Velocity correction step I:
     Compute $u^{*,n+1}$ as in (5.3.8) in $\Omega^q$.
     Set boundary values for $u^{*,n+1}$ on $\partial\Omega^q \cap \partial\Omega$.
    Set RHS of Poisson equation II:
     Comm. boundary values of $\boldsymbol{u}_s^*$.
     Set RHS of (5.3.10) in $\Omega^q$.
    Pressure Poisson equation II:
     Compute $p^{*,n+1}$.
     Comm. boundary values of $p_{it}$ for each iteration step $it$.
    Velocity correction step II:
     Compute $u^{n+1}$ as in (5.3.11) in $\Omega^q$.
     Set boundary values for $u^{n+1}$ on $\partial\Omega^q \cap \partial\Omega$.
    Set $t := t + \delta t$ and $n := n + 1$.
end while
```

5.4 Numerical wave generation

An overview of different approaches to numerical simulation of waves can be found in [Kim et al., 1999] and [Tanizawa, 2000]. There are two main types of wave generation in a numerical wave tank.

- The introduction of a *source term* on the boundary or inside of the simulation domain and
- the introduction of a moving wave maker – e.g. a paddle or flap – to reproduce waves as in an *experimental wave tank (EWT)* in a lab.

Either way, the computational domain of the wave simulation is called *numerical wave tank (NWT).*

The simulations computed for this Thesis use the second approach. Examples of similar NWT simulations in two spatial dimensions can be found in [Mikkola, 2007] or [Park et al., 2001]. However, we are going to employ a three dimensional wave tank, as we will study three dimensional WEC devices.

Examples for the first type of NWT, where a source term induces the waves, include [Lin and Liu, 1999] (2D) and [Ha et al., 2011] (extension to 3D), where a source term s *within* a subdomain Ω_s of the computational domain is introduced for the continuity equation of the Navier–Stokes equations, i.e.

$$\nabla \boldsymbol{u} = s(\boldsymbol{x}, t) \text{ in } \Omega_s.$$

Depending on the source term s, different kind of waves known from wave theory can be reproduced. The corresponding source term is derived from the water surface elevation $\eta(x,t)$, c.f. equations 2.3.1 and 2.3.9. This produces an increase in mass and thus results in a movement of the water surface. Depending on the source term, which is derived from wave theory, c.f. section 2.3, different kinds of waves can be produced.

A similar approach is used in the package `REEF3D` [Bihs, 2011], where the wave motion is prescribed by a velocity field $\boldsymbol{u}_{analytical}$, which is derived from wave theory, i.e. $\boldsymbol{u}_{analytical} = \nabla\Phi$. This analytical velocity field, which is similar to the source term s introduced in [Lin and Liu, 1999,Ha et al., 2011], is then combined with the velocity field $\boldsymbol{u}_{numerical}$ from the computational solution of the Navier–Stokes equations via a space–dependent relaxation function $\Psi(x)$ – therefore called *relaxation method* as introduced by [Larsen and Dancy, 1983].

$$\boldsymbol{u}_{relaxed} = \Psi(x)\boldsymbol{u}_{analytical} + (1 - \Psi(x))\boldsymbol{u}_{numerical}. \tag{5.4.1}$$

This ensures that the waves are a produced in the first part of the computational domain (*wave generation zone*), combined with the computational solution in the next zone (*reflection protection zone*) and the *numerical wave tank* containing the actual produced waves in another region of the domain. Finally, as in most NWT, the waves are absorbed at the so–called *numerical beach*, so that the waves are not reflected off the back wall of the wave tank [Kamath, 2012].

The so–called *LONGTANK* by [Wang et al., 1995] employs modified boundary conditions to generate arbitrary waves. Other examples of this *purely numerical* wave generation – i.e. it cannot be reproduced in an EWT – include [Chen et al., 1999, Watanabe et al., 2005, Choi and Yoon, 2009, Senturk, 2011].

Many wave simulations only consider a *one phase* model, i.e. just the water phase is simulated and the air is disregarded. As we are interested in the air flow induced by the construction of the wave energy converter, a *two phase* model is essential.

Most of the studies involving a source term for wave generation, validate the accuracy of the produced waves by the wave theory model they used to derive the source term – e.g. linear waves, Stokes waves or Cnoidal waves. Many show very good results for a variety of these waves. However, changing the original conditions (either boundary condition or governing equations) might lead to unphysical effects, such as an increase of mass in a region of the computational domain and thus might not lead to physically correct waves. Therefore, we choose the wave generation similar to an EWT.

We employ a wave tank as described in [Peuker, 2011], where different types of wave makers are used to generate consistent waves of different shapes, height and period. Since the piston wave maker, as depicted in figures 2.3.2 and 5.4.1, creates the heighest waves and is easy to control, it is also utilized here. The velocity of the wave maker is prescribed by the time–dependent function

$$A \sin(\pi t/B)$$

with parameters A and B to be determined, c.f. section 6.1. The numerical beach consists of a simple sloped bottom to absord some of the wave energy. In the simulations with the wave energy converters, the devices replace the slope.

Another approach to produced a single wave instead of a number of consecutive ones, is to use a falling water column. This means, that in the initial state of the fluid simulation, the water level is higher at the one end of the wave tank, c.f. figure 6.3.1, which is pulled down by gravity and induces a wave during the simulation. This approach is used in a two–dimensional test case, see section 6.3.

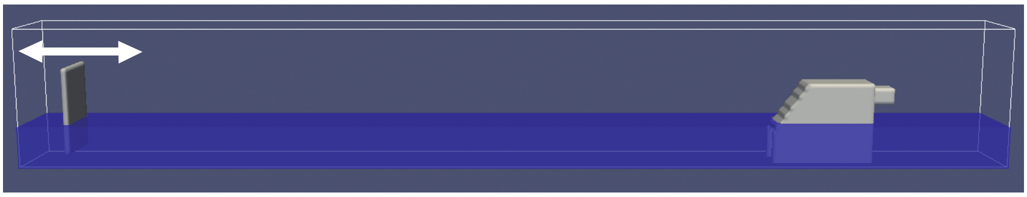


Fig. 5.4.1: Setup of the computational domain including the wavemaker (left) and the OWC Geometry (right) [Peuker, 2011].

6 Numerical Results

The following summarizes the numerical results of the simulations performed to optimize the shape of the wave energy converter. The design is parametrized as described in section 3.2 and two cases with subsets of the complete eleven–dimensional parameter space are considered.

6.1 Optimal wave

First of all, the average wave climate (c.f. section 3.2) has to be simulated in a numerical wave tank as described in the previous chapter. To this end, the wave maker velocity is prescribed by a scaled sine function of the form

$$A \sin(\pi t/B)$$

with parameters A and B to be determined and variable time t. The parameter A controls the height of the sine function and therefore the peak velocity of the wave maker, while B manipulates the period of the wave maker and thus also the period of the induced wave. More precisely, the period of the wave maker is equal to $2B$ and a similar period can be expected for the wave. The wave tank measures $350 \times 50 \times 50$ with a numerical beach consisting of a slope at the end. In order to find a stable wave climate, each simulation lasts 70 seconds of simulation time.

