Alma Mater Studiorum - Università di Bologna

Dottorato di ricerca in

Fisica

Ciclo XX

Settore scientifico disciplinare di afferenza: FIS/04

PATTERN RECOGNITION ANALYSIS ON HEAVY ION REACTION DATA

Presentata da: Dott. Jacopo De Sanctis

Coordinatore Dottorato
Prof. Fabio Ortolani

Relatore
Prof. Mauro Bruno

Esame finale anno 2008

Contents

Introduction

In order to study the mechanisms of heavy ion reactions the main goal is to classify collisions according to some global features; allowing for the identification and the characterization of the source(s). The topology of each collision is, to a large extent, related both to the impact parameter of the collision and to a partial or total transformation of the available centre-of-mass kinetic energy of the relative motion between the two partners of the collisions into disordered motion (heat). This dissipation process is governed by several important ingredients. One of them is the relative velocity between the initial partners of the reaction v_{AA}. The corresponding reduced relative wavelength associated with a nucleon-nucleon collision then reads

$$\frac{\lambda}{2\pi} = \frac{h}{2\pi m v_{AA}}$$

where m is the nucleon mass. According to equation 2, the following values (in the case of symmetrical systems) of $\lambda/2\pi = 6.5, 2.1, 0.67, 0.24$ fm are obtained for 1, 10, 100, 1000 AMeV beam energies, respectively. These values have to be compared with the mean nucleon-nucleon distance in a nucleus (typically 2 fm). If $\lambda/2\pi$ exceeds this distance, a collective behaviour of nucleons during the collision is expected. In other words, mean field (one-body) effects overcome nucleon-nucleon collisions (two body) effects. The situation will be reversed if $\lambda/2\pi$ is smaller than the mean nucleon-nucleon distance. According to this criterion, it turns out that mean field effects are expected to be dominant in the low-energy region (below 15 AMeV).

In general, in the low-energy part the main de-excitation patterns are evaporation [1, 2, 3, 4], decay by giant resonances [5, 6, 7, 8, 9], the production of evaporation residues and fission. The evaporation process is associated with the "chaotic" thermal motion of the nucleons at the nuclear surface. The correlation between thermal energy and temperature provides information on the heat capacity of hot nuclei. The evolution of the so-called level density parameter $a = A/K$ (where A being the mass number and K is a parameter which for low energy has an empirical value of about 8 MeV) reflects the various transitions occurring inside the nuclei as the temperature increases. Small collective motions such as giant resonances can be studied at finite temperature and their characteristic decay properties have been established as a function of excitation energy. In contrast, fission is a large amplitude collective motion. The evolution of its probability with excitation energy reveals the typical times needed to strongly deform a nucleus. It depends on the corresponding nuclear matter viscosity and its dependence on temperature. At low bombarding energy, i.e. when the nucleus-nucleus relative velocity v_{AA} is less than the Fermi velocity $v_F(\approx 0.3c)$, nucleon-nucleon collisions are strongly inhibited by the fact that few final states are available in the exit channel. In this case, energy dissipation occurs mainly because the nucleons of the projectile are retained inside the target (and vice versa) by the corresponding attractive potential, giving rise to one-body dissipation. An increase in the incident energy induces the opening of the available phase space for outgoing nucleons, which leads to some decrease in Pauli blocking effects. In turn, in the limit of relativistic incident energies, dissipation occurs mainly through nucleon-nucleon collisions (two body dissipation).

Another important quantity is the available energy per nucleon, i.e. the maximum excitation energy which can be brought into the system. An incident energy of 30 AMeV, for example, corresponds to an available energy close to the total binding energy for symmetrical nucleus-nucleus

collisions. In other words, the low incident energy domain (below 15 AMeV) will correspond to moderate excitation of final products but much higher excitation can be reached at intermediate or large incident energy. Finally, as already stated, the values of v_{AA} are smaller than the velocity associated with pion production threshold in nucleon-nucleon collisions, which corresponds in vacuum to an energy of 290 AMeV. This confirms again that, in the intermediate energy domain, nucleon excited states will have a negligible influence.

In the low energy domain ($E < 15$ AMeV) the reaction mechanism is mainly governed by the long range of nuclear force, with prevailing deep inelastic collisions and binary reactions. In this region of energies (Coulomb barrier region and below) the reaction mechanisms may be classified according to the impact parameter b or the orbital angular momentum l. Elastic scattering is observed for impact parameters exceeding $b_{max} = R_t + R_p$ values (or the corresponding angular momentum l_{max} which corresponds to grazing collisions. Slightly below b_{max}, quasi-elastic and transfer reactions are observed in which the projectile and target kinematical properties are only slightly perturbed. They mainly reflect the external orbit properties of the interacting nuclei. Dissipative collisions, also called deep inelastic collisions (DIC), and possibly fusion, are observed for more central collisions. They are clear signatures of mean-field effects leading to a collective behaviour of the involved nuclei. In the DIC case, projectile and target nuclei are strongly slowed down due to nuclear matter friction. For a short time they form a "quasi-molecular" state before reseparation. During this step nuclei may exchange nucleons. The corresponding lifetime may be estimated from the rotation angle of the di-nuclear system before the decay. The fusion corresponds to the most central collisions. The angular momentum that separates the fusion and the DIC regions is called the critical angular momentum.

The physical scenario becomes very different with respect to the previous

one if we consider the heavy-ion collisions in the relativistic energy range (0.2 - 1 AGeV). At these energies the dissipation process is dominated by hadronic cascades because the wavelength associated with nucleon-nucleon collisions is shorter than the nucleon size. The corresponding relative velocity between a nucleons of the projectile and target is also much larger than the Fermi velocity v_F. For these of the two reasons, collisions can be safely described by geometrical concepts leading to the so-called participant-spectator picture: nucleons which do not belong to the overlapping zone of the two incoming nuclei do not suffer hard nucleon-nucleon collisions and constitute the spectators while the other ones are the participants. The physics of the participants has been studied for a long time with the first generation of 4π detectors [10, 11, 12, 13, 14]. The corresponding deposited energies per nucleon are far beyond the nuclear binding energies.

Now, we focus on the Fermi energy domain, typically between 15 and 100 AMeV (the domain between low and relativistic energies). It is a transition region in which both one body and two-body behaviours are observed and strongly compete.

In the Fermi energy range, the energy dissipated in a nucleus cannot excite intrinsic states of the nucleon. Thus, the whole energy is available to heat the matter. However, part of the energy is also used to excite collective degrees of freedom associated with deformation, rotation and/or compression. The energy stored in a given mode depends on the typical timescale for the excitation of this mode and also on the initial conditions, i.e. the entrance channel characteristics. For instance, the amount of rotational motion reflects both the entrance orbital angular momentum and the shapes of the interacting nuclei. Indeed, the deformation of outgoing partners in binary dissipative reactions is related both to the geometry (impact parameter) and to nuclear matter viscosity as is already the case at lower energy (the outgoing fragments in a DIC are released as deformed objects).

The Fermi energy range appears to be a very interesting energy region in which pieces of nuclear matter are heat and compressed in a broad range of temperatures and densities. Consequently, a large variety of physical processes are expected to play a role. For example in peripheral and mid-central collisions binary collisions are present, where a quasi-projectile (QP) and a quasi-target (QT) are produced in the collision. If the decay chain is complicated or long, the final products are quite different from the original QP and QT nuclei. Conversely, if it is simple or short, the QP and QT residues still resemble the initial projectile and target nuclei and there is an extra particle emission in between the two sources: this is a first hint for the formation of intermediate structures like the neck-like ones. Indeed, in the overlap region of the two incoming nuclei, a highly dense and hot piece of nuclear matter is produced, which, in some circumstances, may become an independent third source of emission of smallest fragments. In the analysis of experimental data, a clear alignment of the three fragments is observed when one of them has a size of the order of the volume of the overlap zone (see [15]).

The physical scenario in the central collisions and in the Fermi energy domain it is quite important to be studied because from an experimental point of view it is possible to observe (at energy beam of few tens AMeV) the production of a relatively large number of fragments with intermediate mass. In this contest we can speak of multifragmentation phenomena ([16],[17]). The multifragmentation is one of the most important topics in nuclear physics . In this regime it is possible to study the behaviour of the nuclear matter under extreme conditions of temperature and density. The nuclei in their ground state behave like Fermi liquids, and at high excitation energy can be decomposed in a vapor of light particles and clusters (as the Van Der Waals equation of state). It is therefore interesting to study a possible phase transition in the nuclear bulk. The expected associated phase-transition constitutes indeed one of the most intriguing topics of nuclear physics with the study of the

equation of state of the nuclear matter at high density and temperature. Several signals of a phase change have been found, like a *plateau* in the caloric curve ([18],[19]), a sudden onset of multifragmentation and collective expansion, an increased probability for equal sized fragments possibly reminiscent of a spinodal decomposition ([20]), an abnormal enhancement of partial energy fluctuations tentatively associated to a negative branch of the heat capacity ([21]), a bimodal distribution of exclusive observables hinting phase coexistence ([22]), universal scaling laws both for the heaviest fragment and for the droplet distribution ([23],[24]).

In this work, we will focus on the new methods which can be used to perform a centrality sorting of the events. The knowledge of the impact parameter b, defined as the distance between the straight-line trajectories of the centers of the two nuclei before their interaction, would be important at least to discriminate central, semi-peripheral and peripheral collisions. Unfortunately from an experimental point of view it is impossible to measure directly the values of the impact parameter. The most common technique to estimate the values of b, based on one-dimensional analysis, is to find a correlation between physical observables and the impact parameter b.

Since 1990s, there were developed new approaches in the framework of the Neural Networks (NN) (see [25],[26],[27],[28]). In these works, for the first time, the Neural Networks have been used, alternatively to traditional methods, described briefly before, to backtrace the values of the impact parameter. The first step in this direction has been made using the NN by David *et al.* [27]. They have obtained, using simulated $^{197}Au + ^{197}Au$ events only, an improvement by a factor of 4 in the determination of the impact parameter compared to more conventional methods in central collisions. In fact, the NN allows one to lower the standard deviation between the known impact parameter of "theoretical data" and the impact parameter derived from the observables by a factor

of 4 as compared to the use of one observable only. In other words, in the NN analysis it is possible to use at the same time, more observables with respect to traditional methods. In the work of David *et al.* the observables that have been used are the total multiplicity of protons, the largest fragment observed in each collision and the energy ratio in the center of mass system defined by $\sum(p_t^2/2m)/\sum(p_z^2/2m)$. Where p_t and p_z are the impulse along the perpendicular axis of the beam and along the beam axis, respectively. The most recent work using the NN to determine the value of the impact parameter belongs to F. Haddad *et al.* [25], where, for the first time, they have applied the NN on the real experimental events for the reaction $^{40}Ca +^{40} Ca$ at 35 AMeV. Before the application on the experimental data they have trained and checked the NN on the events produced by two different type of nuclear reaction models: a dynamical transport model QMD [29] coupled with GEMINI [30] and an event generator based on a statistical model EUGENE [31]. The observables used in this analysis have been: the charged particle multiplicity, the perpendicular momentum and the energy ratio. The results obtained on the events generated by the two models show that, for all energies, the NN gives the lowest deviation between the values of the impact parameter of "theoretical data" (by the models) and the impact parameter derived from the observables. The NN always allows an improvement of around 25% compared to the others traditional methods. For the real experimental data the results obtained from David *et al.* tends to indicate that the NN can be a valuable tool in data analysis.

The results obtained in these works were encouraging and lead towards more specific analysis. For these reasons we wanted to continue in this way using a new and different approach based on a classifier known as Support Vector Machine (SVM). This method allows to backtrace the values of b through a particular multidimensional analysis in the large field of the pattern recognition techniques.

This analysis consists mainly of two different steps. Similarly to the

methods based on NN. In the first one, known as the *training phase*, SVM learns how to discriminate between peripheral, semi-peripheral and central reactions. These samples are taken from a large number of events generated for different values of b with dynamical models such as Classical Molecular Dynamics (CMD) and Heavy Ion Phase-Space Exploration (HIPSE). In the second step, known as the *test phase*, what has been learned is tested on new events generated by the same models. Our tests demonstrate that, following this approach, central, semi-peripheral and peripheral reactions can be correctly classified for about 85% of the cases.

Our goal is to discriminate objects that belong to different classes, by means of the knowledge of several observable characteristics of each event, i.e, features. Classes are represented by nuclear reactions arising from different values of the impact parameter b. A pattern is a pair of variables $\{\mathbf{x}, y\}$, where $\mathbf{x}$ is a collection of observations or features and y is the label that represents the class to which the event belongs.

In the first chapter we will describe the ideas that are the basis of the Pattern Recognition Analysis and of the SVM algorithms.

In the second step we will shortly the main indicate characteristics of the two physical models used and their differences.

In the third chapter we will give a description of the Principal Components Analysis (PCA) technique, which is the most powerful way to select different classes in the experimental data In the fourth and fifth chapter we will give the different classification results obtained with SVM classifier applied respectively on the CMD model and the HIPSE model data for the nuclear reaction: $^{58}Ni +^{48} Ca$ at 25AMev. Which has been measured by the NUCL-EX group with the CHIMERA apparatus at the INFN "Laboratori Nazionali del SUD" in Catania. After having checked the reliability of classification of SVM on the two models, we have compared the SVM classification results on CMD events with the results obtained by the one-dimensional analysis using the total charged

multiplicity and the PCA analysis.

In the last chapter we will show the main characteristics of the experimental device CHIMERA with a brief description of the procedures of control for the data acquisition. Finally, we apply the classification task on the experimental data obtained with CHIMERA in July 2003.

Chapter 1

Pattern Recognition and Support Vector Machine Algorithms

1.1 Basic concepts of Pattern Analysis

An overview of pattern classification will be given in this Chapter, with particular emphasis on a specific classifier - known as Support Vector Machine (SVM) - which will be used intensively in the rest of this work.

1.1.1 The act of learning: overview

In humans the act of learning is namely the process of gaining knowledge or skill in something by experience. Common and apparently simple human processes as recognizing a landscape, understanding spoken words, reading handwritten characters or identifying an object by touching it, they all belie the act of learning. In fact, the condition for a landscape to be recognized, spoken words to be understood, handwritten characters to be read and objects to be identified, is that the human brain has been previously trained in order to do that, namely it has *learnt* how to do that. This is why it is necessary to admire a landscape several times before recognizing it from a slightly different view, or to hear an unknown foreign word more than once before becoming familiar with it. From the examples discussed above, it is evident that the act of learning

plays a crucial role in all those processes requiring the solution of a pattern recognition task, thus all those processes in which the human brain is required to take an action based on the class of the data it has acquired. For example, hearing a voice and deciding whether it is a male or a female voice, reading a handwritten character and deciding whether it is an $\mathcal{A}$ or a $\mathcal{B}$, touching an object and guessing its temperature, those are typical pattern recognition problems. Notice that this kind of processes represents almost the totality of the processes a human being has to deal with. Finding them a solution has been crucial for humans to survive. For that reason, highly sophisticated neural and cognitive systems have been evolved for such tasks over the past tens of millions of years. The scheme used by the human brain to address pattern recognition tasks is based on two separate phases, namely a training phase and a test phase. In the training phase the human brain gets experienced by dealing with patterns taken from the same population, as landscapes, spoken words, handwritten characters. Then, in the test phase, it applies to patterns of the same population - but previously unseen - what it has learnt during the training phase. In this sense, admiring a known landscape several times - trying to identify its characteristics - represents the training phase, whereas recognizing it from a slightly different view represents the test phase.

As regards machines, the act of learning refers to artificial intelligences - for instance computer programs - which are able to recursively change their own internal structures in response to input patterns in such a manner that their performance in recognizing previously unseen patterns improves. In this context, machine learning is an area of artificial intelligence concerned with the development of techniques which allow machines to learn how to solve pattern recognition problems, whereas learning machines are automate which solve pattern recognition problems. In a similar way to what happens for the human brain, the solution of a pattern recognition problem initially involves the collection

of the data set of training patterns. The learning machine structure is then adapted so as to create a mapping from the input patterns to its output values, such that the latter approximate the attended values as closely as possible over the whole training patterns. The recognition performance of the trained learning machine is then evaluated on a data set of test patterns, namely patterns which were not part of the training data set, but which were taken from the same population.

The success of machine learning - since 1960s up to nowadays - is twofold. First, it is evident that implementing learning processes by using machines is fundamental in order to automatically address pattern recognition problems which - due to their complexity - are almost impossible for a human brain to solve. For example, challenging pattern recognition tasks as speech recognition, fingerprint identification, optical character recognition, DNA sequence identification, video surveillance - and much more - can be easily and automatically addressed by means of learning machines. Second, by trying to give answers and explanations to the numerous questions and doubts arising when implementing such automatic learning systems, a deeper understanding of the processes governing human learning is gained. In the fact, many techniques in machine learning derive from the efforts gone in order to make more precise the theories of human learning through computational models. At the same time, it seems likely also that the concepts being explored by researchers in machine learning may illuminate certain aspects of biological learning.

Before proceeding, it is well worth specifying in more detail the significance of pattern recognition problems from a more technical perspective, (see [32]). As already discussed, all those problems requiring a human or an artificial intelligence to take an action based on the data acquired, are formally defined as pattern recognition problems. That family of problems can be further divided into families of sub - problems. That most common and important ones are *pattern*

classification problems, *regression* problems and *time - series prediction* problems.

Pattern classification problems are those in which the learner is required to learn how to separate the input patterns into two or more classes. A typical pattern classification problem could require - for example - a human brain or a learning machine to separate into classes the handwritten $\mathcal{A}$s and $\mathcal{B}$s taken from a data set of handwritten characters. When the problem do not require to associate the class of membership to an input pattern, but rather to associate a continuous value, a regression problem is faced. A typical regression problem could require a human brain or a learning machine to associate an age to input patterns represented by pictures of human faces. Finally, time - series prediction problems, in which a learning machine is trained to predict the $(n+1)^{th}$ sample in a time series from the previous n sample, is a special case of a regression problem but which assumes that the underlying data generator is stationary, namely its statistical properties are time-independent. In this work, the whole attention will be concentrated on pattern classification, which is actually the most common type of pattern recognition problem.

One of the most important characteristic of learning machines is that they are not programmed by using some a priori knowledge on the probability structure of the data set considered, they are rather trained by being repeatedly shown large numbers of examples for the problem under consideration. In a sense, they learn directly from the data how to separate the different existing classes. This approach determines some important peculiarities of learning machines. First, they are particularly suited for complex classification problems whose solution is difficult to specify a priori. Second, after being trained, they are able to classify data previously not encountered. This is often referred to as the *generalization* ability of learning machines. Finally, since they learn directly from data, then the effective classification solution can be constructed far

more quickly than using traditional approaches entirely reliant on a deep knowledge and experience in the particular field to which data refer. In order to stress the importance of an approach purely based on learning from data - in particular when a dynamical model of what is happening behind the scenes does not exist or whenever the underlying dynamics is too complicated - let us mention one enlightening example borrowed from ([33]):

> "When a human writer decides to write a letter, for example the letter $\mathcal{A}$, the actual outcome is the result of a series of complicated processes which cannot be modeled comprehensively in their entirety. The intensity of the lines depends on chemical properties of ink and paper, their shape on the friction between pencil and paper, on the dynamics of the writer's joints and on motor programmes initiated in the brain, these in turn are based on what the writer has learnt at school. The chain could be continued ad infinitum."

It is evident that, in such a situation, it is nearly impossible to address a classification task which is required to separate different handwritten characters - such as for example $\mathcal{A}$s and $\mathcal{B}$s - by modeling the way in which they are written by hand. For this reason, an approach purely based on learning from data is probably the most appropriate solution.

Nevertheless, some approaches in which the probability structure underlying the classes of the data set is known perfectly - or at most its general form - do exist. For example, as described in [34], Bayesian decision theory is a fundamental statistical approach to the problem is posed in probabilistic terms and that all the relevant probability values are known. In particular, it is based on quantifying the trade offs between various classification decisions using the probability and the costs that accompany such decisions. Unfortunately, for the most part of the applications, the probabilistic structure of the problem is

unknown. At most, only some vague and general knowledge about the situation, together with a number of training data representative of the patterns to classify, do exist. The problem is then to find some way to use this information in order to design the classifier. One approach is to use the training patterns for estimating the unknown probabilities and probability densities and to use the resulting estimates as if they were the true values. Let us quote a further example borrowed from ([46]):

"Suppose, for example, that some temporal sequences of detailed observations of double star systems were given and that the problem is to predict whether, eventually, one of the stars will collapse into a black hole. Given a small set of observations of different double star systems, including target values indicating the eventual outcome, an approach purely based on learning from data would probably have difficulties extracting the desired dependency. A physicist, on the other hand, would infer the star masses from the spectra's periodicity and Doppler shifts, and use the theory of general relativity to predict the eventual fate of the stars."

In this example - differently from the previous one - modeling the stars collapsing into black holes is probably more straightforward, owing to the deep knowledge of those phenomena. For that reason, here it could be more appropriate and effective to address the classification task with a modeling approach rather than with an approach purely based on learning from data.

In this work of thesis, how we said in the beginning of this chapter, we will use for our analysis the new classifier that belongs to the class of pattern classification problems, known as Support Vector Machine (SVM), in which we need to discriminate between different classes of events in nucleus - nucleus collisions.

Such discrimination is done on the base of the knowledge of the value of impact parameter b for each event. As we will see more precisely in the

next chapters and as we have said in the introduction the our goal is to divide the nuclear events collisions in classes of centrality. The division of events in classes on homogeneity is very important in nuclear reactions analysis data, because the physical scenario is very different for example if the nuclear collisions belongs to the class of central events or the class of peripheral events.

The approach to machine learning has changed in the last years, since more attention is now focused on the alternatives of neural networks. Statistical learning theory is nowadays more popular and attractive for researchers than in the early 1960s. In addition, it now plays a more active role rather than covering only the theoretical and formal aspects of machine learning. In fact, after the completion of the general analysis of learning processes, the research in the area of the synthesis of optimal algorithms, which posses the highest level of generalization ability for any number of observations, was started. Thus, in the last decade, many ideas have appeared in the machine learning community deeply inspired by statistical learning theory. On the contrary to previous ideas of developing learning algorithms inspired by the biological learning process, the new ideas were inspired by attempts to minimize theoretical bounds on the error rate obtained as a results of formal analysis of the learning processes. These ideas - often contradicting the biological paradigm - result in algorithms having nice mathematical properties (such as uniqueness of the solution, simple method of treating a large number of examples, independence of dimensionality of the input space) and excellent performance. In particular, they outperform the state-of-the-art solutions obtained by the old methods.

SVM is one of the shining peaks among the many learning algorithms deeply inspired by statistical learning theory and appeared in the machine learning community in the last decades. Nowadays, the Support Vector Machine algorithm is used in many research fields to solve different pattern recognition problems, for example, medical imaging diagnosis

[35],[36],[37], volcanic eruption forecasting [38], and much more [39],[40]. The main goal of a SVM analysis is to obtain a meaningful separation of events in class of similarity [41]. This is obtained by following a protocol,

1. the most meaningful features of the events provided by a model are recognized;
2. a sub-sample of model events is divided in classes;
3. the classes produced by the steps 1 and 2 are analyzed by SVM, which "learns" the links among features and classes;
4. the events not used for the learning stage (2 and 3) are analyzed by SVM which will provide the best separation of these events in class of similarity according to what it was learned during the training stage.
5. Checks are made to estimate the accuracy of the analysis 4.

In the next paragraphs of this chapter we will do a more detailed mathematical description that is the basis of the pattern recognition and SVM algorithms.

1.2 Mathematical formalization of the Pattern Recognition concepts

Pattern analysis deals with the problem of (automatically) detecting and characterising relations in data. Most statistical and machine learning methods of pattern analysis assume that the data are in vectorial form and that the relations can be expressed as classification rules, regression functions, or cluster structures; these approaches often go under the general heading of "statistical pattern recognition" [43].
By patterns we understand any relations, regularities or structure inherent in some source of data. By detecting significant patterns in

the available data, a system can expect to make predictions about new data coming from the same source. In this sense the system has acquired generalisation power by "*learning*" something about the source generating the data.

1.2.1 Data

By data we mean the output of any observation, measurement or recording apparatus. This therefore includes images in digital format; vectors describing the state of a physical system; sequences of DNA; pieces of text; time series; records of commercial transactions, etc. By knowledge we mean something more abstract, at the level of relations between data and patterns within the data. Such knowledge enable us to make predictions about the source of data or draw inferences about the relationships inherent in the data. By exploiting the knowledge extracted from a sample of data, they are often capable of adapting themselves to infer a solution to such tasks. We will call this alternative approach to software design the *learning methodology*. It is also referred to as the *data driven* approach that gives rise to precise specifications of the required algorithms.

