Bernhard Krenmayr, BSc

Experimental parameter finding to model strength and toughness

MASTER THESIS

For obtaining the academic degree

Diplom-Ingenieur

Master programme of Advanced Materials Science

Graz University of Technology

Supervisor:

Assoc.Prof. Dipl.-Ing Dr.techn. Norbert Enzinger

Institute of Materials Science and Welding

Graz, April 2015

Danksagung

Als erstes möchte ich mich bei meinen Eltern bedanken. Dafür dass sie mir ermöglicht haben, mein Studium zu absolvieren und mich dabei immer unterstützt haben.

Weiters möchte ich mich beim Institut für Werkstoffkunde und Schweißtechnik bedanken, dass mir ermöglicht wurde, meine Ideen und Interessen in einer Diplomarbeit zu verfassen. Im Besonderen möchte ich mich bei meiner Betreuerin Monika Schuler und meinem Betreuer Norbert Enzinger bedanken, die mich bei meiner Verwirklichung der Arbeit unterstützt haben.

Da meine Ausbildung nun zu Ende ist, ist es auch an der Zeit meinen Freunden und Begleitern während des Studiums zu danken, die mir immer zur Seite gestanden sind, mich immer unterhalten haben und für Abwechslung während des harten Unialltags gesorgt haben. Besonders danken möchte ich meinen Kollegen der letzten Zeit: Christian Gösweiner, Florian Apolloner, Michael Hollerer, Johannes Steurer und Patrick Kraus.

Zu guter Letzt möchte ich mich noch bei Carolyn Aichhorn bedanken, die mir in der harten finalen Phase immer zur Seite gestanden ist und mich immer wieder motiviert hat.

Danke.

Abstract

At the Institute of Materials Science and Welding, Graz University of Technology, highly heat resisting and creep resistant steels are subject of research.
For modelling the mechanical properties of these steels, experimental data of determinable parameters are required. This thesis concerns the determination of these characteristic parameters and their interpretation.

The first parameter to determine is the dislocation density. It represents the number of dislocation defects in the crystal lattice of the microstructure. That defect density has a big influence on the plastic deformation behaviour of the material. The dislocation density can be evaluated by characteristic diffraction effects of high energetic X-ray radiation. Together with the Faculty of Physics at the University of Vienna, measurements on the synchrotron in Grenoble have been conducted. Furthermore, evaluations of the collected data were done.

The second parameter, which has to be determined, is the impact energy. It is a value for the absorbed energy during fracture of the material and it enables a classification into ductile or brittle failure. This value can be determined with the so called Charpy V-notch impact test, but the measuring setup has in its simplicity several weaknesses. Hence, this thesis is about the evaluation of the instrumented setup, which delivers many additional informations. Aided by the digital measuring signals and a computer programme for the evaluation, the dynamic fracture behaviour should be better understood. The evaluation summarizes the big amount of data in few characteristic values, which describe the analogue value in more detail.

Kurzfassung

Am Institut für Werkstoffkunde und Schweißtechnik der TU Graz wird an hochwarmfesten und kriechbeständigen Stählen für den Kraftwerksbau geforscht.
Zur Modellierung der mechanischen Eigenschaften dieser Stähle werden experimentelle Daten von bestimmten Parametern benötigt. Die Arbeit beschäftigt sich mit zwei von diesen charakteristischen Kennwerten und deren Interpretation.

Der erste zu bestimmende Parameter, die Versetzungsdichte, ist ein Maß für die Anzahl von bestimmten Defekten im Kristallgitter des Gefüges. Diese hat im Werkstoff einen sehr hohen Einfluss auf die Festigkeit und Verformbarkeit. Die Versetzungsdichte kann mit Hilfe von Beugungseffekten aus hochenergetischer Röntgenstrahlung (Synchrotron) ermittelt werden. Gemeinsam mit der Fakultät für Physik der Universität Wien wurden Messungen durchgeführt und ausgewertet.

Beim zweiten zu bestimmenden Parameter handelt es sich um die Kerbschlagarbeit. Sie ist ein Maß dafür, ob ein Werkstoff duktil oder spröde bricht bzw. wie viel Energie bis zum Versagen aufgenommen werden kann. Dieser Wert kann mit Hilfe des Kerbschlagbiegeversuchs bestimmt werden. Der Messaufbau hat in seiner Einfachheit große Schwächen. Deshalb beschäftigt sich diese Arbeit mit der Auswertung des instrumentierten Aufbaus, der viele zusätzliche Informationen liefert. Mit Hilfe der digitalen Messsignale und eines dafür angefertigten Auswerteprogramms, soll das dynamische Bruchverhalten besser erfasst und verstanden werden. Die Auswertung fasst die große Datenmenge in wenige charakteristische Größen zusammen, die den analog ermittelten Wert detaillierter beschreiben.

EIDESSTATTLICHE ERKLÄRUNG

AFFIDAVIT

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche kenntlich gemacht habe. Das in TUGRAZonline hochgeladene Textdokument ist mit der vorliegenden Masterarbeit/Diplomarbeit/Dissertation identisch.

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly indicated all material which has been quoted either literally or by content from the sources used. The text document uploaded to TUGRAZonline is identical to the present master's thesis/diploma thesis/doctoral dissertation.

_____________________________ _____________________________

Datum / Date Unterschrift / Signature

Contents

1. Introduction

The target in engineering modern power plants is to improve the efficiency of the energy generation. To realize that, high performance materials are necessary, that meet the demanding requirements.

At the Institute of Materials Science and Welding, Graz University of Technology, highly heat resistant and creep resistant steels are subject of research, which may be used in such power plants in the future.

Specifically, a currently developed and investigated material is CB2 steel, a highly alloyed chromium steel [1]. For modelling strength and toughness of this material, several researches and simulations of the microstructure with its different phases and precipitations are carried out [2].

As a part of a current PhD project, a more detailed look at two different material characteristics is required.
This consists of two main topics:

- The determination of the dislocation density in the microstructure. This parameter will be required in further phase and precipitation simulations in MatCalc.
- A close investigation of the occurring fracture mechanics to describe the toughness behaviour.

Although these characteristics depend on each other, they are considered as two separate topics in the present thesis.

2. Dislocation density

In this chapter, a method to determine the dislocation density is introduced. With the principle of X-Ray diffraction the microstructures of metals are investigated. In cooperation with the Faculty of Physics at the University of Vienna, the limitations and possibilities of this method have been investigated. In the following sections the basics of the microstructure of metals and the basics of X-Ray diffraction are explained. Consecutive, the evaluation of the measurements with the essential mathematics behind, is illustrated.

2.1. Literature

2.1.1. Microstructure

A crystal is an arranged system of atoms with a highly ordered structure. The crystal lattice consists of unit cells which can be infinitely repeated in all spatial directions. With this information of structure and symmetry of the unit cell (figure 2.1) and translation rules for repeating in space, a macroscopic structure can be built up. [3].

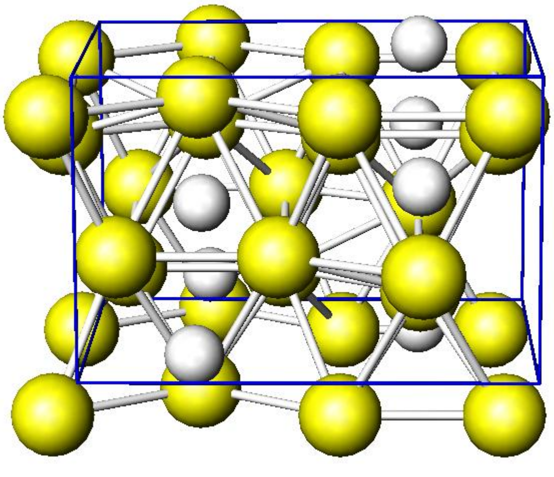


Figure 2.1.: Unit cell of a Fe-C system: Carbon (white) is placed at interstitial positions in the triangular iron (yellow) grid [4]

Such theoretical infinitely structures will not be formed in reality, because the equilibrium conditions which are necessary for a perfect crystal growth are not given and hence real crystals include defects or irregularities.

2.1.1.1. Structural defects

Real crystals are never faultless and they deviate in their disorder far from the thermodynamic equilibrium. For example in reality, it is not possible to cool down a material, from the melt to room temperature infinitely slow to get the thermodynamic equilibrium. There are always limiting conditions, so there will always be defects in the microstructure.

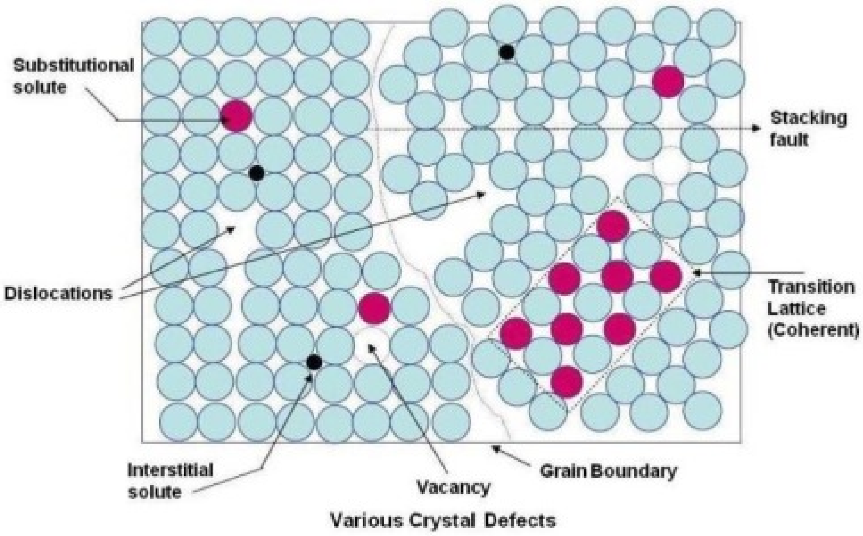


Figure 2.2.: Schematic illustration of different crystal defects [5]

There are four different sorts of crystallographic defects, which are classified in their spatial expansion [6]:

1. Point defects (zero dimensional)
   - vacancies
   - interstitial atoms
   - substitutional atoms
2. Line defects (one dimensional)
   - dislocations

3. Planar defects (two dimensional)
   - grain boundaries
   - stacking faults
4. 3D-defects (three dimensional)
   - phase boundaries
   - precipitations
   - pores

These defects are the main influencing factors of the mechanical properties of a material. With controlled integration of defects in a material by a defined heat treatment, the microstructure can be varied in a broad range. So materials with different mechanical properties can be created.

2.1.1.2. Dislocations

One of the most important mechanisms in a material is the plastic deformation behaviour. Primarily it is the creation and movement of dislocations (figure 2.3).

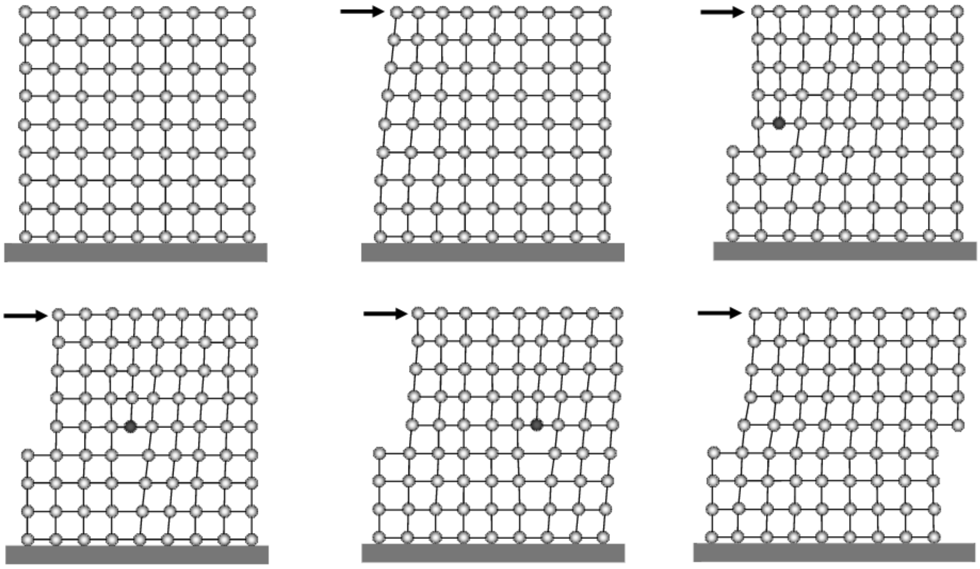


Figure 2.3.: Schematic illustration of creation and movement of an edge dislocation during plastic deformation [3]

A dislocation is a one-dimensional structural defect. It can be illustrated as a disordering along a line through the crystal structure [7].

Two different types of dislocations exist, the edge and the screw dislocations.
In an edge dislocation a half plane of atoms is missing in the structure. Respectively an extra half plane is introduced mid way through the crystal (figure 2.4 left).
The other type is the screw dislocation, where the crystal planes are slipped against each other parallel to the cutting surface (figure 2.4 right).

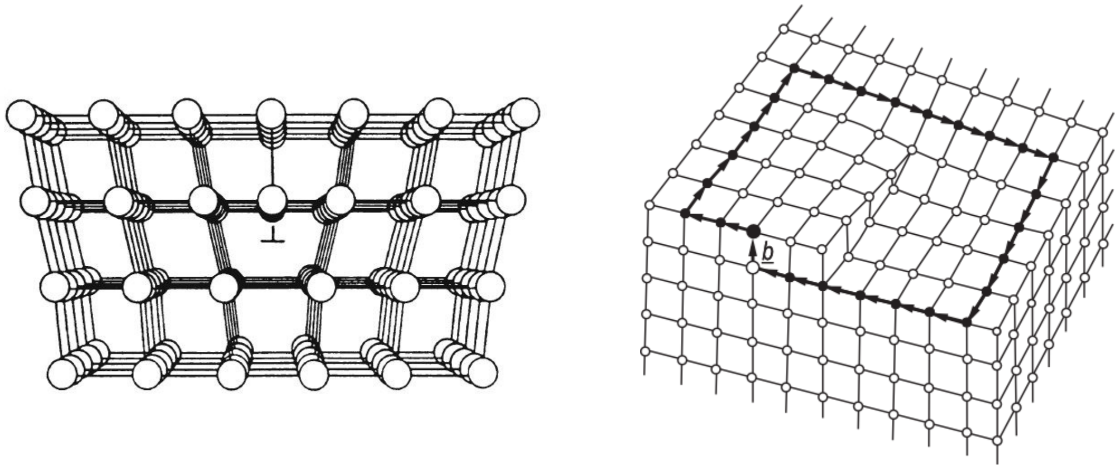


Figure 2.4.: Two different types of dislocations [7]; left) edge dislocation; right) screw dislocation

The edges of such dislocations form the characteristic dislocation lines which can be visualized with the help of transmission electron microscopy (TEM) ,see figure 2.5.

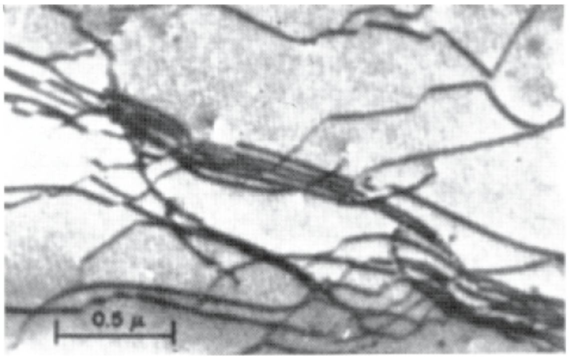


Figure 2.5.: Dark field image on a TEM to visualize dislocation lines in a material [7]

The dislocation density describes how many dislocations are located in a crystal or are present in a volume of a material. Since a dislocation is a line defect, it is defined as the total length of dislocation per unit volume. The unit of the dislocation density is consequently m/m^3, respectively 1/m^2. The magnitude of the density has usually of the order of 10^{10} m^{-2} in a metal, increasing to 10^{15} m^{-2} after work hardening [8].

The dislocation density increases with the internal stresses. The higher stresses are a result of a deformed microstructure or of a higher hardness (figure 2.6).

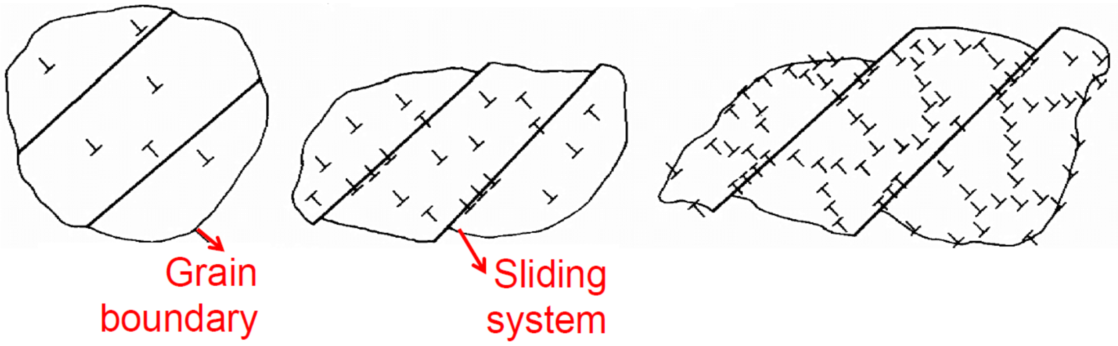


Figure 2.6.: Schematic increase of the dislocation density in the microstructure [9]

2.1.2. X-ray sources

X-rays can be used for non-destructive analysis of solids. The basic principle is the diffraction and interference of X-rays on periodic ordered lattice atoms. Diffraction analysis can be done on organic, inorganic and biological materials and allows to make a prediction about the atomic structure of almost every crystalline material. The microstructure of solids influences the diffraction of X-rays significantly. Hence it is possible to get out some aspects of the microstructure in the characteristics of the diffracted X-rays.

2.1.2.1. Generation of X-rays

X-rays are generated when highly accelerated electrons interact with structures and get slowed down. There are two basic processes [10]:

a) The bremsstrahlung occurs when a fast electron gets deflected and slowed down in the electric field near an atom nucleus (figure 2.7 left). By this interaction, the energy loss during deceleration is emitted as electromagnetic radiation with short wavelength, the so called continuous X-ray spectrum, see figure 2.8.

b) If the energy of the electron is high enough to excite or ionize an electron near the atomic nucleus, then this empty position gets filled up with another orbital electron (figure 2.7 right). The energy-difference between the two states from the electron gets emitted as the so called characteristic radiation, see figure 2.8.

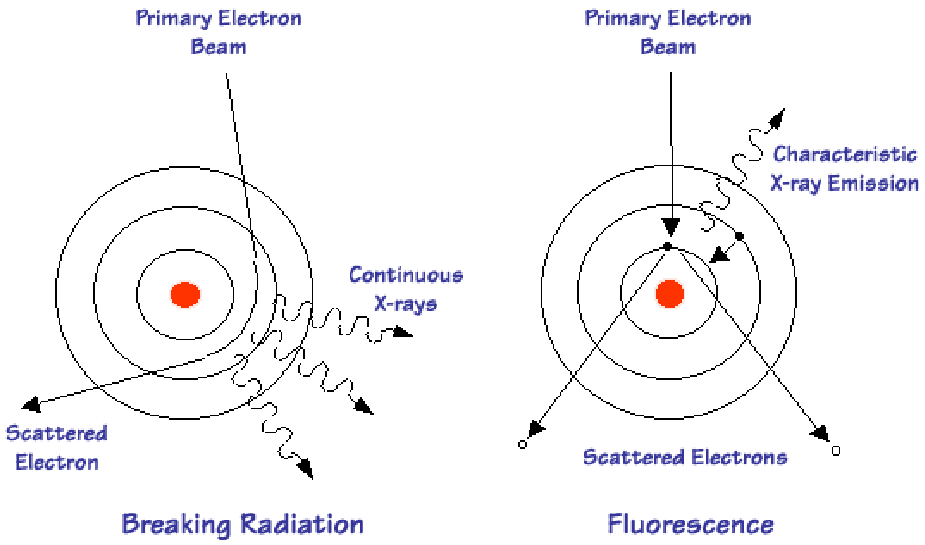


Figure 2.7.: Two basic principles of origination of X-ray radiation [11]

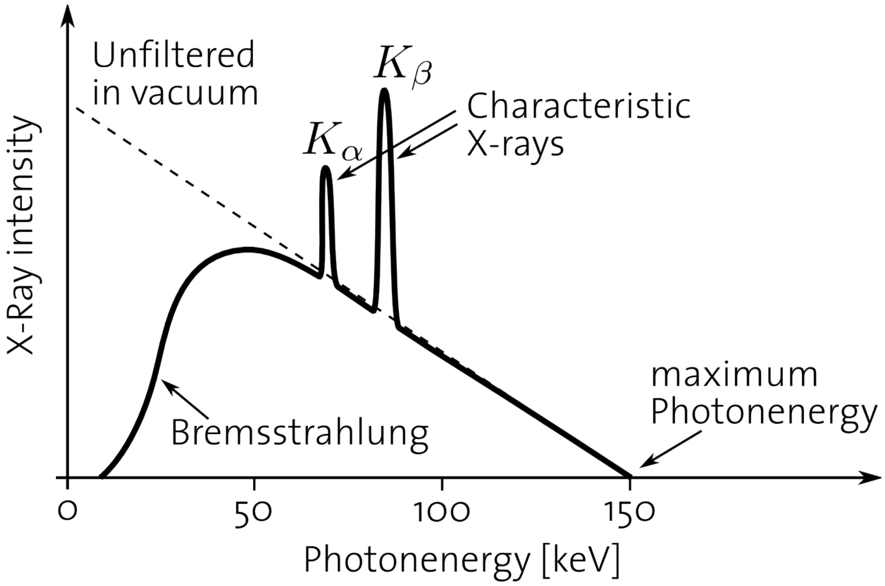


Figure 2.8.: X-ray emission spectrum with the bremsstrahlung and the characteristic X-ray radiation [12]

The most common X-ray source for structural analyses is an X-ray tube with a copper target (figure 2.9). In an x-ray tube electrons emitted by the glowing cathode K are accelerated towards the target anode A due to the high tube voltage U_a. As the electron beam impacts on the target, the high kinetic energy of the electrons is partially converted into x-ray photons. Much of electron energy is released as heat and the target should be cooled by a water stream (W). U_a is the heating voltage applied to the cathode [13].

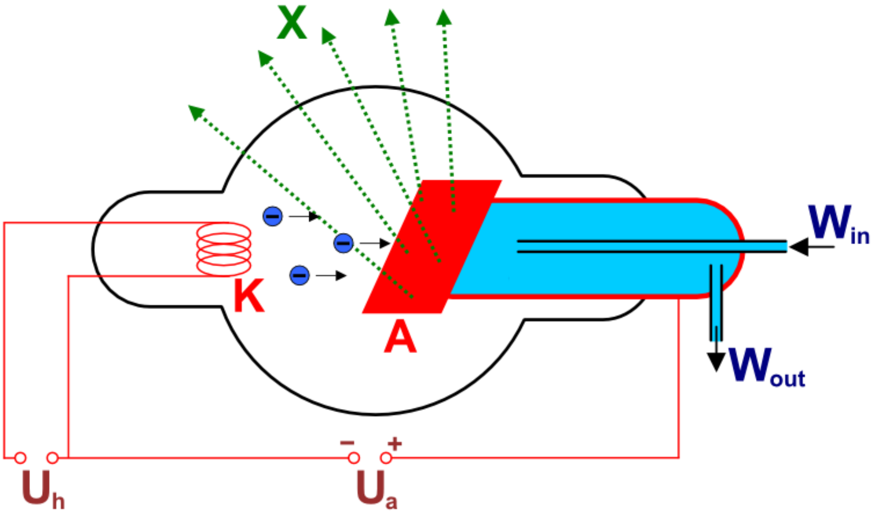


Figure 2.9.: Schematic sketch of a common X-ray tube [14]

The tube emits radiation with an energy of about 8 keV. With that energy, a penetration depth of several microns can be a achieved in steel. That is enough for the most structural analysis near the surface. If higher radiation energies are needed, it is not possible to realise this with the basic X-ray generation tubes. Another advanced method for hard X-ray radiation is needed.

2.1.2.2. Synchrotron radiation

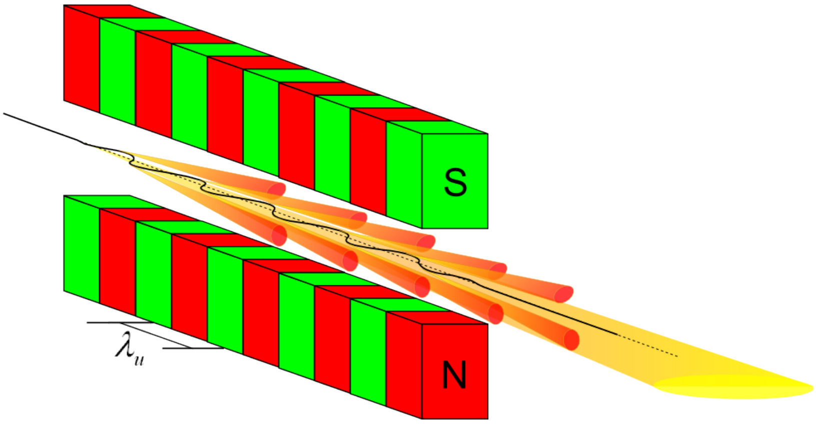


Figure 2.10.: Illustration of the operating mode of an undulator to bend the electron beam [15]

Like in classic X-ray tubes, the radiation originates by the deflection of electrons. Instead of deflection by atoms, in the synchrotron deflection is caused by an alternating magnetic field through which the electrons travel. As a side effect, the emitted radiation is aligned as observed in figure 2.10. Furthermore, a special circular arrangement figure 2.11 enables the generation of a concentrated electron beam for the subsequent X-ray generation.
These X-ray radiation is decoupled from the beam and induced into measuring chambers where the operators can make their experiments.

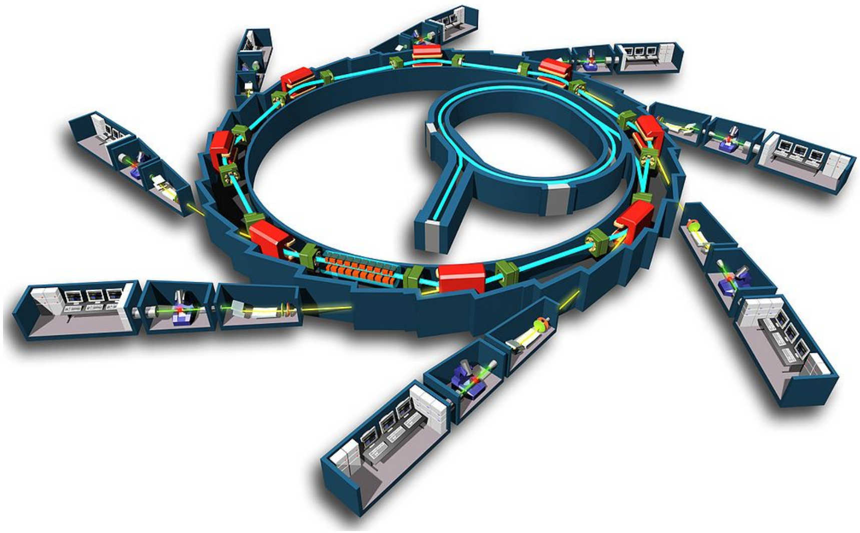


Figure 2.11.: Schematic sketch of the synchrotron SOLEIL [16]

2.1.3. X-ray diffraction

2.1.3.1. Interference

Electromagnetic Radiation with a constant wavelength and a fixed phase relation is called coherent radiation. Coherent radiation has the ability to interfere with other waves and with itself. Constructive interference means that the waves are in phase and amplify each other. If the waves are phase shifted a half wavelength to each other, the waves are annihilated and destructive interference occurs (see figure 2.12a).
Coherent light that enters a crystal gets reflected at the atoms of the crystal and exit the lattice at the same angle they enter. As observed in figure 2.12b, constructive interference occurs at a certain angle.

Geometrical considerations lead to the relation

$$n \cdot \lambda = 2 \cdot d \cdot \sin(\Theta), \tag{2.1}$$

with $n\lambda$ as the multiple of the wavelength λ, the lattice constant d and the diffraction angle Θ. This equation is called the Bragg condition.