Since a good approximation of the ideal average wave climate was found by varying both parameter on a regular grid, no further optimization was necessary. The values $A = 3.0, B = 5.5$ are found to be optimal and produce on average a wave height of 2.495 meters with a period of 7.3825 seconds. Figure 6.1.1 shows the wave height measured at the center of the wave tank during the simulation.

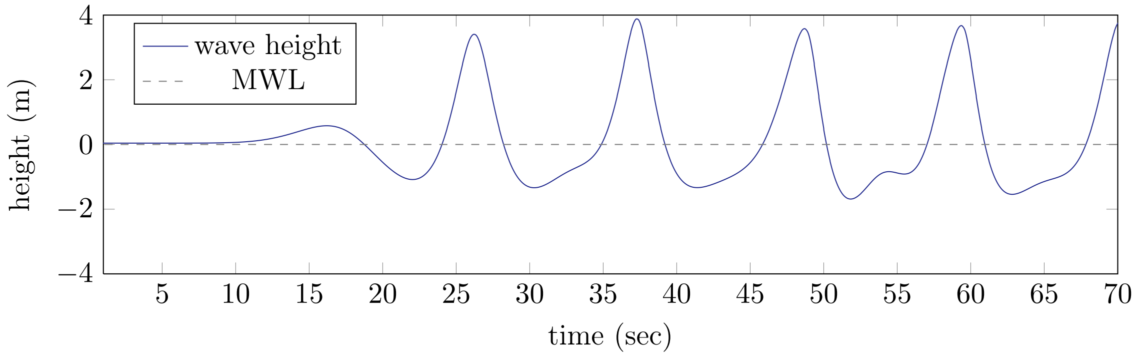


Fig. 6.1.1: Wave height over time, measured at the center of the numerical wave tank.

6.2 Difficulties

As discussed in the previous chapters, the evaluation of the blackbox function is computationally expensive. Several test runs are performed to determine the cost of a single simulation run and estimate the total runtime for the evaluation of a complete set of sample points.

The ideal setup would be a *longtime* simulation (70 seconds of simulation time for about ten wave periods) of an OWC device parametrized by *all eleven parameters* in a *full three–dimensional* wave tank with a high resolution. However, to fullfill (almost) all of these criteria, a single simulation run lasts about three whole days on 64 cores (on an Intel Xeon EM 64 T 3.2 GHz cluster with Myrinet XP interconnect on Ubuntu 10.04 LTS) — with a low resolution of one meter per grid cell. This runtime makes it infeasible to perform a sufficient number of simulations to search the whole eleven–dimensional parameter space — one point evaluation per parameter direction would already take more than a month (assuming only 64 cores were available).

For these reasons, some compromises concerning the simulation time, number of parameters and size of the wave tank have to be made. Since the resolution is already as low as possible (mesh size of one meter), the next step is to shrink the size of the wave tank to reduce the number of degrees of freedom. As the wave maker needs a sufficiently large distance from the OWC to reduce the influence of reflected waves from the wave energy device on the wave maker, the length cannot be shortened. Also, the height of the wave tank is bounded by the height of the wave energy converter. However, the width can be changed if the harbor walls are neglected — which conveniently also reduces the number of parameters to only seven. To further reduce the number of parameters, the width of the OWC is fixed as well. This also ensures a better comparability of the different designs, since the walls of the wave tank can act as harbor walls if they are close to the device. Finally, the simulation time is reduced to only 35 seconds, which gives it just enough time for the first full oscillation inside the turbine shaft to be be observed.

The results of this six–dimensional optimization problem can be found in section 6.4.

In order to get an idea of how the whole optimization works and to get a visual output of the complete metamodel, an even further reduced test case is considered in section 6.3. Only two parameters are considered and the wave maker, which adds a large amount of the computational complexity of the CFD solver, is replaced by a falling water column. Thus, the simulation time is considerably reduced to olny ten seconds.

6.3 Case 1: 2D

The two–dimensional test case is a small version of the actual optimization problem. The wave maker is neglected and instead only one solitary wave is induced by a falling water column. The start configuration is shown in figure 6.3.1. The simulation time of ten seconds grantees that the wave can enter the wave energy plant and induce the water column to rise and fall once. Since only one wave arrives, the device does not develop its full oscillation potential, but at least a first approximation of the oscillation mode can be determined. Each simulation run only takes about 10 to 30 minutes on 32 cores.

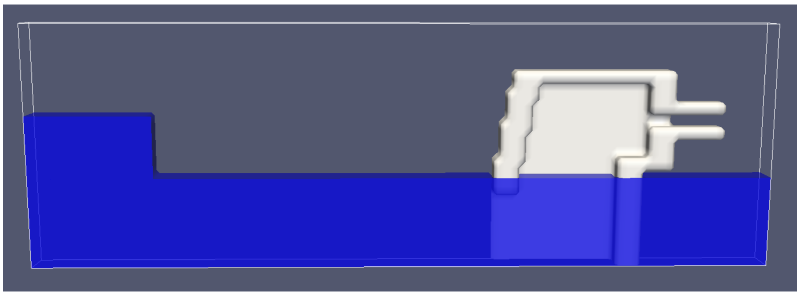


Fig. 6.3.1: Initial setup of falling water column.

The two parameters to be examined are the height a and lip angle angle α, which are confined to the interval $a, \alpha \in [13, 15] \times [0, 40]$. The other measurements are fixed at $b = 3.0, c = 15.0, \beta = 0.0°, f = 6.0, g = 5.0$. The wave tank measures $120 \times 40 \times 6$, with one meter per grid cell. This makes it a quasi–two–dimensional CFD simulation.