The range of problems that have been shown to be amenable to the learning methodology has grown very rapidly in recent years. Examples include text categorization; email filtering; gene detection; protein homology detection; web retrieval; image classification, etc. These tasks are very hard or in some cases impossible to solve using a standard approach, but have all been shown to be tractable with the learning methodology.

In general, the field of pattern analysis studies systems that use the learning methodology to research *patterns in data*. The patterns that are sought include many different types such as classification, regression, cluster analysis (sometimes referred together as *statistical pattern recognition*), feature extraction, grammatical inference and

parsing (sometimes referred to as *syntactical pattern recognition*).

1.2.2 Patterns

If we imagine a dataset containing thousands of observations of planetary positions in the solar system, for example daily records of the positions of each of the nine planets, it is obvious that the position of the planet on a given day is not independent of the position of the same planet in the previous days: it can actually be predicted rather accurately on the basis of the knowledge of these positions. The dataset therefore contains a certain amount of redundancy, information that can be reconstructed from part of the data, and hence is not strictly necessary. In such cases the dataset is called *redundant*: simple laws can be extracted from the data and used to reconstruct the position of each planet on each day. the rules that govern the position of the planets are known as Kepler's laws. Kepler's laws can be viewed as an early example of pattern analysis, or data-driven analysis. By assuming that the laws are invariant, they can be used to make predictions about the outcome of future observations. The laws correspond to regularities present in the planetary data and by inference therefore in the planetary motion itself. They state that the planets move in ellipses with the sun at one focus; that equal areas are swept in equal times by the line joining the planet to the sun; and that the period P and the average distance D from the sun are related by the equation $P^3 = D^2$ for each planet.

From Table 1.1 we can observe two potential proprieties of redundant datasets: on one hand they are *compressible* since we could construct the table from just one column with the help of Kepler's third law, while on the other hand they are *predictable* since we can, for example, infer from the law the distances of newly discovered planets once we have measured their period. The predictive power is a direct consequence of the presence of the possibly hidden relations in the data. These relations indeed enable us to predict and therefore manipulate new data more effectively.

Planets	D	P	D^2	P^3
Mercury	0.24	0.39	0.058	0.059
Venus	0.62	0.72	0.38	0.39
Earth	1.00	1.00	1.00	1.00
Mars	1.88	1.53	3.53	3.58
Jupiter	11.90	5.31	142.00	141.00
Saturn	29.30	9.55	870.00	871.00

Table 1.1: An example of a pattern in data: the quantity D^2/P^3 remains invariant for all the planets. This means that we could compress the data by simply listing one column or that we can predict one of the values for new previously unknown planets, as happened with the discovery of the outer planets.

Typically we anticipate predicting one feature as a function of the remaining features: for example the distance as a function of the period. To be able to do this, the relation must be invertible, so that the desired feature can be expressed as a function of the other values. Indeed we will seek relations that have such an explicit form whenever this is our intention. Other more general relations can also exist within data, and it detected and they can be exploited. For example, if we find a general relation that is expressed as an invariant function f that satisfies

$$f(\mathbf{x}) = 0 \tag{1.1}$$

where $\mathbf{x}$ is a data item, we can use it to identify novel or faulty data items for which the relation fails, i.e. is for which $f(\mathbf{x}) \neq 0$. In such case it is, however, harder to realise the potential for compressibility since it would require us to define a lower-dimensional coordinate system on the manifold defined by equation 1.1. Kepler's laws are accurate and hold for all planets of a given solar system. We refer to such relations as *exact*.
It is clear that we cannot hope to find an exact prediction in cases where there will be factors beyond those available to the system, which may be prove crucial. Learning systems have succeeded in finding some relations. We can specify the relation that holds for much of the data and then simply add a list of exceptions cases. Provided the description of the

relation is succinct and there are not too many exceptions, this will result in a reduction in the size of the dataset. Since the relation holds with a certain probability we will have a good chance that the prediction will be fulfilled. We will call relations that hold with a certain statistical probability.

Typically we may hope that the expected error in the prediction will be small, or that with high probability the true value will be within a certain margin of the prediction, but our search for patterns must necessarily seek an approximate relation. One could claim that Kepler's laws are approximate because they fail to take general relativity into account. In this case of interest to learning systems, however, the approximation will be much looser than those affecting Kepler's laws. Relations that involve some inaccuracy in the values accepted are known as *approximate*. For approximate relations we can still talk of prediction, though we must qualify the accuracy of the estimate and quite possibly probability with which it applies.

Patterns can be deterministic relations like Kepler's laws. As indicated above other relations are approximate or only hold with a certain probability. We are interested in situation where exact laws, especially the ones that can be described as simply as Keplero's, may not exist. For this reason we will understand a *pattern* to be any relation present in the data, whether it be exact, approximate or statistical.

We can also express the pattern described by Kepler's third law in this form

$$f(D, P) = D^2 - P^3 = 0.$$

Alternatively

$$g(D, P) = 2 \log D - 3 \log P = 0.$$

Similarly, if we have a function g that for each data item ($\mathbf{x}$,$\mathbf{y}$) predicts some output values $\mathbf{y}$ as a function of the input features $\mathbf{x}$, we can express

the pattern in the form

$$f(\mathbf{x},\mathbf{y}) = \mathcal{L}(g(\mathbf{x}), \mathbf{y}) = 0,$$

where $\mathcal{L} : Y \times Y \rightarrow R^{+}$ is so-called *loss function* that measures the disagreement between its two arguments giving 0 as output if and only if the two arguments are the same.

Definition 1.1 *A general exact pattern for a data source is a non-trivial function f that satisfies*

$$f(\boldsymbol{x}) = 0$$

for all the data, $\boldsymbol{x}$, that can arise from the source.

The definition only covers exact patterns. We first consider the relaxation required to cover the case of approximate patterns. Taking the example of a function g that predicts the values y as a function of the input features $\mathbf{x}$ for a data item $(\mathbf{x},\mathbf{y})$, if we cannot expect to obtain an exact equality between $g(\mathbf{x})$ and $\mathbf{y}$, we use the loss function $\mathcal{L}$ to measure the amount of mismatch. This can be done by allowing the function to output 0 when the two arguments are similar, but not necessarily identical, or by allowing the function f to output small, non zero positive values. We will adopt the second approach since when combined with probabilistic pattern it gives a distinct and useful notion of probabilistic matching.

Definition 1.2 *A general approximate pattern for a data source is a non-trivial function f that satisfies*

$$f(\boldsymbol{x}) \approx 0$$

for all the data, $\boldsymbol{x}$, that can arise from the source.

Now, we consider statistical patterns. In this case there is a probability distribution that generates the data. In many cases the individual data

items can be assumed to be generate independently and identically,a case often referred to as independently and identically distributed or (i.i.d.). We use the symbol E to denote the *expectation* of some quantity under distribution.

Definition 1.3 *A general statistical pattern for a data source generated i.i.d. according to a distribution $\mathcal{D}$ is a non-trivial non-negative function f that satisfies*

$$\mathsf{E}_{\mathcal{D}} f(\boldsymbol{x}) = \mathsf{E}_X f(\boldsymbol{x}) \approx 0.$$

If the distribution does not satisfy the i.i.d. requirement this is usually as result of dependencies between data items generated in sequence or because of slow changes in the underlying distribution.

1.2.3 Pattern analysis algorithms

Definition 1.4 *A Pattern analysis algorithm takes as input a finite set of examples from the source of data to be analysed. Its output is either an indication that no patterns were detectable in the data, or a positive pattern function f that the algorithm asserts satisfies*

$$\mathsf{E} f(\boldsymbol{x}) \approx 0,$$

where the expectation is with respect to the data generated by the source.

We refer to input data examples as the training instances, the training examples or the training data and to the pattern function f as the hypothesis returned by the algorithm. The value of the expectation is known as the generalisation error.

Identifying patterns in a finite set of data presents very different and distinctive challenges. We will identify key features that a pattern analysis algorithm will be required to exhibit before we will consider it to be effective.

Computational efficiency. Pattern analysis algorithms must be able to handle very large datasets. Hence, it is not sufficient for an algorithm to work well on small toy examples; we require that its performance should scale to large datasets. The study of the computational complexity or scalability of algorithms identifies *efficient algorithms* as those whose resource requirements scale polynomially with the size of the input. This means that we can bound the number of steps and memory that the algorithm requires as a polynomial function of the size of the dataset and other relevant parameters such as the number of features, accuracy required, etc.

Robustness. The second challenge that an effective pattern analysis algorithm must address is the fact that in real-life applications data is often corrupted by noise. By noise we mean that the values of the features for individual data items may be affected by measurement inaccuracies, for example through human error. This is closely related to the notion of approximate patterns discussed above, since even if the underlying relation is exact, once noise has been introduced it will necessarily become approximate and quite possibly statistical. It requires that the algorithms will be able to handle noisy data and identify approximate patterns. They should therefore tolerate a small amount of noise in the sense that it will not affect their output too much. An algorithm with this property it is called *robust*.

Statistical stability. The third property is perhaps the most fundamental, namely that the patterns the algorithm identifies really are genuine patterns of the data source and not just an accidental relation occurring in the finite training set. We can view this property as the *statistical* robustness of the output in the sense that if we rerun the algorithm on a new sample from the same source it should identify a similar pattern. Hence, the output of the algorithm should not be

sensitive to the particular dataset, just to the underlying source of the data. For this reason we will describe an algorithm with this property as *statistically stable*.

1.2.4 Common pattern analysis tasks

When discussing what constitutes a pattern in data, we draw attention to the fact that the aim of pattern analysis is to predict one feature of the data as a function of the vales of the other features. It is therefore to be expected that many pattern analysis tasks isolate one features that it is their intention to predict. Hence, the training data comes in the form

$$(\mathbf{x}, y),$$

where y is the value of the feature that the system aims to predict, and $\mathbf{x}$ is a vector containing the vales of the remaining features. The vector $\mathbf{x}$ is known as the *input*, while y is referred to as the *target output* or *label*.

Supervised tasks. The pattern analysis tasks that have this form are referred to as *supervised*, since each input has an associated label. For this type of task a pattern is sought in the form

$$f(\mathbf{x}, y) = \mathcal{L}(y, g(\mathbf{x})),$$

where g is referred to as the *prediction function* and $\mathcal{L}$ is known as a *loss function*. Since it measures the discrepancy between the output of the prediction function and the correct value y, we may expect the loss to be close to zero when a pattern is detected. When new data are presented the target output is not available and the pattern function is used to predict the value of y for the given input $\mathbf{x}$ using the function $g(\mathbf{x})$. The prediction of $f(\mathbf{x}, y) = 0$ implies a small the discrepancy between the values $g(\mathbf{x})$ and y.

Different supervised pattern analysis tasks are distinguished by the type of the feature y that we aim to predict. *Binary classification*, referring

to the case to the case where $y \in \{-1, 1\}$, is used to indicate that the input vector belongs to a chosen category ($y = +1$), or not ($y = -1$). In this case we use the so-called discrete loss function that give back 1 if its two arguments are different and 0 if they are equal. In this case the generalisation error is the probability that a random test is misclassified. If the training data are labeled as belonging to one of N classes and the system must learn to assign new data points to their class, then y is chosen from the set $\{1, 2, .., N\}$ and the task is referred to as *multiclass classification*. *Regression* refers to the case of supervised pattern analysis in which the unknown feature is real-valued, i.e. $y \in \mathcal{R}$.
Other types of pattern analysis are the **semisupervised tasks** and the **unsupervised tasks**. In this work we used only the supervised methods. Now we introduce some of the basic notation. We denote the input space by X and for supervised tasks use Y to denote the target output domain. The space X is often a subset of $\mathcal{R}^n$. Note that if X is a vector space, the input vectors are given as column vectors. If we wish to form a row vector for an instance $\mathbf{x}$, we can take the transpose $\mathbf{x}'$. For a supervised task the *training set* is usually denoted by

$$S = \{(\mathbf{x}_1, y_1), ..., (\mathbf{x}_l, y_l)\} \subseteq (X \times Y)^l.$$

1.3 Kernel methods

The strategy adopted is to embed the data into a space where the patterns can be discovered as linear relations. It is already possible to stress four main ingredients of kernel methods:

1. data items are embedded into a vector space called the feature space;

2. linear relations are sought among the data items in the feature space;

3. the algorithms are implemented in such a way that the coordinates

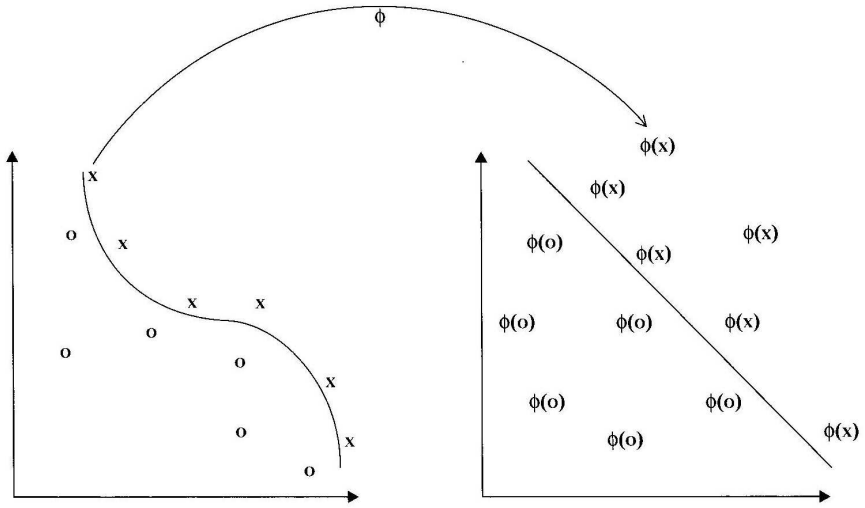

Figure 1.1: The function ϕ embeds the data into a feature space where the nonlinear pattern now appears linear. The kernel computes inner products in the feature space directly from inputs.

of the embedded points are not needed, only their pairwise inner products;

4. the pairwise inner products can be computed efficiently directly from the original data items using a kernel function.

This steps are illustrated in Fig.1.1.

1.3.1 Linear regression in a feature space

Consider the problem of finding a homogeneous real-valued linear function

$$g(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x} \rangle = \mathbf{w}'\mathbf{x} = \sum_{i=1}^{n} w_i x_i,$$

best interpolating a given training set $S = \{(\mathbf{x}_1, y_1), ..., (\mathbf{x}_l, y_l)\}$ of points $\mathbf{x}_i$ in $X \subseteq \mathcal{R}^n$ with corresponding labels y_i in $Y \subseteq \mathcal{R}$. Here, we use the notation $\mathbf{x} = (x_1, x_2, ..., x_n)$ for the n-dimensional input vectors, while $\mathbf{w}'$ denotes the transpose of the vector $\mathbf{w} \in \mathcal{R}^n$. This is naturally one of the simplest relationship one might find in the source $X \times Y$, namely a linear

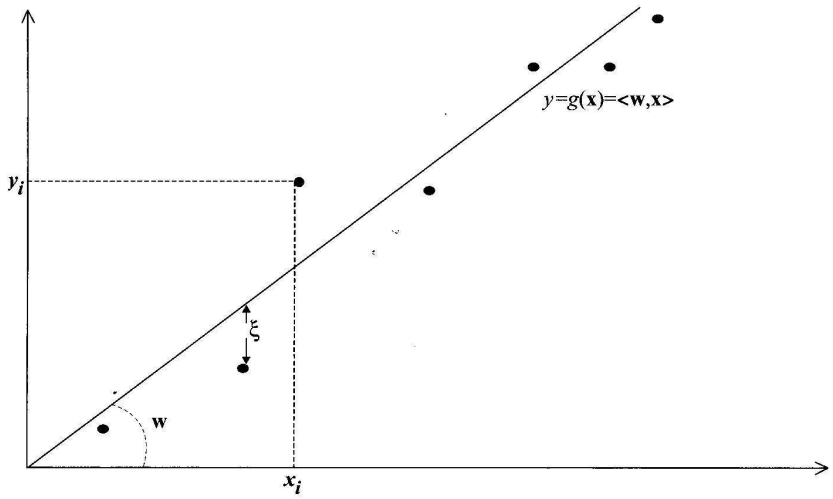

Figure 1.2: A one-dimensional linear regression problem.

function g of the features $\mathbf{x}$ matching the corresponding label y, creating a pattern function that should be approximately equal to zero

$$f((\mathbf{x}, y)) = |y - g(\mathbf{x})| = |y - \langle \mathbf{w}, \mathbf{x} \rangle| \approx 0.$$

This task is also known as *linear interpolation*. Geometrically it corresponds to fitting a hyperplane through the given n-dimensional points. The Fig.1.2 shows an example for $n = 1$.
In the exact case when the data have been generated in the form $(\mathbf{x}, g(\mathbf{x}))$, where $g(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x} \rangle$ and there are exactly $l = n$ linearly independent points, it is possible to find the parameters $\mathbf{w}$ by solving the system of linear equations

$$\mathbf{X}\mathbf{w} = \mathbf{y},$$

where we have used $\mathbf{X}$ to denote the matrix whose rows are the row vectors $\mathbf{x}'_1, ...\mathbf{x}'_l$ and $\mathbf{y}$ to denote the vector $(y_1, ..., y_l)'$. If there are less points than dimensions, there are several possible $\mathbf{w}$ exactly describing the data, and a criterion is needed to chose between them. In this situation the best choice is the vector $\mathbf{w}$ with the smallest norm. If there are more points than dimensions and there is noise in the generation

process, then we should not expect there to be an exact pattern, so that an approximation criterion is needed. In this situation we will select the pattern with the smallest error. The distance shown as ξ in Fig.1.2 is the error of the linear function on the particular training sample, $\xi = (y - g(\mathbf{x}))$. This value is the output of the putative pattern function

$$f((\mathbf{x}, y)) = |y - g(\mathbf{x})| = |\xi| \,.$$

We would like to find a function where all these training errors are small. The sum of the squares of these errors is the most commonly chosen measure of the collective discrepancy between the training data and a particular function

$$\mathcal{L}(g, S) = \mathcal{L}(\mathbf{w}, S) = \sum_{i=1}^{l} (y_i - g(\mathbf{x}_i))^2 = \sum_{i=1}^{l} \xi_i^2 = \sum_{i=1}^{l} \mathcal{L}(g, (\mathbf{x}_i, y_i)),$$

where we have used the same notation $\mathcal{L}(g, (\mathbf{x}_i, y_i)) = \xi_i^2$ to denote the squared error or *loss* of g on example $(\mathbf{x}_i, y_i)$ and $\mathcal{L}(f, S)$ to denote the collective loss of a function f on training set S. The learning problem now becomes the choise of vector $\mathbf{w} \in W$ which minimizes the collective loss. This is a well-studied problem that is applied in almost all the disciplines. It was introduced by Gauss and is known as *least squares approximation*. Using the notation above, the vector of output discrepancies can be written as

$$\xi = \mathbf{y} - \mathbf{X}\mathbf{w}.$$

Hence, the loss function can be written as

$$\mathcal{L}(\mathbf{w}, S) = ||\xi||^2 = (\mathbf{y} - \mathbf{X}\mathbf{w})'(\mathbf{y} - \mathbf{X}\mathbf{w}). \tag{1.2}$$

Note that we again use $\mathbf{X}'$ to denote the transpose of $\mathbf{X}$. We can seek the optimal $\mathbf{w}$ by taking the derivatives of the loss with respect to the parameters $\mathbf{w}$ and setting them equal to zero

$$\frac{\partial \mathcal{L}(\mathbf{w}, S)}{\partial \mathbf{w}} = -\mathbf{X}'(\mathbf{y} - \mathbf{X}\mathbf{w}) + (\mathbf{y} - \mathbf{X}\mathbf{w})'(-\mathbf{X}) =$$

$$
\begin{aligned}
&= -\mathbf{X}'\mathbf{y} + \mathbf{X}'\mathbf{X}\mathbf{w} - \mathbf{y}'\mathbf{X} + (\mathbf{X}\mathbf{w})'\mathbf{X} = \\
&= -\mathbf{X}'\mathbf{y} + \mathbf{X}'\mathbf{X}\mathbf{w} - \mathbf{X}'\mathbf{y} + \mathbf{X}'\mathbf{X}\mathbf{w} \Rightarrow \\
\frac{\partial \mathcal{L}(\mathbf{w}, S)}{\partial \mathbf{w}} &= -2\mathbf{X}'\mathbf{y} + 2\mathbf{X}'\mathbf{X}\mathbf{w} = 0 \qquad\qquad ,
\end{aligned}
$$

where we took into account that $(\mathbf{y}'\mathbf{X})' = \mathbf{X}'\mathbf{y}$. We have obtained the so-called "normal equations"

$$
2\mathbf{X}'\mathbf{X}\mathbf{w} = \mathbf{X}'\mathbf{y} \;\Rightarrow \mathbf{w} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y}. \tag{1.3}
$$

1.3.2 Ridge regression: primal and dual

Notice in equation (1.3) that if the inverse of $\mathbf{X}'\mathbf{X}$ exists, we can express $\mathbf{w}$ in the following way

$$\mathbf{w} = (\mathbf{X}'\mathbf{X})^{-\mathbf{1}}\mathbf{X}'\mathbf{y} = \mathbf{X}'\mathbf{X}(\mathbf{X}'\mathbf{X})^{-\mathbf{2}}\mathbf{X}'\mathbf{y} = \mathbf{X}'\alpha \tag{1.4}$$

making it a linear combination of the training points, $\mathbf{w} = \sum_{i=1}^{l} \alpha_i \mathbf{x}_i$.

If $\mathbf{X}'\mathbf{X}$ is singular, the pseudo-inverse can be used. The goal is to find $\mathbf{w}$ that satisfies the equation (1.3) with minimal norm. Alternatively we can trade off the size of the norm against the loss. This is the approach known as ridge regression, in others terms, it means that there are situations where exactly fitting the data may not be possible.

Ridge regression corresponds to solve the optimisation

$$\min_{\mathbf{w}} \mathcal{L}_\lambda(\mathbf{w}, S) = \min_{\mathbf{w}} \lambda||\mathbf{w}||^2 + \sum_{i=1}^{l}(y_i - g(\mathbf{x_i}))^{\mathbf{2}}, \tag{1.5}$$

where λ is a positive number that defines the relative trade-off between norm and loss and hence controls the degree of regularisation. The learning problem reduces to solve an optimisation problem over $\mathcal{R}^n$.

