Detection and Identification of Phytoplankton Assemblages using Case-Based Reasoning

Author: Albert Vilamala
Date: 06/07/2010
Advisors: Enric Plaza and Josep Lluis Arcos
Detection and Identification of Phytoplankton Assemblages using Case-Based Reasoning

Albert Vilamala
III, Artificial Intelligence Research Institute
CSIC, Spanish National Research Council
Campus UAB, E-08193, Bellaterra, Spain

Editor: Autonomous University of Barcelona

Abstract

Recent studies in marine regions have determined the existence of phytoplankton and zooplankton assemblages near the surface level. Such patches, named thin layers, are of interest for biologists due to their important impact on the ecosystem where they lay in, such as phytoplankton growth dynamics or predation, just to name some. Therefore, new tools to ease the task of biologists are being under development.

The purpose of this project is to develop a system capable of detecting and characterizing thin layers from hyperspectral optical sensors. Specifically, a Particle Swarm Optimization technique has been used to detect thin layers and the Case-Based Reasoning methodology, which takes advantage of previous experience in order to adapt it for new situations, has been applied to successfully characterise them.

Keywords: Case-Based Reasoning, Particle Swarm Optimization, Hyperspectral Optical Sensors.

1. Introduction

For many years, the observation of the flora and fauna in oceanic environments carried out by biologists was directly performed in the field by sampling and posterior examination in a laboratory. This approach was not only costly and time consuming for the researchers but it could also change the environment of interest due to the invasive nature of such traditional techniques, increasing the difficulty of this sampling which would even lead the project to failure if those samples were not gathered with caution.

However, in the last few years, advances on hyperspectral optical sensoring have broadened up the possibilities for these biologists to proceed with their examinations in a more neat way. These new techniques take advantage of the optical properties present in the matter that can be found in underwater systems allowing for the monitoring of their dynamics.

As a result, new tools able to work in-situ without much environment disturbance are being developed, with the advantages that those features represent. Phytoplankton is one of the most recurrent components that are sought using optical sensors. The reason for this fact is the well-known optical property of its main component—the chlorophyll—when daylight comes into contact with it.
On the other hand, increasingly more biologists believe in the presence of thin plankton patches near the surface of marine regions contributing to important impacts on the ecosystems where they have been found. This fact has grown the interest of the community in performing research.

The current research project arises from coupling the two concepts explained above in an attempt to use the new techniques available in the field of marine sampling applied to identify optical significant thin phytoplankton assemblages, namely thin layers (Dekshenieks et al., 2001; Stramska and Stramski, 2005).

This master thesis report the study of two Artificial Intelligence techniques applied on noisy hyperspectral optical data from marine water columns for detecting and characterising thin layers. The resulting system performs in a two step basis: the first phase locates such layers using Particle Swarm Optimization and the second one uses Case-Based Reasoning to determine some of their different features.

The main challenges these techniques must overcome are: first, the capability of properly work on data highly affected by noise and second, due to the early stage of the field under treatment, the difficulty of building a tool wise enough to confront its duties using the currently shallow existing knowledge in the domain.

This monograph is composed of the following sections: first a brief overview on oceanographic concepts and sensors is carried out to introduce the problem to solve; next, some cutting edge AI methods are reviewed followed by a section where these techniques are applied and deeply explained to approach our problem; later, the experiments to evaluate the system performance are commented and finally, some conclusions are stated as well as new future research lines.

2. Marine background

This section introduces some concepts on oceanography such as the definition of the term thin layer and a review of the hyperspectral optical sensors currently available, together with the theory that supports them.

2.1 Thin layers

Thin layers are plankton assemblages present in water column near the surface level, usually in the photic zone in coastal waters (Sieburth and Donaghay, 1993; Johnson et al., 1995; Carpenter et al., 1995). They are composed by either phytoplankton or zooplankton, or a mixture of them. It has been observed that their thickness may range from approximately 10cm up to nearly 3.5m (Donaghay et al., 1992; Cowles and Desiderio, 1993; Hanson and Donaghay, 1998).

As stated in Nielsen et al. (1990); Bjørnsen and Nielsen (1991); Rines et al. (2002), their horizontal size is very variable and it can even extend for several kilometres. Regarding distribution, it varies among space, meaning that different compounds can be found at different places within a thin layer. This same variability is found in the time dimension, that is, thin layers may persist for several days in a certain location.

---

1. Thickness is the distance between the thin layer’s upper level and the bottom one.
Such structures have an important impact for the ecosystems where they live in, since they may alter the biological structure and dynamics of the system (Nielsen et al., 1990; Donaghay and Osborn, 1997; Cowles et al., 1998), as well as its acoustical (Holliday et al., 1998) and optical (Zaneveld and Pegau, 1998) properties.

According to Dekshenieks et al. (2001), thin layers occur more often in certain locations than others, related to physical conditions which are still not known in depth. Nevertheless, even lacking a strong theory about those conditions, it is known that in turbulent waters such as tidally mixed regions or those having mixed wind on surface, thin layers do not happen or very rarely. Contrarily, they are often located in pycnoclines, according to its depth and strength.

2.2 Radiative transfer theory

The sensors presented in this project take advantage of the optical properties of each element in matter existing in the water column that is to be examined. In particular, they are based on the radiative transfer theory.

Radiative transfer theory explains how a sunlight ray that has come into contact with a surface is modified according to the nature of the matter it has impacted. That is, each compound owns a set of properties called inherent optical properties (IOPs), which depend exclusively on the target matter itself and is completely independent from the light field. Hence, whenever a light beam comes into contact with an element, this light can be either absorbed by the matter or scattered. The addition of these two components results in what is known as attenuation. Such properties are measured in form of coefficients directly tied to the element.

On the other hand, if the observer does not focus on the IOPs but the resulting light field after the light beam has come into contact with the matter, the apparent optical properties (AOPs) arise from the properties of this light. In that sense, we can distinguish the irradiance, either downwelling (E_d) or upwelling (E_u) (i.e., whether the examined field is on top of the matter of observation or under it), the reflectance, which is the ratio between E_d and E_u and the diffuse attenuation coefficient (K), being the difference of two consecutive measures of E.

Despite IOPs and AOPs are the main properties used by hyperspectral optical sensors to characterise the environment, matter has other optical properties worth to mention such as fluorescence\(^2\) or luminescence\(^3\). These properties can also be measured using optical sensors, but they will not be set in this project though, given the fact that not all the matter we work with owns them.

An important point regarding IOPs and AOPs is their capacity to be used as proxies to determine the quantity of certain matter in a water column. Thus, this phenomenon will be applied in this project to detect and characterise thin layers.

2.3 Hyperspectral optical sensors

Hyperspectral optical sensors make the most of the radiative transfer theory to successfully perform their task. The main feature of those sensors, used by the literature to classify

---

2. Fluorescence is the property of re-emit the absorbed light in a different wavelength.
3. Luminescence is the property of releasing energy in form of light.
them, is whether they own a light source or not. The performance of each sensor type and the target optical properties depend on this distinct feature. Such classification is done by splitting them up into two main groups namely active and passive sensors.

