TECHNISCHE UNIVERSITÄT WIEN

IμE

DIPLOMA THESIS

Cluster Detection Algorithm to Study Single Charge Trapping Events in TDDS

Performed at the Institute for Microelectronics
at the TU Wien

Supervised by
Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Tibor Grasser
and
Dipl.-Ing. Michael Waltl

by

Marco Huymajer
0126548 / E 066 508

Vienna, on June 16, 2016

Signature

Abstract

The phenomenon that the threshold voltage of metal-oxide-semiconductor field-effect transistors changes, when the device is stressed at elevated temperatures, has been observed first in the 1960s and termed the bias temperature instabilities (BTI). It is commonly accepted that the threshold voltage shifts can be attributed to defects located inside the oxide, so called border states and interface states. By investigating BTI in large area devices the collective response of a vast amount of defects can be measured as a continuous degradation and recovery behavior. However, to model the complex nature of BTI properly, a detailed knowledge of the physical mechanism behind charge trapping of single defects is required. This can be achieved by using nanoscale devices, which in contrast to their large area counterparts, contain only a handful of defects with experimentally resolvable threshold voltage shifts. As a consequence, the intricate charge trapping behavior can thus be studied for each defect individually. For the analysis of single charge trapping, the time-dependent defect spectroscopy (TDDS) has been recently proposed. The substantial amount of manual effort currently necessary to analyze TDDS data call for a more automated TDDS workflow. One particular time-consuming task during the TDDS analysis is to identify clusters in the recorded data. Each cluster is subsequently linked to a particular defect in order to obtain statistical parameters of the charge transition times. A detailed understanding of charge trapping for a certain technology requires the trapping parameters of a large number of defects. To simplify the process and increase the accuracy of the extraction of the charge transition times, a sophisticated data analysis algorithm has been developed. This work describes the implementation of an unsupervised algorithm based on expectation–maximization (EM) to perform an automatic cluster detection. Satisfactory results, compared to manually analyzed data, are achieved using the presented algorithm. In addition, the effort necessary to identify single traps is significantly reduced. Although the algorithm requires further optimization with respect to the assignment of clusters to defects, this work offers the ability to efficiently study numerous single traps.

Kurzfassung

Das Phänomen einer Änderung der Schwellspannung von Metall-Oxid-Halbleiter-Feldeffekttransistoren (MOSFETs) aufgrund von Stress bei höheren Temperaturen wurde erstmals in den 1960er Jahren beobachtet und als temperatur- und gatespannungsabhängige Instabilitäten (Bias Temperature Instabilities – BTI) bezeichnet. Der Grund dieser Schwellspannungsänderungen wird gemeinhin auf Störstellen im Inneren des Oxids, sogenannte Grenzschichtzustände, zurückgeführt. Anhand Untersuchungen von BTI bei Bauelementen kann die gemeinsame Reaktion einer großen Anzahl von Störstellen als kontinuierliches Degradations- und Regenerationsverhalten gemessen werden. Um das komplexe Wesen von BTI angemessen modellieren zu können, ist jedoch ein umfassendes Verständnis der physikalischen Vorgänge des Ladungsaustausches einzelner Störstellen unerlässlich. Dies wird durch den Einsatz nur wenige Nanometer großer Transistoren erreicht, die im Gegensatz zu ihren größer dimensionierten Gegenstücken lediglich eine kleine Anzahl von Störstellen mit experimentell messbaren Veränderungen der Schwellspannung aufweisen. Daher kann das komplexe Verhalten des Ladungsaustausches jeder einzelnen Störstelle gesondert untersucht werden. Die Time-Dependent Defect Spectroscopy (TDDS) wurde kürzlich zur Analyse einzelner Ladungsaustauschvorgänge vorgestellt. Der mit der Analyse von TDDS-Daten verbundene erhebliche Arbeitsaufwand macht eine zunehmende Automatisierung notwendig. Eine besonders zeitaufwändige Tätigkeit während der TDDS-Analyse ist die Identifizierung von Clustern im Datenmaterial. Dabei wird jeder Cluster einer bestimmten Störstelle zugeordnet, um statistische Parameter der Ladungsübergänge zu erhalten. Ein vertieftes Verständnis des Ladungsaustausches in Hinblick auf bestimmte Technologien erfordert die Parameter für eine große Anzahl von Störstellen. Um diesen Prozess zu vereinfachen und die Genauigkeit der extrahierten Ladungsübergangszeiten zu erhöhen, wurde ein ausgeklügelter Algorithmus zur Datenanalyse entwickelt. Die vorliegende Arbeit beschreibt die Implementierung eines auf Expectation-Maximization (EM) beruhenden, unüberwachten Algorithmus zur automatisierten Clustererkennung. Im Vergleich zu manuell analysierten Daten konnte durch den Einsatz des hier vorgestellten Algorithmus zufriedenstellende Ergebnisse erzielt und der erforderliche Arbeitsaufwand zur Identifizierung einzelner Störstellen erheblich reduziert werden. Obwohl der Algorithmus hinsichtlich der Zuweisung der Cluster zu einzelnen Störstellen noch weiterer Verbesserung bedarf, ermöglicht diese Arbeit deren effiziente Untersuchung.

Contents

1 Introduction

Bias temperature instabilities (BTI) are today's major reliability issues occurring in metal-oxide-semiconductor field-effect transistors (MOSFETs) and can be attributed to defects located inside the oxide. Section 1.1 gives a brief overview of BTI. The detailed motivation of this work is given in Section 1.3, but summing up, the aim of this work is to develop a set of machine learning algorithms (c.f. Section 1.2) to further enhance the current workflow of handling measurement data. One major achievement regarding analysis of BTI is the time-dependent defect spectroscopy (TDDS), discussed in Chapter 2. In this context, a clustering algorithm for collected data should be developed, which is a particular unsupervised machine learning algorithm that automatically finds groups in the data. Chapter 3 covers the two clustering techniques employed in this work and Chapter 4 deals with the implementation and how the proposed method is embedded into the data analysis workflow. In Chapter 5 this work concludes with some results on real measurement data, recorded employing the TDDS.

1.1 Bias Temperature Instability

Defects at the semiconductor-oxide interface as well as inside the oxide result in a non-ideal behavior of MOSFETs. In particular, such single defects can get repeatedly charged and discharged during device operation and are responsible for random telegraph (RTN) and 1/f noise as well as for a significant contribution to BTI [10].

1.1.1 Physical Considerations

Oxide defects in MOSFETs can be classified into two different classes according to the underlying physical phenomenon [8, 5]:

Interface Traps — They are commonly attributed to trivalent silicon dangling bonds called P_{b0} and P_{b1} centers on (110)-oriented interfaces and P_b centers on (111) interfaces.

Border Traps and Oxide Traps — Although their exact chemical structure is still controversially discussed, border traps are often associated with E' centers, which are trivalent silicon dangling bonds in the SiO_2. The term border traps applies to oxide defects located within approximately less than 3 nm from the oxide interface. With today's thin oxides of small scale devices this distinction is not longer very usual and the traps are commonly referred to as border traps.

By randomly exchanging an electron or a hole with the substrate, single traps lead to random fluctuations of the drain source current I_{DS}. On small-area devices the single carrier capture and emission events are visible as discrete steps in I_{DS}. Furthermore, under certain bias conditions RTN is observed in I_{DS}.

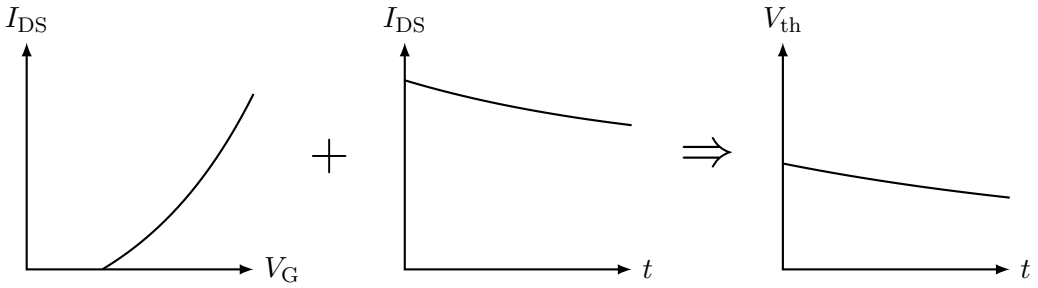
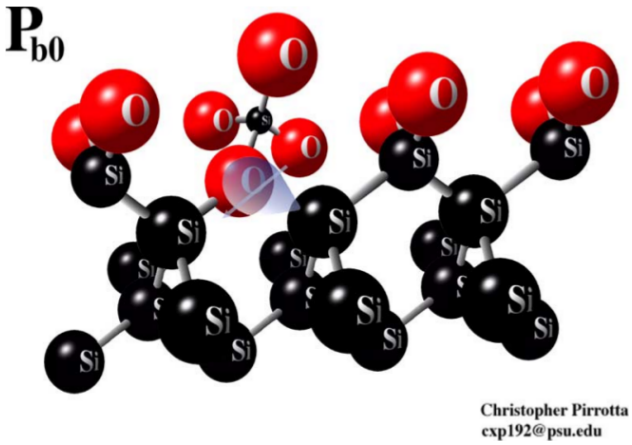


Figure 1.1: The $I_{DS}(V_G)$ characteristic **(left)** together with a recovery trace of the drain current $V_G(t)$ **(middle)** of a device allows for calculating the behavior of the threshold voltage V_{th} during recovery **(right)**.

Figure 1.2: Schematic drawing of the Pb0 Si/SiO_2 interface defects, responsible for BTI [4].



The phenomenon BTI is observed in both n-channel MOSFET (nMOS) and p-channel MOSFET (pMOS) transistors. In common, BTI are classified into positive BTI (PBTI) and negative BTI (NBTI), depending on whether a positive or a negative stress voltage is applied at the gate, respectively. During the long history of BTI, mainly NBTI is studied on pMOSFETs as it is more pronounced compared to PBTI on nMOSFETs. Apparently, less is known about NBTI on nMOSFETs and PBTI on pMOSFETs, which is sometimes difficult to study as most structures show electrostatic discharge (ESD) protection diodes at the gate. Nonetheless, to understand charge trapping for a certain technology, bot NBTI and PBTI, have to be studied in detail.

To compare the impact of BTI between different devices and technologies, BTI is expressed in terms of a gate voltage shift. Therefore ΔV_{th} can be considered as the amount of gate voltage V_G necessary after the switch to maintain the drain current I_{DS} at the same level as before the switch. An example of the conversion of I_{DS} is to an equivalent ΔV_{th} is shown in Fig. 1.1 for a continuous recovery behavior which is typical for large area devices. To study the physical mechanism behind BTI the TDDS has been recently proposed, and will be discussed more detailed in Chapter 2.

Density Functional Theory To account for the chemical structure behind the defects, density functional theory (DFT) calculations are carried out. Thereby, computational expensive simulations considering different atomic configurations are performed. With the results in hand the transition energy barriers between the neutral and charged states of particular structures, for instance E' centers, can be determined. These energy barriers allow to link charge trapping kinetics obtained from TDDS measurements by employing the four-state NMP model with a particular atomic configuration of a defect. A schematic drawing of a P_{b0} center is shown in Fig. 1.2.

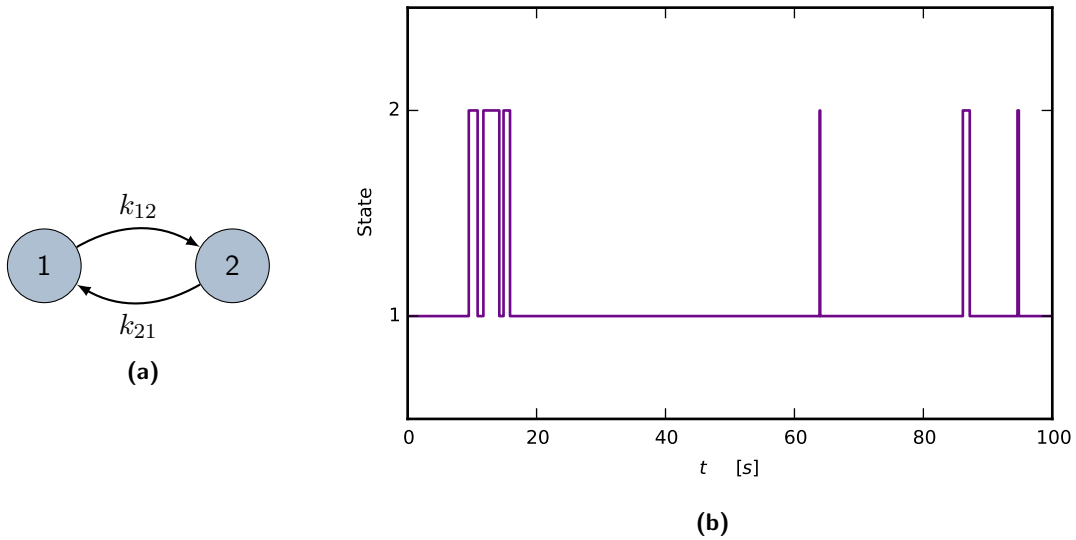


Figure 1.3: (a) The graphical representation of the two-state stochastic defect model. **(b)** An example of the state in dependence of the time for $k_{12} = 1/9\,\mathrm{s}^{-1}$ and $k_{21} = 1\,\mathrm{s}^{-1}$.

1.1.2 Stochastic Defect Modeling

Defects in semiconductors and oxides don't behave deterministically but randomly. The stochastic behavior of such defects can be described by a Markov process, a stochastic process in which the future state only depends on the current state but not on the history of the system [3].

In the simplest assumption the defect can only take two different stable states, a neutral and a charged one. Fig. 1.3a shows a state diagram of a two-state defect, in which the states are represented by circles. In the following it is arbitrarily assumed that state 1 represents the neutral and state 2 the charged state.

The Master Equations The probability for a transition from state i to state j within a infinitesimal small time interval is given by the transition rate k_{ij} and visible as arrows in the diagram in Fig. 1.3a. The system is analytically represented by an ordinary differential equation [10], the solution of which gives the probability of being in the state 2 at any particular time t

$$p_2(t) = p_2(\infty) + [p_2(0) - p_2(\infty)]\mathrm{e}^{-t/\tau}, \tag{1.1}$$

where

$$p_2(\infty) = \frac{k_{12}}{k_{12} + k_{21}} \quad \text{and} \quad \tau = \frac{1}{k_{12} + k_{21}}. \tag{1.2}$$

As the defect must be in either of its two states, one can calculate from $p_1(t) = 1 - p_2(t)$ the probability of being in the state 1

$$p_1(t) = p_1(\infty) + [p_1(0) - p_1(\infty)]\mathrm{e}^{-t/\tau}, \tag{1.3}$$

where

$$p_1(\infty) = \frac{k_{21}}{k_{12} + k_{21}}. \tag{1.4}$$

The equations Eqs. (1.1) and (1.3) are called the master equations which fully describe the two state Markov process.

Capture and Emission Time Constants TDDS, the analysis framework utilized in this work, as well as other measurement rely on measuring the transition of a defect from the charged state to the neutral state, or the other way round. As the time the transition takes place is a stochastic process, it is necessary to know the mean value $\bar{\tau}_{12}$ that a defect makes a transition to state 2 provided that it is in state 1. As $\bar{\tau}_{12}$ does not depend on k_{21} it can be set to 0 without any loss of generality. Together with the initial condition $p_1(0) = 1$ Eq. (1.3) reduces to

$$p_1(t) = \mathrm{e}^{-k_{12}t}. \tag{1.5}$$

The probability that the defect has transitioned to state 2 at a time t is given by the cumulative distribution

$$P(\tau_{12} \leq t) = p_2(\tau_{12}) = 1 - p_1(\tau_{12}) = 1 - \mathrm{e}^{-k_{12}\tau_{12}}. \tag{1.6}$$

One can get the corresponding probability density function by differentiating

$$g(\tau_{12}) = \frac{\mathrm{d}p_2(\tau_{12})}{\mathrm{d}\tau_{12}} = k_{12}\mathrm{e}^{-k_{12}\tau_{12}}. \tag{1.7}$$

This probability density function is the exponential distribution and will be discussed further in Section 3.2.

The average capture time $\bar{\tau}_\mathrm{c}$ equals the expectation value of τ_{12}

$$\bar{\tau}_\mathrm{c} = \mathrm{E}[\tau_{12}] = \int_0^\infty \tau_{12} g(\tau_{12})\,\mathrm{d}\tau_{12} = \frac{1}{k_{12}}. \tag{1.8}$$

Analogously the average emission time $\bar{\tau}_\mathrm{e}$ is the expectation value of τ_{21}

$$\bar{\tau}_\mathrm{e} = \mathrm{E}[\tau_{21}] = \frac{1}{k_{21}}. \tag{1.9}$$

Using this two state model RTN signals can be modeled properly, see Fig. 1.3b. However this simple model is unable to accommodate a number of experimentally observed phenomena. One of these is anomalous RTN (aRTN) or temporary RTN, a form of RTN that is suspended for a specific time. Such behavior can be accounted for by expanding the two state model with a third metastable state. An example of a simulated aRTN can be seen in Fig. 1.4.

1.1.3 Impact on the Device

As aforementioned, defects responsible for the ΔV_th are due to inaccuracies of the perfect atomic structure inside the oxide or a consequence of lattice mismatch between SiO_2 and Si at the interface. The former are considered oxide traps and are typically attributed to slower time constants. In contrast, the latter are referred to as interface states with very fast charge transition times which cannot be measured directly.

In contrast to the time-invariant behavior of an ideal device, the behavior of a device subject to BTI stress detrimentally changes over the time. Its normally assumed that neutral defects do not

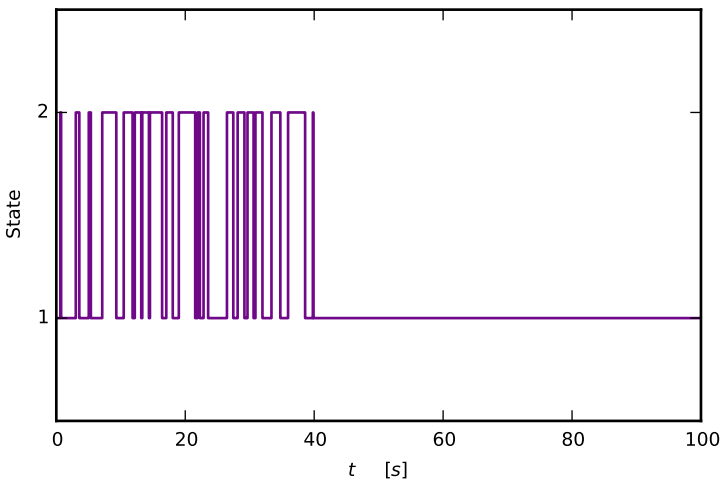


Figure 1.4: A simulated aRTN produced by a multi-state defect. The RTN stops after approximately $t = 40\,\mathrm{s}$.

interact with the remainder of the device, while charged defects result in a threshold voltage change through the generation of a charge layer at the Silicon-oxide interface. In this context one has to differentiate between the charge Q_{it} caused by the fast interface states and the charge Q_{ox} caused by the slower border traps. The effect of these charges on the threshold voltage V_{th} can be written as

$$\Delta V_{\mathrm{th}}(V_{\mathrm{G}}, t) = V_{\mathrm{th}} - \frac{Q_{\mathrm{it}}(V_{\mathrm{G}}, t) + Q_{\mathrm{ox}}(V_{\mathrm{G}}, t)}{C_{\mathrm{ox}}}, \tag{1.10}$$

where the oxide capacity per unit area for an oxide with a thickness of t_{ox} is given by

$$C_{\mathrm{ox}} = \frac{\epsilon_0 \epsilon_{\mathrm{r}}}{t_{\mathrm{ox}}}. \tag{1.11}$$

Although the time constants of the interface states are to small to be measurable as a transition of V_{th}, instead they become noticeable as a increase of the sub-threshold slope compared to the virgin MOS transistor. By means of a charge sheet approximation, i.e. by assuming that the charge of the defect is homogeneously spread across the oxide, one can derive an expression for the charge caused by the border traps

$$Q_{\mathrm{ox}}(V_{\mathrm{G}}, t) = q \frac{1 - x/t_{\mathrm{ox}}}{WL} \eta_{\mathrm{r}} p_2(t), \tag{1.12}$$

with the elementary charge q, the position x of the defect from the surface, the width W and the length L of the transistor, an empirical parameter η_{r} and the probability p_2 of being in state two.

1.2 Machine Learning

One can define machine learning as a set of methods that automatically find patterns in data and make predictions and other decisions based on them [17, 3].

In general, machine learning techniques can be classified in three different categories, see Fig. 1.5:

Supervised Learning The term supervised learning denotes an algorithm that is given the training set $\mathbf{x} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ of N observations together with the corresponding target values

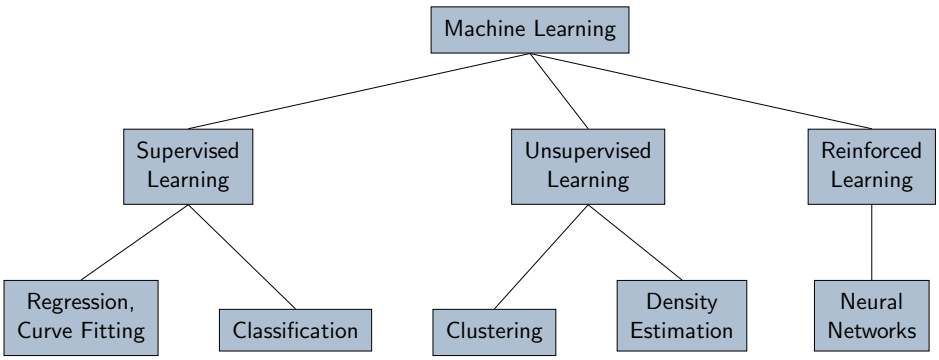


Figure 1.5: Classification (second row) and typical examples (third row) of different machine learning techniques

$\mathbf{t} = \{\mathbf{t}_1, \ldots, \mathbf{t}_N\}$ during the training or learning phase. The objective is to make a prediction of the value $\hat{\mathbf{t}}$ for some new value $\hat{\mathbf{x}}$ during the generalization step by tuning some model coefficient $\boldsymbol{\theta}$. The observations $\mathbf{x}_n$ could be scalars, D-dimensional vectors or even more complex structured objects.