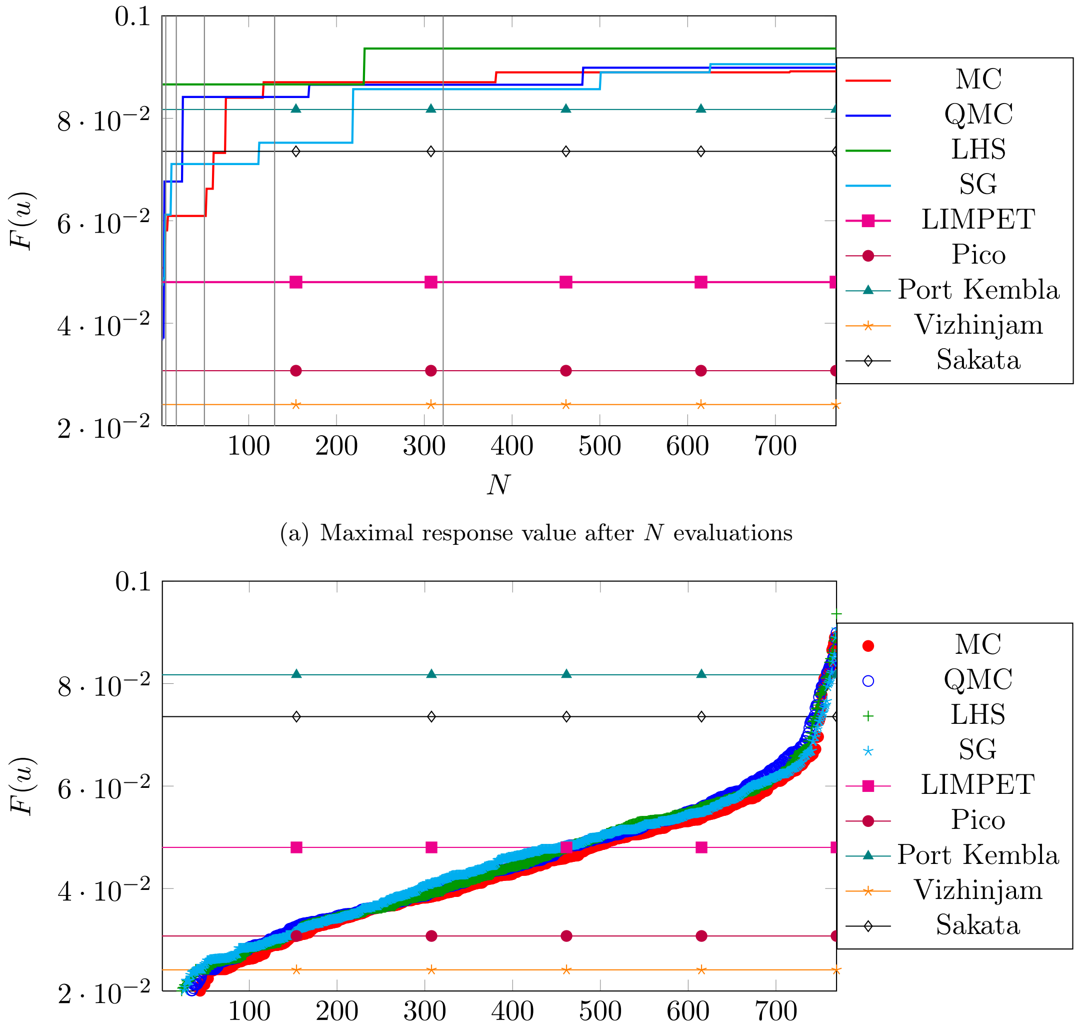
Figure 6.2(a) depicts the maximal response value found after each point evaluation, i.e. the convergence of the sampling procedure. The vertical lines mark each level (of the sparse grid design). Each experimental design contains 769 sample points, corresponding to a sparse grid of level six. The values sorted by magnitude without respect to the order of evaluation are shown in figure 6.2(b). This demonstrates that all experiment designs essentially cover the same spectrum of values. The horizontal lines mark the responses for the five reference power plants. For a fairer comparison, the values of b, c, β, f and g have also been fixed to the values and only a and α are determined by table 3.2.1. All sample point designs were able to find an improvement to the best performing reference OWC, namely the device at Port Kembla with a response value of 0.08171833. All responses are presented dimensionless, since a unit of measure does not make sense for the objective function — which is determined by the average flow (m/s), the volume penalization (m^3) and the boundary penalty term.

The metamodel is constructed with a Gauss correlation model, i.e.

$$\mathcal{R}(w, x) = \prod_{i=1}^{2} \exp(-\theta_i (w_i - x_i)^2)$$

with fixed values $\theta_{1,2} = 0.03$. Although the DACE Toolbox includes an algorithm to find the optimal value of θ_i in each dimension, we choose to fix its value for a better comparison of the different designs. Second order polynomials are used as regression functions. The penalization constants in equation (3.4.1) are set to $\lambda_1 = 7 \cdot 10^{-5}$ and $\lambda_2 = 6 \cdot 10^{-2}$.

Figure 6.3.3 shows the contour plots of the metamodels resulting from the four different set of sample points along with the calculated optimal values.



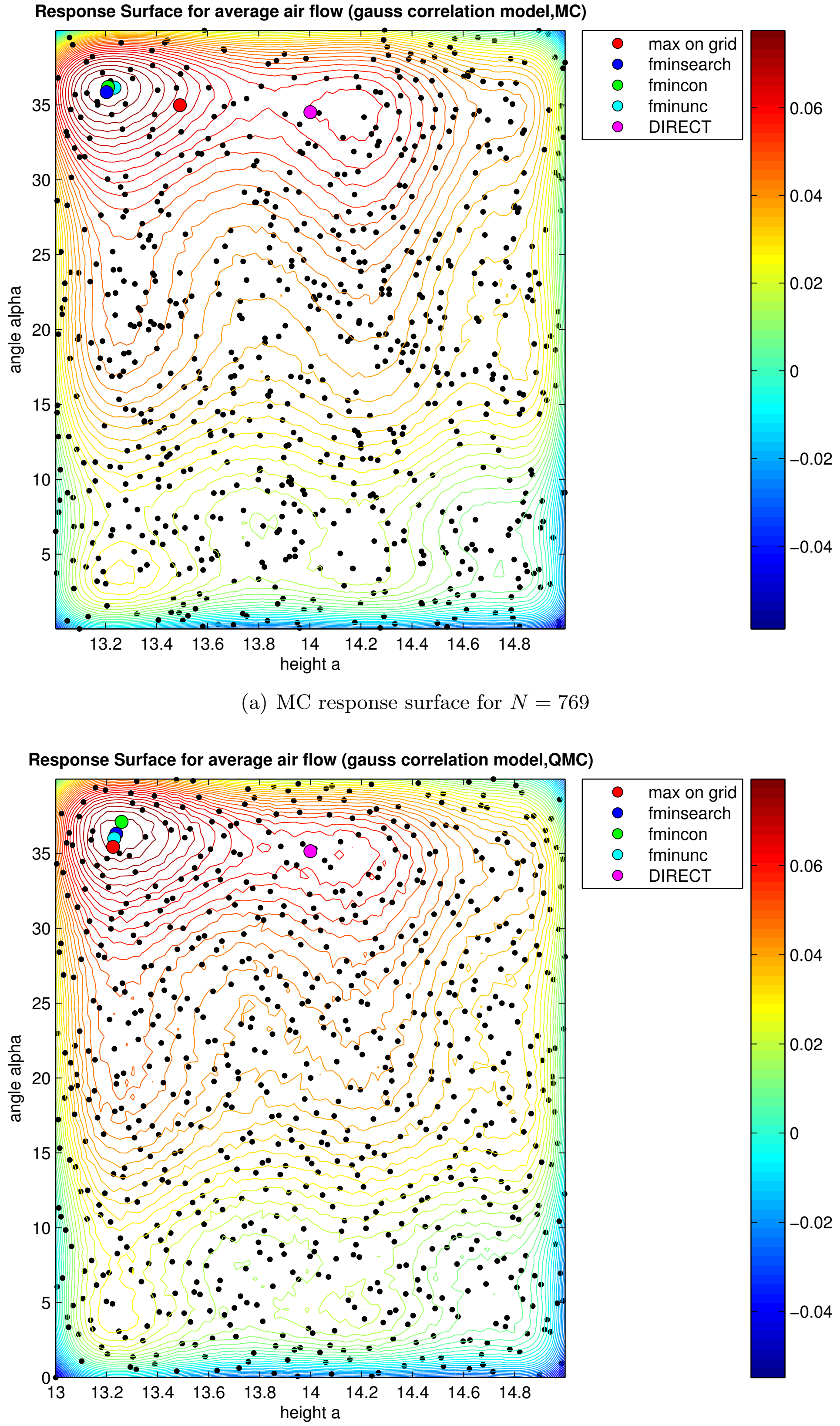
(a) Maximal response value after N evaluations

(b) Response values sorted by magnitude

Fig. 6.3.2: Response values for MC, QMC, LHS and SG for 769 sample points (2D).