Considering Perry (2003), active sensors are the ones that own a light source—either a LED or a lamp—in order to accomplish their function, which normally is to measure IOPs and fluorescence. The advantage of this kind of sensors is that they are capable of measuring in very deep waters, where the sun light barely arrives or not at all. However, their price is a bit more expensive compared to passive sensors and the power consumption is also higher.

Passive sensors are characterised for lacking an emitting light source since their task is performed by using the environmental light provided by the sun. They are mainly used to measure AOPs and bioluminescence. In contrast to active sensors, these ones can only work within the photic zone (i.e., 10m to 150m under the sea level) and only during daylight. The good point is that they are cheaper and they need less power consumption to work. These two facts are strong enough for deciding to use these sensors instead of the active ones for the current project.

3. State of the art

This section presents two main Artificial Intelligence techniques that have been applied in the current project development from their general form to the different contributions that researchers have done. More precisely, an evolutionary technique called Particle Swarm Optimization and a machine learning and reasoning approach called Case-Based Reasoning will be introduced.

3.1 Particle Swarm Optimization

Particle Swarm Optimization (PSO) is a technique that aims at finding local maxima solutions in a search space by combining a cognitive model with a social interaction model. This technique is inspired from the natural model of a flock of birds searching for corn. Even though each one of the birds has its individual purpose, they interact with each other in order to achieve a certain common goal. These birds and interactions among them follow specific rules. The next subsection introduces the PSO algorithm.

3.1.1 Basic algorithm

The PSO basic algorithm—developed by Kennedy and Eberhart (1995), who were influenced by the work done by Heppner and Grenander (1990)—is defined as a set of particles randomly initialised over a search space that keep moving around searching for local maxima until they have converged. At each algorithm iteration, the particles compute a fitness value which assesses the current position.

Notice that the particles are not moving blindly in the search space, but they are heading towards the best position that they know from their own experience. To achieve this behaviour each particle needs to keep information about its current position, its current velocity and the best position it has found so far. This set of features has been called the cognitive model, according to literature.
Furthermore, the social model also plays its role in order to improve the performance achieved by the self-driven particles. Specifically, the particles swarm keeps the global best position —according to the fitness function— its members have found so far. Notice that each particle communicates with the particles in its neighbourhood and is driven according to the best position found. While a single particle may not be able to find best global positions, the interaction among the particles of a swarm is what gives power to this algorithm.

The original PSO equations to update the velocity and position of each particle is the following one:

\[ \vec{v}_i = \vec{v}_i + \vec{U}(0, \phi_1)(\vec{b}_i - \vec{p}_i) + \vec{U}(0, \phi_2)(\vec{g} - \vec{p}_i) \]

\[ \vec{p}_i = \vec{p}_i + \vec{v}_i \]

where,

- \( \vec{p}_i \) is the current position of the particle \( i \);
- \( \vec{v}_i \) is the velocity of the particle \( i \);
- \( \vec{b}_i \) the best position found by the particle \( i \);
- \( \vec{g} \) the global best solution found by the particles; and
- \( \vec{U}(0, \phi_i) \) represents a vector of random numbers uniformly distributed in \([0, \phi_i]\).

However, research on this preliminary PSO algorithm leaded Shi and Eberhart (1998) to increase the algorithm performance by introducing a new parameter called inertia weight. Such parameter allows to have deeper control on the search scope and downplays the importance of the maximum velocity \( V_{\text{max}} \) —in the original PSO, each \( \vec{v}_i \) is kept within \([-V_{\text{max}}, +V_{\text{max}}]\). Intuitively, inertia weight parameter may be seen as the fluidity of particles on the search space. Therefore, a good strategy is to give high values to this parameter at the beginning of the algorithm in order to rapidly reach the regions of interest in the search space and then gradually lower the value of this parameter in order to focus on a specific space region.

Finally, Clerc and Kennedy (2002) proposed an improved method which is the canonical PSO known nowadays. In their study, they proposed to introduce some constriction coefficients which are useful to ensure fast convergence and eliminate the domain-specific maximum velocity parameter. One way of implementing the constriction coefficients is by adding the constant multiplier \( \chi \) (Equation 1). According to their experiments, best results are achieved when \( \chi \) is set to 0.7298 and the acceleration coefficients \( (\phi_i) \) are bounded between 0 and 2.05.

Formally, the movement of every particle in the canonical PSO is driven following the next two equations:

\[ \vec{v}_i = \chi(\vec{v}_i + \vec{U}(0, \phi_1)(\vec{b}_i - \vec{p}_i) + \vec{U}(0, \phi_2)(\vec{g} - \vec{p}_i)) \]

\[ \vec{p}_i = \vec{p}_i + \vec{v}_i \]

where \( \chi \) is the constant multiplier that ensures the convergence.
In order to successfully work with the PSO algorithm, according to Poli et al. (2007), there are several settings that are crucial. One of them is the population size (i.e., the global number of particles) which is usually experimentally set. There is also the need to focus on \( \phi_i \), which are used to balance the forces between the particle best position attraction and the global best position attraction. Choosing the proper initial position of each particle and its initial velocity is also a vital decision.

Another important factor is the selection of particles to update at each iteration. Standard PSO updates all particles in turn, but other strategies just select some of them to be updated (e.g., the best particles in order to easily reach its purpose or the worst ones to improve the performance of the system as a whole).

Also, particle dynamics is another issue that researchers pay attention to. Different approaches such as spherical particles, sliding particles, quantum effects or repulsion among particles have been applied with varied results. Moreover, sometimes particles move to a region where there is nothing of interest. In this case, methods to drive the affected particles out of this region are needed, such as moving the particles to the edges or re-initialising them.

The memory of particles is important as well. In the standard PSO, every particle has its own record about the best position and takes into account the global best position found. However, different approaches have been proposed. For instance, using just some of the surrounding particles’ memory to calculate the best maxima in the neighbourhood—the neighbouring particles might be fixed or might change dynamically.

Finally, improvements in the field of swarm adaptation—self addition or suppression of particles and parameter tuning—have recently being performed. Ongoing research along this path aims at finding simple systems able to self-adapt for optimality.

3.1.2 PSO variants

Using the canonical PSO explained above as the starting point, many researchers have applied different strategies for improving the overall PSO performance or redesigning it for specific purposes. We present the most relevant ones for our current work.

The traditional attempt is the so-called global best topology, where all the particles follow the single best position found by all the particles searching in the space. In contrast to this idea, there is the local best approach (Eberhart and Kennedy, 1995). Here, every particle is only influenced by the \( n \) closest particles, leading to a successful algorithm which searches for multiple local maxima at a time. A third algorithm is proposed by coupling these two techniques—start the algorithm using local best and changing it to the global search at the end of the run (Suganthan, 1999).

In Mendes (2004) PhD thesis, the author presents the fully informed particle swarm (FIPS). Using this technique, each particle is influenced by a stochastic average of all the neighbours. The results presented report fast convergence of particles, implying that the solution can be found more rapidly than using canonical PSO. As a drawback, Mendes points out the high dependency on the particles topology.