One can further distinguish this technique according to the type of the target values $\mathbf{t}$:

- The term *classification* is used where the target value consists of a discrete number of categories. Fitting a curve from noisy observations is a typical example of this technique.
- The term *regression* is used where the target value is a continuous variable. Examples for this learning method include pattern and face recognition.

Unsupervised Learning This method differs from the supervised learning in that the algorithm is not provided a target set $\mathbf{t}$. This technique is often more convenient than supervised learning, since it does not require any human interaction.

Finding groups of similar observations, called *clustering*, or determining model parameters from the observed data, called *density estimation* are typical application of this machine learning technique. Both of them are extensively used in this work.

Reinforced Learning In contrast to supervised learning, the algorithm is not given any training set at all, but must discover it instead by trial and error itself. This is done by finding actions that maximizes some kind of reword. An example would be a *neural network* learning how to play a game.

1.3 Motivation

The outstanding possibilities that the TDDS offers compared to other BTI measuring techniques stand oppose to its biggest disadvantage, the high workload necessary to obtain satisfactory results. Although it is difficult to formulate a generalized work flow for TDDS, one can summarize the following main tasks, see Fig. 1.6:

Prescreening Not all devices contain defects suitable for TDDS analysis. For instance, if defects show emission times too close to each other, they can not be unambiguously

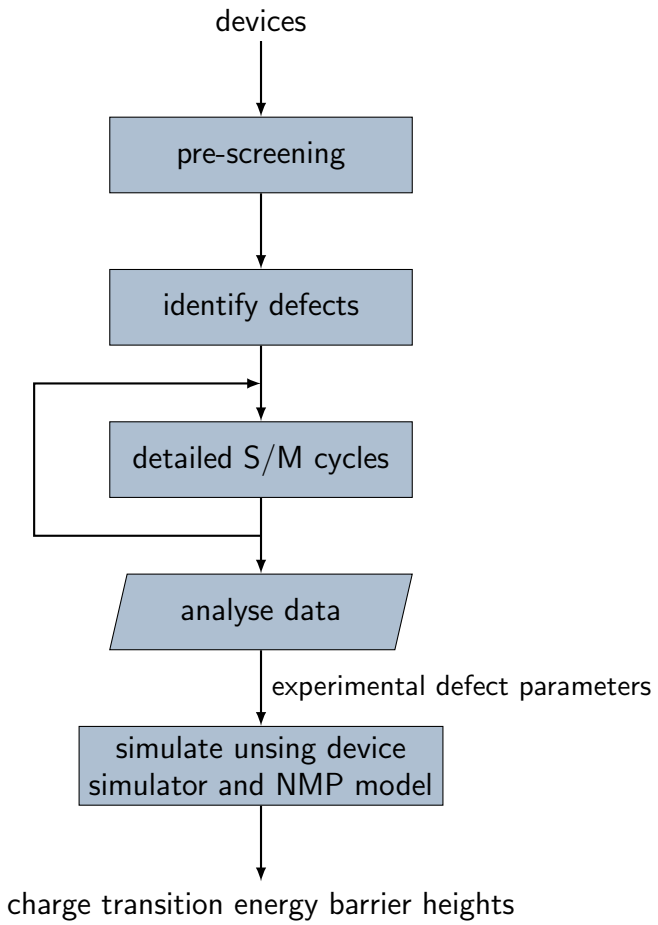


Figure 1.6: The typical TDDS measurement workflow

attributed to a single defect. Furthermore, defects with step heights at the measurement limit are also very hard to resolve.

Defect Identification Once a device is selected, the initial capture and emission times of the present defects can be determined by initial S/M measurements.

Detailed S/M Cycles By deliberately varying the stress and recovery time as well as the stress and recovery bias, the capture and emission time characteristics for a wide gate voltage range can be determined.

Data Analysis The analysis of TDDS data is very challenging and time-consuming. At this point, the cluster detection algorithm investigated in this thesis improves the performance of the evaluation procedure.

Simulation Finally, the four-state nonradiative multi-phonon (NMP) model is used to explain the experimental data.

Note that for the same device the experimental parameters, i.e. t_s, t_r, V_s, V_r, T and V_DS, have to be varied and for each for each set of parameters typically 100 traces have to be recorded and analyzed. There are different reasons for this procedure, which is represented by the loop in Fig. 1.6:

- Statistical significance has to be reached to obtain reliable values for τ_e and τ_c.
- One uses different sets of stress parameters such that a change of the emission probability (depends on t_s) and emission time (depends on V_G in case of switching traps) is beneficial during the subsequent analysis. This kind of deconvolution is actually one of the main ideas behind the TDDS.
- The capture time cannot be measured directly but can be determined by varying the stress time and analyzing the number of emission events.
- The effect of various stress and recovery biases and device temperatures on the capture and emission time is to be investigated.
- One wants to monitor the long term effects of the device.

Apart from repeatedly measuring the same device, there is a strong demand for measuring a huge number of different devices (more than 1000):

- Again, statistical significance to study charge trapping for a particular technology should be reached.
- One wants to investigate for geometries dependent effects.

The manual interaction currently needed during the analysis of the data prevents a fully automated workflow. Specifically, a time-consuming step in the analysis is to identify clusters in the spectral maps, that will be discussed in detail in the following chapter, and relate these clusters to a specific defect. For the reasons stated above the analysis should be fully automated by applying machine learning algorithms.

2 The Time-Dependent Defect Spectroscopy

In [11] a novel analysis method, the TDDS, is proposed to study trapping kinetics of border traps in detail. A rough overview of the typical TDDS workflow, was already given in Fig. 1.6. In principle, TDDS relies on repeatedly measuring the recovery of a nanoscale device subjected to BTI stress, thereby enabling the extraction of charge capture and emission times of singe defects. Based on this data the trap depth can be estimated using the well-established four-state NMP model. In the following the concept behind TDDS is discussed.

Comparison to Other Techniques In general, the response of a single device to BTI stress can be expressed as the sum of contributions from K defects with different ΔV_{th} shifts by

$$\Delta V_{\mathrm{th}} = \sum_{k=1}^{K} \eta_k B_k \tag{2.1}$$

with η_k the step height and B_k the defect occupancy function of defect k. Based on investigation on SiON transistors the number of active traps have been found to scale inversely with the gate area $N \propto (WL)^{-1}$, while the average contribution of a single trap increases when considering scaled transistors $\eta \propto WL$. The TDDS profits from the latter, as in nanoscale devices the contributions of the individual defects are more pronounced and thus visible as discrete steps in the drain-source current I_{DS}, quite in contrast to large area devices where only the mean behavior of many defects is observed. The difference between these two situations can be seen in Fig. 2.1, where additionally different stress and recovery parameters have been chosen. Furthermore, the presence of only a handful of traps in such devices, permits to uniquely assign the ΔV_{th} shifts to a particular defect.

The charge transition times have been initially studied by detecting and analyzing RTN signals. One difference to this classical RTN analysis techniques is that the bias conditions are chosen to systematically force certain defects to a capture and emission event. Furthermore, RTN analysis requires the charge capture and emission to occur within the experimental window. Because of the strong bias dependence of the capture and emission time the trapping kinetics of switching traps could only by studied within a very narrow gate voltage window. In contrast, using TDDS a wide gate voltage range can be used to analyze single traps.

Stress and Recovery A key point in BTI is that the transition rates between the charged and neutral state or correspondingly the capture and emission time strongly depend on the stress and recovery bias conditions and the temperature. By careful selection of stress and recovery biases and times for each single defect, a charge capture during stress and a subsequent charge emission during recovery can be enforced. The process of repeatedly stressing and measuring is termed MSM (measure-stress-measure) and is depicted in Fig. 2.2.

The gate voltage is held at the stress voltage $V_{\mathrm{G}} = V_{\mathrm{S}}$. After the stress time t_{S} has elapsed, the gate bias is switched to the recovery voltage $V_{\mathrm{G}} = V_{\mathrm{R}}$ and the drain source current is recorded for

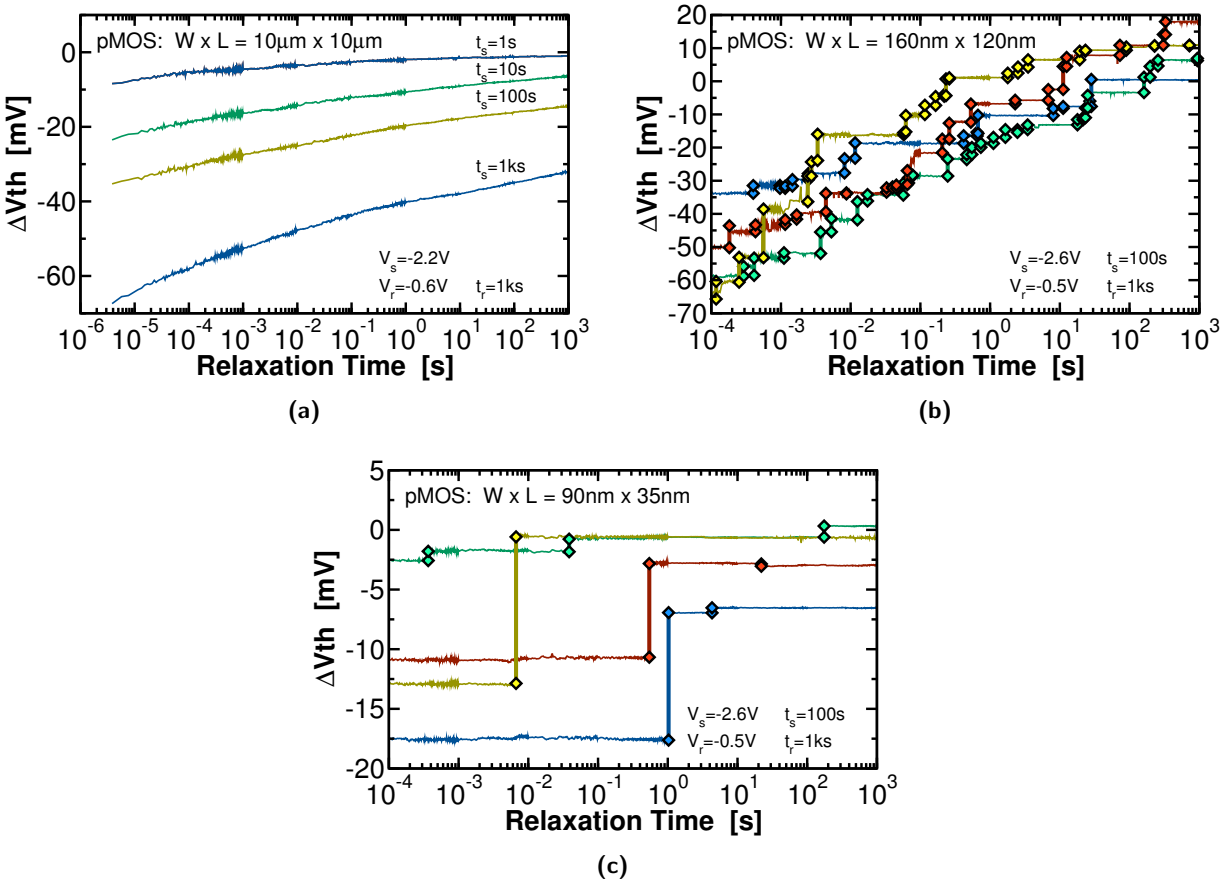
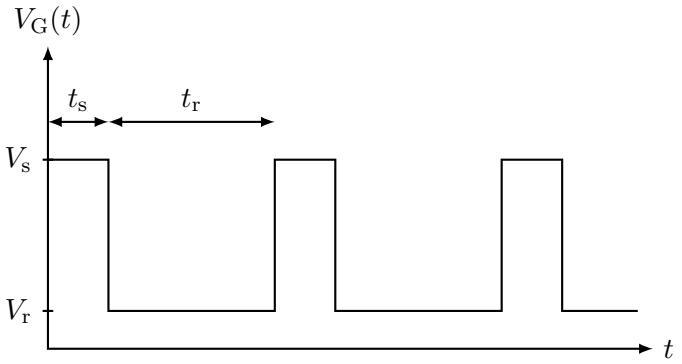


Figure 2.1: Experimentally determined recovery traces of different device areas, **(a)** $W \times L = 10\,\mu\text{m} \times 10\,\mu\text{m}$, **(b)** $W \times L = 160\,\text{nm} \times 120\,\text{nm}$, **(c)** $W \times L = 90\,\text{nm} \times 35\,\text{nm}$.

Figure 2.2: The MSM process repeatedly goes through cycles where stress conditions are applied to the device for a time span of t_s followed by recovery phase lasting t_r where a measurement of I_DS is performed.



a time span of t_R. To avoid Hot Carrier Degradation (HCD) effects during the stress cycle, V_{DS} is settled to $V_{DS} = 0\,V$ and switched to $V_{DS} = \pm 100\,mV$ during recovery, depending on whether an NMOS or PMOS transistor is studied.

Measurement Technique There are two established methods for recording the ΔV_{th} during device recovery [18]:

1. Measuring $\Delta V_G = \Delta V_{th}$, while V_G is controlled with a negative feedback circuit in a way that a constant I_{DS} is obtained, or
2. measuring I_{DS} and subsequently convert it to ΔV_{th} by use of the I_D-V_G characteristic of the device.

The TDDS measurement instrument, a custom-mode setup developed at the IuE currently employs the latter method.

Extraction of the Capture Time In order to obtain a statistical relevant amount of data, the aforementioned stress/recovery cycle is typically repeated $N = 100$ times at the same bias condition. Additional capture and emission times are recorded using variations of the stress conditions (V_S, t_S), the temperature T or the recovery bias voltage V_R. The average capture time τ_c is determined by gradually increasing the stress time from $t_S \ll \tau_c$ (where it is very unlikely for the defect to be charged at all) to $t_S \gg \tau_c$ (where the defect is certainly charged). The capture time can be finally estimated from the capture probability

$$P_c(t_s) = B(1 - e^{-t_s/\tau_c}) = \frac{N_m}{M_s}, \tag{2.2}$$

where B is the defect occupancy, t_s the stress time of the corresponding block, N_m the number of emission events of the defect m and M_s the total number of stress-recovery cycles. For the sake of completeness, note that, the effects of the drain-source voltage V_{DS} are also investigated, especially with a view to hot carrier stress.

Temperature Dependency Like other thermally activated processes the temperature dependency of the emission time could be quantitatively described by Arrhenius' law

$$k_{21} = \frac{1}{\tau_e} = \nu \exp\left(-\frac{E_a}{k_B T}\right), \tag{2.3}$$

where ν is the frequency factor, E_a the activation energy, k_B the Boltzmann constant and T the absolute temperature. Typical values for E_a are in the range of 0.3 eV to 2 eV for SiON MOSFETs [11, 26].

Spectral Maps In TDDS the charge emission events of the recovery traces measured at the same stress and recovery conditions are binned into a two-dimensional histogram, called spectral map. These spectral maps visualize the emission events of the defects defined by its average emission time τ_e and voltage step height. Figure 2.3 illustrates how the spectral map is derived from the measured emission events after negative gate bias (NBTI) stress. As the emission time and step height are characteristic properties of each defect (for certain temperature and bias conditions), individual defects can be clearly identified as disjoint clusters in the spectral map. Each cluster in the obtained spectral map is the fingerprint of an individual defect.

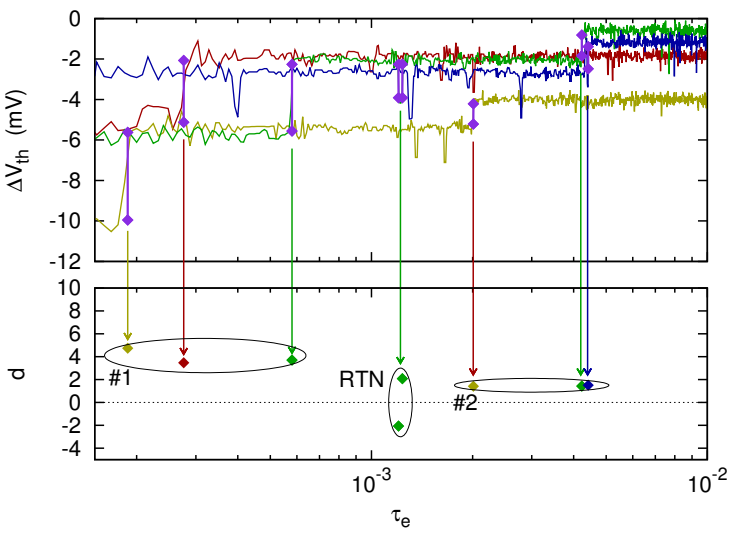


Figure 2.3: Mapping of the single charge capture and emission events from four recorded recovery traces of a PMOS device after NBTI stress **(top)** into the (τ_e, d) plane, called spectral map **(bottom)** [26]. Charge capture and emission events attributed to the same defect are marked with an ellipse. In the case of RTN, multiple capture and emission events symmetrically arranged around the abscissa are observed.

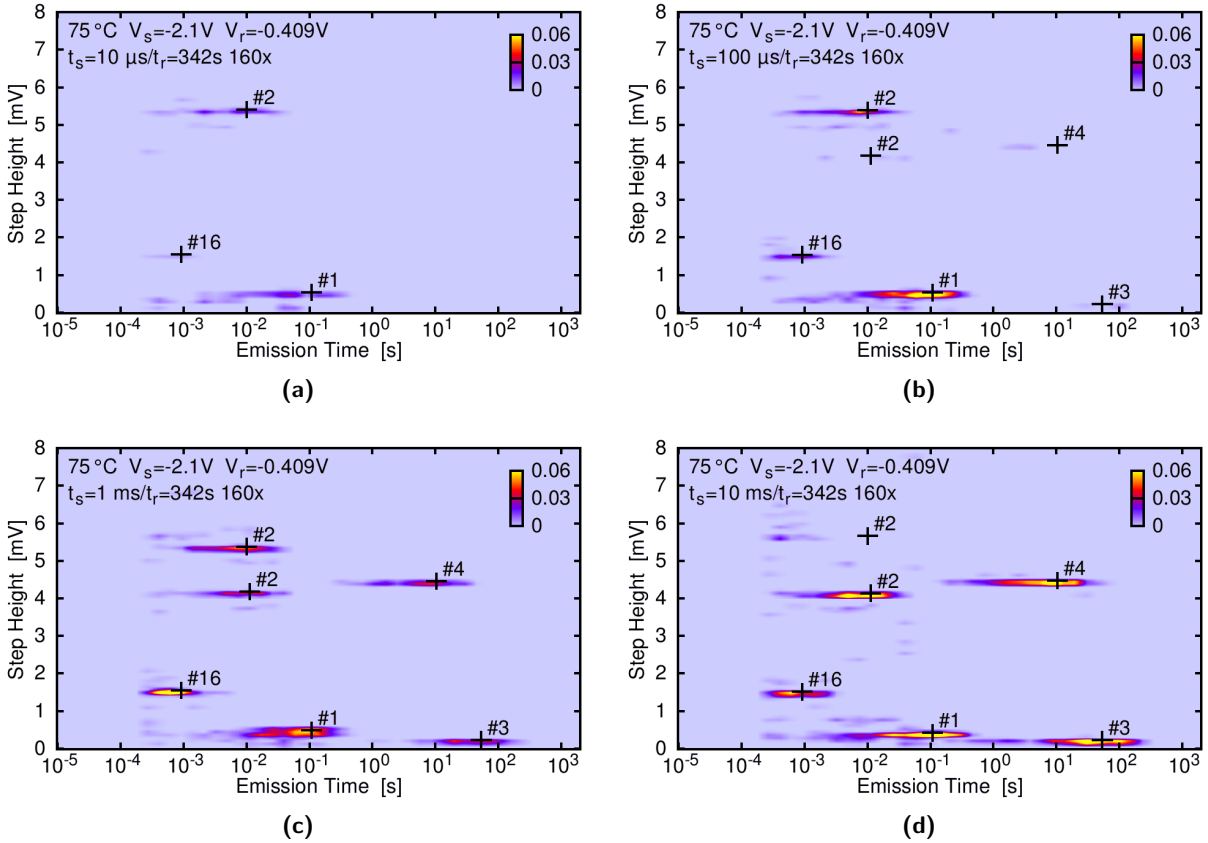


Figure 2.4: Spectral maps obtained after a PMOS transistor was stressed at $V_s = -2.1\,\mathrm{V}$ for an increasing stress time of **(a)** $t_s = 10\,\mu\mathrm{s}$, **(b)** $t_s = 100\,\mu\mathrm{s}$, **(c)** $t_s = 1\,\mathrm{ms}$, **(d)** $t_s = 10\,\mathrm{ms}$. As visible, the clusters get more intense at higher stress times, revealing the strong stress time dependency of the charge capture process. The detailed parameters are given in the top-left corner of the map. Line 1: temperature, stress voltage, recovery voltage. Line 2: stress time, recovery time, number of traces [11].

