A new Python-based Lattice Development Tool and its Applications at BESSY II

Author

Felix Andreas

Supervisors

Prof. Dr. Andreas Jankowiak

Dr. Paul Goslawski

A thesis submitted in partial fulfillment of the requirements for the degree of Master of Science (M.Sc)

in the

Department of Physics

Faculty of Mathematics and Natural Sciences

Humboldt-Universität zu Berlin

December 20, 2021

Abstract

Due to the BESSY-VSR upgrade and the design of the next synchrotron radiation facility BESSY III, there are several lattice development tasks at the Helmholtz-Zentrum Berlin (HZB). Within the last several years, there was a significant shift in the physics community from standalone plotting programs, optimization routines, and numerical libraries towards the Python programming language, which absorbed these tools into different packages. At HZB, Python is the predominant programming language used to set up the execution of simulations and to post-process their results. The issue with existing and mature particle accelerator simulation codes is that it is not trivial to integrate them into a typical Python workflow. Many physicists have written several wrapper scripts around the existing simulation codes to leverage the power of Python's rich ecosystem of scientific tools. The issue with these wrappers is that there are often not reusable because they are very specific to a particular task and often rely on string manipulation of the lattice files or run files, which can be very error-prone and is computationally inefficient. Furthermore, these wrapper scripts only give access to the simulation results and not to the underlying internal models of the simulation codes, like the magnetic lattice or information about individual magnets. Therefore it was decided to develop a new Python package that generates an accelerator model from a given lattice file. This model can be queried for information on individual magnets and on properties of composed structures like the length of a cell. It is designed in such an extensible way that it can be used as a foundation for different simulation methods, which can be built on top of it. Such a method, which is capable of computing the Twiss parameters, dispersion function, chromaticity, emittance, and synchrotron radiation integrals, was implemented. This thesis covers how this new tool was developed and then used to optimize the O5T2off optics for the BESSY-VSR project, adapt the beta functions of the BESSY II storage ring for an emittance exchange experiment, and create a framework of automated lattice summaries, which could be useful for the development of a future BESSY III lattice. The developed code does not aim to replace the more mature and full-featured existing particle accelerator codes. However, it extends the ecosystem of accelerator tools by enabling fundamental lattice development using the Python language.

Acknowledgments

An erster Stelle möchte ich mich bei Herrn Professor Andreas Jankowiak bedanken. Danke, dass Sie mir diese Arbeit am Helmholtz-Zentrum Berlin ermöglicht haben und dass Sie diese Arbeit begutachten.

Ich möchte mich bei Dr. Markus Ries für das gemeinsame Experimentieren beim Nutzerzustandstest der Q5T2-off Optik und beim Emittance-Exchange Experiment bedanken.

Vielen Dank an Dr. Tom Mertens für die Diskussionen und Vorschläge rund um die Entwicklung des LatticeJSON Lattice File Formats. Danke auch für spannende physikalische Debatten.

Ebenfalls möchte ich mich bei Dr. Michael Abo-Bakr bedanken. Für die Unterstützung bei einigen Experimenten, Beantwortung beschleunigerphysikalischer Fragen und viele Diskussionen rund um die Entwicklung der Lattice Summaries.

Mein größter Dank geht an Dr. Paul Goslawski. Danke für Deine intensive Betreuung innerhalb der letzten Jahre seit meiner Bachelorarbeit. Danke für Deine vielen Anregungen, Kommentare und Dein großes Engagement. Danke für Deine Ausdauer beim spannenden nächtlichen Experimentieren. Und danke für viele interessante Diskussionen, manchmal auch etwas am Rande der Beschleunigerphysik.

Zuletzt möchte ich mich bei meinen Eltern bedanken. Danke für die große Unterstützung während der letzten Jahre und danke dafür, dass Ihr mir mein Studium ermöglicht habt.

Contents

Chapter 1

Introduction

This thesis presents the development of the new lattice design tool *apace*, the new JSON-based lattice file format *LatticeJSON*, and some of their applications.

This first chapter provides a brief overview of the third-generation light source BESSY II and some of its current lattices design tasks. Then, it discusses the challenges of integrating the existing simulation codes in a modern and high-level programming language like Python, which sets the motivation for the development of a new optics code as part of this thesis.

The second chapter covers the physical concepts of beam dynamics in electron storage rings needed to implement the new lattice design tool.

The third chapter addresses various challenges and considerations made during the implementation. Furthermore, it gives a short overview of the capabilities of the developed Python package.

The fourth chapter presents the applications of the developed optics simulation program for two optics changes at the BESSY II storage ring.

The last chapter concludes the thesis with a summary of the results and gives an outlook into its potential use cases in the future.

The appendix includes the documentation, the API reference, and the source code of the developed Python package.

1.1 BESSY II - A Third Generation Light Source

The third-generation synchrotron light source BESSY II is located in Berlin Adlershof and is operated by the research institute Helmholtz-Zentrum Berlin (HZB) since 1998. Its purpose is to provide extremely brilliant synchrotron light pulses in the range of terahertz radiation to hard X-rays. The storage ring has a circumference of 240 m and is equipped with 50 beamlines. A graphic overview of BESSY II is shown in Figure 1.1. In addition, the most important parameters of the storage ring are listed in Table 1.1.

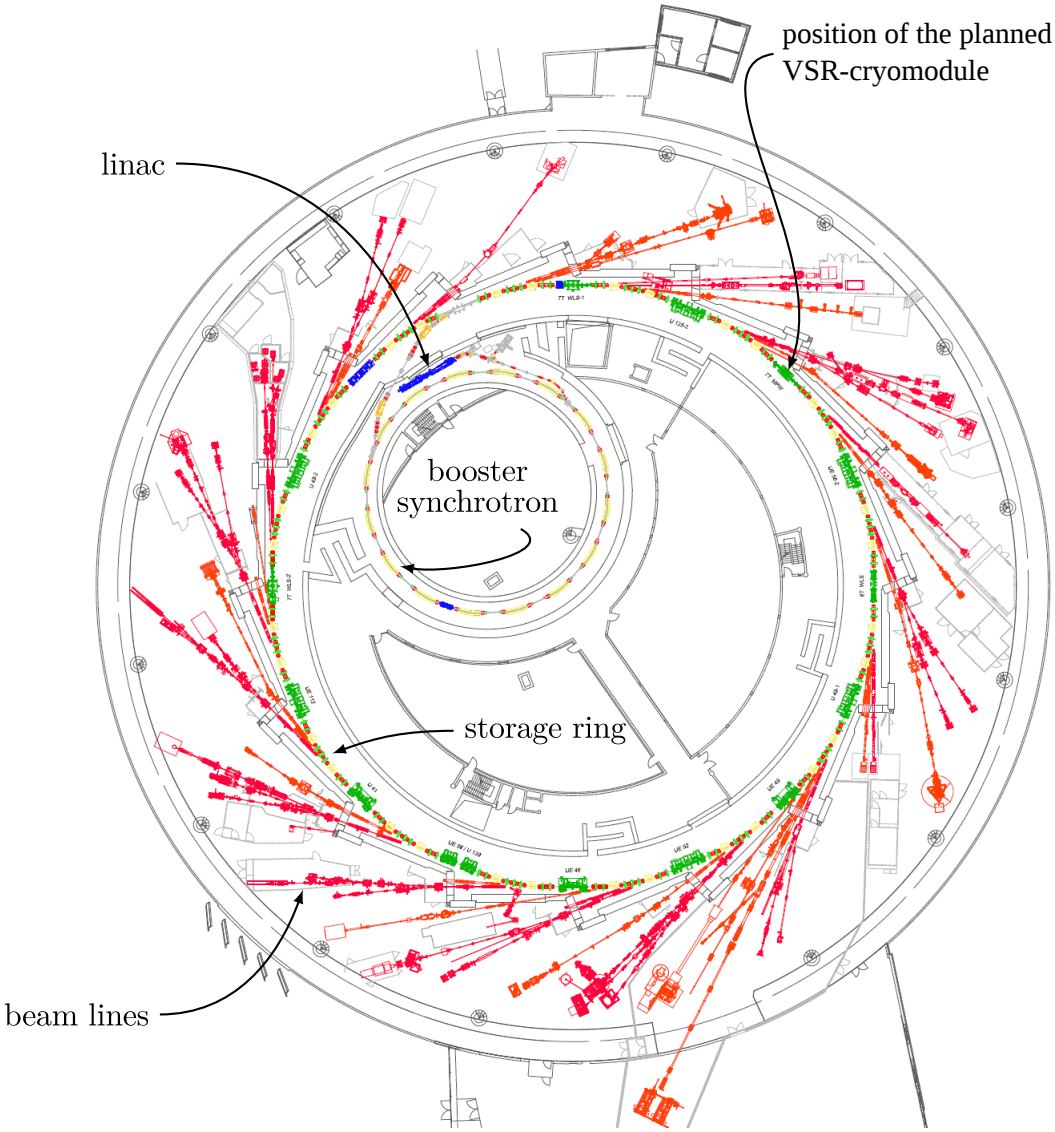

Figure 1.1: Floor plan of the synchrotron light source BESSY II (extracted from [1])

Table 1.1: Parameters of the BESSY II storage ring

Parameter	Value
nominal energy	1.7 GeV
horizontal emittance	≈ 7 nm rad
circumference	240 m
RF-frequency	500 MHz
revolution time	800 ns
beam current	300 mA
number of cells	16
number of bending magnets	32
bending radius	4.354 m
beam-lines	≈ 42

The electrons are emitted by a DC grid cathode and are accelerated up to 90 keV. In the following linear accelerator (LINAC), their energy is increased up to 50 MeV [2]. Next, the electrons are transferred to the booster synchrotron, where they are accelerated up to 1.7 GeV and injected into the storage ring cumulatively so that a beam current of 300 mA is maintained (*top-up*). The electrons can be stored for up to 10 hours and emit, depending on the type of deflection (bending magnet, wiggler, or undulator), photon energies from 10 eV up to 15 keV.

At BESSY II, it is possible to operate the machine in two different modes. Most of the time, the storage ring is set to the standard user optics with 15 ps bunch length. During two weeks of the year, the lattice is changed to the low α optics, which provide buckets with 3 ps bunch length [3]. This can be realized by reducing the momentum compaction factor α_c from $7 \cdot 10^{-4}$ to $4 \cdot 10^{-5}$. The coherent synchrotron radiation instability leads to a limiting bursting threshold current, which scales with ~α_c. Therefore the photon flux has to be reduced significantly in comparison to the standard optics. At this time, high flux users cannot run experiments, which is why the low alpha mode can only be provided for short periods.

1.2 Lattice Development at HZB - Examples

This section introduces some of the lattice development tasks at HZB, which have been addressed using the developed code.

1.2.1 Enlarging the Available Installation Length for the VSR Cryomodule

Due to increasing interest in studies that require very short photon pulses, HZB is developing a new operational mode called Variable pulse-length Storage Ring (VSR) [4], which aims to fulfill the requirements of the users who need short 2 ps bunches and of the users relying on the high average beam current, which is mainly stored in the 15 ps long bunches. The idea is to store long and short bunches in one storage ring simultaneously by establishing a beating pattern of higher harmonic cavities providing alternating longitudinal focusing gradients for two different bunch lengths.

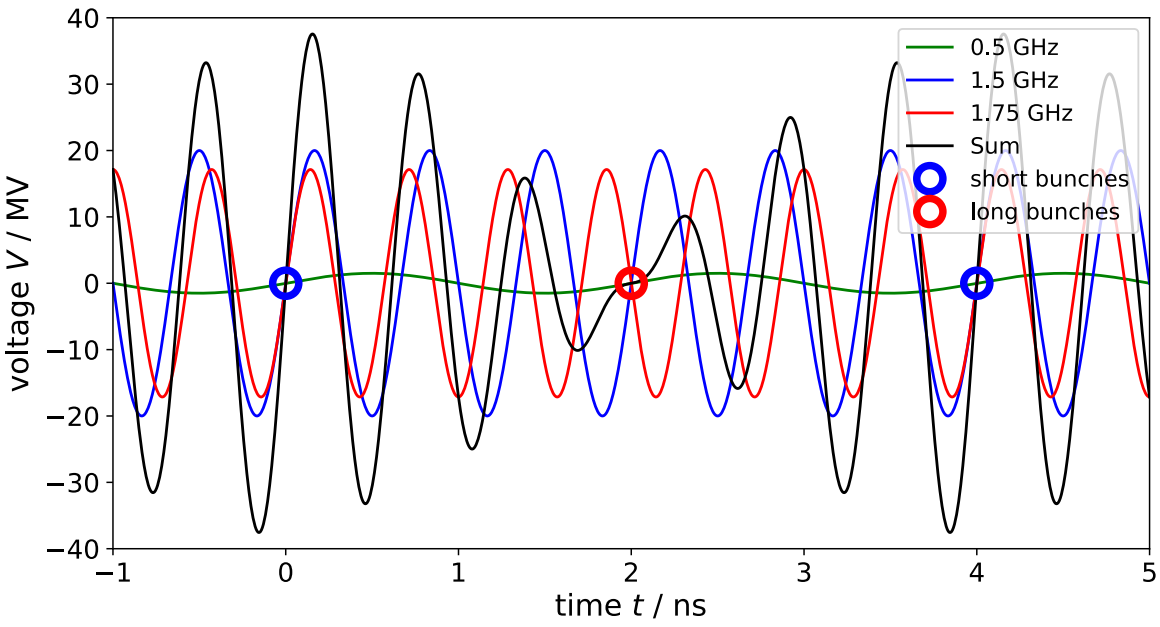

Figure 1.2: Voltage of the VSR cavities and their sum. The alternating large (blue) and small (red) gradients lead to short and longer bunches, respectively. (based on [4] and [1])

This will be achieved by the installation of an additional cavity module consisting of two 1.5 GHz and two 1.75 GHz cavities. The superposition of these two new frequencies and the existing 0.5 GHz cavity leads to alternating high (blue) and low gradients (blue) shown in Figure 1.2. The long bunches are located at the small voltage gradients, where the voltages of the 1.5 GHz and 1.75 GHz cavities cancel out. The short bunches, with higher current than in the low alpha mode, are produced at the high voltage gradient, where the voltages of the cavities add up.

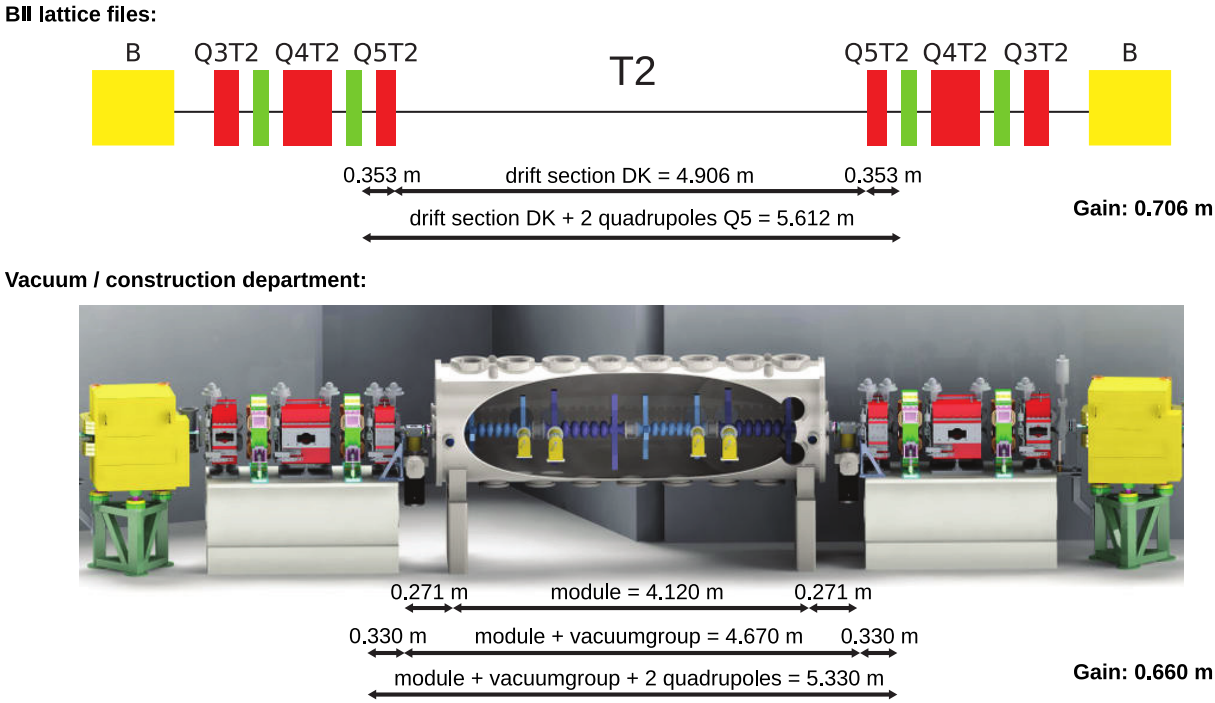

Figure 1.3: The currently available space within the T2 section of the storage ring. Removing the Q5 quadrupoles would lead to about 0.7 m of additional installation lenght for a larger cryomodule. (yellow - Dipole, red - Quadrupole, green - Sextupole)

The additional cavity module will be installed in the T2 section of the BESSY II storage ring, shown in Figure 1.3. One challenge - addressed by my bachelor's thesis [5] - is that the module needs more space than initially assumed. One solution is to remove the two Q5T2 quadrupoles to gain about 0.7 meters of installation length. The goal is to switch off the Q5T2 quadrupoles in simulations while maintaining the most important transverse linear parameters as the beta functions, tune, and momentum compaction factor. The best optics presented showed that a turn-off of the Q5T2 quadrupoles in the T2 straight is possible. Furthermore, the optics was tested at the storage ring, where it was possible to store high current with reasonable lifetime and injection efficiency.

As already stated in the conclusion of the bachelor's thesis, the presented optics have to be further optimized regarding different aspects: There is still a non-negligible beta beat outside of the T2 section. The linear optics could be further improved by overcoming some limitations of the code of the bachelor's thesis. Depending on the number of parameters, some optimization runs took several days. Optimizing time-critical parts of the developed code would allow optimizing using a higher number of quadrupole configurations. Also, the capability to mask certain regions, which would not be taking into account by the objective function, or set the beta functions to the desired value in the cavity module could further improve the obtained optics. That would require a significant rewrite of the developed code. Adjusting the objective function of the optimization method, discussed in Section 4.1, could improve the linear result. Besides the optimization of the linear beam dynamics, the optics has to be further optimized with regard to the non-linear dynamics. Different sextupole settings can be used to optimize the phase and momentum acceptance.

1.2.2 Emittance Exchange Experiment

With the uprise of 4^{th} generation light sources, the discussion of round beams becomes more relevant. A round beam is produced when the emittance is the same in both transversal planes. Such an emittance exchange is helpful in two use cases:

First, discussed in [6], an exchange of the transverse beam emittances can improve the injection efficiency. Since the beam is injected in the horizontal plane and the beam emittance is much larger in the horizontal plane for an electron synchrotron, the horizontal emittance primarily defines the required acceptance for a high injection efficiency. Therefore, reducing the horizontal emittance in the booster synchrotron, shown in Figure 1.4, by partially transferring it into the vertical plane can significantly lower the required acceptance of the storage ring and thus increase the injection efficiency.