Taking the derivative of the cost function with respect to the parameters we obtain the equations

$$\mathbf{X}'\mathbf{X}\mathbf{w} + \lambda\mathbf{w} = (\mathbf{X}'\mathbf{X} + \lambda\mathbf{I_n})\mathbf{w} = \mathbf{X}'\mathbf{y} \tag{1.6}$$

where $\mathbf{I_n}$ is the $n \times n$ identity matrix. In this case the matrix $(\mathbf{X}'\mathbf{X} + \lambda\mathbf{I_n})$ is always invertible if $\lambda > 0$, so that the solution is given by

$$\mathbf{w} = (\mathbf{X}'\mathbf{X} + \lambda\mathbf{I_n})^{-\mathbf{1}}\mathbf{X}'\mathbf{y} \tag{1.7}$$

Solving this equation for $\mathbf{w}$ involves solving a system of linear equations with n unknowns and n equation. The complexity of this task is $O(n^3)$. The resulting prediction function is given by

$$g(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x} \rangle = \mathbf{y}'\mathbf{X}(\mathbf{X}'\mathbf{X} + \lambda\mathbf{I_n})^{-\mathbf{1}}\mathbf{x}$$

The equation (1.7) computes the weight vector explicitly and is known as the *primal solution or primal representation.*

Alternatively, we can rewrite equation (1.6) in terms of $\mathbf{w}$ and we obtain

$$\lambda^{-1}\mathbf{X}'(\mathbf{y} - \mathbf{X}\mathbf{w}) = \mathbf{X}'\alpha$$

showing that once again $\mathbf{w}$ can be written as a linear combination of the training points, $\mathbf{w} = \sum_{i=1}^{l} \alpha_i \mathbf{x}_i$ with $\alpha = \lambda^{-1}(\mathbf{y} - \mathbf{X}\mathbf{w})$. Hence, we have

$$\begin{aligned} \alpha &= \lambda^{-1}(\mathbf{y} - \mathbf{X}\mathbf{w}) \\ &\Rightarrow \lambda\alpha = (\mathbf{y} - \mathbf{X}\mathbf{X}'\alpha) \\ &\Rightarrow (\mathbf{X}\mathbf{X}' + \lambda\mathbf{I_n})\alpha = \mathbf{y} \\ &\Rightarrow \alpha = (\mathbf{G} + \lambda\mathbf{I_n})^{-1}\mathbf{y}\,, \end{aligned} \tag{1.8}$$

where $\mathbf{G} = \mathbf{X}\mathbf{X}'$ or, component-wise, $\mathbf{G}_{ij} = \langle \mathbf{x}_i, \mathbf{x}_j \rangle$. Solving for α involves solving l linear equations with l unknowns, a task of complexity $O(l^3)$. The resulting prediction function is given by

$$g(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x} \rangle = \left\langle \sum_{\mathbf{i=1}}^{\mathbf{l}} \alpha_\mathbf{i} \mathbf{x_i}, \mathbf{x} \right\rangle = \sum_{\mathbf{i=1}}^{\mathbf{l}} \alpha_\mathbf{i} \langle \mathbf{x_i}, \mathbf{x} \rangle = \mathbf{y}'(\mathbf{G} + \lambda\mathbf{I_l})^{-1}\mathbf{k},$$

where $k_i = \langle \mathbf{x_i}, \mathbf{x} \rangle$. The equation (1.8) gives the solution as a linear combination of a training examples and is known as the *dual solution or dual representation.*

The crucial observation about the dual solution of equation (1.8) is that the information from the training examples is given by the inner products between pairs of training points in the matrix $\mathbf{G} = \mathbf{X}\mathbf{X}'$. Similarly, the information about a novel example $\mathbf{x}$ required by the predictive function is just the inner products between the training points and the new example $\mathbf{x}$. The matrix $\mathbf{G}$ is referred to as the *Gram matrix.* The Gram matrix and the matrix $(\mathbf{G} + \lambda\mathbf{I_l})$ have dimensions $l \times l$. If the dimension n of the feature space is larger than the number l of the training examples, it becomes more efficient to solve equation (1.8) rather than the primal equation (1.7) involving the matrix $(\mathbf{X}'\mathbf{X} + \lambda\mathbf{I_n})$ of dimension

$n \times n$. Evaluation of the predictive function in this setting is, however, always more costly since the primal involves $O(n)$ operations, while the complexity of the dual is $O(nl)$.

1.3.3 Kernel for nonlinear feature mappings

The ridge regression method addresses the problem of identifying linear relationship between one selected variable and the remaining features, where the relation is assumed to be functional. The resulting predictive function can be used to estimate the value of the selected variable given the values of the other features. Often, however, the relations that are sought are nonlinear, i.e. the selected variable can only be accurately estimated as a nonlinear function of the remaining features. Following the overall strategy on the remaining features of the data into a new feature space in such a way that the sought relations can be represented in a linear form and hence the ridge regression algorithm described above will be able to detect them.
We will consider an embedding map

$$\varphi : \mathbf{x} \in \mathcal{R}^n \rightarrow \varphi(\mathbf{x}) \in F \subseteq \mathcal{R}^N.$$

The choice of the map φ aims to convert the nonlinear relations into linear ones. Hence, the map reflects our expectations about the relation $y = g(\mathbf{x})$ to be learned. The effect of φ is to recode our dataset S as $\widehat{S} = \{(\varphi(\mathbf{x}_1), y_1), ..., (\varphi(\mathbf{x}_l), y_l)\}$. We can now proceed as above looking for a relation of the form

$$f((\mathbf{x}, y)) = |y - g(\mathbf{x})| = |y - \langle \mathbf{w}, \varphi(\mathbf{x}) \rangle| = |\xi|.$$

Although the primal method could be used, problems will arise if N is very large making the solution of the $N \times N$ system of equation (1.7) very expensive in terms of time machine. If, on the other hand, we consider the dual solution, we have shown that all the information the algorithm needs is the inner products between data points $\langle \varphi(\mathbf{x}), \varphi(\mathbf{z}) \rangle$ in the feature space

F. In particular the predictive function $g(\mathbf{x}) = \mathbf{y}'(\mathbf{G} + \lambda \mathbf{I_n})^{-1}\mathbf{k}$ involves the Gram matrix $\mathbf{G} = \mathbf{X}\mathbf{X}'$ with entries

$$\mathbf{G}_{ij} = \langle \varphi(\mathbf{x}_i), \varphi(\mathbf{x}_j) \rangle \,, \tag{1.9}$$

where the rows of X are now the feature vectors $\varphi(\mathbf{x_1})', ..., \varphi(\mathbf{x_l})'$, and the vector $\mathbf{k}$ contains the values

$$k_i = \langle \varphi(\mathbf{x}_i), \varphi(\mathbf{x}) \rangle \,. \tag{1.10}$$

When the value of N is very large, it is worth taking advantage of the dual solution to avoid solving the large $N \times N$ system. Making the optimistic assumption that the complexity of evaluating φ is $O(n)$, the complexity of evaluating the inner products of equation (1.9) and (1.10) is still $O(n)$ making the overall complexity of computing the vector α equal to $O(l^3 + l^2 N)$, while that of evaluating g on a new example is $O(lN)$.

We have seen that in the dual solution we make use of inner products in the feature space. In the above analysis we assumed that the complexity of evaluating each inner product was proportional to the dimension of the feature space. The inner products can, however, sometimes be computed more efficiently as a direct function of the input features, without explicitly computing the mapping φ. In other words the feature-vector representation step can be by-passed. A function that performs this direct computation is know as a *kernel function.*

Definition 1.5 (Kernel function) *A kernel is a function k that for all* $\boldsymbol{x, z} \in \mathbf{X}$ *satisfies*

$$\mathbf{k}(\mathbf{x}, \mathbf{z}) = \langle \varphi(\mathbf{x}), \varphi(\mathbf{z}) \rangle,$$

where φ *is a mapping from* X *to an (inner product) feature space* F

$$\varphi : \mathbf{x} \to \varphi(\mathbf{x}) \in F.$$

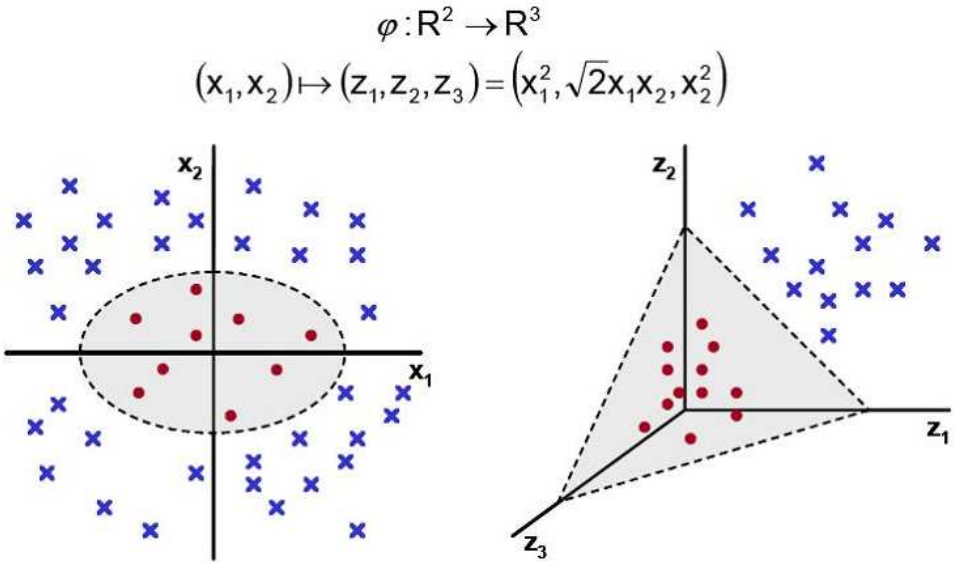

Figure 1.3: A *kernel function* performs a non-linear mapping of the feature vector onto a high-dimensional space that is hidden from inputs or the outputs.

Let's take an example to better understand the previous concept: consider a two-dimensional input space (see left part of Fig. 1.3) $\mathbf{X} \subseteq \mathcal{R}^n$ together with the feature map

$$\varphi : \mathbf{x} = (x_1, x_2) \to \varphi(\mathbf{x}) = (x_1^2, \sqrt{2}x_1x_2, x_2^2) \in F = \mathcal{R}^3.$$

The space of linear functions in F would then be

$$g(\mathbf{x}) = w_{11}x_1^2 + w_{22}x_2^2 + w_{12}\sqrt{2}x_1x_2.$$

The feature map takes the data from a two-dimensional space and it makes a mapping of the data into a three-dimensional space (see right part of Fig. 1.3). The composition of the feature map with the inner product in the feature space can be evaluated as follows

$$\begin{aligned}\langle \varphi(\mathbf{x}), \varphi(\mathbf{z}) \rangle &= \left\langle (x_1^2, \sqrt{2}x_1x_2, x_2^2), (z_1^2, \sqrt{2}z_1z_2, z_2^2) \right\rangle \\ &= x_1^2z_1^2 + x_2^2z_2^2 + 2x_1x_2z_1z_2 \\ &= \langle \mathbf{x}, \mathbf{z} \rangle^2 .\end{aligned}$$

Hence, the function

$$k(\mathbf{x}, \mathbf{z}) = \langle \mathbf{x}, \mathbf{z} \rangle^{\mathbf{2}}$$

is a kernel function and F its corresponding feature space. This means that we can compute the inner product between the projections of two points into the feature space without explicitly evaluating their coordinates.

1.3.4 Kernel matrix

Given a set of vectors, $S = \{\mathbf{x_1}, ..., \mathbf{x_l}\}$ the *Gram matrix* is defined as the $l \times l$ matrix $\mathbf{G}$ whose entries are $\mathbf{G}_{ij} = \langle \mathbf{x_i}, \mathbf{x_j} \rangle$. If we are using a kernel function k to evaluate the inner products in a feature space with feature map φ the associated Gram matrix has entries

$$\mathbf{G}_{ij} = \langle \varphi(\mathbf{x_i}), \varphi(\mathbf{x_j}) \rangle = k(\mathbf{x_i}, \mathbf{x_j}).$$

In this case the matrix is often referred to as the *kernel matrix*. We will use a standard notation for displaying kernel matricies as:

$\mathbf{K}$	1	2	...	l
1	$k(\mathbf{x_1}, \mathbf{x_1})$	$k(\mathbf{x_1}, \mathbf{x_2})$	...	$k(\mathbf{x_1}, \mathbf{x_l})$
2	$k(\mathbf{x_2}, \mathbf{x_1})$	$k(\mathbf{x_2}, \mathbf{x_2})$	...	$k(\mathbf{x_2}, \mathbf{x_l})$
.	.	.	.	.
.	.	.	.	.
.	.	.	.	.
l	$k(\mathbf{x_l}, \mathbf{x_1})$	$k(\mathbf{x_l}, \mathbf{x_2})$	...	$k(\mathbf{x_l}, \mathbf{x_l})$

where the symbol $\mathbf{K}$ in the top left corner indicates that the table represents a kernel matrix. The Gram matrix has already been shown to play an important role in the dual form of some learning algorithms. The matrix is symmetric since $\mathbf{G_{ij}} = \mathbf{G_{ji}}$, that is $\mathbf{G}' = \mathbf{G}$. Furthermore, it contains all the information needed to compute the pairwise distances within the data set as shown above. In the Gram matrix there is of course some information that is lost when compared with the original set of vectors. For example the matrix loses information about the orientation of the original data set with respect to the origin, since the matrix of inner products is invariant for rotations about the origin. *All the information*

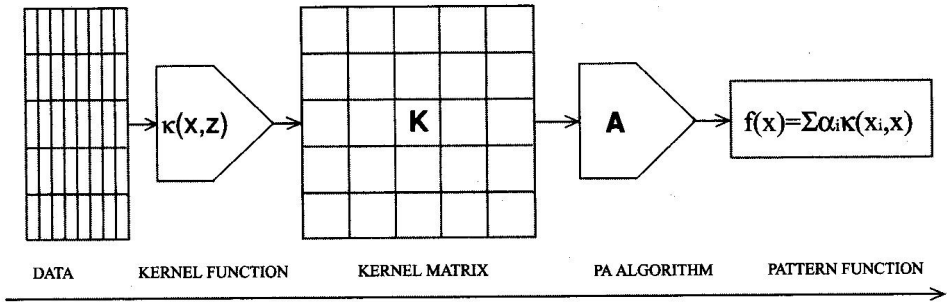

Figure 1.4: The stages involved in the application of kernel methods.

the pattern analysis algorithms can glean about the training data and chosen feature space is contained in the kernel matrix together with any labeling information.

In this sense we can view the matrix as an *information bottleneck* that must transmit enough information about the data for the algorithm to be able to perform its task.

Now, we can summarize all the previous concepts commenting the Fig.1.4. It shows the stages involved in the implementation of kernel pattern analysis. The data is processed using a kernel to create a kernel matrix, which in turn is processed by a pattern analysis algorithm to produce a pattern function. This function is used to process unseen examples.

1.4 Support Vector Machines

Out of the wide variety of classifiers that has been made available by research on classification and learning theory, we have decided to rely on Support Vector Machines (SVM). The main advantages of SVM for our purpose are the fact that they do not require a large training set and that they offer an easy way to control the complexity of the decision surface through the choice of the kernel function.

SVM classifiers constitute a wide research topic, a complete discussion of which lies beyond the scope of the present thesis. Extensive tutorials can be found in [44] and [46]; the "classical" reference [45] provides a concise

introduction. A presentation of the theory underlying SVM can be found in [41], [43].
In the next Section we provide a quick, concise overview of the principles of SVM classifiers, based on the sources quoted above.

1.4.1 Classifiers and training

Following a standard notation, we describe a two-class classifier on an N-dimensional (real) feature space $\mathcal{R}^N$ as a family of functions

$$\{f_\alpha : \alpha \in \Lambda\}, f_\alpha : \mathcal{R}^N \rightarrow \{\pm 1\}. \tag{1.11}$$

This notation expresses the fact that a given classifier (e.g. a *linear discriminants*) can implement a certain type of decision functions f_α (e.g. hyperplanes). Which of these functions actually is implemented depends on a set of tunable parameters represented by the index α (e.g. the weights of the connections). The training process has the purpose of selecting a suitable value for α in the allowed parameter range specified by Λ. The values ± 1 are conventionally selected to indicate the two classes, also called "accept" (+1) and the "reject" (-1) class.
Training is performed on a set of *training examples* for two classes,

$$(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), ..., (\mathbf{x}_l, y_l), \tag{1.12}$$

where $\mathbf{x}_i$ is a vector in $\mathcal{R}^N$ and $y_i \in \{\pm 1\}$ is its corresponding label. We assume that each training example is drawn from a distribution $P(\mathbf{x}, y)$. The aim of the training process is to identify the set of parameters α that minimises the expectation under P of a suitable loss function $\mathcal{L}(f_\alpha(\mathbf{x}), y)$ expressing the cost of a misclassification:

$$E_P(L) = \int \mathcal{L}(f_\alpha(\mathbf{x}), y) dP(\mathbf{x}, y). \tag{1.13}$$

This formulation is quite general, as it takes into account the fact that the two possible types of error, i.e. False Acceptance and False Rejection, will in principle have different costs depending on the application. It is

here convenient to restrict ourselves to the case in which the two types of errors are equally undesirable. In this case, training the classifier amounts to minimizing the expected number of errors under P, a quantity that is known as the *risk*:

$$R(\alpha) = \int \frac{1}{2}|f_\alpha(\mathbf{x}) - y| dP(\mathbf{x}, y). \tag{1.14}$$

The resulting training algorithm, and indeed the classifier that we will be inclined to use, depends at this point on the type of assumptions that we are willing to make about the distribution function $P(\mathbf{x}, y)$. If, for instance, there are reasons to believe that the distributions $P(\mathbf{x}, +1)$ and $P(\mathbf{x}, -1)$ are Gaussians the optimal solution, i.e. Nearest Neighbour classifiers, is provided, by Bayesian decision theory. In the case that no parametric model is available for P, Equation (1.14) can be minimised only by resorting to some *induction principle*.

The most direct and simple approach consist in minimising the *empirical risk*

$$R_{emp}(\alpha) = \frac{1}{l} \sum_{i=1}^{l} \frac{1}{2}|f_\alpha(\mathbf{x}) - y|, \tag{1.15}$$

which represents the fraction of errors over the points of the training set. However, this is do not guaranteed to produce a good generalisation ability if the number of training examples is small with respect to the "capacity" of the classifier, intended as some measure of complexity of the decision functions it can generate (e.g. the number of adjustable parameters or the number of neurons in a Neural Network). This loss of generalisation ability due to an "oversized" classifier is known as *over-fitting*. Although a number of regularization techniques has been devised to overcome this problem [47], they often find their place only as afterthoughts in traditional Neural Network approaches. One of the basic ideas of the Statistical Learning Theory developed by V. Vapnik and co.workers consists in incorporating the control of the complexity of the classifier in the induction principle, which is accomplished by the *Structural Risk Minimisation* principle.

1.4.2 Structural Risk Minimisation

The Structural Risk Minimisation induction principle is based on an upper bound on the risk which is independent the probability distribution $P(\mathbf{x}, y)$. For the learning problem stated above, for any $\alpha \in \Lambda$ and $l > h$, the following inequality holds [41] with probability $1 - \eta$:

$$R(\alpha) \leq R_{emp}(\alpha) + \varphi\left(\frac{h}{l}, \frac{\log(\eta)}{l}\right). \tag{1.16}$$

The *confidence term* φ is defined as

$$\varphi\left(\frac{h}{l}, \frac{\log(\eta)}{l}\right) = \sqrt{\frac{h(\log\frac{2l}{h} + 1) - \log\frac{\eta}{4}}{l}}. \tag{1.17}$$

The parameter h is called the Vapnik-Cervonenkis (VC) dimension of the family of functions $\{f_\alpha\}$; l is the size of training set.

The VC dimension is a measure of the capacity of the family of functions, in a sense that will presently be specified. Given a set of l points in $\mathcal{R}^N$, we consider all the possible 2^l assignments of each of these points to the "accept" and "reject" classes. Now, if for each such assignment there exists a decision function $f_{\bar{\alpha}}$ respecting the division among the two classes, we say that the set of points is *shattered* by the set of functions. The VC dimension of the set of functions $\{f_\alpha\}$ is defined as the maximum number of points that it can shatter. Note that for a set of functions to have VC dimension l it is sufficient that it will shatter one particular set of l points.

The probabilistic bound (1.16) is far from being tight. Indeed, there exist classifiers (such as KNN classifier) that have an infinite VC dimension but in practice have a good generalisation ability. Nevertheless, the bound does indicate that, in order to obtain a good generalisation ability, one should be able to control both the empirical risk $R_{emp}(\alpha)$ and the complexity of the classifier as measured by $h(\{f_\alpha : \alpha \in \Lambda\})$. The empirical risk depends on the particular decision function chosen and can be minimised by appropriately tuning the parameter set α. The VC dimension depends on the set of decision surfaces available to the

classifier, which can be controlled by restricting the choice of α to a subset of Λ. In particular, one can construct a series of classifier of increasing complexity by introducing the series of nested sets $S_k = \{f_\alpha | \alpha \in \Lambda_k\}$, so that

$$S_1 \subset S_2 \subset ... \subset S_n \subset ... \tag{1.18}$$

and therefore

$$h_1 \leq h_2 \leq ... \leq h_n \leq ... \ . \tag{1.19}$$

For a given set of training data $(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), ..., (\mathbf{x}_l, y_l)$ the *Structural Risk Minimisation* principle chooses the function f_{α_n} in the subset S_n for which the right end side of the risk bound (1.16) is minimal. That is to say that a balance is struck between the search for good approximation of the data and the complexity of the family of functions used for such approximation.

1.4.3 Linear Support Vector Machines

Suppose we are dealing with a linearly separable set of observations

$$(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), ..., (\mathbf{x}_l, y_l),$$

meaning that there exists an (oriented) hyperplane

$$f(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} + b = 0 \tag{1.20}$$

such that $f(\mathbf{x_i}) > 0$ if and only if $y_i = +1$. We can define the *Optimal Separating Hyperplane* (OSH) as the one which has the maximal distance d_{max} from the closest example. This requirement alone only defines $f(\mathbf{x})$ in Equation (1.20) up to a multiplicative factor. In order to remove this ambiguity, we define a *canonical form* for the OSH by rescaling $\mathbf{w}$ and b so that $||w|| = 1/d_{max}$. It follows that

$$\begin{cases} \mathbf{w} \cdot \mathbf{x_i} + b \geq +1 \;\; if \;\; y_i = +1, \\ \mathbf{w} \cdot \mathbf{x_i} + b \leq -1 \;\; if \;\; y_i = -1 \end{cases} \tag{1.21}$$

and that the projections of the closest examples from the two classes along the direction of $\mathbf{w}$ are at least $2/||w||$ apart (Fig.1.5). Support Vector Machines assign data to the "accept" (+1) or "reject" (-1) classes based on the (canonical) Optimal Hyperplane decision function $f(\mathbf{x}) \geq 0$. From a practical point of view, the Optimal Separating Hyperplane can be determined by minimising the functional

$$\tau(\mathbf{w}) = \frac{1}{2}||\mathbf{w}||^2 \tag{1.22}$$

subject to the constraints (1.21). This is a convex optimisation problem. A convenient way of accounting for the inequality constraints consists of introducing a Lagrangian

$$L(\mathbf{w}, b, \alpha) = \tau(\mathbf{w}) - \sum_{i=1}^{l} \alpha_i [y_i(\mathbf{w} \cdot \mathbf{x_i} + b) - 1], \tag{1.23}$$

where the α_i are non-negative multipliers. The solution of the constrained optimisation problem then corresponds to the saddle point

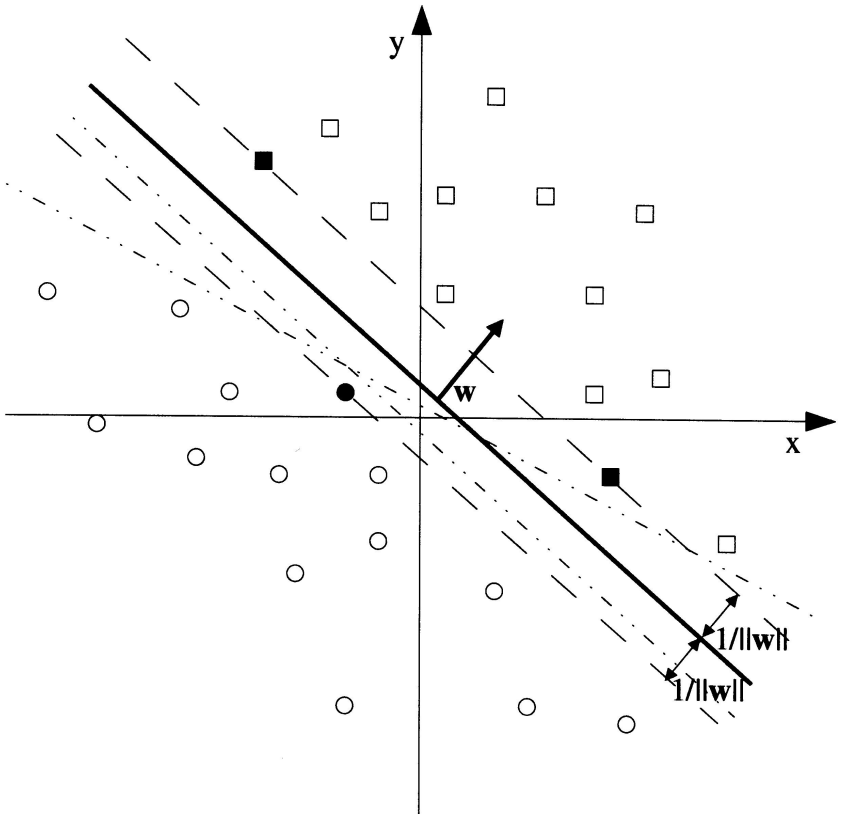

Figure 1.5: A linear Support Vector Machine classifier in $\mathcal{R}^2$ (separable case). Training examples for the (-1) and (+1) classes are represented as circles and squares respectively. Filled symbols denote the support vectors. The thick line represents the Optimal Separating Hyperplane. The dash-dotted lines are examples of suboptimal separating hyperplanes that do not maximise the margin.

that maximises (1.23) with respect to the α_i and minimises it with respect to $\mathbf{w}$ and b. The conditions on the partial derivatives at the saddle point:

$$\frac{\partial L(\mathbf{w}, b, \alpha)}{\partial \mathbf{w}} = 0, \quad \frac{\partial L(\mathbf{w}, b, \alpha)}{\partial b} = 0 \tag{1.24}$$

lead to two equalities

$$\sum_{i=1}^{l} \alpha_i y_i = 0 \tag{1.25}$$

and

$$\mathbf{w} = \sum_{i=1}^{l} \alpha_i y_i \mathbf{x}_i. \tag{1.26}$$

The normal vector to the OSH therefore is a linear combination of the training examples (note that while the solution of the convex problem is unique, the coefficients α_i do not need to be unique, since the training vectors are not constrained to be linearly independent). The Kuhn-Tucker conditions [48, 49] guarantee that, at saddle point, the following relation holds:

$$\alpha_i[y_i(\mathbf{w} \cdot \mathbf{x}_i + b) - 1] = 0, \quad i = 1, ..., l. \tag{1.27}$$

Therefore, the non-zero multipliers α_i correspond to training examples for which the equality sign applies in the constraints (1.21). Such training examples are called the *Support Vectors* and will be indicated as $\mathbf{s}_j$, $j = 1, ..., n$. The equation of the Optimal Separating Hyperplane (1.20) can therefore be rewritten explicitly in terms of the Support Vectors, leading to the following expression for the decision function:

$$f(\mathbf{x}) = \sum_{j=1}^{n} \alpha_j y_j \mathbf{s}_j \cdot \mathbf{x} + b \geq 0 \tag{1.28}$$

The training examples $\mathbf{x}_i$ other than the Support Vectors do not appear in this expression. They automatically satisfy the constraints (1.21) and the solution would be unaffected if they were removed from the training set.