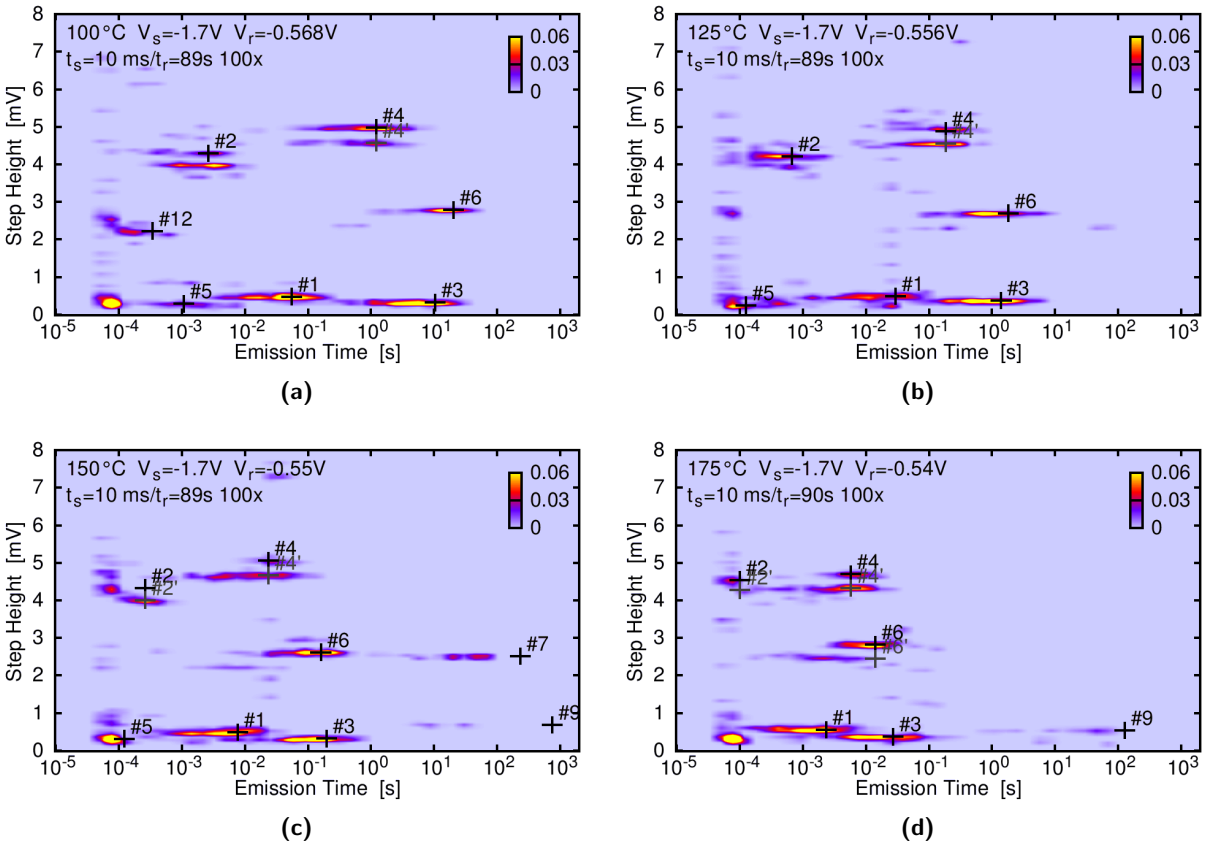


Figure 2.5: Spectral maps obtained after the same PMOS transistor was stressed at $V_S = -1.7\,V$ for $t_S = 10\,s$ at increasing temperatures of **(a)** $T = 100\,°C$, **(b)** $T = 125\,°C$, **(c)** $T = 150\,°C$, **(d)** $T = 175\,°C$. A unique shift of the clusters toward lower emission times is visible when the device temperate is increased. The detailed parameters are given in the top-left corner of the map. Line 1: temperature, stress voltage, recovery voltage. Line 2: stress time, recovery time, number of traces [11].

Fig. 2.4 shows four such spectral maps recorded with increasing stress time t_s, where five defects (#1, #2, #3, #4, #16) could be identified. The capture probability increases with increasing stress time which becomes noticeable by the more pronounced clusters or by the appearance of new ones in the spectral maps recorded at higher stress times. Another particular interesting observation is that defect #2 splits into two peaks, what is explained by the electrostatic interaction with another defect. Fig. 2.5 gives set of spectral maps recorded on the same device but at increasing temperatures. The strong shift to the left of the cluster gives a clear indication of the strong temperature dependency of the emission time. One defect (#12) shifts out of the experimental windows, where another defect (#9) appears. The figures demonstrate the remarkable advantages spectral maps have to offer for the investigation of single trapping.

Experimental Requirements for the Capture and Emission Times Summing up, the basic requirements for a defect to be accessible by the TDDS are:

- The capture time must be within voltage stress pulse width (usually 500 ns to 10 ks), i.e. $\tau_c \lesssim t_s$.
- The emission time is within the experimental window (usually 1 µs to 10 ks), i.e. $\tau_e \lesssim t_r$.
- The step height is large enough to be properly resolved by the measurement equipment (typically $\Delta V_{th} \geq 0.1\,\mathrm{mV}$).

To separate the clusters of different defects accurately:

- There should be at least a difference of $d \approx 0.5\,\mathrm{mV}$ for defects with similar emission time,
- or there should be a difference of two decades in the emission time for defects with similar step height d.

3 Cluster Analysis

In Section 1.2 the basic machine learning terminology has been introduced. The following chapter will give an overview of cluster analysis [1, 15, 16, 6] or clustering for short, which is one particular machine learning technique. The chapter starts with introducing some basic concepts, continues with summarizing the probability theory and statistics machine learning and probabilistic clustering algorithms are based on, and will than finally focus on the two clustering algorithms used in this work, namely the EM-algorithm and DBSCAN.

3.1 Introduction

The vast amount of publications available on cluster analysis suggests that this area of machine learning and data mining received a substantial amount of research activity. As a result there is a plethora of algorithms available exhibiting very different properties. This is motivated by the fact that there is no single clustering technique that is applicable to all of the different problem scenarios.

Terms and Definitions Clustering can be defined as the process of partitioning a given set of data points into a set of groups which are as similar as possible. This unsupervised process of classification contrasts a supervised classification process, the discriminant analysis. While in the former the categories are derived solely from the data, the latter requires a pattern set of already classified data to classify a, yet unclassified, pattern.

Commonly, in the context of clustering the training set is called a pattern set $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ and one of its vectorial elements a pattern. One scalar component $x_{n,d}$ of a pattern $\mathbf{x}_n$ is called a feature. The patterns are often combined into a pattern matrix $\mathbf{X}$ for convenience.

Applications Like other data analysis and statistic techniques, cluster analysis can be motivated from either a exploratory or confirmatory point of view. Tools for data processing, clustering proved to be useful to a number of different research communities, as well in many real-life problems. To illustrate this, following examples could be given [7]:

- In astronomy cluster analysis has been employed to find groups among astronomical objects like planetary nebulae and stars or to find objects with unusual properties compared to the rest in a huge amount of data.
- Different kind of data analysis problems in biology brought forth a whole new field of science called bioinformatics. Genes are particular regions of the DNA that influence the production of proteins, which is called gene expression. Cluster analysis is therein used to identify groups of genes with similar patterns of gene expression.
- In marketing research, cluster analysis has proven useful to find groups of consumers with similar needs, which is also called segmentation in that context.

- In psychiatry cluster analysis is useful as a method to divide patients into diagnostic categories according to their answers on a questionnaire.

Furthermore, clustering can be categorized according to different criteria:

Algorithm There is a manifold of different clustering algorithms available. Most of theme can be assigned to one of the following categories.

- **Distance-Based Algorithms**
  Distance-based algorithms require some measure of similarity or dissimilarity between two patterns.

  A dissimilarity measure is also called distance function d. It is often assumed that the distance function is non-negative $d(\mathbf{x}_1, \mathbf{x}_2) \geq 0$, that the distance of a sample to itself is zero $d(\mathbf{x}, \mathbf{x}) = 0$ and the distance is symmetric $d(\mathbf{x}_1, \mathbf{x}_2) = d(\mathbf{x}_2, \mathbf{x}_1)$. Sometimes it is beneficial to further require the triangle inequality $d(\mathbf{x}_1, \mathbf{x}_3) \leq d(\mathbf{x}_1, \mathbf{x}_2) + d(\mathbf{x}_2, \mathbf{x}_3)$ to be satisfied. The Manhattan, Euclidean or more generally the Minkowski or Mahalanobis distance are popular choices for the distance function.

  In contrast to the dissimilarity function, the similarity function s is often assumed to only take values between zero and one, $0 \leq s(\mathbf{x}_1, \mathbf{x}_2) \leq 1$. The similarity of a sample and itself is obviously $s(\mathbf{x}, \mathbf{x}) = 1$ and the similarity measure must again be reflexive $s(\mathbf{x}_1, \mathbf{x}_2) = s(\mathbf{x}_2, \mathbf{x}_1)$.

- **Probabilistic Algorithms**
  Probabilistic algorithms try to model the underlying generative process of the pattern set. This requires to assume a probabilistic model, the corresponding parameters and cluster membership of which are then typically estimated through a iterative process.

- **Density-Based Algorithms**
  In density-based methods either the number of samples or a smoother kernel density estimate is used as a measure. Grid-based methods can be thought as a special case of density-based methods, where the pattern space is divided into a grid-like structure. Clusters are assumed to be regions of higher density surrounded by regions of lower density.

- **Spectral Algorithms**
  This graph-theoretic approach is related to distance-based algorithms. Basically the adjacency matrix is derived from the similarity graph of samples and then the eigenvalues of the normalized or unnormalized Laplacian matrix is calculated. The eigenvectors corresponding to the top K eigenvalues are finally clustered with conventional techniques.

For the sake of completeness it should be mentioned that there are yet further techniques, like genetic algorithms, that do not fall in any of the categories above.

Data Type Not all clustering algorithms are equally applicable to a particular data type, what has to be taken into consideration during the selection of the algorithm.

- **Quantitative Features**
  Quantitative features are those that can be measured on a ratio scale.

  - continuous value (e.g. the weight of a person)

  - discrete value (e.g. the number of faulty devices in a sample)
  - interval value (e.g. a time interval)

- **Qualitative or Categorical Features**
  They consist of discrete features that can't be measured on a ratio scale.
  - ordinal value (e.g. the letters in the English alphabet)
    These are features that have an implicit meaningful order. One can obtain ordinal values from a continuous feature by binning the value into a discrete number of classes.
  - nominal value (e.g. a postal code)
    These are features that don't have a meaningful numeric representation.

- **Mixed Features**
  They are particular challenging because they need to be handled heterogeneously across different dimensions.

Again there are further more application-centric data types like text data, multimedia data, time-series etc.

Cluster Structure

Algorithms differ in how the resulting clusters are structured:

- **Partitioning Algorithms**
  Partitioning algorithms construct a flat partition of the pattern set into a set of K clusters. Some of these algorithms require the number of clusters K as an input parameter. In a problem scenario in which K is not known in advance, this type of algorithms have the disadvantage that K has to be determined by other means.

- **Hierarchical Algorithms**
  Hierarchical Algorithms create a hierarchical decomposition of the pattern set, which is graphically represented in a dendrogram. This has the advantage that the parameters determining the granularity of the grouping can be chosen after the actual clustering process. Thereby one can further distinguish between:
  - **Agglomerative Algorithms**
    Agglomerative algorithms start from a state where every pattern is assigned to a distinct cluster which are then successively merged until only one cluster is left or a stopping criterion is satisfied.
  - **Divisive Algorithms**
    Divisive Algorithms start from a state where every pattern is assigned to the same cluster and splitting is performed until a stopping criterion is satisfied.

  Both hierarchical methods have the shortcoming that a suboptimal merge or split can't be repaired in a later step. Furthermore it is often difficult to derive an appropriate stopping criterion.

Pattern Assignment

Cluster techniques can further be characterized according to the type of membership of patterns to a particular cluster.

- **Hard**
  Each pattern is assigned to exactly one cluster.

- **Soft or Fuzzy**
  Each feature is assigned a likelihood of belonging to a particular cluster.

One can achieve hard clustering by assigning each pattern resulting from soft clustering to the cluster with the highest likelihood.

Feature Usage

One can distinguish between two types of pattern usage:

- **Polythetic**
  In polythetic clustering decisions are based on a distance function that is a function of all the features. The majority of clustering techniques are of this type.

- **Monothetic**
  In that case a distance function is used sequentially such that only one feature is used at a time.

In view of the algorithms used in this work, the EM-algorithm, used as the main clustering mechanism, is a *probabilistic* technique and will be explained in Sections 3.3 and 4.2. A very minimalistic *distance-based* clustering approach is used during the initialization (Section 4.1) and defect prediction (Section 4.3). DBSCAN is a prototypical example of a *density-based* method and is used in this work for the initialization and treated in Sections 3.4 and 4.1. The EM-algorithm naturally implicates a *fuzzy* pattern assignment which is converted to a *hard* one in order to further process the data more conveniently. Using physical measurement data of reasonable resolution makes the features necessarily *quantitative* with *continuous* values and all algorithms can be considered to be *partitioning* and *polythetic*.

3.2 Statistical Basics

Probability theory and statistics play a vital role in the area of machine learning and not surprisingly in probabilistic clustering techniques used in this work. In the following section fundamental statistical theory, important in the context of machine learning, is informally summarized [3, 24, 23, 21]. The formal justification of some of the results in this section can be established by measure theory.

Probability Density Functions A continuous random variable X has a probability of zero taking exactly one value x, but one can define its probability density function or density function $p(x)$ such that

$$P(a < X < b) = \int_a^b p(x)\,\mathrm{d}x \tag{3.1}$$

gives the probability P that X takes a value in the interval $[a, b]$.

As it doesn't cause any confusion, no distinction between a random variable X and its value x is being made in this work most of the time.

To be a valid probability density the distribution has to satisfy the requirement

$$p(x) \geq 0 \quad \forall x, \tag{3.2}$$

and the normalization

$$\int_{-\infty}^{\infty} p(x)\,\mathrm{d}x = 1. \tag{3.3}$$

Any probability density function in this work is implicitly assumed to be normalized.

Several Continuous Variables In a machine learning context there are typically several continuous random variables that can be written as a D-dimensional vector $\mathbf{x} = (x_1, \ldots, x_D)^\top$. The properties above can be extended to a probability density function of several continuous variables, e.g. Eq. (3.2)

$$p(\mathbf{x}) \geq 0 \quad \forall \mathbf{x}, \tag{3.4}$$

and the normalization Eq. (3.3)

$$\int_{-\infty}^{\infty} p(\mathbf{x})\,\mathrm{d}\mathbf{x} = 1. \tag{3.5}$$

Cumulative Distribution Function The probability that the random variable X with the probability density function $p(x)$ takes a value in the interval $[-\infty, x]$ is given by the cumulative distribution function $q(x)$

$$P(X < x) = q(x) = \int_{-\infty}^{x} p(x')\,\mathrm{d}x'. \tag{3.6}$$

Joint and Marginal Distribution Similar to the one-dimensional case one can define the joint probability function $p(x, y)$ for the two random variables X and Y as the probability

$$P((X, Y) \in A) = \int\int_A p(x, y)\,\mathrm{d}x\,\mathrm{d}y \tag{3.7}$$

that (X, Y) lies in the region A.

The marginal distribution can be found from the joint distribution by integration

$$p(x) = \int_{-\infty}^{\infty} p(x, y)\,\mathrm{d}y. \tag{3.8}$$

In analogy to the similar rule for discrete random variables this is called the sum rule.

It should be noted that $p(x)$ and $p(x, y)$ actually constitute two different functions, which nevertheless are denoted with the same symbol p, in accordance to the machine learning literature.

Conditional Distribution The conditional distribution $p(x\,|\,y)$ of a random variable X given that $Y = y$ is

$$p(x\,|\,y) = \frac{p(x, y)}{p(y)}. \tag{3.9}$$

Because of symmetry reasons the above equation can be written as

$$p(x, y) = p(x\,|\,y)p(y) = p(y, x) = p(y\,|\,x)p(x) \tag{3.10}$$

which is also know as the product rule. In case the joint distribution $p(x, y)$ is not available, one can eliminate it with the above equation to get another form of the sum rule Eq. (3.8)

$$p(x) = \int_{-\infty}^{\infty} p(x\,|\,y)p(y)\,\mathrm{d}y. \tag{3.11}$$

The two random variables X and Y are independent if and only if

$$p(x, y) = p(x)p(y). \tag{3.12}$$

Bayes' Theorem Solving Eq. (3.10) for $p(x\,|\,y)$ and eliminating the marginal distribution $p(y)$ in the denominator with aid of Eq. (3.11), one obtains at Bayes' theorem in the form

$$\begin{aligned} p(x\,|\,y) &= \frac{p(y\,|\,x)p(x)}{p(y)} \\ &= \frac{p(y\,|\,x)p(x)}{\int_{-\infty}^{\infty} p(y\,|\,x')p(x')\,\mathrm{d}x'}. \end{aligned} \tag{3.13}$$

Bayes' theorem plays an extremely important role in a lot of machine learning algorithms. In case one has some observed data $\mathbf{x}$ and some model parameter $\boldsymbol{\theta}$ to be determined Eq. (3.13) can be written as

$$\begin{aligned} p(\boldsymbol{\theta}\,|\,\mathbf{x}) &= \frac{p(\mathbf{x}\,|\,\boldsymbol{\theta})p(\boldsymbol{\theta})}{p(\mathbf{x})} \\ &= \frac{p(\mathbf{x}\,|\,\boldsymbol{\theta})p(\boldsymbol{\theta})}{\int p(\mathbf{x}\,|\,\boldsymbol{\theta}')p(\boldsymbol{\theta}')\,\mathrm{d}\boldsymbol{\theta}'}. \end{aligned} \tag{3.14}$$

In this context the above quantities can be interpreted as follows:

$p(\boldsymbol{\theta})$ is the prior probability of $\boldsymbol{\theta}$ before any data has been observed.
$p(\mathbf{x}\,|\,\boldsymbol{\theta})$ is the likelihood function, which is a measure of how probable the observed data is for a value of the unknown parameter $\boldsymbol{\theta}$.
$p(\boldsymbol{\theta}\,|\,\mathbf{x})$ is the posterior distribution, which is a measure of the probability of the unknown parameter $\boldsymbol{\theta}$ after one has observed the data $\mathbf{x}$.

The integral in the denominator just serves as a normalization constant.

Expectation The expectation or expected value of a random variable X is defined as

$$\mathrm{E}[x] = \int_{-\infty}^{\infty} xp(x)\,\mathrm{d}x, \tag{3.15}$$

which can also be thought as the mean value of the distribution $p(x)$. The law of large numbers states that the average value of the samples of a distribution converges to the expected value of that distribution for a large number of samples. For some distributions (e.g. the Cauchy distribution) the expected value does not exist though [9].

More generally the expected value of a function of the random variable $f(x)$, which represents a new random variable itself, is

$$\mathrm{E}[f(x)] = \mathrm{E}[f] = \int_{-\infty}^{\infty} f(x)p(x)\,\mathrm{d}x. \tag{3.16}$$

This can be extended to a multivariate distribution $p(x, y)$ and a function $f(x, y)$

$$\mathrm{E}[f(x,y)] = \mathrm{E}[f] = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)p(x,y)\,\mathrm{d}x\,\mathrm{d}y, \tag{3.17}$$

or the expectation with respect to x only

$$\mathrm{E}_x[f(x,y)] = \mathrm{E}_x[f] = \int_{-\infty}^{\infty} f(x,y)p(x,y)\,\mathrm{d}x, \tag{3.18}$$

which is a function of of the random variable Y. A further important concept is the conditional expectation

$$\mathrm{E}_x[f(x)\,|\,y] = \mathrm{E}_x[f\,|\,y] = \int_{-\infty}^{\infty} f(x)\,p(x\,|\,y)\,\mathrm{d}x. \tag{3.19}$$

Variance and Covariance The variance of a random variable X is defined as

$$\mathrm{var}[x] = \mathrm{E}[x^2] - (\mathrm{E}[x])^2 \tag{3.20}$$

More generally the variance of a function of the random variable X is

$$\begin{aligned}\mathrm{var}[f] &= \mathrm{E}[(f(x) - \mathrm{E}[f(x)])^2] \\ &= \mathrm{E}[f(x)^2] - (\mathrm{E}[f(x)])^2.\end{aligned} \tag{3.21}$$

Finally the covariance of two random variables X and Y is given by

$$\begin{aligned}\mathrm{cov}[x,y] &= \mathrm{E}_{x,y}[(x - \mathrm{E}[x])(y - \mathrm{E}[y])] \\ &= \mathrm{E}_{x,y}[xy] - \mathrm{E}[x]\,\mathrm{E}[y].\end{aligned} \tag{3.22}$$

Similar results can be derived for discrete random variables.

Parameter Estimation A typical statistical and machine learning problem is to determine an estimate $\hat{\boldsymbol{\theta}}$ for the vector of parameters $\boldsymbol{\theta}$ of a distribution, given a set of N observations $\{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ combined into a $N \times D$ matrix $\mathbf{X}$. There are two approaches for such a parameter estimation problem: 1. Interval estimation, which gives a range of parameters together with the level of confidence and 2. point estimation which gives one value according some criteria. The least squares, the method of moments and the maximum likelihood estimator are examples of the latter.

To obtain a maximum likelihood estimator one starts with the definition of the likelihood

$$p(\mathbf{X}\,|\,\hat{\boldsymbol{\theta}}) = \prod_{n=1}^{N} p(\mathbf{x}_n\,|\,\hat{\boldsymbol{\theta}}), \tag{3.23}$$

which is a measure for the probability of the observed data set $\mathbf{X}$ for given model parameters $\hat{\boldsymbol{\theta}}$. Using Bayes' theorem one can get the opposite, i.e. a measure for the probability of the model parameters for a given set of observed data. Thus for finding the most likely model parameter one can maximize Eq. (3.23). This is equally achieved by maximization of the logarithm of the likelihood function

$$\ln p(\mathbf{X}\,|\,\hat{\boldsymbol{\theta}}) = \sum_{n=1}^{N} \ln[p(\mathbf{x}_n\,|\,\hat{\boldsymbol{\theta}})], \tag{3.24}$$

because the logarithm is a monotonic increasing function. This is advantageous in both analytical and numerical calculations. In the latter case the sum in Eq. (3.24) is less prone to numerical underflows than the product in Eq. (3.23). In a machine learning context the negative log likelihood is also called error function.

There are different quality criteria that apply to estimates, one of which is the bias. An estimator is called unbiased if

$$\mathrm{E}[\hat{\boldsymbol{\theta}}] = \boldsymbol{\theta}, \tag{3.25}$$

which means, that the expected value of the estimate $\hat{\boldsymbol{\theta}}$ equals the true value of the parameter $\boldsymbol{\theta}$. Another desirable property is that the variance of estimates should be as small as possible.

3.2.1 The Gaussian Distribution

All the mentioned properties are valid for any probability density function $p(x)$. One very important density function is the normal or Gaussian distribution. The univariate normal distribution [3] is defined as

$$\mathcal{N}(x\,|\,\mu, \sigma^2) = \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\left\{-\frac{1}{2\sigma^2}(x-\mu)^2\right\}, \tag{3.26}$$

where μ is the distributions mean value and σ^2 its variance. It can be shown, that the distribution satisfies the requirements Eqs. (3.2) and (3.3).