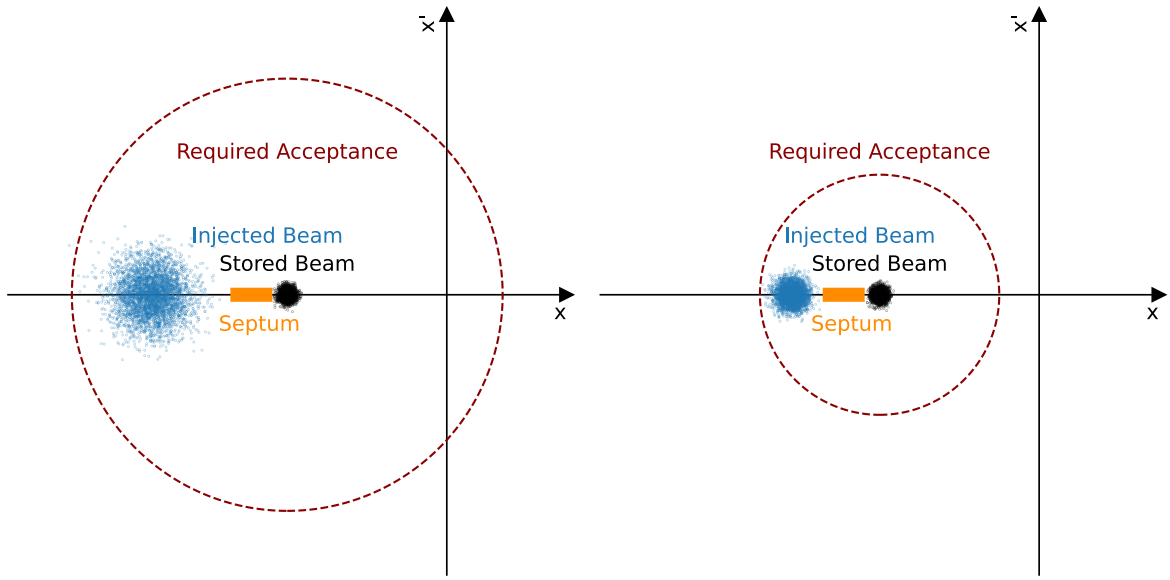

Figure 1.4: Influence of the horizontal emittance of the injected beam on the required aperture for high injection efficiency (based on [6])

Another reason for an emittance exchange is the generation of round beams in the storage ring. The goal is to match the electron beam phase space with the phase space of the emitted photon beam at the radiation source to optimize the photon beam's brilliance and coherent fraction.

There are different techniques to generate round beams. For example, operating on a coupling resonance or driving a skew excitation resonantly with the beam. One experiment to study the influence of the resonant skew excitation on the emittance exchange depending on the size of the beta function required an optics change in one of the triplet sections of the storage ring. Section 4.2 presents how the developed code was used to raise the horizontal beta function from 1.2 m to 15 m at the center of the straight.

1.2.3 Automated Lattice Summaries

At HZB, the development of the Conceptual Design Report (CDR) of the BESSY II successor BESSY III has started. Many lattice candidates arise during the development of an entirely new accelerator lattice, which themselves require serval iterations. These numerous versions of lattices are created by different physicists making different considerations and optimizing for different parameters. Due to the variety of accelerator physics codes, the lattice files and simulation results are stored in different formats. Even though many accelerator codes provide the

option to export the lattice files to different formats, this often does not work flawlessly and requires editing these files manually afterward. That makes sharing and comparing the different lattices candidates during the design phase very cumbersome and unnecessarily time-consuming.

The number of different simulation codes and lattice file formats is probably due to the variety of different lattice development tasks. Certain tasks can be very specific to a given facility and require a feature, which may not be implemented in one of the existing codes. Due to some codes not being open-source or since it can be easier to implement a certain feature from scratch than integrating it into one of the existing codes, this probably led to the historical fragmentation of accelerator physics codes and lattice file formats.

On the other hand, the boundary conditions for a new BESSY III lattice are well defined. Therefore, especially for fundamental lattices development, mainly linear beam dynamics, it should be possible to automate the process of sharing lattice files and simulation results.

Ideally, there would be a shared database of lattices. The lattices files would be automatically generated in multiple formats every time a lattice is added to the database. In the same way, a predefined set of simulations would be run. The simulation results could then be summarized in the form of a lattice report, for example, through a web page. Such a framework of automatically generated lattice summaries would also be useful for existing lattices. For example, to benchmark a BESSY III candidate lattice with one of the existing 4^{th} generation light sources or build an updated version of the Synchrotron Light Source Data Book [7].

As a starting point and to facilitate the creation of such a database, it might be helpful to use a slimmed-down version of lattice file formats. Such a restricted version would only contain basic element types such as drifts and multipoles. As these elements are available in all accelerator codes, this would make a one-to-one translation between the lattice file formats possible. Furthermore, with the lattice files available in the different formats, automated routines could be set up for the different simulations codes.

A prove-of-concept framework of automated lattice summaries was developed using the new lattice design tool and JSON-based lattice file format and will be presented in Section 4.3.

1.2.4 Smaller Lattice Development Tasks at the BESSY II Storage Ring

Sometimes there are smaller lattice development tasks, where, for example, a user needs to know the value of the beta function at a specific position in the storage ring. Often this task requires some post-processing, which at HZB is mainly done in Python. Thus, having a native

Python interface to the accelerator model and the Twiss parameters would be very beneficial for these tasks.

1.3 Motivation: The Need for a Python Interface to Particle Accelerator Simulations

MAD-X [8] and elegant [9] are some of the most mature and commonly used accelerator physics simulation codes, among many others, listed here [10]. Both programs a driven by an input file - often called run-file. This file defines various simulation parameters but can also be used to define an optimization procedure. After the execution, the results are stored in an output file. Over the years, various toolkits and programs emerged to inspect, post-process, and plot these results. Elegant even comes with its own post-processing toolkit SDDS [11]. Sometimes, for complex runs, more flexibility is needed, which is why Python is commonly used for post-processing and analysis of the simulation data.

Python has a rich ecosystem of numerical libraries and optimization tools, which is growing day by day. It has become the go-to language for scientific computing and machine learning. A typical workflow of using MAD-X or elegant with Python would be (see Figure 1.5):

1. Create a run file
2. Run the simulation defined by the run-file
3. Store the results as a file.
4. Load the simulation results into Python and post-process the data.
5. Output the post-processed data. (e.g. a plot or a new optics file)

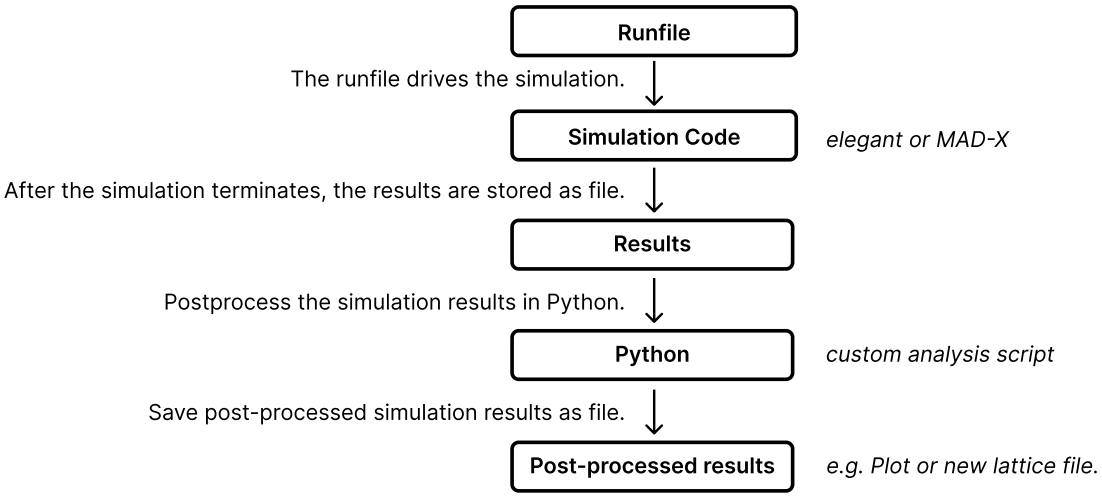

Figure 1.5: An exemplary workflow for using MAD-X or elegant from Python.

One issue of this workflow is that not all information contained within the data structures and models of the simulation software is included in the output files. For example, MAD-X and elegant do not include a complete model of the accelerator's lattice but only an array of elements for the simulated orbit positions. Another issue is that the execution of MAD-X and elegant can only be driven by their respective run-file. Although both run-files provide basic features of a programming language, such as variables, loops, or if-else statements, their capabilities are extremely limited compared to Python. Therefore it would be desirable to drive the execution of the simulation using Python. Full access to the accelerator model of these simulations software would require implementing a Python interface for MAD-X or elegant. PyMAD [12] is an attempt to create a Python API for MAD-X. As Python is a highly object-oriented language and MAD-X is mainly written in C and Fortran, it is not trivial to design an API that fits the different programming paradigms of these languages. The PyMAD project was unmaintained since 2017 but was recently picked up by the Heidelberg Ion-Beam Therapy Center (HIT) [13].

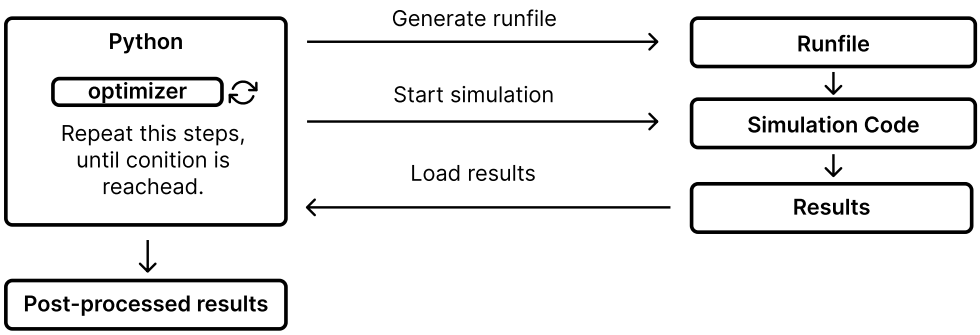

Figure 1.6: Driving the execution of MAD-X or elegant using Python. A new run-file is generated for each iteration of the optimization process.

To still leverage the powerful optimizers of the Python ecosystem, one common workaround is to use a template lattice-file or run-file: At the beginning of each iteration, Python generates a unique input file by inserting the values of the optimization parameters into the template file. Using this new run-file, Python then starts the simulation software as a sub-process, parses the results, calculates the value of the fitness function, and lets the optimizer compute the values of the optimization arguments for the next iteration. This is repeated until the terminating condition of the optimizer is met. Afterward, the final results are saved to a file. This workflow, illustrated in Figure 1.6, has several drawbacks:

- No direct access to the accelerator model.
- It is computationally very inefficient: The simulation software has to parse the run-file and rebuild the accelerator model for each iteration because the memory is freed after the program terminates.
- Storing simulation results in a file and loading them into Python for each iteration is another performance issue: Hard disks are many magnitudes slower than computer memory. Even though this could be enhanced by using a RAM disk, serializing and deserializing the simulation results for each iteration are still not optimal.
- Substituting strings in a run-file is very error-prone and can lead to hard-to-find bugs.

For these reasons, it would be desirable to have a direct Python API to drive the execution of accelerator simulations. As discussed above, integrating one of the existing simulation codes in Python is a difficult task. It was therefore decided to develop a new optics code as a native Python package. As Python is a dynamically typed language and its major implementation

CPython [14] does not convert the source code directly into native machine instruction, ordinary Python programs execute slower than programs written in lower-level languages like C or Fortran. To ensure an extremely fast calculation of the Twiss parameters, which is necessary because high-dimensional optimizations often require millions of iterations, time-critical parts were implemented in the C language.

The code is partially based on scripts written for the Q5T2-off optics of my bachelor's thesis. At the time of this thesis, the code is capable of calculating most of the linear parameters like the Twiss parameters, the dispersion function, the betatron phase, the tune, the momentum compaction factor, the natural chromaticity, the emittance, and synchrotron radiation integrals. Particle tracking is implemented by the matrix method. In addition, an experimental branch can do particle tracking by integrating the equations of motion, which is used for some plots in this thesis but not well tested.

A main feature of the developed code is its accelerator model, which includes an internal dependency graph between the accelerator elements: Whenever an element changes one of its attributes, it automatically notifies all its dependents. For example, suppose a magnet changes its length. In that case, it notifies its containing lattice that its length has to be recomputed, which in turn notifies a simulation method that the linear optics parameters have to be recalculated the next time they are accessed. That is convenient for the users because they do not have to keep track of complex dependency relations and ensure that only outdated properties are recomputed. No performance is wasted on calculating already known values.

The next chapter provides an introduction to the physical foundation of the developed code.

Chapter 2

Beam Dynamics in Electron Storage Rings

This chapter covers all concepts of electron beam dynamics in circular accelerators needed for implementing the developed beam optics code *apace*. The books of H. Wiedemann [15], A. Wolski [16], K. Wille [17] and, F. Hinterberger [18] are the main sources of this chapter.

Electrons in a storage ring oscillate around a curved ideal path. Transforming this ideal orbit away using a different coordinate system than Cartesian coordinates will facilitate our work in the following sections. Therefore, the first section introduces the co-moving Frenet-Serret coordinate system. Next, Section 2.2 derives the equations of motions in this new coordinate system. Linearizing these equations of motion allows for an analytical description of the particle beam, presented in Section 2.3. The following Section 2.4 addresses the effects of energy deviations of the particle beam on the betatron tune and ways to limit these chromatic errors. Section 2.5 provides a brief overview of the effects of synchrotron radiation on the beam properties. Finally, the last section discusses the concepts introduced at the example of different periodic lattices.

2.1 The Co-moving Coordinate System

To describe the motion of a particle in circular accelerators, one usually chooses the co-moving *Frenet-Serret coordinates*, whose origin follows the trajectory of the reference particle.

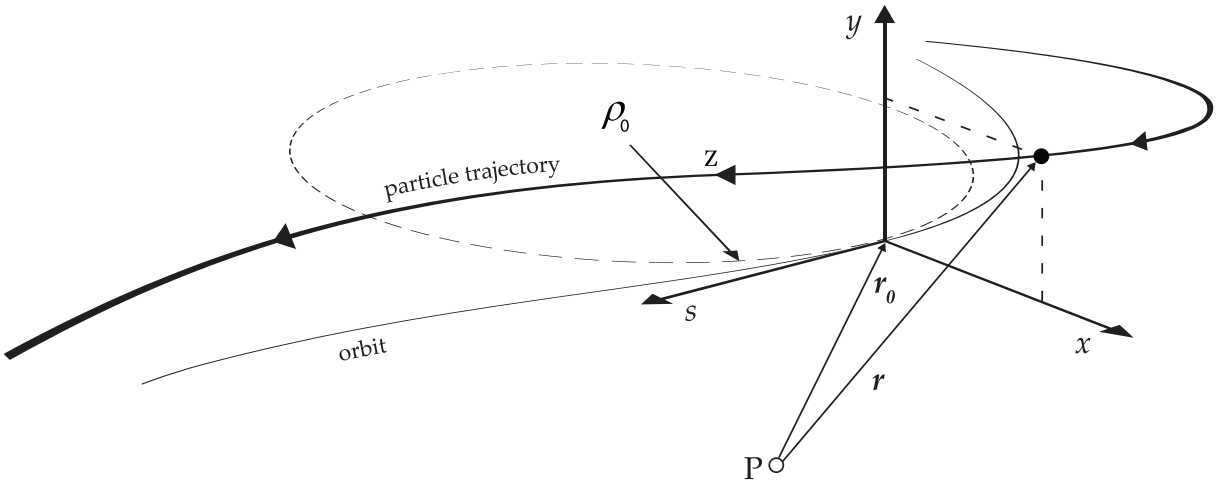

Figure 2.1: Co-moving Frenet-Serret coordinate system

The three basis vectors

$$\begin{aligned} \hat{\mathbf{e}}_\mathrm{s}(s) &= \frac{\mathrm{d}\mathbf{r}_0(s)}{\mathrm{d}s} && \text{tangent basis vector} \\ \hat{\mathbf{e}}_\mathrm{x}(s) & && \text{horizontal basis vector} \\ \hat{\mathbf{e}}_\mathrm{y}(s) &= \hat{\mathbf{e}}_\mathrm{s}(s) \times \hat{\mathbf{e}}_\mathrm{x}(s) && \text{vertical basis vector} \end{aligned} \tag{2.1}$$

span the Frenet-Serret coordinate system. While the $\hat{\mathbf{e}}_\mathrm{s}(s)$ vector moves tangential to the orbit, the horizontal $\hat{\mathbf{e}}_\mathrm{x}(s)$ and vertical $\hat{\mathbf{e}}_\mathrm{y}(s)$ vectors are perpendicular to it. The s coordinate describes the distance covered on the ideal orbit. The z coordinate defines the path length of the individual particle trajectory. The horizontal and vertical coordinates correspond to x and y, respectively. For statements that are valid for both transversal planes, we will use the general variable u.

In the Frenet-Serret system, the sum of the coordinate system's origin $\mathbf{r}_0(s)$, the horizontal displacement $x(s)\hat{\mathbf{e}}_\mathrm{x}(s)$, and the vertical displacement $y(s)\hat{\mathbf{e}}_\mathrm{y}(s)$ from the nominal orbit corresponds to the position vector of the individual particle:

$$\mathbf{r}(x, y, s) = \mathbf{r}_0(s) + x(s)\hat{\mathbf{e}}_\mathrm{x}(s) + y(s)\hat{\mathbf{e}}_\mathrm{y}(s) \tag{2.2}$$

In the following, we use a dot to denote a derivative with respect to time t and a prime mark to denote a derivative with respect to the orbit position s. Then the velocity and acceleration in the Frenet-Serret system are

$$\dot{\mathbf{r}}(x, y, s) = \dot{s}x'\hat{\mathbf{e}}_{\mathrm{x}}(s) + \dot{s}y'\hat{\mathbf{e}}_{\mathrm{y}}(s) + \dot{s}h\hat{\mathbf{e}}_{\mathrm{s}}(s) \tag{2.3}$$

and

$$\begin{aligned}\ddot{\mathbf{r}}(x, y, s) = &\left(x''\dot{s}^2 + x'\ddot{s} - h\kappa_{\mathrm{x0}}\dot{s}^2\right)\hat{\mathbf{e}}_{\mathrm{x}}(s) \\ &+ \left(y''\dot{s}^2 + y'\ddot{s} - h\kappa_{\mathrm{y0}}\dot{s}^2\right)\hat{\mathbf{e}}_{\mathrm{y}}(s) \\ &+ \left(2\kappa_{\mathrm{x0}}x'\dot{s}^2 + 2\kappa_{\mathrm{y0}}y'\dot{s}^2 + h\ddot{s}\right)\hat{\mathbf{e}}_{\mathrm{s}}(s).\end{aligned} \tag{2.4}$$

Here we introduced

$$h = 1 + \kappa_{\mathrm{x0}}x + \kappa_{\mathrm{y0}}y, \tag{2.5}$$

where κ_{x0} and κ_{y0} correspond to the horizontal and vertical curvature of the ideal orbit, respectively.