1.4.4 SVM and Structural Risk Minimisation

The relation between the *Structural Risk Minimisation* induction principle and Support Vector Machine classifiers is clarified by the following bound on the VC dimension of classifiers whose decision functions are hyperplanes in $\mathcal{R}^N$.

Let R be the radius of the smallest ball $\mathbf{B}_R$ in $\mathcal{R}^N$ containing the training examples $\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_l$, and we consider two samples $\mathbf{x}_1$ and $\mathbf{x}_2$ from different class with $(\mathbf{w} \cdot \mathbf{x}_1) + b = 1$ and $(\mathbf{w} \cdot \mathbf{x}_2) + b = -1$, respectively. Then the margin is given by the distance of these two points, measured

perpendicularly to the hyperplane,

$$\left(\frac{\mathbf{w}}{||\mathbf{w}||} \cdot (\mathbf{x}_1 - \mathbf{x}_2)\right) = \frac{2}{||\mathbf{w}||}.$$

The result linking the VC-dimension of the class of separating hyperplanes to the margin or the length of the weight vector $\mathbf{w}$ respectively is given by the following inequalities [41]:

$$h \leq A^2 R^2 + 1 \quad and \quad ||\mathbf{w}||_2 \leq A \tag{1.29}$$

Thus, if we bound the margin of a function class from below, say by $\frac{2}{A}$, we can control its VC-dimension. By maximising the margin, i.e. minimising $||\mathbf{w}||$, we therefore minimise the confidence term in the risk bound Equation (1.16). So far, we have made the assumption that the data are linearly separable, i.e. $R_{emp} = 0$; this condition will be lifted in the following Section.

1.4.5 Soft margin hyperplanes

In the case that no hyperplane decision function exists that correctly separates the vectors of the training set, an Optimal Separating Hyperplane can still be found by making allowance for a few classification errors.

From the formal point of view, this is accomplished by introducing a set of l *slack variables* $\xi_i \geq 0$ and by rewriting the bounds in Equation (1.21) in the following way:

$$\begin{cases} \mathbf{w} \cdot \mathbf{x_i} + b \geq +1 - \xi_i \ \ if \ y_i = +1, \\ \mathbf{w} \cdot \mathbf{x_i} + b \leq -1 + \xi_i \ \ if \ y_i = -1 \ . \end{cases} \tag{1.30}$$

The number of classification errors can be controlled by modifying the objective function for the minimisation:

$$\tau'(\mathbf{w}) = \frac{1}{2}||\mathbf{w}||^2 + C \sum_{i=1}^{l} \xi_i \tag{1.31}$$

which is minimised subject to the new constraints. The solution is the same as in the separable case, the only difference being the existence of

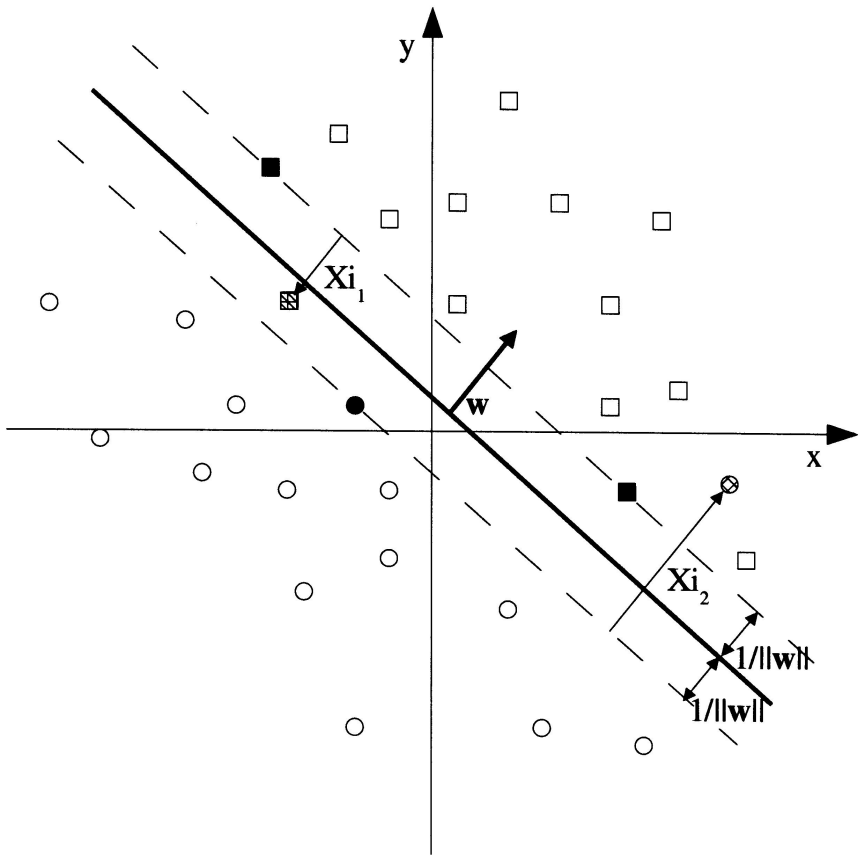

Figure 1.6: A linear Support Vector Machine classifier in $\mathcal{R}^2$ (non-separable case). Training examples for the (-1) and (+1) classes are represented as circles and squares respectively. the thick, slanted line represents the Optimal Separating Hyperplane. The hatched circle and square constitute two classification errors. Note how $\xi_i \approx 1$, while $\xi_2 > 2$.

an upper bound on the coefficients α_i: we now have $0 \leq \alpha_i \leq C$. In intuitive terms, this means that no training vector is allowed to modify the solution "too much".

The term $\sum_{i=1}^{l} \xi_i$ represents an upper bound on the number of classification errors, that is on the *empirical risk* R_{emp} in the risk bound (1.16). However, the sum of the ξ_i can significantly exceed the number of classification errors if many of the slack variables have large values (Fig.1.6). For this reason, Support Vector Machines only implement the Structural Risk Minimisation principle *approximately*. A study of the

dependence of the solution on the constant C, which sets the balance between the control of the VC dimension (maximisation the margin) and the minimisation of classification errors, can be found in [50]. However, from the practical point of view, classification errors can be quite rare when the number l of training examples is very small as compared with the dimensionality N of the vector space.

1.4.6 The nonlinear case

Support Vector Machine classifiers can afford more complex (nonlinear) decision function by re-mapping input vectors to a higher dimensional space in which classification is performed using the Optimal Separating Hyperplane decision function. The SVM optimisation problem only involves the dot products among the training vectors $\mathbf{x}_i$ (see Equations 1.22, 1.23 and 1.26). Therefore, the function $\varphi : \mathcal{R}^N \rightarrow \mathcal{H}$ that maps the vectors $\mathbf{x}_i$ onto the new space (a Hilbert space, in general case) does not need to be given explicitly. One only needs to specify the dot product of any two image vectors $\varphi(\mathbf{x})$ and $\varphi(\mathbf{y})$ in $\mathcal{H}$ through a *kernel function* K defined over $C \times C$, where C is a compact subset of $\mathcal{R}^N$ that includes the training and test vectors:

$$\langle \varphi(\mathbf{x}), \varphi(\mathbf{y}) \rangle := K(\mathbf{x}, \mathbf{y}). \tag{1.32}$$

In order for this definition to be well-posed, K must satisfy the conditions of Mercer's theorem of Functional Analysis [51]. More specifically, $K(\mathbf{x}, \mathbf{y})$ must be symmetric and continuous over $C \times C$. Furthermore, the integral operator over $L^2(C)$

$$(\mathcal{K}f)(\mathbf{y}) = \int_C K(\mathbf{x}, \mathbf{y}) f(\mathbf{x}) d\mathbf{x} \tag{1.33}$$

must be positive, meaning that

$$\int_{C \times C} K(\mathbf{x}, \mathbf{y}) f(\mathbf{x}) f(\mathbf{y}) d\mathbf{x} d\mathbf{y} \geq 0 \tag{1.34}$$

for all $f \in L^2(C)$.

When these conditions are met, it is possible to find a map φ of the input

vectors onto a suitable Hilbert space $\mathcal{H}$ such that (1.32) defines a scalar product in $\mathcal{H}$.

In this case, the dual Lagrangian to be solved is:

$$min\ L_D = \frac{1}{2}\sum_{i,j}^{N} \alpha_i \alpha_j y_i y_j K(x_i, x_j) - \sum_{i=1}^{N} \alpha_i \tag{1.35}$$

$$\alpha_i \geq 0, \quad i = 1, ..., N$$

$$\sum_{i=1}^{N} \alpha_i y_i = 0.$$

The Optimal Separating Hyperplane in $\mathcal{H}$ can be written in terms of the input vectors in $\mathcal{R}^N$, giving the following expression for the decision function:

$$f(\mathbf{x}) = \sum_{j=1}^{n} \alpha_j y_j K(\mathbf{s}_j, \mathbf{x}) + b \geq 0. \tag{1.36}$$

This is the exact analogue of Equation (1.28), but for the fact that the dot product $\mathbf{s}_j \cdot \mathbf{x}$ in $\mathcal{R}^N$, has been substituted by the value $K(\mathbf{s}_j, \mathbf{x})$ of the kernel functions. Since the kernel function is in general nonlinear, the decision surface $f(\mathbf{x}) = 0$ is also nonlinear. Common choices for K are the family of kernel functions [52, 53]

$$K_d(\mathbf{x}, \mathbf{y}) = (\gamma \mathbf{x} \cdot \mathbf{y} + r)^d, \tag{1.37}$$

that lead to polynomial classifiers, and the Gaussian kernel

$$K(\mathbf{x}, \mathbf{y}) = \exp\left(-\frac{||\mathbf{x} - \mathbf{y}||^2}{\sigma^2}\right). \tag{1.38}$$

1.5 Recursive Features Elimination (RFE)

SVM - Recursive Features Elimination is a general method for eliminating features responsible of small changes in the classifier's cost function. In

the specific case of non-linear SVM, the cost function to minimize is that discussed in eq. (1.35) or more compactly:

$$J = \frac{1}{2}\alpha^T \mathbf{H}\alpha - \alpha^T \mathbf{1} \tag{1.39}$$

where $\mathbf{H}$ is the matrix with elements $y_i y_j K(x_i, x_j)$ and $\mathbf{1}$ is an one-dimensional vector of ones. In order to compute the change in the cost function by removing the feature f, one has to compute the matrix $\mathbf{H}(-f)$ where the notation $(-f)$ means that the feature f has been removed. The variation in the cost function J is thus:

$$\delta J = \frac{1}{2}\alpha^T \mathbf{H}\alpha - \frac{1}{2}\alpha^T \mathbf{H}(-f)\alpha. \tag{1.40}$$

The feature corresponding to the smallest $\delta J(f)$ is then removed, SVM is trained once again with the new smaller set of features, and finally tested. The procedure can thus be iterated (feature after feature) until a reasonable small number of features survives or the performances of the classifier start to degrade.

We present below an outline of the algorithm in the linear case, using SVM-train.

Algorithm SVM-RFE:

Inputs:

Training examples

$$X_0 = [\mathbf{x_1}, \mathbf{x_2}, ...\mathbf{x_k}, ...\mathbf{x_l}]^T$$

Class labels

$$\mathbf{y} = [y_1, y_2, ...y_k, ..., y_l]^T$$

Initialize:

Subset of surviving features

$$\mathbf{s} = [1, 2, ...n]$$

Feature ranked list

$$\mathbf{r} = []$$

Repeat until $\mathbf{s} = []$

Restrict training examples to good feature indices

$$X = X_0(:, \mathbf{s})$$

Train the classifier

$$\alpha = SVM - train(X, \mathbf{y})$$

Compute the weight vector of dimension length($\mathbf{s}$)

$$\mathbf{w} = \sum_k \alpha_k y_k \mathbf{x}_k$$

Compute the ranking criteria

$$c_i = (w_i)^2, \text{ for all i}$$

Find the feature with smallest ranking criterion

$$f = argmin(\mathbf{c})$$

Update feature ranked list

$$\mathbf{r} = [\mathbf{s}(f), \mathbf{r}]$$

Eliminate the feature with smallest ranking criterion

$$\mathbf{s} = \mathbf{s}(1 : f - 1, f + 1 : length(\mathbf{s}))$$

Output:

Feature ranked list $\mathbf{r}$.

In this case (linear case), the kernel function is $K(\mathbf{x}_h, \mathbf{x}_k) = \mathbf{x}_h \cdot \mathbf{x}_k$ and $\alpha^T H \alpha = \|\mathbf{w}\|^2$. The variation of the cost function becomes $\delta J = \frac{1}{2}(w_i)^2$. Computationally, the non-linear kernel version of SVM-RFE is a little more expansive than the linear version. However, the change in matrix H must be computed for support vectors only, which makes it affordable for small numbers of support vectors. Additionally, parts of the calculation such as the dot products $\mathbf{x}_h \cdot \mathbf{x}_k$ between support vectors can be cached. This algorithm can be generalized to remove more than one feature per step for speed reasons.

Chapter 2

Classical Molecular Dynamic (CMD) and Heavy Ion Phase-Space Exploration (HIPSE) models

Computer simulation methods have become a very powerful tool to attack the many - body problem in nuclear physics. Although the theoretical description of complex systems in the framework of statistical physics is rather well developed and the experimental techniques for detailed microscopic information are rather sophisticated, it is often only possible to study specific aspects of those systems in great detail via the simulation. On the other hand, simulations need specific input parameters that characterize the system in question, and which come either from theoretical considerations or are provided by experimental data. Having characterized a physical system in terms of model parameters, simulations are often used both to solve theoretical problems beyond certain approximations and to provide a hint to experimentalists for further investigations.

In the case of big experimental facilities it is even often required to prove the potential outcome of an experiment by computer simulations. In that way one can say that the field of computer simulations has developed into a very important branch of science, which on the one hand helps theorists

and experimentalists to go beyond their *inherent limitations* and on the other hand is a scientific field on its own.

The "traditional" simulation methods for many-body system can be divided into two classes of stochastic and deterministic simulations, which are largely covered by the Monte Carlo (MC) method and the molecular dynamics (MD) method, respectively. Monte Carlo simulations probe the configuration space by trial moves of particles. By contrast, MD methods are governed by the system's Hamiltonian and consequently Hamilton's equations of motion

$$\dot{p}_i = -\frac{\partial H}{\partial q_i} \quad , \quad \dot{q}_i = \frac{\partial H}{\partial p_i} \tag{2.1}$$

are integrated to move particles to new positions and to get new velocities at these new positions. This is advantage of MD simulations with respect to MC, since not only the configuration space probed but the whole phase space which gives additional information about the dynamics of the system. Both methods are complementary in nature but they lead to the same averages of static quantities, given that the system under consideration is ergodic and the same statistical ensemble is used.

Although there are different methods to obtain information about complex systems, particle simulations always require a model for the interaction between system constituents. This model has to be tested against experimental results, i.e. it should reproduce or approximate experimental findings like distribution functions or phase diagrams, and theoretical constraints, i.e. it should obey certain fundamental or limiting laws like energy conservation.

How we have seen in the previous chapter the Support Vector Machine algorithm needs of a training phase where the SVM "learns" to discriminate between different classes of events. Therefore for the learning phase we have used two different types of nuclear models: The Classical Molecular Dynamic (CMD) [54] and the Heavy Ion Phase-Space Exploration model (HIPSE) [55]. This last one has been used for two

main motivations: the first one is that the SVM algorithm is an *adaptive process*, therefore it needs to check the capacity of classification of SVM on different datasets that come from different types of nuclear model, the second one is that the HIPSE is a nuclear model that takes into account both the dynamical and statistical effects. In this chapter we will do a brief description of these theoretical models.

2.1 Classical Molecular Dynamic model

In this model the main ingredients for a program are basically threefold:

1. As already mentioned, a model for the interaction among system constituents (in this case nucleons) is needed. It is assumed that the nucleons interact only pairwise. This assumption greatly reduces the computational effort and the work to implement the model into the program.

2. An integrator is needed, which propagates particle positions and velocities from time t to $t + \delta t$. It is a finite difference scheme which moves trajectories discretely in time. The time step δt has properly to be chosen to guarantee stability of the integrator, i.e. there should be no drift in the system's energy.

3. A statistical ensemble has to be chosen, where thermodynamic quantities like pressure, temperature or the number of nucleons are controlled. The natural choice of an ensemble in MD simulations is the micro-canonical ensemble, since the system's Hamiltonian without external potentials is a conserved quantity.

2.1.1 Model for nucleons interaction

A system is completely determined through its Hamiltonian $H = H_0 + H_1$, where H_0 is the *internal* part of the Hamiltonian, given as

$$H_0 = \sum_{i=1}^{N} \frac{\mathbf{p}_i^2}{2m_i} + \sum_{i<j}^{N} u(\mathbf{r}_i, \mathbf{r}_j) + ... \tag{2.2}$$

where $\mathbf{p}$ is the momentum, m the mass of the particles and u is pair-body interaction potentials. H_1 is an external part, which can include time dependent effects or external sources for a force, i.e. the Coulomb field.

In our case, the elementary interaction is chosen such as to give similar properties of nuclei in their *ground state*, i.e. sizes, etc. Particles interact through a two-body potential repulsive at short distances and attractive at larger distances and purely repulsive for identical particles. This is done to mimic somehow the Pauli principle; for instance, two identical particles that we can call protons if they are charged (p) or neutrons (n) if they are not, cannot make a bound state. The CMD model assumes that the equation of state (EOS) for infinite nuclear matter is very similar to the one of a Van der Waals gas. Nevertheless it should be kept in mind that the ground state in this model is a crystal. By solving this kind of model we can keep track of all positions and momenta as a function of time, and thus we can investigate correlations of all order at all time.

The two-body potential is expressed explicitly by these relations:

$$\begin{aligned} V_{np}(r) &= V_r[\exp(-\mu_r r)/r - \exp(-\mu_r r_c)/r_c] \\ &\quad -V_a[\exp(-\mu_a r)/r - \exp(-\mu_a r_a)/r_a], \\ V_{nn}(r) = V_{pp}(r) &= V_0[\exp(-\mu_0 r)/r - \exp(-\mu_0 r_c)/r_c]. \end{aligned} \tag{2.3}$$

where $r_c = 5.4$ fm is a cut-off radius that is introduced as limit beyond of which mutual interactions between particles are neglected. V_{np} is the potential acting between a neutron and a proton, and V_{nn} and V_{pp} are the potential acting between two identical nucleons. The first is attractive at large r and repulsive at small r, while the latter is purely repulsive.

2.1.2 The Integrator

For a given potential model which characterizes the physical system, it is the integrator which is responsible for the accuracy of the simulation results. If the integrator would work without any error the simulation would provide *exact model results* within the errors occurring due to a finite number representation. However, any finite difference integrator is naturally an approximation for a system developing continuously in time. The requirements for the integrator are therefore to be

- accurate, in the sense that it approximates the *true* trajectory very well (this may be checked with simple model systems for which analytical solutions exist)
- stable, in the sense that it conserves energy and that small perturbation do not lead instabilities
- robust, in the sense that it allows for large time steps in order to propagate the system efficiently through phase space.

In the following we will show the most common integrators based on Taylor expansions, in CMD model the classical Hamilton's equation of motion are solved using the Taylor method at order $\mathcal{O}[(\delta t)^3]$.
The simplest and most straight forward way to construct an integrator is by expanding the position and velocities in a Taylor series. For a small enough time step δt the expansion gives

$$\mathbf{r}(t+\delta t) = \mathbf{r}(t) + \mathbf{v}(t)\delta t + \frac{1}{2}\mathbf{a}(t)\delta t^2 + \frac{1}{6}\mathbf{b}(t)\delta t^3 + ... \tag{2.4}$$

$$\mathbf{v}(t+\delta t) = \mathbf{v}(t) + \mathbf{a}(t)\delta t + \frac{1}{2}\mathbf{b}(t)\delta t^2 + \frac{1}{6}\mathbf{c}(t)\delta t^3 + ... \tag{2.5}$$

where $\mathbf{a}, \mathbf{b}, \mathbf{c}$ are the 2nd, 3rd and 4th time derivate of the coordinates. In the same way the expansion may be performed for $\delta t \rightarrow -\delta t$, which gives

$$\mathbf{r}(t-\delta t) = \mathbf{r}(t) - \mathbf{v}(t)\delta t + \frac{1}{2}\mathbf{a}(t)\delta t^2 - \frac{1}{6}\mathbf{b}(t)\delta t^3 \pm ... \tag{2.6}$$

$$\mathbf{v}(t-\delta t) = \mathbf{v}(t) - \mathbf{a}(t)\delta t + \frac{1}{2}\mathbf{b}(t)\delta t^2 - \frac{1}{6}\mathbf{c}(t)\delta t^3 \pm ... \tag{2.7}$$

Adding up Eqs. 2.4, 2.6 and Eqs. 2.5, 2.7 gives for the new positions and velocities

$$\mathbf{r}(t+\delta t) = 2\mathbf{r}(t) - \mathbf{r}(t-\delta t) + \mathbf{a}(t)\delta t^2 + \mathcal{O}(\delta t^4) \tag{2.8}$$

$$\mathbf{v}(t+\delta t) = 2\mathbf{v}(t) - \mathbf{v}(t-\delta t) + \mathbf{b}(t)\delta t^2 + \mathcal{O}(\delta t^4). \tag{2.9}$$

A method whose truncation varies as $\delta t^{(n+1)}$ is called and n-th order method. Eqs. 2.8, 2.9 are therefore of 3rd order. The drawback of Eq. 2.9 is, however, that it requires the 3rd derivative of the coordinates with respect with to time which is not routinely calculated in CMD simulations and thus introduces some additional computational and storage overhead. To overcome this drawback one can simply subtract Eq.2.6 from Eq. 2.4, giving the central difference scheme for the velocity

$$\mathbf{v}(t) = \frac{1}{2\delta t}(\mathbf{r}(t+\delta t) - \mathbf{r}(t-\delta t)) + \mathcal{O}(\delta t^3). \tag{2.10}$$

This is, however, one order less in accuracy than Eq. 2.9 and also provides velocities at timestep t, not at $t+\delta t$. Since this information is not required by Eq. 2.8 to calculate accurately the positions, one may take Eq. 2.10 as an estimate for the velocities from which the kinetic energy of the system is calculated.

From the point of view of storage requirements, Eq. 2.8,2.9 are not optimal, since information is required form positions not only at time t but also at time $t - \delta t$. An equivalent algorithm, which stores only information from one timestep is the so called *velocity Verlet* algorithm, which reads

$$\mathbf{r}(t+\delta t) = \mathbf{r}(t) + \mathbf{v}(t)\delta t + \frac{1}{2}\mathbf{a}(t)\delta t^2 \tag{2.11}$$

$$\mathbf{v}(t+\delta t) = \mathbf{v}(t) + \frac{1}{2}\delta t(\mathbf{a}(t) + \mathbf{a}(t+\delta t)) \tag{2.12}$$

This scheme, however, requires the knowledge of the accelerations, $\mathbf{a}$, at time step $t + \delta t$.