First, all optimization algorithms are given the same initial guess $\boldsymbol{s}_0 = (14.0, 20.0)$, i.e. the center of the parameter space. The maximal number of function evaluations is set to 10000 and each algorithm is allowed enough iterations to reach these. However, both gradient–based algorithms do not converge and stay at a local maximum close to the center. The reason for this is a very small gradient in this area. This leads to small trial steps in each iteration, which is a stopping criterion for the algorithm. The direct search methods also fail to find the global maximum, but are able to find larger local maxima than the other two algorithms.

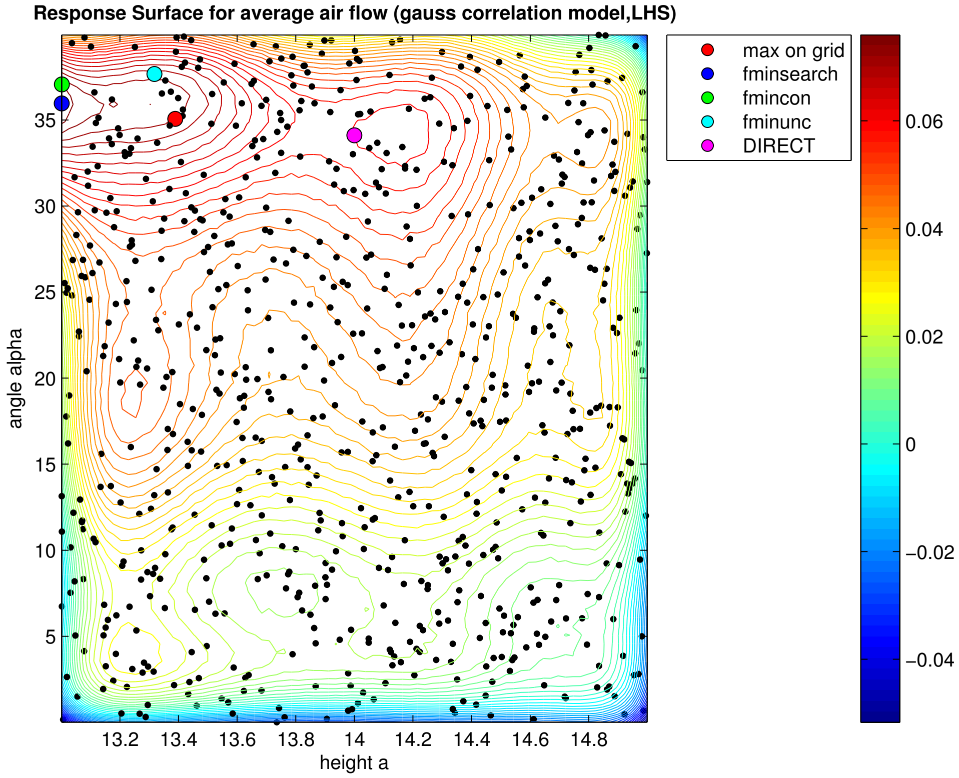
The solution to this problem is to use several different initial starting points. Hence, the algorithms are started several times with random initial guesses inside the domain and the best solution is taken. This yields the global maximum around $(a, \alpha) = (13.20434535.847820)$, c.f. figure 6.3(a). Table 6.3.1 summarizes the maxima found among the sample points, i.e. without the metamodel, and the maximal value found with the optimization algorithms.



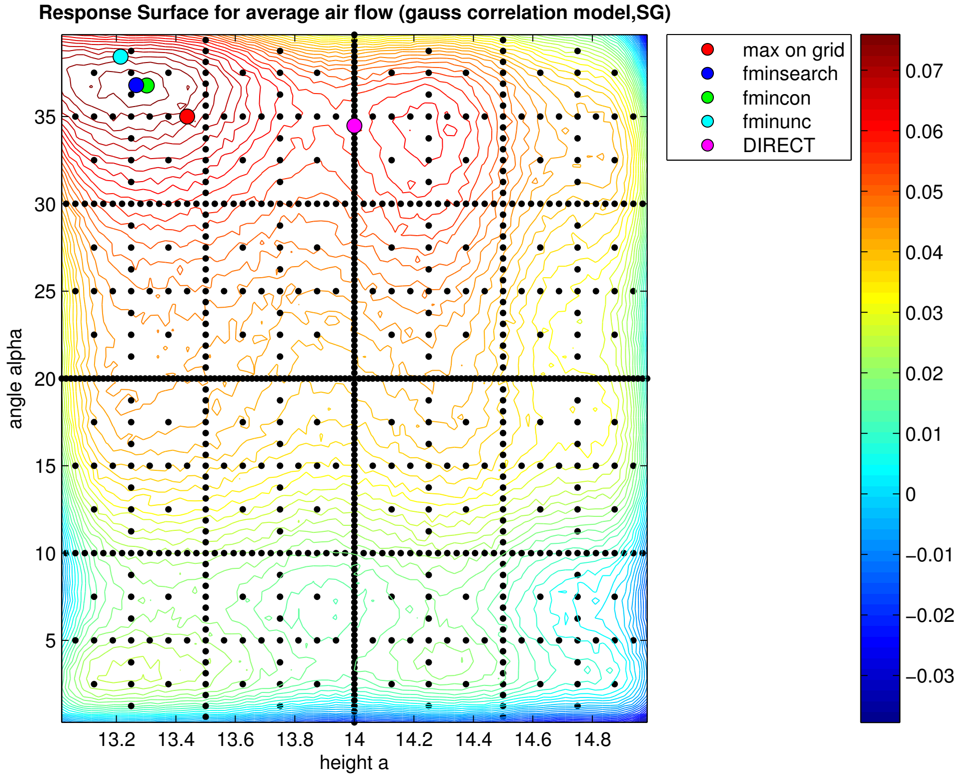
(a) MC response surface for $N = 769$

(b) QMC response surface for $N = 769$

Fig. 6.3.3: Response surfaces for different designs for N sample points.