2.2 Equations of Motion

A particle with the charge q and the mass m moving through a structure of magnets experiences the magnetic part of the Lorentz force, which defines its equations of motion

$$\ddot{\mathbf{r}} = \frac{q}{m}(\dot{\mathbf{r}} \times \mathbf{B}), \tag{2.6}$$

where $\mathbf{B}$ is the magnetic field vector. Assuming a vanishing longitudinal component of the magnetic field $B_{\mathrm{z}} \approx 0$, we obtain the equations of motion in the Frenet-Serret coordinates by substituting Equation 2.3 and Equation 2.4 into Equation 2.6:

$$\begin{aligned}x''\dot{s}^2 + x'\ddot{s} - \dot{s}^2\kappa_{\mathrm{x0}}h &= -\frac{q}{m}B_{\mathrm{y}}h\dot{s} \\ y''\dot{s}^2 + y'\ddot{s} - \dot{s}^2\kappa_{\mathrm{y0}}h &= \frac{q}{m}B_{\mathrm{x}}h\dot{s}\end{aligned} \tag{2.7}$$

The second time derivate of the orbit position $\ddot{s}$ changes when a particle travels with an offset through a bending magnet or when it moves with an angle divergence to the orbit. However, as the transversal velocities of a relativistic particle beam are small compared to the longitudinal components, the first time derivate of the orbit position $\dot{s}$ changes slowly. Therefore we can approximate

$$\ddot{s} \approx 0, \tag{2.8}$$

simplifying the equations of motion

$$\begin{aligned} x'' &= -\frac{qh}{m\dot{s}} B_\mathrm{y} + \kappa_{\mathrm{x}0} h \\ y'' &= \frac{qh}{m\dot{s}} B_\mathrm{x} + \kappa_{\mathrm{y}0} h. \end{aligned} \tag{2.9}$$

When a magnetic field deflects a charged particle, the Lorentz force takes the place of the centripetal force

$$\begin{aligned} \mathbf{F}_\text{centripetal} &= \mathbf{F}_\text{Lorentz} \\ -mv^2\boldsymbol{\kappa} &= q(\dot{\mathbf{r}} \times \mathbf{B}). \end{aligned} \tag{2.10}$$

Here, the horizontal κ_x and vertical κ_y curvatures are defined as the curvatures experienced by an on-momentum particle due to the magnet lattice:

$$\begin{aligned} \kappa_\mathrm{x}(x,y,s) &= \frac{1}{\rho_\mathrm{x}(x,y,s)} = \frac{q}{p_0} B_\mathrm{y}(x,y,s) \\ \kappa_\mathrm{y}(x,y,s) &= \frac{1}{\rho_\mathrm{y}(x,y,s)} = -\frac{q}{p_0} B_\mathrm{x}(x,y,s) \end{aligned} \tag{2.11}$$

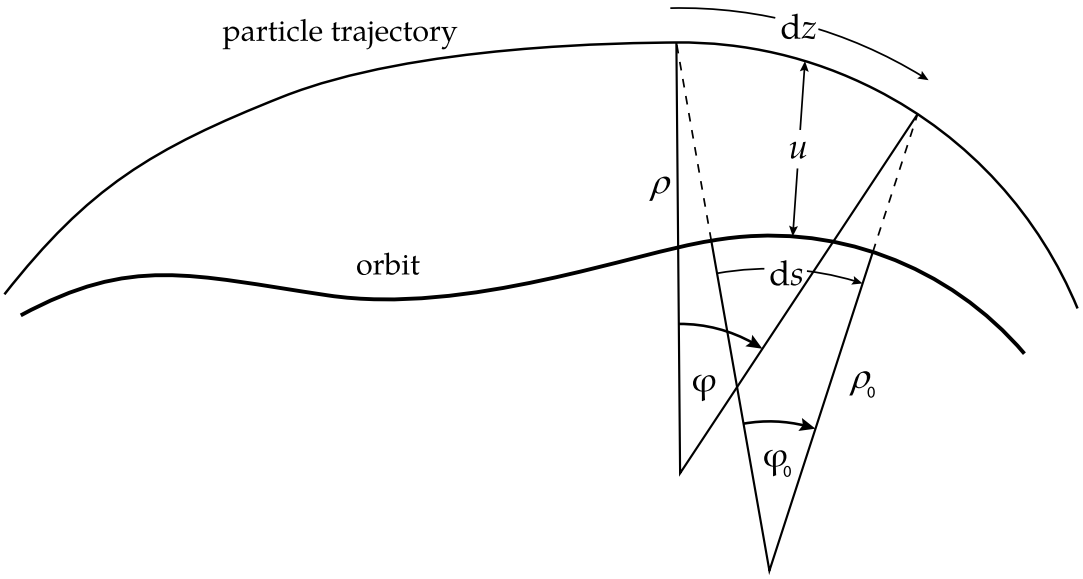

Figure 2.2: Path length difference between the orbit and an individual particle trajectory

Applying the linear approximation of the relationship between the path length of the orbit and the path length of an individual particle trajectory

$$\mathrm{d}z = (1 + \kappa_{\mathrm{x0}} x + \kappa_{\mathrm{y0}} y)\mathrm{d}s + \mathcal{O}(2), \tag{2.12}$$

shown in Figure 2.2, yields the momentum as an expression of the time derivative of the orbit position $\dot{s}$:

$$p = mv \approx (1 + \kappa_{\mathrm{x0}} x + \kappa_{\mathrm{y0}} y)\dot{s} = mh\dot{s} \tag{2.13}$$

We define the relative momentum deviation

$$\delta = \frac{\Delta p}{p_0} \tag{2.14}$$

as the ratio of the momentum deviation Δp and the momentum of the ideal particle p_0. Then we can express a particle's momentum

$$p = p_0 + \Delta p = p_0(1 + \delta) \tag{2.15}$$

in terms of its momentum deviation δ and the ideal momentum p_0.

Subsituting Equation 2.11, Equation 2.13, and Equation 2.15 into Equation 2.9, we obtain a second-order differential equation for the equations of motion

$$\begin{aligned} x'' &= -\frac{h^2}{(1+\delta)}\kappa_{\mathrm{x}} + h\kappa_{\mathrm{x0}} \\ y'' &= \frac{h^2}{(1+\delta)}\kappa_{\mathrm{y}} + h\kappa_{\mathrm{y0}}, \end{aligned} \tag{2.16}$$

completely defined by the geometric multipole strengths of the magnetic lattice of the accelerator

$$\begin{aligned} \kappa_{\mathrm{x}}(x, y, s) &= \kappa_{\mathrm{x0}}(s) + k(s)x + m(s)(x^2 - y^2) + \dots \\ \kappa_{\mathrm{y}}(x, y, s) &= \kappa_{\mathrm{y0}}(s) + k(s)y + m(s)xy + \dots . \end{aligned} \tag{2.17}$$

Integrating Equation 2.16 yields the individual particle trajectories and can be used to track particles through non-linear elements. An experimental implementation of this method is in-

cluded in the developed code, but it is not as performant as other tracking methods of more mature codes. Nevertheless, it was, for example, used to create the plots in Section 2.4.

2.3 Linear Beam Dynamics

Equation 2.16 does not allow for analytical investigations. Courant and Synder [19] developed a formalism that allows describing the particle beam using analytic quantities by linearizing the equations of motion. My bachelor's thesis [5] thoroughly discussed the theory of linear beam dynamics. Therefore this section skips most of the derivations and only provides a summary of the most important results.

2.3.1 Linearized Equations of Motion

To linearize Equation 2.16, we expand the expression

$$\frac{1}{1+\delta} = 1 - \delta + \mathcal{O}(2) \tag{2.18}$$

in δ, leading to:

$$\begin{aligned} x'' &= -(1-\delta)(1+\kappa_{\mathrm{x0}})^2(\kappa_{\mathrm{x0}} + kx) + \kappa_{\mathrm{x0}}(1+\kappa_{\mathrm{x0}}x) \\ y'' &= (1-\delta)(1+\kappa_{\mathrm{x0}})^2 ky \end{aligned} \tag{2.19}$$

Multiplying out the parentheses and only keeping terms linear in x, y, and δ, we obtain the linearized equations of motion:

$$\begin{aligned} x'' + (\kappa_{\mathrm{x0}}^2 + k)x &= \kappa_{\mathrm{x0}}\delta \\ y'' - ky &= 0 \end{aligned} \tag{2.20}$$

Equation 2.20 is a linear second order differential equation, which can be solved analytically. Therefore, the transformation of the particle trajectory

$$\mathbf{X}(s) = \begin{pmatrix} x(s) \\ x'(s) \\ y(s) \\ y'(s) \\ l(s) \\ \delta(s) \end{pmatrix} = \begin{pmatrix} \text{horizontal offset} \\ \text{horizontal slope} \\ \text{vertical offset} \\ \text{vertical slope} \\ \text{longitudinal offset} \\ \text{relative momentum deviation} \end{pmatrix} \tag{2.21}$$

through beam transport elements can be described as a matrix multiplication

$$\mathbf{X}(s+L) = \mathbf{R}(s, s+L)\mathbf{X}(s), \tag{2.22}$$

where $\mathbf{R}$ is a 6×6 transfer matrix. The solutions of Equation 2.20 for a drift section ($\kappa_{\mathrm{x}0} = k = 0$), for a dipole magnet ($\kappa_{\mathrm{x}0} \neq 0, k = 0$), and for a quadrupole ($\kappa_{\mathrm{x}0} = 0, k \neq 0$) are included in Appendix D.

2.3.2 Betatron Oscillation

Tracking a particle through the beam transport system by applying the different transfer matrices of beam transport elements yields the individual particle trajectory. However, Courant and Synder [19] developed a formalism, revealing more insightful analytical properties of the entire particle beam. The *Courant-Snyder functions*, sometimes called *Twiss parameters*, arise from separating the on- and off-momentum motion.

Therefore, we first solve the linearized equations of motion, Equation 2.20, for the *dispersion-free* case, leading us to the most fundamental quantity of the transversal beam motion, the beta function $\beta(s)$. Then, in Section 2.3.4, we introduce the dispersion function $\eta(s)$ to account for the transverse motions caused by momentum deviations.

By neglecting the off-momentum terms and combining the focusing terms into one parameter $K_{\mathrm{x}}(s) = \kappa_{\mathrm{x}0}^2(s) + k(s)$ for the horizontal and $K_{\mathrm{y}}(s) = -k(s)$ for the vertical plane, the linear equations of motion become

$$u''(s) + K_u(s)u(s) = 0. \tag{2.23}$$

Equation 2.23, known as *Hill's equation*, is a second-order linear ordinary differential equation similar to the harmonic oscillator. The difference is that the parameter $K(s) = K(s + C)$ is not constant but is a function periodic with the circumference of the accelerator C. The *Floquet's theorem* states that Hill's equation has two linearly independent solutions, given by

the product of a complex exponential function and a periodic function [20]. The real part of the general solution of Hill's equation is

$$u(s) = \sqrt{\epsilon_u \beta_u(s)} \cos(\psi_u(s) + \psi_{u0}), \tag{2.24}$$

where we identify the integration constants ϵ and ψ_0 with the emittance and the initial betatron phase, respectively. The beta function $\beta_u(s)$ is periodic with the circumference of the accelerator C.

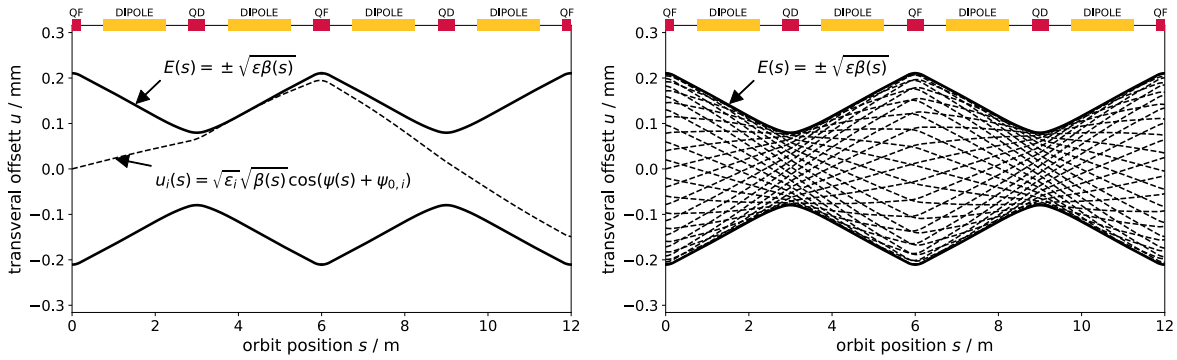

Figure 2.3: The envelope of a particle beam at the example of a FODO cell. The right plot shows the betatron oscillation for 33 electrons with an emittance of 5 nm rad. (extracted from [5])

Figure 2.3 shows the particle trajectories and the beam envelope as defined by Equation 2.24. The individual particles oscillate within the envelope

$$E(s) = \pm\sqrt{\epsilon\beta(s)}. \tag{2.25}$$

As the beta function $\beta(s)$ determines the shape of this envelope, and therefore the transverse beam size, it is considered one of the most important quantities in circular accelerator physics.

By substituting Equation 2.24 and its second derivative into Equation 2.23, we obtain an expression for the *betatron phase*

$$\psi_u(s) = \int_0^s \frac{\mathrm{d}s^*}{\beta_u(s^*)}. \tag{2.26}$$

The *tune*

$$Q_u = \frac{1}{2\pi} \int_s^{s+C} \frac{ds^*}{\beta_u(s^*)} \tag{2.27}$$

describes the number of betatron oscillations per turn.

By rearranging Equation 2.24 and its first derivative with respect to the orbit position s

$$u'(s) = -\frac{\sqrt{\epsilon}}{\sqrt{\beta_u(s)}}(\alpha_u(s)\cos(\psi_u(s)+\psi_{u0}) + \sin(\psi_u(s)+\psi_{u0})), \tag{2.28}$$

we obtain an ellipse equation for the Twiss parameters

$$\epsilon_u = \gamma_u(s)u^2(s) + 2\alpha_u(s)u(s)u'(s) + \beta_u(s)u'^2(s), \tag{2.29}$$

where

$$\alpha(s) := \frac{-\beta'(s)}{2} \tag{2.30}$$

and

$$\gamma(s) := \frac{1+\alpha^2(s)}{\beta(s)}. \tag{2.31}$$

Following Equation 2.29, the trajectory of betatron oscillation forms an ellipse in the (u, u')-phase space. The Twiss parameters $\alpha(s)$, $\beta(s)$, and $\gamma(s)$ determine the shape of this ellipse at the orbit position s. The emittance ϵ, which we introduced as an integration constant, describes the occupied phase space volume of the ellipse $A = \pi\epsilon$, according to *Liouville's theorem* a constant of motion [21].

Furthermore, supposing the Twiss parameters α, β, and γ are know for the positons s and $s+L$. Then, the 2×2-sub-matrices $\mathbf{R}_u^{2\times2}(s, L)$ of $\mathbf{R}(s, L)$ describing the transformation of a particle from the orbit position s to $s+L$ within horizontal or vertical plane is

$$\mathbf{R}_u^{2\times2}(s, L) = \begin{pmatrix} \sqrt{\frac{\beta_1}{\beta_0}}(\cos\psi_1 + \alpha_0\sin\psi_1) & \sqrt{\beta_0\beta_1}\sin\psi_1 \\ \frac{\alpha_0-\alpha_1}{\sqrt{\beta_0\beta_1}}\cos\psi_1 - \frac{1+\alpha_0\alpha_1}{\sqrt{\beta_0\beta_1}}\sin\psi_1 & \sqrt{\frac{\beta_0}{\beta_1}}(\cos\psi_1 - \alpha_1\sin\psi_1), \end{pmatrix} \tag{2.32}$$

where $\beta_0 = \beta_u(s), \beta_1 = \beta_u(s+L), \alpha_0 = \alpha_u(s), \alpha_1 = \alpha_u(s+L)$, and $\psi_1 = \psi_u(s+L)$.

2.3.3 Transformation of the Twiss parameters

Equation 2.29 can be written in matrix form:

$$\epsilon_u = \begin{pmatrix} u(s) & u'(s) \end{pmatrix} \begin{pmatrix} \gamma_u(s) & \alpha_u(s) \\ \alpha_u(s) & \beta_u(s) \end{pmatrix} \begin{pmatrix} u(s) \\ u'(s) \end{pmatrix} \tag{2.33}$$

We define the *Twiss matrix*

$$\mathbf{B}_u(s) = \begin{pmatrix} \beta_u(s) & -\alpha_u(s) \\ -\alpha_u(s) & \gamma_u(s) \end{pmatrix}. \tag{2.34}$$

Since the emittance ϵ is the same at the position s and $s + L$, rearranging Equation 2.33

$$\begin{aligned} \epsilon &= \mathbf{X}^{\mathrm{T}}(s)\mathbf{B}^{-1}(s)\mathbf{X}(s) \\ &= \mathbf{X}^{\mathrm{T}}(s)\mathbf{R}^{\mathrm{T}}(\mathbf{R}^{\mathrm{T}})^{-1}\mathbf{B}^{-1}(s)\mathbf{R}^{-1}\mathbf{R}\mathbf{X}(s) \\ &= (\mathbf{R}\mathbf{X}(s))^{\mathrm{T}}(\mathbf{R}\mathbf{B}(s)\mathbf{R}^{\mathrm{T}})^{-1}(\mathbf{R}\mathbf{X}(s)) \\ &= \mathbf{X}^{\mathrm{T}}(s+L)(\mathbf{R}\mathbf{B}(s)\mathbf{R}^{\mathrm{T}})^{-1}\mathbf{X}(s+L) \\ &\overset{!}{=} \mathbf{X}^{\mathrm{T}}(s+L)\mathbf{B}^{-1}(s+L)\mathbf{X}(s+L), \end{aligned} \tag{2.35}$$

results in an expression for the transformation of the Twiss parameters

$$\mathbf{B}(s+L) = \mathbf{R}(s,L)\mathbf{B}(s)\mathbf{R}^{\mathrm{T}}(s,L). \tag{2.36}$$

Multiplying the matrices of Equation 2.36 yields a system of linear equations

$$\begin{aligned} \beta(s+L) &= R_{11}^2\beta(s) - 2R_{11}R_{12}\alpha(s) + R_{12}^2\gamma(s) \\ \alpha(s+L) &= -R_{11}R_{12}\beta(s) + (R_{11}R_{22} + R_{12}^2)\alpha(s) + R_{12}R_{22}\gamma(s) \\ \gamma(s+L) &= R_{12}^2\beta(s) - 2R_{12}R_{22}\alpha(s) + R_{22}^2\gamma(s). \end{aligned} \tag{2.37}$$

For a stable solution to exist, the Twiss functions must have the same value after one revolution, leading to the periodicity condition for circular accelerators

$$\begin{aligned} \beta(s) &= \beta(s + C_0) \\ \alpha(s) &= \alpha(s + C_0) \\ \gamma(s) &= \gamma(s + C_0), \end{aligned} \tag{2.38}$$

where C_0 is the length of the ideal orbit. Using this condition we can solve the system of linear equations of Equation 2.37, yielding the Twiss functions as expression of the one-turn-matrix $\mathbf{R}(s, C_0)$:

$$\begin{aligned}\beta(s) &= \frac{2R_{12}}{\sqrt{2 - R_{11}^2 - 2R_{12}R_{21} - R_{22}^2}} \\ \alpha(s) &= \frac{R_{11} - R_{22}}{2R_{12}}\beta(s) \\ \gamma(s) &= \frac{1 + \alpha^2(s)}{\beta(s)}.\end{aligned} \tag{2.39}$$

From the expression of the beta function $\beta(s)$ in Equation 2.39, it follows that a stable solution only exists for

$$2 - R_{11}^2 - 2R_{12}R_{21} - R_{22}^2 > 0, \tag{2.40}$$

defining a stability criteria.

2.3.4 Dispersion

The *dispersion function*

$$\eta_u(s) = \frac{\mathrm{d}u(s)}{\mathrm{d}\delta} \tag{2.41}$$

describes the change in the transverse offset $u(s)$ with respect to the relative momentum deviation δ. Therefore, the trajectory of an off-momentum particle is given by the sum of the betatron oscillation $u_\beta(s)$ and the dispersive offset $u_\delta = \eta_u(s)\delta$:

$$u(s) = u_\beta(s) + u_\delta(s) = u_\beta(s) + \eta_u(s)\delta \tag{2.42}$$

Supposing the values of the dispersion function $\eta(s)$ and its derivative $\eta'(s)$ are known at the orbit position s. Then, the transfer matrix $\mathbf{R}(s, L)$ determines their values at the orbit position $s + L$:

$$\begin{aligned}\eta(s + L) &= R_{11}\eta(s) + R_{12}\eta'(s) + R_{16} \\ \eta'(s + L) &= R_{21}\eta(s) + R_{22}\eta'(s) + R_{26}\end{aligned} \tag{2.43}$$

Using the periodicity condition for circular accelerators

$$\begin{aligned} \eta(s) &= \eta(s + C_0) \\ \eta'(s) &= \eta'(s + C_0) \end{aligned} \tag{2.44}$$

the dispersion function $\eta(s)$ and its derivative $\eta'(s)$ can be expressed in terms of the one-turn-matrix $\mathbf{R}(s, C_0)$:

$$\begin{aligned} \eta(s) &= \frac{R_{16}(1 - R_{22}) + R_{12}R_{26}}{2 - R_{11} - R_{22}} \\ \eta'(s) &= \frac{R_{16}(1 - R_{11}) + R_{21}R_{16}}{2 - R_{11} - R_{22}} \end{aligned} \tag{2.45}$$

To calculate the graph of $\eta(s)$ and $\eta'(s)$ along the entire ring, Equation 2.45 can be used to calculate the values $\eta_0 = \eta(s_0)$ and $\eta'_0 = \eta'(s_0)$ at an initial orbit position s_0. Afterward, Equation 2.43 yields the values of $\eta(s)$ and $\eta'(s)$ for all other orbit positions s.