Different methods for searching the best neighbours that should drive a given particle include computing a distance ratio, which compares the fitness of the current particle with the fitness of the neighbours, together with a weighted Euclidean distance (Peram et al.,
selecting random size subpopulations of neighbours (Liang and Suganthan, 2005); organising the particles in a hierarchy-like structure where the most influent particles move up the hierarchy driving the particles that lay beneath them (Janson and Middendorf, 2005) or splitting the particles into several differently sized tribes, where the weakest particles in a competing tribe can be expelled and then they can join other less competitive tribes and vice-versa (Clerc, 2006).

Other research lines have pushed researchers to borrow techniques from several AI areas. Examples are Angeline (1998), who used genetic algorithms; Wei et al. (2002) who computed the velocity of each particle by means of an evolutionary algorithm reporting very good results; and Robinson et al. (2002) who switched from PSO to a genetic algorithm as the execution was being performed.

More research was conducted coupling PSO and hill-climbing (Poli and Stephens, 2004); PSO and ACO4 (Hendtlass, 2001; Holden and Freitas, 2005); and PSO and genetic programming Poli et al. (2005a,b).

Just before finishing this section, special attention must be paid to the multiple particle swarm optimization (mPSO). This technique presented by Blackwell and Branke (2006), proposes to use several swarms to search for multiple ever changing peaks — in location and height.

The existence of multiple swarms rambling in a search space, approaching different peaks, ensures the capability of the algorithm to find different local maxima, eventually finding the global maximum, avoiding to get stuck in a particular local maximum.

Moreover, defining an exclusion operator is a must in order to avoid several swarms approaching the same peak (Fernandez-Marquez and Arcos, 2009). This operator defines an exclusion radius that surrounds the current local maximum found by the swarm, keeping the particles from other swarms out of this circle. Also an anti-convergence operator is needed to keep two particles far enough for ensuring diversity in a given swarm.

It is worth mentioning the importance of choosing the proper swarm size (i.e., the number of particles that the swarm is composed of).

A snapshot of a working mPSO can be seen in Figure 1, where three swarms of ten particles each search for the three peaks existing in a sample domain.

![Figure 1: Evolution of mPSO in a three-peak environment.](image)

4. Ant Colony Optimization was initially proposed by Dorigo (1992).
3.1.3 PSO in noisy environments

PSO has been shown to be a robust method in front of noise according to Parsopoulos and Vrahatis (2001), who tested the algorithm by applying Gaussian distributed random noise to different well-known optimisation fitness functions.

Many researchers have attempted to improve PSO performance when influenced by noise: Pugh et al. (2005) propose a variant where each particle computes multiple evaluations for the same candidate solution to assess a fitness value. Despite increasing the algorithm performance, this method incurs in a higher computational cost when performing such multiple evaluations.

Other methods use a statistical sequential selection procedure to improve the accuracy of the function estimation (Bartz-Beielstein et al., 2007). Another one, called Partitioned Hierarchical PSO (Janson and Middendorf, 2006), organises the swarm neighbourhood in a tree hierarchy that is monitored to detect possible changes in dynamic and noisy environments. And finally, Fernandez-Marquez and Arcos (2009) use an evaporation mechanism to approach similar environments.

3.1.4 Applications

PSO in all its forms has been applied to a large number of real applications of different nature. According to a survey conducted by Poli (2007), who took the IEEE Xplore database in 2007 as a source of information, nearly 700 papers were of PSO applications. The most relevant categories which PSO has been applied are image and video analysis (7.6%); design and restructuring of electricity networks and load dispatching (7.1%); control applications (4.0%), applications in electronics and electromagnetics (5.8%); antenna design (5.8%); power generation and power systems (5.8%) and scheduling (5.6%). Other applications include from robotics to clustering passing through biological and medical applications, prediction and forecasting, fuzzy systems or music generation and games.

3.2 Case-Based Reasoning

Case-Based Reasoning (CBR) (Kolodner, 1993; Leake, 1996; Mantaras et al., 2005) is a lazy learning technique (i.e., stores instances in the learning step, delaying the induction for the problem solving phase) that uses experience from problems tackled in the past in order to seek for solutions that fit the current problem to be solved. CBR is based on the assumption that similar problems can be solved by reusing similar solutions.

3.2.1 CBR Basics

CBR defines a case as the item of information that contains both the problem to be solved and the solution to overcome such problem. Sometimes a case might also contain a value assessing the ‘goodness’ of the proposed solution.

Upon time, different problems in a specific domain are faced and solved, storing this information as cases in a case base. Once this case base is informative enough, the system can start using this experience to solve new problems by applying the CBR cycle (Fig. 2), which is made up of 4 processes:
Detection and Identification of Phytoplankton Assemblages using Case-Based Reasoning

Figure 2: Case-Based Reasoning cycle

First, in the retrieve process, a set of cases whose problem is similar to the problem at hand are sought in the case base and retrieved to be used as possible solutions.

Next, the reuse process comes into play. The purpose of it is to adapt each one of the cases' solution to fit the current problem, select the best one and actually apply it to solve the problem.

Assessing the performance of the solution applied to the current problem is called the revise process. This can be done either manually or automatically, depending on the domain.

Finally, the retain process is performed. Here, if the solution applied to the current problem is useful to be taken into consideration in the future—for its goodness, or on the contrary as a counterexample—the pair is stored in the case base in form of a new case.

The CBR cycle is repeatedly executed and its performance, in principle, increased over time (i.e., the more cases it stores, the better it performs). Nevertheless, different modifications have been applied to this cycle to fit the specific needs of every situation.

3.2.2 Foundations

CBR stands in the junction between several fields as different as cognitive science, some AI subfields such as representation, reasoning and machine learning as well as some influence of mathematics. In this section we review the contributions of each field in the foundations of CBR.

As explained in Mantaras et al. (2005), CBR deals with capturing and using specific experiences, borrowing some knowledge from cognitive sciences, mainly those involving the words experience, memory and analogy.

The work done by Tulving (1972) made a distinction between episodic and semantic memories in human reasoning, which coupled with the Smith and Medin (1981) exemplar view—a concept is defined as the set of exemplars that conform it—, together with the dynamic memory theory (Schank, 1982)—memory is organised in packets—set the basis of CBR.
Moreover, further research noticed that even though CBR shares many properties with human analogical reasoning, there is a big difference among them: while human analogical reasoning reasons on several domains, CBR reasons only on one specific domain.

Another computational model of human analogy reasoning worth to mention for its impact on CBR is that of Falkenhainer et al. (1990).

CBR has taken some techniques from knowledge representation and reasoning in AI rather than developing specific ones for its purposes. Among them, examples are description logics (Díaz-Agudo and González-Calero, 2001), feature terms (Plaza, 1995), and some well-known knowledge representation techniques (Koton, 1988; Aamodt, 1994).

It is also important to use efficient data structures and indices to properly store case information and retrieve cases in a fast enough way. For such purpose, literature explains that techniques used in the database field have been adopted.

CBR has made the most of machine learning techniques to deal with its needs regarding concept learning, that is, inducing a general concept from a set of positive and negative examples. According to Kibler and Aha (1987), concept learning can be achieved by storing the exemplars presented to the learner and whenever an unknown example is found, assign to it the concept class that better matches a specific criterion.