For a D-dimensional vector $\mathbf{x}$ the Gaussian distribution can be generalized to a multivariate distribution

$$\mathcal{N}(\mathbf{x}\,|\,\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{D/2}|\boldsymbol{\Sigma}|^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right\}. \tag{3.27}$$

The mean now takes the form of D-dimensional vector $\boldsymbol{\mu}$, the variance is then represented by $D \times D$ covariance matrix $\boldsymbol{\Sigma}$ and $|\boldsymbol{\Sigma}|$ denotes its determinant. The inverse of the covariance matrix is called the precision matrix

$$\boldsymbol{\Gamma} = \boldsymbol{\Sigma}^{-1}. \tag{3.28}$$

In comparison to the univariate distribution, the square in the exponential changes to the quadratic form

$$\Delta^2 = (\mathbf{x}-\boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu}). \tag{3.29}$$

The quantity Δ is called Mahalanobis distance.

The covariance matrix $\boldsymbol{\Sigma}$ To get the dependency of $\mathcal{N}(\mathbf{x}\,|\,\boldsymbol{\mu}, \boldsymbol{\Sigma})$ on $\mathbf{x}$ one has to investigate the exponent of Eq. (3.27). In the two-dimensional case the probability has a constant value on elliptical surfaces centered around the mean value $\boldsymbol{\mu}$. The orientation and the scaling factor of the axes of the ellipsoid correspond to the eigenvectors and eigenvalues of the inverse covariance matrix $\boldsymbol{\Sigma}^{-1}$ respectively.

Every matrix can be written as a sum of a symmetric and antisymmetric part [22] and it can be shown that the anitsymmetric part cancels out in Eq. (3.27). A real, symmetric matrix has only real eigenvalues λ_i and its eigenvectors $\mathbf{u}_i$ can be chosen to be orthonormal. So the matrix $\boldsymbol{\Sigma}$ can be expanded in terms of its D eigenvalues λ_i and eigenvectors $\mathbf{u}_i$ in the form

$$\boldsymbol{\Sigma} = \sum_{i=1}^{D} \lambda_i \mathbf{u}_i \mathbf{u}_i^\top . \tag{3.30}$$

By resolving to $\boldsymbol{\Sigma}^{-1}$ and substituting this into Eq. (3.29) one finally gets

$$\begin{aligned} \Delta^2 &= \sum_{i=1}^{D} \frac{\mathbf{u}_i^\top (\mathbf{x} - \boldsymbol{\mu})}{\lambda_i} \\ &= \sum_{i=1}^{D} \frac{y_i^2}{\lambda_i}. \end{aligned} \tag{3.31}$$

In two dimensions Eq. (3.31) can be interpreted as the defining equation of an ellipse, whose axes are oriented along the eigenvectors $\mathbf{u}_i$ and have the length $\sqrt{\lambda_i}$.

Fig. 3.1 shows the probability density function of a two-dimensional normal distribution for different values of the covariance matrix $\boldsymbol{\Sigma}$. With the identity matrix $\mathbf{I}$ the covariance matrix in Fig. 3.1a can be written as $\boldsymbol{\Sigma} = \sigma^2 \mathbf{I}$ and is called an isotropic variance. In this case the probability density function takes constant values around circles centered at the mean value $\boldsymbol{\mu}$. In Fig. 3.1b the covariance matrix is diagonal with the values σ_i^2 along the diagonal. The probability density functions takes constant values along axis-aligned ellipses. An example of the most general case in which $\boldsymbol{\Sigma}$ has off-diagonal elements is depicted in Fig. 3.1c.

Likelihood The likelihood of a Gaussian distribution for a set of observations combined into a matrix $\mathbf{X}$ can be written as

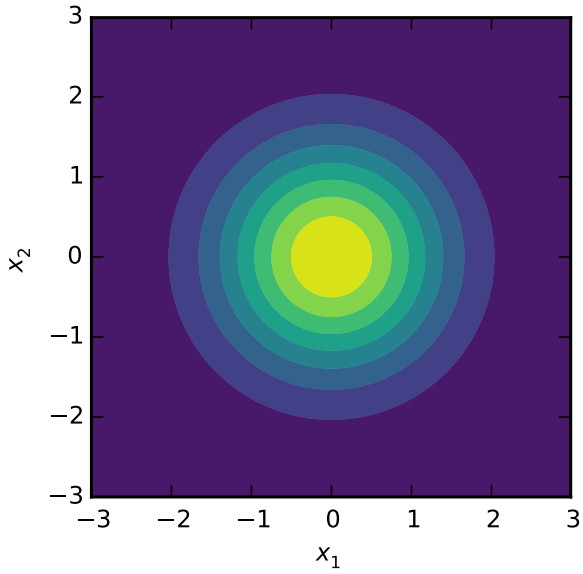
$$p(\mathbf{X}\,|\,\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \prod_{n=1}^{N} \mathcal{N}(\mathbf{x}_n\,|\,\boldsymbol{\mu}, \boldsymbol{\Sigma}) \tag{3.32}$$

and thus the log likelihood

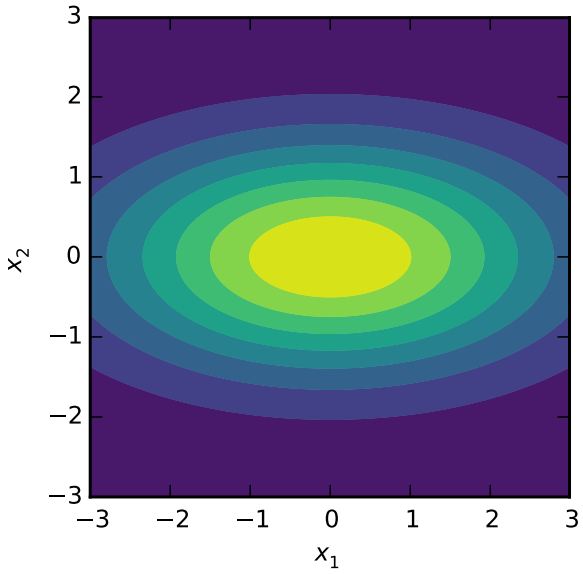
$$\ln p(\mathbf{X}\,|\,\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{n=1}^{N} \ln[\mathcal{N}(\mathbf{x}_n\,|\,\boldsymbol{\mu}, \boldsymbol{\Sigma})]. \tag{3.33}$$

One can analytically obtain a maximum likelihood estimate $\hat{\boldsymbol{\mu}}$ and $\hat{\boldsymbol{\Sigma}}$ of the mean value and the variance for the Gaussian distribution, by substituting Eq. (3.27) into Eq. (3.33) and finding the roots of the derivative. The result of this involved calculation is

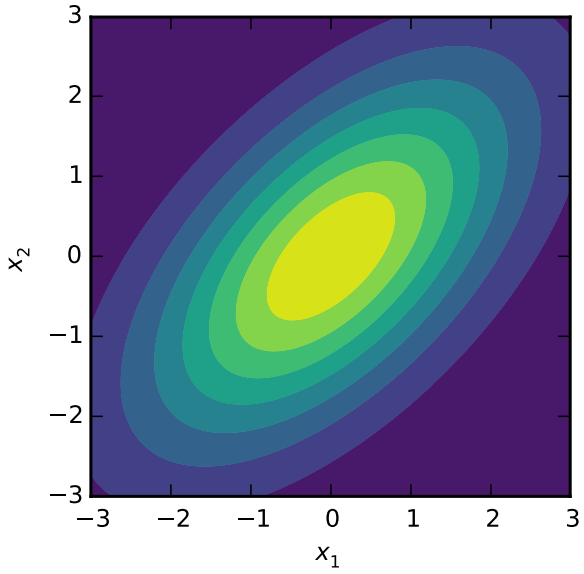
$$\hat{\boldsymbol{\mu}} = \frac{1}{N} \sum_{n=1}^{N} \mathbf{x}_n \tag{3.34}$$



(a) $\boldsymbol{\Sigma} = \left(\begin{smallmatrix} 1 & 0 \\ 0 & 1 \end{smallmatrix}\right)$



(b) $\boldsymbol{\Sigma} = \left(\begin{smallmatrix} 4 & 0 \\ 0 & 1 \end{smallmatrix}\right)$



(c) $\boldsymbol{\Sigma} = \mathbf{R}\left(\begin{smallmatrix} 4 & 0 \\ 0 & 1 \end{smallmatrix}\right)\mathbf{R}^\top$

Figure 3.1: The probability density function of a two-dimensional normal distribution for different values of the covariance matrix $\boldsymbol{\Sigma}$, $\boldsymbol{\mu} = (0, 0)^\top$, with an isotropic variance **(a)**, an covariance matrix with zero of-diagonal elements **(b)** and covariance matrix that is rotated with a rotation matrix $\mathbf{R} = \begin{pmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{pmatrix}$ where $\phi = \frac{\pi}{4}$ **(c)**.

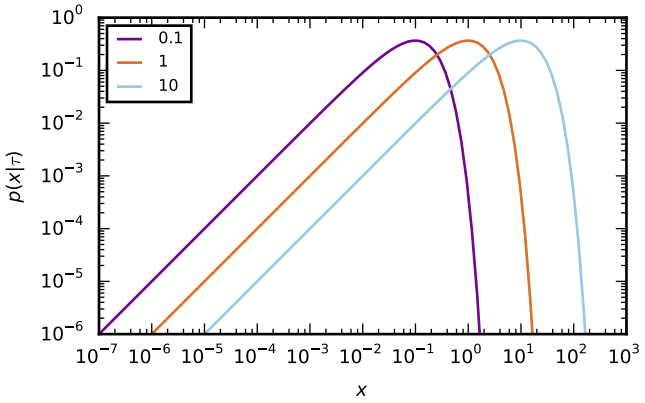


Figure 3.2: The probability density function $p(x\,|\,\tau)$ of the exponential distribution for different values of the parameter τ

and

$$\hat{\boldsymbol{\Sigma}} = \frac{1}{N}\sum_{n=1}^{N}(\mathbf{x}-\hat{\boldsymbol{\mu}})(\mathbf{x}-\hat{\boldsymbol{\mu}})^{\top}. \tag{3.35}$$

3.2.2 The Exponential Distribution

An exponential distributed random variable X has the probability density function

$$p(x\,|\,\lambda) = \begin{cases} \lambda\exp(-\lambda x) & x \geq 0 \\ 0 & x < 0. \end{cases} \tag{3.36}$$

The mean value can be calculated easily

$$\mathrm{E}[x] = \frac{1}{\lambda}. \tag{3.37}$$

The recording on a logarithmic time scale as it is done in the TDDS, can be thought as sampling of a random variable, which is a transformed version of the actual random variable [10]. By a variable transformation, it can be shown that this transformed variable has the probability density function

$$\widetilde{p}(x\,|\,\tau) = \frac{x}{\tau}\exp\left(-\frac{x}{\tau}\right). \tag{3.38}$$

Fig. 3.2 shows the exponential distribution for values of the parameter τ.

3.3 Expectation Maximization and Mixture Models

Some particular clustering techniques rely on the idea that the pattern set is drawn form a linear combination of basis distributions, so called mixture distributions [3, 15, 17]. In this case the clustering problem is to identify the number of components and each of its parameters. One well-studied technique for achieving the latter is the so-called expectation maximization (EM) algorithm.

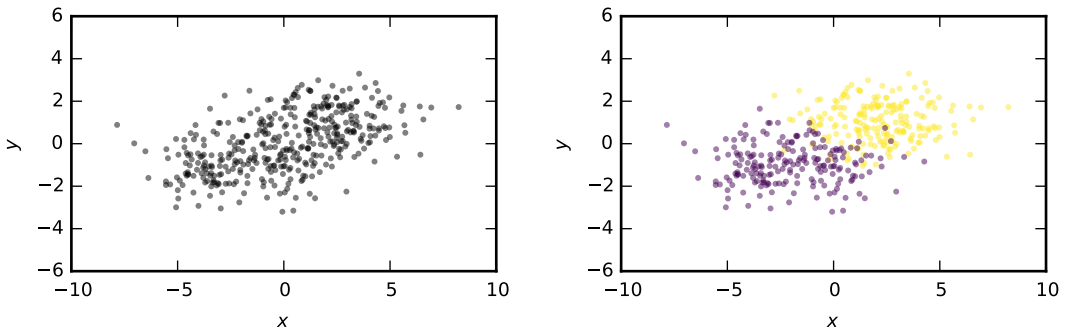


Figure 3.3: (a) 400 random samples from a Gaussian mixture of two components with $\pi_1 = \pi_2 = 0.5$, $\boldsymbol{\mu}_1 = (-2, -1)^\top$, $\boldsymbol{\mu}_2 = (2, 1)^\top$ and $\boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}_2 = \left(\begin{smallmatrix} 4 & 0 \\ 0 & 1 \end{smallmatrix}\right)$. **(b)** The component responsible for generating each sample [3].

So relying on a mixture distribution makes an implicit prior assumption about the form of the clusters, even though the parameters of the distribution are found by means of the algorithm. For a good result the probability density function of the underlying generative process has to be known, such a deep knowledge of the problem domain is not always available tough. The EM-algorithm shows poor results on data sets that have clusters of shapes that diverge significantly from that of the basis distribution. The fact that an arbitrary density function could be used as the basis function, makes this technique yet capable for a lot of situations.

Its analytic properties and frequent occurrence in real life problems make the Gaussian distribution a popular choice for the basis distribution. The superpositions of K Gaussian components $\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ yields the distribution

$$\begin{aligned} p(\mathbf{x}|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) &= \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \\ &= \sum_{k=1}^{K} p(k)\, p(\mathbf{x}|k). \end{aligned} \tag{3.39}$$

The mixture distribution has $3K$ individual parameters: the means $\boldsymbol{\mu} = \{\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K\}$, the covariance matrices $\boldsymbol{\Sigma} = \{\boldsymbol{\Sigma}_1, \ldots, \boldsymbol{\Sigma}_K\}$ and the so called mixing coefficients $\boldsymbol{\pi} = \{\pi_1, \ldots, \pi_K\}$.

Fig. 3.3a shows the distribution that results from drawing 400 random samples from a two-dimensional Gaussian mixture of two components with equal mixture coefficients. In Fig. 3.3b it can be seen, which of the two clusters actually contributed each point.

Integrating both sides of Eq. (3.39) with respect to $\mathbf{x}$ and interchanging the order of the integration and the summation, one obtains the condition

$$\sum_{k=1}^{K} \pi_k = 1. \tag{3.40}$$

If one requires that $\pi_k \geq 0$, π_k fulfill Eqs. (3.2) and (3.3) and this justifies considering π_k as the prior probability $p(k)$. Note, Eq. (3.39) has the form of a discrete version of the product rule Eq. (3.10).

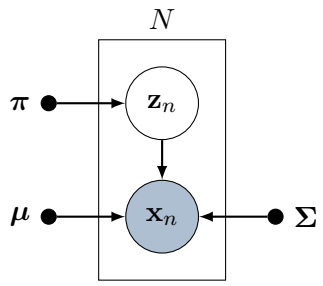


Figure 3.4: The probabilistic graphical model [3]

Latent Variables Trying to fit mixture model on a set of observed data points X is particularly tricky as it is not known which one of the components is responsible for contributing one sample. This situation can be made explicit by introducing a so-called latent variable $\mathbf{z}$, the k^{th} element of which is 1 if the component k of the mixture contributed the point and all the other elements are 0. Thereby, the mixing coefficient can be interpreted as the probability $\pi_k = p(z_k = 1)$. Eq. (3.39) can now be reformulated in terms of a latent variable

$$p(\mathbf{x}) = \sum_{\mathbf{z}} p(\mathbf{z})\, p(\mathbf{x}|\mathbf{z}). \tag{3.41}$$

Probabilistic Graphical Model A probabilistic graphical model is a diagrammatic representation of a probability distribution. In a Bayesian network, a special kind of probabilistic graphical model, the random variables are represented by nodes and the probabilistic relationships are expressed by arrows. Besides visualizing the structure and properties of a probabilistic model, these representation can be used to graphically achieve algebraic manipulations.

Fig. 3.4 gives the graphical model of having observed N samples from a mixture of K Gaussian components. The small nodes indicate the deterministic parameters of the components, the box is used as compact notation for the N independent and identical distributed samples. The shaded node points out that $\mathbf{x}_n$ is the observed variable, whereas the open node is used for the latent variable $\mathbf{z}_n$.

Responsibility The responsibilities $\gamma_k(\mathbf{x})$ are defined as the posterior probabilities and can be found with the aid of Bayes theorem 3.13

$$\begin{aligned} \gamma_k(\mathbf{x}) = p(k\,|\,\mathbf{x}) &= \frac{\pi_k\, \mathcal{N}(\mathbf{x}_n\,|\,\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{k'} \pi_{k'}\, \mathcal{N}(\mathbf{x}_n\,|\,\boldsymbol{\mu}_{k'}, \boldsymbol{\Sigma}_{k'})} \\ &= \frac{p(k)\, p(\mathbf{x}_n\,|\,k)}{\sum_{k'} p(k')\, p(\mathbf{x}_n\,|\,k')}. \end{aligned} \tag{3.42}$$

The responsibilities $\gamma_2(\mathbf{x})$ for the example of the two component Gaussian mixtures is depicted in Fig. 3.5.

Likelihood By extending Eq. (3.33) for the case of a Gaussian mixture, the likelihood is

$$\begin{aligned} p(\mathbf{X}\,|\,\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) &= \prod_{n=1}^{N} p(\mathbf{x}_n\,|\,\boldsymbol{\mu}, \boldsymbol{\Sigma}) \\ &= \prod_{n=1}^{N} \sum_{k=1}^{K} \pi_k\, \mathcal{N}(\mathbf{x}_n\,|\,\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k), \end{aligned} \tag{3.43}$$

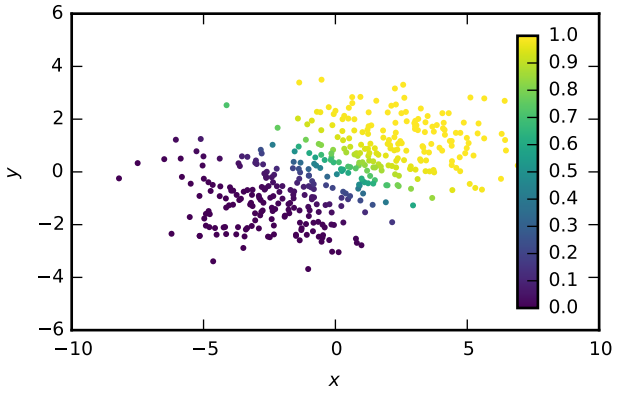


Figure 3.5: The responsibilities γ_2 of the distribution of Fig. 3.3 [3].

and the log likelihood therefore

$$\ln p(\mathbf{X}\,|\,\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\Sigma}) = \sum_{n=1}^{N} \ln \left[\sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n\,|\,\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k) \right]. \tag{3.44}$$

Maximum Likelihood Solution To get a maximum likelihood solution, one can again use the derivative of Eq. (3.44) with respect to $\boldsymbol{\pi}$, $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$. However, there is no closed-form solution, but an iterative procedure could be given instead. In this manner, in the E-step the responsibilities are calculated, which can be thought as the expectation value of the latent variable. In the M-step the parameters are re-estimated on the basis of the current responsibility, i.e. the parameters $\boldsymbol{\pi}$, $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ are chosen such that the expected value of the complete data log likelihood

$$\mathrm{E}_{\mathbf{Z}}[\ln p(\mathbf{X},\mathbf{Z}\,|\,\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\Sigma},)] = \sum_{n=1}^{N}\sum_{k=1}^{K} \gamma_k(\mathbf{z}_n)[\ln \pi_k + \ln \mathcal{N}(\mathbf{x}_n\,|\,\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k)] \tag{3.45}$$

is maximized. The result of such iterations on a pattern set drawn from a Gaussian mixture of two components can be seen in Fig. 3.6.

One problem that has to be taken care of are singular solutions, where one component with $\boldsymbol{\mu}_k$ converges to just on point $\mathbf{x}_n$. In this case the covariance matrix $\boldsymbol{\Sigma}$ gets very small and $\mathcal{N}(\mathbf{x}_n\,|\,\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k) \to \infty$. Another drawback of an EM-base clustering approach is that the number and the initial values of $\boldsymbol{\pi}$, $\boldsymbol{\mu}$, $\boldsymbol{\Sigma}$ have to been known in advance and need to be therefore obtained by an external mechanism. An inadequate choice of the initial values generally leads to slow convergence or even no convergence at all. An inappropriate number of clusters has the even more undesirable effect, of one cluster getting split up, what is illustrated in Fig. 3.7.

3.4 DBSCAN Clustering

In contrast to probabilistic clustering techniques, like the EM, the DBSCAN clustering technique doesn't make any assumption of the stochastic nature of the data source [1, 6, 19]. The driving assumption behind DBSCAN and other density-based clustering techniques is that clusters are high density areas separated from each other by sparse areas.