2.3.5 Momentum Compaction

The *momentum compaction factor*

$$\alpha_\mathrm{c} = \frac{1}{C_0}\frac{\mathrm{d}C}{\mathrm{d}\delta} \quad \text{with} \quad C = C_\beta + \Delta C_\delta, \tag{2.46}$$

is defined as the ratio between the change in the path length of a dispersive particle for one revolution C with respect to the relative momentum deviation δ and the path length of the ideal orbit C_0. While C_β is the path length of an on-momentum particle,

$$\Delta C_\delta = C - C_\beta = \int_0^C \mathrm{d}z - \int_0^{C_\beta} \mathrm{d}z' \tag{2.47}$$

corresponds to the path length difference caused by the momentum deviation. Applying the linear approximation of the path length element $\mathrm{d}z \approx (1 + \kappa_{\mathrm{x}0}x)\mathrm{d}s$, introduced in Equation 2.12, to Equation 2.47 yields

$$\begin{aligned} \Delta C_\delta &= \int_0^{C_0} 1 + \kappa_{\mathrm{x}0}(s)(u_\beta(s) + u_\delta(s))\mathrm{d}s - \int_0^{C_0} 1 + \kappa_{\mathrm{x}0}(s)u_\beta(s)\mathrm{d}s \\ &= \delta \int_0^{C_0} \kappa_{\mathrm{x}0}(s)\eta(s)\mathrm{d}s. \end{aligned} \tag{2.48}$$

Substituting Equation 2.48 into Equation 2.46, we obtain an expression for the momentum compaction factor

$$\alpha_{\mathrm{c}} = \frac{1}{C_0} \int_0^{C_0} \kappa_{\mathrm{x0}}(s)\eta(s)\mathrm{d}s, \tag{2.49}$$

corresponding to the mean value of the product between the curvature of the ideal orbit $\kappa_{\mathrm{x0}}(s)$ and the dispersion function $\eta(s)$.

2.4 Chromaticity

As discussed in the previous section, many effects of beam dynamics can be explained by the analysis of the linear equations of motion. However, certain phenomena can only be understood considering the higher-order terms.

The first subsection discusses the influence of energy deviations on the betatron tune and why they are harmful to the beam quality. Using elements with non-linear field components can limit these effects, described in the following subsection.

2.4.1 Natural Chromaticity in a Storage Ring

The number of betatron oscillations per turn, defined by the tune, is determined by the position and strength of the quadrupole and bending magnets. However, as the experienced multipole strength of a particle depends on its momentum, off-momentum particles will complete a different number of betatron oscillations per turn. Figure 2.4 shows the influence of the momentum deviation on the focal length of a quadrupole. Generally, particles with a higher momentum $\delta > 0$ are less deflected by a quadrupole, resulting in fewer betatron oscillations. On the other hand, particles with a negative momentum deviation $\delta < 0$ undergo a larger deflection, leading to a higher tune.

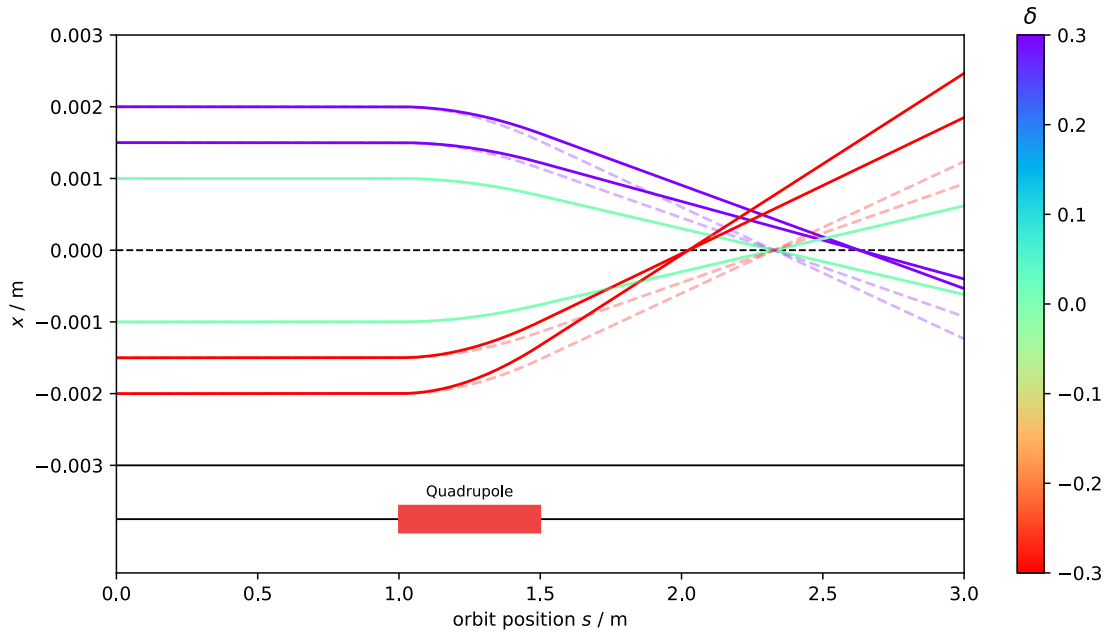

Figure 2.4: Impact of the momentum deviation on the focal length of a quadrupole. The trajectories were calculated by integrating Equation 2.16. The dashed lines correspond to the particle trajectories without momentum deviation $\delta = 0$. The stronger deflection for particles with $\delta < 0$ (red) leads to a shorter focal length. Particles with $\delta > 0$ (blue) are less deflected, resulting in a longer focal length. In the case of on-momentum particle $\delta = 0$ (green), the trajectories are identical to the linear case.

The *chromaticity*

$$\xi_{\mathrm{u}} = \frac{\mathrm{d}Q_{\mathrm{u}}}{\mathrm{d}\delta} \tag{2.50}$$

describes the change in the betatron tune with respect to the relative momentum deviation δ. In linear approximation, the perturbed quadrupole strength caused by the momentum deviation

$$k_{\mathrm{p}}(s,p) = \frac{q}{p}\frac{\mathrm{d}B_{\mathrm{y}}}{\mathrm{d}x} = \frac{q}{p_0(1+\delta)}\frac{\mathrm{d}B_{\mathrm{y}}}{\mathrm{d}x} \approx (1-\delta)\frac{q}{p_0}\frac{\mathrm{d}B_{\mathrm{y}}}{\mathrm{d}x} = k + \Delta k \tag{2.51}$$

has the same effect as a gradient error

$$\Delta k = -k\delta. \tag{2.52}$$

Therefore, to derive the effect of this gradient error, we introduce a perturbation in form of a thin quadrupole

$$\mathbf{Q}_\mathrm{p} = \begin{pmatrix} 1 & 0 \\ -\Delta k \mathrm{d}s & 1 \end{pmatrix} \tag{2.53}$$

with the infinite focal lenth $\frac{1}{\Delta k \mathrm{d}s}$. Here we look at the 2×2-sub matrix, which only transforms the horizontal or vertical componentes of the particle vector $\mathbf{X}$. According to Equation 2.32 the 2×2-one-turn-matrix in the horizontal or vertical plane

$$\mathbf{R} = \begin{pmatrix} \cos\Psi + \alpha_0 \sin\Psi & \beta_0 \sin\Psi \\ -\frac{1+\alpha_0^2}{\beta_0} \sin\Psi & \cos\Psi - \alpha_0 \sin\Psi \end{pmatrix} \tag{2.54}$$

can be written as an expression of the Twiss parameters, where $\Psi = 2\pi Q$. Hence, we can express the perturbed one-turn-matrix in terms of the unperturbed Twiss parameters:

$$\begin{aligned} \mathbf{R}_\mathrm{p} &= \mathbf{R}\mathbf{Q}_\mathrm{p} \\ &= \begin{pmatrix} \cos\Psi + \alpha_0 \sin\Psi & \beta_0 \sin\Psi \\ -\frac{1}{\beta_0} \sin\Psi & \cos\Psi - \alpha_0 \sin\Psi \end{pmatrix} \begin{pmatrix} 1 & 0 \\ -\Delta k \mathrm{d}s & 1 \end{pmatrix} \\ &= \begin{pmatrix} \cos\Psi + \alpha_0 \sin\Psi - \beta_0 \Delta k \mathrm{d}s \sin\Psi & \cdots \\ \cdots & \cos\Psi - \alpha_0 \sin\Psi \end{pmatrix} \end{aligned} \tag{2.55}$$

By calculating the trace of the perturbed one-turn-matrix

$$\begin{aligned} \mathrm{Tr}(\mathbf{R}_\mathrm{p}) &\overset{!}{=} \mathrm{Tr}(\mathbf{R}\mathbf{Q}_\mathrm{p}) \\ 2\cos\Psi_\mathrm{p} &= 2\cos\Psi - \beta_0 \Delta k \mathrm{d}s \sin\Psi \end{aligned} \tag{2.56}$$

we obtain

$$\cos\Psi \cos \mathrm{d}\Delta\Psi - \sin\Psi \sin \mathrm{d}\Delta\Psi = \cos\Psi - \frac{1}{2}\beta_0 \Delta k \mathrm{d}s \sin\Psi, \tag{2.57}$$

where $\Psi_\mathrm{p} = \Psi + \mathrm{d}\Delta\Psi$. For small tune pertubations ΔQ, we can use the small-angle approximiation of the trigonometric functions $\cos \mathrm{d}\Delta\Psi \approx 1$ and $\sin \mathrm{d}\Delta\Psi \approx \mathrm{d}\Delta\Psi$, resulting in an expression for the differential of the tune pertubation

$$\begin{aligned} \mathrm{d}\Delta Q &= \frac{1}{4\pi}\beta_0 \Delta k \mathrm{d}s \\ &= -\frac{1}{4\pi}\delta\beta_0 k \mathrm{d}s. \end{aligned} \tag{2.58}$$

By integrating Equation 2.58 and substituting it into Equation 2.50, we obtain an expression for the *natural chromaticities*

$$\begin{aligned} \xi_{\mathrm{x}}^{\mathrm{n}} &= -\frac{1}{4\pi}\int_0^C \beta_{\mathrm{x}}(s)k(s)\mathrm{d}s \\ \xi_{\mathrm{y}}^{\mathrm{n}} &= +\frac{1}{4\pi}\int_0^C \beta_{\mathrm{y}}(s)k(s)\mathrm{d}s, \end{aligned} \tag{2.59}$$

which only takes the contribution of the momentum-dependent quadrupole strength into account. Since particles with a positive momentum deviation ($\delta > 0$) experience a weaker focusing, the natural chromaticities ξ_u^{n} are always negative.

2.4.2 Chromaticity Correction

Two effects cause chromaticities to be undesirable in circular accelerators, making it necessary to correct for these chromatic errors. First, resonances between the betatron oscillation and the magnetic fields arising at specific values of the betatron tune can lead to a beam loss. Thus, for high chromaticities, where even particles with small momentum deviations experience a large tune shift, moving particles into resonance becomes more likely. Moreover, particles have vastly different tunes for high chromaticities, corresponding to a tune spread in the tune diagram. Hence, the accelerator has to be operated at a distance to the nearest resonances larger than the tune spread, imposing strict constraints on possible tunes and making it challenging to choose an operational tune.

Secondly, so-called *head-tail instabilities*, a collective effect between the electrons of the head and tail of a bunch, grow proportional with the chromaticity.

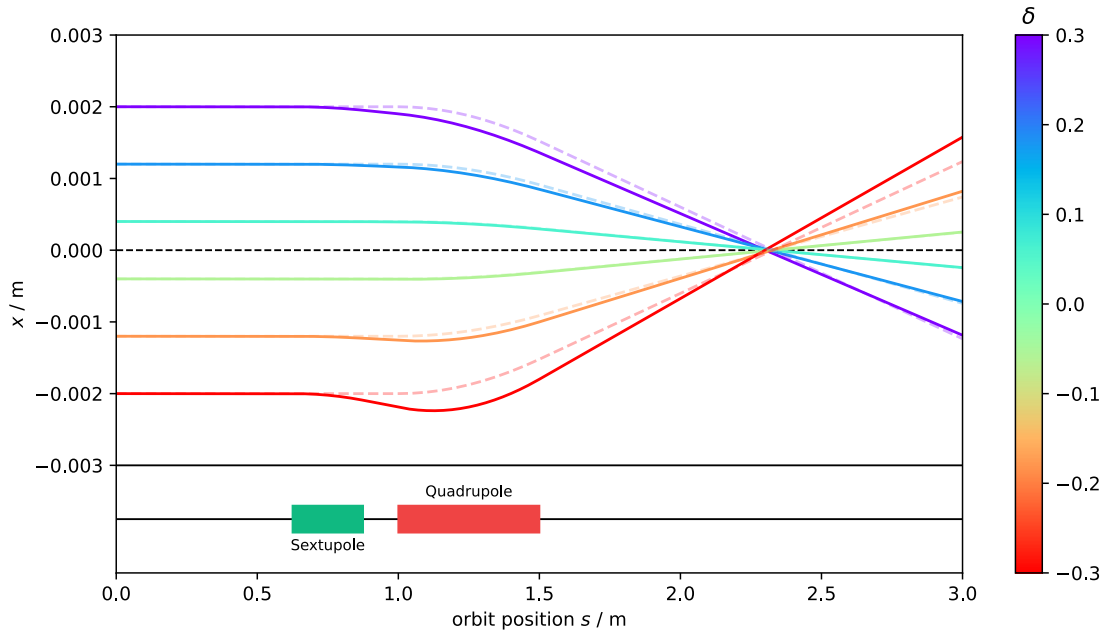

Figure 2.5: Influence of sextupole placed at a position of non-zero dispersion. Similar to Figure 2.4, the trajectories were calculated by integrating Equation 2.16. The dashed lines correspond to the particle trajectories without momentum deviation $\delta = 0$. The non-linearity of the sextupole corrects the chromatic error of the quadrupole, resulting in a focal point of the dispersive particle.

Non-linear elements can compensate for the chromaticities introduced by the momentum-dependent quadrupole strength. The goal is to give particles with a positive momentum deviation $\delta > 0$ an additional kick and counterbalance the stronger kick, which particles with a negative momentum deviation $\delta < 0$ experience. Figure 2.5 shows a sextupole compensating for the chromaticity introduced by a quadrupole. A sextupole magnet behaves in one plane like an amplitude-dependent quadrupole magnet. Therefore, a sextupole has the desired effect at a position of dispersion $\eta \neq 0$, where the particles are sorted by their momentum deviation δ.

Using the additional sextupole kick experienced by a dispersive particle $\Delta k = m\eta\delta$, we can, analog to Section 2.4.1, derive an expression for the part of the chromaticity introduced by the sextupole magnets

$$\begin{aligned} \xi_{\mathrm{x}}^{\mathrm{s}} &= +\frac{1}{4\pi}\int_0^C \beta_{\mathrm{x}}(s) m(s) \eta_{\mathrm{x}}(s) \mathrm{d}s \\ \xi_{\mathrm{y}}^{\mathrm{s}} &= -\frac{1}{4\pi}\int_0^C \beta_{\mathrm{y}}(s) m(s) \eta_{\mathrm{x}}(s) \mathrm{d}s. \end{aligned} \tag{2.60}$$

We obtain the total chromaticity from the sum of the natural chromaticity ξ^{n} and the chromaticity caused by the sextupole magnets ξ^{s}

$$\begin{aligned} \xi_{\mathrm{x}} &= -\frac{1}{4\pi}\int_0^C \beta_{\mathrm{x}}(s)(k(s) - m(s)\eta_{\mathrm{x}}(s))\mathrm{d}s \\ \xi_{\mathrm{y}} &= +\frac{1}{4\pi}\int_0^C \beta_{\mathrm{y}}(s)(k(s) - m(s)\eta_{\mathrm{x}}(s))\mathrm{d}s. \end{aligned} \tag{2.61}$$

2.5 Synchrotron Radiation

A resting electron produces a static electric field but does not emit radiation, which would violate the law of conservation of energy since a stationary electron has no kinetic energy to use. The same must hold for a uniformly moving electron as it is stationary in another inertial frame of reference. However, if a charged particle is accelerated, it produces an electromagnetic wave. According to Maxwell's equations, the change in the electric field results in a magnetic field. In turn, the variations of the magnetic field produce an electric field. These periodical oscillations in the electromagnetic field, which propagate energy at the speed of light c, are known as *electromagnetic radiation*.

In the case of a charged particle in a circular accelerator where it is accelerated radially in a bending magnet, undulator, or wiggler at relativistic speed, this radiation is called *synchrotron radiation*. The previous sections discussed the beam dynamics while neglecting the emission of synchrotron radiation. However, the random emission of photons of a radially accelerated electron influences the amplitude of its betatron and synchrotron oscillations, which changes the beam emittance.

At first, this might seem like a violation of Liouville's theorem [21]: It describes the time evolution of an ensemble of classical systems and states that the phase-space distribution of these systems is constant along any path. However, it is also applicable to a single system of N non-interacting electrons as such a system can be seen equivalent to an ensemble of N systems. Thus, for non-interacting electrons, the emittance, which corresponds to the occupied volume in phase space, must be conserved. However, as soon as an electron emits a photon, the electron beam emittance only occupies a sub-volume of the total electron-photon phase space, spanned by the coordinates of the electron and photon. Thus due to the synchrotron radiation and consistent with Liouville's theorem, the beam emittance can change. Nevertheless, Liouville's theorem still holds for the entire electron-photon system. Therefore, contrary to a

proton beam, where the synchrotron radiation is neglectable, the electron beam emittance is not a constant of motion.

In the following, we will derive the influence of synchrotron radiation on the electron beam emittance, where the report of Sands [22] was the primary source for this section. It is convenient to define the five so-called *synchrotron radiation integrals* [23], to facilitate the mathematical work:

$$I_1 = \int_0^C \kappa_{\mathrm{x}0}(s)\eta_{\mathrm{x}}(s)\mathrm{d}s \tag{2.62}$$

$$I_2 = \int_0^C \kappa_{\mathrm{x}0}(s)^2\mathrm{d}s \tag{2.63}$$

$$I_3 = \int_0^C |\kappa_{\mathrm{x}0}(s)|^3\mathrm{d}s \tag{2.64}$$

$$I_4 = \int_0^C \kappa_{\mathrm{x}0}(s)\eta_{\mathrm{x}}(s)\left(\kappa_{\mathrm{x}0}(s)^2 + 2k(s)\right)\mathrm{d}s \tag{2.65}$$

$$I_5 = \int_0^C |\kappa_{\mathrm{x}0}(s)|^3\mathcal{H}_{\mathrm{x}}(s)\mathrm{d}s \tag{2.66}$$

With

$$\mathcal{H}_{\mathrm{x}} = \gamma_{\mathrm{x}}\eta_{\mathrm{x}}^2 + 2\alpha_{\mathrm{x}}\eta_{\mathrm{x}}\eta_{\mathrm{x}}' + \beta_{\mathrm{x}}\eta_{\mathrm{x}}'^2. \tag{2.67}$$

we introduced the so-called *curly $\mathcal{H}$ function*, fully defined by the Twiss parameters. Note that for an accurate calculation of the fourth synchrotron radiation integral in Equation 2.65 we also have to consider the dipole edge focusing. In the developed code, this was adopted from the MAD-X source [8].