Mathematical methods are broadly used in CBR, mainly aiming at computing similarity measures and learning. Measures are mathematical functions to assess utility. The main class of similarity measures are linear functions with several domain specific coefficients.

To sum up, CBR is made of several different-nature well-established fields, but further research that studies the relationship among the disciplines should be done for a major understanding and improvement of the techniques used in CBR.

### 3.2.3 Representation

In 3.2.1 the basic composition of a case —a problem and a solution— has been explained. However, Kolodner (1993) proposes five pieces that, from her point of view, a case must contain: a situation and its goal; the solution and how it has been derived; the result of carrying it out; explanations of the results; and lessons that can be learned from the experience.

According to Bergmann et al. (2006) traditional ways to represent cases in CBR fall into three main categories: feature vector representations, structured representations and textual representations. Furthermore, some advanced and task specific approaches have also appeared.

Feature vector representation consists of a set of exemplars that represent a category. Therefore, different categories are represented by their particular exemplars. A new case is classified to belong to a category whenever the case that best matches the current one belongs to that category. An example can be appreciated in the PROTOS system (Porter et al., 1990).

Plaza (1995) explains how cases can be represented using feature logics, a form of frame-based representations.

Borrowing ideas from data modelling, object-oriented case representations mimic the object-oriented paradigm to represent cases as a set of objects (Bergmann, 2002).
Finally, the *textual representations* (Lenz and Burkhard, 1996) use documents or texts as cases. In this approach a case is built from text by analyzing it to find the relevant information entities. These cases are organised into a retrieval net whose nodes are linked according to their similarity.

More sophisticated approaches include using *hierarchical representations* where different levels of abstraction are used to represent a case; *generalised cases* where a case covers a subspace of the representation space rather than a single point in that space; and specific representations for particular tasks such as planning (e.g., CHEF by Hammond (1989)) and design (e.g., the FABEL project by Gebhardt et al. (1997)).

### 3.2.4 Retrieve

The retrieval phase of CBR has traditionally focused on resemblance of cases by calculating similarity between problems, since it is based on the statement that similar problems have similar solutions. Nonetheless, some authors have questioned this assumption.

Some approaches assess the similarity between cases by means of their *surface* features. This includes computing a similarity measure and picking the k most similar cases\(^5\). Another way to compare two cases consists in computing the similarity of every feature and finally calculating the global similarity measure by weighting the partial measures. Smyth and McKenna (1998, 2001) proposed a method that is made of two stages: the first one retrieves a subset of possible cases —using a lightweight similarity measure— and the second one refines the search by actually retrieving the interesting ones.

Other authors have reported results on measuring *structural similarities* between cases in certain domains. An example of it is the work conducted by Börner (1983), who defined the structural similarity as the most specific graph structure that two problems have in common and a set of transformation rules needed to determine this structure. Similarity among hierarchical-structured cases can be assessed by searching the cases that are closer in the hierarchy.

Brown (1994) proposed to represent case memory as an interconnected network of nodes capturing case attribute-value combinations.

Bunke and Messmer (1994) presented similarity measures for domains where cases are represented as graphs. They proposed to use graph editing operations as the metrics to calculate the similarity.

Sometimes computing a similarity measure for every pair of cases on runtime is not feasible. For those situations it is wise to store similar cases close to each other and index them.

In order to improve similarity measures, literature provides us with different attempts: The measure can be amended by applying an *adaptive learning process* or on the contrary, by using *knowledge-intensive* measures defined by a domain expert. Nevertheless, the designer of the system must evaluate the pros and cons of each approach.

Right now, the focus has been on similarity measures that aim at comparing the problem part of cases. Notwithstanding, there exist other strategies, as the one presented by Bergmann et al. (2001) who argues that not only the most similar cases to the target case should be retrieved, but the usefully similar. Following the same line Smyth and Keane

---

5. k-nearest neighbour (Cover and Hart, 1967).
Albert Vilamala (1994, 1995a, 1996, 1998) explain that sometimes the most similar cases are also the most difficult to adapt. Therefore, the retrieval should pay attention to this fact and retrieve those cases similar enough and at the same time easy to adapt. Leake et al. (1997) propose an approach to this problem with a system that uses CBR to predict adaptation effort.

3.2.5 Reuse

Reuse is the stage where previous experience of solving a problem is adapted and applied to the problem at hand. In some scenarios such as classification, there is no need to adapt the solution but it is directly applied. In some others, though, adaptation is a must. The methods to commit this purpose can be split into three main categories: substitution adaptation, where some parts of the retrieved solution are reinstantiated; transformation adaptation, which alters the structure of the solution (Kolodner, 1993); and generative adaptation, that replays the method of deriving the retrieved solution on the new problem.

CHEF (Hammond, 1990), a menu-planning system, is a clear example that applies both substitution and transformation adaptation in its reuse phase. The former is used to substitute ingredients in the retrieved recipe to match the ingredients required in the target menu, whilst the latter is used to add or remove steps in the proposed recipe due to the ingredient substitution.

Gómez de Silva Garza and Maher (2000) proposed to use evolutionary methods for the adaptation phase in an architectural design system. According to their approach, the retrieved designs are the initial population for a genetic algorithm. Mutation is a substitution adaptation that randomly alters part of the design and crossover is a transformation adaptation that alters the structure of a design.

Generative adaptation has been applied to Prodigy/Analogy (Veloso and Carbonell, 1994), a general purpose planning system. In this system, when the retrieved solution presents a faulty element, the system recalls how the element was computed and replays a calculation for the new problem.

3.2.6 Revise

The phase where the solution proposed by the CBR system is evaluated is called the revise process. It is usually conducted by an external agent, because the purpose of this process is not only determining the correctness of the proposed solution but finding a correct solution in case the system has obtained an incorrect solution. It is important to notice that, by doing this, the system is able to acquire new knowledge which might be useful in the future.

In most cases, revise is manually done by a human expert on the domain. In other scenarios, a simulator in a virtual environment is in charge of evaluating the solution and providing an alternatively correct one, if needed. The last option is to directly apply the proposed solution to the real world and check its consequences.

3.2.7 Retain

This is the last process in the CBR classic cycle, where the knowledge gained from the last treated episode is incorporated to the system. While some systems straightforwardly keep the problem specification and the solution applied, others retain a more in depth detailed information regarding the last target case, such as how well the solution suited the system’s
goal (Goel et al., 1991) or how the solution was derived from the problem specification (e.g., Veloso and Carbonell (1994)).

It is important to realise that, even if at first sight, storing as many cases as possible seems a good approach — the more cases stored, the more the coverage and hence, the less the cost of adaptation — it might not be the case, since as the case base grows, the retrieval time also increases. Therefore, there is a need for a good trade-off between these two.

Specifically, when growing the case base, new cases are more probable to overlap with the existing ones offering very little improvement on adaptation but significantly increasing the retrieval cost. From then on, the performance of the system starts to degrade.