One may therefore decompose Eq. 2.12 into two steps. First calculate

$$\mathbf{v}(t + \delta t/2) = \mathbf{v}(t) + \frac{1}{2}\delta t \mathbf{a}(t) \tag{2.13}$$

then compute the actual forces on the particles at time $t + \delta t$ and finish the velocity calculation with

$$\mathbf{v}(t + \delta t) = \mathbf{v}(t + \delta t/2) + \frac{1}{2}\mathbf{a}(t + \delta t)\delta t. \tag{2.14}$$

At this point the kinetic energy may be calculated without a time delay of δt, as it was in Eq.2.10.

2.1.3 The Microcanonical Ensemble for CMD

In CMD simulations it is possible to realize different types of thermodynamics ensembles which are characterized by the control of certain thermodynamics quantities.

The microcanonical ensemble may be considered as *natural* ensemble for molecular dynamic simulations (as it is the canonical ensemble for Monte Carlo simulations). If no time dependent external forces are considered, the system's Hamiltonian is constant, implying that the system's dynamics evolves on a constant energy surface. The corresponding probability density in phase space is therefore given by

$$\rho(\mathbf{q}, \mathbf{p}) = \delta(\mathcal{H}(\mathbf{q}, \mathbf{p}) - E) \tag{2.15}$$

In a computer simulation this theoretical condition is generally violated, due to limited accuracy in integrating the equations of motion and due to errors resulting from a limited precision of number representation.

The simplest extension to the microcanonical ensemble is the canonical one, where the number of particles, the volume and temperature are fixed to prescribed values. The temperature T is, in contrast to N and V, an intensive parameter. The extensive counterpart would be the kinetic

energy of the system.

In our CMD simulations the energy and momentum are well conserved. The nucleus is initialized in its ground state by using the frictional cooling method [56]. In other hand, it means that for to find the ground state of the system only, we add a viscosity and a Lagrangian force to $\dot{p}_i = -\frac{\partial H}{\partial q_i}$:

$$\dot{p}_i = -\frac{\partial H}{\partial q_i} - \mu \dot{q}_i - k q_i. \tag{2.16}$$

The second term is a completely artificial one-body viscosity term designed to dissipate energy with time, so that the system relaxes to a state of lowest energy. μ is taken to be ≈ 115 MeV/fm.

The third term is derived from the constraint which requires that $\langle r^2 \rangle$ have a given value. The corresponding Lagrangian potential, $\frac{1}{2} k \sum_i r_i^2$, is not to be included in $\mathcal{H}$ for the purpose of calculating the energy. This Lagrangian oscillator potential was used to compress the system in order to determine the compressibility constant k.

It was also found to be essential to use a small but finite k in order to confine the system until it was nearly cold; then k was set equal to zero for the final relaxation.

The nuclear ground states for projectiles and targets, which can be calculated from the model, contain (at least) two-fold continuous degeneracies. The first is with respect to orientation of the individual nucleons. The second is with respect to the orientations of the momenta of the individual nucleons. Each of these sets of vectors may be rotated independently as a solid body or inverted without affecting the energy. The solid body rotations can be effected simply in terms of the Eulero angles,

$$0 \leq \alpha \leq 2\pi, \quad 0 \leq \beta \leq \pi, \quad 0 \leq \gamma \leq 2\pi,$$

where these are the angles for sequential rotations about the laboratory-fixed axes (say) x, y, z. In a collision calculation, it is necessary to average over initial conditions. For that purpose, one uses the uniform

random variables α, β and γ. Beginning with a standard ground state configuration, then, one generates the starting configuration from a random orientation of the nucleon coordinates and an independent random orientation of the momenta.

Afterward, it is excited at a temperature T giving a Maxwellian velocity distribution to its nucleons by means of a Metropolis sampling [57]. We have studied the reaction $^{58}Ni +^{48} Ca$, investigated at 25 AMeV beam incident energy by NUCL-EX collaboration, at superconducting cyclotron of the Laboratori Nazionali del Sud in Catania with the CHIMERA apparatus.

The events generated by CMD for an impact parameter in the range $[0.5 - 10]$ fm where analyzed with SVM before and after applying a software filter, which simulates the experimental response of each simple module of the apparatus.

2.2 Heavy-Ion Phase-Space Exploration model

One of the most difficult issues in the study of the heavy ions collisions around the Fermi energy domain (say between 20 and 100 AMeV). is related to the fact that this energy region is a transition region indeed one has to deal with a mean-field dominated dynamics (much below the Fermi energy) and a high-energy regime where individual nucleonic degrees of freedom and associated two-body effects become predominant [58]. This competition between mean-field effects and in-medium nucleon-nucleon interactions is a long-standing problem in heavy ion reactions around the Fermi energy and has led to two classes of models. The first one starts from the mean field and extends this latter to account perturbatively for the residual two-body interaction, while in the second model class, the two body interaction is treated exactly and mean-field effects play a secondary role. Intra-nuclear collision and molecular dynamics models

are the prototypes of this second class. For heavy-ion collisions, it has been shown that the transition between mean-field and nucleon-nucleon degrees of freedom is smooth and both should be accounted for at the same time to properly reproduce experimental data. Special attention should thus be paid to the interplay between preequilibrium and postequilibrium effects. In microscopic models, this can hardly be achieved because of the difficulty in properly defining excitation energies or thermal equilibrium conditions.

The Heavy-Ion Phase-Space Exploration model is a dynamical model to account for light as well as massive cluster emission during the reaction. This model naturally accounts for the transition between nucleonic and mean-field effects. It properly connects the preequilibrium stage with the deexcitation phase, introducing the important notion of phase-space exploration during the reaction. The price to pay in order to solve the problem discussed above is to disregard some microscopic effects and to work at a more phenomenological level. The problem very often arises from the fact that there are strong dynamical effects taking place at least in the first instants of the reaction. Let us take multifragmentation, defined as the emission in a very short time scale of several species of atomic number larger than 2 [59] as compared to other decay mechanisms such as the formation of heavy residues or fission. Such a phenomenon is expected to be the ideal tool to study the transition from a liquidlike state (nuclei at normal density) toward a gaslike state associated with the vaporization of the system. The quest for the signals of a nuclear phase transition of the liquid-gas type has led to rather sophisticated analysis. Such recent experimental analysis based on nuclear calorimetry have claimed evidence for a liquid-gas phase transition through the study of various signals. Some of these analyses make extensive use of the thermal multifragmentation statistical models to prove the existence of thermal equilibrium. There are however some uncertainties in using statistical models. This is due to the lack of knowledge of dynamical

effects, in particular, of the fast early processes which could lead to the formation of equilibrated systems. In particular, the phase space explored during the collision is expected to be sensitive to the initial conditions of the reaction. Such a point is addressed in microscopic transport models. These models provide a suitable framework for the description of nuclear collisions at intermediate energies and are able to describe dynamical effects. Unfortunately, although nucleon-nucleon collisions are included, one could not determine if the system has reached a complete thermal equilibrium. Moreover, there is not a direct link in such approaches between the outputs of the simulations and the thermodynamical properties of the excited species produced in the reaction. As a consequence, these models do not give unambiguously important quantities required for statistical model calculations. For instance, internal excitation energies of the created fragments cannot be easily obtained in current microscopic calculations.

The HIPSE model has been developed to find a link between the two extreme approaches, describe above, namely the statistical approach based on the reduction of the reaction to a few important parameters and the microscopic approach based on the transport theory.

2.2.1 Main characteristics of the HIPSE model

This model consists in three main steps. In the first step, namely approaching phase, the two partners (Projectile and Target) of the reactions are at maximum overlap. This phase is considered by solving the classical equation of motion of the two partners in their mutual interaction potential. The state of the system is described by the many-body wave function $|\Psi(t)\rangle$ at a given time. The natural generalization of equation $Q_{value} = M(A_P, Z_P)+M(A_T, Z_T)-M(A, Z)$ consists in defining the potential at a given time using:

$$V^*(t) = \langle H(t)\rangle - \langle H\rangle_\infty = \langle H(t)\rangle - M(A_T, Z_T) - M(A_P, Z_P) \quad (2.17)$$

where H is the many-body Hamiltonian and $\langle H(t)\rangle = \langle \Psi(t)\,|H|\,\Psi(t)\rangle$ while $\langle H(t)\rangle_\infty$ corresponds to the expectation value at infinite distance. The difficulty met in microscopic theories is that the potential part of the energy does not separate from the possible internal excitation or from the kinetic part in the collective space. The interaction potential $V(t)$ thus differs from $V^*(t)$ and cannot be accessed directly. The difficulty can be removed in two limiting approximations:

- **in the frozen density approximation**, it is supposed that the collision is sufficiently fast so that the internal degrees of freedom do not have time to "reorganize" themselves. In that case, the concept of di-nuclear system persists even in the limit of overlapping target and projectile densities. It thus neglects the Pauli principle as well as the saturation properties. The frozen density approximation limit is expected to be valid for high energy collision.

- **the adiabatic approximation** limit assumes in an opposite way that internal degrees of freedom reorganize much faster than the collective degrees of freedom. In that case, the notion of two separate densities looses its meaning and one should treat instead a single entity that undergoes a transition from large deformation toward a compact shape.

The interaction potential obtained either from microscopic or macroscopic theories generally uses one of the two approximations. One should however keep in mind that once the target and projectile come into contact, it is normally not possible to distinguish nucleons of the projectile from nucleons of the target, at least due to the fermionic nature of nucleons. It is however reasonable to assume that a memory of the entrance channel is kept during some time and that we can still (at least during the time of the neck formation) distinguish a target like or a projectile like nucleus. The "true" interaction potential will thus be in-between the two limiting cases.

For a given beam energy E_B, a classical two-body dynamics during the phase of approach of the collision is assumed. Noting $V_{A_T A_P}(r = |r_T - \mathbf{r}_P|)$ the interaction potential between the target and the projectile, we consider the evolution associated with the Hamiltonian $E_0 = p^2/2\mu + V_{A_T A_P}(r)$. $E_0 = [A_T/(A_T + A_P)]\, E_B$ is the available energy in the center of mass while $\mathbf{p}$ is the relative momentum and $\mu = m_T m_P/M$ is the reduced mass with m_T and m_P the target and projectile mass, respectively. The concept of nuclear potential is rather unambiguously defined when the two nuclei are well separated. Then it uses the proximity potential when the relevant observable is reduced to the minimal geometric information on the two nuclei in interaction (i.e. their nuclear radius only).

In order to derive the macroscopic theories, one should first consider the semi-classical version of (2.17):

$$V(t) = \int d^3 r \epsilon(\rho(t)) - \int d^3 r \epsilon(\rho_T) - \int d^3 r \epsilon(\rho_P) \tag{2.18}$$

where $\epsilon(\rho(t))$ is the energy functional which depends on the one-body density $\rho(t)$ and on the effective nucleon-nucleon interaction. In the sudden limit, the single-particle degrees of freedom are frozen. The total density at time t writes in terms of the density of the target ρ_T and of projectile ρ_P in their ground state as: $\rho(\mathbf{x}, t) = \rho_T(\mathbf{x} - \mathbf{r}_T(t)) + \rho_P(\mathbf{x} - \mathbf{r}_P(t))$ where $\mathbf{r}_T(t)$ and $\mathbf{r}_P(t)$ are the target and projectile center of mass, respectively. Reporting this expression in equation (2.18), we end with the doubly folded potential for $V(\mathbf{r})$

$$V(\mathbf{r}) = \int d^3 x d^3 y \rho_T(\mathbf{r}_T(t) - \mathbf{x}) \nu(|x - y|) \rho_P(\mathbf{y} - \mathbf{r}_P(t)) \tag{2.19}$$

where the interaction ν may contain any effective force in addition to the coulomb energy.

The expression (2.19) can be integrated in the case of spherical leptodermous systems, leading to a simple expression for the potential

[60, 61]:

$$V_{prox}(r) = e^2 \frac{Z_T Z_P}{r^2} + 4\pi \frac{C_T C_P}{C_T + C_P} \Phi(\xi) \tag{2.20}$$

where C_T and C_P are the half density radii of the two nuclei. In this expression, the function $\Phi(\xi)$ is a function that measures the interaction energy between two parrallel surface separated by a distance $\xi = r - C_T - C_P$. The proximity potential is not well suited for small relative distance $r \leq R_T + R_P$, where R_T and R_P are radii of, respectively, the target and the projectile. In this case, there is thus a need for another prescription defining nuclear potential between two strongly overlapping nuclei.

Such a potential is to a large extent unknown and should normally depend on different parameters describing the configurations of the system: shape, internal excitation energy, as well as the initial relative energy of the two partners [62]. At very hight relative energy, neglecting the influence of two-body collisions, we do expect that the internal degrees of freedom have no time to reorganize and that the system has a strong memory of the initial conditions. In view of these points, we do not expect a unique potential. As the beam energy increases, the internal degrees of freedom have less time to reorganize and the potential is expected to be sharper. The possible energy dependence of the potential has been included in a phenomenological way. In the following, we use a simple approximation for the construction of the potential. First, it is assumed that $V_{A_T A_P}$ depends on r uniquely even for small relative distances. In order to obtain the potential for $r < R_T + R_P$, we interpolate the potential between $r = 0$ and $r = R_T - R_P$ using a third-order polynomial and assuming continuity of the derivative of the potential at each point. The value retained at $r = 0$ is conveniently expressed as

$$V(r = 0) = \alpha_a V^{Froz}_{A_T A_P}(r = 0) \tag{2.21}$$

where α_a is a parameter to be fixed by comparison with experimental data. $V^{Froz}_{A_T A_P}(r = 0)$ is the energy of the system assuming that the

two densities of the system overlap completely in the frozen density approximation.

The second step in the model is the partition formation phase which corresponds to the rearrangement of the nucleons into several clusters and light particles (hereafter called the partition) according to the impact parameter of the reaction. The partition is built following coalescence rules in momentum and position spaces. The main consequence of this approximation is that the characteristics of the species produced in highly fragmented collisions will exhibit kinetic energy and angular distributions keeping a strong memory of the entrance channel.

The last phase is the exit channel and after-burner phase up to the detectors: the partition is propagated taking into account explicitly reaggregation effects due to the strong nuclear and Coulomb interactions among the various species of the partition. Since these latter are produced in excited states, secondary decays are taken into account by means of an evaporation code.

Chapter 3

Principal Component Analysis (PCA)

3.1 Basic concepts of PCA

PCA is one of the multivariate methods of analysis and has been used widely with large multidimensional data sets. The use of PCA allows the number of variables in a multivariate data set to be reduced, whilst retaining as much as possible the information present in the data set. This reduction is achieved by taking p variables $X_1, X_2, ..., X_p$ and finding the combinations of these to produce principal components (PCs) $PC_1, PC_2, ..., PC_p$, which are uncorrelated because they represent the eigenvectors of the covariance matrix. The lack of correlation is a useful property as it means that the PCs are measuring different "dimensions" in the data. PCs are ordered so that PC_1 exhibits the greatest amount of the information, PC_2 exhibits the second greatest amount of the information, PC_3 exhibits the third greatest amount of the information, and so on. That is $var(PC_1) \geq var(PC_2) \geq var(PC_3 \geq ... \geq var(PC_P)$, where $var(PC_i)$ expresses the variance of PC_i in the data set being considered. $Var(PC_i)$ is also called the eigenvalue of PC_i. When using PCA, it is hoped that the eigenvalues of most of the PCs will be so low as to be virtually negligible. When this is the case, the information in the data set can be adequately described by means of a few PCs where the eigenvalues are not negligible. Accordingly, some degree of economy is accomplished

as the information in the original number of variables (X variables) can be described using a smaller number of new variables (PCs).

3.2 PCA analysis technique

The analysis is performed on a data set of p variables $(X_1, X_2, ..., X_p)$ for n events. From this data set, a corresponding squared covariance or correlation matrix can be calculated. For the covariance matrix the following equation can be used:

$$cov(X_j, X_k) = \frac{\sum_{i=1}^{n}(X_{ij} - \bar{X}_j)(X_{ik} - \bar{X}_k)}{n}$$

where

$$\bar{X}_j = \frac{\sum_{i=1}^{n} X_{ij}}{n}, \ \ and \ j, k = 1, 2, ..., p.$$

The covariance matrix S then has the following form:

$$S = \begin{pmatrix} s_{11} & s_{12} & s_{13}... & s_{1p} \\ s_{21} & s_{22} & s_{23}... & s_{2p} \\ . & . & . & . \\ . & . & . & . \\ s_{p1} & s_{p2} & s_{p3}... & s_{pp} \end{pmatrix}$$

where s_{jk} $(j \neq k)$ is the covariance of variables X_j and X_k and the diagonal element s_{jj} is the variance of variable X_j.

The covariance matrix is used when the variables are measured on comparable scales. However, when variables have different units or widely different scales, as in the current study, a correlation matrix where variables are standardized should be used.

The first principal component (PC_1) is then a linear combination of the original variables $X_1, X_2, ..., X_p$:

$$PC_1 = a_{11}X_1 + a_{12}X_2 + a_{13}X_3 + ... + a_{1p}X_p = \sum_{j=1}^{p} a_{ij}X_j$$

where $a_{11}, a_{12}, ..., a_{1p}$ are the coefficients assigned to the original p variables for PC_1 and these coefficients are subject to the condition

$$a_{11}^2 + a_{12}^2 + a_{13}^2 + ... + a_{1p}^2 = 1$$

Similarly, the second principal component:

$$PC_2 = a_{12}X_1 + a_{22}X_2 + a_{23}X_3 + ... + a_{2p}X_p$$

and its coefficients:

$$a_{21}^2 + a_{22}^2 + a_{23}^2 + ...a_{2p}^2 = 1.$$

PC_2 and PC_1 are uncorrelated. The third component PC_3 can be expressed in similar way. There can be up to p principal components if there are p variables.
Other important properties of eigenvectors or PCs are:

- Eigenvectors can only be found for square matrices.
- Not all square matrices have eigenvectors, however a correlation (or covariance) matrix will always have eigenvectors (the same number as there are variables).
- Length does not affect whether a vector is a particular eigenvector, direction does.
- Eigenvectors and eigenvalues always come in pairs.

Any correlation between the dependent variable (outcome) and the PCs is captured by means of a regression model. The regression step can be done after the creation of the PCs and removal of some PCs that are believed to be not related or of little importance to the effects under study.
In PCA, the number of components extracted is equal to the number of variables being analysed. However, the components are chosen in sequence as the best descriptors of the data. In general, the last few

components do not account for much of the variance, and therefore can be ignored.

To better explain the previous concepts and to summarize, we can say that the event-variables in Principal Components Analysis [63] contain all the information (so called *principal variables*) since they are linear combinations of primary variables. Hence, the projection of the event-variable matrix on the plane (respectively space) defined by 2 (respectively 3) principal variables allows a global visualization of the features of the whole set of events.

The event-variable matrix X (n events, p variables) is equivalent to n point cloud (representing the events) plunged in a p dimensional space (Hilbert space). This space is connected to a normed reference frame, each vector of which is associated with a primary variable. The coordinates of the points in this reference frame are equal to the values taken by the variables for the corresponding events. The directions of the axes are chosen in such a way that the cosines of their relative angles are equal to the correlation coefficients connecting the two corresponding primary variables. In the following, we will deal with the shape of the cloud in the variable space. In order to handle comparable magnitudes and to get rid of the problem of units, each variable is first centered and reduced:

$$(\forall_i \in [1,n]),\ (\forall_j \in [1,p]),\ (x_{ij}^{cr} = \frac{x_{ij} - m_j}{\sigma_j}) \tag{3.1}$$

The cosine of the angle between any two axes is equal to the correlation coefficient of the two corresponding variables. Hence, if two variables are linearly correlated, their axes are superimposed. In this space, each event is represented by a point whose coordinates are equal to the values taken by the corresponding primary variables. The experimental data or the events generated by a model form a cloud. The covariance matrix of the global variables is built and diagonalized, to obtain the eigenvalues and the associated eigenvectors. The eigenvalues

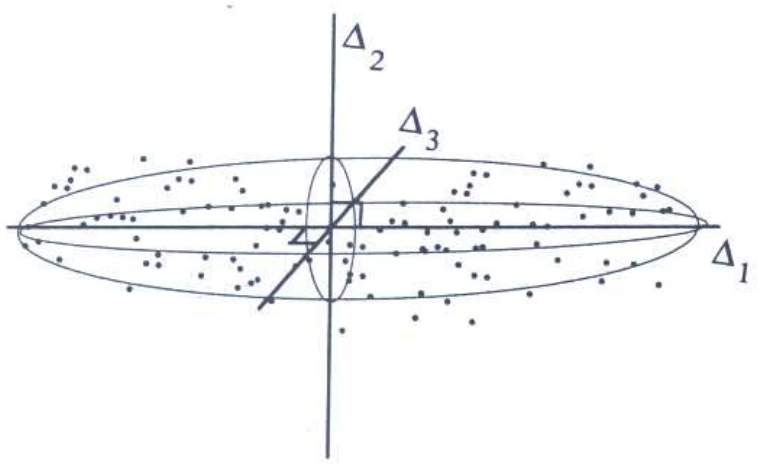

Figure 3.1: First(Δ_1), second(Δ_2) and third (Δ_3) principal axes of a cloud in dimension $p = 3$.

give the statistical information carried by the associated eigenvectors. When eigenvectors, linear combinations of the observable axes, are ordered with respect to the information they yield, principal components are defined. In the space defined by principal components, events having the same features (i.e., similar values of global variables) are grouped together, so defining in a natural way and without any previous selection, event classes. To better understand the previous concepts about PCA analysis we report below an example with the classical mechanical analogy.

3.3 Mechanical analogy

In three dimensions ($p = 3$), this cloud can be assimilated to a set of points having the same mass m (Fig. 3.1). Its centre of gravity g is localized at the origin. $g(g_1, g_2, g_3)$ is the point with respect to which the inertia is minimum:

$$(\forall_j \in [1,3]), \quad \left(\frac{\partial}{\partial g_j} \sum_{i=1}^{n} m \left(\frac{x_{ij} - m_j}{\sigma_j} - g_j \right)^2 = 0 \right)$$

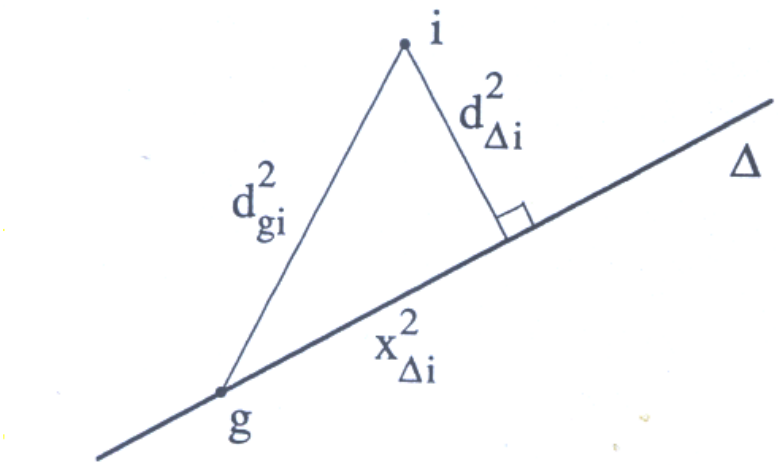

Figure 3.2: Decomposition of the total inertia.

$$(\forall_j \in [1,3]),\ \ (g_j = 0).$$

The inertia of this solid (set of points) is equivalent to that of its inertia ellipsoid whose principal axes are the three eigenvectors of its inertia tensor. The square roots of the corresponding eigenvalues give the length of each principal axis. The first principal axis is the one around which the inertia of the solid is minimum, the second one is the axis perpendicular to the first one around which the inertia is minimum, while the third is perpendicular to the first ones.

The sum of the inertia $I_{proj\Delta}$, of the point cloud projected on an axis Δ passing by the centre of gravity, and of the inertia I_Δ, of the cloud with respect to this axis, is always equal to the inertia with respect to g (Fig. 3.2):

$$\begin{aligned} I_g = \sum_{i=1}^{n} m d_{gi}^2 &= \sum_{i=1}^{n} m d_{\Delta i}^2 + \sum_{i=1}^{n} m x_{\Delta i}^2 \\ &= I_\Delta + I_{proj\Delta} \end{aligned} \tag{3.2}$$

The first principal axis is thus the axis on which the inertia of the projected cloud is maximum ($I_{proj\Delta}$ maximum when I_Δ minimum, see

Eq. (3.2)). In statistics, the projected inertia measures the quantity of information provided by a specified axis.