2.5.1 Radiation Damping of Betatron Oscillations

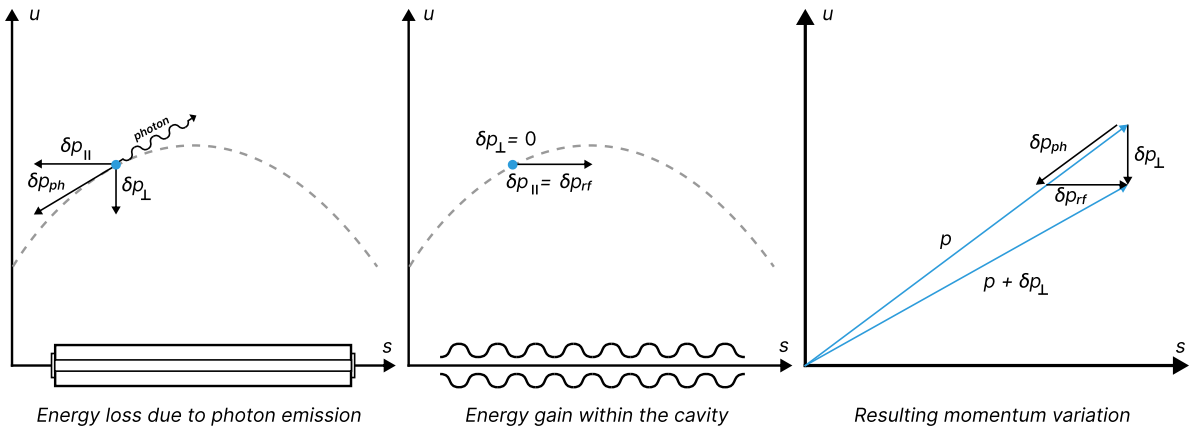

Figure 2.6: Radiation damping of the betatron oscillation. The emission of a photon changes the transverse and longitudinal components of an electron's momentum. But as the cavity only refills the longitudinal component, the transverse momentum components decrease over time.

As shown in Figure 2.6, a radially accelerated electron emits synchrotron radiation in its general direction of motion, leading to a decrease of the longitudinal and transversal components of its momentum. However, the electric field of the cavity only restores the electron's longitudinal momentum. Over time, the interplay of both of these effects leads to a damping of the electron's betatron oscillation. While this effect is commonly called the *radiation damping*, the damping of the betatron oscillations occurs in the cavity. The emission of synchrotron radiation does not - apart from slightly changing the focusing strength of the magnets due to the energy loss - directly affect the amplitude of the betatron oscillations. However, the damping is still a second-order effect of the synchrotron radiation, as, without it, the cavity would not restore the longitudinal momentum.

To quantify the influence of the synchrotron radiation on the amplitude of the betatron oscillations, we make some assumptions:

- First, we assume the emission of synchrotron radiation is precisely parallel to the momentum of the electron.
- Second, we assume a continuous emission of synchrotron radiation over a given path length.
- Finally, we assume the acceleration happens continuously along the ring.

Under these assumptions, the betatron oscillations would dampen to zero. However, due to the quantum nature of photons, the emission happens in discrete chunks of energy, making the second assumption not entirely reasonable. The discrete emissions lead to a counteracting excitation which will be considered in Section 2.5.2.

We recall the definition of the emittance

$$\epsilon_u = \gamma_u u^2 + 2\alpha_u u u' + \beta_u u'^2, \tag{2.68}$$

from Equation 2.29. Note that the coordinates u and u' correspond to the betatron coordinates, which do not include the offset and slope caused by the momentum dispersion. By using the above definition of the emittance, we obtain an expression for the variation of the emittance with respect to the transverse coordinates u and u'

$$\delta\epsilon_u = (2\gamma_u u + 2\alpha_u u')\delta u + (2\alpha_u u + 2\beta_u u')\delta u'. \tag{2.69}$$

Neither the emission of a photon nor the acceleration immediately changes the transverse displacement of an electron u, leading to $\delta u = 0$. The sum of the synchrotron radiation and the cavity forces results in a variation of transverse momentum

$$\delta\mathbf{p}_\perp = \delta\mathbf{p}_{\mathrm{ph}} + \delta\mathbf{p}_{\mathrm{rf}}. \tag{2.70}$$

Therefore the variation of transverse slope u' is

$$u' + \delta u' = \frac{p_\perp + \delta p_\perp}{p_\parallel} = u' + \frac{\delta p_\perp}{p_\parallel}. \tag{2.71}$$

Furthermore, we have to take into account the influence of the momentum deviation. Therefore, we will first solve the dispersion-free vertical case $\eta_\mathrm{y} = 0$ and then add an additional term for the horizontal case $\eta_\mathrm{x}, where \neq 0$.

With $\delta p_\perp = -y'\delta p_{\mathrm{rf}}$ and $p_\parallel \approx p$, we can conclude for the variation of the vertical coordinates

$$\delta y = 0, \quad \delta y' = -y'\frac{\delta p_{\mathrm{rf}}}{p} = -y'\frac{\delta E_{\mathrm{rf}}}{E_0} \tag{2.72}$$

and obtain an expression of the variation of the vertical emittance

$$\delta\epsilon_\mathrm{y} = -(2\alpha_\mathrm{y} y y' + 2\beta_\mathrm{y} y'^2)\frac{\delta E_{\mathrm{rf}}}{E_0}, \tag{2.73}$$

where δE_{rf} corresponds to the energy refilled by the cavity, not the energy decrease due to the damping $\delta E = c\sqrt{p_{\parallel}^2 + (p_{\perp} + \delta p_{\perp})^2} - c\sqrt{p_{\parallel}^2 + p_{\perp}^2} \approx c\frac{p_{\perp}\delta p_{\perp}}{p} \approx cy'\delta p_{\perp}$. Under the assumption of a uniform distribution of betatron phases ψ, we must average over Equation 2.73. Therefore we solve the integrals

$$\begin{aligned}\langle yy' \rangle &= \frac{1}{2\pi}\int_0^{2\pi} yy' \mathrm{d}\psi \\ &= -\frac{1}{2\pi}\int_0^{2\pi} \epsilon_{\mathrm{y}}(\alpha_{\mathrm{y}}(\cos\psi)^2 + \sin\psi\cos\psi)\mathrm{d}\psi \\ &= -\frac{\epsilon_{\mathrm{y}}\alpha_{\mathrm{y}}}{2}\end{aligned} \tag{2.74}$$

and

$$\begin{aligned}\langle y'^2 \rangle &= \frac{1}{2\pi}\int_0^{2\pi} y'^2 \mathrm{d}\psi \\ &= \frac{1}{2\pi}\int_0^{2\pi} \frac{\epsilon_{\mathrm{y}}}{\beta_{\mathrm{y}}}(\alpha_{\mathrm{y}}^2(\cos\psi)^2 + \alpha_{\mathrm{y}}\sin\psi\cos\psi + (\sin\psi)^2)\mathrm{d}\psi \\ &= \frac{\epsilon_{\mathrm{y}}}{2\beta_{\mathrm{y}}}(\alpha_{\mathrm{y}}^2 + 1) \\ &= \frac{\epsilon_{\mathrm{y}}\gamma_{\mathrm{y}}}{2}\end{aligned} \tag{2.75}$$

and substitute them into Equation 2.73:

$$\delta\epsilon_{\mathrm{y}} = \epsilon_{\mathrm{y}}(\alpha_{\mathrm{y}}^2 - \beta_{\mathrm{y}}\gamma_{\mathrm{y}})\frac{\delta E_{\mathrm{rf}}}{E_0} = -\epsilon_{\mathrm{y}}\frac{\delta E_{\mathrm{rf}}}{E_0} \tag{2.76}$$

Over the time of one revolution T_0, the energy chunks δE_{rf} add up to the total radiation loss U_0. Therefore, following our third assumption, we write an expression for the time derivative of the vertical emittance

$$\frac{\mathrm{d}\epsilon_{\mathrm{y}}}{\mathrm{d}t} = \frac{\mathrm{d}\delta\epsilon_{\mathrm{y}}}{\mathrm{d}t} = -\epsilon_{\mathrm{y}}\frac{1}{E_0}\frac{\mathrm{d}\delta E_{\mathrm{rf}}}{\mathrm{d}t} = -\epsilon_{\mathrm{y}}\frac{U_0}{E_0 T_0}. \tag{2.77}$$

For the variation of the horizontal emittance

$$\delta\epsilon_{\mathrm{x}} = -\epsilon_{\mathrm{x}}\frac{U_0}{E_0 T_0} + \delta\epsilon_{\mathrm{x}}^{\delta} \tag{2.78}$$

we have to consider an additional term $\delta\epsilon_{\mathrm{x}}^{\delta}$ caused by the momentum dispersion. As discussed in Section 2.3.4, the total horizontal displacement x_{tot} is sum of betatron displacement $x(s) = \sqrt{\epsilon\beta(s)}\cos(\psi(s)+\psi_0)$ and dispersive displacement $x_\delta(s) = \eta(s)\delta$:

$$x_{\mathrm{tot}}(s) = x(s) + \eta(s)\delta \quad \text{and} \quad x'_{\mathrm{tot}}(s) = x'(s) + \eta'(s)\delta \tag{2.79}$$

As the energy loss due to the emission of a photon cannot immediately change the total displacement x_{tot} or slope x'_{tot} of an electron, the betatron coordinates x and x' must compensate the change in the dispersive coordinates x_δ and x'_δ. Therefore the variation of the betatron coordinates x and x' is

$$\delta x = -x_\delta(s) = -\eta_{\mathrm{x}}\frac{\delta E_{\mathrm{ph}}}{E_0}, \quad \delta x' = -x'_\delta(s) = -\eta'_{\mathrm{x}}\frac{\delta E_{\mathrm{ph}}}{E_0}, \tag{2.80}$$

where δE_{ph} is the variation of the electron energy caused by the synchrotron radiation. Substituting Equation 2.80 into Equation 2.69 yields the additional term of the variation of the horizontal emittance

$$\delta\epsilon_{\mathrm{x}}^{\delta} = -(2\gamma_{\mathrm{x}}x + 2\alpha_{\mathrm{x}}x')\eta_{\mathrm{x}}\frac{\delta E_{\mathrm{ph}}}{E_0} - (2\alpha_{\mathrm{x}}x + 2\beta_{\mathrm{x}}x')\eta'_{\mathrm{x}}\frac{\delta E_{\mathrm{ph}}}{E_0}. \tag{2.81}$$

From [22], we use the expression for the radiation power

$$P_\gamma(s) = C_\gamma\frac{e^2c^3}{2\pi}E^2B^2 = C_\gamma\frac{c}{2\pi}E^4\kappa(s)^2, \tag{2.82}$$

where E and B corresponds to the electric and magnetic fields, respectively, and we introduced a constant

$$C_\gamma = \frac{e^2}{3\epsilon_0(mc^2)^4}. \tag{2.83}$$

As $P \propto E^2B^2$, the radiation power is in linear approximation

$$\begin{aligned} P &\approx P_0 + \frac{\mathrm{d}P}{\mathrm{d}E}\Big|_{E=E_0}\Delta E + \frac{\mathrm{d}P}{\mathrm{d}B}\Big|_{B=B_0}\Delta B \\ &= P_0(1 + 2\frac{\Delta E}{E_0} + 2\frac{\Delta B}{B_0}). \end{aligned} \tag{2.84}$$

Subsituting $\Delta B \approx \frac{\mathrm{d}B}{\mathrm{d}x}x$, $\frac{\mathrm{d}B}{\mathrm{d}x} = \frac{k}{\kappa_{\mathrm{x}0}}B$ and $\Delta E = E_0\delta$ into Equation 2.84, yields

$$P \approx P_0(1 + 2\delta + 2\frac{k}{\kappa_{\mathrm{x}0}}x). \tag{2.85}$$

With $t = z/c$ and Equation 2.85 we can write for the variation of the Energy

$$\delta E_{\mathrm{ph}} = -\frac{P}{c}\delta z \approx -\frac{P}{c}(1 + \kappa_{\mathrm{x}_0}x)\delta s \approx -\frac{1}{c}P_0(1 + 2\delta + 2\frac{k}{\kappa_{\mathrm{x}0}}x)(1 + \kappa_{\mathrm{x}_0}x)\delta s. \tag{2.86}$$

Using Equation 2.86, we can average over all betatron phases ψ

$$\begin{aligned}\langle\delta\epsilon_{\mathrm{x}}^{\delta}\rangle &= -\frac{1}{2\pi}\int_0^{2\pi}((2\gamma_{\mathrm{x}}x + 2\alpha_{\mathrm{x}}x')\eta_{\mathrm{x}} + (2\alpha_{\mathrm{x}}x + 2\beta_{\mathrm{x}}x')\eta_{\mathrm{x}}')\frac{\delta E}{E_0}\mathrm{d}\psi \\ &= \frac{P_0\delta s}{\pi c E_0}\int_0^{2\pi}((\gamma_{\mathrm{x}}x + \alpha_{\mathrm{x}}x')\eta_{\mathrm{x}} + (\alpha_{\mathrm{x}}x + \beta_{\mathrm{x}}x')\eta_{\mathrm{x}}')(1 + 2\delta + 2\frac{k}{\kappa_{\mathrm{x}0}}x)(1 + \kappa_{\mathrm{x}_0}x)\mathrm{d}\psi.\end{aligned} \tag{2.87}$$

Only even terms in x and x' will contribute a finite value, while odd terms in x and x' vanish. Analogous to Equation 2.74 we find that $\langle xx'\rangle = -\frac{\epsilon_{\mathrm{x}}\alpha_{\mathrm{x}}}{2}$ and for average of x^2 we obtain

$$\langle x^2\rangle = \frac{1}{2\pi}\int_0^{2\pi}x^2\mathrm{d}\psi = \frac{1}{2\pi}\int_0^{2\pi}\epsilon_{\mathrm{x}}\beta_{\mathrm{x}}(\cos\psi)^2\mathrm{d}\psi = \frac{\epsilon_{\mathrm{x}}\beta_{\mathrm{x}}}{2}, \tag{2.88}$$

which we substitute into Equation 2.86:

$$\begin{aligned}\langle\delta\epsilon_{\mathrm{x}}^{\delta}\rangle &= \frac{P_0\delta s}{\pi c E_0}\int_0^{2\pi}((\gamma_{\mathrm{x}}x^2 + \alpha_{\mathrm{x}}xx')\eta_{\mathrm{x}} + (\alpha_{\mathrm{x}}x^2 + \beta_{\mathrm{x}}xx')\eta_{\mathrm{x}}')(\kappa_{\mathrm{x}0} + 2\frac{k}{\kappa_{\mathrm{x}0}})\mathrm{d}\psi \\ &= \frac{P_0\delta s}{cE_0}((\gamma_{\mathrm{x}}\epsilon_{\mathrm{x}}\beta_{\mathrm{x}} - \alpha_{\mathrm{x}}\epsilon_{\mathrm{x}}\alpha_{\mathrm{x}})\eta_{\mathrm{x}} + (\alpha_{\mathrm{x}}\epsilon_{\mathrm{x}}\beta_{\mathrm{x}} - \beta_{\mathrm{x}}\epsilon_{\mathrm{x}}\alpha_{\mathrm{x}})\eta_{\mathrm{x}}')(\kappa_{\mathrm{x}0} + 2\frac{k}{\kappa_{\mathrm{x}0}}) \\ &= \frac{P_0\delta s}{cE_0}\epsilon_{\mathrm{x}}(\gamma_{\mathrm{x}}\beta_{\mathrm{x}} - \alpha_{\mathrm{x}}^2)\eta_{\mathrm{x}}(\kappa_{\mathrm{x}0} + 2\frac{k}{\kappa_{\mathrm{x}0}}) \\ &= \frac{P_0\delta s}{cE_0}\epsilon_{\mathrm{x}}\eta_{\mathrm{x}}(\kappa_{\mathrm{x}0} + 2\frac{k}{\kappa_{\mathrm{x}0}})\end{aligned} \tag{2.89}$$

Using the relationships $\mathrm{d}t = \mathrm{d}z/c$, we define nominal radiation loss per turn

$$U_0 = \int_0^{T_0}P_0\mathrm{d}t = \frac{1}{c}\int_0^{C_0}P_0\mathrm{d}s = \frac{C_\gamma E_0^4}{2\pi}\int_0^{C_0}\kappa_{\mathrm{x}0}(s)^2\mathrm{d}s, \tag{2.90}$$

where we made use of the fact that for the nominal particle $s = z$. By comparing Equation 2.82 and Equation 2.90, we obtain an expression for the nominal radiation power

$$P_0(s) = \text{const} \cdot \kappa_{\text{x0}}(s)^2 = \frac{U_0 c}{\int_0^{\text{C}} \kappa_{\text{x0}}(\text{s})^2 \text{ds}} \kappa_{\text{x0}}(s)^2 \tag{2.91}$$

in terms of the curvature of the ideal orbit κ_{x0} and the nominal energy loss per turn U_0. Substituting Equation 2.91 into Equation 2.89 and integrating over one turn yields the change of the horizontal emittance caused by radiation damping

$$\begin{aligned} \frac{\text{d}\epsilon_{\text{x}}}{\text{d}t} &= -\epsilon_{\text{x}} \frac{U_0}{E_0 T_0} \left(1 - \frac{\int_0^C \kappa_{\text{x0}} \eta_{\text{x}} \left(\kappa_{\text{x0}}^2 + 2k \right) \text{d}s}{\int_0^C \kappa_{\text{x0}}^2 \text{d}s} \right) \\ &= -\epsilon_{\text{x}} \frac{U_0}{E_0 T_0} \left(1 - \frac{I_4}{I_2} \right), \end{aligned} \tag{2.92}$$

where we inserted the second and fourth synchrotron radiation integrals defined in Equation 2.63 and Equation 2.65.

2.5.2 Quantum Excitation

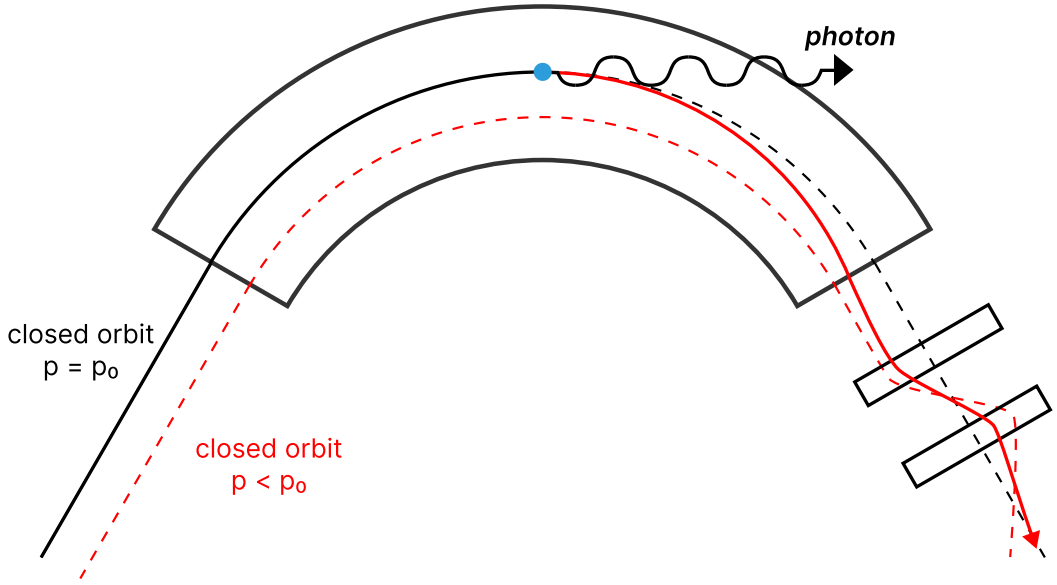

Figure 2.7: Quantum excitation of the betatron oscillation. When an electron with the reference momentum $p = p_0$ emits a photon within a bending magnet, it loses energy and will no longer be on the on-energy closed orbit. It will now perform betatron oscillations around a dispersive closed orbit $p < p_0$. (based on [16])

Until now, we assumed the radiation would be emitted continuously along a given path length parallel to the particle's momentum. If this were the case, then after some time, the betatron amplitude would damp to zero. However, due to the quantum nature of photons, the radiation is not emitted continuously but rather happens in discrete chunks of energy, resulting in an excitation of the emittance called *quantum excitation*. Illustrated in Figure 2.7, the discontinuous jump in energy caused by the emission of a photon of discrete energy forces the electron on a dispersive orbit $p \neq p_0$. Subsequently, the electron now performs betatron oscillations around the new closed orbit.

