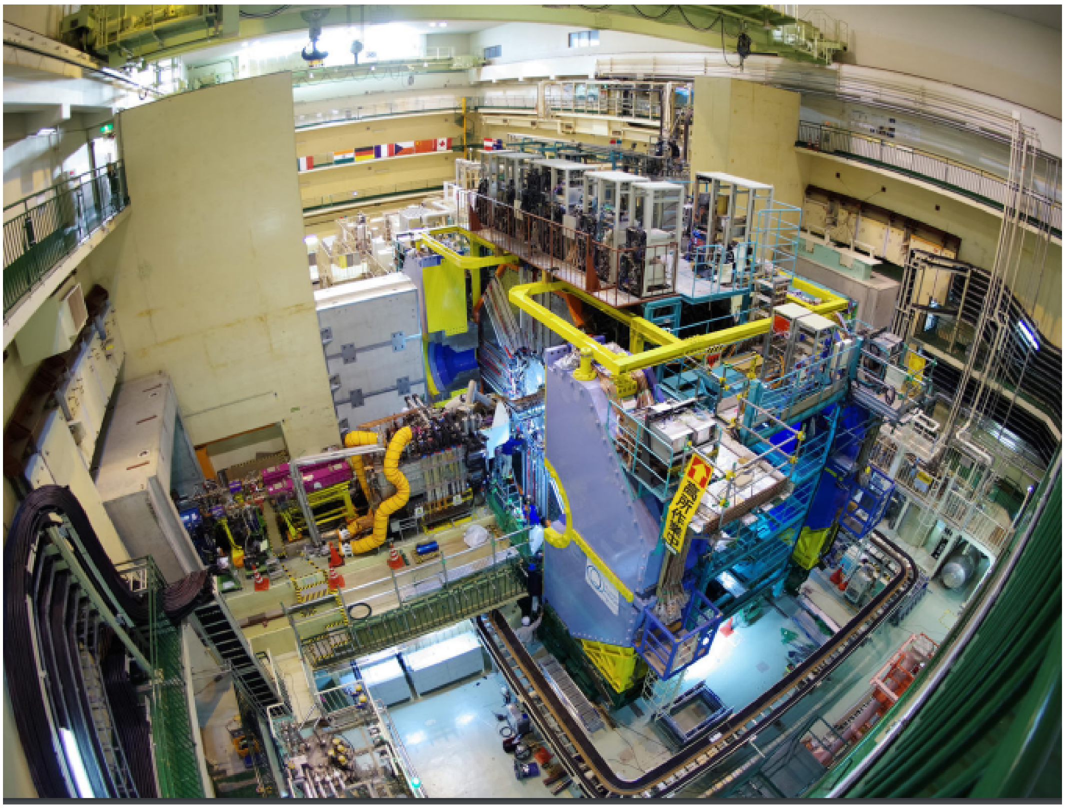
University of Bonn
Advanced Laboratory Course physics601
E215

Particle-antiparticle oscillations at BELLE-II



Version 1.03

Bonn, 23rd February 2022

Contents

CHAPTER 1

Introduction

The University of Bonn participates in the BELLE-II experiment at the KEK accelerator in Japan. Pairs of heavy B mesons are produced in collisions of electron positron beams. Due to the nature of weak interactions, a neutral B meson can transform into its antiparticle. This transformation can be observed as a flavor oscillation between the pure and mixed states. In the experiment students will analyze simulated data from the BELLE-II experiment to extract the oscillation frequency.

Required knowledge:

- Elementary particles and their properties, symmetries and conservation laws, standard model
- Interaction of particles and matter, particle accelerators and detectors
- Statistical analysis, basic python knowledge (to the extent discussed in the script)

The lab exercise has three parts:

- Part one: Study of graphical event representations (event displays). Students learn to distinguish the different kind of collision reactions at BELLE-II and how to measure important collision observables.
- Part two: Study of a specific decay chain of neutral B meson candidates: $\overline{B}^0 \rightarrow D^{*+}\,\pi^-$, $D^{*+} \rightarrow D^0\,\pi^+$, $D^0 \rightarrow K^-\,\pi^+$
- Part three: Observation of particle to antiparticle transformation and measurement of the B^0-$\overline{B}^0$ oscillation frequency

Literature:

- this lab script, available on eCampus [EvT]
- D. Griffiths, Introduction into Elementary Particles [Gri08]
- M. Thomson, Modern Particle Physics [Tho13]

1.1 Questions before the start of the lab

Please be prepared to answer the following questions at the beginning of the lab unit.

- How does a long-lived charged particle interact with matter (detector material). What is a electromagnetic shower, what is a hadronic shower? What is bremsstrahlung?
- What is the difference between a meson and a baryon?
- What is the quark content of the $\Upsilon(4S)$?
- What is the quark content of the B^0 mesons?
- Since we are dealing in detail with specific hadrons, please create a "cheat sheet" of relevant hadrons, their masses, lifetimes and quark content and use it throughout the lab exercise. The relevant particles are: $\Upsilon(4S)$, B^0, $\overline{B}^0$, D^0, D^+, K^-, π^+, π^0, muon, tau-lepton. You find the information on the PDG website https://pdglive.lbl.gov [Gro].
- We consider the decay $\Upsilon(4S) \to B^0\,\overline{B}^0$. Look up the masses of the B^0 meson and the $\Upsilon(4S)$ mesons and calculate the momentum of the B^0 in the $\Upsilon(4S)$ rest frame.
- At BELLE-II the $\Upsilon(4S)$ is created with a boost in the forward direction. Why is the B momentum in the laboratory frame approximately known?
- Why is the B^0 to $\overline{B}^0$ transition possible?
- Which interaction is responsible for the decay of the B^0 meson?
- Which interaction is responsible for the decay of the D^{*+}?
- What is a semi-leptonic B decay?
- What is a hadronic B decay and give an example.
- How are charged pions and kaons distinguished from each other at BELLE?
- If the $\Upsilon(4S)$ decays into two neutral B mesons, are they created in an entangled state?
- What is the definition of the time dependent asymmetry (function)?
- What is a histogram?

CHAPTER 2

Laboratory Instructions

2.1 Visual inspection of BELLE-II events

The tutor will help you getting started with the BELLE-II event display. Learn to distinguish the relevant energy depositions in the tracker (hits) and the calorimeter. You are tasked to scan several sets of BELLE-II events. Each set contains (simulated) events of one physics process. The most relevant processes at BELLE-II are:

- $e^- \, e^+ \rightarrow \Upsilon(4S) \rightarrow B \, \overline{B}$
- $e^- \, e^+ \rightarrow q \, \overline{q} \rightarrow$ hadrons
- $e^- \, e^+ \rightarrow e^- \, e^+$
- $e^- \, e^+ \rightarrow \mu^- \, \mu^+$
- $e^- \, e^+ \rightarrow \tau^- \, \tau^+$
- $e^- \, e^+ \rightarrow \gamma \, \gamma$

Use the BELLE-II event display to identify the correct process for each set. Please keep in mind charged particles interact differently with the calorimeter: Electrons shower in the first layers of the calorimeter, showers initiated by charged hadrons reach intermediate layers while muons do not initiate a shower and traverse the whole detector.

2.2 B meson reconstruction

There are hundreds of different $\overline{B}^0$ decays[1]. The decay chain $\overline{B}^0 \rightarrow D^{*+} \, \ell^- \, \overline{\nu}$, $D^{*+} \rightarrow D^0 \, \pi^+$, $D^0 \rightarrow K^- \, \pi^+$ is however an excellent choice for this exercise, due to the (relatively) high branching ratios and the clear signature of the D^{*+} decay. The D^{*+} meson is an excited state of a charm quark and an anti-d-quark. It decays strongly and with 68% probability into a D^0 and a π^+ particle. Due to the restricted kinematics, the D^{*+} decay can be selected with high purity[2]. The D^0 itself is unstable and

[1] Wherever we refer to a specific decay chain we implicitly mean the anti-particle as well

[2] Purity is defined as the ratio of selected signal events over the total number of selected events.

decays after a short lifespan into a $K^- \pi^+$ pair. All three decay particles are charged and leave tracks in the BELLE-II central tracking chamber. The data of these tracks: angular orientation at the origin and the track momentum are stored in the n-tuple, see section 4.4. It is understood that we actually study not just the decays of $\overline{B}^0$ but also of decays of the B^0. So any mention of a specific particle also implies also its charge conjugate state. The data are given as a table of measurements in a csv file. Each row represents a tuple of 5 tracks, four from the $\overline{B}^0$ decay chain and one additional lepton track that we use for tagging purposes. The data sample contains almost all possbile 5-track combinations with only minor preselection criteria applied to reduce the data size. You will start working with a jupyter notebook and read in the csv file into a pandas dataframe. The whole exercise can be done using pandas and numpy commands only. Further information about python, jupyter, pandas and numpy can be found in section 4.2.

2.2.1 Reconstruction of the D^0 decay

We start with the last decay in the chain: $D^0 \rightarrow K^- \pi^+$.

- Calculate the invariant mass of the kaon pion pairs from the data in the dataset and add the invariant mass as a new column to the pandas dataset. Create a histogram that shows the invariant mass with the D^0 mass peak above the combinatorial background. Which requirements on the pion and kaon charges make sense?
- Try to select a data sample with enhanced D^0 content and measure the purity of your selected candidates. A purity above 65% should be possible.
- Measure the average momentum of the (combinatorial) background in mass sidebands next to the D^0 mass peak.
- Measure the average momentum of D^0 particles. Eliminate the influence of background on your measurement.
- How does your result compare to (a) D^0 coming from hadronic two-body B decays such as $\overline{B}^0 \rightarrow D^0\pi^0$ and (b) PDzero from the decay chain $\overline{B}^0 \rightarrow D^{*+} \ell^- \overline{\nu}$, $D^{*+} \rightarrow D^0 \pi^+$. Only a rough estimate is required for the expected average momentum in (b).