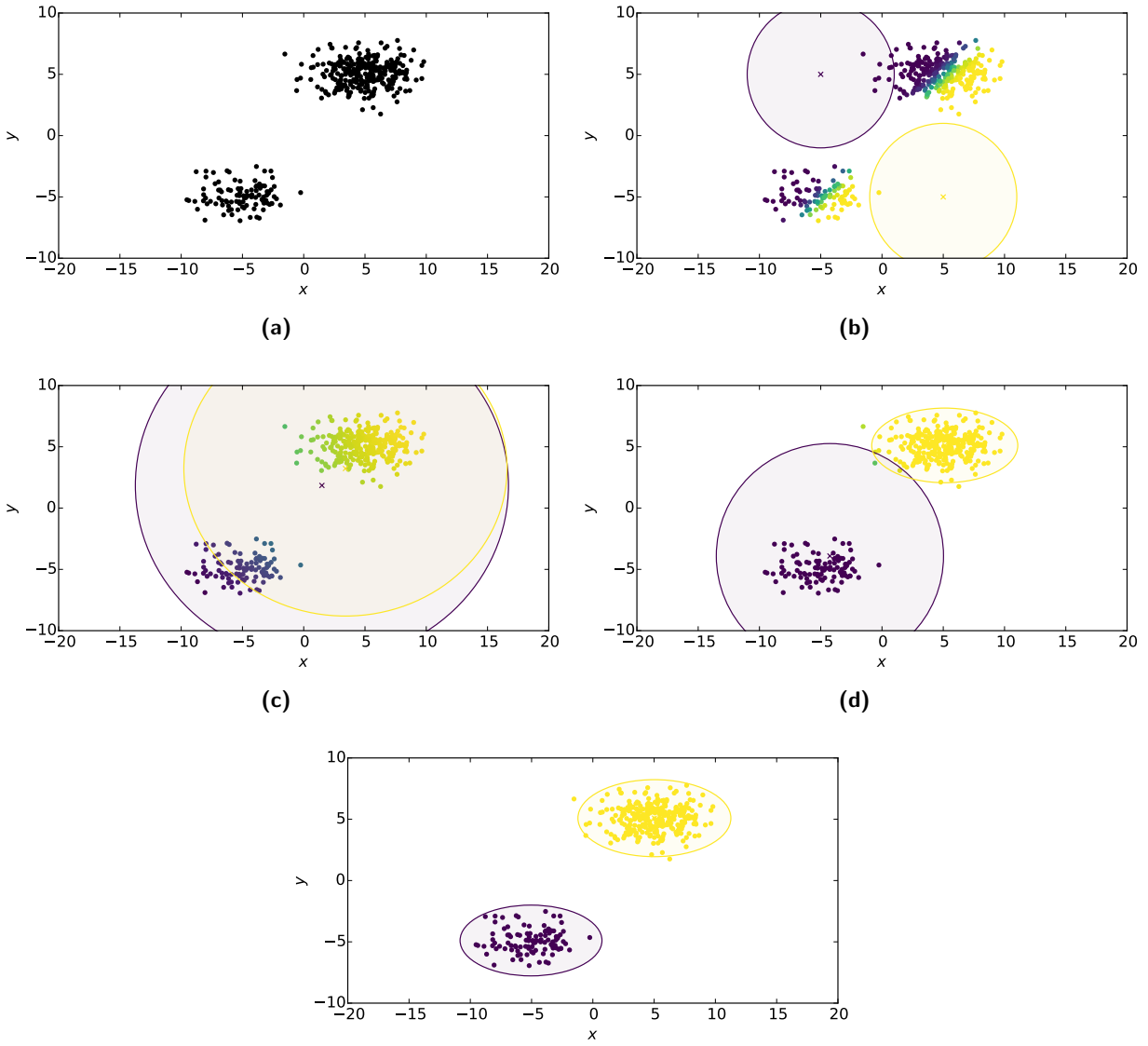


Figure 3.6: (a) $N = 400$ samples from a Gaussian mixture with $\boldsymbol{\mu}_{1,2} = \mp(5,5)^\top$, $\boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}_2 = \left(\begin{smallmatrix} 4 & 0 \\ 0 & 1 \end{smallmatrix}\right)$, $\boldsymbol{\pi} = (0.2, 0.8)^\top$. **(b)** The initial values of the mean values $\boldsymbol{\mu}$ and the covariance matrices $\boldsymbol{\Sigma}$ are represented by the colored cross and the ellipses respectively. The responsibility γ_2 is color-coded as in Fig. 3.3. **(c-e)** The evolution of the estimation of the parameters during iterations of the EM algorithm after the 1st, 3rd, 6th and 9th iteration. Convergence is already reached after the 9th iteration. The covariance matrices are forced to be diagonal. [3]

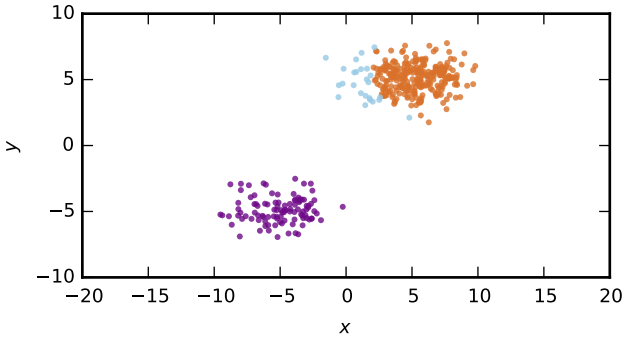


Figure 3.7: The result of the EM-based clustering of the same distribution as in Fig. 3.6, but now with three cluster centers initialized at $\boldsymbol{\mu}_{1,2} = \mp(5,5)^\top$ and $\boldsymbol{\mu}_3 = (0,0)^\top$. The result converges to a solution where on Gaussian component is split up between two clusters. 27 iterations are necessary which is considerably more than in the case of initializing only two clusters.

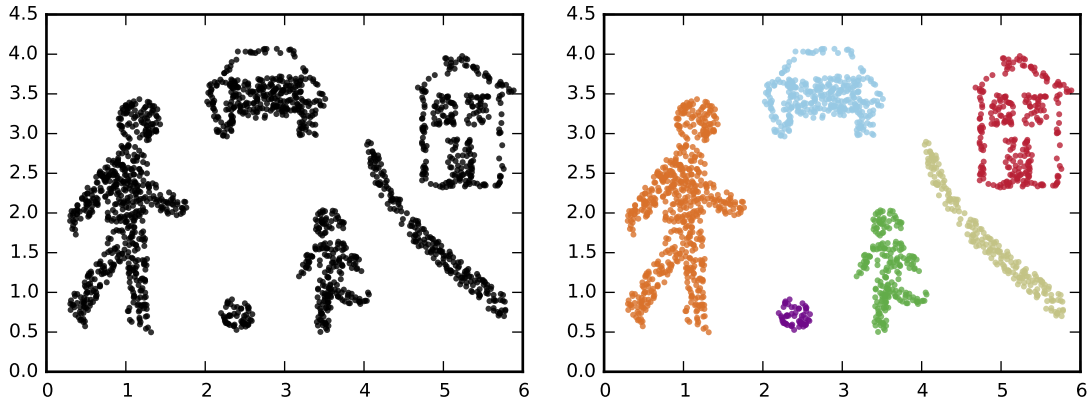


Figure 3.8: **(a)** A distribution of $N = 1712$ samples. **(b)** The result of DBSCAN clustering with $\epsilon = 0.3$.

Where probabilistic techniques tend to break up or merge clusters of irregular shape, DBSCAN shows good results with clusters of any shape. Fig. 3.8 shows a distribution that could not reasonably modeled by a Gaussian mixture model, however, using DBSCAN the clusters are properly resolved. Another advantage of DBSCAN is its simplicity and its low computational efforts compared to other clustering techniques.

One of its major weaknesses is that the algorithm expects clusters of similar and homogenous densities. With clusters having different densities and intermediate noise it could be difficult or even impossible to find parameters that neither merge some clusters nor split them up. This problem is illustrated in Fig. 3.9 for a Gaussian mixture of three components with different variances. On one hand, clustering with ϵ lower than a certain value will merge the violet and orange cluster Fig. 3.9a. On the other hand, clustering with ϵ larger than a certain value will start to break up the low density areas of the blue cluster, which can be seen in Fig. 3.9b. In this case the parameters of the components are chosen such that there is no global value ϵ that avoids both merging the dense clusters and splitting up the sparse one. It has to be emphasized that the problem of splitting is not solely caused by the different variances used for the components, but rather because the samples from a normal distribution, like most other practically important probability density functions, are more scarce in areas more distance from its mode. In [2] an algorithm has been published, that can be considered as an enhanced version of DBSCAN that solves the mentioned problem. Instead of a fixed cluster membership an augmented ordering of the pattern set that represents its density-based clustering structure can be obtained. Another

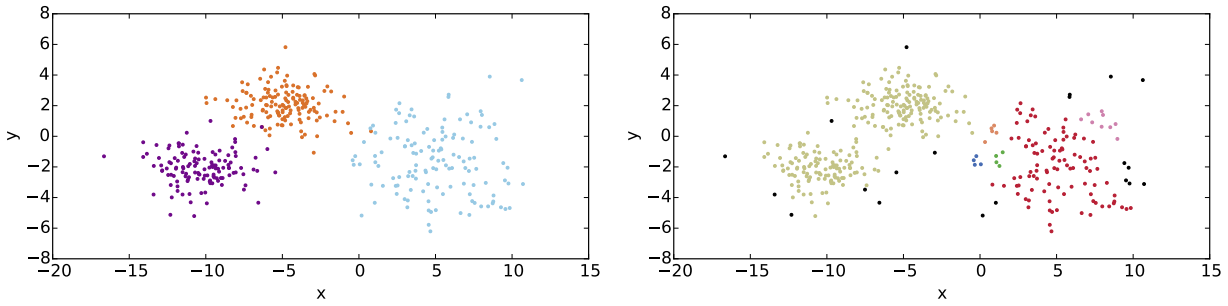


Figure 3.9: (a) $N = 400$ samples from a Gaussian mixture of three components $\boldsymbol{\mu}_1 = (-10, -2)^\top$, $\boldsymbol{\mu}_2 = (-5, 2)^\top$, $\boldsymbol{\mu}_3 = (5, -2)^\top$, $\boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}_2 = \left(\begin{smallmatrix} 4 & 0 \\ 0 & 1 \end{smallmatrix}\right)$, $\boldsymbol{\Sigma}_3 = \left(\begin{smallmatrix} 8 & 0 \\ 0 & 4 \end{smallmatrix}\right)$ and $\boldsymbol{\pi} = (0.34, 0.33, 0.33)^\top$. The violet and orange components are very close, but nevertheless clearly distinct clusters are formed. As visible, the blue component has a bigger variance than the others. **(b)** The result of DBSCAN clustering with $\epsilon = 1$ and $N_{\min} = 4$. It can be seen that most of the points from the two close components are merged into only one cluster by DBSCAN, but some of the remote points in the low density areas of the high variance cluster are divided into multiple clusters.

issue with DBSCAN is that the result is very sensitive to the clustering parameters. A method for choosing the parameter ϵ has been proposed that tries to derive its value from the sparsest cluster. This method however needs some user interaction and can't therefore be used as an unsupervised solution.

Definitions The neighborhood of a point $\mathbf{p}$ of the complete pattern set $\mathbf{X}$ is defined as the set of samples with a distance smaller than ϵ

$$\mathrm{N}_\epsilon(\mathbf{p}) = \{\mathbf{q} \in \mathbf{X} \mid d(\mathbf{p}, \mathbf{q}) \leq \epsilon\}. \tag{3.46}$$

In its simplest form the distance function $d(\mathbf{p}, \mathbf{q})$ corresponds to the euclidean distance. A central concept of DBSCAN is that of a core sample, which is a sample that has at least $N_{\min}$ samples in its neighborhood

$$|\,\mathrm{N}_\epsilon(q)| \leq N_{\min}. \tag{3.47}$$

A none-core sample is a sample in the neighborhood of a core point, but not a core point itself. These concepts are visualized in Fig. 3.10a. A point $\mathbf{p}$ is directly density-reachable from $\mathbf{q}$ if $\mathbf{q}$ is a core point and $\mathbf{p}$ is in the neighborhood of $\mathbf{q}$. A point $\mathbf{p}_n$ is density-reachable from $\mathbf{p}_1$ if there is a chain of points $\mathbf{p}_1, \ldots, \mathbf{p}_n$ such that $\mathbf{p}_{i+1}$ is directly density-reachable from $\mathbf{p}_i$. Two points $\mathbf{p}_1$ and $\mathbf{p}_2$ are called density-connected if there is a third point $\mathbf{p}_3$, both points are density-reachable from. A cluster $\mathbf{C} \subseteq \mathbf{X}$ is the biggest set of density-connected points.

Algorithm The algorithm can be roughly summarized as follows [28]: All points not visited yet is iterated over and marked visited. If a point is a non-core point it is preliminary classified as noise, otherwise a new cluster is created, the point is added to the cluster and its neighbors are further processed. Any unvisited neighbor is checked to be a core point itself and its neighborhood is added to the list of neighbors as well, if this is the case. Finally the current neighbor is added to the cluster too if it is not part of any cluster yet and the next neighbor is processed.

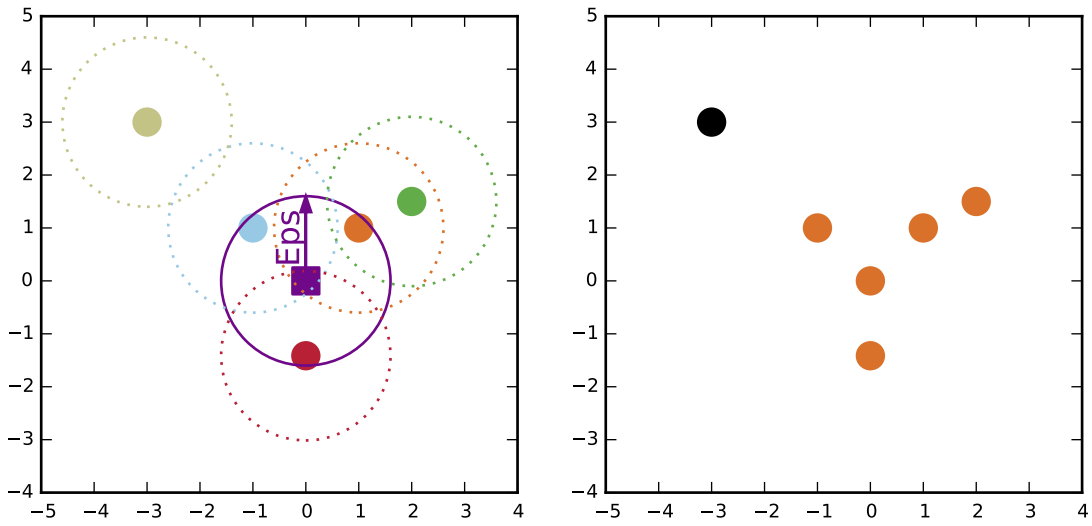


Figure 3.10: (a) The core sample is represented by the violet square, the non-core samples by dots, the neighborhood by the corresponding circles. For $N_{\min} = 3$ the violet sample is the only core sample, as no other circle contains at least 3 other samples. **(b)** The result contains only the orange cluster as the black dot is not connected to any core sample and is therefore classified as noise.

More formally the algorithm can be represented by following pseudocode:

```
procedure DBSCAN(pattern_set)
    for each point in pattern_set do
        if not point visited then
            mark point as visited
            neighbors ← get_neighborhood(pattern_set, point)
            if size of neighbors < MinPts then
                mark point as noise
            else
                clusters ← expand_cluster(pattern_set, point, neighbors)
function EXPAND_CLUSTER(pattern_set, point, neighbors)
    cluster ← new cluster
    add point to cluster
    for each neighbor in neighbors do
        if neighbor not visited then
            mark neighbor as visited
            neighborsneighbors ← get_neighborhood(pattern_set, neighbor)
            if size of neighborsneighbors ≥ MinPts then
                append neighborsneighbors to neighbors
        if neighbor not member of any cluster then
            add neighbor to cluster
            unmark neighbor as noise
    return cluster
```

4 Implementation

The overall flow chart of the TDDS analysis software framework is shown in Fig. 4.1. All the components, excluding those marked with a star, have been implemented in the course of this diploma thesis and will be discussed in detail in the subsequent sections. There is some additional input/output to the file system taking place in the legacy codes which is not shown for simplicity. The purpose of this additional input/output is to cache computation intensive parts or serve as an interface between different components.

In the following the term block is used for a set of measurements with the same stress and recovery conditions, i.e. the stress and recovery times, biases and temperature.

General Design The following general design decisions have been taken or the software patterns have been followed:

- The analysis process should work both, in a batched and an interactive workflow. I.e., by exploiting machine learning algorithms the software should automatically make suitable decisions and predictions, but also provide the possibility to take manual action, if necessary.
- The software should be modular in the sense that specific components should be easily exchangeable in future versions of the software.
- The whole process to be implemented should, to some degree, reassemble the work flow as it is done manually at the moment.
- Furthermore, depending on the stress and recovery scenario, the real-time measurements to obtain TDDS could be very time-consuming. For instance, obtaining 100 traces with a typical configuration of $t_\mathrm{s} = 10\,\mathrm{s}$ and $t_\mathrm{r} = 1\,\mathrm{ks}$ results in a measurement time of more than 28 h. For this reason it is desirable that the analysis is done on-the-fly, so that the preliminary results could by inspected by the user.

Fig. 4.1 makes it very clear that in such a long pipeline of processing, a inferior result of just one single component results in suboptimal values of the extracted parameters or even a subsequent component to fail. The fact that the processing of the TDDS blocks in the first loop is independent of all other blocks makes this chain an obvious candidate for parallelization.

4.1 Cluster Initialization

As pointed out in Section 3.3, the EM algorithm itself is neither able to find a suitable number of clusters nor their initial positions. Therefore these parameters have to be determined by other means. The purpose of this initialization step is to generate the preliminary clusters that are used for initializing the subsequent EM step. The quality of these initial values is essential for a fast convergence (if convergence is reached at all) and a good result of the EM algorithm. In its current implementation it's prohibitive to initialize much more than 10 clusters, which nevertheless should be enough for the typical dataset used during the TDDS analysis.

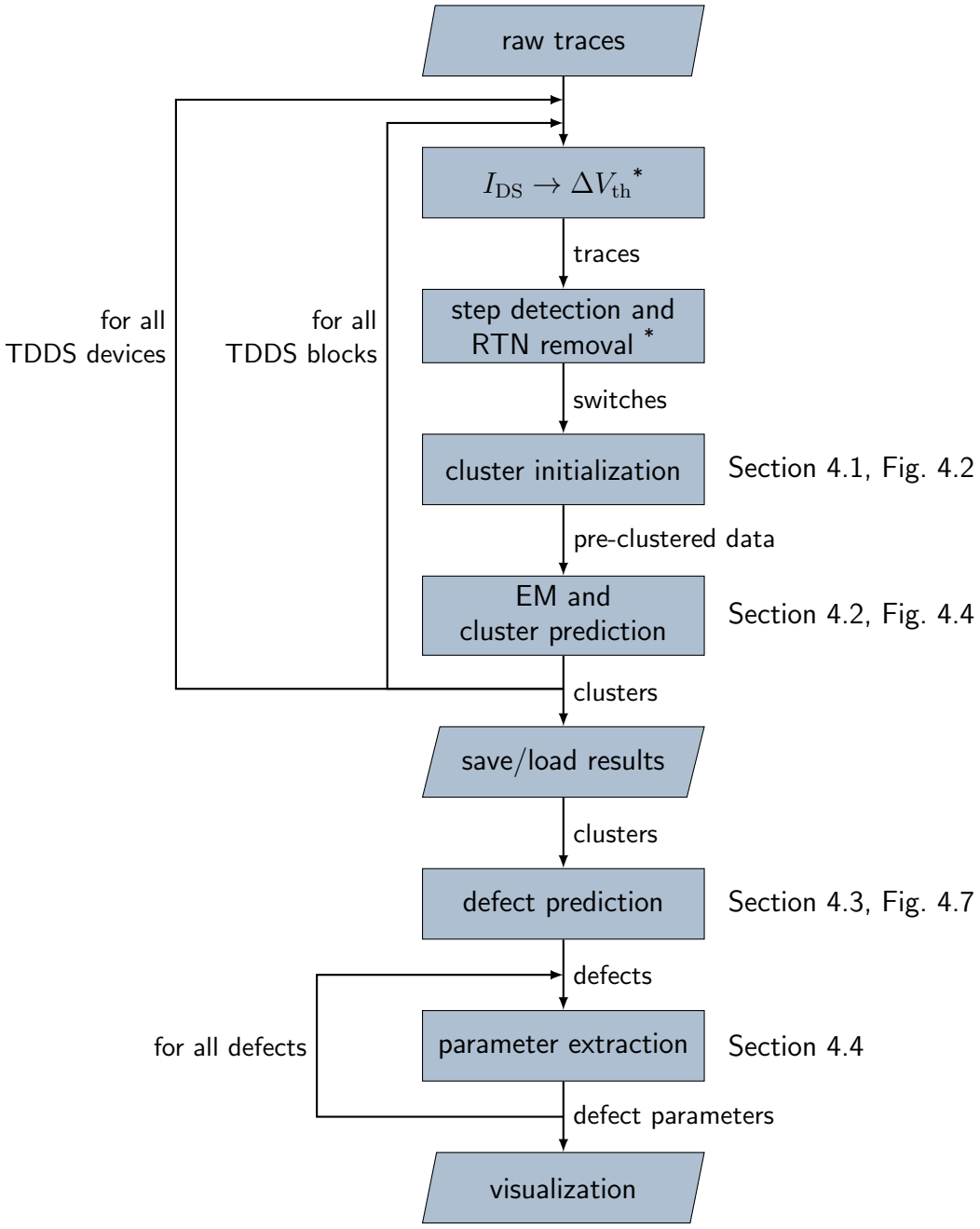


Figure 4.1: The flow chart of the framework to analysis single trap defects using TDDS. Components marked with * are already existing components provided by the IuE.

Since defects are specific to a particular device, different physical devices need certainly be initialized independently. But for different measurements on the same physical device three different ways of finding initial clusters are conceivable:

- Use the distributions of all blocks to generate one set of initial positions used for all blocks during the EM.
- Group together blocks that only differ in their stress time, and use these distributions to generate one set of initial positions used for all blocks of that group. This is motivated by the fact that in the simple two-state model, only the number of emission events is changed as long only the stress time is changed.
- Treat every block independently.

Rationale for Working on a Per-Block Basis Actually all three options have been investigated but the best result has been obtain with an initialization done for each block separately. If the stress time in one block is much lower than the capture time of the single defect, there are no or only very few emissions in that block. By treating the blocks independently the few emission events are either considered as noise and discarded or erroneously assigned to a nearby cluster during the EM step. However, in the first two methods, by reusing the same initial cluster positions across multiple blocks, it is more likely that the single emission events are assigned correctly, if the defect shows a more pronounced cluster in a block at higher stress time. Another advantage in using the same initial cluster positions for multiple blocks is that the cluster affiliations are maintained during the EM step and are then available after finishing processing the entire measurement data from a single device. If the clusters converge to the emissions events of the same defect in all blocks, an additional defect prediction step will be unnecessary.