Furthermore, the photons are not emitted precisely parallel to the electron's velocity but rather in a cone, resulting in a change in the direction of the momentum. That also excites the

betatron oscillation and prevents the emittance from damping to zero, even if the emission of synchrotron radiation would be continuous. However, as the latter effect is much smaller than the former, we will neglect this effect for the following considerations.

For same argument leading to Equation 2.80, the emission of a photon with the discrete energy δE leads to a variation of the betatron offset and betatron slope:

$$\delta x = -\eta_\mathrm{x} \frac{\delta E}{E_0}, \quad \delta x' = -\eta'_\mathrm{x} \frac{\delta E}{E_0} \tag{2.93}$$

Inserting δx and $\delta x'$ into the definition of the phase space ellipse from Equation 2.68, we obtain an expression for the variation of the emittance caused by the emission of a discrete photon of the energy δE

$$\begin{aligned} \delta \epsilon_\mathrm{x}^\mathrm{q} &= \left(\gamma_\mathrm{x} \eta_\mathrm{x}^2 + 2\alpha_\mathrm{x} \eta_\mathrm{x} \eta'_\mathrm{x} + \beta_\mathrm{x} \eta_\mathrm{x}'^2\right) \left(\frac{\delta E}{E_0}\right)^2 \\ &= \mathcal{H}_\mathrm{x} \left(\frac{\delta E}{E_0}\right)^2, \end{aligned} \tag{2.94}$$

where we substituted the curly $\mathcal{H}$ function defined in Equation 2.67. As the photons are emitted randomly with different energies δE, we have to average Equation 2.94 over the energy distribution of the photons. Therefore, we introduce a new quantity $\dot{n}(\epsilon)$ which denotes number of photons emitted per time unit in the energy interval $[\epsilon, \epsilon + \mathrm{d}\epsilon]$.

According to [22], the squared emitted energy per time unit is

$$\dot{N} \langle \epsilon^2 \rangle = \int_0^\infty \epsilon^2 \dot{n}(\epsilon) \mathrm{d}\epsilon = 2 C_\mathrm{q} \gamma^2 \frac{U_0 E_0}{T_0} \frac{|\kappa_{\mathrm{x}0}|^3}{\langle \kappa_{\mathrm{x}0}^2 \rangle_s}, \tag{2.95}$$

where

$$C_\mathrm{q} = \frac{55}{32\sqrt{3}} \frac{\hbar}{mc} \tag{2.96}$$

is to the so-called *quantum constant*.

Using Equation 2.95, we can integrate Equation 2.94 along the length of the ring to obtain an expression for change of the horizontal emittance caused by the quantum excitation

$$\begin{aligned}\frac{\mathrm{d}\epsilon_\mathrm{x}^\mathrm{q}}{\mathrm{d}t} &= \frac{1}{E_0^2}\int_0^C \mathcal{H}_\mathrm{x}\dot{N}\langle(\delta E)^2\rangle \mathrm{d}s \\ &= \frac{2}{E_0^2}\int_0^C \mathcal{H}_\mathrm{x} C_\mathrm{q}\gamma^2 \frac{U_0 E_0}{T_0}\frac{|\kappa_{\mathrm{x}0}|^3}{\langle\kappa_{\mathrm{x}0}^2\rangle_s}\mathrm{d}s \\ &= 2C_\mathrm{q}\gamma^2\frac{U_0}{E_0 T_0}\frac{\int_0^C \mathcal{H}_\mathrm{x}|\kappa_{\mathrm{x}0}|^3\mathrm{d}s}{\int_0^C \kappa_{\mathrm{x}0}^2\mathrm{d}s} \\ &= 2C_\mathrm{q}\gamma^2\frac{U_0}{E_0 T_0}\frac{I_5}{I_2},\end{aligned} \tag{2.97}$$

where in the last step, we substituted the second and fifth synchrotron radiation integrals I_2 and I_5 defined in Equation 2.63 and Equation 2.66. Note that the change of the emittance is positive, which corresponds to an excitation of the emittance.

2.5.3 Equilibrium Emittance

As discussed in the last two subsections, two counteracting effects are changing the amplitude of the betatron oscillations. The total time derivative of the emittance corresponds to the sum of both, i.e., Equation 2.92 and Equation 2.97:

$$\begin{aligned}\frac{\mathrm{d}\epsilon_\mathrm{x}}{\mathrm{d}t} &= \frac{\mathrm{d}\epsilon_\mathrm{x}^\mathrm{q}}{\mathrm{d}t} + \frac{\mathrm{d}\epsilon_\mathrm{x}^\mathrm{r}}{\mathrm{d}t} \\ &= 2C_\mathrm{q}\gamma^2\frac{U_0}{E_0 T_0}\frac{I_5}{I_2} - \epsilon_\mathrm{x}\frac{U_0}{E_0 T_0}\left(1-\frac{I_4}{I_2}\right) \\ &= \frac{U_0}{E_0 T_0}\left(2C_\mathrm{q}\gamma^2\frac{I_5}{I_2} - \epsilon_\mathrm{x}\left(1-\frac{I_4}{I_2}\right)\right)\end{aligned} \tag{2.98}$$

While the synchrotron damping leads to an exponential decay of the emittance, the quantum excitation is independent of the amplitude causing the emittance to grow constantly. Therefore eventually, both effects become equally strong, leading to a zero time derivative of the emittance

$$\begin{aligned}\left.\frac{\mathrm{d}\epsilon_\mathrm{x}}{\mathrm{d}t}\right|_{\epsilon_\mathrm{x}=\epsilon_\mathrm{x}^\mathrm{eq}} &= 0 \\ \epsilon_\mathrm{x}^\mathrm{eq}\left(1-\frac{I_4}{I_2}\right) &= 2C_\mathrm{q}\gamma^2\frac{I_5}{I_2}\end{aligned} \tag{2.99}$$

at a value known as *equilibrium emittance*

$$\epsilon_{\mathrm{x}}^{\mathrm{eq}} = 2C_{\mathrm{q}}\gamma^2 \frac{I_5}{I_2 - I_4}. \tag{2.100}$$

Note that the magnetic lattice fully defines Equation 2.100. For electron storage rings, this has a very practical consequence: The equilibrium emittance is unrelated and therefore not limited by the emittance of the source. Consequently, any arbitrary injected particle distribution damps to the value of the equilibrium emittance.

In the literature, the expression for the equilibrium emittance often differs by a factor of two from Equation 2.100. The reason is that many books use the root-mean-square of the horizontal betatron displacement

$$\sigma_{\mathrm{x}\beta}(s) = \langle x_\beta(s)^2 \rangle = \frac{1}{2\pi}\int_0^{2\pi} \epsilon_{\mathrm{x}}\beta_{\mathrm{x}}(s)\cos(\psi_{\mathrm{x}}(s) + \psi_{\mathrm{x}0})^2 = \frac{1}{2}\epsilon_{\mathrm{x}}\beta_{\mathrm{x}}(s) \tag{2.101}$$

to establish an alternative definition of the emittance

$$\frac{\sigma_{\mathrm{x}\beta}^{\mathrm{eq}}(s)}{\beta_{\mathrm{x}}(s)} = \frac{1}{2}\epsilon_{\mathrm{x}}^{\mathrm{eq}} = C_{\mathrm{q}}\gamma^2 \frac{I_5}{I_2 - I_4}. \tag{2.102}$$

2.6 Lattice Design

As shown in the last section, the choice of magnetic lattice for a circular electron ring defines the magnitude of synchrotron radiation and the equilibrium emittance of the particle beam. The previous sections introduced the most important analytical parameters of electron beam dynamics. In this section, we want to discuss the concepts introduced at the example of different periodic lattices. At first, we will cover the FODO lattice, which is the most simple, strong-focusing lattice. However, it has some disadvantages, making it an unfavorable choice for modern high-energy synchrotron radiation facilities. Therefore, in the following subsection, we move on to the double bend achromat lattice (DBA). Due to its dispersion-free straights, it is a more suitable choice for insertion devices, and it has compared to the FODO lattice a lower equilibrium emittance. All plots are created by the developed code.

2.6.1 The FODO Lattice

The FODO cell is the simplest possible strong-focusing lattice. It consists out of alternating horizontal and vertical focusing quadrupoles with drift spaces in between, which give the cell its name: A horizontal **F**ocusing quadrupole, a drift space (**0** force), a horizontal **D**efocusing, and a drift space (**0** force). A schematic of the FODO lattice is shown in Figure 2.8.

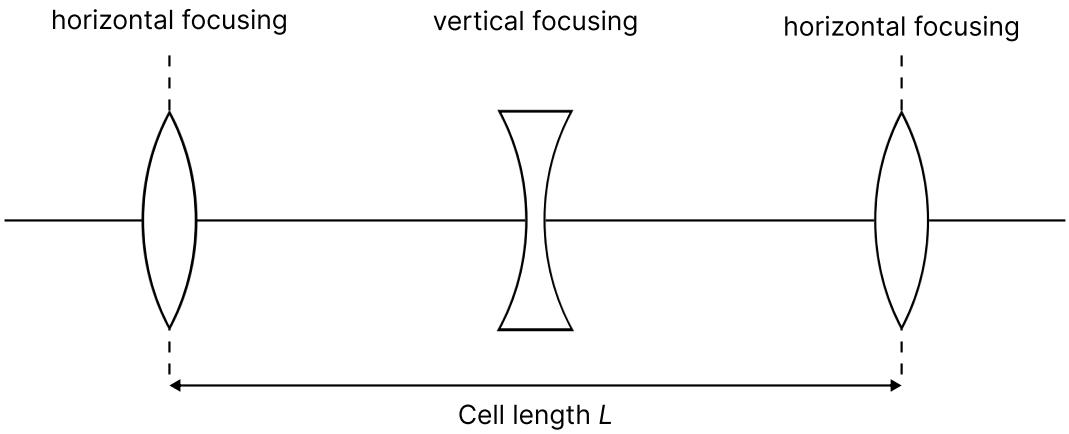

Figure 2.8: Schematic of a FODO cell

We can use the periodic condition of Section 2.3.3 to calculate the optical functions $\beta(s)$ and $\eta(s)$. The periodic solution of the Twiss parameters $\beta(s)$ and $\eta(s)$ for a FODO lattice without dipoles ($R = 0$) are shown in the upper plot of Figure 2.9.

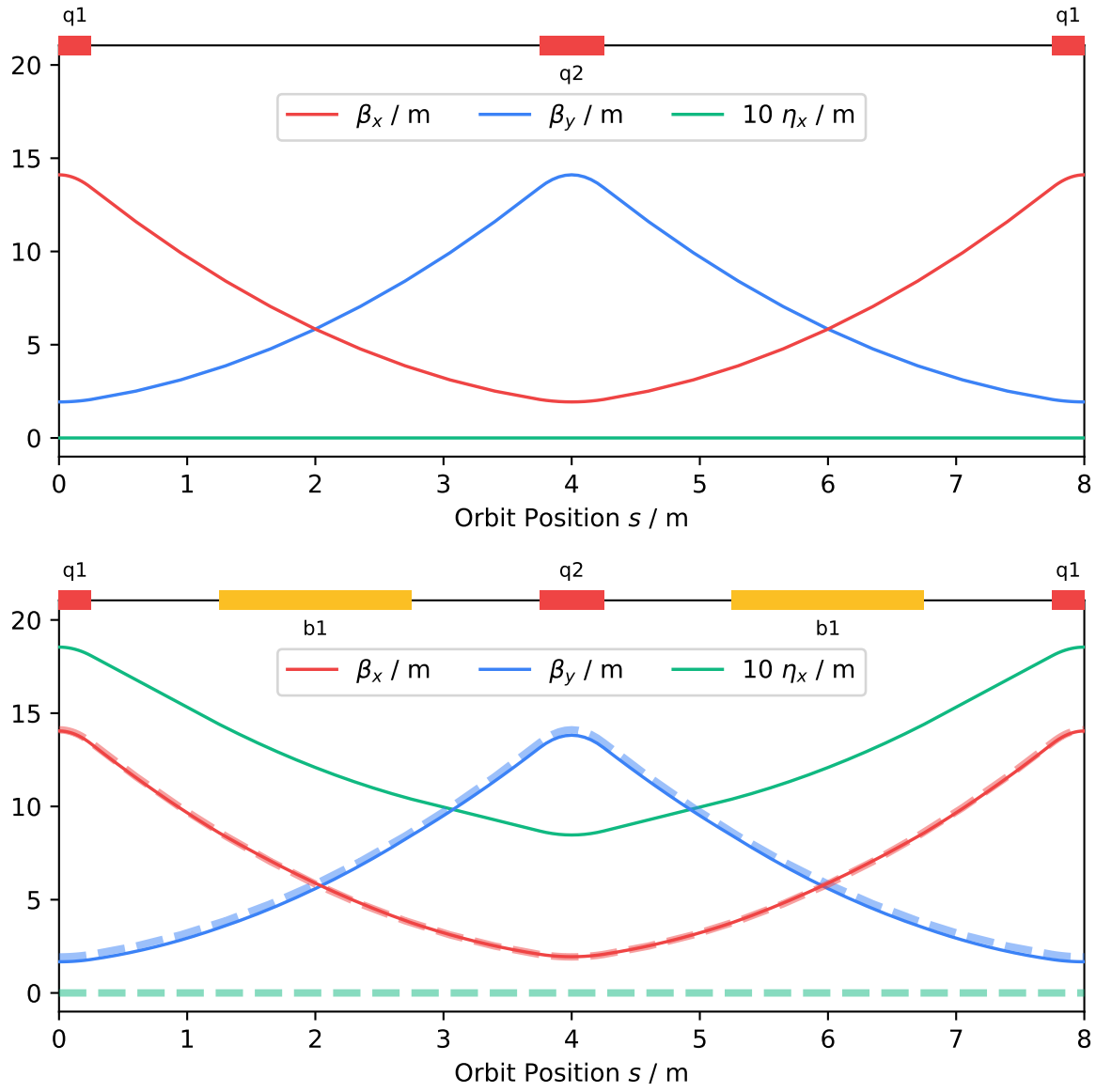

Figure 2.9: Top: Twiss parameters of the FODO cell without dipoles. Bottom: Twiss parameters of a FODO cell with dipoles (solid) compared to a cell without dipoles (dashed, from top).

While the horizontal beta function $\beta_x(s)$ reaches its maximum at the center of the horizontal focusing quadrupole *q1*, the vertical beta function $\beta_y(s)$ has its maximum at the center of the vertical focusing quadrupole *q2*. As the dispersion is introduced by bending magnets, this FODO cell has a vanishing dispersion function $\eta_x(s)$. A high dispersion function is especially undesirable at the location of insertion devices, which would significantly increase the quantum excitation. A circular accelerator must have bending magnets, as - by definition - it has to close at some point, which makes a non-vanishing dispersion function $\eta_x(s)$ inevitable.

However, as discussed in the following subsection, it is still possible to design a lattice with no dispersion within certain sections of the ring.

The simplest way to create a FODO-based circular accelerator is to place a bending magnet in the center of every drift space. The Twiss parameters $\beta(s)$ and $\eta(s)$ for such a FODO cell with dipoles ($R = \frac{\pi}{8}$) are shown in the lower plot of Figure 2.9. Table 2.1 lists a comparison of the lattice parameters for both FODO cells.

Table 2.1: Lattice parameter of the FODO cell from Figure 2.9

	without dipoles	**with dipoles**
Cell length L / m	8.00	8.00
Bending angle φ	0.00	$\frac{\pi}{8}$
Quadrupole strength k / m^{-2}	0.80	0.80
Horizontal tune Q_x	0.28	0.28
Vertical tune Q_y	0.28	0.30
Max. horizontal beta $\beta_{\mathrm{x,max}}$ / m	14.11	14.05
Max. vertical beta $\beta_{\mathrm{y,max}}$ / m	14.11	13.82
Max. dispersion $\eta_{\mathrm{x,max}}$ / m	0.00	1.85

As one can see, the dipoles introduced a non-zero dispersion function $\eta_\mathrm{x}(s)$ with a graph similar to the horizontal beta function: A maximum of 1.8 meters at the center of the horizontal focusing quadrupole *q1* and a minimum of 0.8 meters at the center of the vertical focusing quadrupole *q2*. Due to the horizontal weak-focusing of the dipoles and vertical focusing of the dipole edges, the maxima of the horizontal and vertical beta functions are slightly smaller than in the FODO cell without dipoles. For the same reasons, the vertical tune Q_y is a bit larger.

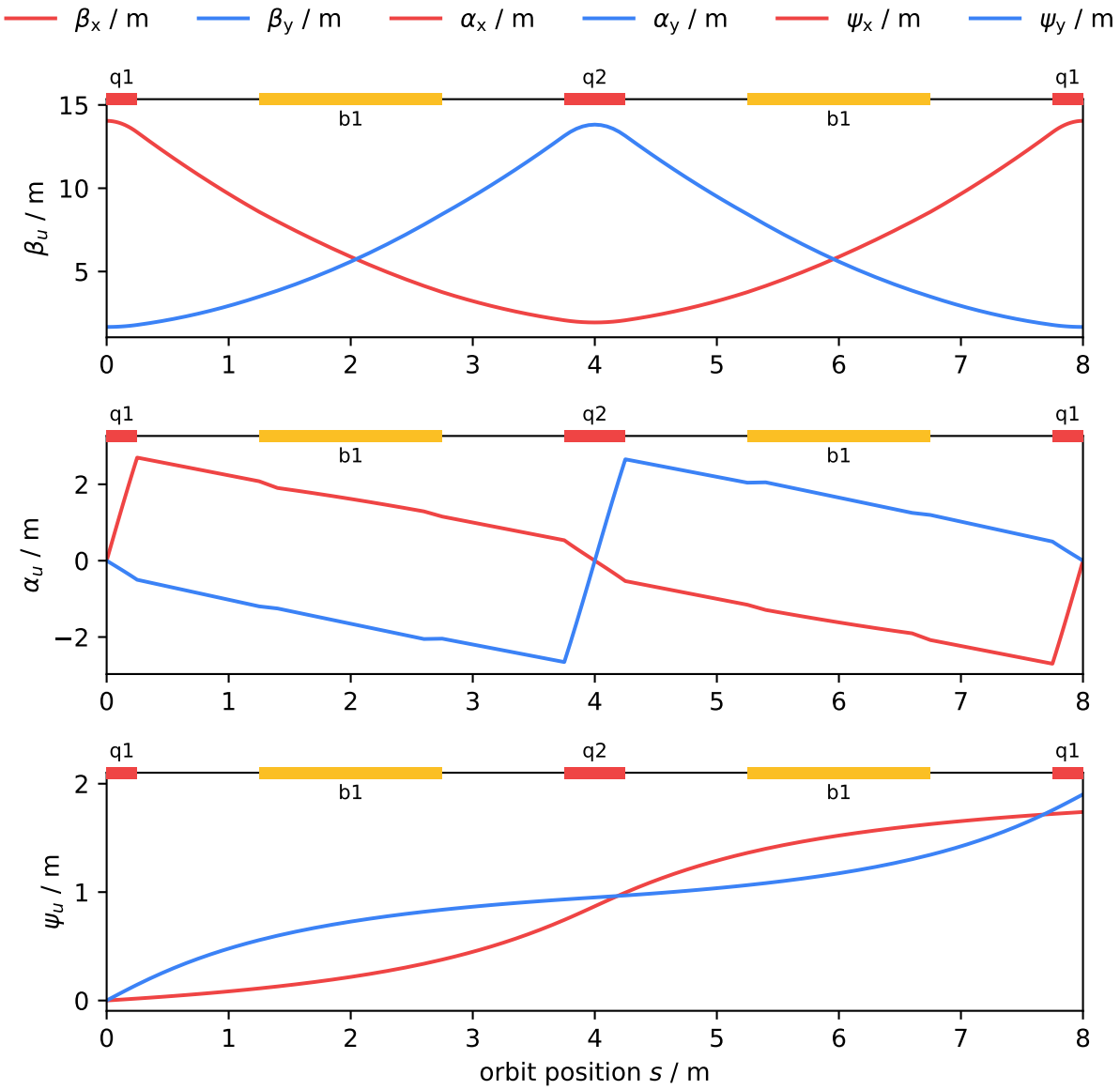

Figure 2.10: Top: The horizontal and vertical beta functions $\beta(s)$. Middle: The horizontal and vertical alpha functions $\alpha(s)$. Bottom: The betatron phase $\psi(s)$

Discussed in Section 2.3.2, the evolution of phase space ellipse of the beam is fully defined by the $\beta_u(s)$, $\alpha_u(s)$ and $\gamma_u(s)$ functions. Figure 2.10 shows the graph of the $\beta_u(s)$ and $\alpha_u(s)$ functions together with betatron phases $\psi_u(s)$. According to their definition $\alpha_u(s) = -\beta'_u(s)/2$, the alpha functions $\alpha_u(s)$ become zero when the beta functions $\beta_u(s)$ are maximum and reach their maximum when the beta functions $\beta_u(s)$ change the most. The horizontal betatron phase $\psi_x(s)$ changes the most in the vertical focusing quadrupole *q2* and its adjacent drift spaces. In contrast, the changes within the starting and ending *q1* quadrupoles are

almost negligible. For the vertical betatron phase $\psi_y(s)$, it is the other way around. However, due to the edge focusing of the bending magnets, the total vertical phase advance Q_y is slightly larger than in the horizontal plane.