End of day one.

2.2.2 Reconstruction of the D^{*+} decay

- Create a two-dimensional scatter plot of D^0 momentum versus slow-pion momentum[3].
- Reconstruct the invariant mass of the three particles that form the D^{*+} candidate. Create a histogram of the D^{*+} mass.
- Create a histogram of the mass difference $M_{\text{diff}} = \text{M}(D^{*+}) - \text{M}(D^0)$. Why is the D^{*+} peak more prominent in M_{diff} than in $\text{M}(D^{*+})$?
- Add $\text{M}(D^{*+})$ and M_{diff} as columns to the pandas dataset.

[3] For a discussion of "slow" pions in D^{*+} decay see section 3.3

- Select a data sample with enhanced D^{*+} content. Use mass and charge cuts to reject background. A purity above 80% should be possible. Measure the purity value.

The selection of the D^{*+} particle is an important step towards reconstructing the full B meson decay chain. Therefore discuss your results with the tutor before you continue.

2.2.3 Reconstruction of a full B meson decay chain

In this part of the lab exercise you select events compatible with one B particle (the signal B) decaying into $D^{*+}\ \ell^-\ \overline{\nu}$ and the other B particle (the tag B) decaying into a non-specific semi-leptonic decay mode. Therefore we talk about tag lepton and signal lepton. Since it is not apriori clear which lepton is which, each event appears twice in the dataset with tag and signal lepton interchanged. The n-tuple contains the information from the signal and tag leptons. Use the signal lepton and try to form good decay chain candidates for B to $D^{*+}\ \ell^-\ \overline{\nu}$. Try to avoid situations where both leptons (tag and signal) fulfill your criteria. Detailed instructions:

- Plot the difference between the z_0 positions[4] of signal and tag lepton.
- Plot as a histogram the average z_0 position of your D^{*+} by combining the z_0 values of the three decay particles from the D^{*+}.
- Create a scatter plot of $z_0(\ell_s)$ versus $z_0(D^{*+})$. Do you see a correlation?
- Plot as a histogram the invariant mass of the visible part of the signal B.
- Select a data sample consistent with the $\overline{B}^0 \to D^{*+}\ \ell^-\ \overline{\nu}$ hypothesis. Find selection criteria for the signal lepton candidate such that fits to the D^{*+}. Plot the above histograms for your final selection.

2.3 Measurement of oscillation frequency

The measurement of $B^0\ \overline{B}^0$ oscillations is based on the reconstructed $B \to D^{*+}\ell\nu$ decay chain and the flavor of the lepton of the other B meson (tag lepton). If both leptons have opposite charge the candidate entry counts as not-oscillated. If the leptons have the same charge the entry counts as oscillated. The oscillations manifests itself in the asymmetry as a function of $|\Delta t|$. Detailed instructions:

- Determine the asymmetry function $\mathcal{A}$ as a function of $|\Delta t|$. Use the $|\Delta t|$ variable (`deltat`) from the dataframe (see Sec. 4.4).
  $$\mathcal{A} = \frac{N_{\text{not-osc}} - N_{\text{osc}}}{N_{\text{not-osc}} + N_{\text{osc}}}$$
  Discuss the obtained asymmetry function with your tutor.
- To be done at home: Measure the oscillation frequency by applying a χ^2-fit to the function. Due to the complications discussed in section 3.4 restrict the fit to a range $|\Delta t| < 8 - 12$ ps.
- To be done at home: Determine the dilution factor.

[4] The z_0 of a charged particle is a track parameter that measures the track's closest approach to the beam axis. It is a good approximation to the z coordinate of the particle's point of origin.

CHAPTER 3

Important physics concepts

3.1 SuperKEKB accelerator complex

The BELLE-II detector is located at the Super-KEKB accelerator complex in Japan[Ohn+13]. The last upgrade of the KEKB accelerator complex in Japan resulted in an electron positron ring accelerator with unprecedented luminosity (Super-KEKB). The purpose of the Super-KEKB is to produce B mesons in the decay of the $\Upsilon(4S)$ resonance. This is a bound $b\bar{b}$ state with cardinal quantum number $n = 4$ and $J^{PC} = 1^{--}$. It is the first resonance heavy enough to decay into a pair of B mesons. The $\Upsilon(4S)$ decays to approximately 50% into $B^0\,\overline{B}^0$ and 50% into $B^+\,B^-$.

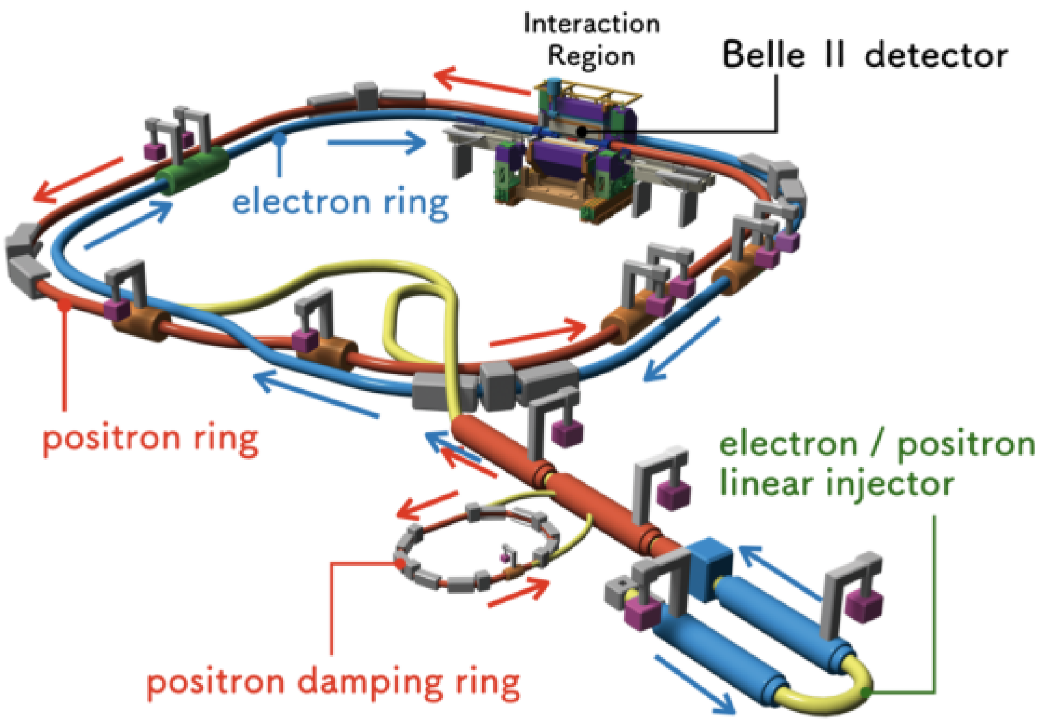


Figure 3.1: The Super-KEKB accelerator complex [web].

The accelerator operates at the $\Upsilon(4S)$ resonance. In order to provide a boost to the decay products the beams are asymmetric with the electron ring at 7.0 GeV beam energy and the positron beam at 4.0 GeV. The center-of-mass energy is equal to the mass of the $\Upsilon(4S)$. The resulting boost to the $\Upsilon(4S)$ is 0.28. Since the mass of the $\Upsilon(4S)$ is very close to the sum of the two B mesons each B meson inherits approximately the same boost and half the $\Upsilon(4S)$ momentum with which they travel almost exactly along the beam axis (z-direction). Both mesons travel down the z-direction and decay with an average lifetime of about 1.5 ps at different positions in z. The z-position of a track's start point is thus a good estimator of the decay point of a B meson and also of its flight length and life span. Please note that the start point of the B mesons flight path is smeared out by the beam spot which is centered around the coordinate systems origin but spreads out by several centimeters in z-direction. Besides $\Upsilon(4S)$ $\rightarrow$ BB there are also other processes. The dominant process is actually $e^-e^+ \rightarrow q\overline{q} \rightarrow$ two jets, with u,d,s,c as possible quark species. The cross section for $e^-e^+ \rightarrow q\overline{q}$ is about 3 times larger than the $\Upsilon(4S)$ production cross section.

The $\Upsilon(4S)$ decays with about 50% to $B^0\overline{B}^0$ and 50% to B^+B^-. Other decay modes are expected to be extremely rare.

3.2 BELLE-II detector

The BELLE-II detector is an upgrade of the BELLE detector that operated a KEK from 1999 to 2010. It consists of the following subdetector systems:

- Tracker: this detector component consist of low-material silicon detectors and gas-filled detectors to measure the trajectory of long-lived charged particles. Due to the presence of a magnetic field parallel to the incoming beams, the charged particle trajectories are bent in the plane perpendicular to the beams [Par]. The momentum of the particle can be determined from the track's curvature.

- Calorimeter: A relatively dense and thick detector component that induces particle showers in the passage of most particles. (all particles except muons and neutrinos).

- Cherenkov Detector. This detector essentially measures the velocity of charged particles. In conjunction with the momentum we obtain the mass and thus the particle species.

- KLM detector. The outermost part of the BELLE-II detector is the K^0-Long and muon identification system. It consists of multiple layers of scintillators and resistive plate chambers, located inbetween the magnetic flux-return iron plates.

Further information may be found at https://belle2.jp/detector.