Different approaches include selecting the proper cases to add into the case base by eliminating directly redundant cases, eliminating noisy cases or even using a metric to assess the cost of adding a case compared to its expected savings (Minton, 1990). Even more, other authors propose to properly select some cases as representative of a particular subspace (Smyth and Keane, 1995b).

To conclude this section, notice the importance of maintenance in CBR systems. Along the system lifetime many different sources might degrade its performance. It may be due to the fact that indexes are out of date and ought to be updated (Muñoz-Avila, 2001; Craw et al., 2001); some cases that were relevant in the past are no longer needed; etc. Consequently, a maintenance policy is very important for the success of the system. Maintenance polices may be triggered periodically (e.g., at every case addition), conditionally (e.g., when retrieval time increases to a pre-specified threshold) or ad-hoc (e.g., by a random human maintainer). Reinartz et al. (2001) propose to add two more phases in the classical CBR cycle named review — to monitor the quality of the system knowledge — and restore — to select and apply maintenance operations.

4. Approach

Identifying thin layers is carried out by a two step approach: The former is detection, where the depth at which the thin layer is located has to be determined, and the latter is called characterisation, that aims at finding the algal group which the target thin layer belongs to as well as predicting its real thickness and concentration (Fig. 3).

![Figure 3: Thin layer identification approach.](image-url)
The first challenge we face when dealing with the detection step is the large dimensionality of the data to be treated. Therefore, we must select an algorithm that exploits the nature of the data efficiently, leading to detect the location of the main thin layers. Moreover, we already know that data supplied by sensors might suffer from highly noise levels. According to literature, Particle Swarm Optimization is a technique able to successfully meet all these requirements.

Regarding the characterisation phase, it can be split into two sub-steps. The first one is determining the algal group membership of the main alga present in the thin layer. Then, the second one is calculating the thin layer thickness and concentration. The first step is especially crucial for the correct behaviour of the system because algal type directly determines the meaning of the hyperspectral optical data. For instance, the same value of $K_d$ might mean a high level of algal concentration in a thin layer for a certain algal group, whereas it might mean a very low concentration level for another group.

Another important factor that must be taken into consideration for the design of this system is the lack of information about the domain at hand. For this reason, it is very important to build a flexible system capable of adapting to unseen phenomena. Case-Based Reasoning is a technique that perfectly tackles the mentioned challenges with the advantage that can incorporate these new phenomena by learning from its own experience.

In this section we present the data we have used for building the system and explain the different decisions that we have taken to apply the before-mentioned AI techniques to fit our specific domain.

![Figure 4: Radiative transfer equation.](image)
4.1 Data

Ideally, we would use passive sensors to measure the AOPs owned by the thin layers we want to detect. Nevertheless, to that end we can not use data from sensors given the fact that these sensors are not available for this project, yet. Therefore, working with real data is not feasible right now. In order to overcome this problem, we model our techniques upon synthetic data generated using the radiative transfer equation. The theory behind this equation states that it is possible to deduce AOPs from IOPs and vice-versa (Gordon et al., 1975) (Fig. 4).

To generate these synthetic data (Fig. 5), we have used the Hydrolight-Ecolight 5.0 radiative transfer model (Mobley and Sundman, 2008) which uses the radiative transfer equation to achieve its purpose. In our case, we have defined a set of scenarios composed by several thin layers at different depths, ranging in concentration and thickness and we have set specific IOPs depending on the compounds of these thin layers. Moreover, we have also specified standard environmental conditions such as wind speed or cloud coverage, to name few. This information has been used to feed the model which, in turn has used the radiative transfer equation to provide us with the AOPs associated to

Figure 5: Data for a two thin layers scenario. The plot shows the $K_d$ values at every sampled depth and wavelength. Data is perturbed by 30% noise.
our predefined thin layers. Such AOPs come in form of three-dimensional data, having the depth where the simulated AOP is measured in one dimension, the wavelengths along the visible spectrum in the other dimension and the actual AOP in the third dimension.

Notice that even if the model provides a set of different properties (e.g., irradiance, radiance, attenuation coefficient) we have used the downwelling irradiance \( E_d \) to calculate the downwelling attenuation coefficient \( K_d \) following the next equation:

\[
K_d(z, \lambda) = -\frac{\ln[E_d(z_2, \lambda)/E_d(z_1, \lambda)]}{z_2 - z_1}
\]

where \( z_1, z_2 \) are the previous and subsequent depths to the target \( z \).

The reason for using this approach has been purely empirical meaning that our system performed better with the \( K_d \) calculated by hand rather than using the values from the model.

4.2 Thin layer detection

In the first stage of the application (Fig. 6), mPSO is executed to discover irradiance peaks over the \( K_d \) landscape. As stated in 4.1, the inputs received are the irradiance coefficients for different depths along the visual spectrum. By applying the mPSO, we have been able to determine not only the depth (\( \delta \)) under the sea level where the thin

![Figure 6: The application workflow.](image-url)
layers are located —position containing the maximum phytoplankton concentration— but also the local maximum \( K_d \) value (\( \kappa \)) which will be used as an estimator of the thin layer concentration.

For determining the ‘real’ phytoplankton concentration from the \( \kappa \) estimator, knowing the algal type we are facing is of vital importance because there is a direct relation between algal type, \( \kappa \) and real concentration. After several failed attempts of using only \( \kappa \) as algal group discriminator, a good method to successfully determine the algal group has been to focus on the contour defined by \( K_d \) at \( \delta \) along the spectrum —spectral signature—, since all the members in an algal group share similar spectral signature, differing with other groups (Fig. 7(b)).

\[ (a) \ K_d \text{ value (X axis) at different depths (Y axis) for the best wavelength.} \quad (b) \ K_d \text{ value (Y axis) at different wavelength (X axis) for the best depth.} \]

Figure 7: Example of a vertical profile and spectral signature.

Notice that such signature is a series of values, meaning that managing various signatures to perform operations over them —such as comparing and contrasting in order to assess the membership of a given thin layer to a certain algal group— is tedious due to its large dimensionality. According to Agrawal et al. (1993), an efficient dimensionality reduction with minor information loss can be achieved by computing a Discrete Fourier Transform (DFT) to the contour, keeping just the first significant coefficients. The advantage of this technique is twofold: it aids at reducing the data’s dimensionality and it defines out of the box operators between signatures since those coefficients can be described as a point in an Euclidean space.

Particularly, in this application only the four first DFT coefficients have been sufficient to determine the contours. Therefore, a contour is expressed as a point (\( \rho \)) in an 8-dimensional space\(^6\).

---

\(^6\) DFT coefficients are complex numbers.
Additionally, the most suitable wavelength among the visible spectrum can be obtained from the mPSO output. This wavelength is especially important because it allows to track the vertical profile of the water column (Fig. 7(a)). Such profiles clearly represent the matter assemblages as Gaussian curves.

The morphology of these curves are used to infer an estimator ($\omega$) for the thin layer’s real thickness ($\theta$). $\omega$ corresponds to the full width at half maximum (FWHM) of the Gaussian curve defined by the values around $\delta$; being FWHM the distance between the two points whose $K_d$ value is half the maximum $K_d$ value in $\delta$ (Fig. 8).

Figure 8: Depth, concentration and thickness estimators.