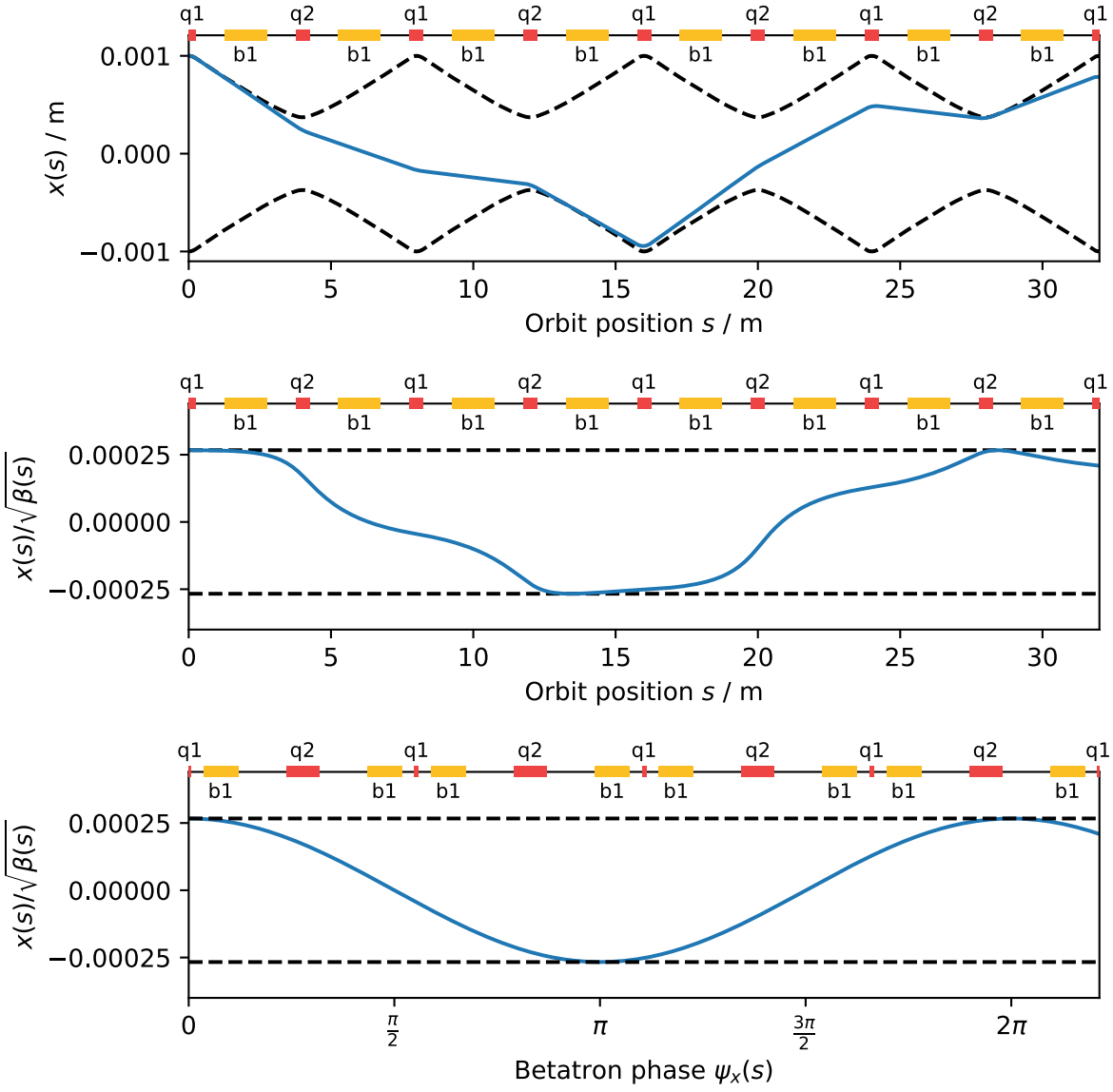

Figure 2.11: Top: exemplary trajectory of a single particle (solid blue) and beam envelope (dashed black). Middle: Normalized particle trajectory along the orbit position (floquet transformation). Bottom: Normalized particle trajectory as a function of the betatron phase.

The product of the periodic beam amplitude $\sqrt{\epsilon\beta(s)}$ and the cosine of the betatron phase $\psi_u(s)$ defines the trajectory of a particle in a strictly linear lattice

$$u(s) = \sqrt{\epsilon\beta_u(s)}\cos\left(\psi_u(s) + \psi_0\right), \tag{2.103}$$

as shown at the example of four consecutive FODO cells in the upper part of Figure 2.11. The particle performs betatron oscillations according to its betatron phase $\psi_u(s) + \psi_{u,0}$ within the beam envelope $\sqrt{\epsilon\beta(s)}$. The second plot of Figure 2.11 shows the same trajectory using Floquet's coordinates $u(s)/\sqrt{\beta(s)}$. In these coordinates, the beam envelope $\sqrt{\epsilon\beta(s)}$ becomes a constant and the particle performs oscillations distorted by the betatron phase $\psi_u(s)$. When plotting the particle trajectory against the betatron phase ψ_u(s) instead of the orbit position s, shown in the third plot of Figure 2.11, the oscillation has a perfect sinusoidal shape. Note that in this representation, the elements of magnetic lattice become distorted according to their phase advance $\psi_u(s_1, s_2)$.

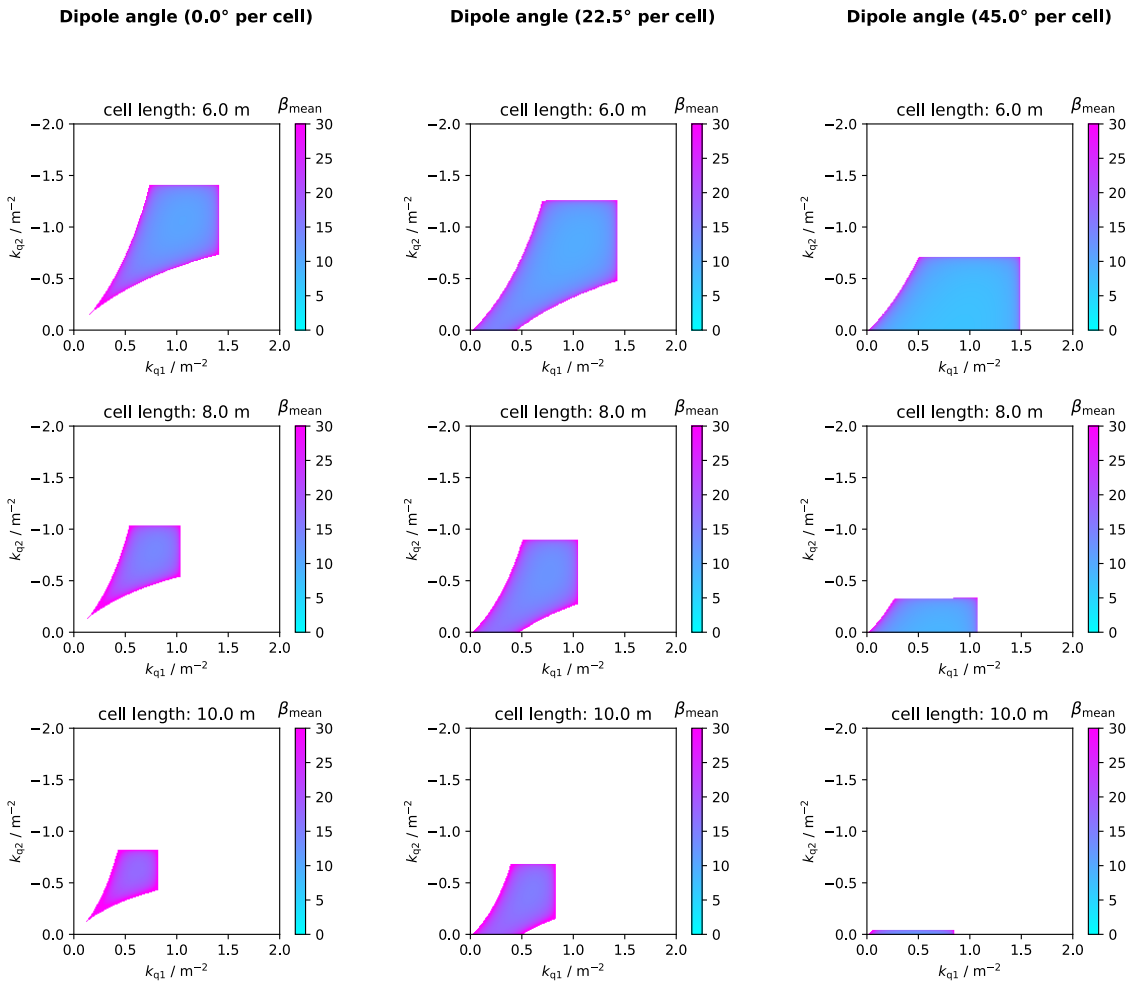

Figure 2.12: Mean beta function for different configurations of a FODO cell. For the FODO lattice, the space of stable quadrupole configurations has the form of a necktie and is therefore often called Necktie-Plot.

Figure 2.12 shows a scan over possible quadrupole settings for FODO cells with different lengths L and deflection angles, where we used the stability condition from Equation 2.40. Not

all configurations of quadrupole strengths result in a stable lattice. One condition is that the focal length of a quadrupole must be greater than its distance to the next quadrupole. Assuming the focal length would be smaller, then the transverse displacement u of a particle and its derivative $\frac{du}{ds}$ would have the same sign at the start of the following quadrupole, leading to a negative feedback loop. For each pass-through, the deflection would become stronger and stronger in the next quadrupole, eventually leading to the loss of the particle beam.

The area of stable solutions for a FODO cell without bending magnets is symmetrical to the diagonal. For low quadrupole strengths, both quadrupoles need to have a similar strength to create a stable lattice. With increasing quadrupole strength, a larger deviation of both quadrupole strengths is possible, leading to the stable area's necktie shape. The maximum quadrupole strength is limited by the condition that the focal length of the quadrupoles must be greater than their distance from each other. The smallest average beta function β_{mean} for a FODO cell with a length of 6 meters without bending magnets is achieved when both quadrupoles are set to about $1.1\,\text{m}^{-2}$.

With increasing cell length the area of stable solutions decreases. This is again due to the requirement that the focal length of the quadrupoles must be larger than their distance. This means that moving the quadrupoles closer together allows for greater field strengths.

The second and third row of Figure 2.12 shows the stability plots for different FODO cells with dipoles with bending angle per cell of $\pi/8$ and $\pi/4$, respectively. The dipoles edges lead to an additional defocusing in the horizontal plane and focusing in the vertical plane. Therefore, the necktie plot is shifted towards smaller values on the axis of the vertical focusing quadrupole *q2* depending on the dipole's deflection angle. Furthermore, dipole edge's vertical focusing widens the space of stable configurations for lower quadrupole strengths, as a stable solution is possible even if the quadrupole *q2* is turned off.

The smallest average beta function β_{mean} for a FODO cell with a deflection angle of $\pi/8$ per cell and with a length of 6 meters is achieved when the horizontal focusing quadrupole is set to $0.8\,\text{m}^{-2}$ and the vertical focusing quadrupole is set to $1.1\,\text{m}^{-2}$. Similar to the FODO cell without dipoles, the area of stable solutions is increases when the cell length decreases.

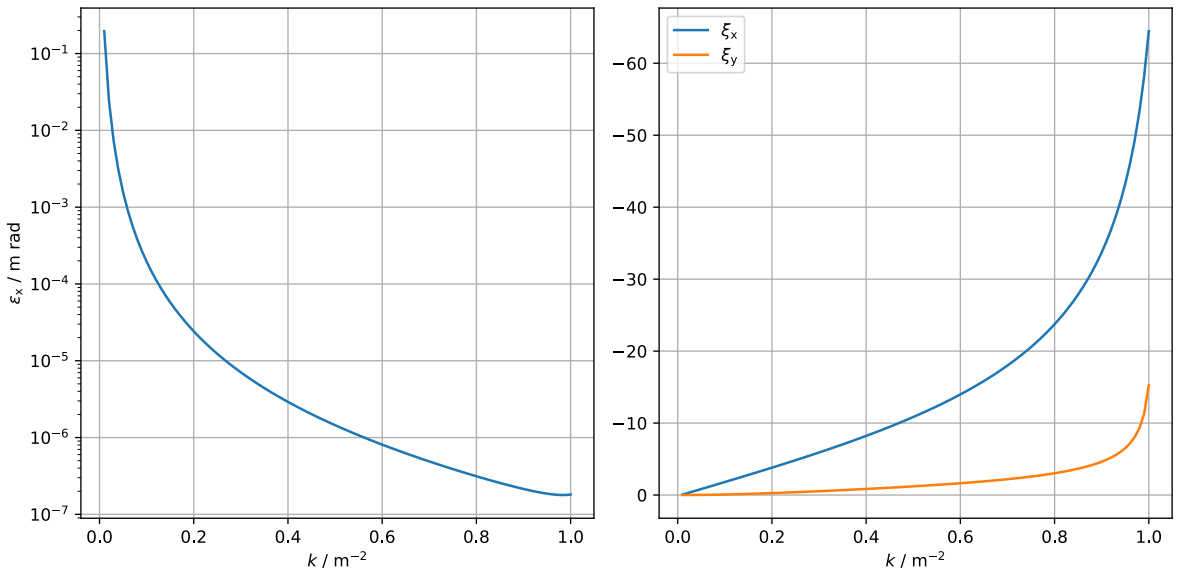

Figure 2.13: Chromaticity and equilibrium emittance of a FODO lattice at 1 GeV.

The left part of Figure 2.13 shows the influence of the quadrupole strength on the equilibrium emittance of a FODO cell, where both the horizontal and vertical focusing quadrupoles have the same strength. As one can see, due to the choice of stronger quadrupole, it is possible to reduce the emittance by multiple orders of magnitude. Unfortunately, as shown in the right part of Figure 2.13, the chromaticity rises with stronger quadrupole values, limiting the achievable emittance as the compensation of the chromaticities leads to a limitation of the dynamic aperture.

All in all, the FODO lattice is less optimal for modern low-emittance synchrotron light sources. For high-energy colliders, the FODO lattice is useful as it maximizes the ratio of bendings magnets per circumference. However, in a synchrotron facility, one generally wants to maximize the ratio of free straights per circumference to utilize insertion devices. Furthermore, the high dispersion function of the FODO lattice leads to a large equilibrium emittance, which is increased even more due to the additional quantum excitation of insertion devices at the location of high dispersion functions. Therefore, the following section presents a lattice with a smaller equilibrium emittance that is more suitable for insertion devices.

2.6.2 The DBA Lattice

As discussed in Section 2.5.2, quantum excitation increases the emittance of the particle beam, which is a crucial quantity for synchrotron radiation users. The *double bend achromat* (DBA) lattice, also known as *Chasman–Green lattice* [24], is one type of lattice to achieve a lower emittance than with the FODO lattice presented in the last subsection. In addition, the DBA lattice's dispersion-free sections are suitable for placing insertion devices, minimizing their contribution to the quantum excitation.

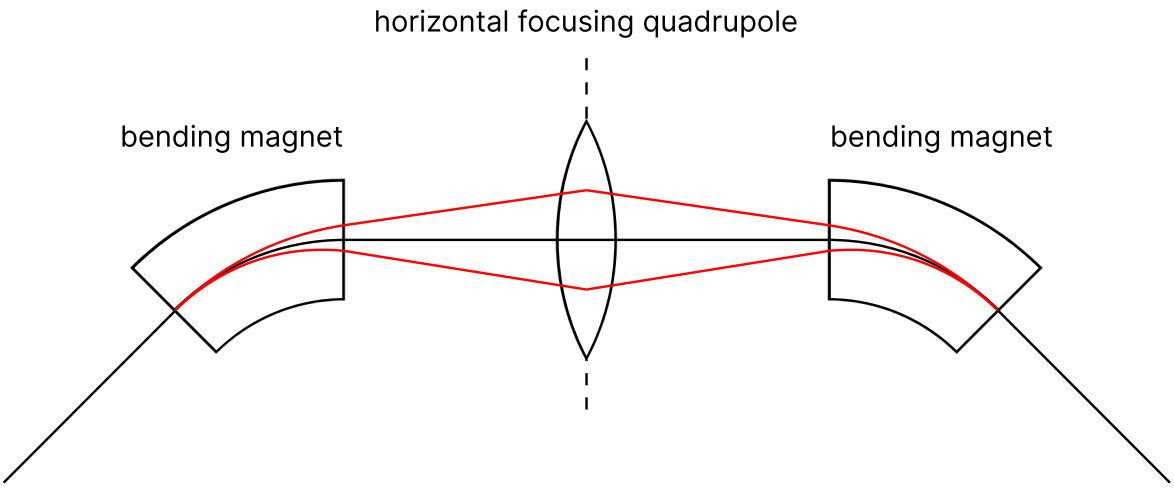

Figure 2.14: Schematic of the DBA. The dispersive particle trajectories are highlighted in red.

Figure 2.14 shows the working principle of a DBA cell: In its simplest form, the DBA consists of two dipoles and one quadrupole. The horizontal focusing quadrupole stands between two bending magnets. The quadruple strength is chosen in such a way that it reverses the gradient of the dispersion function $\eta'_{\mathrm{x}}(s)$ so that the second bending magnet cancels out the dispersion $\eta_{\mathrm{x}}(s)$ introduced by the first dipole.