3.3 B mesons and their decays

In particle collisions, energy and momentum are conserved [MT]. Since energy and momentum form the relativistic four-momentum we also talk about 4-momentum conservation (each component is conserved). The lorentz product of a lorentz vector with itself is invariant under change of reference

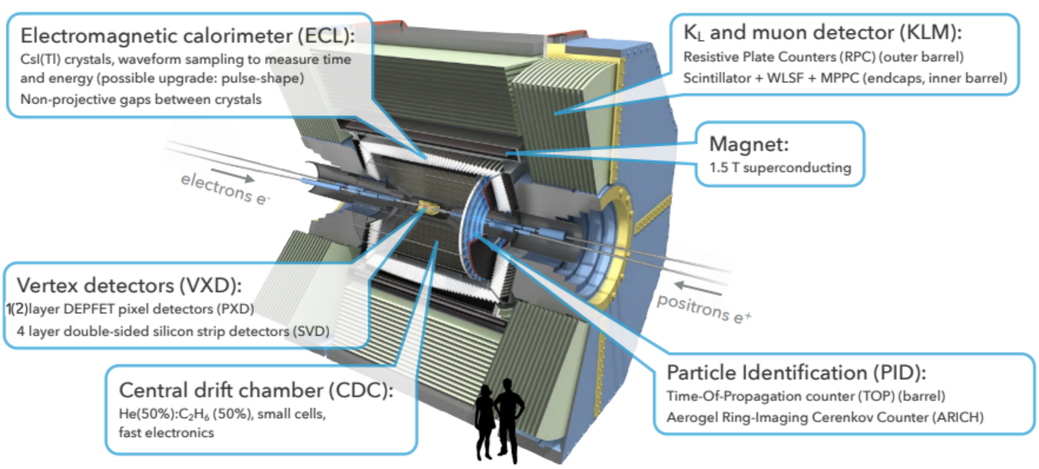


Figure 3.2: The BELLE-II detector [Kou+19].

systems (lorentz boosts). The lorentz product of a four momentum with itself is the squared invariant mass in natural units (where c=1).

$$E^2 - P_x^2 - P_y^2 - P_z^2 = M^2$$

Since four momentum conservation ensures that the sum of daughter momenta in a particle decay is equal to the mother particle four-momentum, it is easy to calculate the invariant mass of the mother if you have measured the daughter four-momenta.

For brevity we only discuss $\overline{B}^0$ decays. But all said applies to B^0, B^+, B^- accordingly. The $\overline{B}^0$ is the lowest-mass meson state with $b\,\overline{d}$ content. It can thus decay only by changing flavor. This proceeds via a weak transition of the b quark to a charm quark plus a virtual W boson. This is similar to β-decay in nuclear physics. There is enough phase space for the virtual W boson to transform either into a lepton-neutrino pair as in beta decays or into a q anti-q' pair like $u\,\overline{d}$. The light quark that accompanies the b-quark is in most decays just a spectator and forms, together with the charm quark, a charmed meson (Fig 3.3).

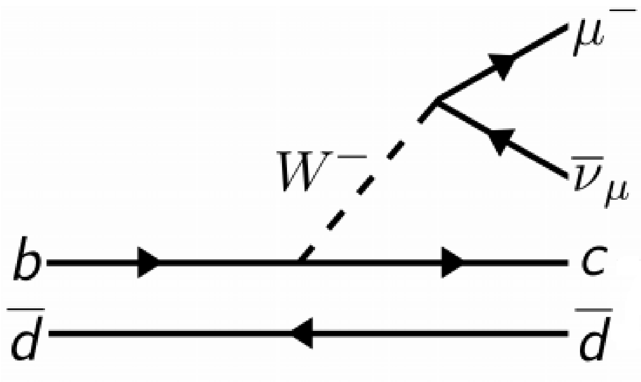


Figure 3.3: Example of a semi-leptonic B decay.

The simplest spectator decay with a lepton in the final state is

$$b\overline{d} \rightarrow cW^{*-}\overline{d} \rightarrow c\overline{d}e^{-}\overline{\nu}_e$$

B decays with a lepton and a neutrino plus hadron(s) in the final state are called semi-leptonic decays. B decays without lepton are called hadronic B decays. About 25% of all B decays are semi-leptonic, about 75% are hadronic. There are also non-spectator decays, decays without charm and other exotic decays, but there is no time to discuss them here.

Several different mesons with $c\overline{d}$ content exist, the lowest being the D^+ meson. The meson is held together by strong interaction and similar to atoms can form excited states. There are about a dozen excited states known with $c\overline{d}$ content. The most prominent excited state is the D^{*+} which is labelled as D*(2010)$^+$ in the charmed meson section of the Particle Data Group (PDG web page: https://pdglive.lbl.gov/). The $D^{*+} \rightarrow D^0\,\pi^+$ decay has two important features:

1. There is little momentum transfer to the decay products as the mass sum of D^0 and π^+ is just below the D^{*+} mass so that there is little phase space left.

2. There is a large disparity in the masses of the two decay particles. As a consequence the pion momentum is much smaller than the D meson momentum in the lab frame.

Due to the low momentum, the pion in the $D^{*+} \rightarrow D^0\,\pi^+$ decay chain is called ”slow pion”.

3.4 Flavor oscillations

Oscillations between particle and antiparticle were first observed in the $K^0\overline{K}^0$ system in the 1950s [CG09]. Meanwhile oscillations have been established in $B^0\overline{B}^0$ in the 1980s [Alb+87] and later in a few other meson systems. The time evolution of a $B^0\,\overline{B}^0$ system is given by a phenomenological Hamiltonian matrix H:

$$i\hbar\frac{\partial}{\partial t}\begin{pmatrix}B^0\\ \overline{B}^0\end{pmatrix} = H\begin{pmatrix}B^0\\ \overline{B}^0\end{pmatrix} \tag{3.1}$$

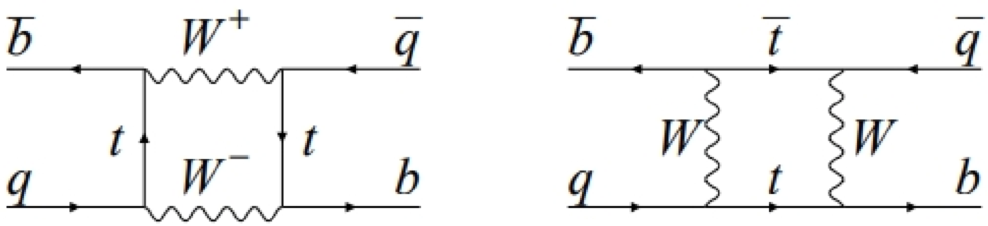


Figure 3.4: Second order weak transitions in the B system (from [Sch]).

The solution to the time evolution problem is exp($it\ H\left|B_{start}\right\rangle$), with $\left|B_{start}\right\rangle$ being the state at t=0. If we neglect B meson decay, H is hermitian. The eigenstates of H are identified as mass eigenstates. These mass eigenstates are linear combinations of B^0 and $\overline{B}^0$ and only identical to B^0 and $\overline{B}^0$ if the off-diagonal elements of H would vanish. However, the time evolution of the system allows for particle-antiparticle transitions due to second order weak interactions shown in fig. 3.4. If one assumes CP conservation the mass eigenstates are also CP eigenstates and are given by

$B_1 = \frac{1}{\sqrt{2}}(B^0 + \overline{B}^0)$, $B_2 = \frac{1}{\sqrt{2}}(B^0 - \overline{B}^0)$ with masses $M_{1,2}$. The two mass eigentstates are not required to have exactly the same mass and a small mass difference is expected. The time evolution in terms of mass eigenstates is simple [Tho13; Kuh12]:

$$\left|B_{1,2}(t)\right\rangle = \left|B_{1,2}(0)\right\rangle e^{im_{1,2}c^2 t/\hbar} = \left|B_{1,2}(0)\right\rangle e^{im_{1,2}t}, \text{ in natural units} \tag{3.2}$$

The time evolution for a linear combination of eigenstates $B = c_1 B_1 + c_2 B_2$

$$|B(t)\rangle = c_1 \left|B_1(t)\right\rangle + c_2 \left|B_2(t)\right\rangle . \tag{3.3}$$

For example, the time evolution of a initially pure B^0 state is straightforward $B^0 = \frac{1}{2}(B_1 + B_2)$.

If we take the possibility of particle decays into account, H becomes non-hermitian and can then be parametrized as a sum $H = M + i\Gamma$, with $M = \frac{1}{2}(H + H^\dagger)$ and $\Gamma = \frac{1}{2i}(H - H^\dagger)$ being the hermitian mass and the hermitian decay matrix. The two mass eigentstates still exist and have their own lifetimes. Depending on the meson system these lifetime differences can be significant like in the $K^0\,\overline{K}^0$ system. The B^0-$\overline{B}^0$ system however is quite the opposite: the measured lifetime difference is compatible with zero and throughout this lab we assume the difference to be exactly zero, as this simplifies our calculations significantly. In this approximation the decay matrix is diagonal with two identical diagonal elements $\Gamma_0 = 1/\tau$, where τ is the average lifetime of the B^0 meson.

This mass differences has a significant impact on the time evolution of the system. Let us assume a particle state that is pure B^0 at t=0. The probability to find a B^0 at time t is given by

$$W_{B^0}(t) \;=\; \frac{1}{2}e^{-\Gamma t}(1 + \cos(\Delta M t)) \tag{3.4}$$

and

$$W_{\bar{B}^0}(t) \;=\; \frac{1}{2}e^{-\Gamma t}(1 - \cos(\Delta M t)) \tag{3.5}$$

The oscillation frequency (angular frequency) is equal to ΔM in natural units and is typically expressed in inverse picoseconds. If a pure B^0 state is observed at t=0 and a $\overline{B}^0$ (is observed) at a later time, or vice versa $\overline{B}^0$ going to B^0, we call this state **oscillated**. We call the number of observed oscillated states N_{osc}. If a pure B^0 state is observed at t=0 and a B^0 at a later time we call this **not-oscillated** and the number of those states $N_{\text{not-osc}}$. In general a pure state measured at $t = 0$ will be a mixture of both components at a later time. This is also known as B-mixing.