4.3 Thin layer characterisation

To that end, we have got $\delta$ as the ‘real’ depth where the thin layer is located, $\rho$ as the group indicator and $\kappa$ and $\omega$ as the concentration and thickness estimators respectively. What is left is the ‘real’ thin layer’s algal group ($\alpha$), the ‘real’ thin layer’s concentration ($\gamma$) and the ‘real’ thin layer’s thickness ($\theta$). Thus, $\alpha$, $\gamma$ and $\theta$ will be computed by feeding the CBR module with $\rho$, $\kappa$ and $\omega$.

Taking into consideration the module’s goal explained above, a modified version of the classical CBR has been implemented in this project. As a reminder, the most important concept in CBR systems is the case definition. In this particular domain, a case has been defined as follows:

\[
\text{Case} = < P, S >
\]

\[
P = < \rho, \kappa, \omega >
\]

\[
S = < \alpha, \gamma, \theta >
\]

where,

- $\rho$ are the first DFT coefficients;
- $\kappa$ is the maximum attenuation coefficient;
- $\omega$ is the full width at half maximum (FWHM);
\(\alpha\) is the algal group; \\
\(\gamma\) is the concentration; and \\
\(\theta\) is the thickness.

Taking into account this case definition, the case base needs to be built. For this purpose, the experience accumulated through time is retained in form of cases that are being added iteratively into the case base.

Then, whenever a new problem arrives to the module, a proper associated solution must be obtained. Notice that in this context we have to achieve two different tasks which have to be evaluated in a specific order. First, we have to determine \(\alpha\) and secondly, from this information, we can predict \(\gamma\) and \(\theta\). These two tasks will be achieved in the retrieve and reuse processes respectively.

In the retrieve process, the system focuses on the \(\rho\) part of the problem and applies a k-Nearest Neighbour approach over the case base by performing an Euclidean distance, retrieving the most similar cases.

Next a majority voting on the algal groups (\(\alpha\)) from the retrieved cases is performed in order to decide the appropriate \(\alpha\) for the current case, which is kept as part of the solution.

Furthermore, \(\alpha\) is given to the reuse process in order to calculate \(\gamma\) and \(\theta\). More precisely, \(\alpha\) and \(\kappa\) are used to run the concentration model which predicts \(\gamma\). Similarly, by using \(\alpha\) and \(\omega\), the thickness model can be applied obtaining a prediction of \(\theta\). Right now, the three components of the solution have been already computed and are embedded in the final solution of the CBR.

Notice that the general form of CBR specifies a revise and a retain process which can be either automatic or manual. In the current approach, we have decided that a marine expert should manually evaluate the solution proposed by the system and decide if it is useful enough to be stored in the system. The reasons for the unfeasibility of automating this task are the lack of knowledge on the domain and the uncertainty in front of real scenarios.

The models used for \(\gamma\) and \(\theta\) predictions have been built by performing several linear regressions over the entire case base. Regarding concentration, a linear regression for each one of the owned algal groups has been done (Fig. 9(a)). This same approach has been applied for modelling thickness behaviour (Fig. 9(b)).

Finally, coupling the PSO and CBR outputs, the actual output of the system made of a thin layer’s classification into one of the possible algal groups (\(\alpha\)), a prediction of the real thin layer’s location (\(\delta\)), concentration (\(\gamma\)) and thickness (\(\theta\)) is given to the user, accomplishing the proposed application’s goal (Fig. 6).

5. Experiments

This section reports the experiments that have been carried out in order to proof the system performance. We have tested not only the suitability of the methods but also robustness, accuracy and precision. That is, we have performed three different sets of tests, each one of them aiming at validating a certain module along the application workflow. In that sense, we have designed tests to proof the detection of thin layers, which clearly correspond to validate the applied mPSO; we have also designed experiments for assessing the algal group membership identification and for testing the concentration and thickness prediction. These
two last sets of experiments have been performed to evaluate the thin layers characterisation by means of CBR.

5.1 Benchmark

For testing our system, we have used a controlled environment made of 300 water column scenarios generated by experts using the Hydrolight-Ecolight 5.0 (Mobley and Sundman, 2008) radiative transfer model. Each one of these scenarios contains a set of data that correspond to the simulation of a hyperspectral optical sensor measuring the attenuation coefficient ($K_d$) from the surface level to 15m deep. The measures are taken every 5cm. Moreover, due to the hyperspectral dimension, each measure is composed by a set of frequencies —between 400nm and 650nm— with a spectral resolution of 1nm.

Regarding the water composition, we have designed the scenarios for owning two thin layers per water column. This means that we have got up to 600 thin layers, each one ranging in depth, concentration, thickness and algal group membership. Thin layers are located between 4m and 12m deep, with a concentration ranging from 4mg/cm$^3$ to 20mg/cm$^3$ and their thickness is included in the interval that range from 0.4m to 1.4m. Furthermore, every thin layer has been designed to be of one of the four most usual algal types.

The last important fact that must be taken into account when defining the benchmark considering the present data is the gradually application of random white noise levels up to 50% in order to test robustness (Table 1).
<table>
<thead>
<tr>
<th></th>
<th>Depth</th>
<th>Concentration</th>
<th>Thickness</th>
<th>Spectrum</th>
<th>Noise</th>
</tr>
</thead>
<tbody>
<tr>
<td>Min.</td>
<td>4 m</td>
<td>4 mg/cm³</td>
<td>40 cm</td>
<td>400 nm</td>
<td>0%</td>
</tr>
<tr>
<td>Max.</td>
<td>12 m</td>
<td>20 mg/cm³</td>
<td>140 cm</td>
<td>650 nm</td>
<td>50%</td>
</tr>
</tbody>
</table>

Table 1: Composition of the 600 thin layers used for testing the system. Each thin layer belongs to A, B, C or D algal group.

5.2 Thin layer detection

5.2.1 Detection set up

The first parameters that have to be set when launching an mPSO algorithm are the ones regarding the number of swarms and particles. In this case, we used four swarms containing ten particles each. The reason for using ten particles has been purely empirical. However, it is a plausible choice since we need a good trade-off between having enough particles to take advantage of the cognitive part of the module and not overloading the system which would lead it to achieve poor performance. The decision of using only four swarms, apart from being the quantity which report better results, is because the data we are dealing with contain two clear parallel ridges along the spectrum, each one at depth where a thin layer is present. Hence, we have got two swarms approaching the main peak in each ridge, another one searching for new unknown peaks (e.g., they might randomly appear even in a controlled environment) and the last that has been left to allow some degree of failure.

Next, it is important to set a proper exclusion radius parameter because it will be useful for avoiding two swarms approaching the same local maximum. According to Blackwell and Branke (2006), the optimal exclusion parameter must be set as:

\[ r_{\text{excl}} = 0.5d_{\text{boa}} \]

where, \( d_{\text{boa}} \) is the linear diameter of the basin of attraction of any peak.