Due to the high number of atoms and the deviation of the orientation of the crystal lattice, statistically always some constructive interferences occur. But in comparison to the Bragg condition, the observed intensity is rather low.

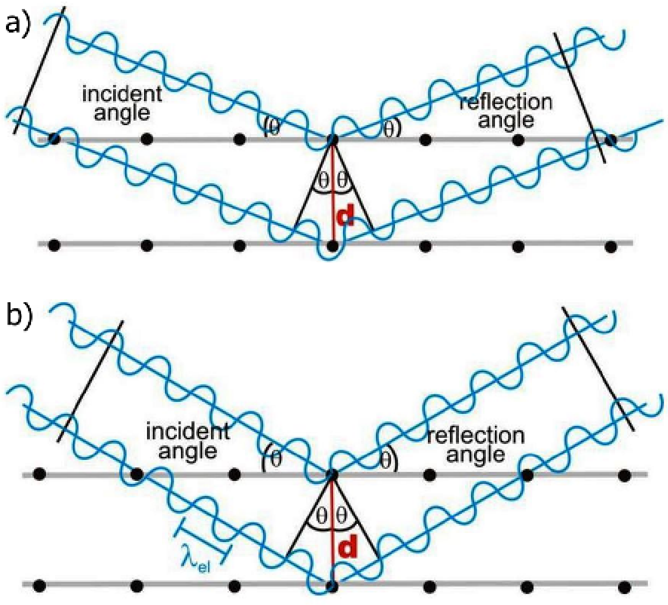


Figure 2.12.: Geometric illustration of a) destructive interference and b) constructive interference, which represents the Bragg condition [17].

2.1.3.2. Diffraction diagram

A crystal is a highly ordered structure where atoms are arranged in crystal-lattices. The X-rays will be diffracted on the characteristic planes. In dependence of the lattice orientation several planes can be defined (figure 2.13).

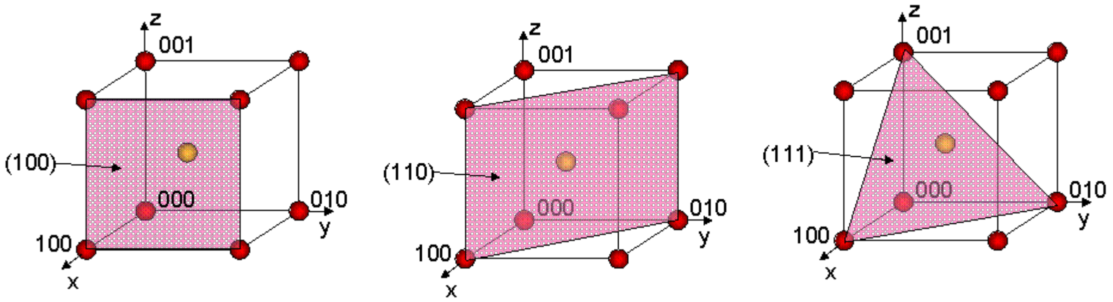


Figure 2.13.: Different plane orientations in a crystal [18]

When the intensity is plotted versus the Bragg-angle 2Θ, characteristic signal peaks occur at certain angles, such an angular scan can be observed as shown in figure 2.14.

In this diffraction diagram, the peaks represent the different orientations of the planes where the Bragg-condition is fulfilled. At the orientations with higher orders, the X-rays have a lower diffraction probability and hence a lower intensity in the diffraction diagram. The

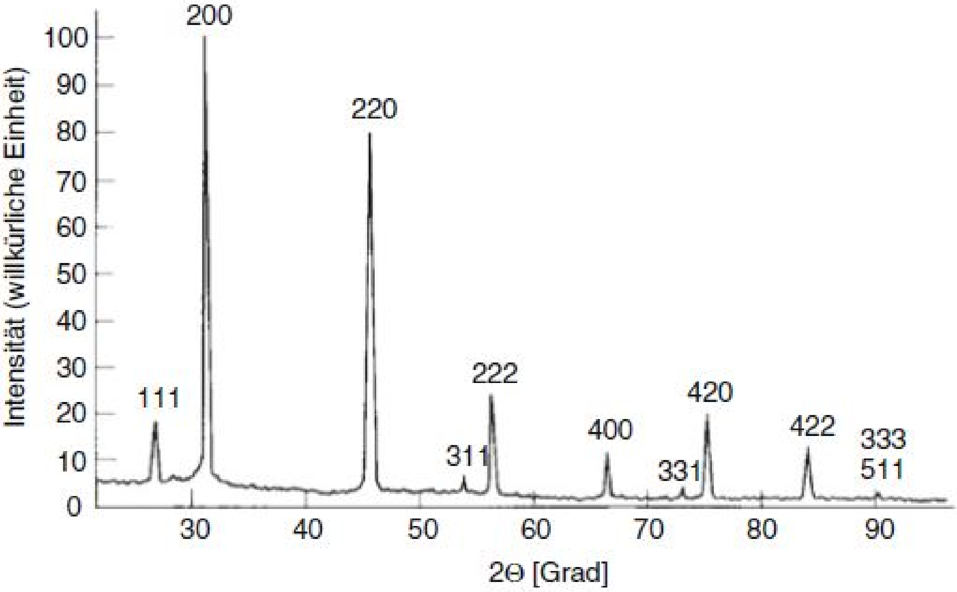


Figure 2.14.: Example of diffraction diagram of NaCl powder [7]

reason for these effects are geometrical conditions.

In contradiction to the theoretic ideal model, real crystals have a limited volume and poly crystalline materials have a finite geometry. Because of these issues there is talk of coherent scattering regions CSR. These are the regions in the sample, where the X-ray radiation is coherent scattered when the Bragg condition is complied [19].

The diffraction diagram carries three informations: the diffraction angle, the intensity of the diffracted radiation and the shape of the diffraction peak. Beside these characteristics, there is another measured signal, the so called background radiation. This radiation originates mostly from the coherent and incoherent scattering on the material and the surrounding air.

1) Diffraction angle 2Θ

As it can be seen in figure 2.12 and figure 2.14, the diffraction angle 2Θ is the characteristic angle for constructive interference. This angle is defined by size and symmetry of the unit cell and by the wavelength of the used X-ray radiation. With the angle of the diffraction peaks, the dimensions of the unit cell respectively the indication of the lattice planes can be determined. Hence a qualitative phase analysis is possible [19].
For example, for a cubic structure, the relationship between diffraction angle 2Θ and the lattice parameter a is

$$a = \frac{\lambda}{2\sin(\Theta)}\sqrt{h^2 + k^2 + l^2} \tag{2.2}$$

with the wavelength λ of the X-rays, the half diffraction angle Θ and the miller indices h, k, l for the indication of the lattice planes.

2) Intensity

The intensity of the diffracted radiation is affected by the type and arrangement of the atoms in the crystal. Hence, information of the three dimensional arrangement of the atoms in a crystalline material can be obtained. Also the experimental setup, like intensity of the primary beam, the sample volume and the measuring setup influence the measured intensity. The scattered intensity generally depends on several factors which are treated in more detail in the thesis D. Egger [10].

3) Peak shape

In further analyses (following section 2.1.4), the shape of the diffraction peak allows conclusions about the particle size and micro strains which are caused by structural defects.

2.1.4. Mathematical analysis of the dislocation density

The diffraction diagram contains many informations of the microstructure of a material. Hence it is necessary to characterise and evaluate the measured data precisely to obtain as much information as possible. Beside the position and the intensity of the peak, it is the shape of the peak itself, which contains interesting informations about the microstructure, like grain size or structural defects. In the next sections, the evaluation of these informations together with the required mathematical background will be discussed.

2.1.4.1. Peak broadening

Every real crystal is imperfect, that lead to a broadening of the ideal peak at the Bragg angle, towards a Gaussian distribution as illustrated in figure 2.15.

On the one hand there is the broadening from the X-ray diffractometer itself, with the non monochromatic X-ray source, the limited collimation of the radiation and the limited precision of the measuring sensor. On the other hand there is the broadening due to small crystal size and crystal defects [10].

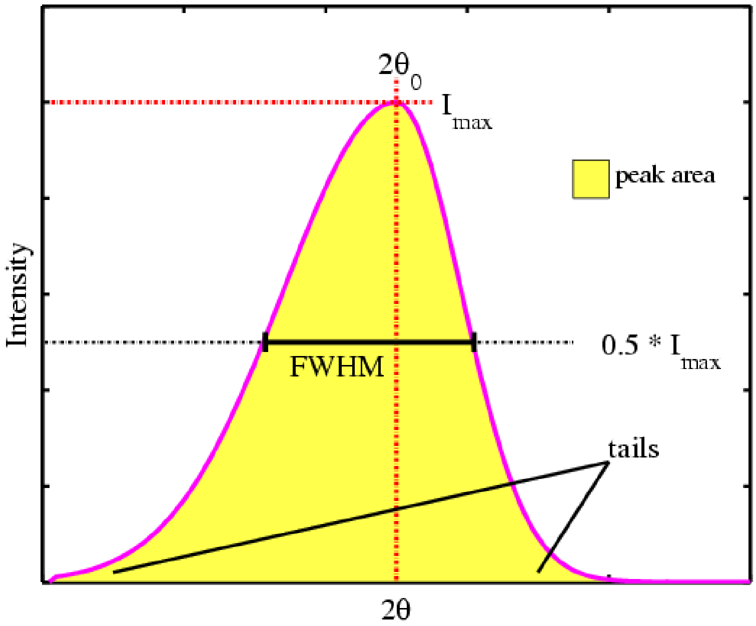


Figure 2.15.: Shape of a diffraction peak with the characteristic values of the peak [20].

So, the profile of the diffraction peak is a composition of the instrumentation and the sample influence. The three main influence factors will be considered separately as follows:

1) Instrumentation

Beside the analysed signal from the sample, an undesired contribution to the measured signal is included in the broadening. This contribution is caused by the used X-ray source and the focusing arrangement of the radiation path.
The profile of the peak $h(\varepsilon)$ is the convolution of the pure diffraction profile $f(\varepsilon)$ and the weight function of the apparatus $g(\eta)$:

$$h(\varepsilon) = \int_{-\infty}^{+\infty} g(\eta) f(\varepsilon - \eta)\ \mathrm{d}\eta \tag{2.3}$$

The function g expresses the total sum of the apparatus effects upon the pure function being measured. The variable ε measures the angular deviation of any point from the theoretical scattering angle $2\Theta_0$. In equation (2.3) ε and the auxiliary variable η possess the same dimensions as 2Θ [21].

2) Crystal size

The broadening of the diffraction peaks caused by a small crystal size can be compared to the diffraction of monochromatic light at a grid. A limited number of scattering centres broaden the diffraction pattern. The fewer slits a grid consists of, the more broadened and slurred the interference pattern becomes. With the same principle, the X-ray radiation is diffracted on a limited number of lattice planes [22].

3) Crystal defects

Every defect in the microstructure deforms the crystal structure and is therefore a deviation from the perfect lattice.
The more defects in the microstructure are, the more the peak gets broadened (figure 2.16), shifted and asymmetric deformed [21].
Due to the fact that there are so many broadening factors, this thesis only treats the influence of the dislocations in more detail.

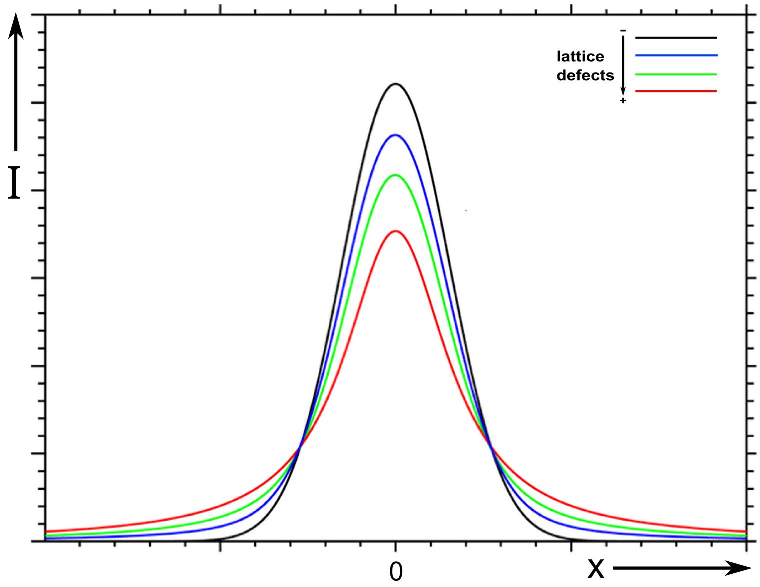


Figure 2.16.: Exemplary diagram of the broadening of the diffraction peak by defects in the microstructure

2.1.4.2. Mathematical evaluation

To determine the dislocation density from the measurements, the modified Williamson-Hall method is used. This method and the Warren-Averbach analysis are classical methods of the line-profile analysis [23]. Contrary to other methods, the exact shape of the peaks is not analysed. Instead, the full width at half maximum (FWHM) is investigated, see figure 2.15.

Williamson and Hall suggested that the FWHM (ΔK) of line profiles can be written as [24]:

$$\Delta K = 0.9/d + \Delta K^D \tag{2.4}$$

and

$$K = \frac{2\sin(\Theta)}{\lambda}, \tag{2.5}$$

where ΔK^D is the strain contribution to line broadening, d is the average grain size, Θ the diffraction angle and λ the wavelength of X-rays.

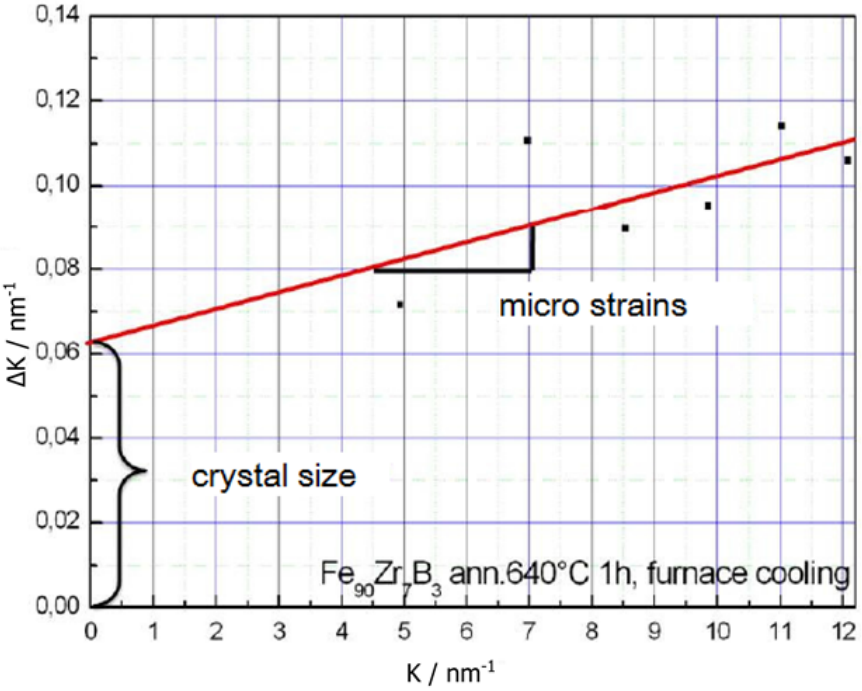


Figure 2.17.: Williamson-Hall plot of a $Fe_{90}Zr_7B_3$ diffraction profile [10]

From this plot, the value for the crystal size can be derived, because at K = 0 there can not be a broadening due to distortion (see figure 2.17). Additionally the micro strains which caused structural defects can be determined by considering the slope of the Williamson-Hall plot. The higher the slope is, the bigger is the contribution to the broadening caused by distortions.

For the case, that the distortion is caused by dislocations, the modified Williamson-Hall plot (figure 2.18) is used. In this diagram is not ΔK versus K plotted, but ΔK versus $KC^{\frac{1}{2}}$, with C the contrast factor of the current peak. The contrast factor considers the anisotropy of the peak broadening that occurs from the different directions of the diffraction vector. Considering the influence of the dislocation contrast to the distortion part of the line broadening ΔK^D, the equation (2.4) results in [24]:

$$\Delta K = 0.9/D + (\pi A b^2/2)^{1/2}\rho^{1/2}(K\bar{C}^{1/2}) + (\pi A' b^2/2)Q^{1/2}(K^2\bar{C}) \tag{2.6}$$

with the constants A and A', the length of the burgers vector b, the dislocation density ρ, the averaged contrast factor $\bar{C}$ and the correlation term for the dislocations Q.

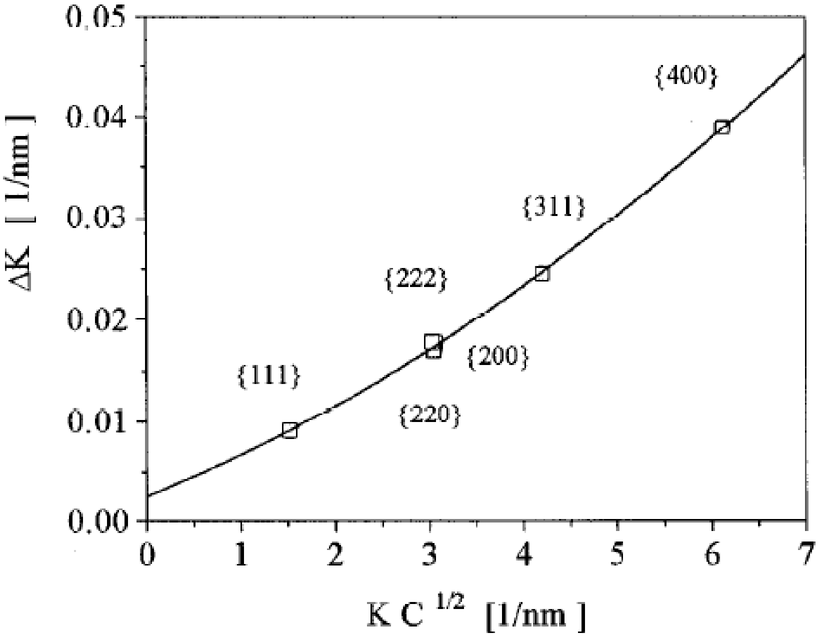


Figure 2.18.: Modified Williamson-Hall plot - example [24]

The modified Williamson-Hall analyses uses the fact that the influencing factors are not homogeneous, this contributes to the broadening of the peak. The broadening caused by dislocation density and the dislocation contrast raises not monotonous with the diffraction angle [25].

In further numerical analysis the dislocation density ρ can be determined [24].

2.2. Experiments and measurements

To get an idea for what is possible, there were first experiment with a XRD-measuring device executed of the Institute of Solid State Physics at Graz University of Technology. With this device it should be possible to see some Bragg-peaks of the material to get a first microstructural insight.

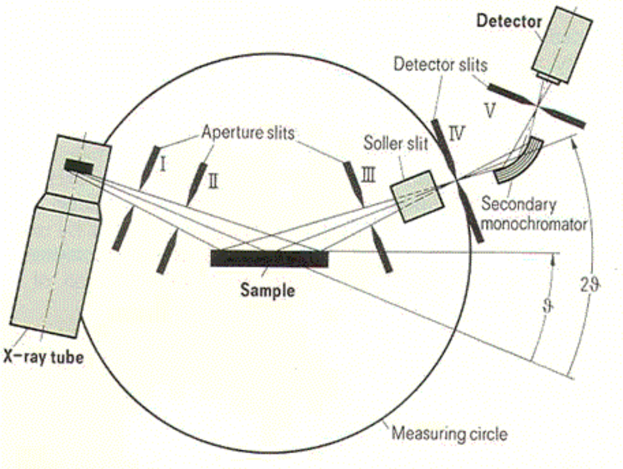


Figure 2.19.: Experimental setup of the 2Θ diffractometer [26]

In this constellation (figure 2.19), the diffraction is measured in reflection mode, so the surface roughness is a critical factor. For sufficient experimental data, the surface of the sample has to be as smooth as possible.

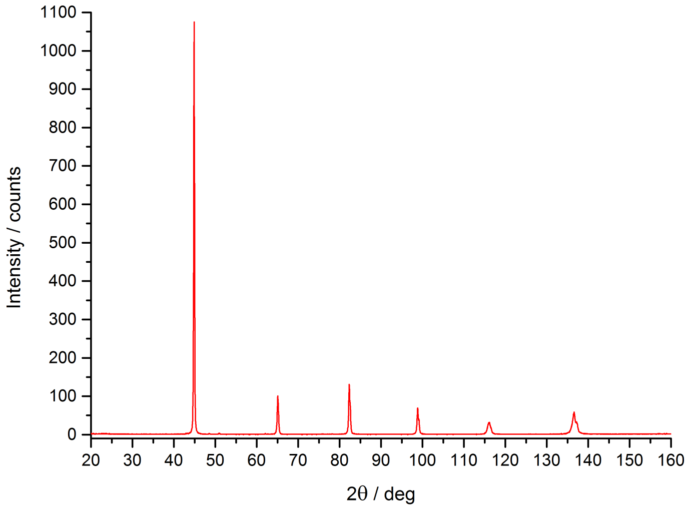


Figure 2.20.: Diffraction diagram of the 2Θ measuring in reflection mode.

For a mathematical analysis of the dislocation density the area of peak bottom is important. The conclusion of this first measurement is, that the intensity is far too low at the tails to get a reasonable calculation of the dislocation density (figure 2.20).

To solve this problem either the measuring time can be raised or a stronger X-ray source can be used, for example from a circular accelerator.

2.2.1. Experimental setup - synchrotron

In cooperation with the Faculty of Physics at the University of Vienna some measurements at the synchrotron in Grenoble have been carried out.

At the synchrotron, the measuring is not in reflection mode like on a diffractometer with low radiation power, but in transmission mode like shown in figure 2.21. The beam is directed at the sample and behind it the detector system is located. The detector system consists of a small detector that scans the whole area step by step. The three dimensional measurement is converted into a two dimensional diffraction diagram, only with the scattering angle and the intensity as output.

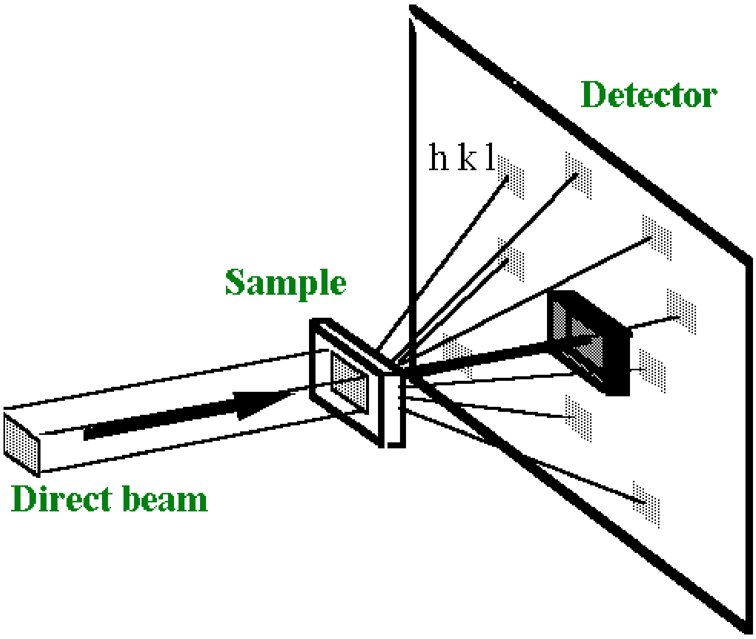


Figure 2.21.: Experimental setup on the beam-line [27]

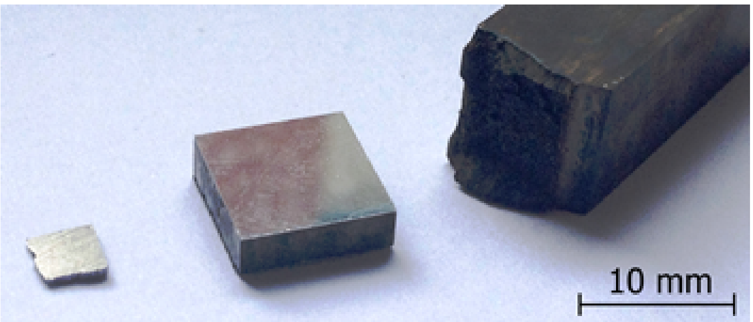


Figure 2.22.: Preparation of the sample

Main difference for sample preparation is, that the spectrum is recorded in transmission mode and not in reflection mode as in the first measurements. Therefore the sample has to be prepared as a thin plate (figure 2.22).
The thickness of the sample has to be chosen as thin as possible, but as thick as necessary, so that no information of the microstructure is lost. Hence the lower limit of the thickness is the grain size of the microstructure. If that lower limit is not reached, many information of the grain boundaries get lost. With these considerations the final geometry of the sample is 4x4x1 mm.

On one sample six measuring spots are analysed to get an average value of the microstructure. The spot size depends on the beam and in case of the beam line in Grenoble, the spot has a diameter of 50 μm.

2.2.2. Analysed material

The analysed material is a creep resistant chromium steel and its weldment. The welding joint can be separated in three different zones (see figure 2.23 and figure 2.24).

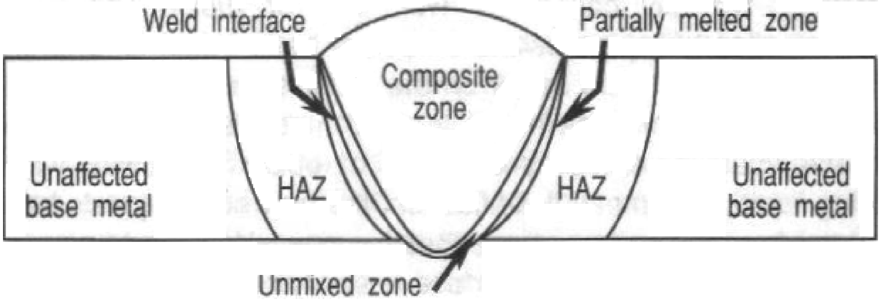


Figure 2.23.: Characteristic areas of a weldment [28]

The differnet zones can be characterised as follows:

1. Base material: CB2 cast metal
2. Heat affected zone (HAZ)
3. Multilayer weldmetal of the welding joint

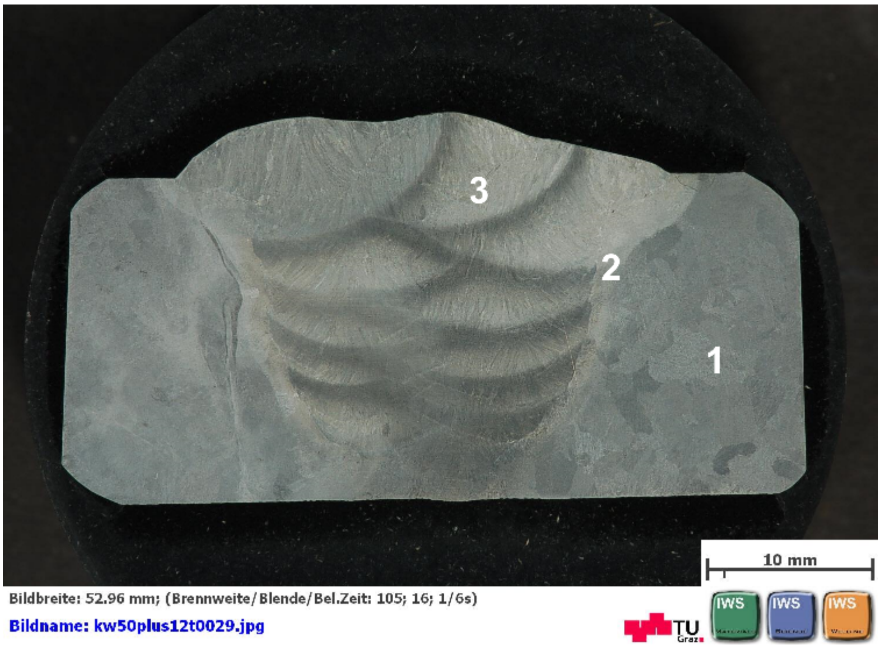

3
2
1
10 mm
Bildbreite: 52.96 mm; (Brennweite/Blende/Bel.Zeit: 105; 16; 1/6s)
Bildname: kw50plus12t0029.jpg
TU Graz
IWS
IWS
IWS

Figure 2.24.: Cross section of a weldment with marks of the different zones [2]

The base material and the HAZ are chemically the same material, but in different heat treatment conditions. The weld metal is chemically slightly deviating from the base material, but the main elements are contained in the same quantity.