We define the asymmetry function as

$$\mathcal{A} = \frac{N_{\text{not-osc}} - N_{\text{osc}}}{N_{\text{not-osc}} + N_{\text{osc}}} \tag{3.6}$$

The asymmetry is a function of time t. Using equations 3.4 and 3.5 we obtain for the asymmetry function

$$\mathcal{A} = \frac{N_{\text{not-osc}}(t) - N_{\text{osc}}(t)}{N_{\text{not-osc}}(t) + N_{\text{osc}}(t)} = \cos(\Delta M\, t) \tag{3.7}$$

An oscillation measurement requires a measurement of the B-meson's flavor at a time t=0 and at a later time $t > 0$. The $\Upsilon(4S)$ decays into an entangled state of $B^0\,\overline{B}^0$. The time evolution leaves the entangled state unchanged except for a trivial phase. The entangled state maintains its $B^0\,\overline{B}^0$ content.

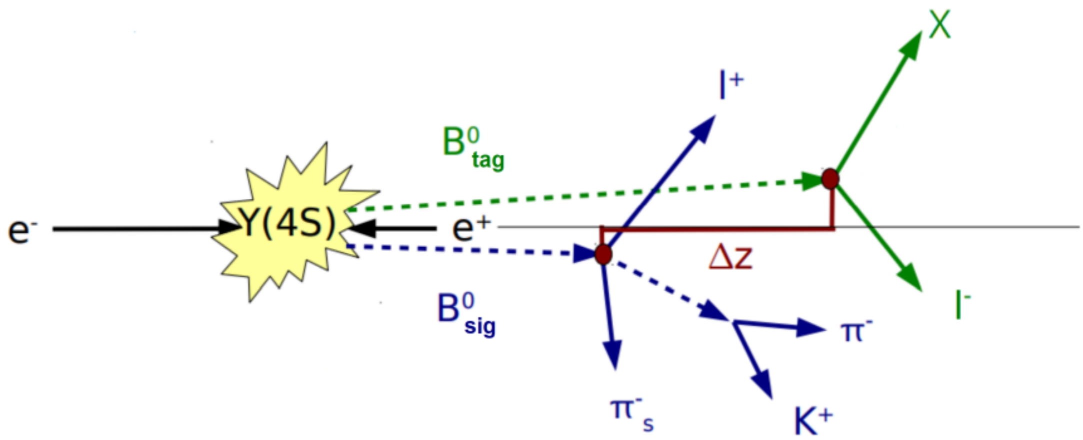


Figure 3.5: Decay of the B mesons as observed in the laboratory (from [RW21]).

This means once the first B decays, we know that the other B must be a pure state of opposite flavor at that point in time. We set the decay time of the first B meson to be t=0. The remaining B is no longer in an entangled state and evolves in time according to the Hamiltonian matrix in 3.1. When the second B meson decays the flavor of the second B meson at decay time can be deduced from the decay particles. The relevant time t is the difference between the two B meson decay times. The difference in decay time can be easily observed at BELLE-II due to the boost to the $\Upsilon(4S)$ and thus to the B mesons. To a very good approximation the B mesons travel along z direction (in beam direction) with known momentum. The time difference is directly related to the difference in z position of the decay points: $\Delta z = \beta\gamma c \Delta t$. See also Fig. 3.5.

3.4.1 $\mathcal{A}$ measurement details

There are several aspects relevant for measuring the asymmetry $\mathcal{A}$ due to imperfections in the measurement process, such as background contamination, flaovr mis-tags and general measurement uncertainties.

Asymmetry measurements with non-zero mis-tag probability The measurement of the B flavors may be associated with a probability of mis-identification (mis-tag). Let us call the probability to mis-identify an oscillated state as not-oscillated (or vice versa) P_{wrong} and assume that this probability is independent of Δt. The number of observed oscillated states depends on the true oscillated states (N_{osc}) andf the true non-oscillated states (Nnosc).

$$N_{\text{osc}}^{\text{meas}} = (1 - P_{\text{wrong}})N_{\text{osc}} + P_{\text{wrong}}N_{\text{not-osc}} \tag{3.8}$$

It is quite straightforward to calculate the measured asymmetry function.

$$\mathcal{A}_{\text{meas}} = \frac{N_{\text{not-osc}}^{\text{meas}}(t) - N_{\text{osc}}^{\text{meas}}(t)}{N_{\text{not-osc}}^{\text{meas}}(t) + N_{\text{osc}}^{\text{meas}}(t)} = (1 - 2P_{\text{wrong}})\ \cos(\Delta M\ t) \tag{3.9}$$

The term $(1 - 2P_{\text{wrong}})$ is called the dilution $\mathcal{D}$ (see also [Bev+14; Kou+19]). There is also the possibility of other processes than $B^0\,\overline{B}^0$ to be mis-identified as signal. A background that is observed evenly in N_{osc} and $N_{\text{not-osc}}$ leads to a further reduction of the oscillation amplitude.

Influence of charged-B misidentification on asymmetry measurements There are charged B decays that are semi-leptonic and involve excited D mesons that may end up in a final state similar to the one we consider. If a charged track is missed or simply not considered, we might accept a charged B decay as a neutral B decay. Because charged B pairs do not oscillate due to conservation of charge, they usually contribute to $N_{\text{not-osc}}$ only. Any charged B mesons that would pass our selection criteria for the neutral decay chain would add an offset to the asymmetry measurement. Assuming that the mis-identification probability of charged Bs as neutral B mesons is independent of Δt, the effect can be calculated in a similar mannner as in the section before. We obtain roughly an offset that becomes worse with increasing Δt because charged Bs are slightly more long-lived than neutral B mesons. This offset should be avoided by finding selection criteria that veto most candidates from charged B decays.

Influence of Δt mis-measurements on $\mathcal{A}_{\text{meas}}$ There is of course a measurement uncertainty on $|\Delta t|$. A B meson pair with a small $|\Delta t|$ may be measured to have a larger $|\Delta t|$ or vice versa, but the mis-measurement effect is far from being symmetric. There are much fewer observations at larger $|\Delta t|$ due to the exponential decay, so events "migrating" from small $|\Delta t|$ where there are many events to larger $|\Delta t|$ where there are fewer can have a significant impact by skewing the measurement at larger values. We advise to exclude events with a $|\Delta t|$ larger than roughly 8 to 12 picoseconds from the fit to minimize the influence of $|\Delta t|$ mis-measurements.

CHAPTER 4

Tools

4.1 The Event Display

The BELLE-II event display provides a graphical representation of collision events. The user interface is shown in Fig. 4.1. The event display allows the user to scan files containing several collision events.

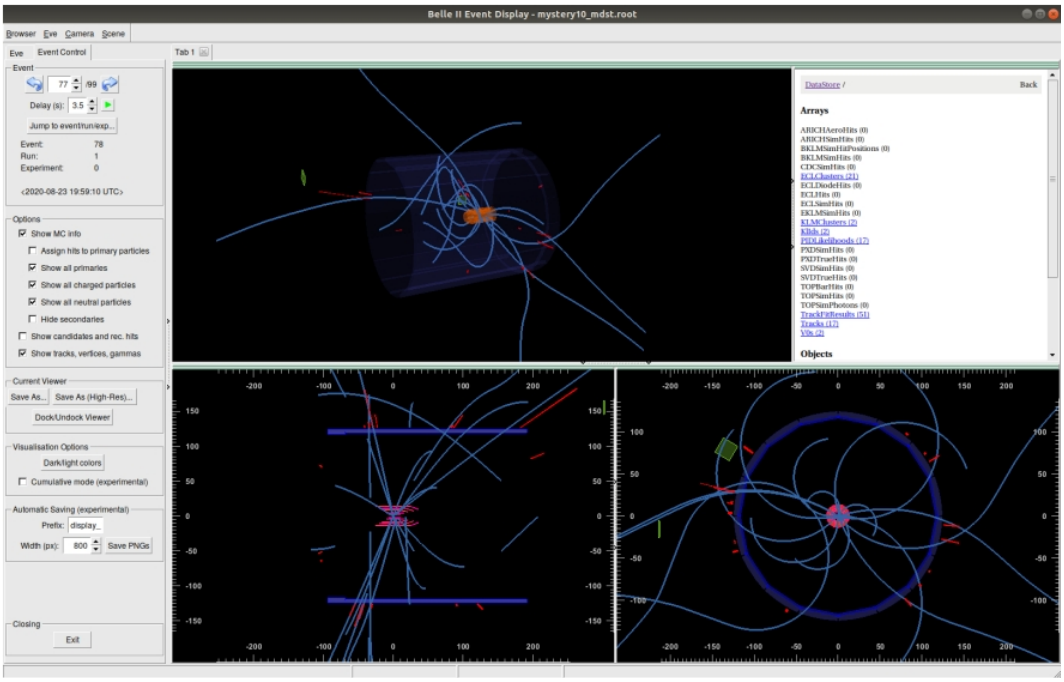


Figure 4.1: Event display of the BELLE-II experiment.