For example, if all the events were projected on the same point on a given axis (null projected inertia), then the knowledge of the projection of a given event on this axis will give no information about the event. The sum of the projected inertiae on the three principal axes is equal to the inertia with respect to g : $I_g = \sum_{j=1}^{3} I_{proj\Delta_j}$.

It is often more significant to give the proportion of inertia projected on the axis: $I_{proj\Delta}/I_g$.

The eigenvalue is equal to the inertia projected on the axis defined by eigenvector. All the properties defined above in the case $p = 3$ remain true irrespective of the number of variables. It can be noticed that there is an absolute identity between the notions of *mean value* and *centre of gravity* as well as between the notions of *variance* and *inertia* in the case when the mass of each point is equal to $1/n$.

In conclusion the Principal Components Analysis is a tool which is well suited for the first steps of the analysis of an experiment in nuclear physics, when one wishes to have a global vision of the observables. It allows for example the exploration of the multidimensional cloud of events to extract different classes. PCA permits also the handling of variables preserving as well as possible the input information.

In this work we have applied the PCA on the events generated with CMD model to discriminate between central, semi-peripheral and peripheral collisions.

Chapter 4

SVM analysis on CMD and HIPSE events.

In this chapter we will show the first results of SVM analysis on the events that were generated by CMD and HIPSE model.
The events generated by CMD and HIPSE for an impact parameter in the range $0.5 \div 10$ fm were analyzed with SVM before and after applying a software filter, which simulates the experimental response of each module of the apparatus.

4.1 Global Variables

A basic global variable is the event **charged particle multiplicity** (Nc). The multiplicity represents the number of particles emitted in each event and could be related to the violence of the collision (centrality of the collision). From the experimental point o view the total detected multiplicity depends on the angular coverage of the apparatus and don the kind of detectors of the apparatus. For the CHIMERA apparatus, described in chapter 6, a first estimation of the total charged particle multiplicity can be obtained by taking into account that particles stopped in the silicon detectors give ToF and Si signals, particles that punch through the silicon detectors give a Si and CsI(Tl) signals, and fast light charged particles, which don't produce a signal in the Si detectors, are

detected by the CsI(Tl) scintillators and identified by means of the fast-slow components. This "estimated" multiplicity N_{tot}, has the advantage that it can be easily built online (during the measurements).

Another global variable that characterizes the events is the **charge distribution** of the reaction products i.e. the relative yield of the different charges emitted in the reaction. The charge distribution can give qualitative information on the reaction mechanism.

An important quantity, related to the charge of the fragments, is the **sum of the detected charges** (Z_{TOT}), which gives information on the efficiency of the apparatus. In fact, as the sum of the charges in the entrance channel is known ($Z_{TOT} = Z_{Projectile} + Z_{Target}$), it is interesting to study the response of the apparatus with respect to the collection of the emitted charged particles, i.e. to the reconstruction of the event. In general, global variables reflect the efficiency of the apparatus so that particles with an energy lower than the detector thresholds or emitted in an angular polar range not in the apparatus geometrical acceptance (in our case is $1^\circ \leq \vartheta \leq 30^\circ$), cannot be included in the considered global variables (multiplicity, total charge, etc). Even though CHIMERA covers 94% of the total solid angle and shows very low detection thresholds not all the particles emitted in an event are detected.

Another particle multiplicity can be constructed taking into account only the **intermediate mass fragments** (IMF), i.e. fragments with a charge $Z \geq 3$.

A further global variable that can be built in order to get information on the centrality of the collision is the **transverse energy**. This is defined as the kinetic energy calculated considering only the component of the velocity perpendicular to the beam axis. The sharing of the incident kinetic energy of the projectiles in a perpendicular direction is frequently interpreted as an indication of the centrality, or dissipation, of the collision. In fact, the more central is the collision, the higher the transverse energy, inducing to define the transverse energy as an indicator

of the loss of the entrance channel memory. In addition, for low values, the transverse energy can be considered a linear function of the impact parameter, thus allowing an impact parameter selection.
The total transverse energy is thus defined as

$$E_{Trans} = \sum_{i=1}^{Ntot} \frac{(\vec{p_i} \cdot \vec{k})^2}{2m_i} = \sum_{i=1}^{Ntot} E_i sin^2\theta_i \tag{4.1}$$

where p_i, m_i, E_i represent respectively the momentum, the mass and the kinetic energy of a reaction product, θ_i the emission polar angle and k_i the beam direction. The sum is extended to all products emitted in each event.
The knowledge of the values of the charge of the particles and its multiplicity for each event allows to construct other observables that are correlated with the centrality of the reaction, such as, i.e. $0^{th}, 1^{st}, and\, 2^{nd}$ **moments of the charge distribution**

$$M_K = \frac{\sum_{i=1}^{Nc} Z_i^K}{Z_{big}} \tag{4.2}$$

where into the sum the contribution of the charge of the biggest fragment (Z_{big}) is not included. The moment of the second order for example provide us to measure the magnitude of fluctuations around the average value. Greater fluctuations were founded for central collisions rather than for peripheral collisions.
The **total impulse** along the beam axis (qz) and the **velocity of the biggest fragments** (V_{Zbig}) are other observables that are correlated with the impact parameter. For example qz, in the central collisions, has values around the velocity of the center of mass of the system, while for peripheral collisions qz is approximately close the velocity of the beam.
In this way also the velocity of the biggest fragment (V_{Zbig}) could be a good observable to discriminate the centrality of the reaction.

4.2 Observables in CMD and HIPSE analysis

For each event generated by CMD and HIPSE, some global observables are chosen as features, i.e., the charge particle multiplicity (Nc), the transverse energy (Et), the charge of the biggest fragment ($Zbig$), the number of intermediate mass fragments with $Z \geq 3$ $Nimf$, the $0^{th}, 1^{st}$, $and\, 2^{nd}$ moments of the charge distribution (M0,M1,M2), the total parallel momentum along the beam axis (qz), the total charge ($Ztot$) and the velocity of the biggest fragment ($VZbig$). The structure of the output events of the two models is shown in Tab. 4.1. For each event, a filter which simulates the experimental apparatus is also used and, as in the experimental case, only the events with $Nc \geq 2$ are selected for subsequent analysis. The dataset is divided in three different classes of events: central collisions (CEN) from 0.5 fm up to 3.5 fm, semi-peripheral collisions (SEM) from 4 fm up to 6.5 fm, and peripheral collisions (PER) from 7 fm up to 10 fm. Of course these sharp intervals are largely arbitrary, but we have used the same ones for all the data and predictions in this work. The experimental filter consists mainly on an angular filter in order to simulate the position of the detectors and in a energy filter in order to have the upper and lower limit in the detection thresholds.

The total nucleus-nucleus cross section is well approximated by the geometrical cross-section. As a result, the total reaction cross section can be calculated from the radii of heavy ions, namely, as $\sigma_g = \pi(R_t + R_p)^2$, where R_t and R_p are the equivalent hard-sphere radii of the target and projectile nuclei, respectively.

Therefore, (without the filter application) the event distribution generated by CMD and HIPSE follows a triangular distribution.

The total dataset generated with CMD consists of 109930 events and, since only the events with $Nc \geq 2$ have been taken into account, the total number of events for the analysis is 63571. 14656 events with b

Impact Par.	Features									
b	Nc	Et	Zbig	Nimf	M0	M1	M2	qz	Ztot	VZbig

Table 4.1: Generic structure of the data output of CMD and HIPSE for the reaction $^{58}Ni + ^{48}Ca$ at 25 AMeV.

between 0.5 and 3.5 fm are labelled as CEN events, 32412 events with b between 4.0 and 6.5 fm as SEM events, and 16503 events with b between 7.0 and 10.0 fm as PER events. For the HIPSE model we generated for the first class 14646 events, for the second class 32236 events and for the third class 17521 events, respectively CEN, SEM, PER ($Nc \geq 2$).

The **classification accuracy** is defined as the number of objects correctly classified over the total number of objects.

We have performed several tests on model events in order to understand if the learning phase was an *adaptable* process. The two models indeed show very different characteristics in the nuclear collisions simulation since they have different structure of the interaction potential. In addition, we would like to understand which is the best SVM classifier between the polynomial kernel with degree 1 (SVM-1), with degree 2 (SVM-2) and with degree 3 (SVM-3).

We started by experimenting with linear and quadratic classification but the best global accuracy has been obtained for the SVM-3 and in the following all the results refer to this classification.

We used four different approaches: first, we applied SVM-3 on CMD data, using some of these data for the training phase (50% of the total dataset for each class, i.e., 7323 events for central collisions, 16206 for semi-peripheral and 8251 for peripheral) and the remaining part in the test phase. SVM is then trained over the training set and tested over the test set. A polynomial kernel with $d = 3$ $\gamma = 1$ $r = 0$ is used (see Eq. 1.37). In this case we obtained, with a separation in 3-classes, a global accuracy of about 90% (see Fig. 4.1).

In the second case we applied the classifier on HIPSE model data (for

		Class predicted by SVM for filtered CMD events		
		CEN	SEM	PER
	CEN	91.9%	8.1%	0.0%
Actual Class	SEM	3.1%	90.9%	6.0%
	PER	0.0%	13.7%	86.3%

Table 4.2: SVM classification: confusion matrix for filtered CMD events.

		Class predicted by SVM for filtered HIPSE events		
		CEN	SEM	PER
	CEN	85.2%	14.7%	0.0%
Actual Class	SEM	12.0%	77.4%	10.6%
	PER	0.0%	16.0%	84.0%

Table 4.3: SVM classification: confusion matrix for filtered HIPSE events.

		Class predicted by SVM for filtered CMD and HIPSE events (Learning on CMD and test on HIPSE events).		
		CEN	SEM	PER
	CEN	46.3%	53.7%	0.0%
Actual Class	SEM	8.6%	82.3%	9.0%
	PER	0.0%	25.6%	74.2%

Table 4.4: SVM classification: confusion matrix for filtered CMD and HIPSE events. Learning on CMD and test on HIPSE events.

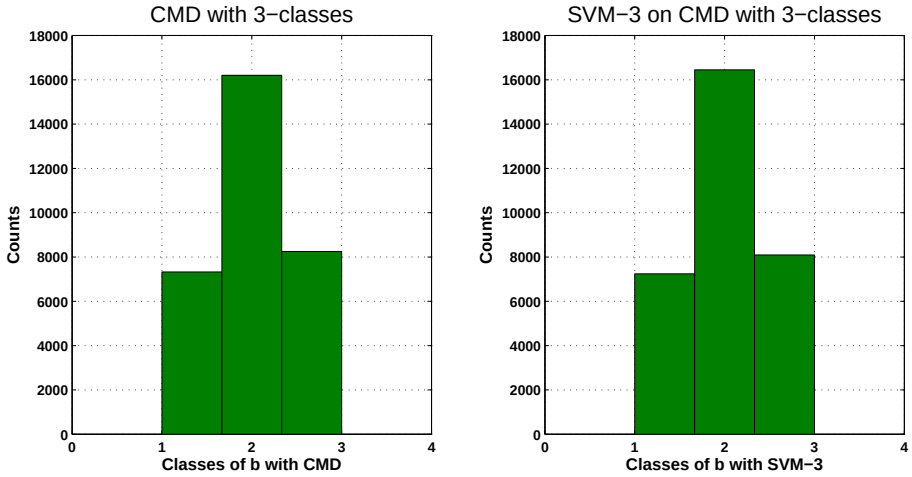

Figure 4.1: Classification of b in three different classes through the CMD model. Global accuracy of about 90%.

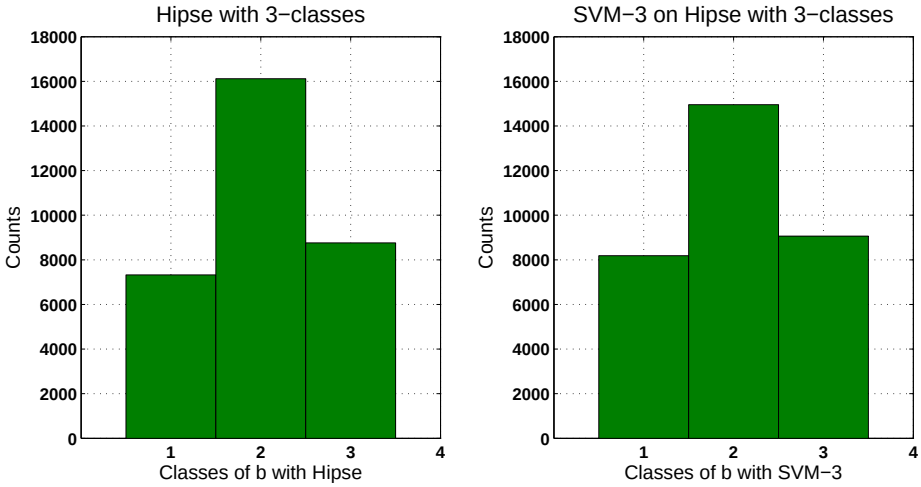

Figure 4.2: Classification of b in three different classes through the HIPSE model. Global accuracy of about 81%.

the training phase we used the 50% of the data for each class) in the same way as CMD and we had a global accuracy of 80.9% (see Fig. 4.2), slightly less than CMD. This could depend on the different structure of the interaction potential in the two models.

We have also developed the learning phase of SVM-3 on CMD data, by using all the dataset available, testing the results on HIPSE data obtaining a global accuracy of about 72% (see Fig. 4.3).

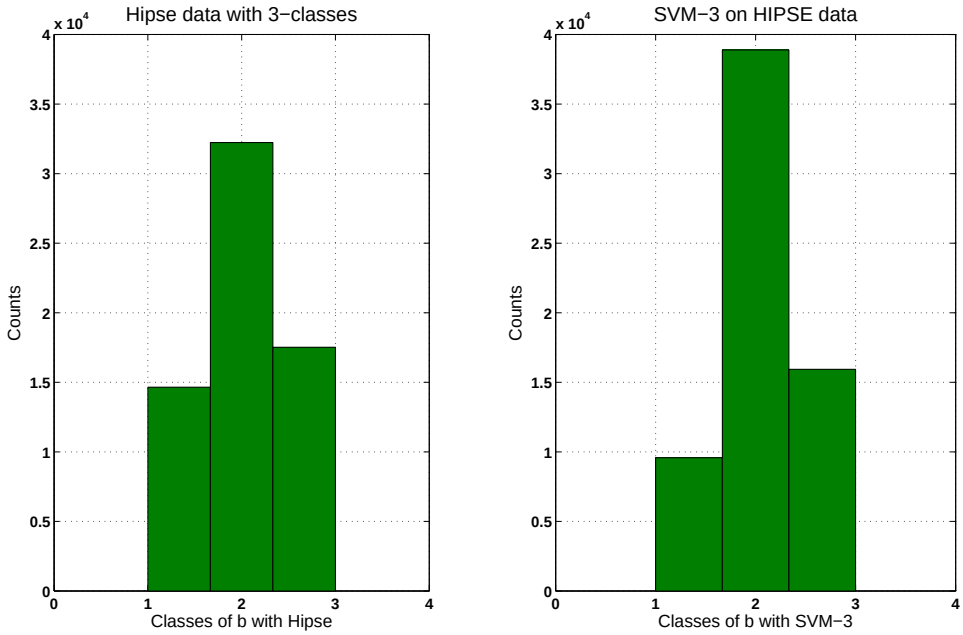

Figure 4.3: Classification of b in three different classes. Learning on CMD and test on HIPSE data. Global accuracy of about 72%.

		Class predicted by SVM for filtered HIPSE and CMD events (Learning on HIPSE and test on CMD events).		
		CEN	SEM	PER
	CEN	95.3%	4.7%	0.0%
Actual Class	SEM	22.9%	67.1%	10.0%
	PER	0.0%	15.0%	84.4%

Table 4.5: SVM classification: confusion matrix for filtered HIPSE and CMD events. Learning on HIPSE and test on CMD events.

The last result for this first attempt has been obtained using for the learning phase all the dataset available of HIPSE, and testing the results on CMD data output, obtaining a global accuracy of about 78% (see Fig. 4.4).

For more details, the correspondence between the class predicted by SVM and the actual one is reported separately for each single class in the confusion matrix. In table 4.2 it is shown the confusion matrix for the

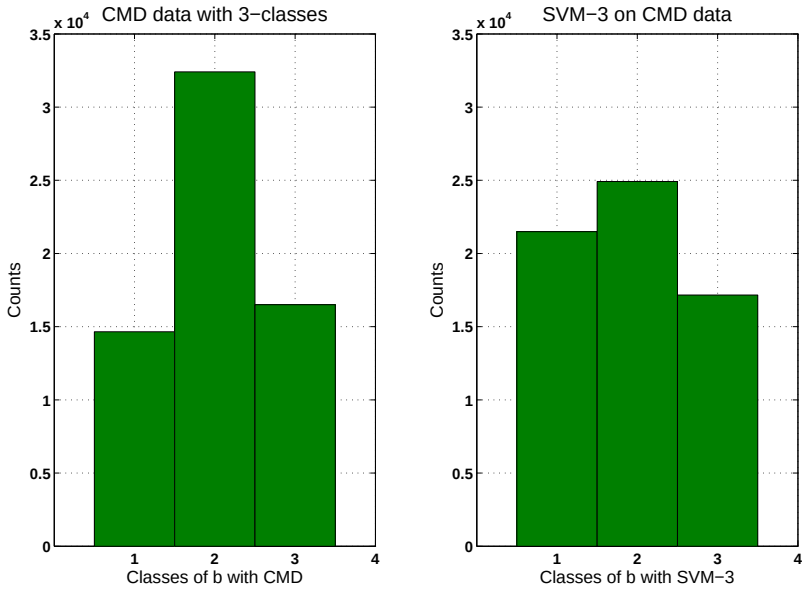

Figure 4.4: Classification of b in three different classes. Learning on HIPSE and test on CMD data. Global accuracy of about 78%.

first case shown in Fig. 4.1. By looking at the first row, it emerges that the actual CEN events are correctly classified as CEN events up to 91.9% of the times, whereas they are misclassified as SEM events up to 8.1% of the times. Similary, the actual SEM events are correctly classified as SEM events up to 90.9% of the times, whereas they are misclassified as CEN events up to 3.1% of the time and as PER events up to 6.0% of the times. Finally, the actual PER events are correctly classified as PER events up to 86.3% of the times, whereas they are misclassified as SEM events up to 13.7% of the times. In the same way, we show in tables (4.3), (4.4), (4.5) the confusion matrix for the three remaining cases described above.

In conclusions, we can say that, in this first approach, the pattern recognition analysis is a good and new techniques to extract the impact parameter value from the CMD (HIPSE) model data by using many features at the same time. With respect to traditional methods, this technique also works well when used to discriminate central collisions.

Indeed, if we analyze the first class for each case taken into account in this work, we can see that central collisions are correctly classified with an accuracy of about 90%. Only in the case shown in table (4.4) we have a low classification accuracy for the first class of about 47%. Furthermore, by means of the training phase that is an adaptable process, it will be possible to apply it to the experimental data measured by NUCL-EX collaboration. This part of analysis is discussed in the paragraph 6.3.

Chapter 5

Comparison between the SVM results and $\hat{b}$ and PCA techniques on CMD events.

The second part of our work is described in this and next chapter. Since we have previously shown that the classification procedure works better with CMD predictions, we will neglect in the following the results obtained through HIPSE predictions. In the next section we show the results obtained with the *Recursive Features Elimination* analysis (see section 1.5) on the CMD events. This procedure is very important to understand which are the features that are more correlated with the impact parameter.

5.1 Results of the SVM-RFE method applied to CMD filtered events

To evaluate the relevance of each feature in the classification problem discussed so far, SVM-RFE is applied. To this purpose, SVM is first trained with the whole bunch of 10 features, achieving 90% of global accuracy. Here, the transverse energy *Et* turns out to be the feature responsible for the lowest variation in the classifier's cost function J (see Eq. 1.40), thus the less important in the solution of the classification

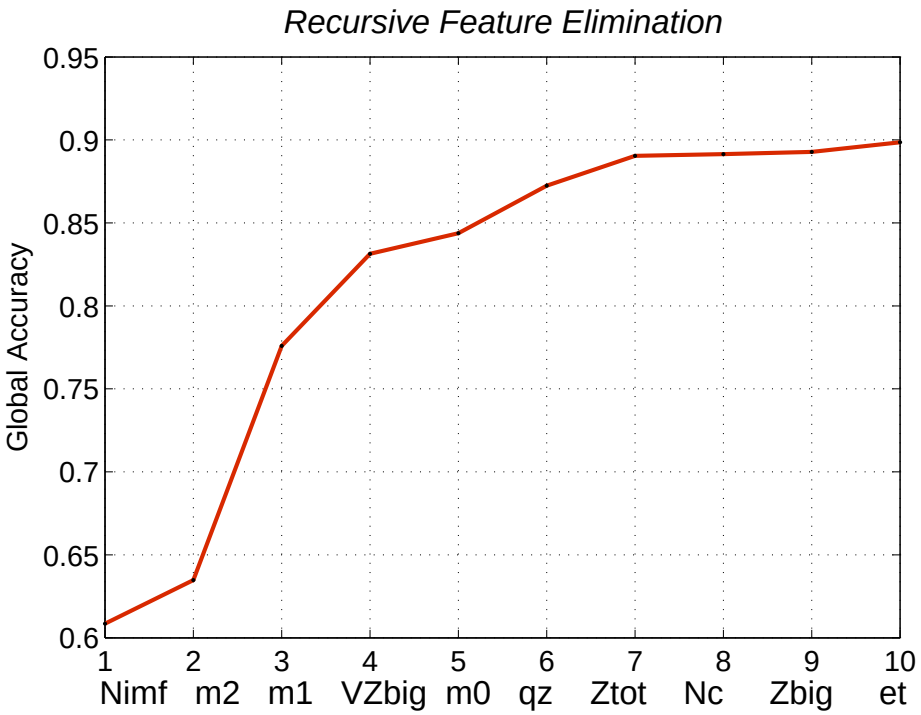

Figure 5.1: SVM-RFE applied to CMD events

problem. SVM is therefore trained over the training set with a bunch of just 9 features (i.e., all but Et), then tested over the test set where it achieves 89.1% of global accuracy. Here, the feature responsible for the lowest change in classifier's cost function J is the total charge $Ztot$. SVM is thus trained once again over the training set with a bunch of just 8 features (i.e., all but Et and $Ztot$), then tested over the test set where it reaches 88.4% of global accuracy. The procedure goes on in a similar way eliminating recursively $Zbig$, Nc, qz, $M0$, $VZbig$, $M1$, $M2$, and $Nimf$, see Fig. 5.1. From the results of SVM-RFE it is possible to see that the whole amount of the features taken into account are important to classify the classes because each observable gives non-negligible contribution to the global accuracy of classification (see Fig. 5.1).

5.2 Classification of CMD events by b estimator

As already pointed out the impact parameter represents the distance between the center of the target nucleus and the flight axis of the projectile nucleus. Since projectile and target cannot be detected before the collision, an evaluation from the final state is necessary to classify the events by the impact parameter. We report here the results obtained by evaluating the impact parameter through only one observable i.e. the charged particle multiplicity. Indeed one of the experimental methods often used to classify the centrality of the impact considers exclusively the correlation between a single observable and the "violence" of the collision. The most used variable is the total multiplicity of charged particles in an event. In other terms, the impact parameter is estimated by means of the following estimate function:

$$\hat{b} = \frac{b(X)}{b_{max}} = \sqrt{\int_X^\infty \frac{dP(Y)}{dY} dY}, \tag{5.1}$$

where P is the distribution of the observable (in this case the multiplicity Nc) proportional to the cross-section and b_{max} is the maximum impact parameter of the considered reaction.

The more central the collision occurs, the more "violent" it is, and more outgoing particles are produced. In average, a decreasing impact parameter leads to an increase of the charged particle multiplicity (Nc) (see Fig. 5.2). Thus, central collisions correspond to the highest multiplicity ones. The events with the highest multiplicity correspond approximately to selecting collisions with small values of impact parameter. The mean value of the multiplicity in the final state, strongly correlated with the impact parameter b, decreases monotonously as a function of b.