The base material CB2 is investigated in its initial state as it is delivered from the foundry (stress relieving heat treatment at 730°C / 12 h).
The HAZ was physically simulated at the Gleeble 3800 machine. So in fact it is a CB2 cast metal with a special heat treatment to simulate a characteristic part in the HAZ. The weld metal itself, is produced from a rutile CB2 filler wire in seven layers and 15 passes [2]. The sample is directly cut out of center of the weldment.

2.3. Results

In cooperation with the University of Vienna, these three different samples were investigated and evaluated.
For the dislocation density measurements different heat treatment states of these samples are possible:

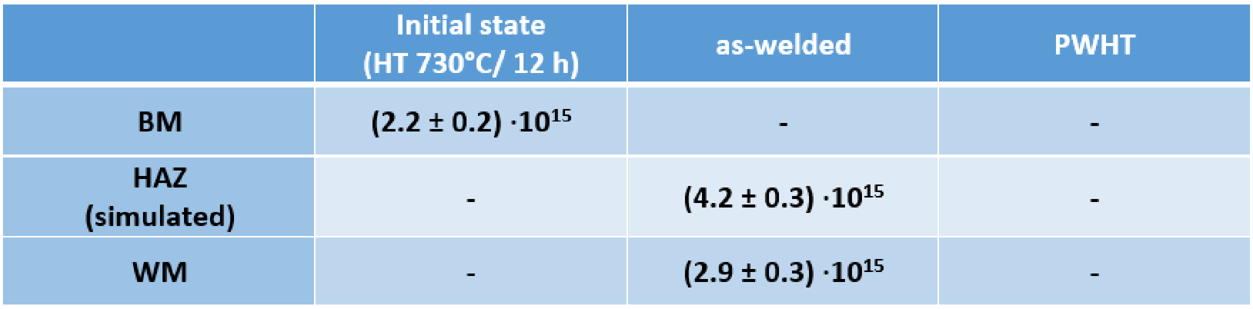
a) Initial state

b) As-welded state

c) Post weld heat treatment (PWHT) state

For each material only one interesting state was chosen.

Table 2.1.: Results of the dislocation density ρ of the analysed materials



	Initial state (HT 730°C/ 12 h)	as-welded	PWHT
BM	(2.2 ± 0.2) $\cdot 10^{15}$	-	-
HAZ (simulated)	-	(4.2 ± 0.3) $\cdot 10^{15}$	-
WM	-	(2.9 ± 0.3) $\cdot 10^{15}$	-

The results of the dislocation density are in the expected magnitude of 10^{15} m^{-2}, [29].
The base material (heat treated at 730°C / 12 h) is the one with the lowest dislocation density, this confirms that the material with the most homogeneous and relaxed microstructure is the material with the lowest amount of defects (see table 2.1).

The weld metal itself has a higher amount of dislocations, due to the higher thermal energy input during the welding process. Through multilayer techniques the upper welding layers are partially heat treating the lower layers. Therefore the investigated microstructure is divided in regions with low internal stresses (heat treated, fine grained regions) and high internal stresses (dendritic region). A hardness profile picture (figure 2.25) shows the differences in hardness and hence gives an idea of the expected dislocation density in an as-welded state.

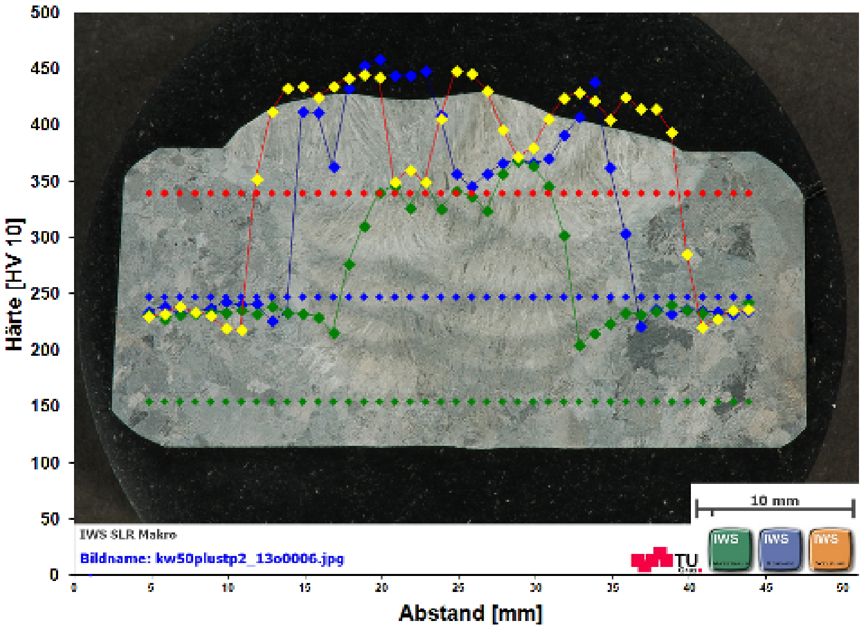


Figure 2.25.: Hardness map of a weldment

It is unclear from which region of the welding joint, the sample for the synchrotron was cut out and where the measuring spots were set. So, it is also unclear if the measured values come from a region with high internal stresses, a region with low internal stresses or from a mixed zone. Because the dislocation density is not much higher than in the base material, it is estimated that it was from a zone with low internal stresses and hence a low hardness.

The heat affected zone has the highest dislocation density. In producing the Gleeble simulated HAZ, high heating and cooling rates were applied, so high temperature gradients have obviously caused the highest internal stresses.

Discussion

A multilayer weld with a completely inhomogeneous microstruture was investigated. The HAZ consists of differently grained regions and even the martensitic base material is not really homogeneous [30, 31].
The problem is, that the measurement requires a defined minimal spot size. If the microstructure varies in this area, an averaged value is measured. This restricts the integrity of the results of this investigation.

3. Impact toughness

Second important objective of this thesis was to get a better understanding of the impact toughness behaviour of a material. With the instrumented Charpy V-notch test much data during the fracture of a specimen is obtained. To evaluate the big amount of data, a computer-program was developed in the course of this project.

3.1. Literature

3.1.1. Toughness

In general, toughness is the ability of a material to absorb energy and deform plastically without fracturing [32]. It is the resistance against a sudden and uncontrolled crack propagation (appendix A.1), which is typical for a brittle fracture [33].

A simple notch influences the fracture behaviour of a material, deformability and ductility can change into brittleness without plastic deformation. Higher loading rates and lower temperatures also lead to a more brittle behaviour.
It is determinable by different material tests, if a material is able to reduce stress peaks by plastic deformation at the crack tip [33].

A technological value to describe the fracture behaviour is the impact energy. It determines the amount of absorbed energy of a specimen during fracture. This value depends on the ductility of the material, which is among others a function of the temperature and the geometry of the specimen.

To define the fracture mode of a material, an "impact-energy versus temperature curve" must be created (figure 3.1). Then the transition from ductile to brittle failure depending on the temperature can be observed. A brittle fracture results in a low impact energy, while a ductile fracture results in a high impact energy. Between the brittle and the ductile fracture, at the transition temperature zone, a mixed fracture occurs.

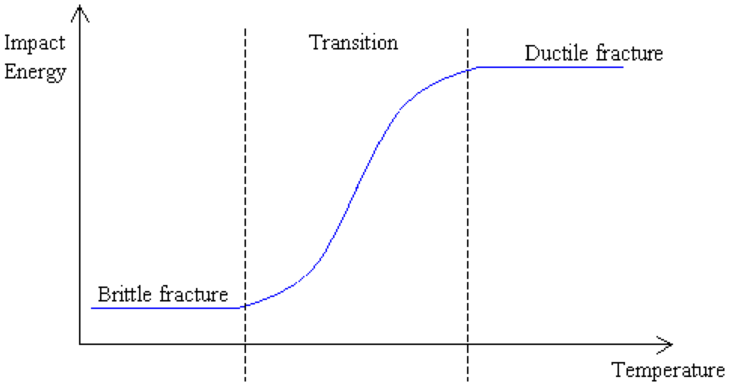


Figure 3.1.: Characteristic impact energy - temperature curve for a body centred cubic structured material [34]

Furthermore, the fractured surfaces of the sample have to be investigated as well, to determine the fracture mode.

Examples of three broken impact toughness specimen are shown in figure 3.2. The different fracture modes can be clearly defined, they are ductile (figure 3.2 right), brittle (figure 3.2 left) and mixed fracture (figure 3.2 center).

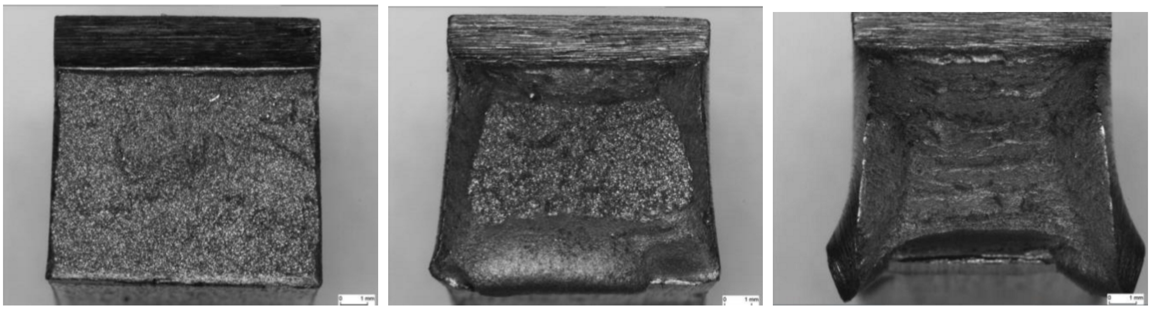


Figure 3.2.: Different types of the fracture [35]. Left) Brittle fracture. Center) Mixed fracture. Right) Tough fracture.

Since the Charpy V-notch toughness value can not be involved into fracture mechanic calculations, other experimental methods like the Crack Tip Opening Displacement (CTOD) test to determine characteristic material values have to be used [36].

This thesis only deals with the Charpy V-notch test, the instrumentation and its evaluation with the correct interpretation of the data.

3.1.2. Charpy V-notch impact test

The Charpy V-notch impact test is used to determine the impact energy as indicator for the impact toughness of a material.

In a simple experimental setup, a hammer on a pendulum breaks through a notched specimen. The residual height (h') is measured (see figure 3.3b) and the impact energy A_V can be determined.

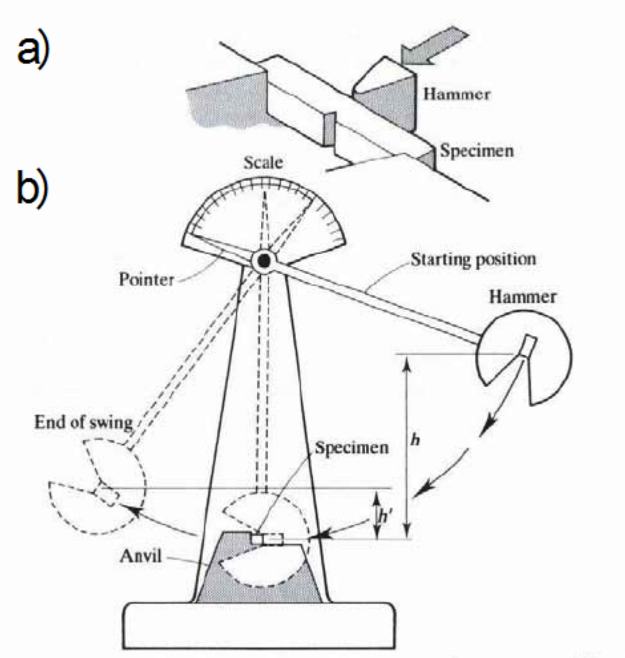


Figure 3.3.: a) Position of the specimen b) Principle set-up of the Charpy V-notch impact pendulum test [37]

The specimen has a standardised prismatic geometry of (55 x 10 x 10) mm and a V-notch for crack initiation with a depth of 2 mm and notch angle of 45° [38].

When the hammer breaks through the specimen, energy is needed for deformation respectively cracking the sample. This energy which is transferred to the sample during fracture is called the impact work. It can be obtained by calculating the height difference of the hammer before (h) and after (h') the test (see figure 3.3b).

$$A_V = W_1 - W_2 = mg(h - h') \tag{3.1}$$

A_V is the impact energy in Joule absorbed from the sample, m is the mass of the hammer and g the gravity constant.

This simple mathematical relation with the simple setup and its fast way of measurement has made this test so popular.

It should be mentioned that in later sections there is a difference between measured and calculated impact work. In this thesis the analogue ($\hat{=}$ measured) value is referred as A_V and the digital ($\hat{=}$ calculated) value as C_V.

The test is simple, but it has one big disadvantage. Only the energy which is absorbed of the sample can be measured, but no one knows what is happening during the fracture and how the measured value results. To solve this issue, the impact test can be equipped with an additional electronic measurement system.

3.1.3. Instrumented Charpy V-notch impact test

With the electronic measurement system more details during the fracture are gained. It is an advanced measuring setup where strain gauges in the hammer fin for force measurement are implemented. A second measuring system to determine the deflection of the sample is installed. The measuring signals are recorded in an external transient recorder with a high measuring frequency of 1 MS/s. Afterwards the data can be evaluated.

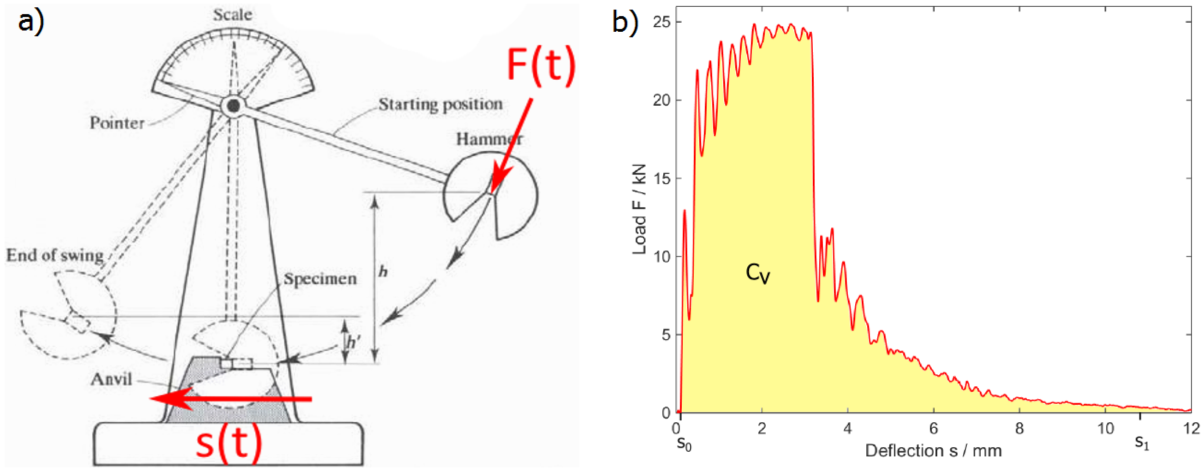


Figure 3.4.: a) Instrumentation of the impact test [37]. b) Recorded load-deflection curve of a fractured sample

The force F(t) and the deflection s(t) are both recorded as a function of the time (see figure 3.4a). In combining these two functions in one diagram, the impact energy C_V can be calculated by evaluation of the area under the load-deflection curve F(s) (see figure 3.4b).

The mathematical expression of the area under the load-deflection curve is given by the following equation (3.2). The impact energy C_V is the force $F(s)$ integrated over the deflection s.

$$C_V = \int_{s_0}^{s_1} F(s) \cdot \, \mathrm{d}s \tag{3.2}$$

3.1.3.1. Load measurement F(t)

The force is indirectly measured by four strain gauges (DMS) which are implemented the hammer fin. Two strain gauges are active and two are passive, their configuration is shown in figure 3.5. The active strain gauges build a force measuring element and with the passive compensation strain gauges, a full measuring bridge is established.

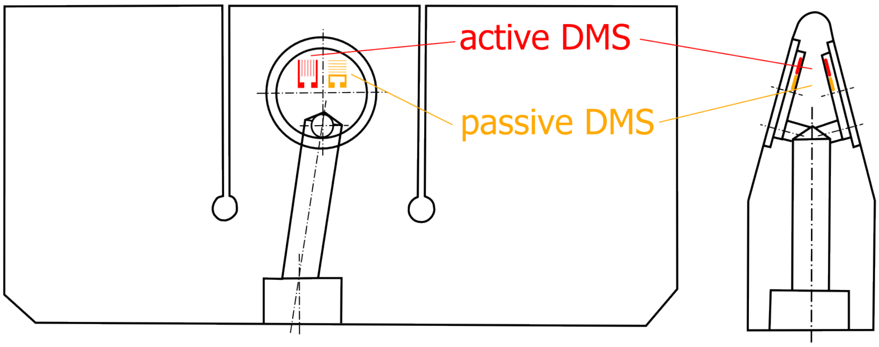


Figure 3.5.: Layout of the hammer fin with positions of the strain gauges

Because the strain gauges are quite far away from the fin edge, it is possible that the real force in the sample is not measured. Reason for this is the damping in the fin material. Even if this influence factor is quite small, it should not be completely disregarded.

3.1.3.2. Deflection measurement s(t)

For the measurement of the deflection of the sample, an inductive eddy current sensor in the bottom center of the pendulum is used. A metal chock on the bottom of the hammer and the inductive sensor build the measuring couple (figure 3.6 left).

When the hammer swings trough the bottom dead center and brings the chock over the sensor, a voltage is induced which is proportional to the deflection (figure 3.6 right).

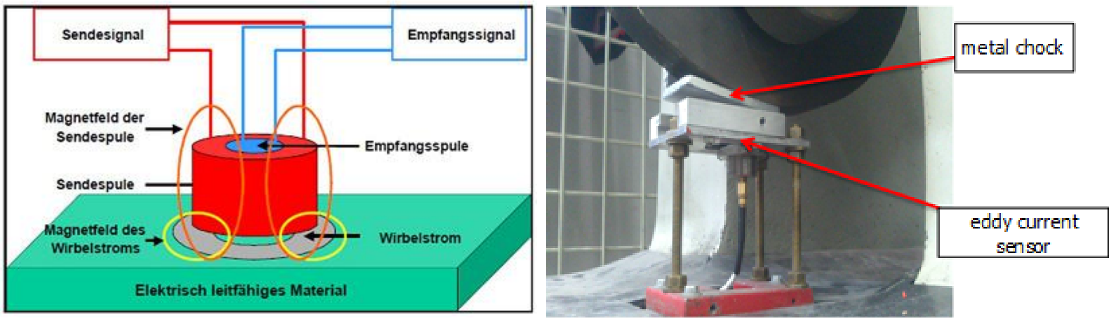


Figure 3.6.: Left) Schematic principal of a eddy current sensor [39]; Right) Position of the inductive sensor beneath the hammer

More precisely the eddy current sensor generates an alternating magnetic field that will be disturbed by the chock. The measured eddy current loss is proportional to the distance as it is expressed in the following equations:

$$U_{ind} = \frac{\Phi_m}{\mathrm{d}t} \tag{3.3}$$

with

$$\Phi_m \sim s$$

U_{ind} is the induced voltage, Φ_m the magnetic flux, t the time and s the deflection. The induced voltage U_{ind} is proportional to the time-dependent change of the magnetic flux Φ_m/dt (equation (3.3)) and this can lead to measurement inaccuracies, as it can be seen later in section 3.3.2.

This indirect measuring system generates a value for the the deflectuion of the sample as a funtion of time.

If there is no possibility to measure the deflection, it can be calculated as well.
Due to the fact that the potential energy of the hammer transformed into kinetic energy right before the impact and hence it has a defined impact velocity v_0, the deflection $s(t)$ can be calculated with simple kinematics in considering that the deformation of the sample will slow down the hammer [40].

$$s(t) = \int_{t_0}^{t_1} v(t)\ \mathrm{d}t \tag{3.4}$$

with

$$v(t) = v_0 - \frac{1}{m} \cdot \int_{t_0}^{t_1} F(t)\ \mathrm{d}t \tag{3.5}$$

So the deflection $s(t)$ is the change of the velocity $v(t)$ over time (equation (3.4)). Furthermore the velocity is the impact velocity v_0 reduced by the change of the force $F(t)$ over time, which is the momentum, with the indirect factor of the reduced mass m of the hammer (equation (3.5)).

3.1.3.3. Oscillations

As it can be seen in figure 3.4b, there are significant oscillations in the curve. It can be excluded that the measurement system produces these oscillations, because the measurement frequency of the transient recorder is several scales higher than the oscillation frequency. It is published that the origin of these oscillations can be a vibration system between the hammer and the impacted sample [41]. Until today, this issue is not clarified completely, but a recent work by Nakaruma et al. [42, 43] showed that these effects can be neglected in many cases.

3.1.4. Evaluation according to standard

For comparability of the evaluation from different testing laboratories, the german standard institute created the EN ISO 14556 [40] for the instrumented Charpy V-notch impact test. In this standard the evaluation of the load-deflection curve is regulated. Nevertheless, this standard admits much freedom to interpret the data.

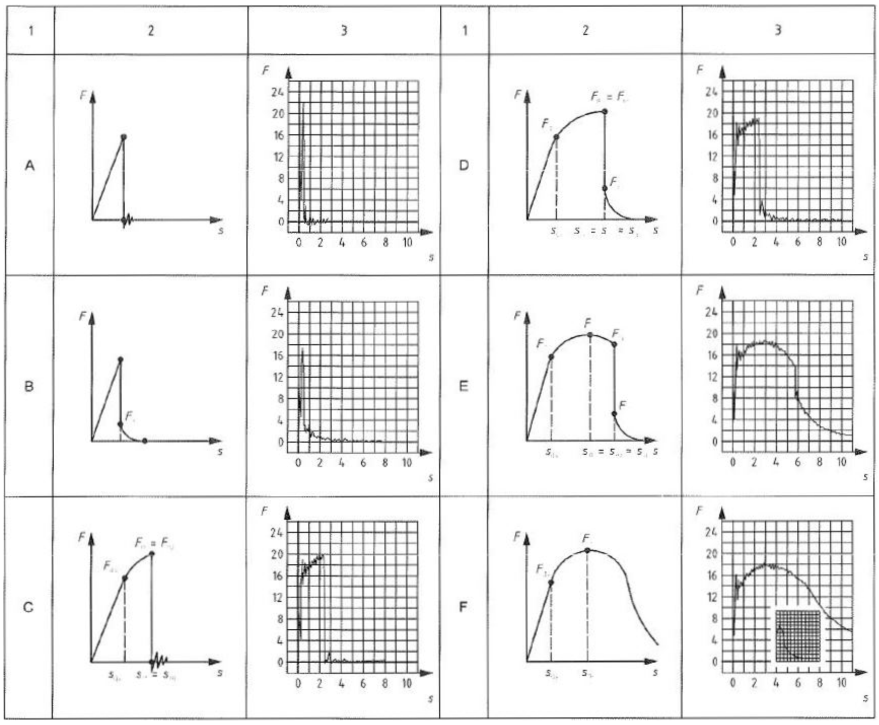


Figure 3.7.: Different types of fracture curves - EN ISO 14556 [40]

Generally the Charpy V-notch test and its load-deflection curves are split up into six different fracture cases, shown in figure 3.7, and every case has its own characteristics.

Every possible characteristic value is depicted in figure 3.8 based on case E.

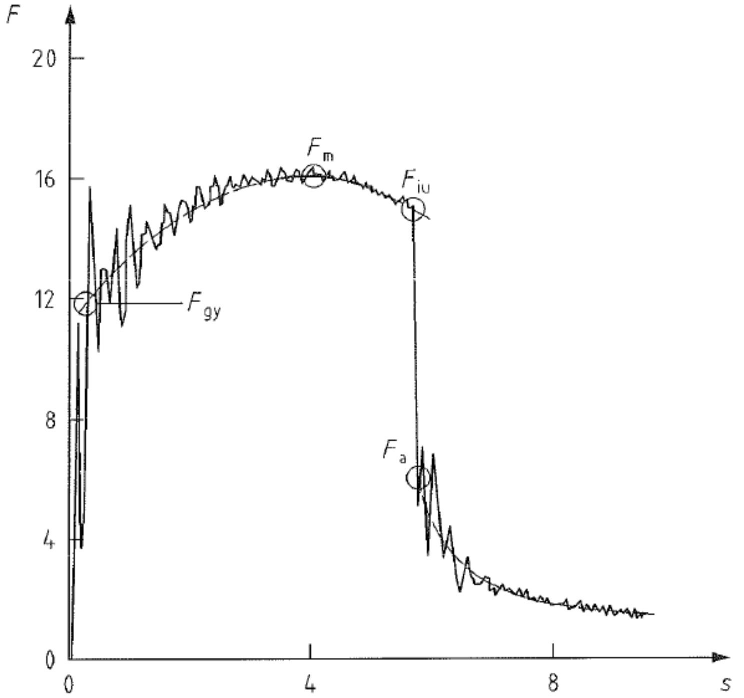


Figure 3.8.: Characteristic values in case E - EN ISO 14556 [40]

- F_{gy} = dynamic yield force
- F_m = maximum force
- F_{iu} = crack initiation force
- F_a = crack arrest force

From the impact until F_{gy} is reached, which is shown in figure 3.8, a linear behaviour is assumed.

Between F_{gy} and F_{iu}, a section where a stable crack propagation (appendix A.1) is assumed, the maximum force F_m is observed. This curve development between F_{gy} and F_{iu} can be described by a polynomial curve 2nd order.

The following sudden linear decrease of the force from F_{iu} to F_a, represents an unstable crack propagation until crack arrest.

The remaining rest of the curve can be interpreted as a further stable crack propagation, where shear lips of a fractured sample are formed. Until the complete break trough the development can be described by a polynomial curve 2nd order, too.

3.2. Evaluation program

Before the program routine was created in the frame of this thesis, the evaluation of the data and the setup of the correct fitting curves was very time-consuming. The results were usually dependent on the interpretation of the user who evaluated the data. To avoid this issue, a computer program based on the standard was developed. Now it is easier and faster for the operator to analyse the data and the program eliminates the individual interpretation of the user.

The created software is a standalone-program which uses the runtime codec of MATLAB. The MATLAB programming language is widely used by physicists and it has no difficulties to handle a big amount of data. A disadvantage is, that programming a user interface with MATLAB is in the early stages of development and not trivial. But the advantages of the easy data processing prevail.

3.2.1. Fitting rules

To reproduce the cases according standard, see figure 3.7, some fitting rules have been created for the program routine.

To enable the program to find all the characteristic points in the data plot, fitting rules have to be defined. These fitting rules are defined in this Master Thesis according to standard as follows:

- Case A: Linear up to F_{gy}=F_{max}
- Case B: Linear up to F_{gy}=F_{max} - polynomial fit 2nd order after crack arrest
- Case C: Linear up to F_{gy} - polynomial fit 2nd order from F_{gy} to F_{max}
- Case D: Linear up to F_{gy} - polynomial fit 2nd order from F_{gy} to F_{max} - polynomial fit 2nd order after crack arrest
- Case E: Linear up to F_{gy} - polynomial fit 2nd order from F_{gy} to F_{iu} - polynomial fit 2nd order after crack arrest

- Case F: Linear up to F_{gy} - polynomial fit 10th order from F_{gy} to the end

F$_{gy}$ is the intersection of the fitted plateau and the edge of the second oscillation of the measured curve. The plateau is fitted as a polynomial curve of 2nd order.

F_{iu} is the intersection between the polynomial fit of the plateau and the unstable crack propagation edge and F_m is the maximum value of the plateau. F_a is the intersection of the crack propagation edge and the remaining fitted data after the crack arrest. Only in case F no unstable crack propagation exists and the entire curve has to be fitted after the linear part at F_{gy}. For best alignment between fit and data, a polynomial fit with a high order has to be taken, in the evaluation program a polynomial fit of 10th order is used.

In Case A and B F_{gy}, F_{max} and F_{iu} are identical, in Case B additionally just F_a exists. In Case A there are only linear fit functions.
A correct evaluation and fitting is therefore not possible according to standard [40]. But for comparability Case A and B get treated in this thesis.

3.2.2. Program structure

In this section the main part of the program is explained step by step.