In our case, though, there is no such basin of attraction and in turn, no linear diameter for the basis. The cause for this statement is the configuration of the data retrieved from the sensors. As said, there is a ridge in each depth where the core of the thin layer is located, lasting along the visual spectrum. Therefore, the need to differentiate between the two dimensions (i.e., wavelength and depth) is a must. Having said that and coming back to the exclusion radius parameter, it is obvious that we need to set a \( r_{\text{excl}} \) to keep other swarms away in the depth dimension \( -excl_d- \) and another one in the spectral dimension \( -excl_s- \) (Fig. 10).

Regarding depth dimension, the diameter of the basin formed by the thin layers of our data set range between 2 m and 8 m averaging 4 m, so the \( excl_d \) parameter has been given the value of 2.

In the spectral dimension, there is no such basin, but a ridge along it instead. Then, we have considered that another swarm should not search in the same depth, at any wavelength, unless the current swarm is searching near the limits of the search space. Thus, the value of the \( excl_s \) has been set to 150.
Figure 10: Exclusion distance on depth and spectrum dimensions.

Similarly, a parameter to determine when the particles of a swarm have converged needs to be set depending on the domain. In this project we have split the parameter into two to have deeper control over the convergence as it has been done for the exclusion radius. Here, though, after performing the appropriate tuning, both convergence parameters have been given the value 2. Despite this coincidence, we think that it is important to differentiate these parameters because they might be useful for further studies when using other data sources. In other words, such coincidence has been purely chance.

The last parameters worth to mention for setting the mPSO benchmark are those used to calculate the velocity of each particle. Reminding Equation 1, the $\chi$ and $\phi$ are crucial to obtain proper velocities. Standard PSO values have been given to those parameters according to Blackwell (2007). Such values are $\chi = 0.729843788$ and $\phi = 2.05$.

The set of tests consists in launching the mPSO algorithm 20 times for each scenario and average the results. Besides, different levels of random white noise have been applied to the raw data to test robustness. Specifically, we have applied noise levels of 10%, 20%, 30%, 40% and 50% performing a detailed study for each one of them. The purpose of applying such high levels of noise is because we have to develop a very robust system that will be able to work in real scenarios, still unknown, in a near future.

5.2.2 Detection results

One of the strong points of using mPSO to approach this kind of problems is its capacity to access just few positions within the search space, yet achieving quite high performance. In our case, the algorithm has accessed 13% of the existing 75,000 measurements.

The first issue that worried us about detecting thin layers was the possibility that the system could miss a thin layer when both ones present in a water column were too close to each other. Intuitively, this idea seems pretty plausible, since it is hard even for humans to determine when a thin layer finishes and the other one starts when we are facing two too close thin layers. From the available data, however, this was not an issue. The scenarios that present closest thin layers were those containing two layers 3.5m separated from their core. These situations have been demonstrated to be innocuous regarding the performance of our system.

Contrarily, the problems arouse for certain situations where a thin layer was not detected. This happened exclusively when the upper thin layer in the water column contained

---

7. A scenario consists of a water column profile.
a very high level of concentration, whilst the bottom one presented very low concentration. In such scenarios, the bottom thin layer was not always detected.

<table>
<thead>
<tr>
<th>Noise</th>
<th>0%</th>
<th>10%</th>
<th>20%</th>
<th>30%</th>
<th>40%</th>
<th>50%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Failures</td>
<td>1.38</td>
<td>1.62</td>
<td>1.64</td>
<td>1.67</td>
<td>1.75</td>
<td>1.82</td>
</tr>
<tr>
<td>Error</td>
<td>2.8</td>
<td>3.3</td>
<td>3.4</td>
<td>3.9</td>
<td>4.1</td>
<td>4.2</td>
</tr>
<tr>
<td>Std.Dev.</td>
<td>3.0</td>
<td>6.0</td>
<td>7.4</td>
<td>8.3</td>
<td>9.8</td>
<td>10.8</td>
</tr>
</tbody>
</table>

Table 2: Peak detection failures(%), error(cm), and standard deviation(cm), given noise levels ranging from 0% to 50%.

Due to the fact explained above, our tests focused on first calculating the failure rate—both false positives and false negatives—over different levels of noise. Table 2 shows this rate which goes from 1.38% in a no-noisy data to 1.82% of failures on data affected by 50% of noise. Clearly, even if this failure on detecting certain thin layers exists, it rarely occurs according to our tests.

For those thin layers that have been properly detected, we need to measure the accuracy on locating such thin layers. In this case, Table 2 also shows the good response obtained by the system, since the error rate ranges from 2.8\( \text{cm} \) when no noise is found to the system up to only 4.2\( \text{cm} \) when the maximum level of noise in our experiments has been applied. Notice here, that even for the highest error level, it is smaller than the depth resolution—5\( \text{cm} \). This means that the real error rate of the system might be even less if we use data with a higher resolution. Giving these low error rates, we can infer that the depth where the thin layer is located does not affect the module final result.

Even if the average error rate for the 20 runs gives good results, it is interesting to calculate its standard deviation in order to assure the real response of the system and that we are not being fooled by the results. In that sense, notice the low deviation—from 3.0\( \text{cm} \) to 10.8\( \text{cm} \) depending on noise—which leads us to trust the accuracy just shown and claim the robustness of the thin layers detection module.

5.3 Thin layer characterisation

5.3.1 Determining algal group membership

In order to assess the performance of the CBR module, we have started by focusing on the algal group identification part. Specifically, the DFT coefficients retrieved by the mPSO module corresponding to the contour defined by \( K_d \) along the spectrum for a thin layer’s depth (\( \rho \)) have been used as an n-dimensional point in order to determine the main algal group (\( \alpha \))—either A, B, C or D. A 10-fold cross validation technique has been applied on every pair of values consisting in a DFT point and the algal group.

Several tests for tuning the parameters regarding the k-Nearest Neighbour approach have been performed. We have used different values of k (i.e., 1, 3 and 5) obtaining the best results when k=1, where its accuracy is over 97% for noise levels under 30%. From that point, the system accuracy decreases steadily until achieving 95% of accuracy for noise
levels up to 50%. For all other values of \( k \), the accuracy is lower, but always reaching more than 90% accuracy, as it is in the worst case. Such results are shown in Figure 11.

![Figure 11: Accuracy in algal group identification when applying noise levels up to 50%.

At the same time, different voting methods such as majority voting (Equation 2) and weighted voting (Equation 3), which takes into account the distances between target point and their neighbours, have also been conducted. In spite of the different methodologies, the results obtained have been exactly the same. Therefore, we have decided to use the majority voting due to its lower computational cost.

\[
\hat{f}(x_q) \leftarrow \arg\max_{v \in V} \sum_{i=1}^{k} \mu(v, f(x_i))
\]  

(2)

\[
\hat{f}(x_q) \leftarrow \arg\max_{v \in V} \sum_{i=1}^{k} \frac{1}{\|x_q - x_i\|} \mu(v, f(x_i))
\]  

(3)

where,

\( f(x) \) is the membership function;
\( V \) is the set of algal groups; and
\( \mu(a, b) = 1 \) if \( a = b \); 0 otherwise.
5.3.2 Concentration and thickness set up

The second block of the thin layers characterisation aims at evaluating the prediction of real concentration ($\gamma$) and thickness ($\theta$) in the CBR module, which corresponds to the last phase of the system.