In the top left corner of the display the current event number is shown together with buttons to move forward and backward in the event file. The center of the diplays shows three graphical representations of the event: a 3D view, a side view and a view perpendicular to the beam direction. Shown are tracks, the energy deposition in the electromagnetic calorimeter (shown as a red bar) and hits in the KLM detector (in green).

Particles may be identified based on their representations in the display. For example electrons loose all their energy in the electromagnetic calorimeter (Fig. 4.2), whereas muons (Fig. 4.3) traverse the complete system of electromagnetic calorimeter and KLM. Hadrons enter the KLM and leave hits there. Please note that the displayed trajectory extends to all detector components, irrespective of the actual flight path. To estimate the flight path from the displayed trajectory one must pay attention to the hits associated to the track. For example in figure 4.3 the trajectories are accompanied by hits in the KLM detector, indicating that the particle went through the KLM. In contrast to that you can see that the two trajectories in figure 4.2 are not accompanied by KLM hits indicating that the particles were stopped before entering the KLM.

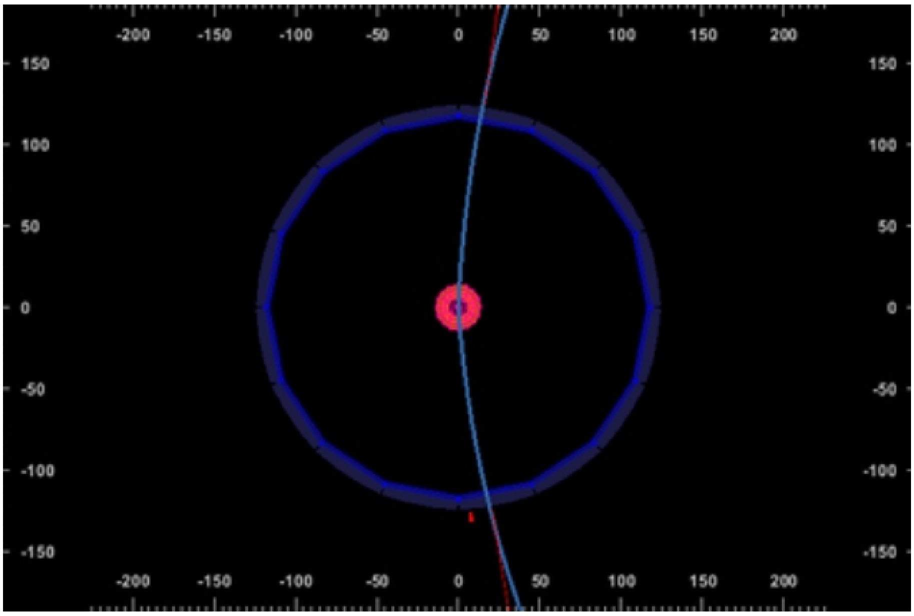


Figure 4.2: Example for an event with an electron positron pair (Bhabha scattering event). The energy deposition in the electromagnetic calorimeter is shown as a red bar.

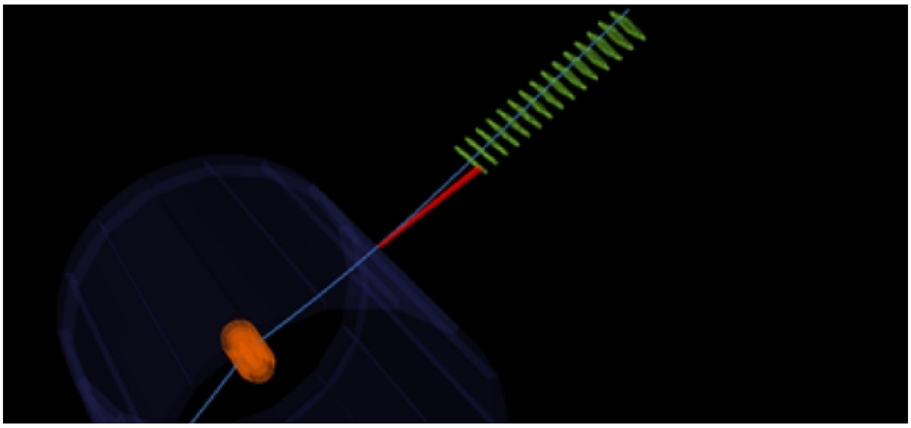


Figure 4.3: Example of a track with hits in the KLM detector (in green). This is almost certainly a muon.

4.1.1 Starting the event display

The event display is a part of the BELLE-II experiment's software. It depends on other packages and thus runs only on our BELLE cluster. You will need your Uni Bonn student ID and password to log in to the cluster. Once online, start a terminal window and activate the BELLE-II software with the command
`source /cvmfs/belle.cern.ch/tools/b2setup release-XX-YY-ZZ`
where release-XX-YY-ZZ is the current release number you will get from the tutor (currently it is release-06-00-00). You start the event display with the command
`b2display filename`
where filename is the name of the file you want to inspect.

4.2 Working with python

In the following we discuss features of the python language and of the modules necessary for this lab exercise: pandas, numpy, jupyter and matplotlib.

4.2.1 Python basics

Python is one of the two major programming languages in science, the other being C++ (or C). Python code is being interpreted and thus slower than C++. Extensive calculations in python are therefore delegated to external modules such as pandas and numpy. Python being interpreted allows the usage in an interactive mode where chunks of python code are executed within a browser. Calculation results including graphics are shown in the same window along side the source code. This modus of operation is provided in python by the jupyter package and is used throughout this lab exercise. We provide a set of examples in the form of juypter notebooks that can be found on eCampus [ECa].

Python versions: Some text books use python2 as a reference, however that version became obsolete around 2020 and we will only use python3[1] Some systems map the command python to (old) python2 versions and use the `python3` command for newer versions. To find out your python version run `python --version`. Please note that, except for print statements, most python3 code is downward compatible[2].

Data types: Python is an interpreted language with basic data types like integer, float, strings. It is an object oriented language that allows the definition of classes. Each class acts as a (new) data type.

Print statements: The command print followed by a list of arguments in brackets, example
```
print("hello world", "2+2=", 2+2)
```
The argument list in this example consists of a string literal in double quotes (single quotes are also allowed), followed by the expression `2+2` that is evaluated as integer 4. It gives as output `hello world 2+2= 4`.

Assignments have the syntax
```
variable_name = expression
```
The expression on the right is evaluated at run time and the value along with the variable name and data type is stored in a table of objects. Should the variable_name already be entered, its entry is replaced by the new one. The data type is determined directly from the expression, no type declaration is necessary in python. To find out the data type and value for a given entry, use for example a print statement.
```
my_var = 3.5
print(type(my_var), my_var)
```
The print command yields **float 3.5**.

Functions: A function in python calculates a value based on a list of arguments. The argument list may have any length including zero. A function is called by invoking the name of the function

[1] Python3 is any python version number ≥3.0.0. We currently (February 2022) use 3.9.2.
[2] Python2 expects print statements without brackets `()` around the print arguments.

followed by the argument list in brackets. In case of a zero-length argument list you still have to provide the brackets to invoke the function. You can define your own function using the following syntax:

```
def myfunction(x,y):
  return x+y
```

This function with name `myfunction` has two arguments. It adds the arguments together and returns the result.

Block indentation: Please note the indentation after the function header in the previous example. The indentation indicates a block of code. This block ends when the indentation returns to the previous level. This is an important general feature in python. A block of code starts where the indentation level begins and ends where it returns to normal. Block indentation occurs with function definitions but also with `for` and `while` loops and `if` statements. As an example look at this if clause

```
if a>2:
  print("a is larger than 2:  a=",a)
  a = a+1
```

Compare this to:

```
if a>2:
  print("a is larger than 2:  a=",a)
a = a+1
```

In the former case a is incremented only if a>2, in the latter it is incremented independent of the value of a.

Import: There is a large number of python packages that you can use in your work with python: numpy, pandas, matplotlib are good examples. In order use such packages in your code you need to provide an `import` statement. The simplest import statement is `import packagename`. In the following we use pandas as an example:

```
import pandas
```

After including this line in your python code you have access to all commands and data types defined within the pandas package. You can access them with the syntax `package_name.pandas_variable`. For example `pandas.read_csv` invokes the read_csv function of the pandas package. This will invoke a large number of occurrences of the word `pandas` in your source code. Since pandas is a long word, one often abbreviates it to `pd`. The import statement

```
import pandas as pd
```

imports the package under the abbreviated name. Other common abbreviations: numpy → np, and matplotlib.pyplot → plt. This lab requires the following packages to run properly: pandas, numpy, matplotlib, and jupyter. If you want to import just a single function from a package, you can use the `from` syntax:

```
from math import sin,cos,sqrt
```

This imports three functions: sine, cosine and square root from the math package. Once imported you can use the function names without prefixes. Example: `a=sqrt(2)`.

4.2.2 Pandas

The pandas package provides a simple and elegant solution to the problem of handling large data samples that are stored in a simple format with rows and columns, the dataframe. Values of belonging to one variable are stored in a column. Each rows represents one element of your data. This examples has 3 elements (particles) with 4 variables each: energy and three momentum components.

index	energy	p_x	p_y	p_z
particle0	3.5	1.8	1.98	-2.1
particle1	1.5	0.8	0.98	-1.1
particle2	0.8	0.5	-0.4	0.1

Such a data structure can be created as a pandas dataframe from scratch with this syntax:

```
import pandas as pd
df = pd.dataframe( {'e':[3.5, 1.5, 0.8], 'px':[1.8,0.8,0.5], 'py':[1.98,0.98,-0.4],
'pz':[-2.1,-1.1,0.1] })
print(df)
```

Here, the dataframe is created from a dictionary with column names as keys followed the list of values for each column. We will create dataframes for the lab's data by converting a text file in csv format[3] into a dataframe using an assignment `df = pd.read_csv(file_name)` storing the dataframe under variable name `df`. If you want to access a single dataframe column, say px, use the column name `df['px']`. Alternatively you can also use the "dot"-syntax. `df.px`
`print(df['px'])` or `print(df.px)` would print the column values.