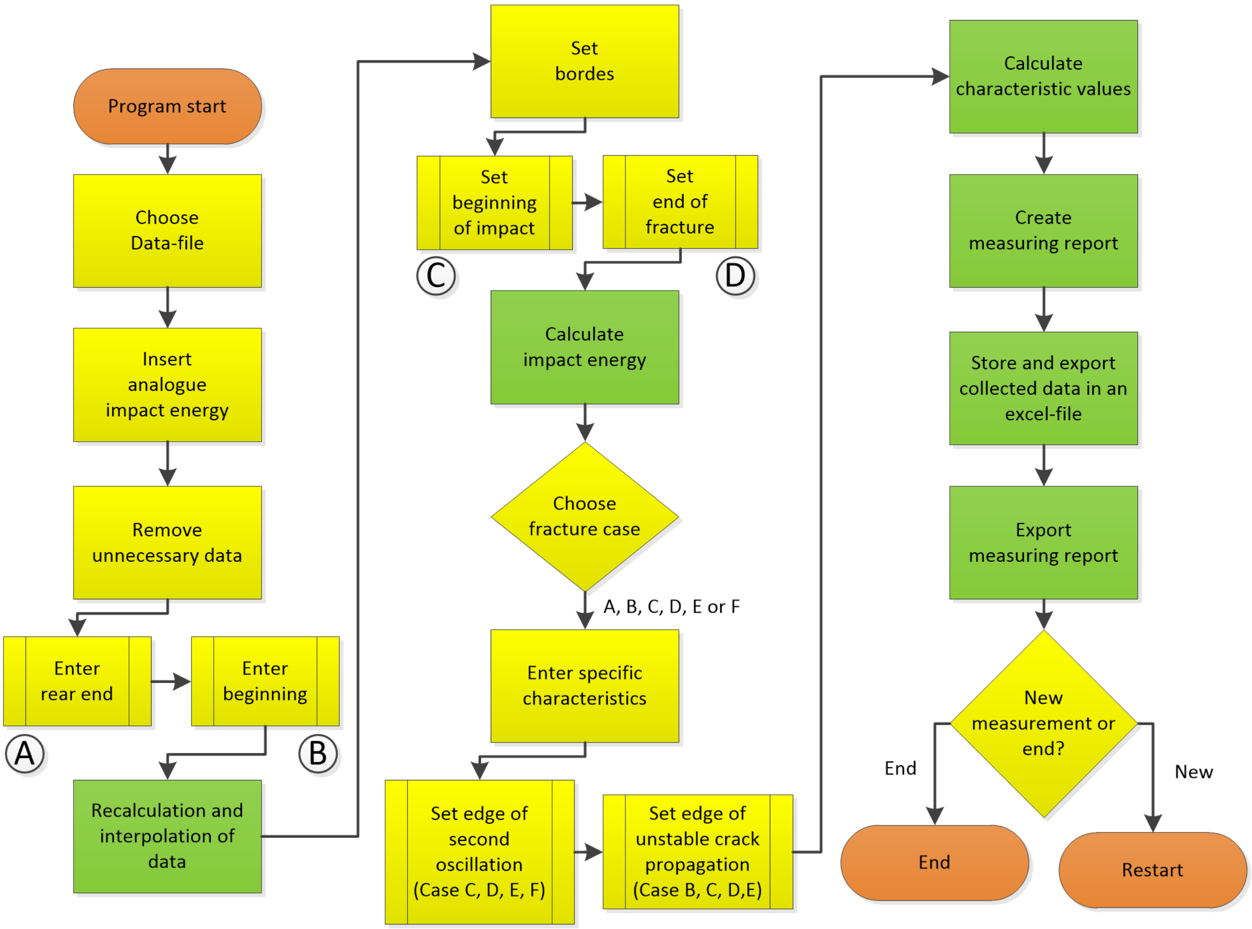


Figure 3.9.: Flow chart of the evaluation program. Yellow $\hat{=}$ user part, green $\hat{=}$ program part

Starting point is an ASCII-file from the transient recorder with the recorded test data of measured time, deflection and load. This raw-data file contains data points which are unnecessary for evaluation, because the transient recorder determines not only the part of the the fracture but also large sections before and after the impact.

First of all, the useless data are removed by the program, by asking the user what is the interesting part and requesting him to set the limits right before Ⓑ and shortly after Ⓐ the fracture (figure 3.10).
Afterwards the operator has to set the beginning Ⓒ of the impact. That point is used as new starting point and also as lower limit of the integral which is used afterwards for calculating the impact energy (figure 3.11 left).
For the impact energy a second limitation is needed at the point where the fracture is

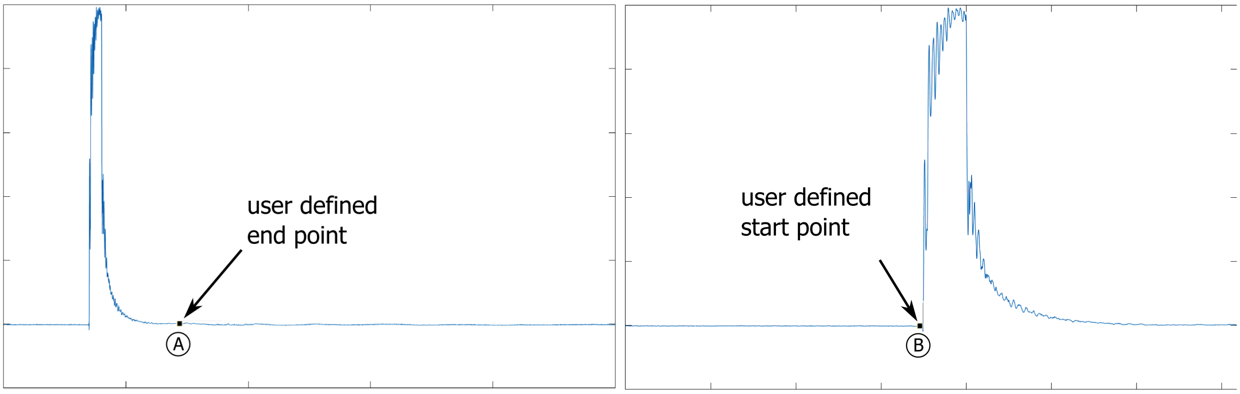


Figure 3.10.: Removal of unnecessary data

finished Ⓓ. It is responsibility of the user to set a reasonable end of the fracture, but as a thumb rule about two percent of the measured maximum force can be taken (figure 3.11 right).

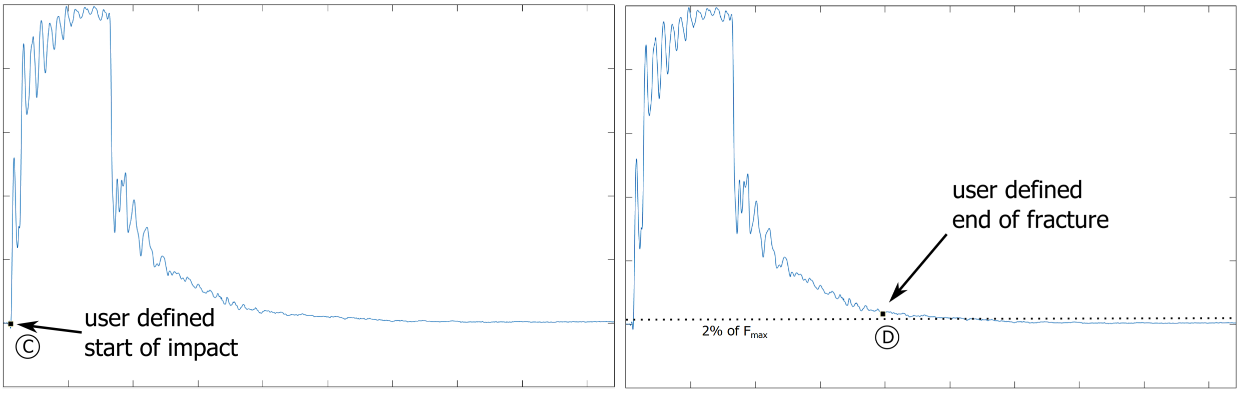
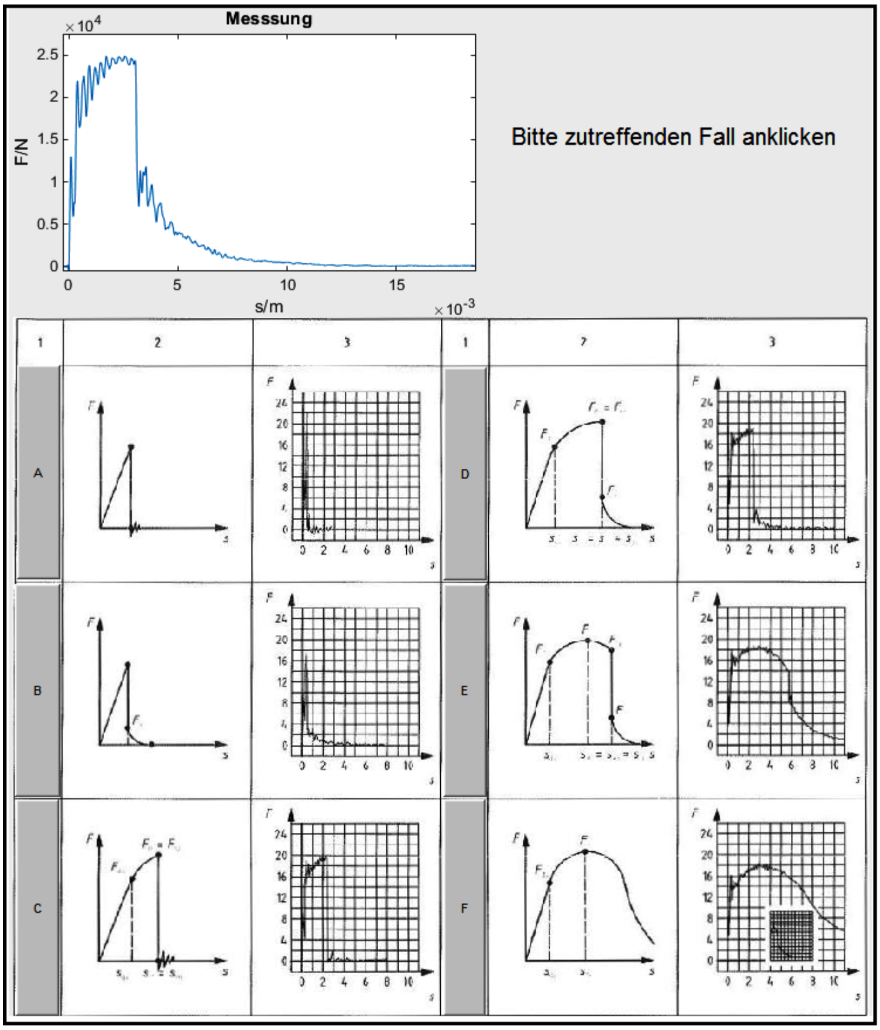


Figure 3.11.: Setting the limits of the fracture for calculation of the work function

When the parameters for calculation are set, only one more thing is missing. The program needs to know which standard case it has to take for calculating and fitting the measurements (figure 3.12). So, the user has to choose the correct case (1 out of 6) by comparing the shape of the measured curve with standard cases [40].
Now the program fits the curve according chosen standard case and evaluates important values. All the collected data are summarised in a test report.



Bitte zutreffenden Fall anklicken

Figure 3.12.: Selection of the correct fracture case by comparison with the standard

3.2.3. Test report

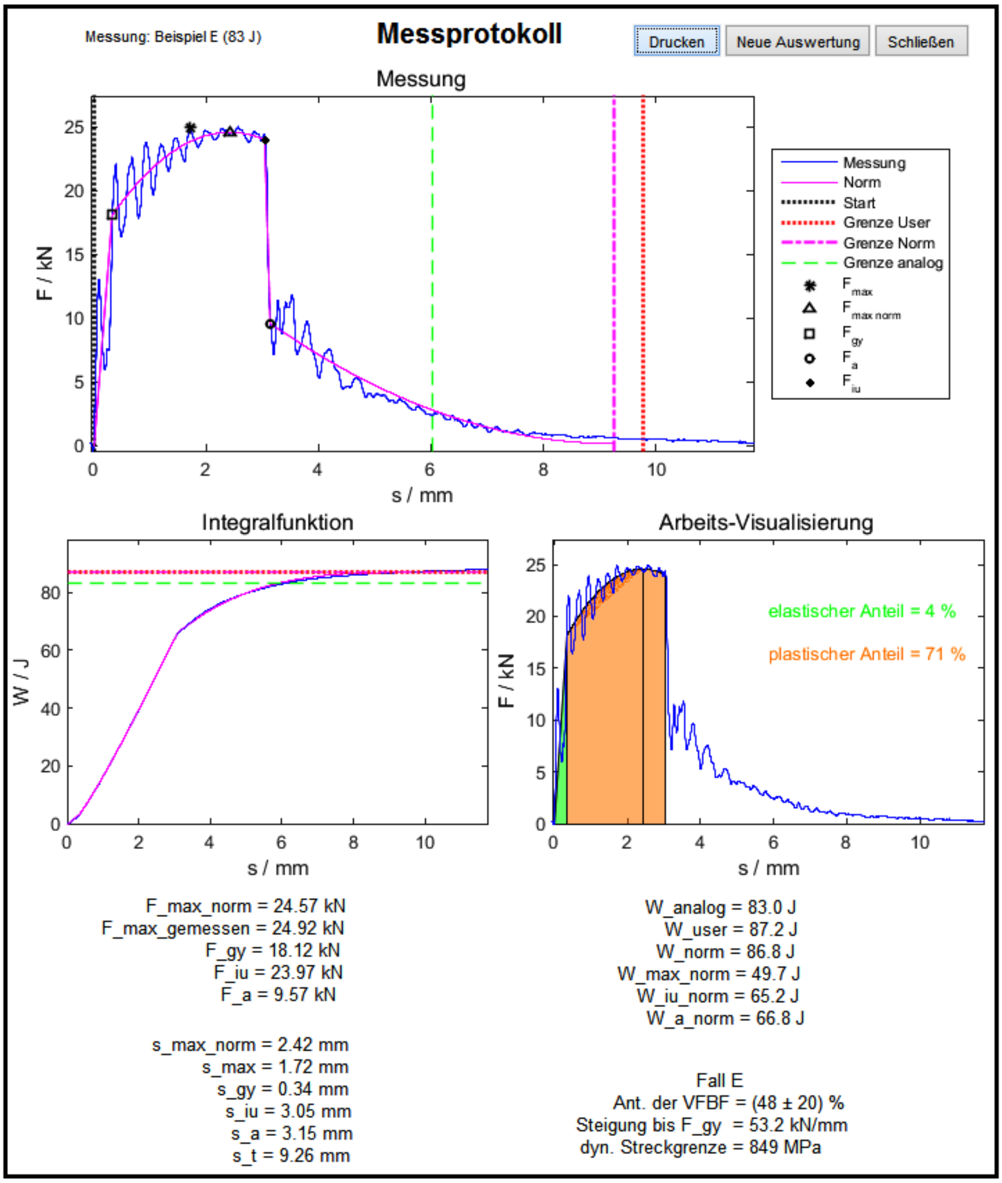
The test report is an important output of the evaluation program. In the report all evaluated data are documented and all interesting characteristic values are listed.

In the top figure (figure 3.13) the load-deflection curve of the measuring data (blue) and the fitted data (magenta) is shown. The vertical lines represent the calculation limits respectively the end of the fracture in different cases.

The left bottom figure illustrates the continuous integration of the load-deflection curve. It should be a helping tool for the user to know the deviation between the measured and the fitted data. If the lines match good, then the evaluation according to standard is reasonable.

The right bottom figure displays the percentage of the elastic and the plastic deformation part until failure.

Afterwards the test report is saved as a pdf-file and all collected data are exported and stored into an excel-file for later calculations and evaluations.



F_max_norm = 24.57 kN
F_max_gemessen = 24.92 kN
F_gy = 18.12 kN
F_iu = 23.97 kN
F_a = 9.57 kN

s_max_norm = 2.42 mm
s_max = 1.72 mm
s_gy = 0.34 mm
s_iu = 3.05 mm
s_a = 3.15 mm
s_t = 9.26 mm

W_analog = 83.0 J
W_user = 87.2 J
W_norm = 86.8 J
W_max_norm = 49.7 J
W_iu_norm = 65.2 J
W_a_norm = 66.8 J

Fall E
Ant. der VFBF = (48 ± 20) %
Steigung bis F_gy = 53.2 kN/mm
dyn. Streckgrenze = 849 MPa

Figure 3.13.: Summarised data in a test report

3.2.4. Program details

To understand what is calculated in background and how the program works, some details of the MATLAB-Code are listed. The complete code with all comments is placed in the appendix (appendix A.2.1).

Import of the measuring data

For the information of the location where MATLAB can find the data file, the user has to set the file-path. With the user interface command *uigetfile* the program reads out the file with *importdata* and saves the data in a new matrix.

```
[Dateiname,Dateipfad] = uigetfile('*.txt','Messung auswaehlen');
Datei = importdata(fullfile(Dateipfad,Dateiname));
Messdaten = Datei.data;
```

Input of the analogue impact work

In every test the analogue impact energy is measured with a trailing pointer on the device. For later visualization in the test report, the program requests the user to prompt the analogue value in a dialogue box with the *inputdlg* command.

```
prompt = {'Analogen Messwert eingeben und mit "OK" bestartigen:'};
dlg_title = sprintf('%s',regexprep(Dateiname,'.txt',''));
Schlagarbeit_analog = inputdlg(prompt,dlg_title,[1,length(dlg_title)+35]);
Schlagarbeit_analog = sscanf(sprintf('%s*',Schlagarbeit_analog{:}),'%f*');
```

Because the *inputdlg*-routine creates a cell-array, the value has to be converted in a double variable for further calculations.

Removal of unnecessary data

The transient recorder records more data than it is necessary for evaluation. So the user has to set manually two points, which the program uses as limitations and all other data above and below this limits get deleted.

```
set(fig,'units','normalized','position',[0.1,0.1,0.8,0.8],...
    'Name','Kraft-Zeit-Diagramm','Numbertitle','off','Toolbar','none',...
    'Menubar','none')
h = uicontrol('Style','togglebutton','String','und hier "bestaetigen"',...
              'units','normalized','Position', [0.5,0.53,0.16,0.03],...
              'Fontsize',12);
plot(Messdaten(1:length(Messdaten)*0.5,1),...
     Messdaten(1:length(Messdaten)*0.5,2))
```

```
Cursor = datacursormode(fig);
set(Cursor,'DisplayStyle','datatip','SnapToDataVertex','On','Enable','on');
waitfor(h,'value');
Cursorinformation = getCursorInfo(Cursor);
Endpunkt = Cursorinformation.DataIndex;
```

The user input is implemented with the *datacursormode*. The user has to set the cursor. The information is read out with *getCursorInfo* and the associated index information. Now the unnecessary data will be deleted.

```
Messdaten = Messdaten(Anfangspunkt:Endpunkt,:);
```

Reprocessing and interpolation of the data

The first important program feature is to reprocess the data for later calculations.
A big problem is that the data points of the deflection are not equidistant. Because of this issue the difference between two points gets partly negative and the load-deflection curve follows the trend like a loop. That does not occur in reality and also the subsequent interpolation routine can not handle that.

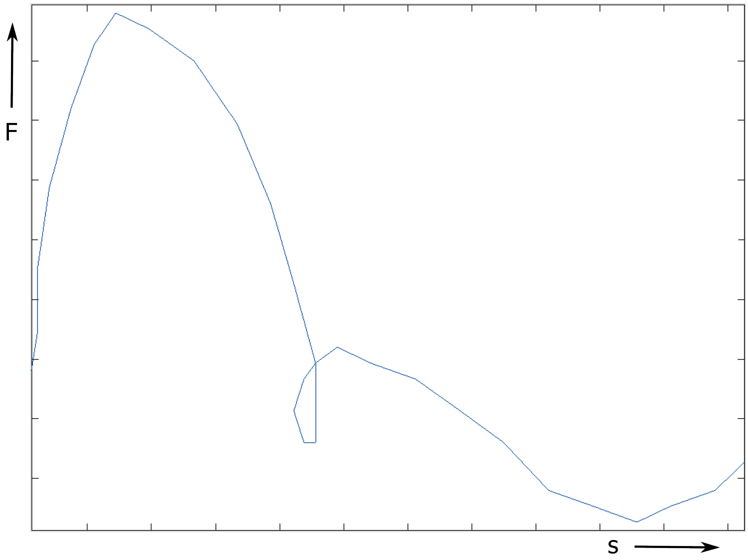


Figure 3.14.: Unwanted measuring loop in the recorded data

The solution is to find the loops and eliminate them, even if they are little and not important for the result.

```
Schleifenindex=find(diff(Messdaten(:,3))<=0);
while ~isempty(Schleifenindex)
    Messdaten(Schleifenindex,:)=[];
    Schleifenindex = find(diff(Messdaten(:,3))<=0);
end
```

To realize that, the program looks for a negative difference between two adjacent data points of the deflection and then delete the data pair with a negative difference.

Now it is possible to interpolate the data with the *spline*-routine. The program compiles a new deflection vector in the range of the measured points with equidistant values (*linspace*), then calculates the spline coefficients and recalculates the data with the piecewise polynomial evaluation routine *ppval*.

```
s_spl = linspace(min(Messdaten(:,3)),max(Messdaten(:,3)),length(Messdaten));
f_spl = spline(Messdaten(:,3),Messdaten(:,2));
F_spl = ppval(f_spl,s_spl);
```

Calculation of the impact work

Before the program can calculate the impact work with the *cumtrapz*-routine, the user has to set the integral limits. The *cumtrapz*-routine evaluates towards the Gaussian area formula.

```
Arbeit = cumtrapz(s_spl(Int_start:end),F_spl(Int_start:end));
```

The question rises, if the cumtrapz-routine is accurate enough. Of course it is, because so many data points exist, much more than necessary for a correct evaluation.

Choosing the correct standard case

A correct evaluation according to the standard only is possible when the program knows which case it has to take. With the MATALB-feature "imaging processing toolbox", a user interface is created, where the user has to choose the correct case per button click.
The user sees the actual load-deflection curve of the measurement in an extra axes-environment above the case selection.

```
axes(handles.axes1)
plot(s_spl,F_spl)
```

Beneath the load-deflection curve the six cases of the standard are shown in an image-environment.

```
Dateiname = 'Fallunterscheidung.jpg';
Bild = imread(Dateiname);
axes(handles.axes2);
imshow(Bild);
uiwait(handles.figure1);
```

There are implemented buttons with a specific callback-function, with which the user can choose the case and the program knows which one is taken.

```
function pushbutton1_Callback(hObject, eventdata, handles)
assignin('base', 'Fall', 'A');
```

Fitting the data according to standard

The next big program feature is the correct interpretation of the standard, in other words the correct fitting of the curve.

Main part is to find the intersections between the data curve and the standard fits as in section 3.1.4 explained.

To know approximately where the position of the intersection can be located, the user has to mark the characteristic edges of the measuring (figure 3.15).

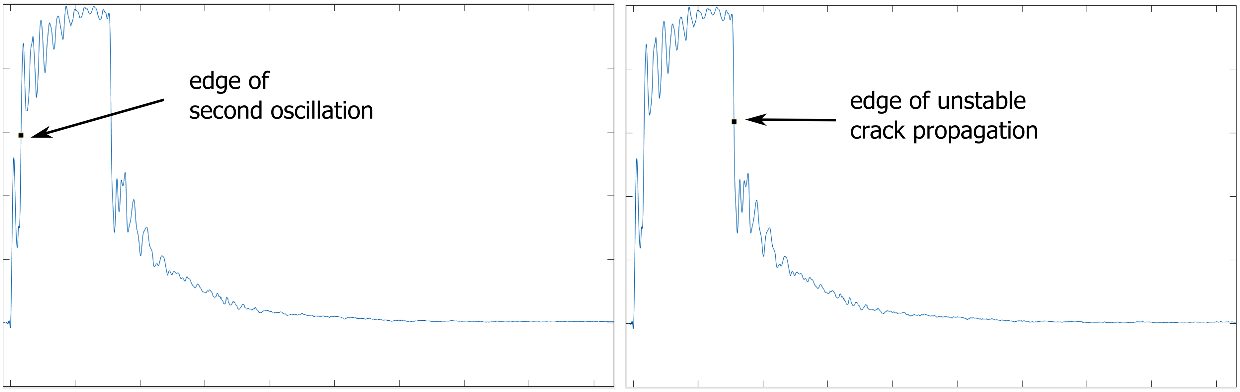


Figure 3.15.: Setting characteristic edges of the measuring data. Left) Marking of the second oscillation. Right) Marking of the unstable crack propagation edge.

With that information, the next zero point of the derivative of the curve can be located and used as search limitation.

```
ind = find(diff(F_spl)<0);
ind = ind-steigende_Flanke;
ind = ind(ind>0);
ind = ind+steigende_Flanke;
Fgy_ind = ind(1);
```

These limitations are also used as limits for the fit curves.

```
fit_plat = polyfit(s_spl(Fgy_ind:Fiu_ind),F_spl(Fgy_ind:Fiu_ind),2);
F_fit_plat = polyval(fit_plat,s_spl);
```

To find the closest data points to the intersection, the difference between the fit and the measured curve is calculated. The values before and after the zero difference build the intersection pairs. Now the intersection can be calculated in a system of linear equations.

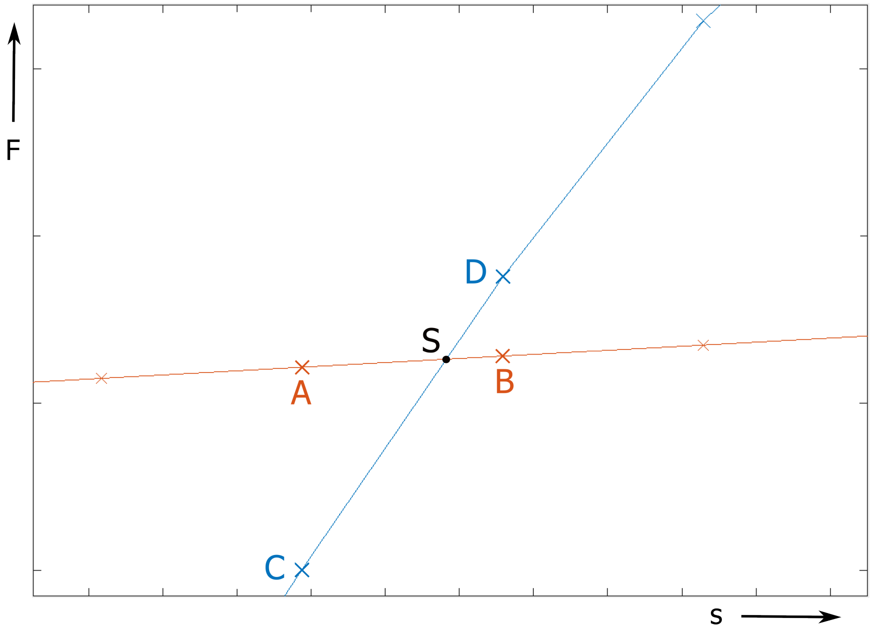


Figure 3.16.: Intersection between two lines. The nearest data points (A-D) to the intersection (S) are marked

```
M = [B(1)-A(1) D(1)-C(1);B(2)-A(2) D(2)-C(2)];
b = [C(1)-A(1);C(2)-A(2)];
x = M\b;
S = [A(1);A(2)]+x(1).*[B(1)-A(1);B(2)-A(2)];
```

Display the data in a measurement report

Now all necessary data are available and can be summed up and presented in a test report. Again the image toolbox is used. With that feature, any kind of text fields, buttons and graphs can be created in the report where the characteristic values and curves are shown. Additionally the report can be directly printed .

Export the data and the report

The report is exported automatically as a PDF-file with the same file name as the measurement file.

```
print(gcf,'-dpdf','-opengl','-r300',...
      regexprep(fullfile(Dateipfad,Dateiname),'.txt',''))
```

For eventually further processing, also all the collected data are saved in an extra excel-file.

```
Dateiname = regexprep(Dateiname,'.txt','');
Excelpfad = fullfile(Dateipfad,Dateiname);
xlswrite(Excelpfad,Export)
```

3.3. Data analysis

To test the program and to compare the analogue and the digital value of the impact work with help of the evaluation program (section 3.3.2), a first measuring series of a C45-steel was made.