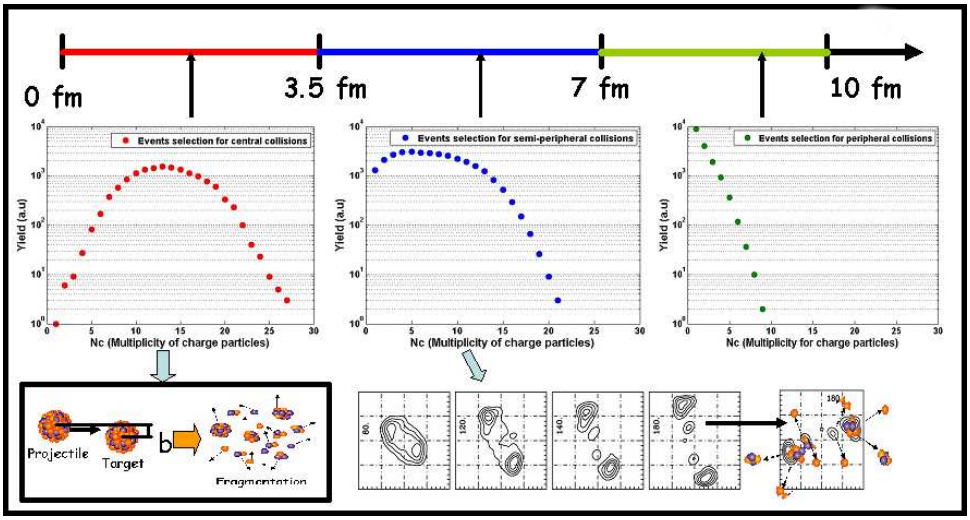

Figure 5.2: Behavior of the multiplicity as a function of the impact parameter b as calculated within the CMD.

5.2.1 Results of the estimator of $\hat{b}$ applied to CMD events

The global accuracy reached over the test set using the estimated value $\hat{b} = b/b_{max}$ approach is of about 61.4% whereas the related confusion matrix is reported in Tab. 5.1. To have an idea on the accuracy of the SVM and $\hat{b}$ approaches in the evaluation of physical quantities associated to the collision, the charge distribution is a meaningful example.

The charge distribution of each event is given by the occurrences per

		Class predicted by estimated value of $\hat{b} = b/b_{max}$ for CMD with filter		
		CEN	SEM	PER
	CEN	58.5%	41.3%	0.1%
Actual Class	SEM	0.7%	43.2%	56.0%
	PER	0.0%	0.0%	100%

Table 5.1: Estimate value of $\hat{b} = b/b_{max}$ classification for filtered CMD events: confusion matrix.

event of the different atomic numbers $Z = 1, 2, ..., etc.$, for the whole

amount of fragments produced during the collision. Taking as reference the charge distribution of the actual CEN, SEM, and PER events separately, the charge distribution of the SVM-classified and estimate function of b - classified CEN, SEM, and PER events are compared in Fig.5.3. By summing up the total number of counts for each different charge distribution, it turns out that the sum for the actual CEN events is 7156, for the actual SEM events 16212, and for the actual PER events 8253. As far as the correctly SVM-classified events are concerned the sum for the events classified as CEN events is 6590, for the events classified as SEM events is 14654, and for the events classified as PER events is 7123. The events correctly estimated by the value of these sums are 4193, 7004 and 8248, respectively.

Looking both at the global accuracies and confusion matrices reported

		Class predicted by estimated value of $\hat{b} = b/b_{max}$ for CMD without filter		
		CEN	SEM	PER
	CEN	75.6%	9.0%	15.2%
Actual Class	SEM	2.8%	80.0%	16.5%
	PER	0.0%	3.2%	96.7%

Table 5.2: Estimate value of $\hat{b} = b/b_{max}$ classification for unfiltered CMD events: confusion matrix.

		Class predicted by SVM for CMD without filter		
		CEN	SEM	PER
	CEN	84.7%	15.3%	0.0%
Actual Class	SEM	10%	80.4%	9.6%
	PER	0.0%	3.0%	97%

Table 5.3: SVM classification: confusion matrix for CMD without filter.

in the tables (4.2), (5.3), (5.1), (5.2) for the CMD filtered and unfiltered

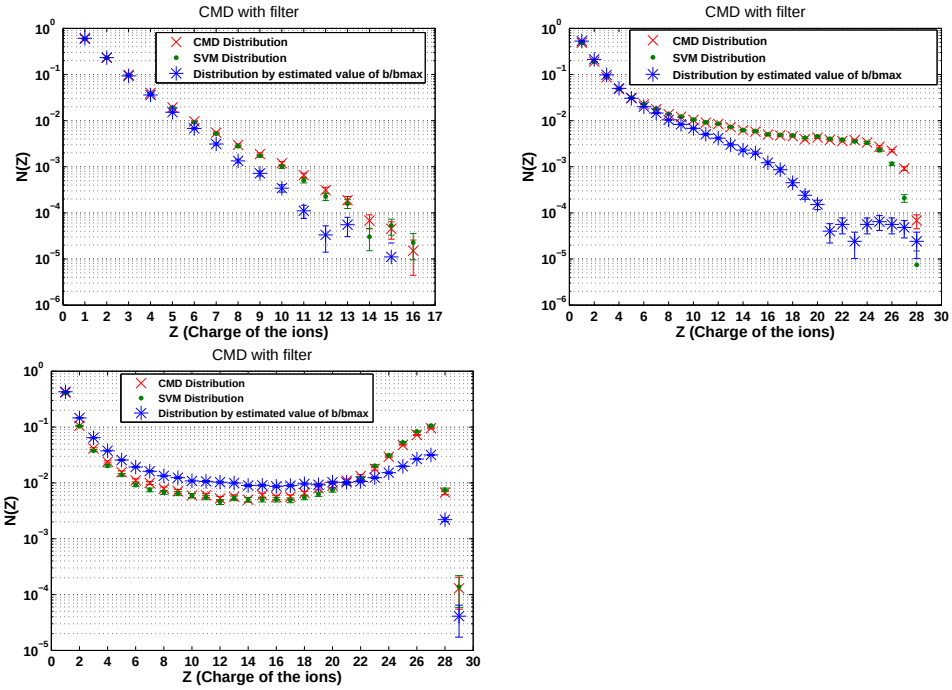

Figure 5.3: SVM versus estimated value of $\hat{b} = b/b_{max}$ classification on CMD filtered events: charge distribution of CEN events (upper part), SEM events (middle) and PER events (lower part). The number of ions has been normalized on the total number of ions products.

events, SVM achieves better results than the method based on the total multiplicity. As it has been shown in the Fig. 5.3, the SVM charge distribution has the same behaviour of the distribution generated by CMD model.

5.3 Classification of CMD events by PCA analysis

As pointed out in chapter 3, the goal of PCA is to condense information onto a minimum number of variables which are linear combinations of primary variables. Hence, the projection of the event-variable matrix on the plane (respectively space) defined by 2 (respectively 3) principal variables allows a global visualization of the features of the whole set of events. We analyzed through the PCA the same observables that are used in SVM and $\hat{b}$ analysis. The events are plugged into a n-dimensional

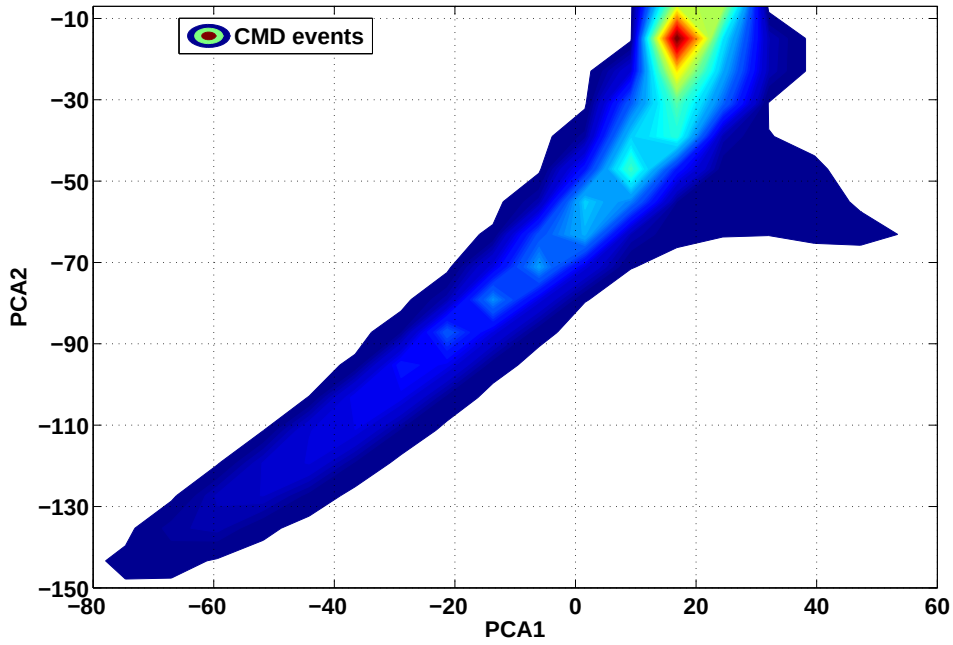

Figure 5.4: Scatter plot of the CMD filtered events in the plane PCA1-PCA2.

Hilbert space, where each axis of the basis corresponds to an observable variable. As in SVM and $\hat{b}$ analysis, each variable X is firstly standardized (centered and reduced), i.e., replaced by $X' = (X - \langle X \rangle + 2\sigma(X))/4\sigma(X)$, where $\langle X \rangle$ is the mean value of X and $\sigma(X)$ its standard deviation. The cosine of the angle between any two axes is equal to the correlation coefficient of the two corresponding variables. Hence, if two variables are linearly correlated, their axes are superimposed. In this space, each event is represented by a point whose coordinates are equal to the values taken by the corresponding primary variables. The experimental events form a cloud. The covariance matrix of the global variables is built and diagonalized, to obtain the eigenvalues and the associated eigenvectors. The eigenvalues give the statistical information carried by the associated eigenvectors. When eigenvectors, linear combinations of the observable axes, are ordered with respect to the information they yield, principal components are defined. In the space defined by principal components,

events having the same features (i.e., similar values of global variables) are grouped together, so defining in a model-independent way and without any previous selection, event classes. We found that about 90% of the original information contained in global observables is retained by the first three principal components. In particular, the percentage of the information carried by PCA1, PCA1+PCA2 and PCA1+PCA2+PCA3 is 73%, 84% and 90%, respectively. Therefore, we analyzed CMD events in the 3-dimensional space defined by the three principal components, where the CMD events exhaust most of the information. In Fig. 5.4 we show the measured events projected on the PCA1-PCA2 plane. The impact parameter (not used in the PCA) was then used to check the sorting of events by the principal components and thus the capability of PCA to provide information on the centrality of the reaction was tested.

5.3.1 Results of PCA analysis

In Fig. 5.5 we show the correlation between PCA1 and the impact parameter, after (top panel) and before (bottom panel) the software replica of the apparatus. In both cases the correlation is nearly linear.
First, we have checked the linearity versus b to understand if the PCA1 was a good observable to discriminate class of events and then we used the PCA1 to obtain the value of the impact parameter in the same way used for $\hat{b}$. In other words, we have applied the estimate function (eq. 5.1) of b using the PCA1 in place of the total multiplicity. In this way, we have obtained the confusions matrix for CMD events without and with the filter of the apparatus.

Through the PCA analysis we have obtained a global accuracy of about 64.8%, as shown in Tab. 5.4, for the CMD filtered events. For the central collisions, the events estimated correctly are about 67.1%. The PCA analysis achieves better results than $\hat{b}$ for the central events (see Tab. 5.1 and Tab. 5.4), but also in this case the SVM classification appears the best method with respect to PCA analysis and $\hat{b}$ method. Only in CMD

		Class predicted by estimated value of b for CMD with filter in PCA analysis		
		CEN	SEM	PER
	CEN	67.1%	32.8%	0.0%
Actual Class	SEM	0.2%	47.4%	52.3%
	PER	0.0%	2.8%	97.1%

Table 5.4: Estimate value of b classification in PCA analysis for filtered CMD events : confusion matrix.

		Class predicted by estimated value of b for CMD without filter in PCA analysis		
		CEN	SEM	PER
	CEN	85.6%	1.9%	12.4%
Actual Class	SEM	1.3%	80%	18.6%
	PER	0.0%	2.4%	97.5%

Table 5.5: Estimate value of b classification in PCA analysis for unfiltered CMD events: confusion matrix.

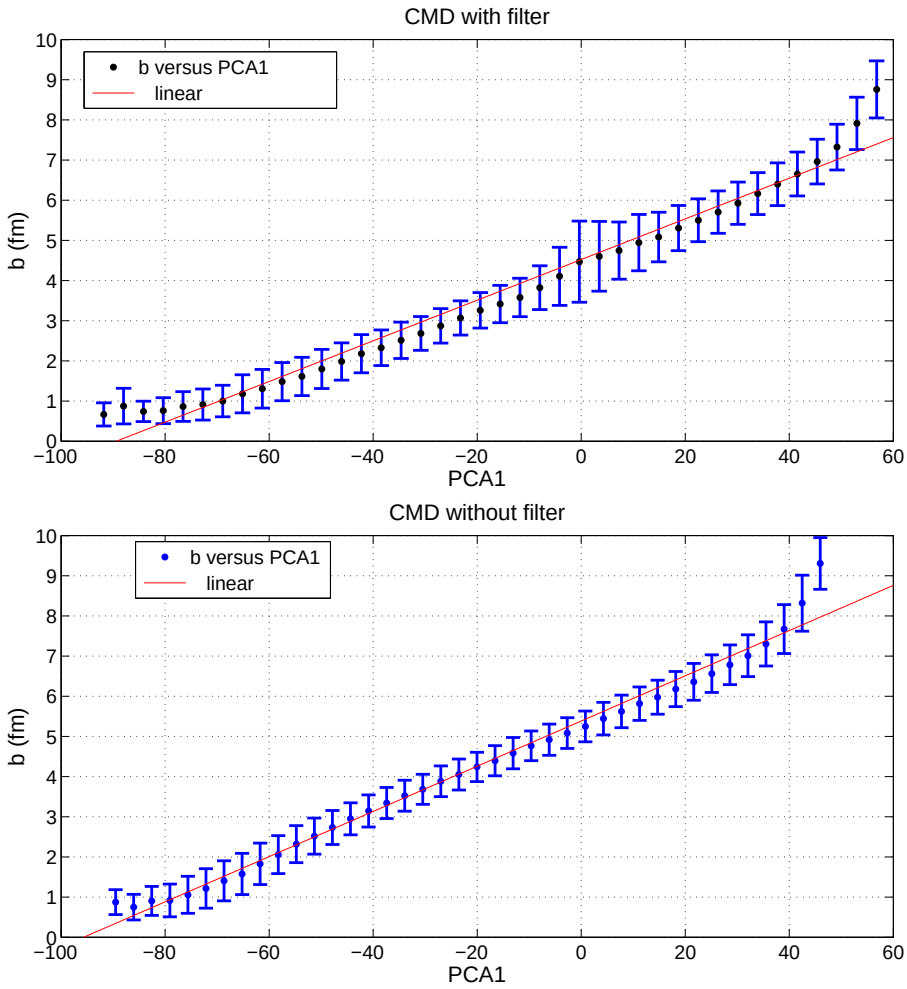

Figure 5.5: Average value of the impact parameter $b \pm$ its standard deviation in bins of PCA1. Top (bottom) panel corresponds to CMD events after (before) the filter. The red line is the linear fit of the correlation between b and PCA1.

events without filter SVM and PCA analysis achieve approximately the same results, as it is possible to see in Tab. 5.3 and Tab. 5.5, respectively.

Chapter 6

SVM applied on the experimental data measured by CHIMERA apparatus for the reaction ^{58}Ni $+^{48}$ Ca.

6.1 CHIMERA apparatus

CHIMERA apparatus [70] is a 4π multi-element detector array designed to study nuclear multifragmentation in heavy ion collisions in the Fermi energy domain ($20AMeV < E/A < 100AMeV$). The detector is composed of 1192 telescopes arranged in a cylindrical geometry around the beam axis, covering the polar laboratory angles between 1° and 176°. Each telescope is composed by a silicon detector, as first step, and a cesium iodide thallium activated scintillation detector. The apparatus consists of 35 rings. The forward 18 rings are assembled in 9 wheels covering the polar laboratory angles between 1° and 30°. Each wheel is divided in the azimuthal angle into 16, 24, 32, 40 or 48 trapezoidal cells, depending on its polar coordinate. The telescopes are placed at variable distances from the target in a range between 350 cm, for the telescopes belonging to the first ring (1°), and 100 cm, for the eighteenth ring's telescopes placed at 30°. The remaining 17 rings, covering the angular range 30°-176°, are assembled in a sphere 40 cm in radius; 15 rings are segmented into 32 cells, while the more backward rings are divided into

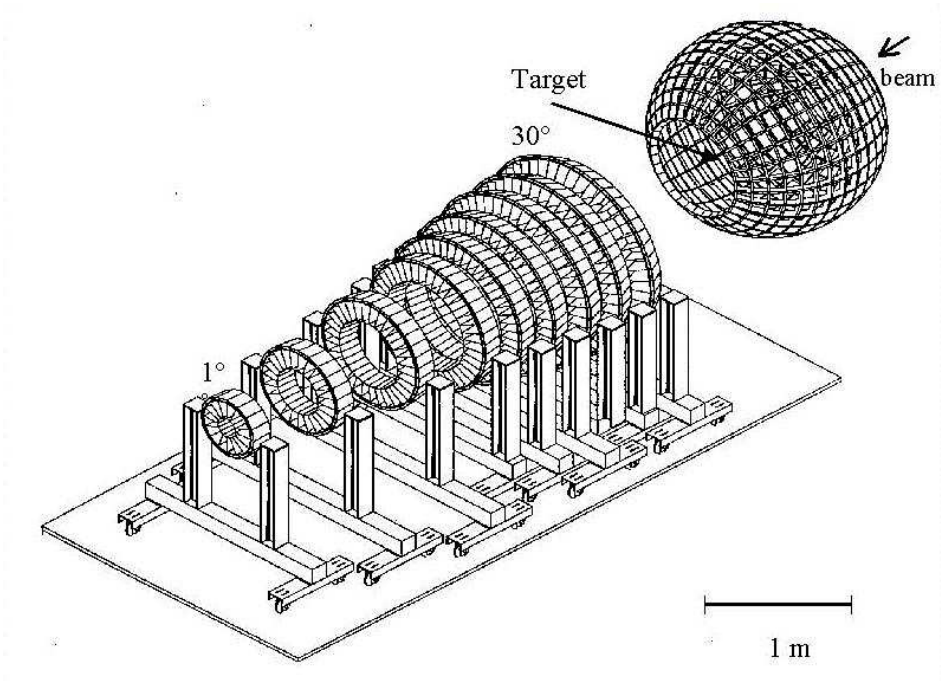

Figure 6.1: CHIMERA mechanical structure.

16 and 8 cells. Considering the beam entrance and outgoing holes, the frames of the detectors and of the target, an average laboratory solid angle coverage of about 94% can be obtained

CHIMERA, as most of other experimental devices for charged particles, does not measure neutrons. The features characterizing the events are estimated from masses charges, laboratory kinetic energies and detection angles of charged ejectiles.

6.1.1 Detection techniques

An elementary detection module of CHIMERA consists of a silicon detector followed by a CsI(TI) crystal, coupled to a photodiode. The thickness of silicon detectors ranges from $260\,\mu m$ to $300\,\mu m$, except for the first wheel where $220\,\mu m$ silicon detectors have been mounted. The thickness of the crystals varies from 12 cm to 3 cm, with the increasing polar angle, in order to stop the more energetic particles in the whole

dynamical range.

The shape and dimensions of CHIMERA apparatus enable to exploit a systematic Time of Flight (TOF) technique using as stop the Radio Frequency (RF) signal delivered by Super Conductor Cyclotron (SCC) and as start the signal generated by silicon detectors.

TOF technique allows mass identification of the fragments stopped in the first stage of telescopes and velocity measurements for all detected fragments. In fact, for particles stopped in the Silicon detector, experimental quantities, such as velocity, kinetic energy, polar and azimuthal angles are measured, and mass measurements can be extracted. Moreover, for particles stopped in the second step of the telescope, $\Delta E - E$ technique can be used for charge and mass identification. An energy threshold, ranging from 6 MeV/nucleon (^7Li) to 12 MeV/nucleon (^{16}O), calculated imposing that the particle passes through 280 μm silicon detectors, is obtained. The features of the detection system, as obtained in the first campaign of experiments, will be described in the next paragraph.

6.1.2 Silicon detectors

The first step of CHIMERA telescopes is constituted by a 280μm (average value) silicon detector obtained by the planar technology. This technique allows to have well defined detector thickness, very sharp active zones, extremely thin (500 $\mathring{A}$) and homogeneous junction thickness.

In order to ensure a good electrical contact the front and rear faces of the silicon slice are covered by a 300 $\mathring{A}$ Aluminium foil. Although this feature slightly worsens the energy resolution, due to the introduction of a dead layer, the rise time becomes nearly independent of the impact point of the detected particles giving better overall timing performances.

The resistivity of the detectors ranges between 3000 and 5000 Ωcm, while their capacitance varies from 500 to 2200 pF.

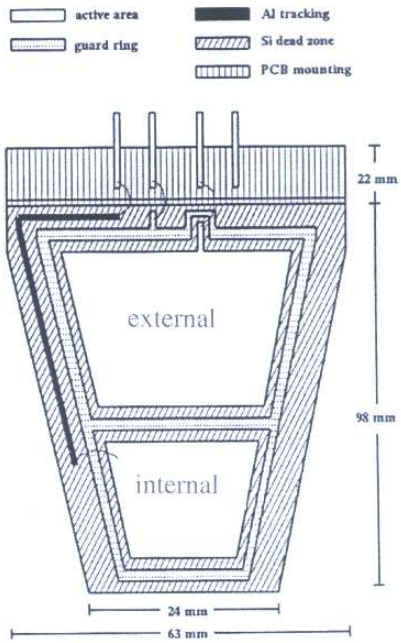

Figure 6.2: A schematic view of the two PAD silicon detector.

In order to minimize the dead layer of silicon detectors and to maximize the detection solid angle, the active zones of the internal and external detectors of each wheel are extracted from a single 6 inch silicon slice. They are realized on a trapezoidal shape bulk, implanting two windows separated by $300\,\mu m$ on the common n^+ contact. A schematic view of the two PAD detector is shown in Fig. 6.2. A $500\mu m$ dead layer coming from passivation along the external border of the detectors contains a guard ring which surrounds the two pads at distance of $50\mu m$ from the edges of the active area. A $50\mu m$ wide aluminium strip is deposited on the passivated zone in order to electrically connect the smaller pad. The silicon bulk is glued along the larger basis of the trapezoid to a PCB support which houses four contacts: two for the connections of the two pads, one for guard ring connection and one for the n^+ common contact. The PCB support is used also to connect attach the detector to the mechanical structure, reducing the dead detection zones only to the passivated area. The guard ring has the property to

restrict the electric field inside the detector, avoiding the effect caused by a non-complete depletion of the zone close to the border [64].
The two-pad design can be applied only to the detectors belonging to the first nine wheels. In fact, in order to minimize the shadow zone on the other detectors, the detectors of the sphere present a single active trapezoidal zone. The corners house the signals for the active zone, the guard ring and the ground contact, and they are used to fix the detector to the support box.

6.1.3 CsI(TI) detectors

The second step of the CHIMERA telescopes consists of a CsI(TI) crystal. The density of this crystal ($4.51\,g/cm^3$) allows full absorption of light charged particles over a short distance, in comparison to other scintillation detectors ($2-4g/cm^3$).
The shape of the crystal is a truncated pyramid with a trapezoidal base. The dimensions of the front surface are the same of the silicon detectors. The backward surface of CsI(TI) are bigger than the front one, depending on the thickness of the crystal.

RING	Theta	Thickness(cm)	$E_{max}^{proton}(MeV)$
1-16	1°-24°	12	190
17-18	24°-30°	10	160
19-22	30°-62°	8	140
23-25	62°-86°	6	120
26-28	86°-110°	5	95
29-32	110°-142°	4	95
33-35	142°-176°	3	90

Table 6.1: CsI(TI) thickness for different rings; E_{max}^{proton} represents the maximum energy of a proton stopped in the detector at 1.5-2 cm before the photodiode (in order to assure an energy measurement independent of the particle impact point.