The mentioned advantages of using the same initial values for multiple blocks are countered by certain disadvantages. The fact that the defect parameters considerably change as a result of altering the stress biases and times makes it impossible to obtain good initial values that are generally adequate. Even worse than inappropriate values for $\boldsymbol{\pi}$, $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ is a wrong number of initialized clusters which leads to results where the emissions of one defect are divided between the surplus cluster and another one (c.f. Fig. 3.7). The reason of such a surplus cluster is that the defect has no emission events because the stress time in one specific block is much smaller than its capture time or a volatile defect is involved.

Description of the Implementation Again, different implementations have been evaluated and overall satisfying results have been achieved with an implementation that is presented in Fig. 4.2. The initialization code receives its data, which consists of an array of emission times τ_e with respect to the beginning of the data acquisition and the step heights ΔV_th, from the previously performed step detection and RTN removal step.

First the complete distribution is clustered with DBSCAN. Every cluster, the number of points of which is surpassing a certain size `max_points`, is then further processed: First the kernel density estimation (KDE) is calculated and the local maxima are determined and its values are normalized to the biggest maximum. Any maximum smaller than a certain `discard_factor` is discarded and the remaining ones are used as new cluster centroids. Finally, every point of the distribution is assigned to the cluster with the smallest Mahalanobis distance with respect to a certain diagonal

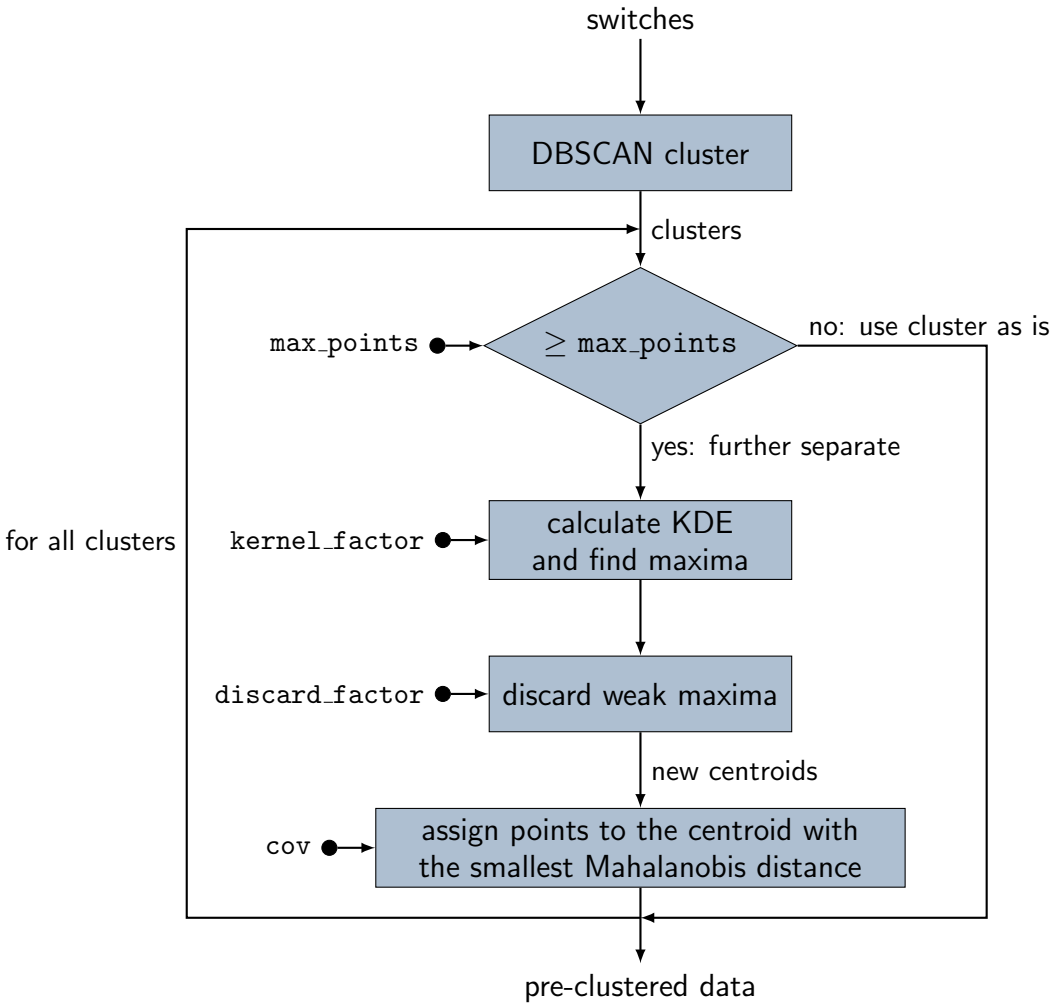


Figure 4.2: The flow chart of the cluster initialization. Configuration parameters are marked with a solid dot.

covariance matrix. For the KDE Gaussian basis functions are used and the bandwidth h is chosen according to Scott's Rule [20]

$$h = \kappa N^{-1/(D+4)}, \tag{4.1}$$

where $D = 2$ is the number of dimensions and N is the number of emissions in that particular block and κ is a user-configurable constant (`kernel_factor`). Clusters from the DBSCAN with less or equal `max_points` emissions are not further separated.

Finally a list of clusters is returned. This allows for initializing not only the vectors of the mean values $\boldsymbol{\mu}$, but also the vectors of the mixing coefficients $\boldsymbol{\pi}$ and the standard deviations $\boldsymbol{\sigma}$ during the EM-algorithm with very close estimates, which typically results in fast convergence.

Rationale for the Two Step Approach This two step approach tries to mitigate the weakness that both of the techniques have on their own. On one hand, as explained in Section 3.4, DBSCAN tend to either merge clusters together if the intermediate noise is to high or the clusters come to close, or conversely split them up if they get to sparse. On the other hand, if there are clusters of very high density close enough, but yet clearly separated, the maximum of the KDE could be so weak that it is undetectable. The reason is that the contribution of a few points of a small cluster to the KDE could be much less than the tail of a very high density cluster. This issue is illustrated in Fig. 4.3.

In this implementation a rather low value for ϵ is used that tend to merge clusters together as they get split again anyway in the second step if they exhibit some clear density peaks. This combined approach proved to be relative insensitive to the parameters and showed satisfying results in the majority of cases.

4.2 Clustering of the Trap Emissions

Chapter 3 gave a short overview of different clustering algorithms and covered two algorithms in detail, namely DBSCAN and the EM-algorithm. Section 4.1 showed how the former is used together with KDE and the Mahalanobis distance to determine the initial parameters. These parameters are now used as an input for the actual clustering mechanism, the EM-algorithm the implementation of which is described in the following.

Rationale for Using a EM-Algorithm Base on Mixture Models Several clustering techniques have been investigated during this work with only a moderate quality of the results compared to the EM-algorithm. There are several arguments in favor of using the EM-algorithm as the main clustering technique: By using a probabilistic model for the clusters accordingly, there is a direct link to the physical model of the defect. This is yet very flexible, as an arbitrary function could be used as the likelihood function. So the estimates of the model parameters are available directly after clustering, provided that the algorithm has converged. In addition to the model parameters, the algorithm provides the responsibilities $\boldsymbol{\gamma}$, which serve as a measure of how likely one particular point belongs to one particular cluster and could be used to give an error estimate in the subsequent defect parameter extraction.

Nonetheless, building upon a mixture of base distributions entails, that the probability density function of the underlying generating process has to be known in advance. As the probabilistic defect models are able to explain the observed data reasonably well this does not seem to be a

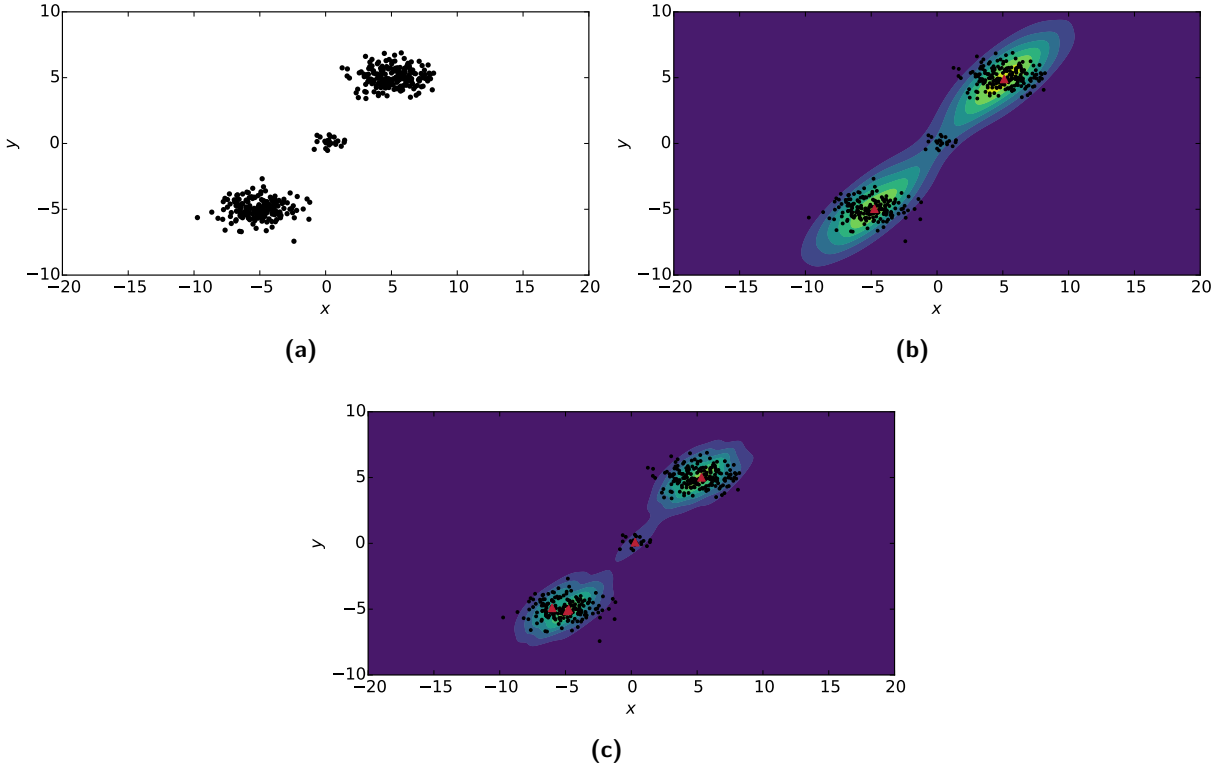


Figure 4.3: (a) A Gaussian mixture with three component $\boldsymbol{\mu}_{1,3} = \mp(5,5)^\top$, $\boldsymbol{\mu}_2 = (0,0)^\top$, $\boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}_3 = (2,0.5)^\top$, $\boldsymbol{\Sigma}_2 = (0.5,0.125)^\top$, $\boldsymbol{\pi} = (0.48,0.04,0.48)^\top$ and $N = 400$ samples. The KDE with a bandwidth according to Eq. (4.1) with $k = 1.2$ **(b)** and $k = 0.7$ **(c)**. The local maxima of the KDE are marked with red triangles. Some very weak local maxima are omitted for clarity. In (b) the bandwidth is to large for the second component to cause a maximum in the KDE. With the reduced bandwidth in (c) the second component is indeed detectable but the KDE over the first component exhibits already two maxima.

problem anymore. Besides, the mixture model makes the implementation implicitly tied to a special distribution and makes it therefore difficult to re-use the same implementation on other problem domains. Yet another main issue of the EM-algorithm is its numeric complexity compared to other clustering algorithms, especially when considering high number of dimensions. This however doesn't seem to be an issue in the TDDS analysis as analyzing each block separately results in only two dimensions and typically less than ten clusters, each well below 1000 points.

Some considerations have been made of regarding the stress and recovery parameters as additional deterministic variables in the likelihood and doing a simultaneous EM across all blocks. There are, however, many arguments against doing so. The iterations are substantially more numerically demanding and very slow convergence is to be expected. To get a satisfying result the quantitative influence of all stress and recovery parameters on the likelihood have to be known analytically. Volatile and interacting defects cause additional intricacies that need to be handled manually. Moreover there is no easy way to manually interact with this kind of processing.

Description of the Implementation In all the subsequent calculations statistical independence between τ_e and ΔV_th is supposed and the following probability density function

$$p(\mathbf{x}\,|\,\boldsymbol{\mu},\boldsymbol{\sigma},k) = p(\mathbf{x}\,|\,\mu_k,\sigma_k) = f(\delta\mu_{\tau,k},\delta\mu_{V,k})\frac{\tau_\mathrm{e}}{\mu_{\tau,k}}\mathrm{e}^{-\frac{\tau_\mathrm{e}}{\mu_{\tau,k}}}\,\mathcal{N}(\Delta V_\mathrm{th}\,|\,\mu_{V,k},\sigma_k) \tag{4.2}$$

is used. This probability density function incorporates the exponential distribution over τ_e and the normal distribution over the ΔV_th. The additional factor $f(\delta\mu_{\tau,k},\delta\mu_{V,k})$ is a function of the difference of the estimated mean value of τ_e and ΔV_th during the EM-iterations from their initial values. The purpose of this is to avoid a drift-away of the mean values in case there is some prior knowledge of their approximate values, or to avoid splitting up clusters in case of sub-optimal initialization.

The flow chart of the EM-algorithm is given in Fig. 4.4. In the following it is supposed that the EM algorithm receives K clusters from the initialization and that the observed data $\mathbf{X} = \{\mathbf{x}_1,\ldots,\mathbf{x}_N\}$ is made up by the N overall data points in all clusters. Each data point $\mathbf{x}_n = (\tau_{\mathrm{e},n},\Delta V_{\mathrm{th},n})^\top$ is a two-dimensional vector with one component representing the emission time τ_e and the other the step height ΔV_th. The initial mean values $\boldsymbol{\mu} = \{\boldsymbol{\mu}_1,\ldots,\boldsymbol{\mu}_K\}$ are calculated as the arithmetic mean values with respect to τ_e and ΔV_th of every cluster received from the initialization. As the exponential distribution has only one parameter, the standard deviations $\boldsymbol{\sigma} = \{\sigma_1,\ldots,\sigma_K\}$ are initialized to the standard deviations with respect to ΔV_th only. The mixing coefficients $\boldsymbol{\pi} = \{\pi_1,\ldots,\pi_K\}$ are initialized with

$$\pi_k = \frac{N_K}{N}, \tag{4.3}$$

where N_K is the number of points of the cluster K.

The process continues with the E-step by calculating the matrix of the responsibilities where the responsibility of the component k for the data points $\mathbf{x}_n$ is given by

$$\gamma_{n,k} = \frac{\pi_k\,p(\mathbf{x}_n\,|\,\boldsymbol{\mu}_k,\sigma_k)}{\sum_{k'=1}^{K}\pi_{k'}\,p(\mathbf{x}_n\,|\,\boldsymbol{\mu}_{k'},\sigma_{k'})}. \tag{4.4}$$

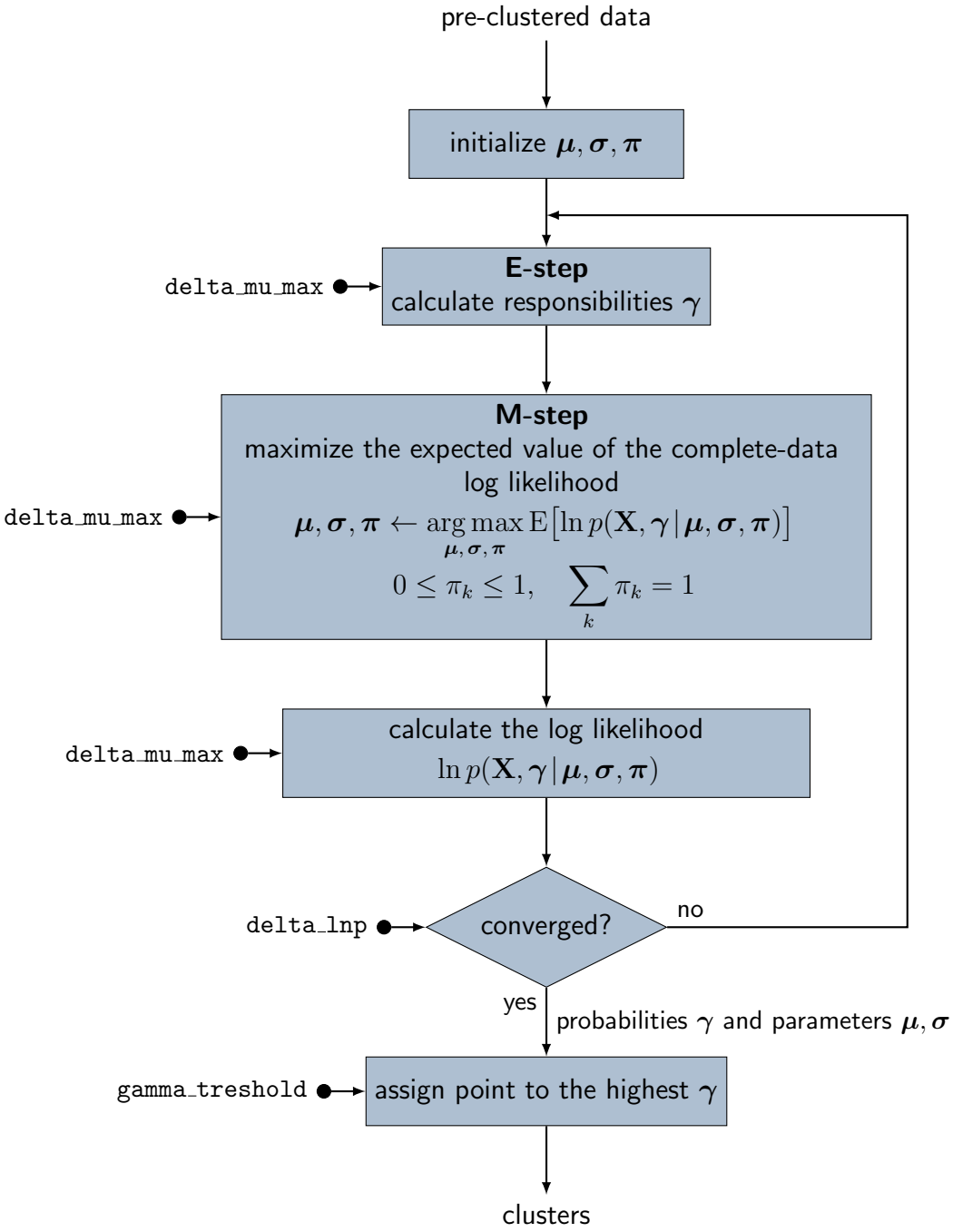


Figure 4.4: The flow chart of implementation of the EM algorithm. Configuration parameters are marked with a solid dot.

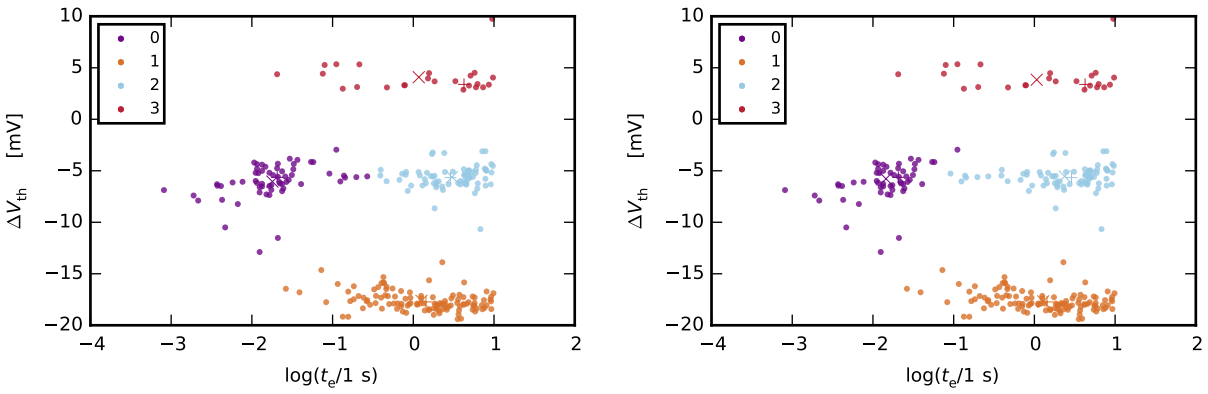


Figure 4.5: Result of EM-based clustering using assuming a bivariate Gaussian distribution **(a)** and exponential distributed over τ_e and Gaussian distributed over ΔV_{th} **(b)**

The expected value of the complete-data likelihood is

$$\mathrm{E}[p(\mathbf{X}\,|\,\boldsymbol{\mu},\boldsymbol{\sigma},\boldsymbol{\pi})] = \prod_{n=1}^{N}\prod_{k=1}^{K}\pi_k\, p(\mathbf{x}_n\,|\,\boldsymbol{\mu}_k,\sigma_k) \tag{4.5}$$

and the expected value of the complete-data log likelihood

$$\mathrm{E}[\ln p(\mathbf{X}\,|\,\boldsymbol{\mu},\boldsymbol{\sigma},\boldsymbol{\pi})] = \sum_{n=1}^{N}\sum_{k=1}^{K}\gamma_{n,k}[\ln\pi_k + p(\mathbf{x}_n\,|\,\boldsymbol{\mu}_k,\sigma_k)]. \tag{4.6}$$

The M-step consists of maximizing Eq. (4.6) with respect to $\boldsymbol{\mu}$, $\boldsymbol{\sigma}$ and $\boldsymbol{\pi}$ under the constraints

$$0 \le \pi_k \le 1 \quad \text{and} \quad \sum_{k=1}^{K}\pi_k = 1. \tag{4.7}$$

This is done numerically with the aid of the sequential least squares (SLSQP) algorithm [20]. Although distributions of the exponential family, exhibit some beneficial analytical properties relying on numerical methods for optimizing, have the advantage that the implementation gets less dependent on the underlying probabilistic model. In the next step the log likelihood is calculated with the new parameters $\boldsymbol{\mu}$, $\boldsymbol{\sigma}$ and $\boldsymbol{\pi}$. If the difference of the log likelihood between two consecutive iterations is smaller than a certain value, convergence is assumed and the iterations are stopped, otherwise the algorithm starts over from the E-step.