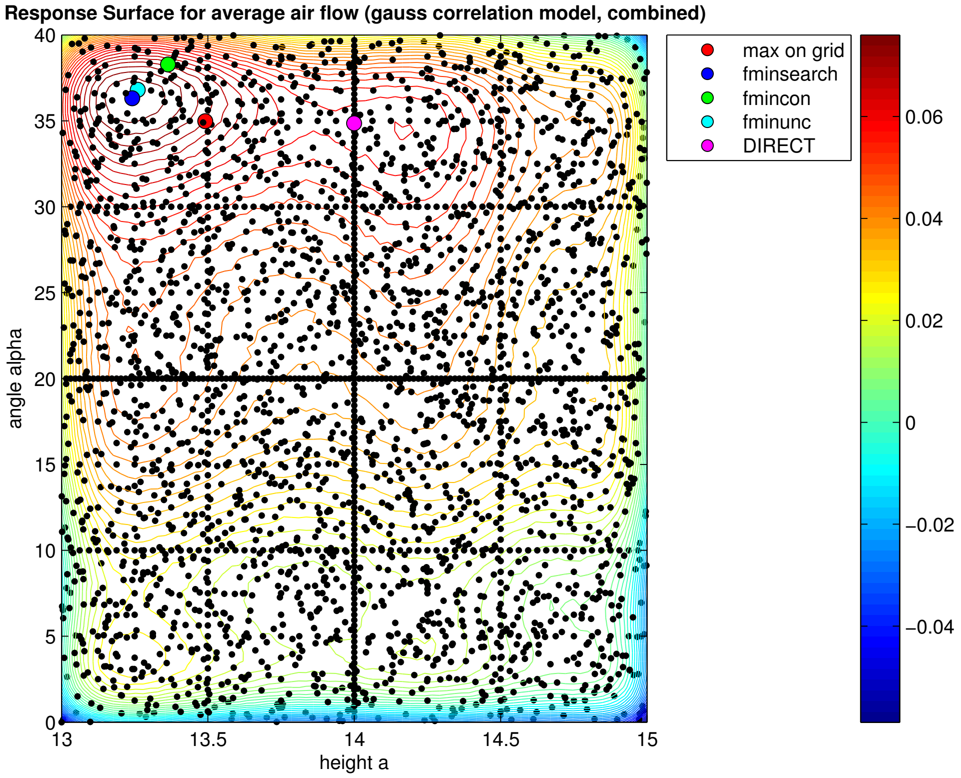


(c) LHS response surface for $N = 769$



(d) LHS response surface for $N = 769$

Fig. 6.3.3: Response surfaces for different designs for N sample points (cont.).



(e) Combined response surface for $N = 3061$

Fig. 6.3.3: Response surfaces for different designs for N sample points (cont.).

Note, that the maxima on the respective grids are greater than the ones calculated with the response surface. This is due to the Gauss correlation function, which smooths the metamodel and does not interpolate the responses. Nevertheless, the surrogate model suggests, that there is a global maximum on the interval $[13.0, 13.3] \times [35.9, 36.8]$, i.e. in the upper left corner in figure 6.3.3. The best value, as shown in table 6.3.1 was always achieved by the `fminsearch` algorithm, i.e. a direct search via the Nelder–Mead–Simplex method.

Table 6.3.1: Optimization results with and without the metamodel in 2D.

design	max. value (without metamodel)	(a, α)	max. value (with metamodel)	(a, α)
MC	0.088798	(13.49100034.983000)	0.080618	(13.20434535.847820)
QMC	0.088106	(13.22500035.430000)	0.082245	(13.23716636.303537)
LHS	0.088573	(13.38900035.062000)	0.078473	(13.00122335.964302)
SG	0.088667	(13.43800035.000000)	0.077735	(13.26692836.799194)
all	0.088798	(13.49100034.983000)	0.078810	(13.24149836.315449)

6.4 Case 2: 6D optimization

The second setup is more complex and involves all the ingredients described in this thesis. As mentioned above, the full eleven–dimensional case is very time consuming and requires a very large high–performance computing (HPC) cluster. Therefore, a reduced six–dimensional example is investigated. The harbor walls and width measurements are neglected and only the parameters a, b, c, α, β and g are considered. Furthermore, the waves are generated by a piston wave maker with prescribed velocity function as specified in section 6.1. Therefore, the length of the wave tank needs to be increased to ensure an appropriate distance between the wave maker and the OWC device. Additionally, the width is increased, because the wave maker requires a minimum distance from the wave tank walls (note, that this is due to implementation issues; in theory, the wave maker can touch the walls). The height is slightly raised as well to give more room for the out flowing air in order to avoid any influence of the boundary on the air flow. The total measurements are $350 \times 50 \times 40$ with a meshsize of one meter. The simulation time is 35 seconds. On average, each simulation takes about three to four hours on 32 cores. A total of 461 sample points are selected for the sparse grid design. These are 545 points corresponding to a sparse grid of level three, minus the points rejected by the angular constraint $c_{\boldsymbol{S}} + d(\alpha_{\boldsymbol{S}}) > e(\beta_{\boldsymbol{S}}) + g_{\boldsymbol{S}} + 4.0$. For the other three designs, a total of 507 sample points is chosen. Since Monte Carlo and Lathin Hypercube are random designs, other configurations with less points would also be possible due to the angular constraint.

The same constants, correlation and regression functions are employed as in the previous example.

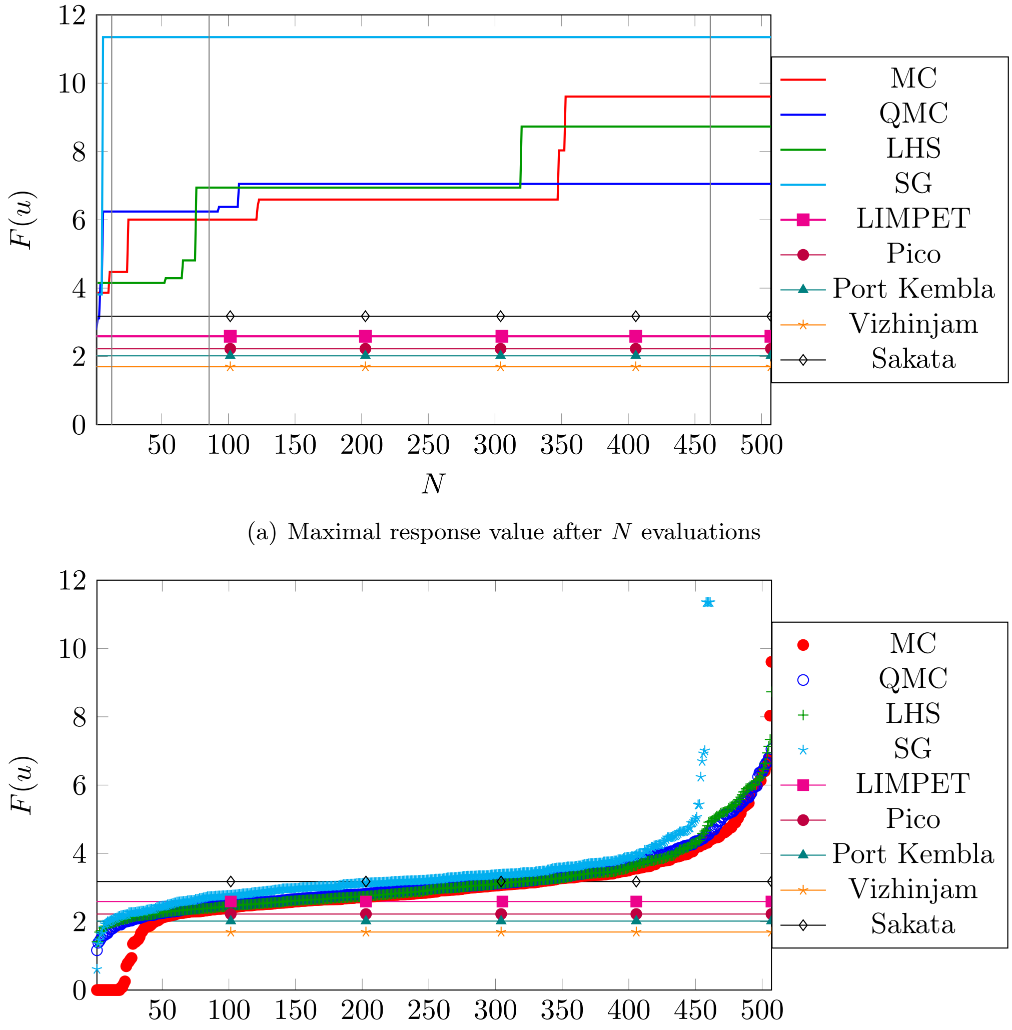
Figure 6.1(a) shows the maximal response value found during the point evaluation process, i.e. the convergence of the sampling procedure. All four designs are able to find a much greater response than the value of any of the reference OWC devices. The Sakata wave power plant shows the best response with a value of 3.17685602, whereas the maximal value found by the sparse grid design found a response of 11.342959 after only six point evaluations. Thus, an optimal response more than five times the highest value of the best performing reference device is already found before constructing the metamodel.