3.3.1. C45 - measurement series

The C45-steel was chosen, because samples are widely available and this steel is already well studied, so no surprises were expected.

Table 3.1.: Charpy V-notch impact energy of C45-steel

temerature	#	A_V / J		C_V / J	
		measuring	average	calculation	average
-48°C	1 2 3	54 43 63	53 ± 10	54 40 62	52 ± 11
-20°C	1 2 3	66 50 50	55 ± 9	69 47 72	63 ± 14
-10°C	1 2 3	76 47 73	65 ± 16	84 47 80	70 ± 20
0°C	1 2 3	62 47 71	60 ± 12	64 47 77	63 ± 15
10°C	1 2 3	62 64 68	64 ± 3	65 65 71	67 ± 3
23°C	1 2 3	72 67 90	76 ± 12	74 70 102	82 ± 17
35°C	1 2 3	63 44 81	62 ± 19	64 40 89	64 ± 25
60°C	1 2 3	67 54 72	64 ± 9	70 55 77	67 ± 11

The series consists of the measurement of the impact energies over a temperature range from -48°C to 60°C. Each data point was recorded and the average of three measurements with the same parameters (same temperature) was calculated, as it is showed in table 3.1 and figure 3.17.

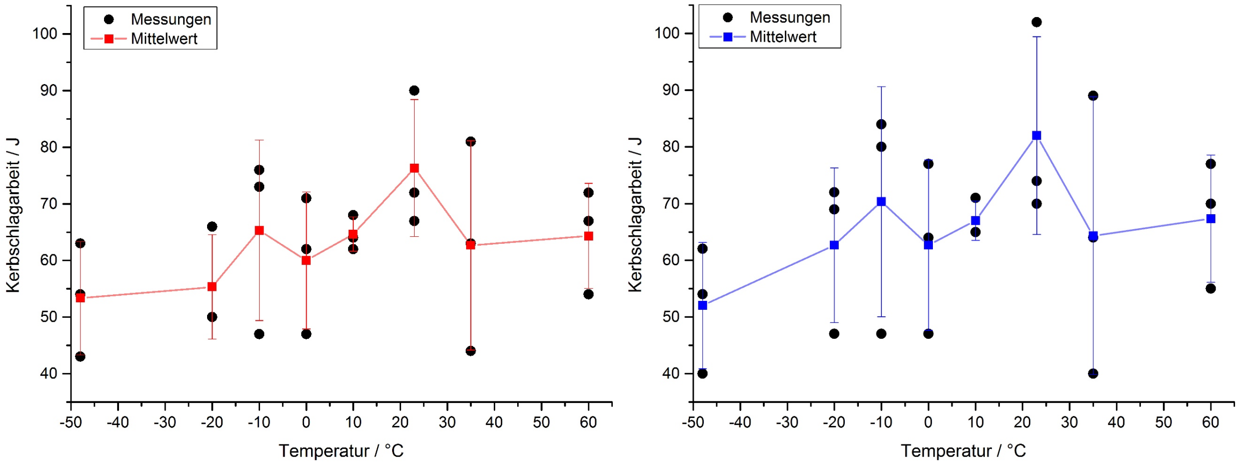


Figure 3.17.: Impact energy-temperature diagram of a C45-steel. Left) Measured analogue data (red average). Right) Evaluated digital data from the computer program (blue average).

All samples have nominally the same preparation method and heat treatment (quenched and tempered 45 min at 550°C), nevertheless the data of this steel shows at the first glance an unexpected unsteady behaviour of absorbed energy as shown in figure 3.17. The big variation of the impact energy at same measuring parameters make a reasonable evaluation of the transition temperature, the upper shelf and the lower shelf, insensate. But in comparison with the literature [44], it is obvious that the temperature range of the measurement is below the theoretical transition temperature about 70 °C (figure 3.18) and hence the measurement occurs right under the transition area in the lower shelf.

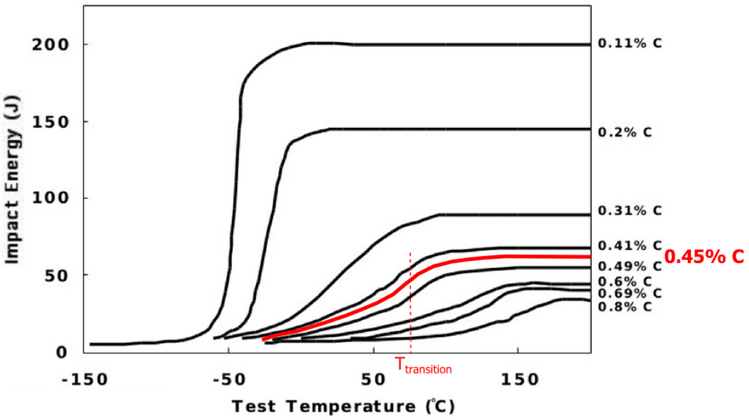


Figure 3.18.: Impact energy curves of carbon steels with variation of the carbon content [44]. In red is the theoretical trend of a 0,45% carbon steel (as the investigated C45).

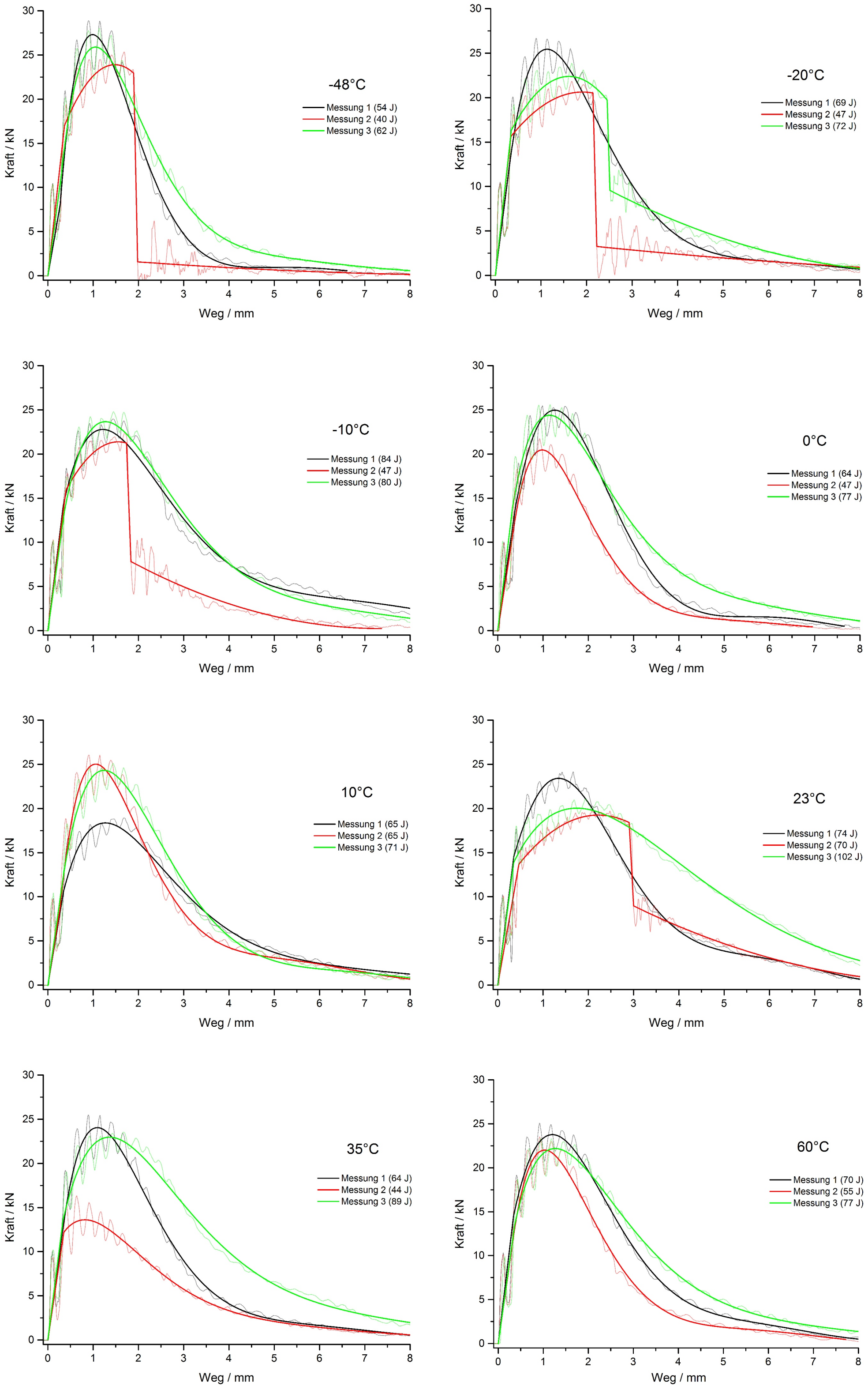


Figure 3.19.: Deflection-load curves of C45-steel at different temperatures

Interpretation

To have a closer look on the test series and to understand this issue better, every measurement with the same temperature is visualised in figure 3.19.

Instantaneous to see is, that the curves with the same parameters don't match among each other very good. Despite a sufficient statistic (three measurements with equal parameters), a good repeat accuracy can't be reached, because of the big variance of the values.

It also can be observed that there are fractures with an unstable crack propagation at lower testing temperatures. The theory implies that the more brittle a material is, the bigger is the chance for an unstable crack propagation and hence a lower impact energy. Because brittleness is a function of the testing temperature, it should be observed a higher probability for unstable crack propagation at lower temperatures. In the measurements (figure 3.19) no such a significant trend is obvious. Also at higher temperatures (room temperature) an unstable crack propagation is observable. The reason for that can be a different rolling direction of the microstructure, a different heat treatment of the sample, macroscopic defects in the microstructure or other still unknown influencing factors.

There is another interesting characteristic. The samples with the lowest impact energy are always the ones with the unstable crack propagation. Additionally the unstable ones have the lowest maximum strength. A reason for that can be, that the strengthening mechanism during the raising load [36] in these samples isn't at the same level as the other ones. So because of this issue they can not be the full load and additionally it comes to an unstable crack propagation that weakens the material too.

3.3.2. Comparison of analogue and digital impact energy

Through all analyses, a significant trend is occurred. The higher the impact energy is, the more the analogue and the digital value differ from each other.

With no further investigations, the possibility of a dynamic calibration of the impact energy after Winkler [45] exists. The objective of this mathematical calibration is to minimize the differences between the measured and calculated value.

The calculated calibration after Winkler considers the measured impact energy A_V and the dynamic behaviour of the hammer during fracture.

$$C = \frac{m \cdot v_0}{\int_0^t F'(t)dt} \left[1 - \frac{A_v}{\frac{m \cdot v_0^2}{2}} \right] \tag{3.6}$$

$$F(t) = C \cdot F'(t) \tag{3.7}$$

C is the calibration constant, m the mass of the hammer, v_0 the impact velocity, A_V the analogue value, F' the measured and F the calibrated impact load.

The combination of equation (3.2) and equation (3.7) shows as a result

$$C_V^{cal} = C \cdot C_V \tag{3.8}$$

with C_V^{cal} the calibrated calculated impact energy. Hence the impact energy C_V can be directly calibrated without further recalculations, once the calibration constant C is evaluated.

Because this calibration method only acts as the correction of load measurement errors, the desired adjustment on the impact value could not be achieved. Probably an inaccuracy of deflection measurement has a bigger influence on the result and hence no reasonable calibration is possible.

First of all, it is necessary to evaluate and remove the measurement inaccuracies before an implementation to the program makes sense and hence this method isn't used in the evaluation program yet.

In further investigations, two possible influence factors on the deflection measurement are treated in more detail:

3.3.2.1. Static calibration of the eddy current sensor

In figure 3.20 it can be observed, that at low impact energies the analogue value is higher than the calculated one and at higher impact energies it is the other way round.

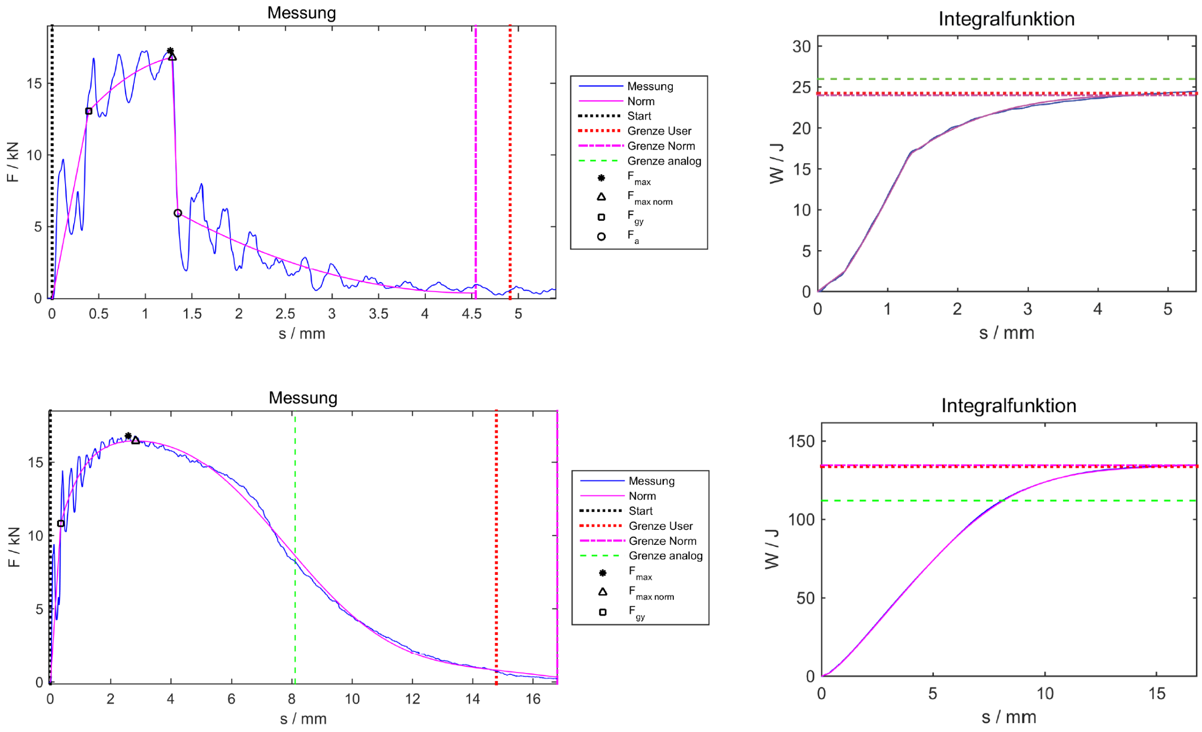


Figure 3.20.: Deflection-load curves of C45-steel at different temperatures with associated impact energy curve. The green dotted line represents the analogue value and the magenta dotted line the calculated digital value. Top) Brittle fracture; Bottom) Tough fracture.

The eddy current sensor is statically calibrated. That means that the deflection of the hammer is adjusted without a movement. Because the eddy current sensor is dependent on dynamic effects (see section 3.1.3.2), a difference between static and dynamic measurement can occur. A tougher material with a higher impact energy has a higher deflection until fracture, the eddy current sensor measures a higher deflection during movement than it really appears and hence a higher impact energy would be evaluated. If there are further investigations, the deflection measurement system has to be calibrated dynamically first.

3.3.2.2. Influence of oscillations

The load-deflection curve has these characteristic oscillations (see figure 3.4) that indicate a vibration of the hammer during fracture. These minimal vibrations can cause measuring inaccuracies, because of the dynamic behaviour of the eddy current sensor.

To get an overview how big these oscillations influence the measurement, the measured (black) and the calculated (red) deflection (equation (3.4)) are opposed against each other.

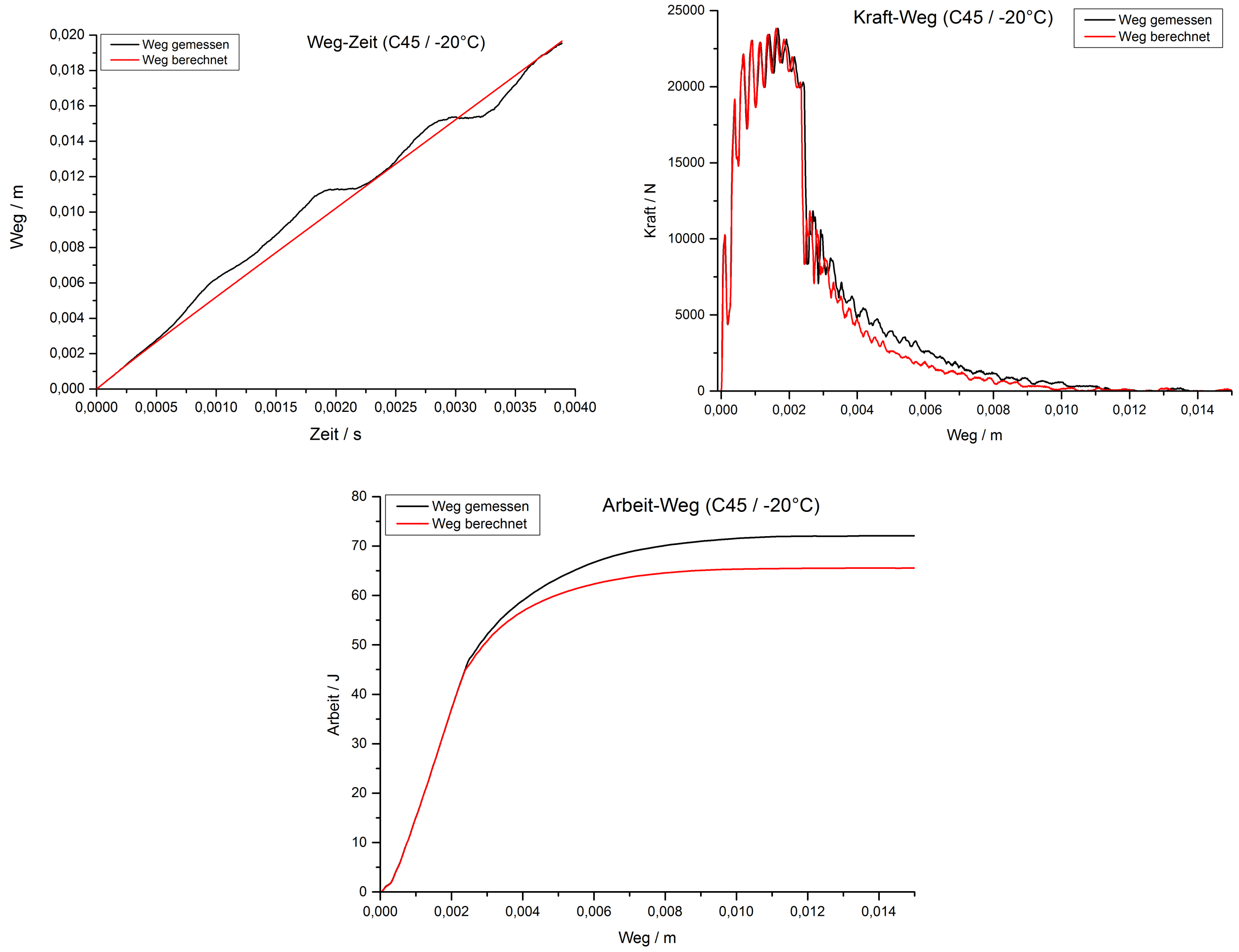


Figure 3.21.: Comparison of the impact test curves of measured (black) and calculated (red) deflection. a) time-deflection curve; b) load-deflection curve; c) impact energy-deflection curve

As it is obvious in the deflection-time curve (figure 3.21a), there are some significant differences of the measured deflection (black) and the almost linear calculated deflection (red). How the digital impact energy is influenced by this issue, is shown in figure 3.21b-c. At the first glance there is no significant difference visible in the load-deflection curve. But in the impact energy-deflection curve, a big mismatch can be noticed.

In the present example the total energy changes from 61 J to 72 J which is a difference of about 20 %.

4. Summary, conclusion and outlook

4.1. Dislocation density

A common method to evaluate the dislocation density in materials is the usage of the diffraction behaviour of X-rays. In this thesis one of the main objectives was to investigate, whether this method can be used on a the creep resistant chromium steel CB2. The results are summarized as follows:

1. With the first diffraction measurements on the base material at the Institute of Solid State Physics, it was concluded that this basic diffraction method gains too little intensity.

2. To conduct improved measurements, a cooperation with the Faculty of Physics at the University of Vienna was established.

3. Samples were prepared for beamline measurements and were investigated at the synchrotron in Grenoble.

4. The results of the latter measurements and their evaluation were carried out by Prof. Schafler and his team at the University of Vienna.

Conclusion and outlook

With the diffraction behaviour of high energy X-ray radiation, it is possible to make a quantitative statement of the dislocation density in the microstructure of materials and even of weldments. But it has to be kept in mind, that there are limitations to this method: The measurement averages its value over the investigated area and should only be applied on a homogeneous structure. In any case, a close observation of the measurement and the corresponding area is required.

The cooperation with the University of Vienna is still established and further investigations can be carried out.

4.2. Impact toughness

The second main objective was to develop an evaluation program for the instrumented Charpy V-notch impact test. The impact test is generally used to gain impact energy values of materials at different temperatures. To get additional information about materials behaviour during fracture, the test was instrumented with an electronic measurement system. The time consuming evaluation of the big amount of data was a big problem for the investigators and this program will be the solution of this issue.
The evaluation steps and results are summarized as follows:

1. Creation of a flow chart for the data evaluation.
2. The development of an evaluation program in MATLAB with the following features:
   - Evaluation and calculation of the impact energy.
   - Implementation of EN ISO 14556 and extended output data.
   - Full test report with visualisations and characteristic values.
3. An impact test series of a C45-steel showed a big variance in impact energies. This was caused by the temperature range of the specimens which was right below the transition temperature of the material.
4. An investigation of the results of the impact energy showed a systematic deviation between analogue and digital values.

Conclusion and outlook

The computer program enables a more exhaustive evaluation of the instrumented impact test. The programme was applied to evaluate the impact tests of a C45 test series, which were found to be indifferent at the measured temperature.
The programme also showed the limited accuracy of the measurement system.

The correct adjustment and dynamiccalibration of the deflection measurement system is the first step required, before further investigations will be done.
A high speed camera setup for visualising the crack propagation is planned and manufactured in the course of a Bachelor’s thesis.
Analysing the frequency of the oscillations would be an interesting addition to the evaluation programme.

5. Bibliography

Printed sources

[1] M Staubli, R. Hanus, T. Weber, K-H. Mayer, and T-U. Kern. “The European efforts in development of new high temperature casting materials”. In: *Proceedings of the 8th Liège Conference*. Vol. 53. 855-870. European Commission. Université de Liège, 2006.

[2] M. Schuler, S. Baumgartner, R. Schnitzer, and N. Enzinger. “Creep investigation and simulation of CB2 joints using similar rutile CB2 flux-cored wire”. In: *Welding in the world* (Oct. 2013).

[3] S. Hunklinger. *Festkörperphysik*. Oldenbourg Verlag, 2011.

[6] C. Kittel. *Einführung in die Festkörperphysik*. Oldenburg, 2013.

[7] G. Gottstein. *Materialwissenschaft und Werkstofftechnik: Physikalische Grundlagen*. Springer-Lehrbuch. Springer Berlin Heidelberg, 2014.

[9] C. Poletti. “Platizität, Ur- und Umformprozesse - Skriptum”. In: Graz University of Technology. 2012.

[10] D. Egger. “Röntgenbeugung an schmelzgesponnenen nanokristallinen Fe-basierten Legierungen”. MA thesis. Graz University of Technology, Oct. 2010.

[19] R. Schuster. “Charakterisierung von nanokristallinen Metallen durch Röntgen - Linienprofilanalyse”. MA thesis. Universität Wien, 2011.

[21] H. P. Klug and L. E. Alexander. *X-Ray diffraction procedures: for polycrystalline and amorphous materials*. 2nd ed. 1974.

[22] C. E. Krill, R Haberkorn, and Birrunger R. *Handbook of Nanostructured Materials and Nanotechnologie*. Ed. by Hari Sing Nalwa. Vol. 2. Academic Press San Diego, 2000.

[23] F. H. Chung and D. K. Smith. *Industrail applications of X-Ray Diffraction*. Ed. by T. Ungár. pp. 847-867. Marcel Denker, Inc., 1999.

[24] T. Ungár and A. Borbály. “The effect of dislocation contrast on X-Ray line broadening: A new approach to line profile analysis”. In: *Applied Physics Letters* 69.21 (1996), pp. 3173–3175.

[25] T. Ungár, C. Revesz, and A. Borbály. "Dislocations and Grain Size in Electrodeposited Nanocrystalline Ni Determined by the Modifed Williamson-Hall and Warren- Averbach Procedures". In: *Applied Crystallography* 31.4 (1998), pp. 554–558.

[29] E. Schafler, M. Zehetbauer, A. Borbály, and T. Ungár. "Dislocation densities and internal stresses in large strain cold worked pure iron". In: *Materials Science & Engineering* A234-236 (1997), pp. 445–448.

[30] A. Saric. "Charakterisierung der Wärmeeinflusszone von Mehrlagenschweißungen". MA thesis. Graz University of Technology, 2014.

[31] F. Mendez Martin. "Microstructural characterisation of the HAZ of new creep resistant 9% chromium steels using weld simulation technique". MA thesis. Graz University of Technology, 2006.

[33] W. Seidel and F. Hahn. *Werkstofftechnik*. Carl Hanser Verlag GmbH & Co. KG, 2012.

[36] T.L. Anderson. *Fracture Mechanics - Fundamentals and Apllications*. CRC Press, 1991.

[38] *Metallische Werkstoffe - Kerbschlagbiegeversuch nach Charpy - EN ISO 148-1*. Europäisches Komitee für Normung, Dec. 2010.

[40] *Stahl - Kerbschlagbiegeversuch nach Charpy(V-Kerb) - Instrumentiertes Prüfverfahren - EN ISO 14556*. Europäisches Komitee für Normung, Dec. 2007.

[41] K. Döttinger. "Aufnahme von Kraft/Weg-Diagrammen im Kerbschlagbiegeversuch". In: *Messtechnische Briefe 2* (1969).

[42] T. Nakamura, C.F. Shih, and L.B. Freund. "Analysis of a dynamically loaded three-point-bend ductile fracture specimen". In: *Engeneering Fracture Mechanics* 25 (1986), pp. 323–339.

[43] T. Nakamura, C. Shih, and L.B. Freund. "Three-dimensional transient analysis of dynamcally loaded three-point-bend ductile fracture specimen". In: *American Siciety of Testing and Material, Philadelphia* 1 (1989), pp. 217–241.

[44] R. W. K. Honeycombe and H. K. D. H. Bhadeshia. *Steels - microstructure and properties*. 2nd ed. Edward Arnold, 1995.

[45] S. Winkler. "Dynamische Kalibrierung von Pendelschlagwerken". In: *T 2/88* (Jan. 1988). Frauenhofer-Institut für Werkstofftechnik.

Online sources

[4] Geocities. *Crystal structures*. Oct. 2014. URL: `http://www.geocities.jp/ohba_lab_ob_page/structure7.htm`.

[5] K. P. Shah. *Mechanical Maintenance*. Nov. 2014. URL: http://practicalmaintenance.net/?p=1085.

[8] Cambridge University. *MATTER glssary*. Feb. 2015. URL: http://www.matter.org.uk/glossary/detail.asp?dbid=547.

[11] Robert Gordon University. *X-ray analysis*. Feb. 2015. URL: http://www2.rgu.ac.uk/life_semweb/xray.html.

[12] ETH-Zürich. *Characteristic X-ray radiation*. Feb. 2015. URL: http://people.ee.ethz.ch/~cattin/MIA-ETH/01-Modalities-I-media/figs/Characteristic_Spectrum.png.

[13] F.G. Banica. *The X-ray tube*. Mar. 2015. URL: http://www.nt.ntnu.no/users/floban/KJ%20%203055/X%20%20Ray/x-ray%20tube/x-ray%20tube.htm.

[14] Wikipedia. *X-ray tube*. Dec. 2014. URL: http://en.wikipedia.org/wiki/X-ray_tube.