Finally the soft cluster assignment of each point is converted to a hard one by assigning each point n to the cluster k such that $\gamma_{n,k}$ is a maximum. At this stage a point n could be discarded from a cluster k and classified as noise according to two parameters, the minimum responsibility if $\gamma_{n,k} < \gamma_{\min}$ and the minimum probability if $p(\mathbf{x}_n\,|\,\mu_k,\boldsymbol{\sigma}_k) < p_{\min}$.

The result of the discussed initialization and subsequent clustering of real TDDS data can be seen in Fig. 4.5 in which (a) is based on a bivariate Gaussian distribution and (b) is based on the distribution Eq. (4.2). Apart from the clearly higher quality of the result when considering the temporal exponential distribution, faster converge on the TDDS data can be observed as well, which is plausible, as the likelihood function is better adapted to the underlying probabilistic model.

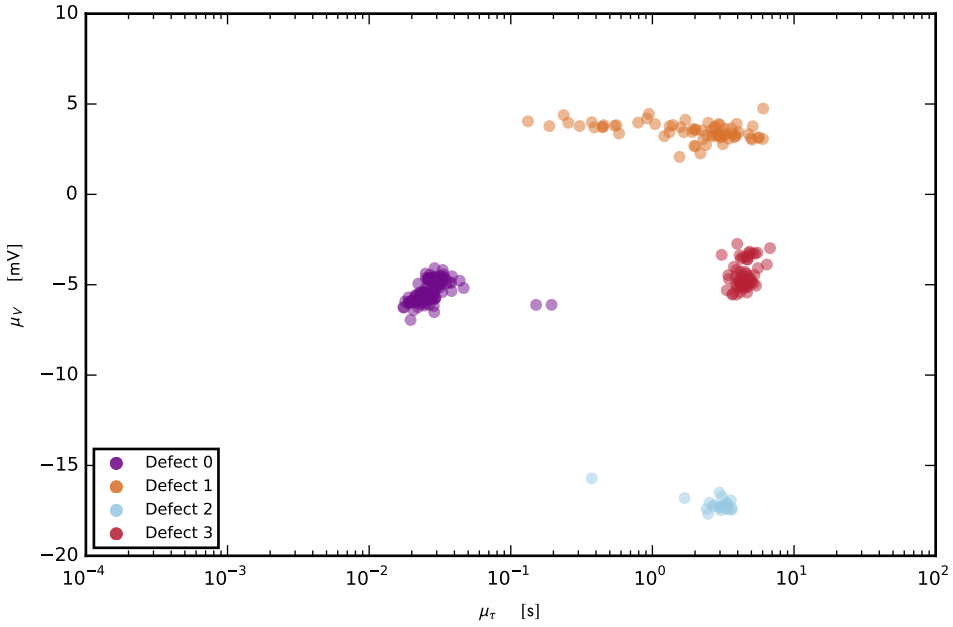


Figure 4.6: The mean values $\boldsymbol{\mu} = (\mu_\tau, \mu_V)^\top$ of the clusters found by the EM-algorithm. The color gives the result of the defect prediction. It can be seen that the mean values of the clusters again form clusters, a fact which is exploited during the defect prediction.

4.3 Defect Prediction

When reaching the cluster prediction step, all measurements of one physical device have been clustered, i.e. the emission events in one block that are similar according to the clustering algorithm are grouped together. Across different blocks some of these, at this point still independent clusters, belong together as well, because the emission events are due to the same physical defect. The aim of this step is to predict the defect each cluster is belonging to.

Rationale The naïve idea is that the mean values of the clusters across different blocks form clusters again and to use this as a means of defect prediction, what can be seen in Fig. 4.6. With a perfect clustering there should be a one-to-one mapping between clusters in each block and the defects. In reality, however, this is not always the case as emission events of one defect could be split into multiple clusters or the emission events of different defects are merged into one common cluster. Most of the time this is the result of a suboptimal initialization. Further, more sever complexities arise from the fact that the defect parameters change considerably in dependence of the stress and recover conditions.

One of the advantages of this defect prediction technique is that a the emission event of one defect that erroneously got split up into multiple cluster in a previous step are still assigned to the same defect as long as the cluster mean lies close enough to the defect centroid. Additionally all bad defect assignments could be easily corrected manually by the user on a per-cluster basis.

The major weakness of this approach is its incapability of handling clusters that move significantly in response to changes of the stress or recovery conditions. As a consequence the mean values of the clusters do not longer form clusters themselves but can be thought as stochastic deviations from trajectories that are defined by the stress and recover conditions. Making again use of some

kind of maximum likelihood algorithm is conceivable, but this case is much more complex, because there are more parameters and latent variables. In concluding, it may be said that the current implementation works fine in cases where the mean values of the clusters only move a moderate distance across different blocks, but a much more sophisticated algorithm is necessary to handle other cases.

Description of the Implementation The process of defect prediction is quite similar to the cluster initialization and is sketched in Fig. 4.7. Even if the temperature dependence is not the same for all defects, in order to at least partially compensate the temperature dependence, a compensated emission time τ_e' is calculated with

$$\tau_e' = \tau_e 10^{\Delta T k_T}, \tag{4.8}$$

where τ_e is the original emission time, $\Delta T = T - 100°$ is the deviation from the default measurement temperature and k_T is the compensation constant. But instead of operating on single points of one block as in the initialization, the mean values of the clusters of all blocks received from the cluster detection are used.

Based on the compensated clusters the KDE of the mean values of the clusters of all blocks is calculated. The bandwidth is again selected according to Eq. (4.1), where N is the number of clusters found in all blocks and k is chosen smaller than during the initialization. The maxima of the KDE serve as the centroids of the defects and the Mahalanobis distance with respect to a constant diagonal covariance matrix is calculated. Every cluster is assigned to the defect with the smallest Mahalanobis between the cluster mean and the defect centroid and clusters whose Mahalanobis distance surpasses a certain limit are considered as outliers and simply discarded.

4.4 Parameter Extraction

The emission time τ_e can, in a good approximation, be calculated as the mean value of each defect cluster with respect to the time. The capture time τ_c and occupancy B is extracted by doing a least square fit on the capture probability Eq. (2.2). The occupancy B is thus defined as the capture probability for long stress times $B = P_c(t_s \to \infty)$.

For a good quality of fit of B there need to be at least a few blocks where $t_s \gg \tau_c$. For a good quality of fit of τ_c there need to be at least a few block in the intermediate regime, where $t_s \approx \tau_c$. This situation is illustrated in Fig. 4.8.

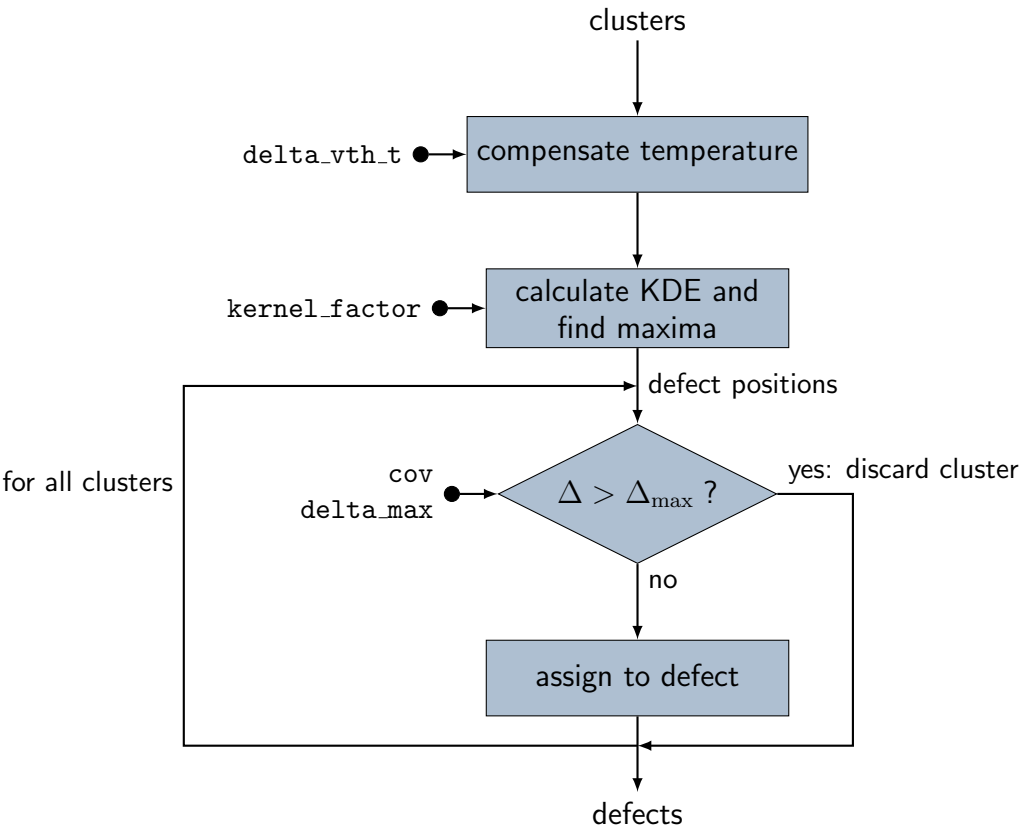


Figure 4.7: The flow chart of the defect prediction. Configuration parameters are marked with a solid dot.

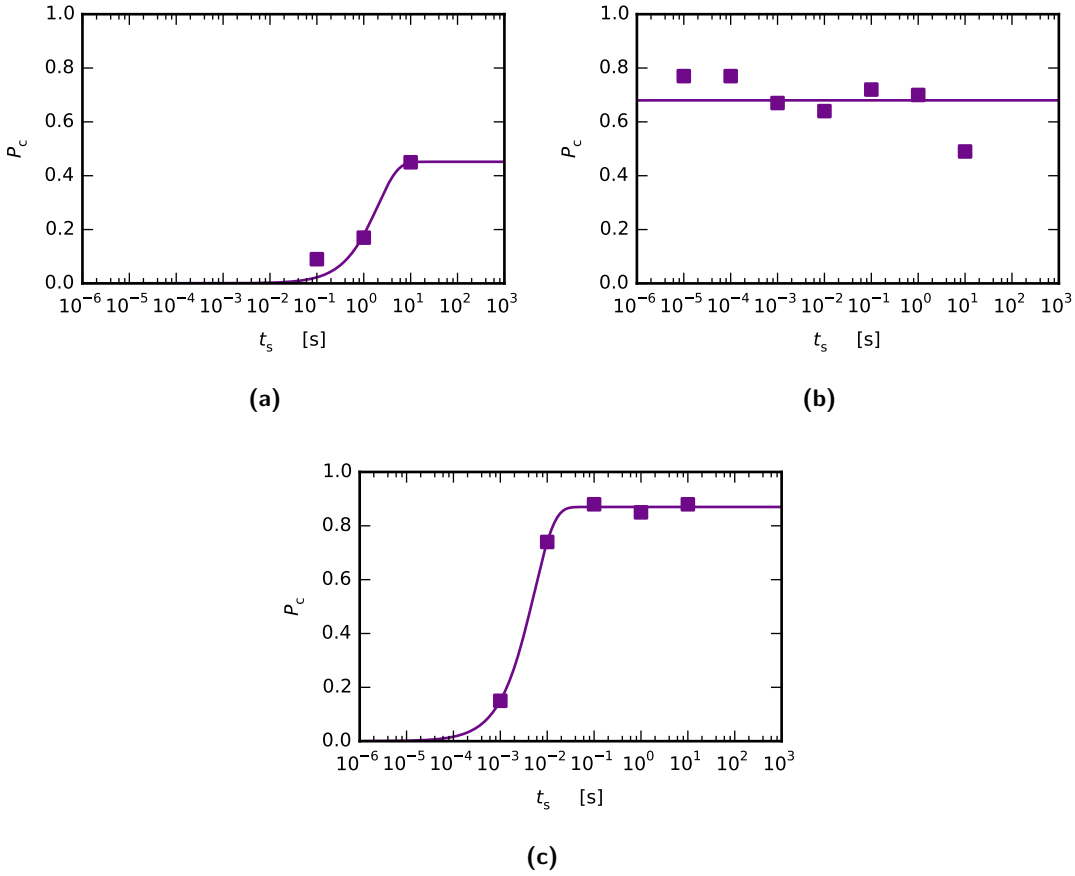


Figure 4.8: Extraction of the capture time τ_c and the occupancy B of three different defects. **(a)** There is not a sufficient number of measurements with a stress time t_s much larger than the expected capture time τ_c of the defect, so no reasonable value for the occupancy B could be extracted. **(b)** The stress time t_s of all measurements is much larger than the capture time τ_c of the defect and therefore no conclusion about the quantitative value of τ_c can be drawn. **(c)** There is a sufficient number of measurements in the intermediate regime where $t_s \approx \tau_c$ and some blocks with $t_s \gg \tau_c$, so the extracted value of both τ_c and B could therefore expected to be of feasible quality.

5 Results

Results of the clustering algorithm and other low-level components have already been discussed in previous chapters. The following chapter gives some results of the cluster detection algorithm applied on real TDDS data.

5.1 Granularity of the Clustering Result

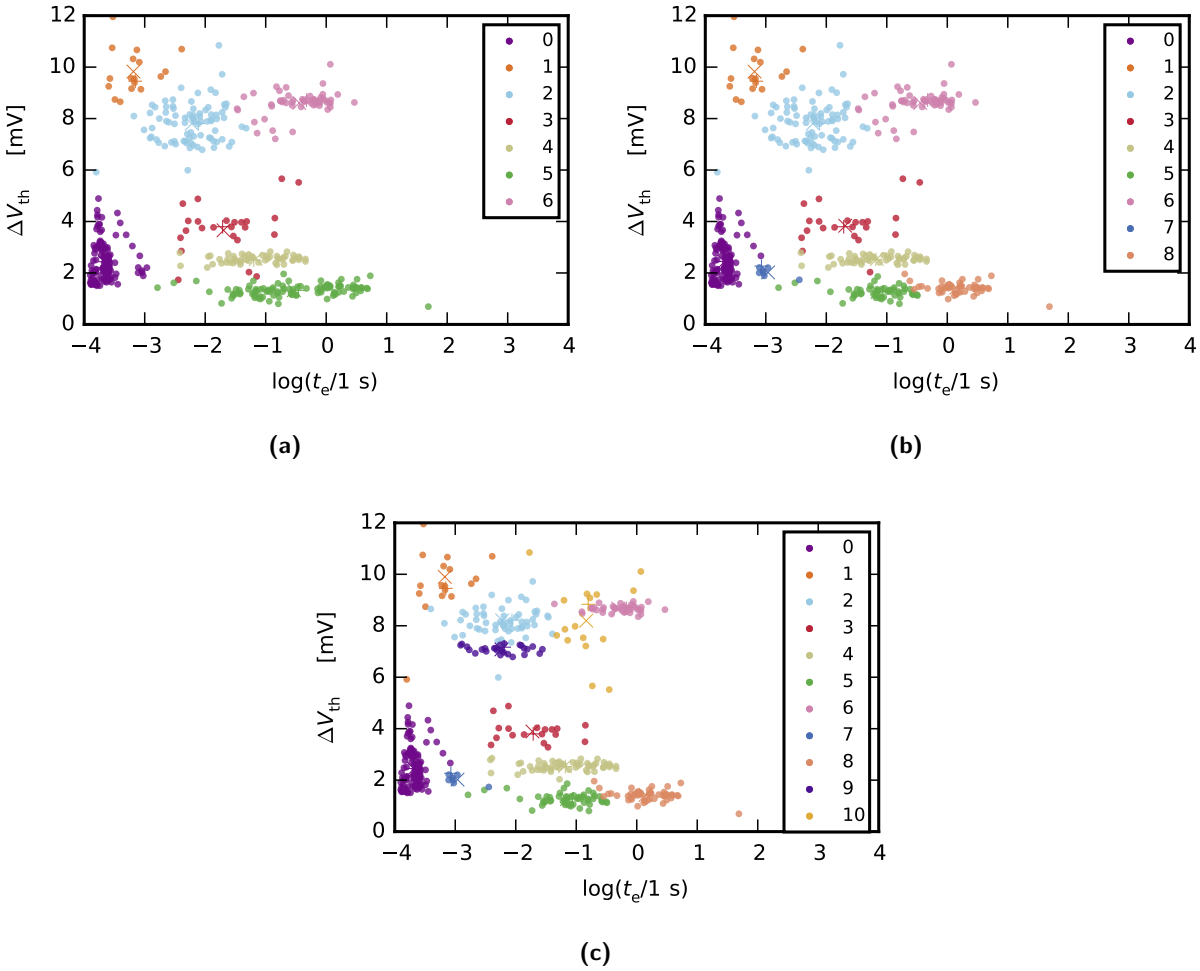
Fig. 5.1 shows how the number of clusters can by determined by the kernel factor κ described in Section 4.1. A smaller kernel factor results in more peaks in the kernel density estimation and therefore in a larger number of initialized clusters. This example clearly points out that there is no unique solution to a clustering problem. In the context of TDDS additional insight can be gained by varying the stress and recovery times, the stress and recovery bias and the device temperature.

5.2 Dependence of the Emission Time on the Temperature

As aforementioned, the temperature plays an important role in the defect characterization. The capture and emission times are very sensitive to changes of the device temperature. This behavior is formally described by the Arrhenius' law Eq. (2.3). Fig. 5.2 shows the mean values of five detected clusters analyzed from recorded TDDS data of a single nanoscale device. As expected similar emission times are observed when the stress time and stress bias is changed. This fact can be further used to check if the detected clusters from each stress/measurement sequence are correctly assigned to the corresponding defect. The outliers are a consequence of the recovery bias dependence of the switching traps. The temperature dependence of the emission time τ_e becomes visible through a shift of the mean values of the detected clusters at 125 °C (red) compared to the clusters at 100 °C (blue), see Fig. 5.3. Taking defect #4 as an example, it can be seen that the implemented defect prediction is able to assign the cluster to the correct defect even with a considerable temperature induced change of the defect parameters.

5.3 Trapping Kinetics

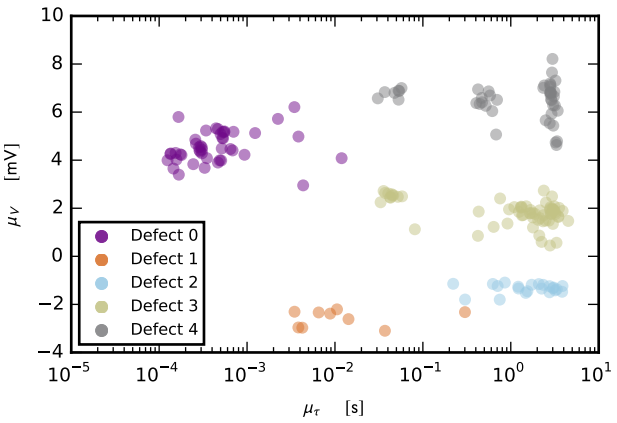
The gate voltage dependence of the time constants of one particular defect is shown in Fig. 5.4. The bias independent emission time τ_e is typical for fixed oxide traps. Quite contrary, the capture time varies over multiple decades in response to a change of the stress gate bias of less than one volt, specifically the capture time decreases with an increase of the absolute value of the gate voltage V_G. In addition to the bias dependence, the emission time τ_e and capture time τ_c change with different device temperatures.



(a) **(b)** **(c)**

Figure 5.1: The result of clustering with different values of the kernel factor of the initialization **(a)** $\kappa = 1.7$, **(b)** $\kappa = 1.3$ and **(c)** $\kappa = 1.0$.

Figure 5.2: The mean values of the predicted clusters resulting from 26 measurement sequences of TDDS data of a PMOS device.



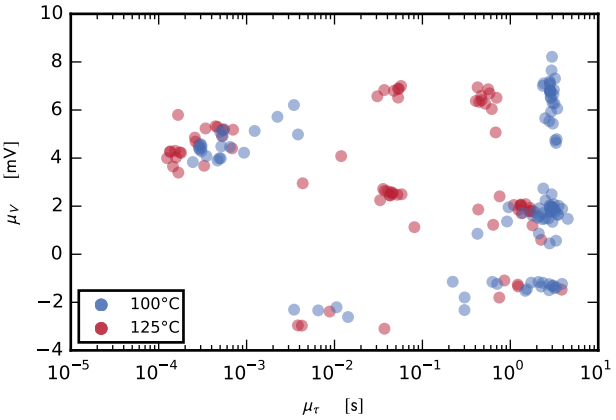


Figure 5.3: The temperature dependence of the mean values of the clusters from Fig. 5.2.

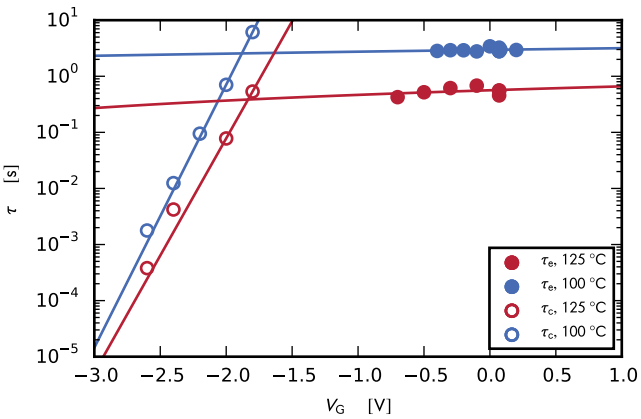


Figure 5.4: Gate voltage dependence of the capture and emission times of one defect studied in a nanoscale p-channel MOSFET ($W \times L = 160\,\mathrm{nm} \times 120\,\mathrm{nm}$) for two different temperatures.