Since the problem is six–dimensional, the complete metamodel cannot be depicted. Instead, table 6.4.1 demonstrates the optimization process. The optimal values found by the optimization of the metamodel are shown. First of all, it should be noted, that an even higher value was found with the help of the metamodel. The sparse grid based model returned 11.984163, which is already a slight increase compared to its maximal value without the response surface. This value was found by both `fminsearch` and `fmincon`. Finally, the combined surrogate model delivered the optimal value of 12.090718, which was also determined by `fminsearch`. Since the six–dimensional metamodel cannot be plotted, it is not clear whether this maximum is the global maximum or not. But the data at least suggests, that the other three sets of sample points found different local maxima than the sparse grid design.

The final optimal design defined by

$$(a, b, c, \alpha, \beta, g) = (13.76, 3.82, 311.23, 24.99, 25.00, -0.03)$$

is depicted in figure 6.4.2.



(a) Maximal response value after N evaluations

(b) Response values sorted by magnitude

Fig. 6.4.1: Response values for 461 (SG) and 507 (MC, QMC, LHS) sample points (6D).

Table 6.4.1: Optimization results with and without the metamodel in 6D.

design	max. value (without metamodel)	$(a, b, c, \alpha, \beta, g)$
MC	9.520051	$(14.973000, 3.105000, 13.725000, 45.871000, 40.356000, 7.046000)$
QMC	7.036090	$(13.076000, 3.551000, 14.707000, 23.633000, 31.055000, -8.516000)$
LHS	8.646090	$(14.547000, 3.412000, 10.400000, 49.551000, 11.539000, -9.965000)$
SG	11.342959	$(13.750000, 3.875000, 11.250000, 25.000000, 25.000000, 0.000000)$
all	11.342959	$(13.750000, 3.875000, 11.250000, 25.000000, 25.000000, 0.000000)$
design	max. value (with metamodel)	$(a, b, c, \alpha, \beta, g)$
MC	8.088654	$(14.202786, 3.310767, 11.593907, 27.772063, 28.998330, -1.055030)$
QMC	6.807009	$(13.510053, 3.398983, 10.685681, 38.267004, 37.245975, -9.220258)$
LHS	7.332093	$(14.505524, 3.819245, 14.515013, 37.458303, 31.736926, -9.407385)$
SG	11.984163	$(13.750350, 3.827286, 11.252087, 25.000828, 24.999760, -0.002708)$
all	12.090718	$(13.757924, 3.81788, 311.231243, 24.992624, 25.003900, -0.028400)$

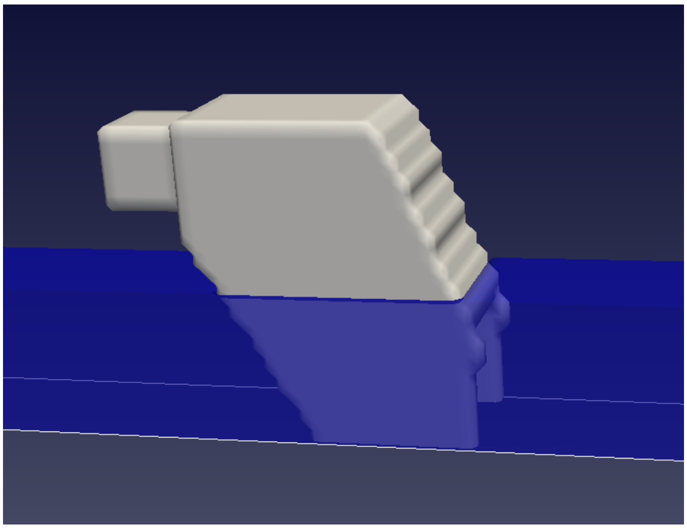


Fig. 6.4.2: Final optimal design.

7 Conclusion and Outlook

Summary

Within the scope of this thesis, different global optimization techniques have been discussed for the optimization of a special kind of wave energy converter — an Oscillating Water Columns device.

The first step was to parametrize the wave power plant with a set of parameters. With the help of five reference devices, that have actually been built and tested at sea, these parameters and their bounds were determined to define a high dimensional parameter space. Subsequently, an objective function was determined in order to define the value of interest. This function depends on an expensive CFD simulation, which cannot be represented by some algebraic formula and is thereafter treated as a blackbox. This means, that no information on the properties of the objective function are known, which adds to the difficulty of the global optimization problem.

In order to solve this, a number of sample points have been chosen within the paramter domain, for which the objective function was evaluated. These so–called designs, or sets of sample points, need to fulfill different criteria to serve as a suitable basis for a metamodel. This model was constructed by Kriging and could then be used for further optimization. This way, the expensive unknown blackbox function was transferred to a computationally cheap function defined by a surrogate model.

Two cases have been considered to optimize the design of the OWC. The first one was a simple two–dimensional test case and very condensed version of the actual problem. It was very useful to demonstrate the complete optimization procedure and to get a visual output as well. All four considered designs produced an improvement with respect to the reference designs and a global optimum was found with the help of the metamodel.

The second case was a six–dimensional study, which showed an even greater improvement than the first one. The sparse grid design was able to produce a significant improvement in the response compared to the best reference configuration and even more with the help of the metamodel. Even though we do not know if this is the global optimum, this (local) maximum is promising.

Outlook

The two cases discussed in this thesis are only two examples of possible parametrizations and optimizations of a wave energy converters. Both models were simplified versions of the original problem. A full model would include a turbine, have a higher resolution and a longer runtime. However, all of the factors contribute to the computational complexity and therefore also increase the required the computing time. With the help of large High–Performance Cluster Computers this can still be possible.

Moreover, the presented metamodels can be further investigated. The first step would be evaluate the optimal points determined by the surrogate in order to refine the model. This is the most common approach, along with the method used by the *EGO* (Efficient Global Optimization) algorithm [Jones et al., 1998]. The latter uses a metamodel obtained with DACE and its standard error to determine the *expected improvement* of the predictor. Accordingly, the next sample point is not only chosen by the current optimal value, but also by the improvement to be expected in largely unexplored areas of the parameter space.

Yet another kind of improvement can be achieved by a model reduction. To this end, a screening process via ANOVA (analysis of variance) decomposition or HDMR (high dimensional model representation) could determine the most important factors. The parameters with negligible influence are subsequently eliminated and further sampling can be used to define a refined, lower dimensional model.