[15] DESY. *Undulator*. Jan. 2015. URL: http://photon-science.desy.de/research/studentsteaching/primers/synchrotron_radiation/index_eng.html.

[16] Wikipedia. *Synchrotron SOLEIL*. Dec. 2014. URL: http://en.wikipedia.org/wiki/SOLEIL.

[17] ETH-Zürich. *Bragg's law of diffraction*. Sept. 2014. URL: http://www.microscopy.ethz.ch/bragg.htm.

[18] IPCI-Project. *Crystal origentations*. Jan. 2015. URL: http://81.161.252.57/ipci/courses/technology/index_59.htm.

[20] Universty of London. *The concept of a peak shape*. Jan. 2015. URL: http://pd.chem.ucl.ac.uk/pdnn/peaks/peakcon.htm.

[26] KS Analytical Systems. *XRD-setup*. Feb. 2015. URL: http://www.ksanalytical.com/images/I-XRD-Diagram.gif.

[27] ESRF. *Beam-line*. Jan. 2015. URL: http://www.esrf.eu/files/live/sites/www/files/UsersAndScience/Experiments/Imaging/ID19/Techniques/Diffraction/image48_transp.gif.

[28] Oocities. *Crosssection weldment*. Jan. 2015. URL: http://www.oocities.org/ferritec_eng/_borders/fig20.jpg.

[32] NDT Resource Center. *Materials and Processes Cource*. Mar. 2015. URL: https://www.nde-ed.org/EducationResources/CommunityCollege/Materials/Mechanical/Toughness.htm.

[34] University of Camebridge. *Ductile-to-brittle transition.* Feb. 2015. URL: `http://www.doitpoms.ac.uk/tlplib/BD6/ductile-to-brittle.php`.

[35] Fachhochschule Köln. *Bruchmechanismen.* Mar. 2015. URL: `http://www.gm.fh-koeln.de/~werkst/Versuche/versuchkerbschlag-w.html`.

[37] Inc. Engineering 2 Production. *Mechanical Testing -Charpy Impact Test.* Nov. 2014. URL: `http://e2pro.us/home/mechtest.html`.

[39] DGZfP. *Magnetooptische Sensoren.* Jan. 2015. URL: `http://www.ndt.net/article/dgzfp02/papers/v24/v24.htm`.

6. List of figures

7. List of tables

A. Appendix

A.1. Fracture mechanics - crack propagation

In the fracture mechanics, several different fracture cases can be distinguished. The two fundamental theories of elastic and inelastic fracture are combined to extended behaviours. Depending on the fracture case, failure can be handled with linear-elastic or elastic-plastic fracture mechanics [33].

There are two different types of crack propagation, the stable (figure A.1a) and the unstable one (figure A.1b).

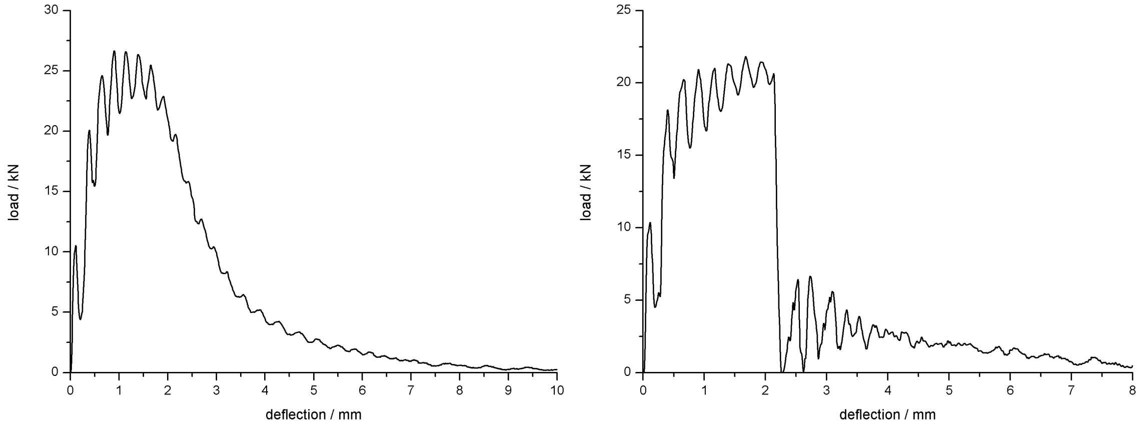


Figure A.1.: a) Stable crack propagation b) Combination of stable and unstable crack propagation

In figure A.1a) the case stable crack propagation and in figure A.1b) a failure after an unstable crack propagation are shown.

A.2. Evaluation program

A.2.1. Main program

```
halt = 0;
while halt ~= 1
% Vor Beginn der Auswertung mit diesem Programm, müssen die Messungen in
% Perception selber noch in ein ASCII-File konvertiert werden.
clear all
halt = 0;
close all % alle geöffneten Fenster von Matlab schließen

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Messdatenfile auswählen und Messdaten in eingen Matrix speichern:

% User soll Dateipfad des ASCII-Files angeben:
[Dateiname,Dateipfad] = uigetfile('*.txt','Messung auswählen');
% Messdatenfile auslesen
Datei = importdata(fullfile(Dateipfad,Dateiname));
% Messdaten in neue Matrix
Messdaten = Datei.data; % Messdaten in neue Matrix

% Messdaten --> 3 Spalten Matrix
% 1.Spalte: Zeit
% 2.Spalte: Kraft
% 3.Spalte: Weg

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% User soll mechanisch gemessenen Wert eingeben.
% Für dies eine UserInterface Routine verwendet:

% Zeilenbeschriftung:
prompt = {'Analogen Messwert eingeben und mit "OK" bestätigen:'};
% Fensterbeschriftung:
dlg_title = sprintf('%s',regexprep(Dateiname,'.txt',''));
% Eingabeaufforderung zum Zahlenwerteigeben, Rückgabe als "Cell":
Schlagarbeit_analog = inputdlg(prompt,dlg_title,[1,length(dlg_title)+35]);
% "Cell" konvertieren in "Double"-Wert:
Schlagarbeit_analog = sscanf(sprintf('%s*',Schlagarbeit_analog{:}),'%f*');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
% Die Rohdaten sind haben ein viel größeren Wertebereich als nötig.
% Deswegen wird im Kraft-Zeit-Diagramm der Anfangswert und der Endwert des
% benötigten Bereichs vom User definiert und anschließend die unötigen
% Daten gelöscht.

% Endpunkt festlegen:

% Figurehandle definieren
fig = figure;
% Eigenschaften des Fensters defineiren: Fenstergröße, keine Toolbar,
% keine Menueleiste, Titel festlegen
set(fig,'units','normalized','position',[0.1,0.1,0.8,0.8],...
    'Name','Kraft-Zeit-Diagramm','Numbertitle','off','Toolbar','none',...
    'Menubar','none')
% Achsengröße im Fester festlegen
axes('Units','Normalized','Position',[0.1,0.1,0.8,0.8])
% Button für Bestätigung einfügen
h = uicontrol('Style','togglebutton','String','und hier "bestätigen"',...
              'units','normalized','Position', [0.5,0.53,0.16,0.03],...
              'Fontsize',12);
% Daten ploten, allerdings nur die erste Hälfte der Datenpunkte
plot(Messdaten(1:length(Messdaten)*0.5,1),...
     Messdaten(1:length(Messdaten)*0.5,2))
xlim([min(Messdaten(:,1)) max(Messdaten(:,1)*0.5)])
% Anweisungen an User ins Fester schreiben
text('units','normalized','Position',[0.5,0.6],'String',...
    'Endpunkt mit der Maus setzen', 'Color','k','Fontsize',12);
% Cursormode aktivieren
Cursor = datacursormode(fig);
% Cursordaten auslesen
set(Cursor,'DisplayStyle','datatip','SnapToDataVertex','On','Enable','on');
% Warten bis User Punkt ausgewählt hat
waitfor(h,'value');
% Index des Cursors auslesen
Cursorinformation = getCursorInfo(Cursor);
% Endpunkt aus Cursordaten schreiben
Endpunkt = Cursorinformation.DataIndex;
%Fenster ......
delete(gca)
delete(h)


%Messdatenmatrix mit neuen Bereich (bis zum Endpunkt) überschreiben
Messdaten=Messdaten(1:Endpunkt,:);
```

```
% Angangspunkt festlegen:

set(fig,'units','normalized','position',[0.1,0.1,0.8,0.8],...
    'Name','Kraft-Zeit-Diagramm','Numbertitle','off','Toolbar','none',...
    'Menubar','none')
axes('Units','Normalized','Position',[0.1,0.1,0.8,0.8])
h = uicontrol('Style','togglebutton','String','und hier "bestätigen"',...
              'units','normalized','Position', [0.14,0.53,0.16,0.03],...
              'Fontsize',12);
plot(Messdaten(:,1),Messdaten(:,2))
xlim([min(Messdaten(:,1)) max(Messdaten(:,1))])
text('units','normalized','Position',[0.05,0.6],'String',...
    'Anfangspunkt mit der Maus setzen', 'Color','k','Fontsize',12);
Cursor = datacursormode(fig);
set(Cursor,'DisplayStyle','datatip','SnapToDataVertex','off','Enable','on')
waitfor(h,'value');
Cursorinformation = getCursorInfo(Cursor);
Anfangspunkt = Cursorinformation.DataIndex;
delete(gca)
delete(h)

% Messdatenmatrix mit neuen Bereich (vom Anfangspunkt bis zum Ende)
% überschreiben
Messdaten = Messdaten(Anfangspunkt:end,:);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Anfangspunktdaten als neuen Nullpunkt setzen(for-Schleife wäre eleganter,
% aber nicht notwendig. Für Kraftnullung ausreichend. Für den Weg wird
% nachher der Intagralstart als Nullpunkt gesetzt.

% Anfangswert von den Werten abziehen:
Messdaten(:,1) = Messdaten(:,1)- Messdaten(1,1);
Messdaten(:,2) = Messdaten(:,2)- Messdaten(1,2);
Messdaten(:,3) = (Messdaten(:,3)- Messdaten(1,3));

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Es kann sein, dass sich in den Daten Schleifen befinden und diese
% verursachen jedoch bei der nachfolgenden Punktinterpolation einen Fehler.
% Diese "Schleifen' müssen jetzt gefunden und eleminiert werden.

% Elemente suchen bei denen die Differenz zum vorigem Wert negativ ist
% und als Index ausgeben:
Schleifenindex=find(diff(Messdaten(:,3))<=0);
% Diese Zeilen mit diesen Werten aus der Matrix löschen, bis keine Schleife
% mehr gefunden wird:
while ~isempty(Schleifenindex) % so lange sich negative Elemente finden
    Messdaten(Schleifenindex,:)=[]; % Elemente löschen
    Schleifenindex = find(diff(Messdaten(:,3))<=0);% erneut suchen
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Messdaten mit äquidistanten Wegwerten interpolieren

% Wegvektor für Interpolation erzeugen:
s_spl = linspace(min(Messdaten(:,3)),max(Messdaten(:,3)),length(Messdaten));
% Erzeugen der zugehörigen Interpolationswerte:
f_spl = spline(Messdaten(:,3),Messdaten(:,2));
% Erzeugen der zugehörigen Kraftwerte
F_spl = ppval(f_spl,s_spl);

% Datenaufbereitung abgschlossen. Ab hier können die Daten ausgewertet
% werden.

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Für eine erste Berechnung der Schlagarbeit werden die Integralgrenzen vom
% User manuell bestimmt.

% Integralstart festlegen:

set(fig,'units','normalized','position',[0.1,0.1,0.8,0.8],...
    'Name','Kraft-Weg-Diagramm','Numbertitle','off','Toolbar','none',...
    'Menubar','none')
axes('Units','Normalized','Position',[0.1,0.1,0.8,0.8])
h = uicontrol('Style','togglebutton','String','und hier "bestätigen"',...
              'units','normalized','Position', [0.5,0.578,0.16,0.03],...
              'Fontsize',12);
plot(s_spl,F_spl)
xlim([min(s_spl) max(s_spl)])
text('units','normalized','Position',[0.5,0.6],'String',...
    {'Vordere Intergralgrenze setzen','','',...
    'oder nur bestätigen falls','diese schon passt.'} ,...
    'Color','k','Fontsize',12);
Cursor = datacursormode(fig);
```

```
set(Cursor,'DisplayStyle','datatip','SnapToDataVertex','off','Enable','on')
waitfor(h,'value');
Cursorinformation = getCursorInfo(Cursor);

% Falls der User den Messdatenbereich schon passend gesetzt hat, kann
% dieser den Schritt überspringen.

% Abfrage ob ein Cursor gesetzt wurde. Falls nicht wird als Integralstart
% einfach das erste Element verwendet:
if isempty(Cursorinformation) == 1
    Int_start = 1;
else
    Int_start = Cursorinformation.DataIndex;
end
delete(gca)
delete(h)

% Integralende festlegen:

set(fig,'units','normalized','position',[0.1,0.1,0.8,0.8],...
    'Name','Kraft-Weg-Diagramm','Numbertitle','off','Toolbar','none',...
    'Menubar','none')
axes('Units','Normalized','Position',[0.1,0.1,0.8,0.8])
h = uicontrol('Style','togglebutton','String','und hier "bestätigen"',...
              'units','normalized','Position', [0.5,0.578,0.16,0.03],...
              'Fontsize',12);
plot(s_spl,F_spl)
xlim([min(s_spl) max(s_spl)])
text('units','normalized','Position',[0.5,0.6],'String',...
    {'Hintere Intergralgrenze setzen','','',...
    'oder nur bestätigen falls','diese schon passt.'} ,...
    'Color','k','Fontsize',12);
Cursor = datacursormode(fig);
set(Cursor,'DisplayStyle','datatip','SnapToDataVertex','off','Enable','on')
waitfor(h,'value');
Cursorinformation = getCursorInfo(Cursor);
% Falls kein Cursor gesetz wurde, wird das letzte Element als hintere
% Integralgrenze verwendet
if isempty(Cursorinformation) == 1
    Int_ende = length(s_spl);
else
    Int_ende = Cursorinformation.DataIndex;
end
close
```

```
% Der manuell gesetze Intergralstart soll nun als Nullpunkt für den Weg
% verwendet werden. Dazu wird einfach der Zahlenwert beim Start von allen
% Elemnten subtrahiert.
s_spl = s_spl-s_spl(Int_start);

% Mit den gesetzten Grenzen kann jetzt die verbrauchte Arbeit berechnet
% werden durch Integration der Kraft-Weg-Kurve.

% Integrieren der interpolierten Messung mit den ausgewählten
% Integralgrenzen: Integration mit cumtrapz ausreichend, da genügend
% Datenpunkte vorhanden sind)

% Integralfunktion der Arbeit vom Integralstart bin zum Messdatenende:
Arbeit = cumtrapz(s_spl(Int_start:end),F_spl(Int_start:end));

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Dem User soll jetzt gezeigt werden wie die Messung aufbereitet und
% weiterverarbeitet wurde, sodass der dann weiter Entscheidungen treffen
% kann was mit der Messung weiter gschieht:

% Für die Darstellung der fiktiven Integralgrenze des analogen Messertes
% wird der Index bei dem der Funktionswert der Arbeit dem analogen Wert
% am besten entspricht bestimmt:

% Abfragen ob der analoge Wert überhaupt im Wertebereich liegt. Wenn ja
% wird Index mit find-Routine bestimmt und falls nicht wird der maximale
% Index der Arbeitsfunktion verwendet, als Zeichen dass die analoge Wert
% viel größer ist.

if Schlagarbeit_analog <= max(Arbeit)
    Schlagarbeit_analog_ind = find(Arbeit>=Schlagarbeit_analog, 1 );
else
    Schlagarbeit_analog_ind = length(Arbeit)-1;
end

% Weiters muss die digitale Schlagrbeit, die vom User durch setzen der
% Integralgrenzen bestimmt wird, berechnet werden:

Schlagarbeit_digital = round(Arbeit(Int_ende-Int_start),1);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
% Der User soll jetzt entscheiden um welchen Fall laut Norm es sich
% handelt. Dazu soll das Programm "Fallunterscheidung" aufgerufen werden:
Fallunterscheidung; % Rückgabewert "Fall" wird in den Workspace geschrieben

% Für die nachfolgenden Berechnungen werden verschiedenste Werte benötigt.
% Diese werden entweder, wenn sie allgemein gültig sind gleich im
% Hauptprogramm berechnet oder sonst in den Unterprogrammen:

% charakteristische Werte: für alle Fälle

% maximale Kraft mit zugehöroge Werten und Index der Messung
[F_max,Index_max] = max(F_spl);
s_max = round(s_spl(Index_max)*10^3,2);

s_t_manuell = round(s_spl(Int_ende)*10^3,2);
% Die restlichen Werte werden für jeden Fall nachher einzeln berechnet

% Jetzt soll für den jeweiligen Fall die Messung nach Norm gefittet und
% ausgewertet werden. Dies geschieht in den jeweiligen Unterprogrammen.
switch Fall
    case 'A'
        Fall_A
    case 'B'
        Fall_B
    case 'C'
        Fall_C
    case 'D'
        Fall_D
    case 'E'
        Fall_E
    case 'F'
        Fall_F
end

% Falls manche Werte noch nicht vorhanden sind und aber für das
% Messprotokoll zumindest eine Variable vorhanden sein muss, werden diese
% in einem Unterprogramm erstellt:
Leerdaten;

% Alle gesammelten Daten werden in eine Excel-Datei exportiert
Dateiexport;

% Aus all den gesammelten Werten wird nun ein übersichtliches Messprotokoll
% erstellt:
```

```
Messprotokoll;

end
```

A.2.2. Measuring report

```
function varargout = Messprotokoll(varargin)
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @Messprotokoll_OpeningFcn, ...
                   'gui_OutputFcn',  @Messprotokoll_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end
if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end

function Messprotokoll_OpeningFcn(hObject, eventdata, handles, varargin)
handles.output = hObject;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% ab hier beginnt das Programm vom User

% Zuerst werden für die Function alle Variablen aus dem Worspace benötigt.
% Dazu werden alle Variablen nacheinander in diesen Workspace geschrieben:

% Funktionen:
s_spl = evalin('base','s_spl');
s_spl = s_spl/10^-3;
F_spl = evalin('base','F_spl');
F_spl = F_spl/10^3;
s_norm = evalin('base','s_norm');
s_norm = s_norm/10^-3;
Fit_norm = evalin('base','Fit_norm');
Fit_norm = Fit_norm/10^3;
Arbeit = evalin('base','Arbeit');
Arbeit_norm = evalin('base','Arbeit_norm');

% Indices:
```

```
Int_start = evalin('base','Int_start');
Int_ende = evalin('base','Int_ende');
Schlagarbeit_analog_ind = evalin('base','Schlagarbeit_analog_ind');
F_max_ind = evalin('base','F_max_ind');
Fiu_norm_ind = evalin('base','Fiu_norm_ind');
Int_ende_norm = evalin('base','Int_ende_norm');

% Kräfte:
F_max = evalin('base','F_max');
F_max_norm = evalin('base','F_max_norm');
F_max_kal = evalin('base','F_max_kal');
F_gy = evalin('base','F_gy');
F_gy_kal = evalin('base','F_gy_kal');
F_a = evalin('base','F_a');
F_a_kal = evalin('base','F_a_kal');
F_iu = evalin('base','F_iu');
F_iu_kal = evalin('base','F_iu_kal');

% Wege:
s_max = evalin('base','s_max');
s_max_norm = evalin('base','s_max_norm');
s_gy = evalin('base','s_gy');
s_a = evalin('base','s_a');
s_iu = evalin('base','s_iu');
s_t = evalin('base','s_t');

% Arbeiten:
Schlagarbeit_analog = evalin('base','Schlagarbeit_analog');
Schlagarbeit_digital = evalin('base','Schlagarbeit_digital');
Schlagarbeit_norm = evalin('base','Schlagarbeit_norm');
Schlagarbeit_kal = evalin('base','Schlagarbeit_kal');
W_max_norm = evalin('base','W_max_norm');
W_max_kal = evalin('base','W_max_kal');
W_a_norm = evalin('base','W_a_norm');
W_a_kal = evalin('base','W_a_kal');
W_iu_norm = evalin('base','W_iu_norm');
W_iu_kal = evalin('base','W_iu_kal');

%besondere Werte:
Dateipfad = evalin('base','Dateipfad');
Dateiname = evalin('base','Dateiname');
Messung = sprintf('Messung: %s', regexprep(Dateiname,'.txt',''));
Fall = evalin('base','Fall');
AnteilVBF = evalin('base','AnteilVBF');
```

```
Steigung = evalin('base','Steigung');
elastisch = evalin('base','elastisch');
plastisch = evalin('base','plastisch');
sigma_gy = evalin('base','sigma_gy');

% Zur korrekten Zuweisung des Messprotikoll, wird der Messname vermerkt
set(handles.text2,'String', Messung); % Messung ausgeben

% Dann wird die Messung, die gefittete nach Norm und die Grenzen
% geplottet
axes(handles.axes1)% aktivieren des ersten axes-Object
l1 = line([s_spl(Int_start) s_spl(Int_start)],[min(F_spl) max(F_spl)*1.1],...
    'Color','k','Linestyle',':','LineWidth',2.0);
l2 = line([s_spl(Int_ende) s_spl(Int_ende)],[min(F_spl) max(F_spl)*1.1],...
    'Color','r','Linestyle',':','LineWidth',2.0);
l3 = line([s_norm(Int_ende_norm) s_norm(Int_ende_norm)],...
    [min(F_spl) max(F_spl)*1.1],'Color','m','Linestyle','-.',...
    'LineWidth',1.5);
if ~isempty(Schlagarbeit_analog_ind)
l4 = line([s_spl(Schlagarbeit_analog_ind+Int_start) ...
    s_spl(Schlagarbeit_analog_ind+Int_start)],[min(F_spl) max(F_spl)*1.1],...
    'Color','g','Linestyle','--','LineWidth',1.0);
end

switch Fall
    case 'A'
        hold on
        p1 = plot(s_spl,F_spl,'-b');
        p2 = plot(s_norm,Fit_norm,'-m');
        p3 = plot(s_max,F_max,'*k','Markersize',5,'LineWidth',1.);
        hold off
        leg = legend([p1 p2 l1 l2 l3 l4 p3 ],{'Messung','Norm',...
        'Start','Grenze User','Grenze Norm','Grenze analog','F_{max}'}...
        ,'Location','Eastoutside');
    case 'B'
        hold on
        p1 = plot(s_spl,F_spl,'-b');
        p2 = plot(s_norm(1:Int_ende_norm),Fit_norm(1:Int_ende_norm),'-m');
        p3 = plot(s_max,F_max,'*k','Markersize',5,'LineWidth',1.2);
        p4 = plot(s_a,F_a,'ok','Markersize',5,'LineWidth',1.2);
        hold off
        leg = legend([p1 p2 l1 l2 l3 l4 p3 p4],{'Messung','Norm',...
        'Start','Grenze User','Grenze Norm','Grenze analog','F_{max}',...
        'F_{a}'},'Location','Eastoutside');
    case 'C'
        hold on
        p1 = plot(s_spl,F_spl,'-b');
        p2 = plot(s_norm,Fit_norm,'-m');
        p3 = plot(s_max,F_max,'*k','Markersize',5,'LineWidth',1.2);
        p4 = plot(s_max_norm,F_max_norm,'^k','Markersize',5,'LineWidth',1.2);
        p5 = plot(s_gy,F_gy,'sk','Markersize',5,'LineWidth',1.2);
        hold off
        leg = legend([p1 p2 l1 l2 l3 l4 p3 p4 p5],{'Messung','Norm',...
        'Start','Grenze User','Grenze Norm','Grenze analog','F_{max}',...
        'F_{max norm}','F_{gy}'},'Location','Eastoutside');
    case 'D'
        hold on
        p1 = plot(s_spl,F_spl,'-b');
        p2 = plot(s_norm(1:Int_ende_norm),Fit_norm(1:Int_ende_norm),'-m');
        p3 = plot(s_max,F_max,'*k','Markersize',5,'LineWidth',1.2);
        p4 = plot(s_max_norm,F_max_norm,'^k','Markersize',5,'LineWidth',1.2);
        p5 = plot(s_gy,F_gy,'sk','Markersize',5,'LineWidth',1.2);
        p6 = plot(s_a,F_a,'ok','Markersize',5,'LineWidth',1.2);
        hold off
        leg = legend([p1 p2 l1 l2 l3 l4 p3 p4 p5 p6],{'Messung','Norm',...
        'Start','Grenze User','Grenze Norm','Grenze analog','F_{max}',...
        'F_{max norm}','F_{gy}','F_{a}'},'Location','Eastoutside');

    case 'E'
        hold on
        plot(s_spl,F_spl,'-b',s_norm,Fit_norm,'-m');
        p1 = plot(s_spl,F_spl,'-b');
        p2 = plot(s_norm(1:Int_ende_norm),Fit_norm(1:Int_ende_norm),'-m');
        p3 = plot(s_max,F_max,'*k','Markersize',5,'LineWidth',1.2);
        p4 = plot(s_max_norm,F_max_norm,'^k','Markersize',5,'LineWidth',1.2);
        p5 = plot(s_gy,F_gy,'sk','Markersize',5,'LineWidth',1.2);
        p6 = plot(s_a,F_a,'ok','Markersize',5,'LineWidth',1.2);
        p7 = plot(s_iu,F_iu,'.k','Markersize',15,'LineWidth',3);
        hold off
        leg = legend([p1 p2 l1 l2 l3 l4 p3 p4 p5 p6 p7],{'Messung','Norm',...
        'Start','Grenze User','Grenze Norm','Grenze analog','F_{max}',...
        'F_{max norm}','F_{gy}','F_{a}','F_{iu}'},'Location','Eastoutside');
    case 'F'
        hold on
        plot(s_spl,F_spl,'-b',s_norm,Fit_norm,'-m');
        p1 = plot(s_spl,F_spl,'-b');
        p2 = plot(s_norm(1:Int_ende_norm),Fit_norm(1:Int_ende_norm),'-m');
        p3 = plot(s_max,F_max,'*k','Markersize',5,'LineWidth',1.2);
        p4 = plot(s_max_norm,F_max_norm,'^k','Markersize',5,'LineWidth',1.2);
        p5 = plot(s_gy,F_gy,'sk','Markersize',5,'LineWidth',1.2);
        hold off
        leg = legend([p1 p2 l1 l2 l3 l4 p3 p4 p5],{'Messung','Norm',...
        'Start','Grenze User','Grenze Norm','Grenze analog','F_{max}',...
        'F_{max norm}','F_{gy}'},'Location','Eastoutside');
end

box on
xlabel('s / mm'); ylabel('F / kN'); title('\fontsize{12}Messung')

if s_spl(Int_ende) >= s_spl(Schlagarbeit_analog_ind+Int_start) && ...
         s_spl(Int_ende) >= s_norm(Int_ende_norm)
     xmax = s_spl(Int_ende)*1.2;
     ymax = max(Arbeit)*1.1;
elseif s_spl(Schlagarbeit_analog_ind+Int_start) >= s_spl(Int_ende) && ...
        s_spl(Schlagarbeit_analog_ind+Int_start) >= s_norm(Int_ende_norm)
    if s_spl(Schlagarbeit_analog_ind+Int_start) >= s_spl(Int_ende)*1.3
        xmax = s_spl(Int_ende)*1.1;
    else
        xmax = s_spl(Schlagarbeit_analog_ind+Int_start)*1.1;
    end
    ymax = max(Arbeit)*1.2;
elseif s_norm(Int_ende_norm) >= s_spl(Schlagarbeit_analog_ind+Int_start)...
        && s_norm(Int_ende_norm) >= s_spl(Int_ende)
    if s_norm(Int_ende_norm) >= s_spl(Int_ende)*1.5
        xmax = s_spl(Int_ende)*1.01;
    else
        xmax = s_norm(Int_ende_norm);
    end
    ymax = Arbeit(Int_ende_norm)*1.2;
else
    xmax = s_spl(Int_ende)*1.5;
end

axis([-0.05 xmax min(F_spl) max(F_spl)*1.1])
set(leg,'Fontsize',8)

axes(handles.axes2)% aktivieren des zweiten axes-Object

plot(s_spl(Int_start:end),Arbeit,'-b',s_norm(1:Int_ende_norm),...
    Arbeit_norm(1:Int_ende_norm),'-m')
line([0 max(s_spl)], [Arbeit(Int_ende-Int_start) ...
    Arbeit(Int_ende-Int_start)],'Color','r','Linestyle',':',...
    'LineWidth',2.0)
line([0 max(s_spl)], [Schlagarbeit_analog Schlagarbeit_analog],...
    'Color','g','Linestyle','--','LineWidth',1.0)
line([0 max(s_spl)], [Schlagarbeit_norm Schlagarbeit_norm],'Color','m',...
    'Linestyle','-.','LineWidth',1.0)
title('Integralfunktion','Fontsize',12,'Fontweight','normal'); ...
    xlabel('s / mm'); ylabel('W / J')
axis([0 xmax 0 ymax])

axes(handles.axes3) % aktivieren des dritten axes-Object
hold on
switch Fall
    case {'A', 'B'}
         patch([0 s_max_norm s_max_norm],[0 F_max_norm 0],'g',...
             'FaceAlpha',0.5);
    case 'C'
         patch([0 s_gy s_gy],[0 F_gy 0],'g','FaceAlpha',0.5);
         patch([s_gy s_norm(2:Fiu_norm_ind) s_norm(Fiu_norm_ind)],...
             [0 Fit_norm(2:Fiu_norm_ind) 0],[1 0.5 0],'FaceAlpha',0.5);
         elastischerW = sprintf('%s',['elastischer Anteil = ' ...
             num2str(elastisch,'%2.0f') ' %']);
         text('units','normalized','Position',[0.5,0.75],'String',...
             elastischerW,'Color','g');
         plastischerW = sprintf('%s',['plastischer Anteil = ' ...
             num2str(plastisch,'%2.0f') ' %']);
         text('units','normalized','Position',[0.5,0.6],'String',...
             plastischerW,'Color',[1 0.5 0]);
    case 'D'
         patch([0 s_gy s_gy],[0 F_gy 0],'g','FaceAlpha',0.5);
         patch([s_gy s_norm(2:Fiu_norm_ind) s_norm(Fiu_norm_ind)],...
             [0 Fit_norm(2:Fiu_norm_ind) 0],[1 0.5 0],'FaceAlpha',0.5);
         elastischerW = sprintf('%s',['elastischer Anteil = ' ...
             num2str(elastisch,'%2.0f') ' %']);
         text('units','normalized','Position',[0.5,0.75],'String',...
             elastischerW,'Color','g');
         plastischerW = sprintf('%s',['plastischer Anteil = ' ...
             num2str(plastisch,'%2.0f') ' %']);
         text('units','normalized','Position',[0.5,0.6],'String',...
             plastischerW,'Color',[1 0.5 0]);
    case 'E'
         elastischerW = sprintf('%s',['elastischer Anteil = ' ...
             num2str(elastisch,'%2.0f') ' %']);
         text('units','normalized','Position',[0.5,0.75],'String',...
             elastischerW,'Color','g');
         plastischerW = sprintf('%s',['plastischer Anteil = ' ...
             num2str(plastisch,'%2.0f') ' %']);
         text('units','normalized','Position',[0.5,0.6],'String',...
             plastischerW,'Color',[1 0.5 0]);
         patch([0 s_gy s_gy],[0 F_gy 0],'g','FaceAlpha',0.5);
         patch([s_gy s_norm(2:F_max_ind) s_max_norm],...
             [0 Fit_norm(2:F_max_ind) 0],[1 0.5 0],'FaceAlpha',0.5);
         patch([s_max_norm s_norm(F_max_ind:Fiu_norm_ind) ...
             s_norm(Fiu_norm_ind)],[0 Fit_norm(F_max_ind:Fiu_norm_ind) 0]...
             ,[1 0.5 0],'FaceAlpha',0.5);
    case 'F'
         elastischerW = sprintf('%s',['elastischer Anteil = ' ...
             num2str(elastisch,'%2.0f') ' %']);
         text('units','normalized','Position',[0.5,0.75],'String',...
             elastischerW,'Color','g');
         plastischerW = sprintf('%s',['plastischer Anteil = ' ...
             num2str(plastisch,'%2.0f') ' %']);
         text('units','normalized','Position',[0.5,0.6],'String',...
             plastischerW,'Color',[1 0.5 0]);
         patch([0 s_gy s_gy],[0 F_gy 0],'g','FaceAlpha',0.5);
         patch([s_gy s_norm(2:F_max_ind) s_max_norm],...
             [0 Fit_norm(2:F_max_ind) 0],[1 0.5 0],'FaceAlpha',0.5);
         patch([s_max_norm s_norm(F_max_ind:Int_ende_norm) ...
             s_norm(Int_ende_norm)],[0 Fit_norm(F_max_ind:Int_ende_norm)...
             0],[1 0.5 0],'FaceAlpha',0.5);
end
plot(s_spl,F_spl,'-b');
hold off
box on
xlabel('s / mm'); ylabel('F / kN')
title('\fontsize{12}Arbeits-Visualisierung')
axis([-0.05 xmax 0 max(F_spl)*1.1])

guidata(hObject, handles);

% charakteristische Werte in Textboxen schreiben
Kraefte = sprintf('%s\n',...
    ['      F_max_norm = ' num2str(F_max_norm,'%5.2f') ' kN'],...
    ['F_max_gemessen = ' num2str(F_max,'%5.2f') ' kN'],...
    ['              F_gy = ' num2str(F_gy,'%5.2f') ' kN'],...
    ['               F_iu = ' num2str(F_iu,'%5.2f') ' kN'],...
    ['                F_a = ' num2str(F_a,'%5.2f') ' kN']);
set(handles.text3, 'String', Kraefte);
```

```
Wege = sprintf('%s\n',...
    ['   s_max_norm = ' num2str(s_max_norm,'%5.2f') ' mm'],...
    ['          s_max = ' num2str(s_max,'%5.2f') ' mm'],...
    ['            s_gy = ' num2str(s_gy,'%5.2f') ' mm'],...
    ['             s_iu = ' num2str(s_iu,'%5.2f') ' mm'],...
    ['              s_a = ' num2str(s_a,'%5.2f') ' mm  '],...
    ['              s_t = ' num2str(s_t,'%5.2f') ' mm  ']);
  set(handles.text4, 'String', Wege);

Arbeiten = sprintf('%s\n',...
    ['   W_analog = ' num2str(Schlagarbeit_analog,'%5.1f') ' J'],...
    ['      W_user = ' num2str(Schlagarbeit_digital,'%5.1f') ' J'],...
    ['     W_norm = ' num2str(Schlagarbeit_norm,'%5.1f') ' J'],...
    ['W_max_norm = ' num2str(W_max_norm,'%5.1f') ' J'],...
    ['  W_iu_norm = ' num2str(W_iu_norm,'%5.1f') ' J'],...
    ['    W_a_norm = ' num2str(W_a_norm,'%5.1f') ' J']);
set(handles.text5, 'String', Arbeiten);

besWerte = sprintf('%s\n',...
    ['              Fall ' Fall,'                          '],...
    ['     Ant. der VFBF = (' num2str(AnteilVBF,'%3.0f') ' ' ...
    char(177) ' 20) %'],...
    ['Steigung bis F_gy  = ' num2str(Steigung,'%3.1f') ' kN/mm'],...
    ['dyn. Streckgrenze = ' num2str(sigma_gy,'%4.0f') ' MPa']);%,...
set(handles.text7, 'String', besWerte);

% Exportieren des Messprotokolls in eine PDF-Datei mit selbigen Namen
print(gcf,'-dpdf','-opengl','-r300',regexprep(fullfile(Dateipfad,Dateiname),...
    '.txt',''))

function varargout = Messprotokoll_OutputFcn(hObject, eventdata, handles)
varargout{1} = handles.output;
uiwait(gcf);

% Button zum Drucken erzeugen
function pushbutton1_Callback(hObject, eventdata, handles)
print(gcf,'-opengl','dwinc','-v','-r300');

% Button für erneute Auswertung
function pushbutton2_Callback(hObject, eventdata, handles)
halt = 1;
assignin('base','halt',halt)
uiresume(gcbf)
close all
```

```
%Button zum Schließen des Programms
function pushbutton3_Callback(hObject, eventdata, handles)
uiresume(gcbf)
close all
```