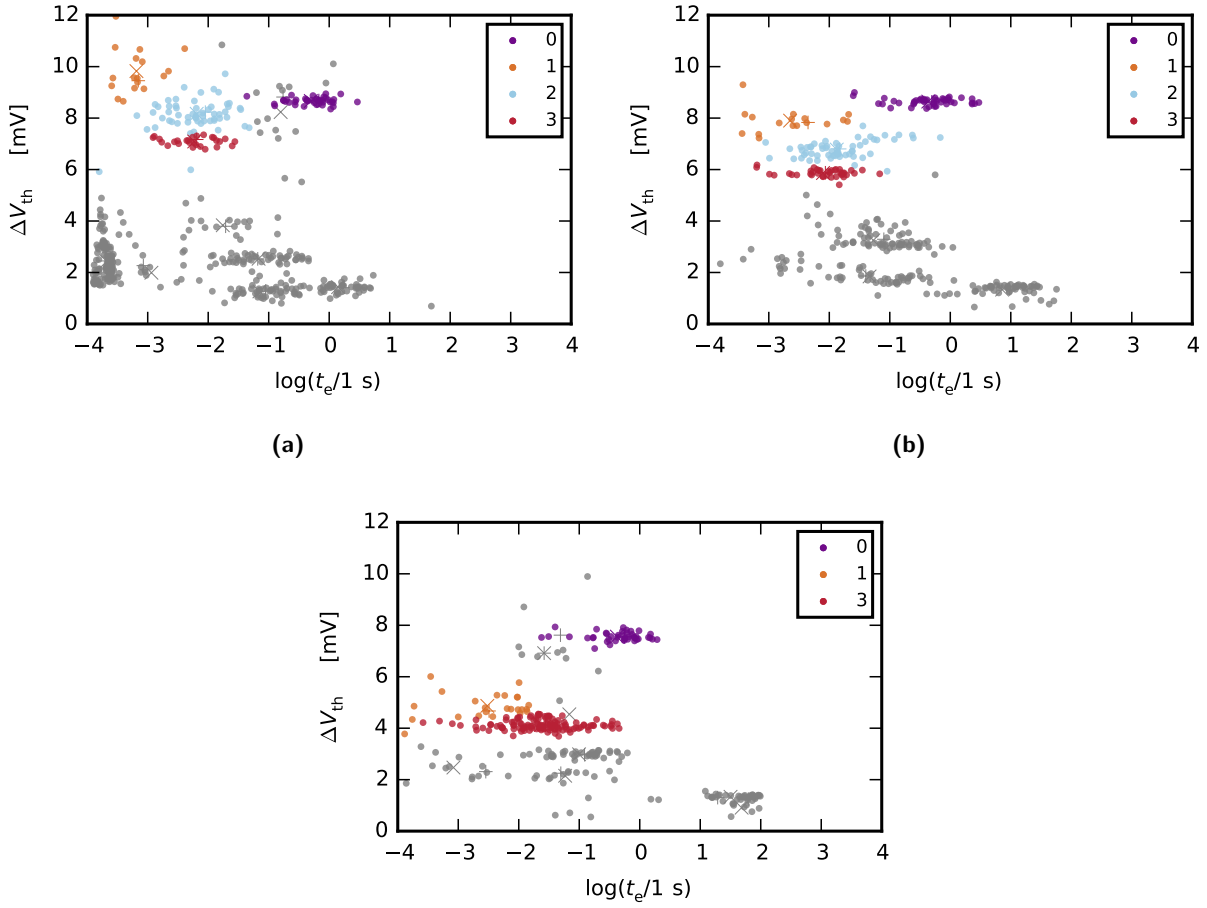


Figure 5.5: A selection of cluster obtained from TDDS data for different readout voltages **(a)** $V_G = -400\,\mathrm{mV}$, **(b)** $V_G = -500\,\mathrm{mV}$ and **(c)** $V_G = -600\,\mathrm{mV}$.

5.4 Limit for an Automated Analysis

Fig. 5.5 illustrates the limit of an automated analysis of TDDS data in case of coalescing clusters as well as a pronounced change of the emission parameters as a result of a change of the recovery condition. In both Fig. 5.5a and Fig. 5.5b the selected clusters are clearly separated and easily identifiable. However, in Fig. 5.5c some of the clusters have coalesced into a single bigger one. It's impossible for both a human and a machine to make an unambiguous assignment of the emission events to a particular defect. Because of the pronounced change of the cluster means in Fig. 5.5 this is also an example of a case where the current defect prediction algorithm fails to determine to which defect the cluster belongs to.

5.5 Recovery Bias Dependence of Single Traps

To study the dependence of the step height of single traps on different drain-biases during recovery, a pMOSFET ($W \times L = 160\,\mathrm{nm} \times 120\,\mathrm{nm}$) is investigated. By using the TDDS and subsequently

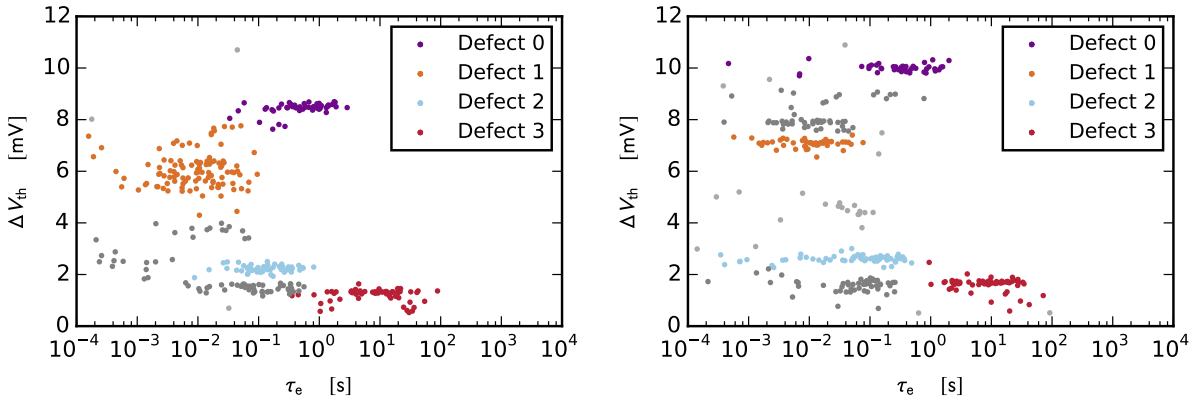


Figure 5.6: Using the proposed algorithm four defects can be identified in the spectral maps recorded using **(a)** $V_{\mathrm{DS}} = -100\,\mathrm{mV}$, **(b)** $V_{\mathrm{DS}} = -900\,\mathrm{mV}$ and $V_{\mathrm{G}} = -500\,\mathrm{mV}$ during recovery. The device has been stressed using $V_{\mathrm{DS}} = 0\,\mathrm{V}$ and $V_{\mathrm{G}} = -1.8\,\mathrm{V}$ and the measurements are performed at $T = 60\,^{\circ}\mathrm{C}$.

applying the EM based cluster detection algorithm to the spectral maps extracted from the measurements, four traps could be identified, see Figs. 5.6 and 5.7. Noteworthy, even with a high amount of intermediate noise and very close clusters, the presented algorithm leads to satisfying results.

Finally, the recovery drain bias V_{DS} dependence of the mean value of the step height of these four defects is shown in Fig. 5.8. A remarkable dependency of the step heights on the recovery drain-source bias can been, as the average step heights decrease with a smaller absolute value of V_{DS}.

5.6 Consideration of Runtime and Number of Iterations

Fig. 5.9 shows a scatter matrix of points in a block, the number of initialized clusters, the number of iterations and the time of the EM-algorithm until convergence. As only a very small sample of 108 blocks in total are considered for this analysis, conclusions have to be taken with caution. Only in rare cases more than four iterations are necessary to reach convergence. Rather intuitively the number of iterations increases with the number of clusters, but are nearly independent of the number of points in a block. The situation is quite different for the time to complete the EM-algorithm, especially in dependence of the number of cluster. This can be reasoned by the fact that every cluster adds three additional parameters to the likelihood function that has to be optimized. Indeed, very poor convergence during the optimization step is observed with test data with more than 10 clusters.

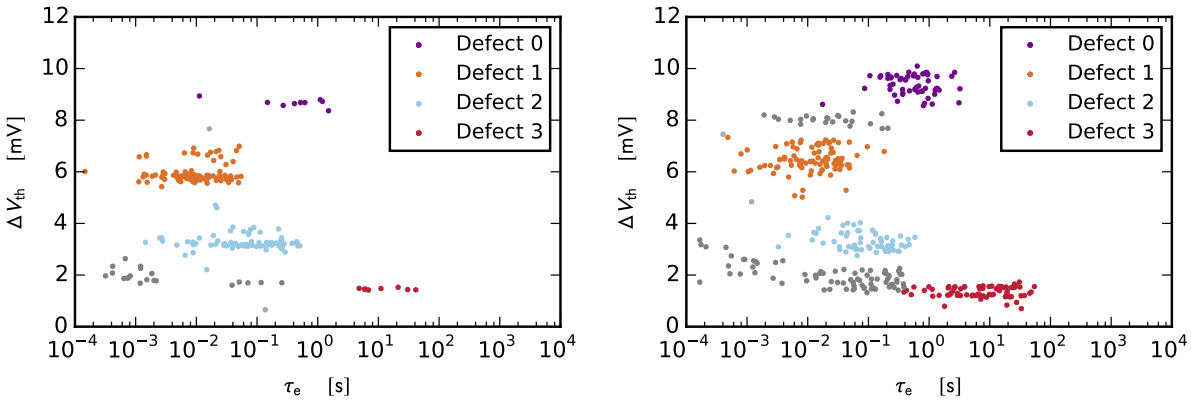
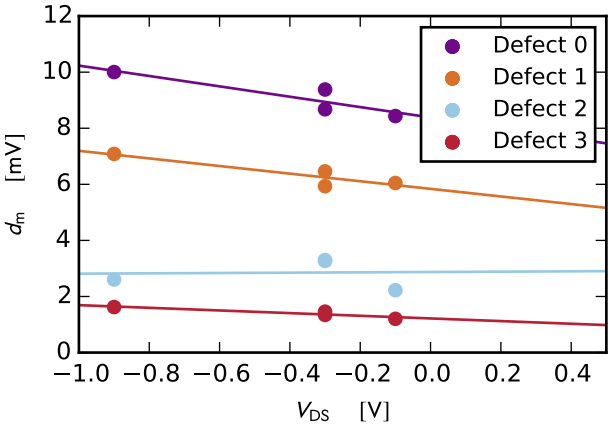


Figure 5.7: With an increasing stress time from **(a)** $t_\mathrm{s} = 100\,\mathrm{ms}$ to **(b)** $t_\mathrm{s} = 2\,\mathrm{s}$ a new defect close to defect $\#2$ appears. Nevertheless, the cluster detection algorithm is able to detect the defect correctly. Both measurements are recorded applying $V_\mathrm{DS} = -300\,\mathrm{mV}$ at $T = 60\,^\circ\mathrm{C}$. The stress biases are the same as in Fig. 5.6.

Figure 5.8: The step heights, i.e. mean value of ΔV_th shifts produced by the single defects in Figs. 5.6 and 5.7, is plotted against V_DS during recovery. The average step heights of the emissions become smaller for a smaller absolute value of V_DS. All experiments are performed on the same p-channel MOSFET with a geometry of $W \times L = 160\,\mathrm{nm} \times 120\,\mathrm{nm}$.



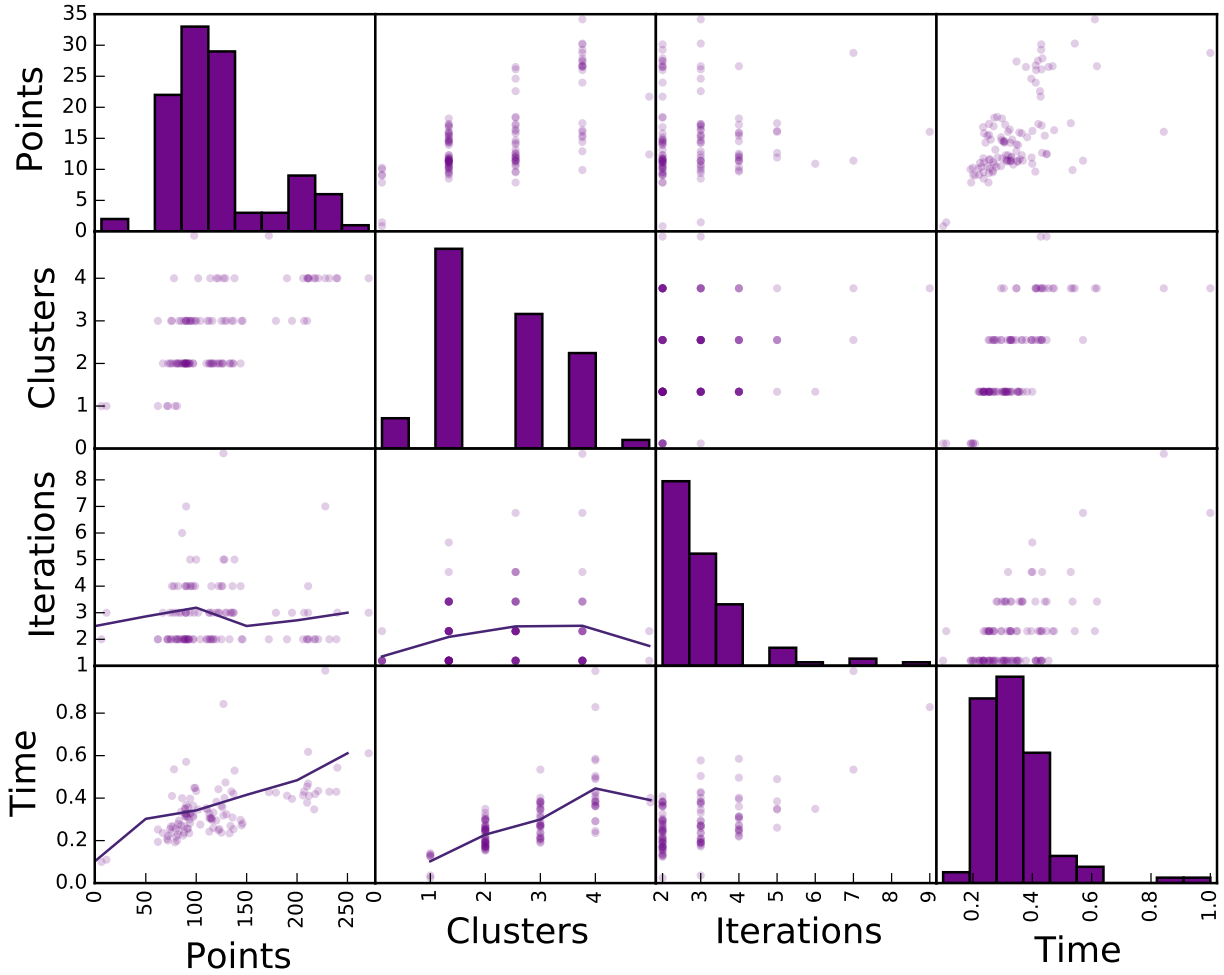


Figure 5.9: Scatter matrix plot of the number of points in the switch file, the number of clusters found during initialization, the number of iterations during the EM-algorithm, and the normalized computation time until convergence of the EM-algorithm. The data is based on 108 blocks of TDDS data and the values are considered on a per-block basis. The diagonal elements show a histogram of the corresponding value. The lines represent the mean values. For the number of points the value is binned into bins of 50 points.

6 Conclusion

A clustering algorithm based on expectation maximization (EM) as well as an implementation for its initialization and a subsequent defect prediction has been demonstrated in this work. The good results of the clustering algorithm can be attributed to the probabilistic nature of the EM algorithm that accounts for the statistical behavior of the underlying physical process. Even with seemingly nice-looking clusters the result has to be used with caution, though. By converting the fuzzy cluster assignment to a hard one by assigning every point to the most probable cluster, will presumably lead to clusters with a systematic bias toward an underestimated variance. Roughly speaking, there is the tendency to smaller, denser and strictly separated clusters over overlapping sparse ones. This however does not seem to cause big problems when it comes to the subsequent TDDS analysis. A full probabilistic treatment of the subsequent components is conceivable, but the considerable implementation effort is certainly the biggest counterargument. It has been observed that the result of the clustering algorithm strongly depends on the initial values that can either be supplied manually or obtained automatically from the presented algorithm. The former manner of supplying initial values is beneficial in cases where the user wants to designate the approximate cluster positions while still taking advantage of the subsequent EM-based clustering. With basically two parameters of the automatic initialization algorithm the clustering result can be gradually tuned to yield a larger number of smaller clusters or a smaller number of larger clusters. Furthermore, reasonably fast convergence in most cases is achieved by the presented initialization. The computational demand of the implementation calls for parallelization which could be achieved easily because of the missing interdependency between different TDDS measurements during the cluster detection. Such an interdependency comes into play in a defect prediction step subsequent to the EM. Perfect results are achieved with the proposed straightforward implementation of the defect prediction approach in cases where the mean values of the emission events of the defects only move moderately as a result of a change of the stress and recovery bias conditions and temperature. A much more sophisticated implementation is necessary to handle other cases. The compelling reason of a separate cluster prediction step is that the defect assignment of a cluster can be easily done manually for cases in which the automatic prediction fails. The output of the clustering algorithm is still valuable in such cases due to the time-saving compared to manually marking the clusters. In concluding, it may be said that the implemented clustering algorithm proved capable of performing an unsupervised analysis of TDDS data in a lot of cases, which opens up the opportunity for an automated study of both single devices and whole transistor arrays.

Bibliography

[1] Charu C. Aggarwal and Chandan K. Reddy, editors. *Data clustering; algorithms and applications*. Chapman & Hall/CRC data mining and knowledge discovery series. CRC Press, 2014.

[2] Mihael Ankerst, Markus M. Breunig, Hans-Peter Kriegel, and Jörg Sander. Optics: Ordering points to identify the clustering structure. *SIGMOD Rec.*, 28(2):49–60, June 1999.

[3] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[4] J. P. Campbell, P. M. Lenahan, C. J. Cochrane, A. T. Krishnan, and S. Krishnan. Atomic-scale defects involved in the negative-bias temperature instability. *IEEE Transactions on Device and Materials Reliability*, 7(4):540–557, Dec 2007.

[5] B. E. Deal. Standardized Terminology for Oxide Charges Associated with Thermally Oxidized Silicon. *J.Electrochem.Soc.*, 127(4):979–981, 1980.

[6] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. pages 226–231. AAAI Press, 1996.

[7] Brian S Everitt and Brian Everitt; Sabine Landau. *Cluster analysis : Wiley series in probability and statistics*. Wiley, 5 edition, 2010.

[8] D. M. Fleetwood. 'Border traps' in MOS devices. *IEEE Transactions on Nuclear Science*, 39(2):269–271, Apr 1992.

[9] P.L. Gatti. *Probability Theory and Mathematical Statistics for Engineers*. Structural Engineering: Mechanics and Design. Taylor & Francis, 2004.

[10] T. Grasser. Stochastic charge trapping in oxides: From random telegraph noise to bias temperature instabilities. *Microelectronics Reliability*, 52:39–70, October 2012.

[11] T. Grasser, H. Reisinger, P.-J. Wagner, W. Goes, F. Schanovsky, and B. Kaczer. The Time Dependent Defect Spectroscopy (TDDS) for the Characterization of the Bias Temperature Instability. In *Proceedings of the 2010 International Reliability Physics Symposium*, pages 16–25, May 2010.

[12] T. Grasser, H. Reisinger, P.-J. Wagner, F. Schanovsky, W. Goes, and B. Kaczer. The Time Dependent Defect Spectroscopy (TDDS) for the Characterization of the Bias Temperature Instability. In *Proceedings of the 2010 International Reliability Physics Symposium*, pages 16–25, May 2010.

[13] T. Grasser, K. Rott, H. Reisinger, M. Waltl, J. Franco, and B.Kaczer. A Unified Perspective of RTN and BTI. In *Proceedings of the International Reliability Physics Symposium*, 2014.

[14] Tibor Grasser, editor. *Bias Temperature Instability for Devices and Circuits*. Springer-Verlag New York, 2014.

[15] A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: A review. *ACM Comput. Surv.*, 31(3):264–323, September 1999.

[16] Leonard Kaufman and Peter J. Rousseeuw. *Finding groups in data; an introduction to cluster analysis*. Wiley series in probability and mathematical statistics : Applied probability and statistics. Wiley, New York, NY [u.a.], 1990.

[17] Kevin P. Murphy, editor. *Machine Learning – A Probabilistic Perspective*. Adaptive computation and machine learning. Massachusetts Institute of Technology, 2012.

[18] H. Reisinger, O. Blank, W. Heinrigs, W. Gustin, and C. Schlunder. A comparison of very fast to very slow components in degradation and recovery due to nbti and bulk hole trapping to existing physical models. *IEEE Transactions on Device and Materials Reliability*, 7(1):119–129, March 2007.

[19] scikit learn. Documentation. `http://scikit-learn.org/stable/documentation.html`, June 2016.

[20] Scipy. Documentation. `http://docs.scipy.org/doc/`, June 2016.

[21] T.T. Soong. *Fundamentals of Probability and Statistics for Engineers*. John Wiley & Sons, 2004.

[22] Kwong-Tin Tang. *Mathematical Methods for Engineers and Scientists*. Springer, 2007.

[23] Ronald E. Walpole and Raymond H. Myers. *Probability and statistics for engineers and scientists*. Macmillan, New York, NY, 2 edition, 1978.

[24] Ronald E. Walpole, Raymond H. Myers, Sharon L. Myers, and Keying Ye. *Essentials of Probability & Statistics for Engineers & Scientists*. Pearson, 2013.

[25] M. Waltl, P.-J. Wagner, H. Reisinger, K. Rott, and T. Grasser. Advanced Data Analysis Algorithms for the Time–Dependent Defect Spectroscopy of NBTI. In *Proc. Intl.Integrated Reliability Workshop*, pages 74–79, 2012.

[26] M. Waltl, P.-J. Wagner, H. Reisinger, K. Rott, and T. Grasser. A Single-Trap Study of PBTI in SiON nMOS Transistors: Similarities and Differences to the NBTI/pMOS Case. In *Proceedings of the 2010 International Reliability Physics Symposium*, pages XT18.1–XT18.5, 2014.

[27] Michael Waltl. Change Point Detection in Time Dependent Defect Spectroscopy Data. Master's thesis, Vienna University of Technology, 2011.

[28] Wikipedia. DBSCAN. `https://de.wikipedia.org/wiki/DBSCAN`, June 2016.

List of Figures