Creating a new column Tasks like creating new data columns are very easy with pandas because pandas handles data in an dynamic format. A new column is added to the dataframe with this syntax `df['new_column_name']= expression`. Example:

```
df['C']= 2 * (df['A'] + df['B'])
```

Alternatively one can also use `df.A` instead of `df['A']` in the expression. The syntax covers all basic algebraic operations, even squares (df.A**2) or square roots (df.A**0.5). Another way of defining a new column uses the `eval` function, example:

```
df['C']= df.eval('2 * (A + B)')
```

There are also more sophisticated ways of filling a new column but the simple one will suffice here.

Subframes Creating a subset of the data from a dataframe creates a subframe, also called a slice. There are different basic kinds of slices, slices on a subset of columns:
`sub_df1 = df [ ['A','B'] ]`, this selects a sub frame with only columns A and B,
`sub_df2 = df [ df['A']>0.5 ]`, this selects all columns but only rows with A>0.5
Please note that slicing and new column creation are operations that do not commute. A newly added column to a dataframe will not be part of any previously derived slice. We suggest you always add variables to the full data frame and then re-create the subframe.

Interface to matplotlib Creating a histogram to plot all values of a dataframe column is simple, use

```
df['A'].hist(bins=30)
```

[3] csv stands for comma separated values.

It is also possible to create scatter plots and similar. Just look at the example jupyter scripts that we provide.

Converting a pandas column or a dataframe to a numpy array: Use the function `to_numpy`. The assignment
`ar = df.to_numpy()`
creates a numpy array and stores it under variable name `ar`. As single column is stored via
`column = df['column_name'].to_numpy()`

4.2.3 Numpy

Numpy is a powerful and fast tool for numeric calculations in python. Anything that cannot be calculated in pandas directly can be done in numpy. We will mostly use numpy implicitly, as pandas columns translate into one-dimensional numpy arrays and matplotlib often uses numpy arrays as input. More information on numpy can be found in the general python literature and in the example jupyter notebook on eCampus **NumpyExample.ipynb**.

4.2.4 Matplotlib

is a powerful tool to create many different plots, like histograms or scatterplots. Please have a look at the matplotlib web site for examples and have a look at the matplotlib example on eCampus and Fig. 4.4

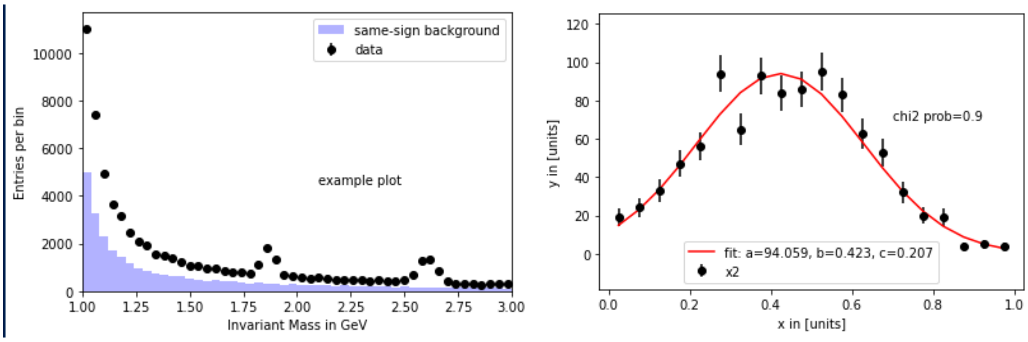


Figure 4.4: Two plot from the jupyter examples on eCampus [ECa]. Both plots were created with matplotlib.

4.2.5 Using python in the lab

You are encouraged to install python and VSCode on your computer or laptop and try out the python examples before the start of the lab, ecspecially if you have not worked with python before. However, there is a desktop computer available in the laboratory room to which you will have access during the execution of the lab. The computer belongs to the Bonn Analysis Facility (BAF). The particle physics groups of the physics institute (PI) share this common computing facility. The operating system is Linux. Python3 is already installed on the desktop computer with all necessary packages (jupyter, pandas, numpy and matplotlib). To run python, open a terminal and type `python3`. However there is a much more elegant way of interacting with python via an integrated development environment

(IDE). We will use Visual Studio Code (VS code) throughout this exercise, as it is a powerful IDE that is available on lLnux, Windows and macOS as a free software. You could use an alternative IDE but we advise against it, simply because your tutor can help you with any issues regarding VSCode, but most likely not so for other IDEs. VSCode supports syntax highlighting, file management, etc. It also allows the execution of jupyter scripts within the IDE. To run VSCode on the cluster simply type `code`.

4.2.6 Visual Studio Code (VSCode)

VSCode is the editor of choice for the exercise. As a preparation for the lab we advise you to go through the example jupyter scripts from the eCampus website of E215 and to run them with VSCode. The jupyter scripts from eCampus should also run on your laptop, once you have python and VSCode installed. Instructions on how to install python on your computer can be easily obtained from the internet. Same holds for installing VSCode[4].

Download the juyper scripts (ending .ipynb) to a local directory that you please create for the exercise. Here we name it `E215`. Open a terminal and type `code`. A window opens like in Fig. 4.5. You have typically three subdivisions. The left margin displays all files in the selected directory/project. The upper main area is the file content. Several files may be open at the same time. You select one of the open files by selecting its tab. Open a new file by double clicking on the file list. You can hide/display the file list by clicking on the symbol above the magnifying glass in the left margin. At the bottom you have a terminal open to run shell commands if necessary. You can close the terminal anytime and reopen it in the top menu, item `Terminal/New Terminal`.

Open a new project. Open a new project/folder by selecting `File/Open Folder` from the top menu. Then navigate to the directory E215 and select the folder. Each VSCode window can display only one project. However you can open several windows with different projects at once.

Open a Jupyter notebook. Make sure that the filelist is displayed on the left and select the file PhysicsExample.ipynb (you get it from the eCampus web page). If you run VSCode for the first time, it might invite you to install python-specific packages. Just click ok. It will also display a list of python versions on your system and asks you to select the one for the jupyter script. On the BAF select the python version `/usr/local/python3` or a similar name. You can see the jupyter script displayed in the main area divided into cells, blocks of python code. Each cell can be run on its own. The results of the run are displayed below the cell as either text or as a plot in case of matplotlib commands. As cells may be executed in any order and possibly several times, there is a clean way of executing a notebook: at the top of the browser window click on `Clear output`, `Restart` and `Run All`.

Creating a new jupyter notebook. Use the key combination `Ctrl-Shift-p` that opens an interactive command window. Type in jupyter and then select `create new blank jupyter notebook` from the list of suggested commands. There is one empty cell. Select it and start typing code. You can add cells by clicking on one of the **+** buttons. Fig. 4.6 shows the jupyter browser within VSCode and the first cell of the notebook along with the text output created by the cell's execution.

[4] If in doubt search the internet for `install visual studio code` and `install python`.

Running a single cell. Once you have entered code into a cell, clicking on the run symbol in the cells top left corner. Make sure that the cell you want to run is activated. Click into the cell to make sure it is activated. Please note that the jupyter browser stores a table of objects and updates it with any objects created/modified during cell execution. As a consequence that table of objects depends on the order in which cells have been executed and how many times they were executed. While developing a jupyter notebook one normally creates one cell after the other executing the cells on the way, sometimes several times until the cell in question works as expected.

Obtaining final (reliable) results with jupyter notebooks. Once all cells work, final results can be obtained by the restarting the jupyter kernel, thus clearing the table of objects. Use the combination `Clear output`, `Restart` and `Run All`.

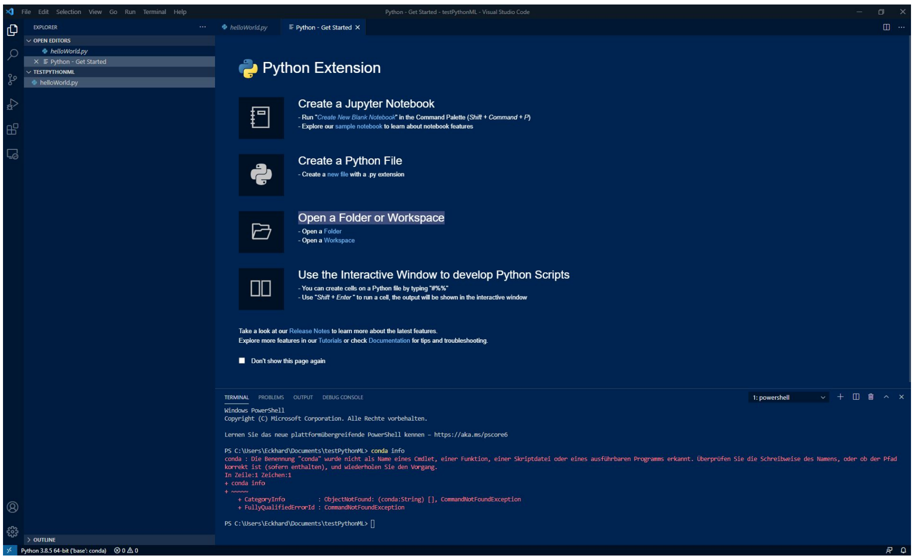


Figure 4.5: Visual Studio Code window.