A.2.3. Intersection evaluation

```
% Zur Schnittpunktsberechnung werden vier Punkte an die Function übergeben.
% Gelöst wird das Problem mittels lösen eines linearen Gleichungssystem in
% Matrixform

function S = Schnittpunkt(A,B,C,D)

% Erstellen der Koeffizientenmatrix
M = [B(1)-A(1) D(1)-C(1);B(2)-A(2) D(2)-C(2)];
% Erstellen des Spaltenvektors
b = [C(1)-A(1);C(2)-A(2)];

% Erhalten des Lösungsvektors durch lösen des Glg.-systems
x = M\b;

% Berechnung des Schittpunkts
S = [A(1);A(2)]+x(1).*[B(1)-A(1);B(2)-A(2)];
```

A.2.4. File-export

```
% Zum Export der Daten werden in eine Matrix alle wichtigen Vektoren und
% Variablen geschrieben.


% Erzeugung einer leeren Matrix(cell)
Export = cell(length(s_spl)+2,13,1);

% Anchließend werden die jeweiligen Elemente an die gewünschte Postion
% geschrieben
Export(1,1:4) = {'s_spl/mm','F_spl/N','s_norm/mm','F_norm/N'};
Export(1,6:13)={'Kräfte:','','Wege:','','Arbeiten:','','bes. Werte:',''};

Export(3:end,1:2) = [num2cell(s_spl',2), num2cell(F_spl',2)];
Export(3:length(s_norm)+2,3:4)=[num2cell(s_norm',2),num2cell(Fit_norm',2)];

Export(3:11,6:7) = [{'F_max_norm/kN',F_max_norm};...
                    {'F_max_kal/kN',F_max_kal};...
                    {'F_max_gemessen/kN',round(F_max,2)};...
```

```
                   {'F_gy/kN',F_gy};...
                   {'F_gy_kal/kN',F_gy_kal};...
                   {'F_iu/kN',F_iu};...
                   {'F_iu_kal/kN',F_iu_kal};...
                   {'F_a/kN',F_a};...
                   {'F_a_kal/kN',F_a_kal}];

Export(3:8,8:9) = [{'s_max_norm/mm',s_max_norm};...
                   {'s_max/mm',s_max};...
                   {'s_gy/mm',s_gy};...
                   {'s_iu/mm',s_iu};...
                   {'s_a/mm',s_a};...
                   {'s_t/mm',s_t}];

Export(3:12,10:11) = [{'W_analog/J',Schlagarbeit_analog};...
                     {'W_user/J',Schlagarbeit_digital};...
                     {'W_norm/J',Schlagarbeit_norm};...
                     {'W_kal/J',Schlagarbeit_norm};...
                     {'W_max_norm/J',W_max_norm};...
                     {'W_max_kal/J',W_max_kal};...
                     {'W_iu_norm/J',W_iu_norm};...
                     {'W_iu_kal/J',W_iu_kal};...
                     {'W_a_norm/J',W_a_norm};...
                     {'W_a_kal/J',W_a_kal}];

Export(3:7,12:13) = [{'Anteil der Verformbruchfläche/%',AnteilVBF};...
                     {'Steigung/kN*mm^1',Steigung};...
                     {'Kalibrierwert',round((CF-1)*100,1)};...
                     {'Sigma_gy/MPa',sigma_gy};...
                     {'Fall',Fall}];

% Die Excel-Datei soll den selben Namen des Messdatenfiles haben
Dateiname = regexprep(Dateiname,'.txt','');
Excelpfad = fullfile(Dateipfad,Dateiname);

% Export der Datenmatrix als Excel-File
xlswrite(Excelpfad,Export)
```

A.2.5. Different cases

Case A

```
% Wie und warum das Unterprogrogramm funktioniert ist im Fall E detailiert
% erklärt!

s_norm = [s_spl(Int_start), s_spl(Index_max), s_spl(Index_max),s_spl(end)];
Fit_norm = [F_spl(Int_start), F_max, 0, 0];

Arbeit_norm = cumtrapz(s_norm,Fit_norm);

Int_ende_norm = 3;

s_norm = s_norm(1:Int_ende_norm);
Fit_norm = Fit_norm(1:Int_ende_norm);

F_max_norm = round(F_max/10^3,2);
F_max_kal = round(F_max_norm*CF,2);

s_max_norm = round(s_spl(Index_max)*10^3,2);

Schlagarbeit_norm = round(max(Arbeit_norm),1);
Schlagarbeit_kal = round(Schlagarbeit_norm*CF,1);

Steigung = round(F_max_norm/s_max_norm,2);

B = 0.01; S = 0.04; L = 0.008;
sigma_gy = round(3*F_max_norm*10^3*S/(4*B*L^2)*10^-6,0);
```

Case B

```
% Wie und warum das Unterprogrogramm funktioniert ist im Fall E detailiert
% erklärt!

fig = figure;
set(fig,'units','normalized','position',[0.1,0.1,0.8,0.8],...
    'Name','Kraft-Weg-Diagramm','Numbertitle','off','Toolbar','none',...
    'Menubar','none')
axes('Units','Normalized','Position',[0.1,0.1,0.8,0.8])
h = uicontrol('Style','togglebutton','String','und hier "bestätigen"',...
              'units','normalized','Position', [0.5,0.578,0.16,0.03],...
              'Fontsize',12);
plot(s_spl,F_spl)
xlim([min(s_spl) max(s_spl)])
```

```
text('units','normalized','Position',[0.5,0.66],'String',...
    'Flanke des Steilabfalls markieren', 'Color','k','Fontsize',12);
Cursor = datacursormode(fig);
set(Cursor,'DisplayStyle','datatip','SnapToDataVertex','off','Enable','on')
waitfor(h,'value');
Cursorinformation = getCursorInfo(Cursor);
Steilabfall = Cursorinformation.DataIndex;
close

ind = find(diff(F_spl)>0);
ind = ind-Steilabfall;
ind = ind(ind>0);
ind = ind+Steilabfall+1;
Fa_ind = ind(1);

for i=Fa_ind:length(F_spl)
    if F_spl(i) <= 0.01*F_max && i >= Int_ende
        Fit_ende = i;
        break
    else
        Fit_ende = length(F_spl);
    end
end

fit_auffang = polyfit(s_spl(Fa_ind:Fit_ende),F_spl(Fa_ind:Fit_ende),2);
F_fit_auffang = polyval(fit_auffang,s_spl);
Differenz = F_spl-F_fit_auffang;

for i=Index_max:Fa_ind
    if Differenz(i) > 0 && Differenz(i+1) <0
        Index_naechste = [i i+1];
        A = [s_spl(Index_naechste(1)) F_spl(Index_naechste(1))];
        B = [s_spl(Index_naechste(2)) F_spl(Index_naechste(2))];
        C = [s_spl(Index_naechste(1)) F_fit_auffang(Index_naechste(1))];
        D = [s_spl(Index_naechste(2)) F_fit_auffang(Index_naechste(2))];
        break
    elseif i == Fa_ind
        clear Differenz
        fit_steil = polyfit(s_spl(Index_max:Fa_ind),...
                    F_spl(Index_max:Fa_ind),1);
        F_fit_steil = polyval(fit_steil,s_spl);
        Differenz = F_fit_steil-F_fit_auffang;
        for j=Index_max:length(s_spl)-1
            if Differenz(j) > 0 && Differenz(j+1) < 0
            Index_naechste = [j j+1];
            end
        end
        A = [s_spl(Index_naechste(1)) F_fit_steil(Index_naechste(1))];
        B = [s_spl(Index_naechste(2)) F_fit_steil(Index_naechste(2))];
        C = [s_spl(Index_naechste(1)) F_fit_auffang(Index_naechste(1))];
        D = [s_spl(Index_naechste(2)) F_fit_auffang(Index_naechste(2))];
        break
    end
end

S_1 = Schnittpunkt(A,B,C,D);

[F_max,Index_max] = max(F_spl);

s_norm = [s_spl(Int_start), s_spl(Index_max), S_1(1), ...
    s_spl(Index_naechste(2):end)];
Fit_norm = [F_spl(Int_start), F_max, S_1(2), ...
    F_fit_auffang(Index_naechste(2):end)];

[F_min,Index_min] = min(Fit_norm(3:end));

for i = 3:length(s_norm)
    if i == length(s_norm)
        Int_ende_norm = Index_min;
        break
    elseif Fit_norm(i) > 0.02*F_max && Fit_norm(i+1) <= 0.02*F_max
        Int_ende_norm = i+1;
        break
    end
end

s_norm = s_norm(1:Int_ende_norm);
Fit_norm = Fit_norm(1:Int_ende_norm);

Arbeit_norm = cumtrapz(s_norm,Fit_norm);

F_max_norm = round(F_max/10^3,2);
F_max_kal = round(F_max_norm*CF,2);
F_a = round(S_1(2)/10^3,2);
F_a_kal = round(F_a*CF,2);

s_max_norm = round(s_spl(Index_max)*10^3,2);
s_a = round(S_1(1)*10^3,2);
```

```
s_t = round(s_norm(Int_ende_norm)*10^3,2);

Schlagarbeit_norm = round(Arbeit_norm(Int_ende_norm),1);
Schlagarbeit_kal = round(Schlagarbeit_norm*CF,1);

Steigung = round(F_max_norm/s_max_norm,2);

B = 0.01; S = 0.04; L = 0.008;
sigma_gy = round(3*F_max_norm*10^3*S/(4*B*L^2)*10^-6,0);
```

Case C

```
% Wie und warum das Unterprogrogramm funktioniert ist im Fall E detailiert
% erklärt!

fig = figure;
set(fig,'units','normalized','position',[0.1,0.1,0.8,0.8],...
    'Name','Kraft-Weg-Diagramm','Numbertitle','off','Toolbar','none',...
    'Menubar','none')
axes('Units','Normalized','Position',[0.1,0.1,0.8,0.8])
h = uicontrol('Style','togglebutton','String','und hier "bestätigen"',...
              'units','normalized','Position', [0.5,0.578,0.16,0.03],...
              'Fontsize',12);
plot(s_spl,F_spl)
xlim([min(s_spl) max(s_spl)])
text('units','normalized','Position',[0.5,0.68],'String',...
    {'Flanke des Steilanstiegs der', '"zweiten" Schwingung markieren'},...
    'Color','k','Fontsize',12);
Cursor = datacursormode(fig);
set(Cursor,'DisplayStyle','datatip','SnapToDataVertex','off','Enable','on')
waitfor(h,'value');
Cursorinformation = getCursorInfo(Cursor);
steigende_Flanke = Cursorinformation.DataIndex;
delete(gca)
delete(h)

set(fig,'units','normalized','position',[0.1,0.1,0.8,0.8],...
    'Name','Kraft-Weg-Diagramm','Numbertitle','off','Toolbar','none',...
    'Menubar','none')
axes('Units','Normalized','Position',[0.1,0.1,0.8,0.8])
h = uicontrol('Style','togglebutton','String','und hier "bestätigen"',...
              'units','normalized','Position', [0.5,0.578,0.16,0.03],...
              'Fontsize',12);
plot(s_spl,F_spl)
xlim([min(s_spl) max(s_spl)])
```

```
text('units','normalized','Position',[0.5,0.66],'String',...
    'Flanke des Steilabfalls markieren', 'Color','k','Fontsize',12);
Cursor = datacursormode(fig);
set(Cursor,'DisplayStyle','datatip','SnapToDataVertex','off','Enable','on')
waitfor(h,'value');
Cursorinformation = getCursorInfo(Cursor);
Steilabfall = Cursorinformation.DataIndex;
close

ind = find(diff(F_spl)<0);
ind = ind-steigende_Flanke;
ind = ind(ind>0);
ind = ind+steigende_Flanke;
Fgy_ind = ind(1);

ind = find(diff(F_spl)<0);
ind = ind-steigende_Flanke;
ind = ind(ind<0);
ind = ind+steigende_Flanke;
Steil_min_ind = ind(end);

ind = find(diff(F_spl)>0);
ind = ind-Steilabfall;
ind = ind(ind<0);
ind = ind+Steilabfall+1;
Fiu_ind = ind(end);

ind = find(diff(F_spl)>0);
ind = ind-Steilabfall;
ind = ind(ind>0);
ind = ind+Steilabfall+1;
Fa_ind = ind(1);

fit_plat = polyfit(s_spl(Fgy_ind:Fiu_ind),F_spl(Fgy_ind:Fiu_ind),2);
F_fit_plat = polyval(fit_plat,s_spl);

Differenz = F_spl-F_fit_plat;

for i=Steil_min_ind:length(Differenz)-1
    if Differenz(i) < 0 && Differenz(i+1) > 0
        Index_naechste = [i i+1];
        A = [s_spl(Index_naechste(1)) F_spl(Index_naechste(1))];
        B = [s_spl(Index_naechste(2)) F_spl(Index_naechste(2))];
        C = [s_spl(Index_naechste(1)) F_fit_plat(Index_naechste(1))];
        D = [s_spl(Index_naechste(2)) F_fit_plat(Index_naechste(2))];
        break
    elseif i == length(Differenz)-1
        clear Differenz
        fit_anstieg = polyfit(s_spl(Steil_min_ind:Fgy_ind),...
                      F_spl(Steil_min_ind:Fgy_ind),1);
        F_fit_anstieg = polyval(fit_anstieg,s_spl);
        Differenz = F_fit_anstieg-F_fit_plat;
        for j=Steil_min_ind:length(s_spl)-1
            if Differenz(j) < 0 && Differenz(j+1) > 0
            Index_naechste = [j j+1];
            end
        end
        A = [s_spl(Index_naechste(1)) F_fit_anstieg(Index_naechste(1))];
        B = [s_spl(Index_naechste(2)) F_fit_anstieg(Index_naechste(2))];
        C = [s_spl(Index_naechste(1)) F_fit_plat(Index_naechste(1))];
        D = [s_spl(Index_naechste(2)) F_fit_plat(Index_naechste(2))];
        break
    end
end

S_1 = Schnittpunkt(A,B,C,D);

Index_neu = Index_naechste(2);

Differenz = F_spl-F_fit_plat;

clear Index_naechste
for i=Fiu_ind:Fa_ind
    if Differenz(i) > 0 && Differenz(i+1) <0
        Index_naechste = [i-1 i];
        A = [s_spl(Index_naechste(1)) F_spl(Index_naechste(1))];
        B = [s_spl(Index_naechste(2)) F_spl(Index_naechste(2))];
        C = [s_spl(Index_naechste(1)) F_fit_plat(Index_naechste(1))];
        D = [s_spl(Index_naechste(2)) F_fit_plat(Index_naechste(2))];
        break
    elseif i == Fa_ind
        clear Differenz
        fit_steil = polyfit(s_spl(Fiu_ind:Steilabfall),...
        F_spl(Fiu_ind:Steilabfall),1);
        F_fit_steil = polyval(fit_steil,s_spl);
        Differenz = F_fit_plat-F_fit_steil;
        for j=Fgy_ind:Fa_ind
            if Differenz(j) < 0 && Differenz(j+1) > 0
            Index_naechste = [j j+1];
            end
        end
        A = [s_spl(Index_naechste(1)) F_fit_plat(Index_naechste(1))];
        B = [s_spl(Index_naechste(2)) F_fit_plat(Index_naechste(2))];
        C = [s_spl(Index_naechste(1)) F_fit_steil(Index_naechste(1))];
        D = [s_spl(Index_naechste(2)) F_fit_steil(Index_naechste(2))];
        break
    end
end

S_2 = Schnittpunkt(A,B,C,D);

Index_neu = [Index_neu Index_naechste(1)];

for i=Fiu_ind:length(s_spl)
    if F_spl(i) > 0 && F_spl(i+1) <0
        Index_null = [i i+1];
        break
    end
end

A = [s_spl(Index_null(1)) F_spl(Index_null(1))];
B = [s_spl(Index_null(2)) F_spl(Index_null(2))];
C = [s_spl(Index_null(1)) 0];
D = [s_spl(Index_null(2)) 0];

S_3 = Schnittpunkt(A,B,C,D);

Index_neu = [Index_neu Index_null(2)];

s_norm = [s_spl(Int_start), S_1(1), s_spl(Index_neu(1):Index_neu(2)),...
    S_2(1), S_3(1),s_spl(end)];
Fit_norm = [F_spl(Int_start), S_1(2), ...
    F_fit_plat(Index_neu(1):Index_neu(2)), S_2(2), S_3(2),0];

Arbeit_norm = cumtrapz(s_norm,Fit_norm);

Int_ende_norm = Index_neu(2)-Index_neu(1)+5;

s_norm = s_norm(1:Int_ende_norm);
Fit_norm = Fit_norm(1:Int_ende_norm);

Fiu_norm_ind = (Index_neu(2)-Index_neu(1))+4;
```

```
Fa_norm_ind = (Index_neu(2)-Index_neu(1))+5;


[F_max_norm,F_max_ind] = max(Fit_norm);
F_max_norm = round(F_max_norm/10^3,2);
F_max_kal = round(F_max_norm*CF,2);
F_gy = round(S_1(2)/10^3,2);
F_gy_kal = round(F_gy*CF,2);

s_max_norm = round(s_norm(F_max_ind)*10^3,2);
s_gy = round(S_1(1)*10^3,2);
s_iu = round(S_2(1)*10^3,2);
s_a = round(S_3(1)*10^3,2);
s_t = round(S_3(1)*10^3,2);

Schlagarbeit_norm = round(max(Arbeit_norm),1);
Schlagarbeit_kal = round(Schlagarbeit_norm*CF,1);
W_max_norm = round(Arbeit_norm(F_max_ind),1);
W_max_kal = round(W_max_norm*CF,1);


Steigung = round(F_max_norm/s_max_norm,2);

B = 0.01; S = 0.04; L = 0.008;
sigma_gy = round(3*F_gy*10^3*S/(4*B*L^2)*10^-6,0);

elastisch = Fit_norm(2)*s_norm(2)/2/Schlagarbeit_norm*100;
plastisch = max(cumtrapz(s_norm(2:(Index_neu(2)-Index_neu(1)+2)),...
    Fit_norm(2:(Index_neu(2)-Index_neu(1)+2))))/Schlagarbeit_norm*100;
```