In conclusion, global optimization — and in particular metamodelling techniques — have proven to be very useful for high dimensional design optimization of wave energy converters and there still exists a variety of ways to enhance its output.

Acknowledgments The author would like to thank the Institute for Numerical Simulation (INS) at the University of Bonn, where the largest part of the simulations were performed on the High–Performance Cluster Computers. Many thanks also to the High Performance Computing Center Stuttgart (HLRS) of the University of Stuttgart, where a small portion of the simulations was computed on the CRAY XE6 system (HERMIT).

List of Figures

Bibliography

[LIM, n.d.] (n.d.). Information from Voith Hydro about their owc projects. Retrieved on 8 May 2013 from `http://voith.com/en/products-services/hydro-power/ocean-energies/wave-power-plants-590.html`.

[Sak, n.d.] (n.d.). Picture of owc device at sakata port, japan. Retrieved on 8 May 2013 from `http://2nd.geocities.jp/kadolikes96jp/photo/sinrigan3_640.jpg`.

[Viz, n.d.] (n.d.). Picture of owc device at vizhinjam, india. Retrieved on 8 May 2013 from `http://www.vizhinjamport.org/images/OWC_vizhinjam.jpg`.

[MK1, n.d.] (n.d.). Picture of owc device mk1 at port kembla, australia. Retrieved on 8 May 2013 from `http://oceanlinx.com/images/Projects/summary/1.%20MK1%20440x200.png`.

[Pic, n.d.] (n.d.). Picture of owc device on the pico island, azores. Retrieved on 8 May 2013 from `http://www.pico-owc.net/files/33/gal_9cfdf10e8fc047a44b08ed031e1f0ed1.jpg`.

[Affholder and Valiron, 2001] Affholder, M. and Valiron, F. (2001). *Descriptive physical oceanography*. Taylor & Francis.

[Airy, 1841] Airy, G. B. (1841). Tides and waves.

[Beena and Ganguli, 2010] Beena, P. and Ganguli, R. (2010). Response surface approximation using sparse grid design. *International Journal of Computational and Applied Mathematics*, 5(4):459–478.

[Bellman, 1961] Bellman, R. (1961). Adaptive control processes.

[Bihs, 2011] Bihs, H. (2011). *Three-Dimensional Numerical Modeling of Local Scouring in Open Channel Flow*. PhD thesis, Norwegian University of Science and Technology.

[Boake et al., 2002] Boake, C. B., Whittaker, T. J., Folley, M., Ellen, H., et al. (2002). Overview and initial operational experience of the limpet wave energy plant. In *Proceedings of the 12th international offshore and polar engineering conference*, volume 1, pages 26–31.

[Box and Behnken, 1960] Box, G. E. and Behnken, D. (1960). Some new three level designs for the study of quantitative variables. *Technometrics*, 2(4):455–475.

[Box and Wilson, 1951] Box, G. E. and Wilson, K. (1951). On the experimental attainment of optimum conditions. *Journal of the Royal Statistical Society. Series B (Methodological)*, 13(1):1–45.

[Bungartz and Griebel, 2004] Bungartz, H.-J. and Griebel, M. (2004). Sparse grids. *Acta Numerica*, 13:1–123.

[Chen et al., 1999] Chen, G., Kharif, C., Zaleski, S., and Li, J. (1999). Two-dimensional Navier-Stokes simulation of breaking waves. *Physics of Fluids*, 11(1):121–133.

[Choi and Yoon, 2009] Choi, J. and Yoon, S. (2009). Numerical simulations using momentum source wave-maker applied to RANS equation model. *Coastal Engineering*, 56(10):1043–1060.

[Chorin, 1968] Chorin, A. (1968). Numerical Solution of the Navier-Stokes Equations. *Mathematics of Computation*, 22:745–762.

[Clément et al., 2002] Clément, A., McCullen, P., Falcão, A., Fiorentino, A., Gardner, F., Hammarlund, K., Lemonis, G., Lewis, T., Nielsen, K., Petroncini, S., et al. (2002). Wave energy in Europe: current status and perspectives. *Renewable and Sustainable Energy Reviews*, 6(5):405–431.

[Craik, 2004] Craik, A. D. (2004). The origins of water wave theory. *Annu. Rev. Fluid Mech.*, 36:1–28.

[Croce et al., 2009] Croce, R., Griebel, M., and Schweitzer, M. A. (2009). Numerical Simulation of Bubble and Droplet-Deformation by a Level Set Approach with Surface Tension in Three Dimensions. *International Journal for Numerical Methods in Fluids*, 62(9):963–993. Also available as SFB 611 Preprint no 431.

[Dean and Dalrymple, 1990] Dean, R. and Dalrymple, R. (1990). *Water wave mechanics for engineers and scientists*, volume 2 of *Advanced Series on Ocean Engineering*, chapter 6, pages 170–185. World Scientific Publishing.

[Drew et al., 2009] Drew, B., Plummer, A., and Sahinkaya, M. N. (2009). A review of wave energy converter technology. *Proceedings of the Institution of Mechanical Engineers, Part A: Journal of Power and Energy*, 223(8):887–902.

[Falcão, 2010] Falcão, A. F. d. O. (2010). Wave energy utilization: A review of the technologies. *Renewable and sustainable energy reviews*, 14(3):899–918.

[Finkel, 2003] Finkel, D. E. (2003). Direct optimization algorithm user guide. *Center for Research in Scientific Computation, North Carolina State University*, 2.

[Forum, 1993] Forum, T. M. (1993). Mpi: A message passing interface.

[Galanti and Jung, 1997] Galanti, S. and Jung, A. (1997). Low-discrepancy sequences: Monte carlo simulation of option prices. *The Journal of Derivatives*, 5(1):63–83.

[Gerstner and Griebel, 1998] Gerstner, T. and Griebel, M. (1998). Numerical integration using sparse grids. *Numerical algorithms*, 18(3-4):209–232.

[Giunta et al., 2003] Giunta, A. A., Wojtkiewicz, S. F., Eldred, M. S., et al. (2003). Overview of modern design of experiments methods for computational simulations. In *Proceedings of the 41st AIAA Aerospace Sciences Meeting and Exhibit, AIAA-2003-0649*.

[Graw, 1995] Graw, K.-U. (1995). *Wellenenergie - eine hydromechanische Analyse*. Habilitation, Bergische Universität - Gesamthochschule Wuppertal.

[Griebel et al., 1998] Griebel, M., Dornseifer, T., and Neunhoeffer, T. (1998). *Numerical Simulation in Fluid Dynamics, a Practical Introduction* . SIAM, Philadelphia.

[Ha et al., 2011] Ha, T., Lee, J., and Cho, Y.-S. (2011). Internal wave maker for navier-stokes equations in a three-dimensional numerical model. *JOURNAL OF COASTAL RESEARCH*, pages 511–515.

[Hudson et al., 2001] Hudson, P., McCormick, M., and Browne, S. (2001). A Low-Cost Wave-Sediment-Towing Tank.

[Jones et al., 1993] Jones, D., Perttunen, C., and Stuckman, B. (1993). Lipschitzian optimization without the lipschitz constant. *Journal of Optimization Theory and Applications*, 79(1):157–181.