It has been stated that the prediction of $\gamma$ and $\theta$ is the result of applying linear regression on the concentration and thickness estimators, which are $\kappa$ and $\omega$ respectively. Hence, the importance of calculating proper estimators is a must for ensuring precise and accurate predictions. This condition still holds whenever more noise is applied to raw data. Because we want to build a robust system able to deal with data highly influenced by noise, we had to apply a smoothing technique to remove the noise on the data and extract reliable $\kappa$ and $\omega$ estimators.

The smoothing technique used on the system consists in calculating the mean value of every measure in its neighbourhood along the depth axis by defining a smoothing grade which is the number of neighbouring measures immediately before and after the current measure in order to soften the existing peaks derived by the noise.

Another important parameter regarding the estimators worth it to comment is the sliding window. This parameter specifies the interval around a measure where to search for the highest peak. It is especially important for noisy environments since it only focuses on the highest peak within the window, ignoring other lower peaks that might arise from the noise.

Good $\kappa$ and $\omega$ estimators in noisy data are obtained by joining the smoothing and sliding window techniques. Nevertheless, tuning those parameters is a compulsory task for achieving good system performance. The results shown in Figure 12 have been achieved by applying different values for the smoothing and sliding window parameters to tweak first concentration and then thickness on all the scenarios for different levels of noise.

![Figure 12: Accuracy on smoothing parameter tuning for data perturbed by different levels of noise.](a) Concentration tuning. (b) Thickness tuning.]

Notice that the best performance for concentration has been reached setting the smoothing parameter to 6 (Fig. 12(a)) when dealing with highly noisy data — noise levels greater than 40% — and using 5 as the value for the parameter when facing data perturbed by low
noise levels — under 30%. In all those cases, though, either of the previous values for the parameter can be used since the differences in accuracy are very low.

Regarding thickness, Figure 12(b) shows the need for using very different parameters depending on the noise affecting our data. Whilst data affected by low noise levels — up to 10% — perform well using a smoothing parameter equals to 6, those samples perturbed by 20% reach their best performance setting the parameter to 22. Withal, data affected by more than 30% of noise need to use 20 as the value for the smoothing parameter. Otherwise the accuracy drops rapidly. From these statements we can generalise that 6 is a good value for low-noisy data and around 20 is good for high-noisy data. Nevertheless, data-specific parameter tuning should be done in each case to ensure good performance.

The most suitable value for the sliding window parameter has been found to be 1. Even if the implementation of this technique is compulsory for the smoothing to work, tuning its parameter is not as critical as the smoothing one. Hence, we have used a sliding window big enough to avoid finding two different peaks as being the maximum for a single thin layer due to noise variations or sampling errors and at the same time small enough to focus only on one thin layer at a time. After several trials, sliding window equals to 1 have given the best results.

5.3.3 Concentration and thickness results

Apart from those tests for tweaking the parameters, other experiments to evaluate the suitability of the methods used to predict $\gamma$ and $\omega$ have also been conducted. Figure 13 exposes the accuracy variability on concentration and thickness at different noise levels for the best parameters possible in each case. The high values achieved demonstrate the suitability of using linear regression techniques for predicting $\gamma$ and $\omega$. Nonetheless, we cannot ignore the fact that B type algae are the ones that perform worst in average, so further study must be carried out.

Focusing on Figure 13(b) B type algae accuracy, the thickness prediction technique reaches its minimum — slightly under 70% accuracy — for noise levels of 10%. The reason for this is that the smoothing parameter has to be set to 6 for low noise levels yet must be set to 22 for high noise levels. In this case, noise levels of 10% seem to be the threshold between low and high noise perturbation, together with the B type algae problematic issues lead the system to such lower performance.

Notice that these studies have evidenced the weaker performance of $\omega$ compared to $\kappa$. Those differences on accuracy rely on the nature of each estimator. In other words, noise perturbations on a Gaussian-like function affect more significantly the FWHM rather than its maximum value.

<table>
<thead>
<tr>
<th>Algal Types</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>Avg.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Concentration</td>
<td>0.95</td>
<td>0.88</td>
<td>0.95</td>
<td>0.93</td>
<td>0.93</td>
</tr>
<tr>
<td>Thickness</td>
<td>0.81</td>
<td>0.74</td>
<td>0.82</td>
<td>0.82</td>
<td>0.80</td>
</tr>
</tbody>
</table>

Table 3: Concentration and thickness accuracy on the four algal types for data perturbed by 50% of noise.
Detection and Identification of Phytoplankton Assemblages using Case-Based Reasoning

Finally, other tests to emphasise the differences among algal types regarding concentration and thickness prediction have been carried out. As can be seen in Table 3, algal type B is the one that achieves lowest results. Despite reaching 88% accuracy for concentration and 74% for thickness when 50% of white noise has been applied, this value is lower than its homologous A, C and D types. We think that the cause for such phenomenon is the fact that B-type algae show lower $K_{d}$ values for the same values of $\gamma$ than A, C or D types. Therefore, because the multiplication factor of the linear regression on B-type algae is greater, the errors derived from smoothing are also magnified leading to poorer accuracy on $\theta$.

6. Conclusions and future work

In this master thesis we have introduced an ongoing research topic consisting in detecting and characterising phytoplanktonic thin layers from data supplied by hyperspectral optical sensors. The task has been to determine not only the location, but also concentration, thickness and type membership of thin layers.

The final purpose of this study is to embed the developed techniques into an oceano-graphic probe able to retrieve biologically interesting water samples in real marine scenarios. Specifically, what is presented here corresponds to the project’s first phase.

Two powerful (and yet very different) Artificial Intelligence techniques have been reviewed and applied to meet the project goals. In particular, Particle Swarm Optimization has been run for locating thin layers whilst Case-Based Reasoning has been used to characterise them.

Despite using synthetically generated data for implementing and evaluating the methods, they featured great dimensionality and high noise perturbation in an attempt to build a robust system able to deal with the unpredictable real data. These two facts have been crucial in selecting the two AI techniques we have employed.

Furthermore, Discrete Fourier Transform has been applied to drastically reduce the data dimensionality and a smoothing technique has been required to soften the consequences of noise.
Concentration and thickness have been predicted by means of a linear regression model. The methods’ suitability and robustness have been assessed by performing different tests focusing on the thin layer detection performance, algal group classification and concentration and thickness prediction.

The methodology presented in this work should be applied whenever real data from sensors is feasibly available. However, further studies are needed to face this new scenario.

Another relevant research line, which might be found in nature, is a mixture of several algal types in a single thin layer. Hence, this could be an interesting approach that remains future work.

Finally, future research will be done in the second phase of the project where the information retrieved by the techniques presented will be used to wisely determine the best locations where the samples should be gathered.

Acknowledgments

The author wants to thank his supervisors Enric Plaza and Josep Lluis Arcos, the marine biologists and probe engineers involved in the project. This work was partially funded by projects ANERIS (CSIC-PIF08-15-2), Next-CBR (TIN2009-13692-C03-01), and by the Generalitat de Catalunya under the grant 2009-SGR-1434.

References


31