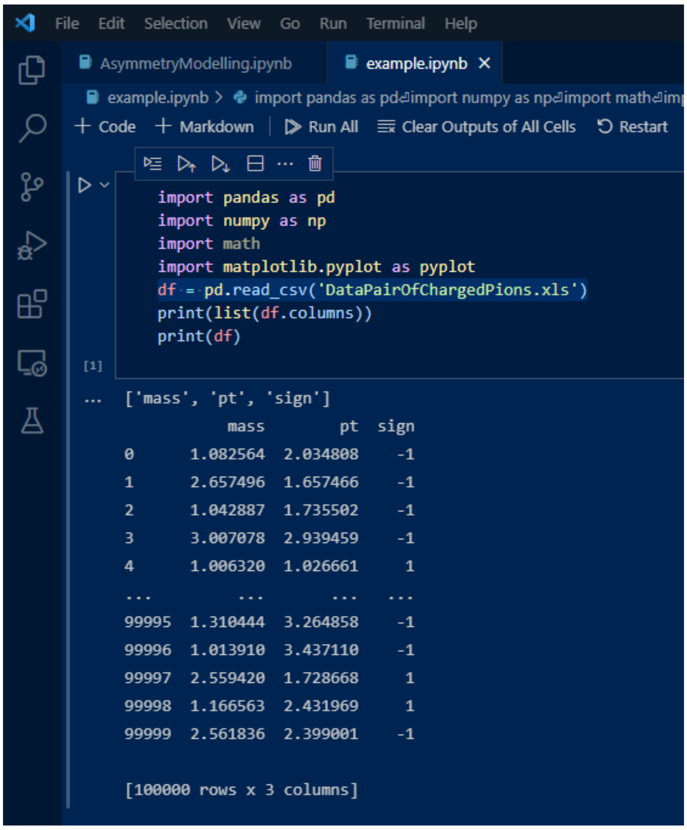


Figure 4.6: The jupyter browser within VSCode and the first cell of the example notebook along with the text output created by the cell's execution. The execute-cell button is located to the left of the activated cell.

4.3 Useful statistics concepts

Due to the quantum mechanical nature of particle collisions and decays and the necessity for large data sets, statistics plays an important part in experimental particle physics.

4.3.1 Signal region and purity

Consider the example histogram of the invariant mass of Kaon pion pairs in Fig. 4.7. You observe a mass peak from the decay $D^0 \to K\pi$ a above a linear background of random Kaon-pion combinations. Selecting a region around that peak as indicated in the Figure will contain most signal events in that region except for tail events while rejecting most background events. We call the acepted events signal region events or signal candidates. Obvioulsy the signal region will also contain besides signal also background events. A quantity of interest is the purity defined as

$$\alpha_{\text{purity}} = N_{\text{Sig.}}/N_{\text{Cand.}} \tag{4.1}$$

the ratio of signal events to the number of candidate events. Events in our signal region we call signal candidates or simply candidates (also: signal region events). Candidates fall into the categories signal

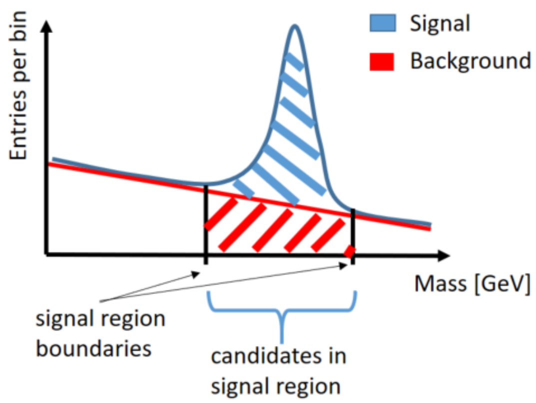


Figure 4.7: Example of a mass peak with candidates in a signal window with signal and background contributions inside the window.

or background:

$$N_{\text{Cand.}} = N_{\text{Sig.}} + N_{\text{Bgd.}}$$

Please note that the number of candidates can be trivially obtained by counting events. To obtain the purity it is sufficient to either measure the background or signal number, For example, if we determined $N_{\text{Bgd.}}$ either from a χ^2-fit or by estimating the background from background-only sidebands next to the signal region, we have

$$\alpha_{\text{purity}} = \frac{N_{\text{Sig.}}}{N_{\text{Cand.}}} = 1 - \frac{N_{\text{Bgd.}}}{N_{\text{Cand.}}} \tag{4.2}$$

The error on the background estimation may be determined from the errors of the χ^2-fit or the counting error of the sideband regions. The binomial error for the α_{purity} is given by

$$\sigma(\alpha_{\text{purity}}) = \sqrt{\alpha_{\text{purity}}(1-\alpha_{\text{purity}})/N_{\text{cand.}}} \tag{4.3}$$

This error is based on the standard deviation of the binomial distribution plus Gaussian error propagation [Bar94].

4.3.2 Measuring average values in the presence of background

The average of a physical quantity has an associated error that depends on the error of the individual measurements. We denote the physical quantity as x and its average as

$$\langle x \rangle = \frac{1}{n}\sum_{i=1}^{n} x_i \tag{4.4}$$

where x_i are the individual measurements of quantity x. The question if the errors on individual measurements are independent from each other is important for the correct determination of the error on the average. In the following we assume that errors on individual measurements x_i are uncorrelated. We denote the error on a quantity x as $\sigma(x)$ or σ_x. Gauss law of error propagation for $\sigma(F)$ depending on two measurements x_1, x_2 in the absence of correlations between errors is

$$\sigma(F(x_1,x_2)) = \sqrt{\left(\frac{\partial F}{\partial x_1}\sigma(x_1)\right)^2 + \left(\frac{\partial F}{\partial x_2}\sigma(x_2)\right)^2} \tag{4.5}$$

The error on the average $\langle x \rangle$ of a physical quantity x is

$$\sigma(\langle x \rangle) = \sqrt{\sum_{i=1}^{n}\left(\frac{\sigma(x_i)}{n}\right)^2} \tag{4.6}$$

If individual errors are the same, $\sigma(x_i) = \sigma_x = const$, this simplifies to

$$\sigma(\langle x \rangle) = \sqrt{\sum_{i=1}^{n}\left(\frac{\sigma_x}{n}\right)^2} = \sqrt{n\left(\frac{\sigma_x}{n}\right)^2} = \frac{\sigma_x}{\sqrt{n}} \tag{4.7}$$

The error on the average is smaller than the individual error, that is why we compute averages! Often the standard deviation is taken as individual error and variations in individual errors are neglected. In this case the error on the average is standard deviation devided by $\sqrt{n}$.

The situation becomes more complex if the data sample for which we want to determine the average is contaminated by background with a potentially different average value. Simply ignoring the background to obtain the desired average would skew our result if the average of the backgroud is shifted with respect to the signal average. The observed average of quantity x, calculated over all candidates, is related to the averages of signal and background as follows.

$$\langle x_{\text{obs}} \rangle = \alpha_{\text{purity}} \left\langle x_{\text{signal}} \right\rangle + (1-\alpha_{\text{purity}}) \left\langle x_{\text{background}} \right\rangle, \tag{4.8}$$

where α_{purity} is the purity. The average value of x for background events can easily be obtained from background-only data samples, for example from sidebands near a signal mass peak. The error on the average for the signal-only data sample can be obtained by solving eqn. 4.8 for $\langle x_{signal} \rangle$ and applying error propagation.

Statistical and systematic uncertainties of an average value As outlined in the previous section the statistical error on the average goes to zero with increasing number of measurements. However a limit of zero measurement error is not a realistic expectation, as the following example shows. Consider measuring the length of an object with a ruler. You perform a single measurement and obtain a reading error for it. Then you repeat the readings and form the average. It is safe to assume that the readings are done independently. Errors on the readings are uncorrelated and thus the gaussian law of error propagation without correlations apply. The error on the average is $\sigma_{average} = \sigma_{read}/\sqrt{n}$. This would be the statistical error. It is typical for statistical errors to approach zero as the number of measurements goes to infinity.

Other effects might also play a role that give rise to systematic uncertainties not covered by the statistical ones. One obvious systematic effect is caused by any imperfection in the ruler. A ruler is subject to ageing effects. A wooden ruler for example undergoes changes in dryness of the wood, thus changing the physical size of the ruler. It might shrink with time. This may be only a small effect but it is an effect that would affect every measurement in the same way. This is a fully correlated error as far as the error propagation law is concerned. Its effect on the final result is therefore not affected by the averaging process and would become the dominant error as the reading error's contribution goes to zero due to the averaging.

One way to obtain an estimate on the systematic effect due to ruler imperfections would be a measurement campaign with several rulers like the one in table 4.1. The reading error is taken to be 1.0 mm. For each ruler one would derive an average value with relatively small statistical uncertainties.

Ruler type	M1	M2	M3	M4	M5	Average	std.dev	Error on Average
wooden_1	12.4	12.5	12.4	12.4	12.6	12.46	0.080	0.044
wooden_2	12.3	12.3	12.4	12.2	12.4	12.32	0.075	0.044
plastic_1	12.5	12.3	12.3	12.4	12.4	12.38	0.075	0.044
plastic_2	12.3	12.2	12.1	12.3	12.4	12.26	0.102	0.044

Table 4.1: Results for different measurement campaigns in cm. Averages, standard deviations and errors on averages are also given.

The uncertainty on the average is either reading error divided by $\sqrt{5}$ or measurement colum standard deviation divided by $\sqrt{5}$. There is little difference between the two variants. The total statistical error is then $\sigma_{stat} = 0.02$ cm and the final result is the average over all measurements: 12.36 cm.