Case D

```
% Wie und warum das Unterprogrogramm funktioniert ist im Fall E detailiert
% erklärt!

fig = figure;
set(fig,'units','normalized','position',[0.1,0.1,0.8,0.8],...
    'Name','Kraft-Weg-Diagramm','Numbertitle','off','Toolbar','none',...
    'Menubar','none')
axes('Units','Normalized','Position',[0.1,0.1,0.8,0.8])
h = uicontrol('Style','togglebutton','String','und hier "bestätigen"',...
              'units','normalized','Position', [0.5,0.578,0.16,0.03],...
              'Fontsize',12);
plot(s_spl,F_spl)
xlim([min(s_spl) max(s_spl)])
```

```
text('units','normalized','Position',[0.5,0.68],'String',...
    {'Flanke des Steilanstiegs der', '"zweiten" Schwingung markieren'},...
    'Color','k','Fontsize',12);
Cursor = datacursormode(fig);
set(Cursor,'DisplayStyle','datatip','SnapToDataVertex','off','Enable','on')
waitfor(h,'value');
Cursorinformation = getCursorInfo(Cursor);
steigende_Flanke = Cursorinformation.DataIndex;
delete(gca)
delete(h)

set(fig,'units','normalized','position',[0.1,0.1,0.8,0.8],...
    'Name','Kraft-Weg-Diagramm','Numbertitle','off','Toolbar','none',...
    'Menubar','none')
axes('Units','Normalized','Position',[0.1,0.1,0.8,0.8])
h = uicontrol('Style','togglebutton','String','und hier "bestätigen"',...
              'units','normalized','Position', [0.5,0.578,0.16,0.03],...
              'Fontsize',12);
plot(s_spl,F_spl)
xlim([min(s_spl) max(s_spl)])
text('units','normalized','Position',[0.5,0.66],'String',...
    'Flanke des Steilabfalls markieren', 'Color','k','Fontsize',12);
Cursor = datacursormode(fig);
set(Cursor,'DisplayStyle','datatip','SnapToDataVertex','off','Enable','on')
waitfor(h,'value');
Cursorinformation = getCursorInfo(Cursor);
Steilabfall = Cursorinformation.DataIndex;
close

ind = find(diff(F_spl)<0);
ind = ind-steigende_Flanke;
ind = ind(ind>0);
ind = ind+steigende_Flanke;
Fgy_ind = ind(1);

ind = find(diff(F_spl)<0);
ind = ind-steigende_Flanke;
ind = ind(ind<0);
ind = ind+steigende_Flanke;
Steil_min_ind = ind(end);

ind = find(diff(F_spl)>0);
ind = ind-Steilabfall;
ind = ind(ind<0);
ind = ind+Steilabfall+1;
Fiu_ind = ind(end);


ind = find(diff(F_spl)>0);
ind = ind-Steilabfall;
ind = ind(ind>0);
ind = ind+Steilabfall+1;
Fa_ind = ind(1);

fit_plat = polyfit(s_spl(Fgy_ind:Fiu_ind),F_spl(Fgy_ind:Fiu_ind),2);
F_fit_plat = polyval(fit_plat,s_spl); % Fitfunktion

for i=Fa_ind:length(F_spl)
    if F_spl(i) <= 0.01*F_max
        if i <= Int_ende
            Fit_ende = length(F_spl);
            break
        else
            Fit_ende = i;
            break
        end
    else
        Fit_ende = length(F_spl);
    end
end

fit_auffang = polyfit(s_spl(Fa_ind:Fit_ende),F_spl(Fa_ind:Fit_ende),2);
F_fit_auffang = polyval(fit_auffang,s_spl);

Differenz = F_spl-F_fit_plat;

for i=Steil_min_ind:length(Differenz)-1
    if Differenz(i) < 0 && Differenz(i+1) > 0
        Index_naechste = [i i+1];
        A = [s_spl(Index_naechste(1)) F_spl(Index_naechste(1))];
        B = [s_spl(Index_naechste(2)) F_spl(Index_naechste(2))];
        C = [s_spl(Index_naechste(1)) F_fit_plat(Index_naechste(1))];
        D = [s_spl(Index_naechste(2)) F_fit_plat(Index_naechste(2))];
        break
    else
        clear Differenz
        fit_anstieg = polyfit(s_spl(Steil_min_ind:Fgy_ind),F_spl(Steil_min_ind:Fgy_ind),1
        F_fit_anstieg = polyval(fit_anstieg,s_spl);
        Differenz = F_fit_anstieg-F_fit_plat;
        for j=Steil_min_ind:length(s_spl)-1
            if Differenz(j) < 0 && Differenz(j+1) > 0
            Index_naechste = [j j+1];
            end
        end
        A = [s_spl(Index_naechste(1)) F_fit_anstieg(Index_naechste(1))];
        B = [s_spl(Index_naechste(2)) F_fit_anstieg(Index_naechste(2))];
        C = [s_spl(Index_naechste(1)) F_fit_plat(Index_naechste(1))];
        D = [s_spl(Index_naechste(2)) F_fit_plat(Index_naechste(2))];
        break
    end
end

S_1 = Schnittpunkt(A,B,C,D);

Index_neu = Index_naechste(2);
Differenz = F_spl-F_fit_plat;

clear Index_naechste
for i=Fiu_ind:Fa_ind
    if Differenz(i) > 0 && Differenz(i+1) <0
        Index_naechste = [i-1 i];
        A = [s_spl(Index_naechste(1)) F_spl(Index_naechste(1))];
        B = [s_spl(Index_naechste(2)) F_spl(Index_naechste(2))];
        C = [s_spl(Index_naechste(1)) F_fit_plat(Index_naechste(1))];
        D = [s_spl(Index_naechste(2)) F_fit_plat(Index_naechste(2))];
        break
    else
        clear Differenz
        fit_steil = polyfit(s_spl(Fiu_ind:Steilabfall),...
            F_spl(Fiu_ind:Steilabfall),1);
        F_fit_steil = polyval(fit_steil,s_spl);
        Differenz = F_fit_plat-F_fit_steil;
        for j=Fgy_ind:Fa_ind
            if Differenz(j) < 0 && Differenz(j+1) > 0
            Index_naechste = [j j+1];
            end
        end
        A = [s_spl(Index_naechste(1)) F_fit_plat(Index_naechste(1))];
        B = [s_spl(Index_naechste(2)) F_fit_plat(Index_naechste(2))];
        C = [s_spl(Index_naechste(1)) F_fit_steil(Index_naechste(1))];
        D = [s_spl(Index_naechste(2)) F_fit_steil(Index_naechste(2))];
        break
    end
end

S_2 = Schnittpunkt(A,B,C,D);

Index_neu = [Index_neu Index_naechste(1)];

Differenz = F_spl(Fiu_ind:Fa_ind)-F_fit_auffang(Fiu_ind:Fa_ind);

for i=1:length(Differenz)-1
    if Differenz(i) > 0 && Differenz(i+1) <0
        Index_naechste = [i+Fiu_ind i+1+Fiu_ind];
        A = [s_spl(Index_naechste(1)) F_spl(Index_naechste(1))];
        B = [s_spl(Index_naechste(2)) F_spl(Index_naechste(2))];
        C = [s_spl(Index_naechste(1)) F_fit_auffang(Index_naechste(1))];
        D = [s_spl(Index_naechste(2)) F_fit_auffang(Index_naechste(2))];
        break
    elseif i == length(Differenz)-1
        clear Differenz
        fit_steil = polyfit(s_spl(Fiu_ind:Steilabfall),...
        F_spl(Fiu_ind:Steilabfall),1);
        F_fit_steil = polyval(fit_steil,s_spl);
        Differenz = F_fit_steil-F_fit_auffang;
        for j=Fiu_ind:length(s_spl)-1
            if Differenz(j) < 0 && Differenz(j+1) > 0
            Index_naechste = [j j+1];
            end
        end
        A = [s_spl(Index_naechste(1)) F_fit_steil(Index_naechste(1))];
        B = [s_spl(Index_naechste(2)) F_fit_steil(Index_naechste(2))];
        C = [s_spl(Index_naechste(1)) F_fit_auffang(Index_naechste(1))];
        D = [s_spl(Index_naechste(2)) F_fit_auffang(Index_naechste(2))];
        break
    end
end

S_3 = Schnittpunkt(A,B,C,D);

Index_neu = [Index_neu Index_naechste(2)];

s_norm = [s_spl(Int_start), S_1(1), s_spl(Index_neu(1):Index_neu(2)),...
    S_2(1), S_3(1), s_spl((Index_neu(3)+Int_start):end)];
Fit_norm = [F_spl(Int_start), S_1(2), ...
    F_fit_plat(Index_neu(1):Index_neu(2)), S_2(2), S_3(2), ...
    F_fit_auffang((Index_neu(3)+Int_start):end)];

for i = Index_neu(3):length(s_norm)-1
    if Fit_norm(i) > 0.02*F_max && Fit_norm(i+1) <= 0.02*F_max
        Int_ende_norm = i+1;
        break
    else
        [egal,min_Auffang_ind]=min(Fit_norm(Index_neu(3):length(s_norm)));
        Int_ende_norm = min_Auffang_ind+Index_neu(3);
        break
    end
end

s_norm = s_norm(1:Int_ende_norm);
Fit_norm = Fit_norm(1:Int_ende_norm);

Fiu_norm_ind = (Index_neu(2)-Index_neu(1))+4;
Fa_norm_ind = (Index_neu(2)-Index_neu(1))+5;

Arbeit_norm = cumtrapz(s_norm,Fit_norm);

[F_max_norm,F_max_ind] = max(Fit_norm);
F_max_norm = round(F_max_norm/10^3,2);
F_max_kal = round(F_max_norm*CF,2);
F_gy = round(S_1(2)/10^3,2);
F_gy_kal = round(F_gy*CF,2);
F_a = round(S_3(2)/10^3,2);
F_a_kal = round(F_a*CF,2);
F_iu = F_max_norm;
F_iu_kal = F_max_kal;

s_max_norm = round(s_norm(F_max_ind)*10^3,2);
s_gy = round(S_1(1)*10^3,2);
s_iu = round(S_2(1)*10^3,2);
s_a = round(S_3(1)*10^3,2);
s_t = round(S_3(1)*10^3,2);

Schlagarbeit_norm = round(Arbeit_norm(Int_ende_norm),1);
Schlagarbeit_kal = round(Schlagarbeit_norm*CF,1);
W_max_norm = round(Arbeit_norm(F_max_ind),1);
W_max_kal = round(W_max_norm*CF,1);
W_a_norm = round(Arbeit_norm(Fa_norm_ind),1);
W_a_kal = round(W_a_norm*CF,1);
W_iu_norm = W_max_norm;
```

```
Steigung = round(F_max_norm/s_max_norm,2);

B = 0.01; S = 0.04; L = 0.008;
sigma_gy = round(3*F_gy*10^3*S/(4*B*L^2)*10^-6,0);

K =0.5;
AnteilVBF = round((1-(F_iu-F_a)/(F_max_norm+K*(F_max_norm-F_gy)))*100,0);


elastisch = Fit_norm(2)*s_norm(2)/2/Schlagarbeit_norm*100;
plastisch = max(cumtrapz(s_norm(2:(Index_neu(2)-Index_neu(1)+2)),Fit_norm(2:(Index_neu(2)
```

Case E

```
% Das Plateau soll vom ersten Maximum (Spitze) nach der "zweiten"
% Schwingung bis zur letzen Maxiumum vor dem Steilabfall gefittet werden.

% Flanke der zweiten Schwingung markieren
fig = figure;
set(fig,'units','normalized','position',[0.1,0.1,0.8,0.8],...
    'Name','Kraft-Weg-Diagramm','Numbertitle','off','Toolbar','none',...
    'Menubar','none')
axes('Units','Normalized','Position',[0.1,0.1,0.8,0.8])
h = uicontrol('Style','togglebutton','String','und hier "bestätigen"',...
              'units','normalized','Position', [0.5,0.578,0.16,0.03],...
              'Fontsize',12);
plot(s_spl,F_spl)
xlim([min(s_spl) max(s_spl)])
text('units','normalized','Position',[0.5,0.68],'String',...
    {'Flanke des Steilanstiegs der', '"zweiten" Schwingung markieren'},...
    'Color','k','Fontsize',12);
Cursor = datacursormode(fig);
set(Cursor,'DisplayStyle','datatip','SnapToDataVertex','off','Enable','on')
waitfor(h,'value');
Cursorinformation = getCursorInfo(Cursor);
steigende_Flanke = Cursorinformation.DataIndex;
delete(gca)
delete(h)

% Flanke des Steilabfalls markieren
set(fig,'units','normalized','position',[0.1,0.1,0.8,0.8],...
    'Name','Kraft-Weg-Diagramm','Numbertitle','off','Toolbar','none',...
    'Menubar','none')
axes('Units','Normalized','Position',[0.1,0.1,0.8,0.8])
```

```
h = uicontrol('Style','togglebutton','String','und hier "bestätigen"',...
              'units','normalized','Position', [0.5,0.578,0.16,0.03],...
              'Fontsize',12);
plot(s_spl,F_spl)
xlim([min(s_spl) max(s_spl)])
text('units','normalized','Position',[0.5,0.66],'String',...
    'Flanke des Steilabfalls markieren', 'Color','k','Fontsize',12);
Cursor = datacursormode(fig);
set(Cursor,'DisplayStyle','datatip','SnapToDataVertex','off','Enable','on')
waitfor(h,'value');
Cursorinformation = getCursorInfo(Cursor);
Steilabfall = Cursorinformation.DataIndex;
close


% Flanken durch markante Punkte abgrenzen

% Nächste Nullstelle der Ableitung nach Steilanstieg finden
% Alle Indizes der Ableitung finden die kleiner Null sind
ind = find(diff(F_spl)<0);
% Alle Indices verschieben, sodass der erste Wert sicher auf der Flanke
% liegt
ind = ind-steigende_Flanke;
% Somit kann mit logischer Indizierung das nächste Maximum gesucht werden
% und alle anderen gelöscht werden
ind = ind(ind>0);
% Flankenindex wieder dazurechnen und der erste Wert ist das der gesuchte
% Wendepunkt
ind = ind+steigende_Flanke;
Fgy_ind = ind(1);

% Letzte Nullstelle der Ableitung vor Steilanstieg finden
ind = find(diff(F_spl)<0);
ind = ind-steigende_Flanke;
ind = ind(ind<0);
ind = ind+steigende_Flanke;
Steil_min_ind = ind(end);

% Letzte Nullstelle der Ableitung vor Steilabfall finden
ind = find(diff(F_spl)>0);
ind = ind-Steilabfall;
ind = ind(ind<0);
ind = ind+Steilabfall+1; % +1 wegen diff!
Fiu_ind = ind(end);
```

```
% Nächste Nullstelle der Ableitung nach Steilabfall finden
ind = find(diff(F_spl)>0);
ind = ind-Steilabfall;
ind = ind(ind>0);
ind = ind+Steilabfall+1;
Fa_ind = ind(1);

% Fitten der Plateuschwingung und des Auffangsschwingung

% Polynomischer Fit 2ten Grades von Maxima zu Maxima des Plateaus
fit_plat = polyfit(s_spl(Fgy_ind:Fiu_ind),F_spl(Fgy_ind:Fiu_ind),2);
% Fitfunktion
F_fit_plat = polyval(fit_plat,s_spl);

%Ende festlegen wie weit gefittet werden soll
% --> bis einmal 1% der Maximalkraft erreicht wurde
for i=Fa_ind:length(F_spl)
    if F_spl(i) <= 0.01*F_max
        if i <= Int_ende
            Fit_ende = length(F_spl);
            break
        else
            Fit_ende = i;
            break
        end
    else
        Fit_ende = length(F_spl);
    end
end

% Polynomischer Fit 2ten Grades der Auffangschwingungen
fit_auffang = polyfit(s_spl(Fa_ind:Fit_ende),F_spl(Fa_ind:Fit_ende),2);
F_fit_auffang = polyval(fit_auffang,s_spl);


% Schnittpunkt zwischen Plateufit und Steilanstieg berechnen
% Es kann sein das kein Schnittpunkt vorhanden ist, um dieses Problem zu
% beheben wird eine lineare Verlängerung des Steilanstieg berechnet und
% diese zur Schnittpunktsberechnung herangezogen

% Zuerst die ungefähre Nähe des Schnittpunktes bestimmen
Differenz = F_spl-F_fit_plat; % Differenz zwischen Fit und Messung
```

```
% Suchintervall soll auf den Steilanstieg begrenzet werden
for i=Steil_min_ind:length(Differenz)-1
    % Nächsleigende Punkte mit Hilfe der Differenz suchen und als extra
    % Werte abspeichern
    if Differenz(i) < 0 && Differenz(i+1) > 0
        Index_naechste = [i i+1];
        A = [s_spl(Index_naechste(1)) F_spl(Index_naechste(1))];
        B = [s_spl(Index_naechste(2)) F_spl(Index_naechste(2))];
        C = [s_spl(Index_naechste(1)) F_fit_plat(Index_naechste(1))];
        D = [s_spl(Index_naechste(2)) F_fit_plat(Index_naechste(2))];
        break
    elseif i == length(Differenz)-1
        clear Differenz
        fit_anstieg = polyfit(s_spl(Steil_min_ind:Fgy_ind),...
                      F_spl(Steil_min_ind:Fgy_ind),1);
        F_fit_anstieg = polyval(fit_anstieg,s_spl);
        Differenz = F_fit_anstieg-F_fit_plat;
        for j=Steil_min_ind:length(s_spl)-1
            if Differenz(j) < 0 && Differenz(j+1) > 0
            Index_naechste = [j j+1];
            end
        end
        A = [s_spl(Index_naechste(1)) F_fit_anstieg(Index_naechste(1))];
        B = [s_spl(Index_naechste(2)) F_fit_anstieg(Index_naechste(2))];
        C = [s_spl(Index_naechste(1)) F_fit_plat(Index_naechste(1))];
        D = [s_spl(Index_naechste(2)) F_fit_plat(Index_naechste(2))];
        break
    end
end

% Punkte an Schinittpunktsfunktion übergeben
S_1 = Schnittpunkt(A,B,C,D);

Index_neu = Index_naechste(2); %Index herrausspeichern für spätere Zwecke

% Schnittpunkt zwischen Plateaufit Steilabfall  berechnen
% Es kann sein das kein Schnittpunkt vorhanden ist, um dieses Problem zu
% beheben wird eine lineare Verlängerung des Steilabfalls berechnet und
% diese zur Schnittpunktsberechnung herangezogen

% Zuerst die ungefähre Nähe des Schnittpunktes bestimmen
Differenz = F_spl-F_fit_plat; % Differenz zwischen Fit und Messung
```

```
% Suchintervall soll auf den Steilanstieg begrenzet werden
clear Index_naechste
for i=Fiu_ind:Fa_ind
    if Differenz(i) > 0 && Differenz(i+1) <0
        Index_naechste = [i-1 i];
        A = [s_spl(Index_naechste(1)) F_spl(Index_naechste(1))];
        B = [s_spl(Index_naechste(2)) F_spl(Index_naechste(2))];
        C = [s_spl(Index_naechste(1)) F_fit_plat(Index_naechste(1))];
        D = [s_spl(Index_naechste(2)) F_fit_plat(Index_naechste(2))];
        break
    elseif i == Fa_ind
        clear Differenz
        fit_steil = polyfit(s_spl(Fiu_ind:Fa_ind),F_spl(Fiu_ind:Fa_ind),1);
        F_fit_steil = polyval(fit_steil,s_spl);
        Differenz = F_fit_plat-F_fit_steil;
        for j=1:Fa_ind
            if Differenz(j) < 0 && Differenz(j+1) > 0
            Index_naechste = [j j+1];
            end
        end
        A = [s_spl(Index_naechste(1)) F_fit_plat(Index_naechste(1))];
        B = [s_spl(Index_naechste(2)) F_fit_plat(Index_naechste(2))];
        C = [s_spl(Index_naechste(1)) F_fit_steil(Index_naechste(1))];
        D = [s_spl(Index_naechste(2)) F_fit_steil(Index_naechste(2))];
        break
    end
end

S_2 = Schnittpunkt(A,B,C,D);

%Index herrausspeichern für spätere Zwecke
Index_neu = [Index_neu Index_naechste(1)];

% Schnittpunkt zwischen Steilabfall und Auffangfit berechnen

% Zuerst die ungefähre Nähe des Schnittpunktes bestimmen
Differenz = F_spl(Fiu_ind:Fa_ind)-F_fit_auffang(Fiu_ind:Fa_ind);

for i=1:length(Differenz)-1
    if Differenz(i) > 0 && Differenz(i+1) <0
        Index_naechste = [i+Fiu_ind i+1+Fiu_ind];
        A = [s_spl(Index_naechste(1)) F_spl(Index_naechste(1))];
        B = [s_spl(Index_naechste(2)) F_spl(Index_naechste(2))];
        C = [s_spl(Index_naechste(1)) F_fit_auffang(Index_naechste(1))];
        D = [s_spl(Index_naechste(2)) F_fit_auffang(Index_naechste(2))];
        break
    elseif i == length(Differenz)-1
        clear Differenz
        fit_steil = polyfit(s_spl(Fiu_ind:Steilabfall),...
            F_spl(Fiu_ind:Steilabfall),1);
        F_fit_steil = polyval(fit_steil,s_spl);
        Differenz = F_fit_steil-F_fit_auffang;
        for j=Fiu_ind:length(s_spl)-1
            if Differenz(j) < 0 && Differenz(j+1) > 0
            Index_naechste = [j j+1];
            end
        end
        A = [s_spl(Index_naechste(1)) F_fit_steil(Index_naechste(1))];
        B = [s_spl(Index_naechste(2)) F_fit_steil(Index_naechste(2))];
        C = [s_spl(Index_naechste(1)) F_fit_auffang(Index_naechste(1))];
        D = [s_spl(Index_naechste(2)) F_fit_auffang(Index_naechste(2))];
        break
    end
end


S_3 = Schnittpunkt(A,B,C,D);

Index_neu = [Index_neu Index_naechste(2)];

% Neue Messdaten aus der Verknüpfung der Fits schreiben
s_norm = [s_spl(Int_start), S_1(1), s_spl(Index_neu(1):Index_neu(2)),...
    S_2(1), S_3(1), s_spl((Index_neu(3)+Int_start):end)];
Fit_norm = [0, S_1(2), F_fit_plat(Index_neu(1):Index_neu(2)), S_2(2),...
    S_3(2), F_fit_auffang((Index_neu(3)+Int_start):end)];



% Integralende laut Norm ist wenn F = 0.02*F_max
F_max_norm = max(Fit_norm);
for i = Index_neu(3):length(s_norm)-1
    if Fit_norm(i) > 0.02*F_max_norm && Fit_norm(i+1) <= 0.02*F_max_norm
        Int_ende_norm = i+1;
        break
    else
        [egal,min_Auffang_ind] = min(Fit_norm(Index_neu(3):length(s_norm)));
        Int_ende_norm = min_Auffang_ind+Index_neu(3);
        break
    end
end

s_norm = s_norm(1:Int_ende_norm);
Fit_norm = Fit_norm(1:Int_ende_norm);

Arbeit_norm = cumtrapz(s_norm,Fit_norm);


% Charakteristische Werte berrechnen:
Fiu_norm_ind = (Index_neu(2)-Index_neu(1))+4;
Fa_norm_ind = (Index_neu(2)-Index_neu(1))+5;

[F_max_norm,F_max_ind] = max(Fit_norm);

F_gy = round(Fit_norm(2)/10^3,2);
F_gy_kal = round(F_gy*CF,2);
F_iu = round(S_2(2)/10^3,2);
F_iu_kal = round(F_iu*CF,2);
F_a = round(S_3(2)/10^3,2);
F_a_kal = round(F_a*CF,2);
F_max_norm = round(F_max_norm/10^3,2);
F_max_kal = round(F_max_norm*CF,2);


s_gy = round(s_norm(2)*10^3,2);
s_max_norm = round(s_norm(F_max_ind)*10^3,2);
s_iu = round(S_2(1)*10^3,2);
s_a = round(S_3(1)*10^3,2);
s_t = round(s_norm(Int_ende_norm)*10^3,2);

Schlagarbeit_norm = round(Arbeit_norm(Int_ende_norm-Int_start),1);
Schlagarbeit_kal = round(Schlagarbeit_norm*CF,1);
W_max_norm = round(Arbeit_norm(F_max_ind),1);
W_max_kal = round(W_max_norm*CF,1);
W_iu_norm = round(Arbeit_norm(Fiu_norm_ind),1);
W_iu_kal = round(W_iu_norm*CF,1);
W_a_norm = round(Arbeit_norm(Fa_norm_ind),1);
W_a_kal = round(W_a_norm*CF,1);

K =0.5;
AnteilVBF = round((1-(F_iu-F_a)/(F_max_norm+K*(F_max_norm-F_gy)))*100,0);

Steigung = round(S_1(2)/(S_1(1))*10^-6,2);
```

```
B = 0.01; S = 0.04; L = 0.008;
sigma_gy = round(3*F_gy*10^3*S/(4*B*L^2)*10^-6,0);

elastisch = Fit_norm(2)*s_norm(2)/2/Schlagarbeit_norm*100;
plastisch = max(cumtrapz(s_norm(2:(Index_neu(2)-Index_neu(1)+2)),...
    Fit_norm(2:(Index_neu(2)-Index_neu(1)+2))))/Schlagarbeit_norm*100;
```

Case F

```
% Wie und warum das Unterprogrogramm funktioniert ist im Fall E detailiert
% erklärt!

fig = figure;
set(fig,'units','normalized','position',[0.1,0.1,0.8,0.8],...
    'Name','Kraft-Weg-Diagramm','Numbertitle','off','Toolbar','none',...
    'Menubar','none')
axes('Units','Normalized','Position',[0.1,0.1,0.8,0.8])
h = uicontrol('Style','togglebutton','String','und hier "bestätigen"',...
              'units','normalized','Position', [0.5,0.578,0.16,0.03],...
              'Fontsize',12);
plot(s_spl,F_spl)
xlim([min(s_spl) max(s_spl)])
text('units','normalized','Position',[0.5,0.68],'String',...
    {'Flanke des Steilanstiegs der', '"zweiten" Schwingung markieren'},...
    'Color','k','Fontsize',12);
Cursor = datacursormode(fig);
set(Cursor,'DisplayStyle','datatip','SnapToDataVertex','off','Enable','on')
waitfor(h,'value');
Cursorinformation = getCursorInfo(Cursor);
steigende_Flanke = Cursorinformation.DataIndex;
close

ind = find(diff(F_spl)<0);
ind = ind-steigende_Flanke;
ind = ind(ind>0);
ind = ind+steigende_Flanke;
Fgy_ind = ind(1);

ind = find(diff(F_spl)<0);
ind = ind-steigende_Flanke;
ind = ind(ind<0);
ind = ind+steigende_Flanke;
Steil_min_ind = ind(end);
```

```
[fit_rest,s1,u1] = polyfit(s_spl(Fgy_ind:end),F_spl(Fgy_ind:end),10);
F_fit_rest = polyval(fit_rest,s_spl,s1,u1);

Differenz = F_spl-F_fit_rest;

for i=Steil_min_ind:length(Differenz)-1
    if Differenz(i) < 0 && Differenz(i+1) > 0
        Index_naechste = [i i+1];
        A = [s_spl(Index_naechste(1)) F_spl(Index_naechste(1))];
        B = [s_spl(Index_naechste(2)) F_spl(Index_naechste(2))];
        C = [s_spl(Index_naechste(1)) F_fit_rest(Index_naechste(1))];
        D = [s_spl(Index_naechste(2)) F_fit_rest(Index_naechste(2))];
        break
    elseif i == length(Differenz)-1
        clear Differenz
        fit_anstieg = polyfit(s_spl(Steil_min_ind:Fgy_ind),F_spl(Steil_min_ind:Fgy_ind),1
        F_fit_anstieg = polyval(fit_anstieg,s_spl); % Fitfunktion
        Differenz = F_fit_anstieg-F_fit_rest;
        for j=Steil_min_ind:length(s_spl)-1
            if Differenz(j) < 0 && Differenz(j+1) > 0
            Index_naechste = [j j+1];
            end
        end
        A = [s_spl(Index_naechste(1)) F_fit_anstieg(Index_naechste(1))];
        B = [s_spl(Index_naechste(2)) F_fit_anstieg(Index_naechste(2))];
        C = [s_spl(Index_naechste(1)) F_fit_rest(Index_naechste(1))];
        D = [s_spl(Index_naechste(2)) F_fit_rest(Index_naechste(2))];
        break
    end
end

S_1 = Schnittpunkt(A,B,C,D);

Index_neu = Index_naechste(2);
s_norm = [s_spl(Int_start), S_1(1), s_spl(Index_neu:end)];
Fit_norm = [F_spl(Int_start), S_1(2), F_fit_rest(Index_neu:end)];

Arbeit_norm = cumtrapz(s_norm,Fit_norm);

[F_max_norm,F_max_ind] = max(Fit_norm);

for i = Index_neu:length(s_norm)-1
    if Fit_norm(i) > 0.02*F_max && Fit_norm(i+1) <= 0.02*F_max
        Int_ende_norm = i+1;
        break
    else
        Int_ende_norm = length(s_norm);
    end
end

s_norm = s_norm(1:Int_ende_norm);
Fit_norm = Fit_norm(1:Int_ende_norm);


F_gy = round(S_1(2)/10^3,2);
F_gy_kal = round(F_gy*CF,2);
F_max_norm = round(F_max_norm/10^3,2);
F_max_kal = round(F_max_norm*CF,2);

s_gy = round(S_1(1)*10^3,2);
s_max_norm = round(s_norm(F_max_ind)*10^3,2);
s_t = round(s_norm(Int_ende_norm)*10^3,2);

Schlagarbeit_norm = round(Arbeit_norm(Int_ende_norm-Int_start),1);
Schlagarbeit_kal = round(Schlagarbeit_norm*CF,1);
W_max_norm = round(Arbeit_norm(F_max_ind),1);
W_max_kal = round(W_max_norm*CF,1);

Steigung = round(S_1(2)/(S_1(1))*10^-6,2);

B = 0.01; S = 0.04; L = 0.008;
sigma_gy = round(3*F_gy*10^3*S/(4*B*L^2)*10^-6,0);

elastisch = Fit_norm(2)*s_norm(2)/2/Schlagarbeit_norm*100;
plastisch = max(cumtrapz(s_norm(2:Int_ende_norm),...
    Fit_norm(2:Int_ende_norm)))/Schlagarbeit_norm*100;
```