The thickness of the crystals, chosen considering the requirement to

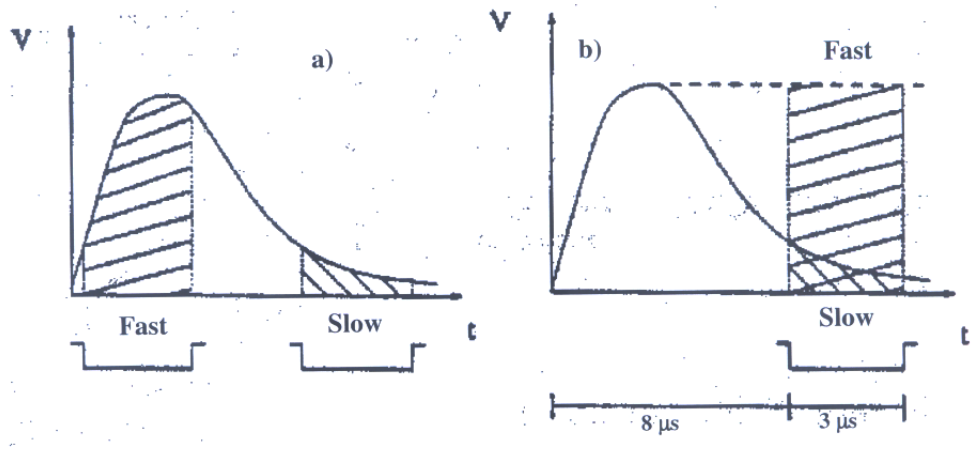

Figure 6.3: Integration of the Fast and Slow components of a CsI(TI) detectors. a) the two gates method; b) one common gate method.

stop the most penetrating particles like protons, is shown in Tab. 6.1 for each ring. The front and back faces of the crystal are polished while the sides are sanded. The front surface is covered with a $2\mu m$ reflecting foil of aluminised Mylar, while the other sides (lateral and rear) are wrapped in a $150\mu m$ thick Teflon layer and coated with a $50\mu m$ thick aluminum foil in order to optimise the light collection throughout the crystals.
The light output of CsI(TI) is characterized by a combination of two exponential functions with different time constants:

$$L(t) = L_1 \exp(-t/\tau_1) + L_2 \exp(-t/\tau_2)$$

where L(t) is the light pulse amplitude at time t and L_1 and L_2 are the light amplitudes for the fast and slow components, which have decay time constants $0.4 \leq \tau_1 \leq 0.7\,\mu s$ and $\tau_2 \approx 3.2\mu s$ respectively [65]. The fast and slow components depend, in a different way, on the ionization density and, therefore, on the mass, charge and energy of the detected particle. This dependence permits particle discrimination by pulse shape analysis. The used discrimination technique consists of a two gates method [66] adapted for photodiode readout [67]. The L(t) amplitude, integrated by the photodiode, passes throughout the amplifier that forms the signal with a time constant of about $1 \div 2\,\mu s$. The information relative to the

fast component L_1 is thus maintained unchanged, while the information relative to the slow component L_2, characterized by a longer time than the amplifier formation time, is cut off and influences mainly the tail of the signal.

Integrating two different parts of the signal produced by the CsI(TI) amplifier, by means of two gates, as shown in Fig. 6.3 , it is possible to obtain two signals proportional to the fast and slow components. An isotopic identification is then achieved by plotting the fast versus the slow integrated component. The isotopic discrimination works for light charged particles with charge $Z \leq 3$ as the dependence of the CsI(TI) components from the ionization density saturates for $Z = 4$. In order to work with a common gate integration system and to avoid introducing additional delays, the amplifier output signal is splitted: one signal is stretched when it reaches its maximum amplitude while the other remains unchanged. It is, thus, possible to use a single common gate (Fig. 6.3 b)) of $3\mu s$ that integrates the tail of the signal, proportional to the slow component, and a part of the stretched signal, proportional to the fast component.

The isotopic identification threshold, obtained by triggering the gates with the time signal of the CsI(TI), for $Z = 1$, allowing separation of protons, deuterons and tritons, range from 20 to 25 MeV, while a discrimination between $Z = 1$ and $Z = 2$ particles is obtained for proton energies higher than 10 MeV. The use of Radio Frequency as reference time has produced several advantages. One of them is to grossly reduce the identification thresholds, which resulted of the order of 20 MeV, for the purpose of clearly separating p, d, t, and of 4-5 MeV proton equivalent to separate Z=1 and Z=2. In addition, an identification threshold lower than 2 MeV proton equivalent has been obtained to discriminate charged particles from γ-rays. Considering the large size of the crystals used in CHIMERA, these results, obtained under beam conditions, are rather good as compared with the best ones reported in literature [68, 69]. In any

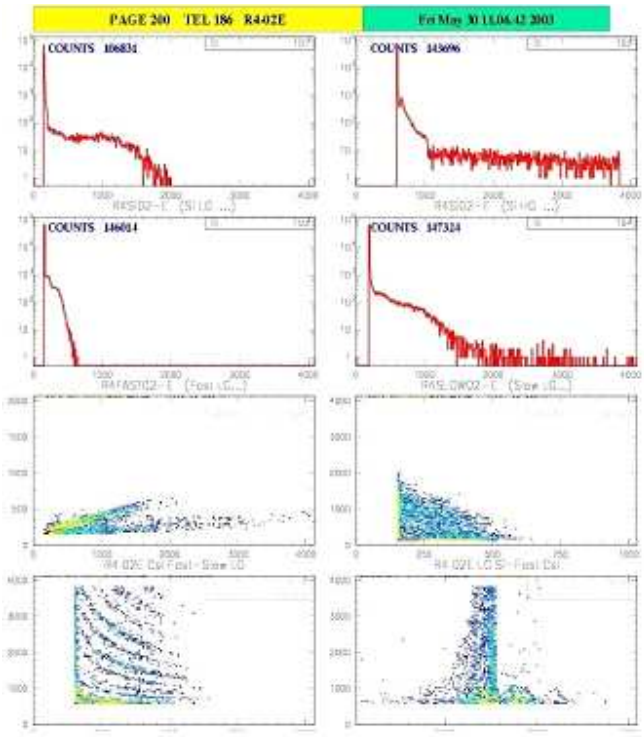

Figure 6.4: Graphics interface to check the acquisition data on-line for each telescope.

case, the low energy particles that do not exceed the energy thresholds for CsI(TI) identification will be identified by $\Delta E - E$ method.

6.2 Experimental measures by CHIMERA apparatus

The CHIMERA apparatus described above has been used in the first six months of the year 2003, for many experiments. In this section we will describe briefly the procedures performed during the measurements in order to check the data acquired.

6.2.1 Control procedures of the data acquisition

The control of the data acquired has been performed on-line by the single spectra and the 2-dimensional correlation matrix between data of the

same event. For each telescope one graphics interface has been built in which 8 spectra at the same time are shown, allowing the monitoring of all detectors.

As it is possible to see, for example, in Fig. 6.4 there are four single spectra:

- Low-gain signal of the Si;
- Hight-gain signal of the Si;
- Fast-component of the CsI;
- Slow-component of the CsI

and four 2-dimensional matrix:

- Fast-slow matrix of the CsI
- matrix Si low-gain versus fast-component of the CsI;
- matrix Si hight-gain versus fast-component of the CsI;
- matrix Si low gain versus time of flight.

Through this graphics interface it is also possible to have a good control on-line of the multiplicity of the fragments products event-by-event. In Fig. 6.5 the spectra of the total multiplicity for event is shown. The multiplicity corresponds to the number of detectors in coincidence. The detectors take into account in this case are the silicon detectors. In figure 6.5 the spectra of multiplicity obtained with a *trigger* of multiplicity greater or equal than two, is shown, in other words, it means that at least two telescope must be affected at the same time by the fragments of the reaction.

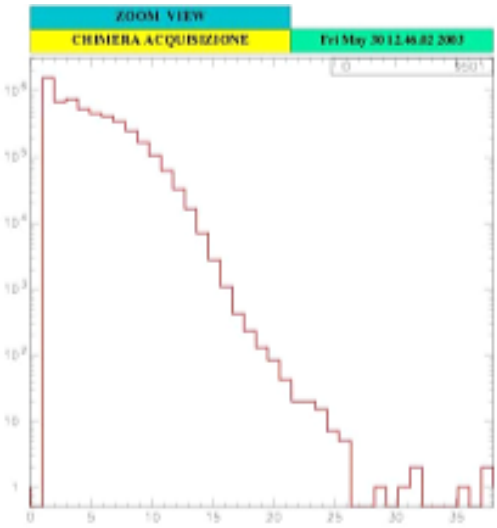

Figure 6.5: Graphics interface to check the total multiplicity of the fragments products event-by-event.

6.3 Classification of experimental events: SVM versus estimator of $\hat{b}$.

With the purpose of exploring the performances of the SVM analysis on real data, 79785 experimental events are analyzed. The experimental data have been measured with Chimera device in July 2003.

SVM is then trained over the whole amount of 63571 CMD events using the best configuration with a polynomial kernel with $d = 3$, $\gamma = 1$, $r = 0$ and considering the complete set of features. The charge distribution of the experimental events classified by SVM and estimate function of b as CEN, SEM, and PER, respectively are compared, in Fig. 6.6. The sum for the experimental events classified by SVM as CEN events is 4934, for those classified as SEM events is 58136, and for those classified as PER events is 16715. For the estimated value of $\hat{b} = b/b_{max}$ events these sums are 6703, 26435, and 46647, respectively. On the upper panel of the Fig. 6.6, for the most central collisions events selected by SVM and by $\hat{b}$, it is possible to see the emission of many light-particles and

fragments which may be interpreted as the coexistence of the liquid and gas phases (multifragmentation regime), while for the events selected by the estimated value of the impact parameter it is possible to see also the presence of quasi-projectile residue. It can depend on the fact that the $\hat{b}$ method produces more errors than SVM in the classification task. As the values of impact parameter become larger, the involved excitation energy is limited and the decay corresponds to light-particles evaporation leading to a quasi-projectile residue (liquid nuclear matter), as it is shown in Fig. 6.6 for peripheral collisions.

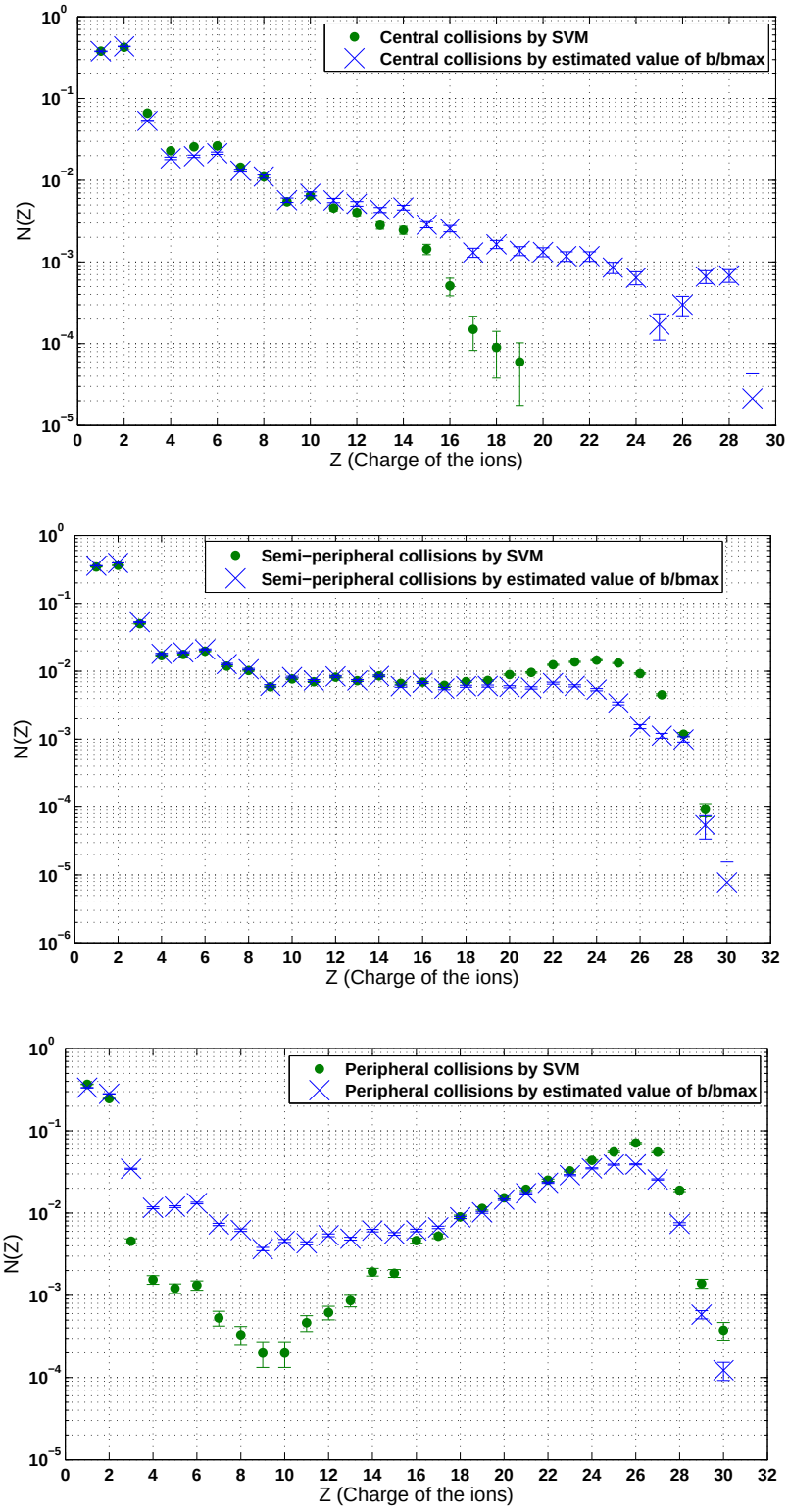

Figure 6.6: SVM versus estimated value of $\hat{b} = b/b_{max}$ classification on experimental data: charge distribution of CEN events (upper part), SEM events (half part) and PER events (lower part). The number of ions has been normalized on the total number of ions products.

Chapter 7

Conclusions

In this work we have applied a new technique for the analysis of nucleus-nucleus collisions data. This method allows to classify the data corresponding to impact parameter intervals through a particular multidimensional analysis based on a Support Vector Machine classifier. The SVM algorithm consists of two main steps. In the first one, known as the training phase, SVM learns how to discriminate between peripheral, semi-peripheral and central reactions through the knowledge of samples that are taken from a large number of events generated by a model. In this work we have used Classical Molecular Dynamics (CMD) and Heavy Ion Phase-Space Exploration (HIPSE) models. In the training phase the values of the impact parameter are well known. In the second one, known as the test phase, what has been learned is tested on new events generated by the same models. In the first part of this work we have checked the SVM classifier results on the CMD and HIPSE events in order to understand if the SVM algorithm is an *adaptable* process. The performances showed that, even though the two models are very different in terms of interaction potential, the SVM provides us an acceptable global accuracy for each test. In other terms, the learning phase has a good capacity of adaptability also in the test phase on the events coming from different models. The SVM distribution of CMD events shows a good agreement with the original CMD ion charge distribution. Looking at both the global accuracies and confusion matrices reported

in chapter 5, SVM achieves better results than the estimate function for the impact parameter $\hat{b}$ and the estimate value of b by PCA analysis. This SVM technique works well, in particular, when used to discriminate central collisions. Indeed, central collisions are correctly classified with an accuracy of about 92% as shown in Fig. 5.3. In addition, the Recursive Feature Elimination (RFE) technique allows us to extract the observables more correlated to the impact parameter and to quantify the contribution of each observable to the classification process of the classes. As shown in Fig. 5.1, it is possible to see that the most important contribution for the classification process is given by the number of intermediate mass fragments (Nimf). It is important to stress that each observable is important to discriminate between different classes because the global accuracy of classification increases with the number of the features taken into account. On the basis of these results, we can assert that the SVM analysis proposed is a good and new approach to extract the impact parameter value from the CMD model data by using many features at the same time. By means of the training phase which is an *adaptable* process, we have applied the SVM classifier also to the experimental data measured in the framework of the NUCL-EX collaboration. In this second part of the work we have performed the learning phase of SVM on the total amount of CMD events and then we have extracted the classes for the experimental data. We have compared these results with the same distibutions obtained by the estimated function $\hat{b}$. As shown in Fig. 6.6, for the central collisions the charge distribution obtained with the estimate function contains also non-central binary events with a quasi-projectile residue while in the SVM distribution this is not the case. In conclusion the SVM classifier succeds to select events more homogeneous. It is important to stress that this new technique of analysis is model dependent, whereas the other two methods are indipendent from any model.

Further work on this subject is needed in order to understand if this

method could be widely used or should be some limitation in the applicability.

Bibliography

[1] V. Weisskopf 1937 *Phys. Rev.* **52** 295.

[2] N.Bohr 1936 *Nature* **137** 351.

[3] T. Ericson 1960 *Adv. Phys.* **9** 425.

[4] W. Friedman and W. G. Lynch 1983 *Phys. Rev.* C **28** 16.

[5] J.J. Gaarhoje 1992 *Ann. Rev. Nucl. Part. Sci.* **42** 483.

[6] 1996 Proc. Groningen Conf. on Giant Resonances *Nucl. Phys.* A **599** 1c.

[7] 1988 First topical meeting on giant resonance in heavy-ion collisions *Nucl. Phys.* A **482** 1c.

[8] K.A. Snover 1986 *Ann. Rev. Nucl. Part. Sci.* **36** 545

[9] 1999 Proc. Topical Conf. on Giant Resonances *Nucl. Phys.* A **649** 1c.

[10] J. P. Alard et al. 1987 *Nucl. Instrum. Methods* **261** 379.

[11] J. A. Hauger et al. 1998 *Phys. Rev.* C **57** 764.

[12] W. Reisdorf 1998 *Nucl. Phys.* A **630** 15c.

[13] W. Reisdorf et al. 1999 *Proc. XXVII Winter Meeting (Hirshegg)*

[14] R. Stock 1986 *Phys. Rep.* **135** 261.

[15] J. F. Lecolley et al. 1995 *Phys. Lett.* B **354** 202.

[16] E. Suraud, Ch. Grégoire et B. Tamain, Prog. Part. Nuc. Phys. **23** (1989) 357.

[17] D. H. E. Gross, Rep. Prog. in Phys., **53** (1990) 605.

[18] J. Pochodzalla et al., *Phys. Rev. Lett.* **75** (1995) 1040.

[19] J. B. Natowitz et al., *Phys. Rev.* C **65** (2002) 34618.

[20] B. Borderie et al., *Nucl. Phys.* A **734** (2004) 495.

[21] M. D'Agostino et al., *Phys. Lett.* B **473** (2000) 219; *Nucl. Phys.* A **699** (2002) 795.

[22] B. Tamain et al., in *Phase transitions in strongly interacting matter* Prague, (2004). *Nucl. Phys* A.

[23] J. B. Elliott et al., *Phys. Rev. Lett.* **88** (2002) 042701.

[24] M. D'Agostino et al., *Nucl. Phys* A **724** (2003) 455.

[25] F. Haddad et al., *Phys. Rev. C* Vol. 55, (1997), 1371.

[26] S. A. Bass et al., *J. Phys.* G **20**, L21 (1994).

[27] C. David et al., *Phys. Rev.* C **51** 1453 (1995).

[28] S. A. Bass et al., *Phys. Rev.* C **53**, 2358 (1996).

[29] T. Naruyama *et al.*, Prog. Theor. Phys. **87**, 1367 (1992).

[30] B. Charity *et al.*, Nucl. Phys. **A 483**, 371 (1988).

[31] D. Durand, Nucl. Phys. **A 541**, 266 (1992).

[32] L. Tarassenko (1998), *Guide to Neural Computing Applications.* Butterwoth - Heinemann.

[33] B. Scholköpf. *Support Vector Learning.* Ph.D. thesis, Universität Berlin.

[34] R.O. Duda et al. (2000). *Pattern Classification*. John Wiley & Sons, Inc., New York, 2nd edn.

[35] M. Masotti, *A ranklet-based image representation for mass classification in digital mammograms*, Medical Physics, 33(10) (2006) 3951-3961.

[36] E. Angelini, R. Campanini, E. Iampieri, N. Lanconelli, M. Masotti, M. Roffilli, *Testing the performances of different image representations for mass classification in digital mammograms*, International Journal of Modern Physics C, 17(1) (2006) 113-131.

[37] R. Campanini, D. Dongiovanni, E. Iampieri, N. Lanconelli, M. Masotti, G. Palermo, A. Riccardi, M. Roffilli, *A novel featureless approach to mass detection in digital mammograms based on Support Vector Machines*, Physics in Medicine and Biology, 49(6) (2004) 961-976.

[38] M. Masotti, S. Falsaperla, H. Langer, S. Spampinato, R. Campanini *Application of Support Vector Machine to the classification of volcanic tremor at Etna*, Italy, Geophysical Research Letters, 33(20) (2006) L20304.

[39] http://www.clopinet.com/isabelle/Projects/SVM/applist.html

[40] R. Guitierrez-Osuma, "Introduction to Pattern Analysis", *Texas A&M University*.

[41] V.N.Vapnik, *The nature of statistical learning theory*, Springer Verlag, New York, 1995.

[42] K.R.Müller, S. Mika et al., *IEEE Transactions on Neural Networks*, Vol.12, no.2, March, 2001.

[43] John Shawe-Taylor and Nello Cristianini, *Kernel Methods for Pattern Analysis*, Cambridge University Press, 2004.

[44] C.J.C.Burges. A tutorial on Support Vector Machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2):1-47, 1998.

[45] C.Cortes and V.Vapnik. Support-Vector Networks. *Machine Learning*, 20:273-297, 1995.

[46] B. Scholköpf. *Support vector learning*. R.Oldenbourg Verlag, Munich, 1997.

[47] C.M. Bishop. *Neural networks for pattern recognition*. Oxford University Press, Oxford, 1995.

[48] L.S. Lasdon. *Optimization theory for Large Systems*. Operations Research Series. MacMillan, 1970.

[49] D.G. Luenberger. *Optimization by vector space methods*. John Wiley and Sons, 1969.

[50] M. Pontil and A. Verri. Proprieties of Support Vector Machines. *Neural Computation*, 10:977-996, 1998.

[51] J. Mercer. Functions of positive and negative type and their connection with the theory of integral equations. *Philosophical Transactions of the Royal Society of London*, Series A **209** (1909): 415-446.

[52] B.E. Boser, I.M. Guyon, and V.N. Vapnik. A training algorithm for optimal margin classifiers. In D. Haussler, editor, *Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory (COLT)* ACM Press, July 1992, pp. 144-152.

[53] G. Wahba. Support vector machines, reproducing kernel Hilbert spaces and the randomized GACV. In B. Schölkopf, C.J.C. Burges, and A.J. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*. MIT Press, pp. 69-88.

[54] V. Latora, M. Belkacem and A. Bonasera, *Phys. Rev. C*, Vol. 52, (1994), 271.

[55] D. Lacroix et al. Physical Review C 69, 054604 (2004).

[56] L. Wilet, E.M. Henley at al., *Nucl. Phys.* A 282, pp. 341 (1977).

[57] S.E. Koonin and D.C. Meredith, *Computational Physics*, (Addison Wesley, Redwood City, California, 1990).

[58] D. Durand, E. Suraud, and B. Tamain, *Nuclear Dynamics in the Nucleonic Regime* (IOP Publishing, Bristol, UK, and Philadelphia, 2001).

[59] L. Moretto and G. Wozniak, Annu. Rev. Nucl. Part. Sci. **43**, 379 (1993); *Proceedings of the XXVII International Workshop on Gross Properties on Nuclei and Nuclear Excitations, Hirshegg, 1999*, edited by H. Feldmeier (GSI, Darmstadt, 1999).

[60] J. Randrup, W.J. Swiatecki and C.F. Tsang, Lawrence Berkeley Laboratory report, LBL-3603 (1974).

[61] J. Bloki, J. Randrup, W.J. Swiatecki and C.F. Tsang, Ann. Phys. (N.Y) **105** (1977) 427.

[62] W. U. Schroder and J. R. Huizenga, in *Treatise on Heavy Ion Science*, edited by A. Bromley (Plenum, New York, 1984), Vol. 2.

[63] P. Désesquelles, Ann. Phys. (Paris) 20 (1995).

[64] S. Aiello et al., Nucl. Instr. and Meth. A 385 (1997) 3060.

[65] F. Benrachi et al., Nucl. Instr. and Meth. 281 (1989) 137.

[66] J. Alarja et al., Nucl. Instr. and Meth. A 242 (1986) 352.

[67] D. Guinet et al., Nucl. Instr. and Meth. A 278 (1989) 614.

[68] M. Moszyski et al., Nucl. Instr. and Meth. A336 (1993) 587.

[69] W. Skulski et al., Nucl. Instr. and Meth. 458 (2001) 759.

[70] http://www.lns.infn.it/research/chimera/index.html