The statematic effects due to ruler imperfections could then be derived by comparing the averages for different ruler types, for example by taking the standard deviation of the averages for different rulers. For example the different ruler results exhibit a standard deviation of $np.std$([12.46, 12.32, 12.38, 12.26]) = 0.074 cm which is more than we would expect due to statistical fluctuations. A conservative estimate of the systematic uncertainty would be the variation due to ruler type ($\sigma_{syst} = 0.074$ cm). If we assume no correlations between ruler types we could quote a reduced systematic uncertainty on the total average by applying error propagation to the overall average $\sigma_{syst} = 0.074$ cm/sqrt(4)=0.037 cm.

It is then customary (at least in particle physics) to quote uncertainties split into statistical and systematic uncertainties: We measured the length of the object to be

$$d = 12.36\ \text{cm} \pm 0.02\ \text{cm (stat.)} \pm 0.04\ \text{cm (syst.)}$$

We encourage you to quote statistical and systematic errors like this wherever necessary. A complete analysis of systematic effects is often beyond the scope of an academic course such as this master level laboratory. However this specific lab experiment provides high statistics data and statistical errors may be so small that systematic uncertainties might become the dominant source of uncertainty.

4.3.3 χ^2 fits

To fit measured data to a parameterized function is a frequently occuring problem in physics, often tackled by χ^2-fits (chi-square fits). The data M_i, measured at points x_i, shall be described by a function $f(x, p_1, p_2..., p_N)$, where f depends on variable x and parameters p_1 to p_N. The goal of the fit is to find the optimimal set of parameters by minimizing a quantity known as χ^2. The quantity χ^2 is defined for this purpose as:

$$\chi^2 = \sum_i \frac{[M_i - f(x_i, p_1, .., p_N)]^2}{\sigma_i^2} \tag{4.9}$$

In the case of constant errors, $\sigma_i = const$, this reduces to the least-square method. If the measurements stem from a counting experiment one applies the usual poissonian error $\sigma_{\text{count}} = \sqrt{n_{\text{count}}}$.

The optimized parameters are obtained when the χ^2 is at minimum. Detailed discussion can be found in Barlow [Bar94]. One interesting feature of χ^2 fits is that the χ^2 value at the minimum is a measure for the goodness of fit. High χ^2 means that the measurement is not well described by the fit function f. The good χ^2 is roughly as large as the number of degrees of freedom (ndf), e.g. $\chi^2/\text{ndf} \approx 1$. The ndf is equal to the number of measurements used in the fit minus the number of fitted parameters. A very small χ^2 value is also problematic as it usually indicates problems with the errors or with the measurement procedure. χ^2-fits are implemented in python. For an example see our web-site of python examples [ECa] and section 4.2.

Statistical and systematic errors in a χ^2 fit Another appealing feature of χ^2-fits is the straightforward approach to obtain uncertainties for the fitted function parameters and any software implementation of χ^2 fits provides them. These parameter errors are considered to be due to statistical fluctuations in the data sample and are thus deemed statistical uncertainties. They rely on the correctness of the model, i.e. the correct choice of parameterized function.

An important and often dominant systematic uncertainty in χ^2 fits comes from the model dependency. As an example assume that we want to fit a distribution that follows a quadratic behavior but we fit a linear function to it. The results for the constant and linear coefficient will be skewed due to insufficient model choice. As a consequence the fit works best in a region where the data follows the linear model and the fit starts to fail when the quadratic term becomes important. It is thus prudent to choose a sufficient fit model and to apply it only to regions where the model is valid. Systematic uncertainties due to the restricted range in which the fit model applies can often be estimated by varying the fit region.

4.4 Structure of the data sample

This dataframe contain all data used for the decay chain reconstruction and the oscillation measurement. Quintuples of tracks are selected from an event under the following requirements.

- Two tracks must be identified as leptons (called signal and tag lepton).
- The kaon and pion tracks must have measurements in the Cherenkov detectors that are consistent with the particle hypothesis.
- The kaon-pion pair fulfills loose restrictions on the invariant mass.
- The kaon-pion-slowpion triplet fulfills loose restrictions on M_{diff}.

As there are no restictions on the lepton properties, each quintuple appears twice in the dataframe with signal and tag lepton interchanged.
Signal and tag lepton properties. Signal lepton quantities start with `Ls_` , tag leptons with prefix `Lt_`.

- **Lx_p** Momentum in the lab frame in GeV
- **Lx_px** Momentum x-component in the lab frame in GeV
- **Lx_py** Momentum y-component in the lab frame in GeV
- **Lx_pz** Momentum z-component in the lab frame in GeV
- **Lx_E** Energy in the lab frame in GeV
- **Lx_pst** Momentum in the $\Upsilon(4S)$ frame in GeV
- **Lx_z0** z-coordinate of the track origin in cm.
- **Lx_d0** Closest distance of the track from beam axis in the plane transverse to the beams in cm
- **Lx_t0** B lifetime in B rest frame as estimated from the lepton z-position. Unit = pico seconds
- **Lx_typ** Lepton type: 11=electron, 13 = muon
- **Lx_nnear** Number of tracks found with a similar z-position as this lepton
- **Lx_q** Electric charge

D^{*+} decay chain particles. Properties of the three tracks from the decays $D^{*+} \rightarrow D^0\pi_{slow}, D^0 \rightarrow K^-\pi^+$. The kaon quantities have a prefix k_, the pion (from D^0 decay) pi_, the pion from D* decay pis_.

- **k_p** Kaon momentum in the lab frame in GeV
- **k_px** Kaon momentum x-component in the lab frame in GeV

- **k_py** Kaon momentum y-component in the lab frame in GeV
- **k_pz** Kaon momentum z-component in the lab frame in GeV
- **k_E** Kaon energy in the lab frame in GeV
- **k_theta** Kaon polar angle in the lab frame in radians
- **k_phi** Kaon azimuthal angle in the lab frame in radians
- **k_z0** Kaon z-coordinate of the track origin in cm.
- **k_q** Kaon electric charge
- **pi_...** Pion properties
- **...**
- **pis_...** Properties of the pion from D^{*+} decay (=slow pion)
- **...**

Other observables

- **nlep** Total number of leptons found
- **deltat** Absolute value of time difference between signal- and tag-lepton in picoseconds

Acknowledgements

I would like to thank Christian Wessel and Jannick Stammnitz for valuable contributions to the event display task. Stefan Duell provided the raw BELLE-II data on which the main task is based and Sebastian Müller tested the oscillation frequency part. Both deserve much praise for their work. I would also like to thank Ralf Farkas and Max Mucha who read and criticised different versions of the laboratory script. This script was set in Latex using Ian Brock's excellent Uni Bonn Thesis package [Bro].

Eckhard von Törne, February 2022

Bibliography

[EvT] E.v.Toerne, *Particle anti-particle oscillations at BELLE-II*, ECampus page for E215 experiment (cit. on p. 1).

[Gri08] D. Griffith, *Introduction to Elementary Particles*, 2nd ed., Wiley, 2008 (cit. on p. 1).

[Tho13] M. Thomson, *Modern Particle Physics*, 1st ed., Cambridge U. Press, 2013 (cit. on pp. 1, 11).

[Gro] P. D. Group, *Particle properties and review articles*, URL: https://pdglive.lbl.gov (cit. on p. 2).

[Ohn+13] Y. Ohnishi et al., *Accelerator design at SuperKEKB*, PTEP **2013** (2013) 03A011 (cit. on p. 7).

[web] S. website, URL: https://www-superkekb.kek.jp/ (cit. on p. 7).

[Par] P. Z. et al. (Particle Data Group), *Measurement of particle momenta in a uniform magnetic field, contribution to the particle data booklet section 35.12*, URL: https://pdg.lbl.gov/2021/reviews/rpp2020-rev-particle-detectors-accel.pdf (cit. on p. 8).

[Kou+19] E. Kou et al., *The Belle II Physics Book*, Progress of Theoretical and Experimental Physics **2019** (2019), URL: http://dx.doi.org/10.1093/ptep/ptz106 (cit. on pp. 9, 13).

[MT] D. J. Miller and D. Tovey, *Kinematics, contribution to the particle data booklet*, URL: https://pdg.lbl.gov/2020/reviews/rpp2020-rev-kinematics.pdf (cit. on p. 8).

[CG09] R. N. Cahn and G. Goldhaber, *The Experimental Foundations of Particle Physics*, 2nd ed., Cambridge U. Press, 2009 (cit. on p. 10).

[Alb+87] H. Albrecht et al., *Observation of B0 - anti-B0 Mixing*, (1987), ed. by J. Tran Thanh Van (cit. on p. 10).

[Sch] O. Schneider, *B mixing, contribution to the particle data booklet*, URL: https://pdg.lbl.gov/2021/reviews/rpp2020-rev-b-bar-mixing.pdf (cit. on p. 10).

[Kuh12] T. Kuhr, *Flavor Physics at the Tevatron*, 1st ed., Springer, 2012 (cit. on p. 11).

[RW21] A. Reuss and H. Wetter, *private communication*, 2021 (cit. on p. 12).

[Bev+14] A. J. Bevan et al., *The Physics of the B Factories*, (2014), arXiv: 1406.6311 [hep-ex] (cit. on p. 13).

[ECa] ECampus, *Python example code can be found on the ECampus page for lab unit E215* (cit. on pp. 18, 21, 28).

[Bar94] R. Barlow, *Statistics: A Guide to the Use of Statistical Methods in the Physical Sciences*, Wiley, 1994 (cit. on pp. 26, 28).

[Bro] I. Brock, *Uni Bonn thesis latex package*, URL: `https://www.pi.uni-bonn.de/lehre/uni-bonn-thesis` (cit. on p. 31).