[Jones et al., 1998] Jones, D. R., Schonlau, M., and Welch, W. J. (1998). Efficient global optimization of expensive black-box functions. *Journal of Global optimization*, 13(4):455–492.

[Kamath, 2012] Kamath, A. M. (2012). *Calculation of Wave Forces on Structures using REEF3D*. PhD thesis, Norwegian University of Science and Technology.

[Keith A. Sverdrup and Duxbury, 2000] Keith A. Sverdrup, A. C. D. and Duxbury, A. (2000). An introduction to the world's oceans.

[Kim et al., 1999] Kim, C., Clement, A., and Tanizawa, K. (1999). Recent Research and Development of Numerical Wave Tanks - A Review. *International Journal of Offshore and Polar Engineering*, 9:241–256.

[Krige, 1951] Krige, D. (1951). *A Statistical Approach to Some Mine Valuation and Allied Problems on the Witwatersrand: By DG Krige*. PhD thesis, University of the Witwatersrand.

[Larsen and Dancy, 1983] Larsen, J. and Dancy, H. (1983). Open boundaries in short wave simulations—a new approach. *Coastal Engineering*, 7(3):285–297.

[Lin and Liu, 1999] Lin, P. and Liu, P.-F. (1999). Internal Wave-Maker for Navier-Stokes Equation Models. *Journal of Waterway, Port, Coastal, and Ocean Engineering*, 125(4):207–215.

[Lophaven et al., 2002] Lophaven, S. N., Nielsen, H. B., and Søndergaard, J. (2002). Dace - a matlab kriging toolbox. *Technical University of Denmark, Kongens Lyngby, Technical Report No. IMM-TR-2002-12*.

[Matheron, 1962] Matheron, G. (1962). *Traité de géostatistique appliquée*, volume 1. Editions Technip.

[McKay et al., 1979] McKay, M. D., Beckman, R. J., and Conover, W. J. (1979). Comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 21(2):239–245.

[Mikkola, 2007] Mikkola, T. (2007). Simulation of Plunger type wave makers. *Journal of Structural Mechanics*, 40(4):19–39.

[Miles, 1957] Miles, J. W. (1957). On the generation of surface waves by shear flows. *J. Fluid Mech*, 3(Pt 2):185–204.

[Morokoff and Caflisch, 1995] Morokoff, W. J. and Caflisch, R. E. (1995). Quasi-monte carlo integration. *Journal of computational physics*, 122(2):218–230.

[Munk, 1950] Munk, W. (1950). Origin and generation of waves. *Proceedings of the International Conference on Coastal Engineering*, 1(1).

[Nakayama and Yamamoto, 2005] Nakayama, Y. and Yamamoto, R. (2005). Simulation method to resolve hydrodynamic interactions in colloidal dispersions. *Phys. Rev. E*, 71(3):036707.

[Osher and Sethian, 1988] Osher, S. and Sethian, J. A. (1988). Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton-Jacobi formulations. *Journal of Computational Physics*, 79(1):12 – 49.

[Park et al., 2001] Park, J., Y.Uno, Matsuo, H., Sato, T., and Miyata, H. (2001). Reproduction of Fully-Nonlinear Multi-Directional Waves by a 3D Viscous Numerical Wave Tank. In *Proceedings of the Eleventh (2001) International Offshore and Polar Engineering Conference*.

[Peuker, 2011] Peuker, C. (2011). Numerische Erzeugung von Wasserwellen zur Analyse von Wellenkraftwerken. Bachelor thesis, Institut für Numerische Simulation, Universität Bonn.

[Phillips, 1957] Phillips, O. M. (1957). On the generation of waves by turbulent wind. *Journal of fluid mechanics*, 2(05):417–445.

[Plackett and Burman, 1946] Plackett, R. L. and Burman, J. P. (1946). The design of optimum multifactorial experiments. *Biometrika*, 33(4):305–325.

[Pontes et al., 2005] Pontes, M. T., Aguiar, R., and Pires, H. O. (2005). A nearshore wave energy atlas for portugal. *TRANSACTIONS-AMERICAN SOCIETY OF MECHANICAL ENGINEERS JOURNAL OF OFFSHORE MECHANICS AND ARCTIC ENGINEERING*, 127(3):249.

[Rios and Sahinidis, 2012] Rios, L. M. and Sahinidis, N. V. (2012). Derivative-free optimization: A review of algorithms and comparison of software implementations. *Journal of Global Optimization*, pages 1–47.

[Sacks et al., 1989] Sacks, J., Welch, W., Mitchell, T., and Wynn, H. (1989). Design and analysis of computer experiments. *Statistical science*, 4(4):409–423.

[Senturk, 2011] Senturk, U. (2011). Modeling nonlinear waves in a numerical wave tank with localized meshless RBF method. *Computers & Fluids*, 44(1):221–228.

[Shan and Wang, 2010] Shan, S. and Wang, G. G. (2010). Survey of modeling and optimization strategies to solve high-dimensional design problems with computationally-expensive black-box functions. *Structural and Multidisciplinary Optimization*, 41(2):219–241.

[Shubert, 1972] Shubert, B. O. (1972). A sequential method seeking the global maximum of a function. *SIAM Journal on Numerical Analysis*, 9(3):379–388.

[Smolyak, 1963] Smolyak, S. A. (1963). Quadrature and interpolation formulas for tensor products of certain classes of functions. In *Dokl. Akad. Nauk SSSR*, volume 4, page 111.

[Stokes, 1847] Stokes, G. G. (1847). On the theory of oscillatory waves. *Trans Cambridge Philos Soc*, 8:441–473.

[Sussman et al., 1994] Sussman, M., Smereka, P., and Osher, S. (1994). A level set approach for computing solutions to incompressible two-phase flow. *J. Comput. Phys.*, 114:146–159.

[Takahashi et al., 1992] Takahashi, S., Nakada, H., Ohneda, H., and Shikamori, M. (1992). Wave power conversion by a prototype wave power extracting caisson in sakata port. *Coastal Engineering Proceedings*, 1(23).

[Tanizawa, 2000] Tanizawa, K. (2000). The State of the Art on Numerical Wave Tank. In *Proc. of 4th Osaka Colloquium on Seakeeping Performance of Ships*, pages 95–114.

[Témam, 1969] Témam, R. (1969). Sur l'approximation de la solution des équations de Navier-Stokes par la méthode des pas fractionnaires (II). *Archive for Rational Mechanics and Analysis*, 33:377–385. 10.1007/BF00247696.

[Wang et al., 1995] Wang, P., Yao, Y., and Tulin, M. (1995). An efficient numerical tank for non-linear water waves, based on the multi-subdomain approach with BEM. *International Journal for Numerical Methods in Fluids*, 20(12):1315–1336.

[Watanabe et al., 2005] Watanabe, Y., Seaki, H., and Hosking, R. (2005). Three-dimensional vortex structures under breaking waves. *Journal of Fluid Mechanics*, 545:291–328.

[Webb et al., 2005] Webb, I., Seaman, C., and Jackson, G. (2005). Marine energy challenge: Oscillating water column wave energy converter evaluation report. *The Carbon Trust.*

[Weise, 2009] Weise, T. (2009). *Global Optimization Algorithms – Theory and Application.* it-weise.de (self-published): Germany.