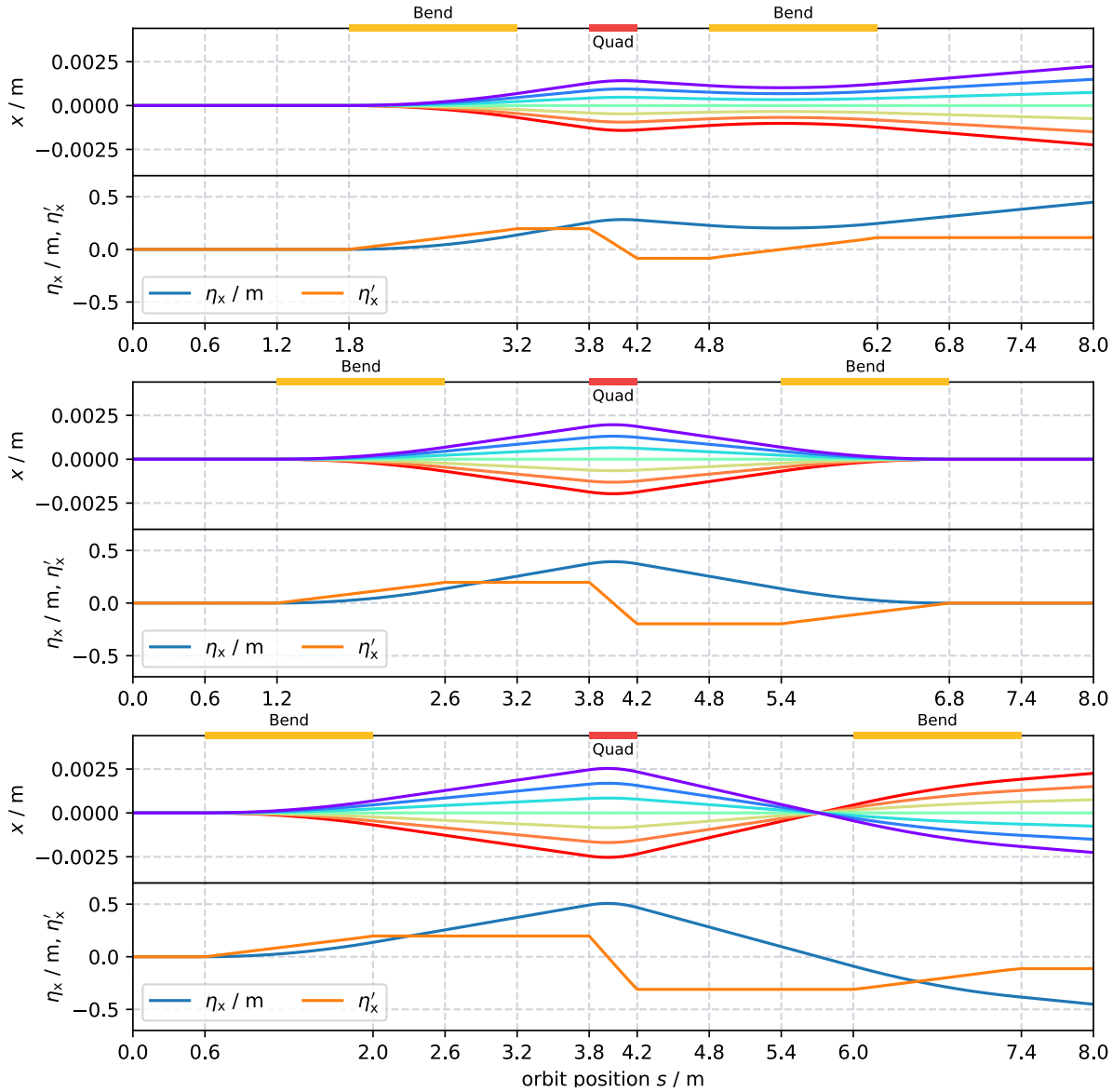

Figure 2.15: Three DBA cells with different distances between bending and quadrupole magnet.

Not only the quadrupole's strength but also its distance to the surrounding dipoles must be chosen correctly to fulfill the achromatic condition. Figure 2.15 shows the dispersion function η_x, its derivative with respect to the orbit position η'_x and the trajectory of multiple dispersive particles for three different DBA cells, where the quadrupole strengths k are identical, but the distance between bending magnets and quadrupole is different. The color of the particle trajectories indicates if the momentum deviation is positive or negative. In analogy with the rainbow spectrum of visible light, a particle trajectory drawn in color towards purple represents a posi-

tive momentum deviation $\delta > 0$. In contrast, a color towards red corresponds to a negative momentum deviation $\delta < 0$.

In the upper plot, the distance between the dipole magnets and the quadrupole is too short. Here, the dispersion function η_x did not have enough space to build up so that the quadrupole could fully reverse the gradient of the dispersion function η_x'. Consequently, after the beam exits the cell, there is a non-vanishing dispersion function $\eta_x > 0$. Alternatively, one could increase the quadrupole strength k to meet the achromatic condition. In the second case, the quadrupole strength k and the distance between the dipole magnets and the quadrupole match perfectly. As a result, the quadrupole exactly reverses the gradient of the dispersion function η_x' so that the dispersion η_x outside the cell vanishes. In the last plot, the distance between the bending magnets and the quadrupole is too large, resulting in a negative dispersion function $\eta_x < 0$ at the end of the cell. In this case, choosing a lower quadrupole strength k would satisfy the achromatic condition.

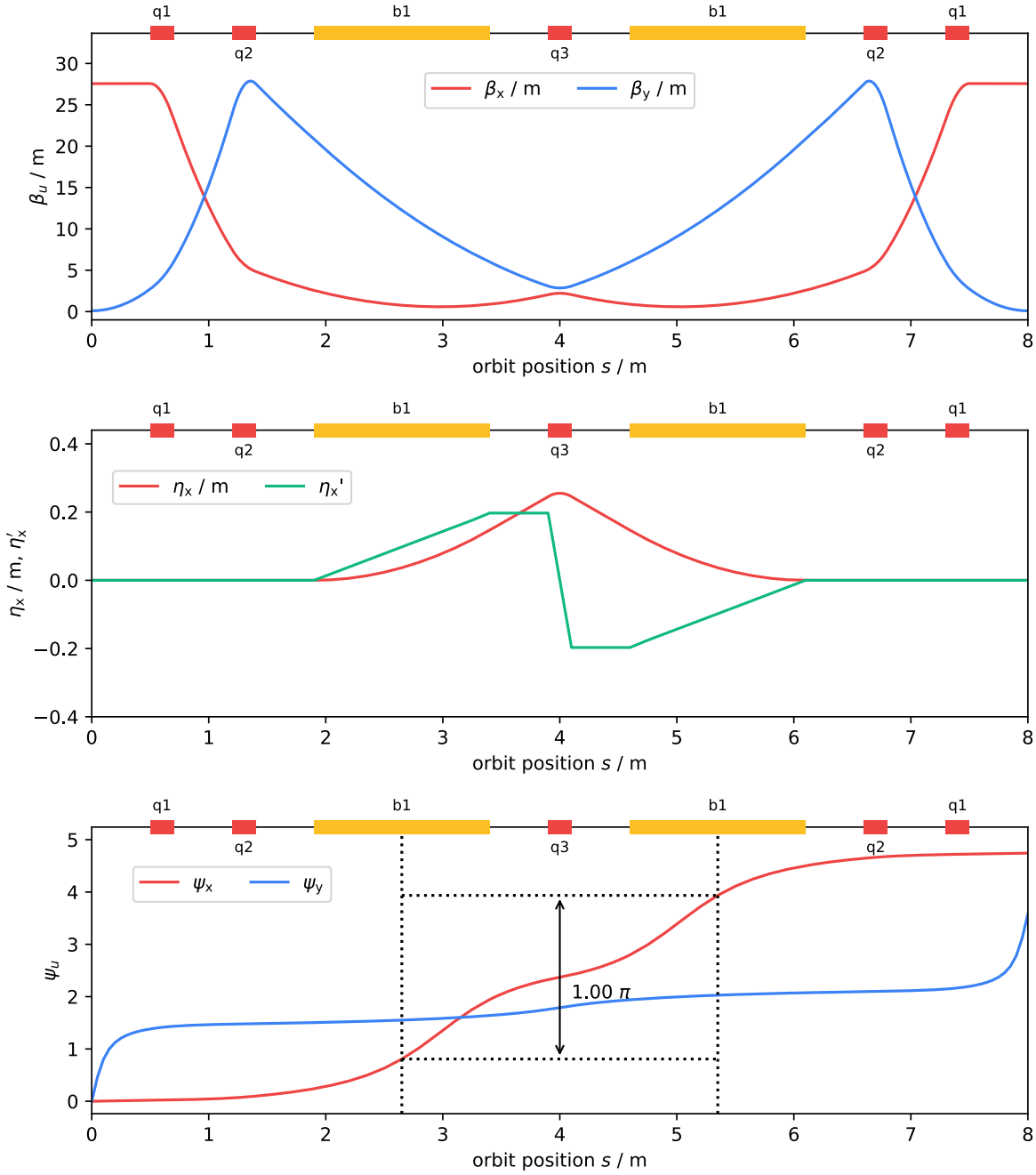

Figure 2.16: Twiss parameters of the DBA cell.

Usually, there are additional quadrupoles outside the achromat to form a complete DBA cell. Figure 2.16 shows the graphs of the beta functions $\beta_u(s)$, the dispersion functions $\eta_x(s)$ and $\eta'_x(s)$ and the betatron phases $\psi_u(s)$ of a DBA cell with two additional quadrupoles *q1* and *q2* outside of the achromat. Additionally, Table 2.2 lists important lattice parameters of this DBA cell.

Table 2.2: DBA lattice parameters

	DBA
Cell length L / m	8.00
Bending angle φ	$\frac{\pi}{16}$
Quadrupole strength k_{Q1} / m^{-2}	4.10
Quadrupole strength k_{Q2} / m^{-2}	-6.05
Quadrupole strength k_{Q3} / m^{-2}	7.82
Horizontal tune Q_{x}	0.75
Vertical tune Q_{y}	0.57
Maximum horizontal beta $\beta_{\mathrm{x,max}}$ / m	27.57
Maximum vertical beta $\beta_{\mathrm{y,max}}$ / m	27.88
Maximum dispersion $\eta_{\mathrm{x,max}}$ / m	0.26

Here the quadrupoles *q1* and *q2* provide the horizontal and vertical focusing of the particle beam. As discussed above, the quadrupole *q3* reverses the gradient of the dispersion function η_{x}'. For the second dipole to have precisely the opposite effect on dispersion as the first, the particles must perform exactly half a betatron oscillation from the center of the first dipole to the center of the second dipole. This condition corresponds to a phase advance of $\psi_{\mathrm{x}}(s_{\mathrm{center}_1}, s_{\mathrm{center}_2}) = \pi$, marked in the bottom plot of Figure 2.16.

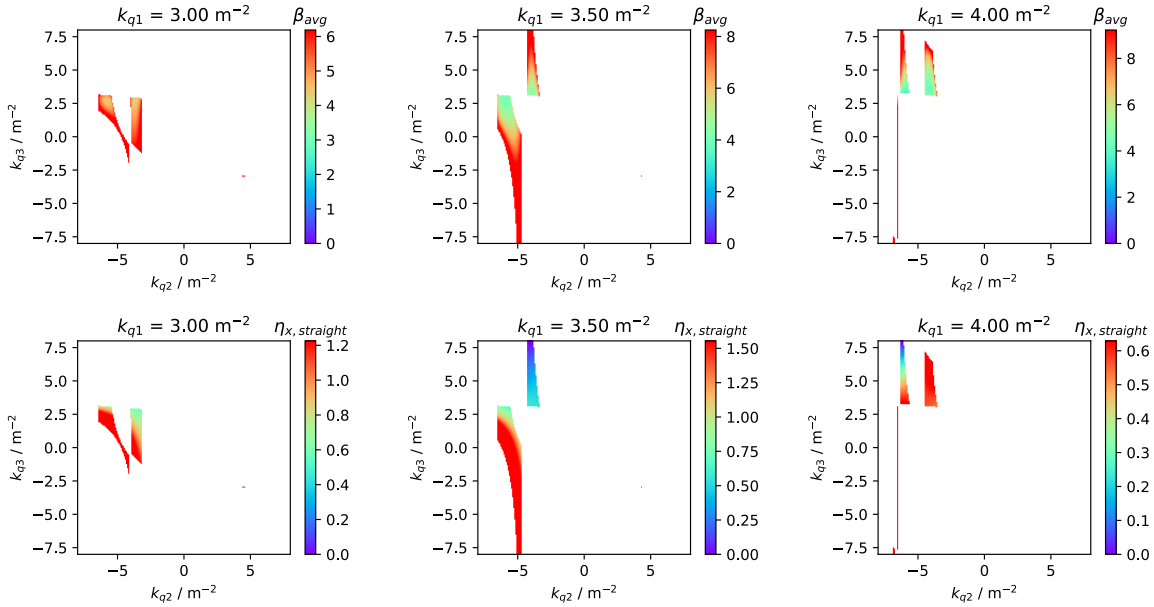

Figure 2.17: Quadrupole scan for a DBA with three quadrupole families. In constrast to the Necktie-plot here, different islands of stable lattice configurations emerge.

Not all quadrupole configurations satisfy the stability condition from Equation 2.40, making it significantly more difficult to find the optimal quadrupole setting for a given objective. In the case of the FODO cell, which has only two parameters, it is relatively straightforward as it is possible to scan over all possible quadrupole settings, as shown in Figure 2.12. Furthermore, even though unstable lattice configurations exist, there is only one contiguous area of stable configurations. However, as shown in Figure 2.17, even for the simple DBA lattice of Figure 2.16, which only has three different quadrupole families, there emerge multiple areas where a stable solution exists. While it is for three quadrupole families still possible to scan over all possible values, this becomes unfeasible for six or more magnets [5]. These discontinuities, leading to the islands of stable lattice configurations, make optimizers only useable within a given island and therefore pose one of the most challenging tasks in lattice development. Section 4.1.3 discusses how some of these problems have been dealt with for the Q5T2off lattice of the BESSY II storage ring.

Overall, the DBA lattice is suitable for a low emittance synchrotron light source and therefore was a very common choice among many third-generation light sources, including BESSY II. However, the multi-bend achromat (MBA) lattice allows for even smaller emittances, making it a popular choice for fourth-generation light sources. The additional quadrupoles

placed between the bending magnets keep the dispersion function further down, resulting in a smaller emittance.

Note that the achromatic condition does not necessarily lead to the smallest equilibrium emittance, as defined in Equation 2.100. As demonstrated in [25], another trend is to break the achromatic condition and leak a finite value of the dispersion function η_x into the straight sections. That effectively distributes the dispersion function η_x along the ring and reduces it within the dipole magnets. As long as the dispersion function η_x does not become too large within the insertion devices, distributing the dispersion function over the entire ring can reduce the equilibrium emittance.

Chapter 3

Implementation

This chapter covers the challenges and considerations made in implementing the developed optics code *apace*. The first section presents a brief overview of the used technologies. Section 3.2 presents the optimizations made to improve the performance of the computation of the Twiss parameters. While a high-level overview of the functionality of the accelerator model is given in Section 3.3, Section 3.4 goes into more detail on a usage example. Finally, Section 3.5 introduces the new JSON-based lattice file format *LatticeJSON*.

The complete documentation of the developed code is provided by Appendix F or at:

https://apace.readthedocs.io

The source code is available at:

https://github.com/andreasfelix/apace

3.1 Overview of Used Technology

The new lattice development tool is implemented in the Python language. While Python's large ecosystem of scientific libraries and high-level syntax, sometimes almost similar to plain English, are major benefits, its dynamic nature makes it necessary to make considerations concerning the performance. For example, in Python, even numbers are rich objects at runtime, and integers can hold values of arbitrary size. In addition, collection types like lists only hold references to objects, leading to non-contiguous chunks of data and, therefore, to less efficient memory layouts. Python's dynamic type system offers much flexibility but provides fewer guarantees, leading to many lookups, inefficient loops, and generally involves more steps for any operation than in a statically typed language. Furthermore, the most widely used Python implementation CPython [14] does not compile to native machine code but instead translates

Python files to an intermediate format called bytecode and then executes it on a virtual machine, leading to a performance penalty.

There are various ongoing efforts to improve the performance of CPython [26]–[28] or to create an alternative faster implementation of the Python language [29], [30]. Nevertheless, even with these optimizations, pure Python cannot compete with low-level languages in numerical performance, making it not well suited to implement a computational-heavy library.

One popular solution to this performance problem is the NumPy package [31], which is so widely used that it can be almost considered part of Python when it comes to scientific computing. NumPy provides a highly efficient multidimensional array type, which is, contrary to Python's built-in list, fixed in size, a contiguous chunk of memory, and restricts all elements to be the same type with a fixed bit length. In addition, NumPy comes with a collection of mathematical routines implemented in lower-level languages like C or Fortran. These routines, like matrix multiplication or element-wise operations, are not run by Python's bytecode interpreter but separated from the language runtime at native speed. Thus, using the NumPy package, scientists get the best of both worlds: Python's flexibility and short development times and the performance of natively compiled libraries when working with numerical data.

All data represented by arrays like the beta or dispersion functions utilize NumPy's array type in the developed code. A limiting factor is when an operation is not expressible by a single NumPy function but has to be composed out of many function calls. Crossing the boundaries between NumPy and Python worlds imposes a performance cost. This performance penalty is minimized the more time is spent within a NumPy routine. For the computation of the Twiss product or the one-turn-matrix of the storage ring, which includes many consecutive multiplications of small 6x6-matrices, the cost of crossing the boundaries, therefore, becomes a limiting factor. For algorithms requiring much indexing, using a Numpy array can even be slower than pure Python code.

Solutions to improve the performance of algorithms in combination with NumPy arrays are the Numba [32] or Pythran [33] compilers, which can translate a subset of the Python language and some NumPy operations to native machine code. However, to achieve maximum performance, it was decided to implement the time-critical parts of the developed code in C. Therefore, the CFFI package [34] was used to call the C functions from Python. Furthermore, to fully utilize the hardware, some of the C routines were written with multithreading using the OpenMP API [35].

With a share of about 95 percent, most of the code is still implemented in Python. All of this should be invisible to a user of the developed library.

Another popular Python package is the matplotlib library [36], which provides a framework to create scientific plots. It is very common when plotting the Twiss parameters to visualize the magnetic lattice or annotate the different accelerator sections. Therefore, the developed code provides some functions for this purpose utilizing the matplotlib framework. Furthermore, a function to draw a floorplan for a given magnetic lattice is included.

3.2 Improving the Performance of the Twiss Calculation

This section discusses some optimizations to improve the performance of the computation of the one-turn-matrix and the Twiss parameters. The steps for calculating the Twiss parameters can be summarized:

1. Choose an arbitrary point on the orbit as an initial position for the periodicity condition
2. Create an array of N elements starting from the initial point

$$[\mathrm{A}, \mathrm{B}, \mathrm{C}, \mathrm{D}, ...] \tag{3.1}$$

3. Map the array of elements to an array of transfer matrices (transfer matrices for individual elements)

$$[\mathbf{R}_\mathrm{A}, \mathbf{R}_\mathrm{B}, \mathbf{R}_\mathrm{C}, \mathbf{R}_\mathrm{D}, ...] \tag{3.2}$$

4. Accumulate the transfer matrices to the different orbit positions (transfer matrics from starting point to orbit position of given element). The last matrix of this array $\mathbf{R}_N$ corresponds to the one-turn-matrix.

$$[\mathbf{R}_1, \mathbf{R}_2, \mathbf{R}_3, ..., \mathbf{R}_N] \text{ with } \mathbf{R}_1 = \mathbf{R}_\mathrm{A},\ \mathbf{R}_2 = \mathbf{R}_\mathrm{B}\mathbf{R}_1,\ \mathbf{R}_3 = \mathbf{R}_\mathrm{C}\mathbf{R}_2, ... \tag{3.3}$$

5. Calculate initial Twiss parameters from one-turn-matrix $\mathbf{R}_N$ using (periodic solution)
6. Calculate the Twiss product of the initial Twiss matrix with the accumulated transfer matrix at every orbit position

$$\mathbf{B}_n = \mathbf{R}_n \cdot \mathbf{B}_0 \cdot \mathbf{R}_n^\mathrm{T}, \tag{3.4}$$

where $\mathbf{B}_0$ corresponds to the Twiss parameters at the initial position.

During a scan or optimization, the Twiss parameters are calculated multiple times in a row, which allows for a relatively simple but effective optimization in step 3: Supposing not every magnet of the lattice changes during each iteration. Then, if the code keeps track of the changed elements, only the transfer matrices for changed elements must be recomputed. The transfer matrices of the remaining elements can be cached.

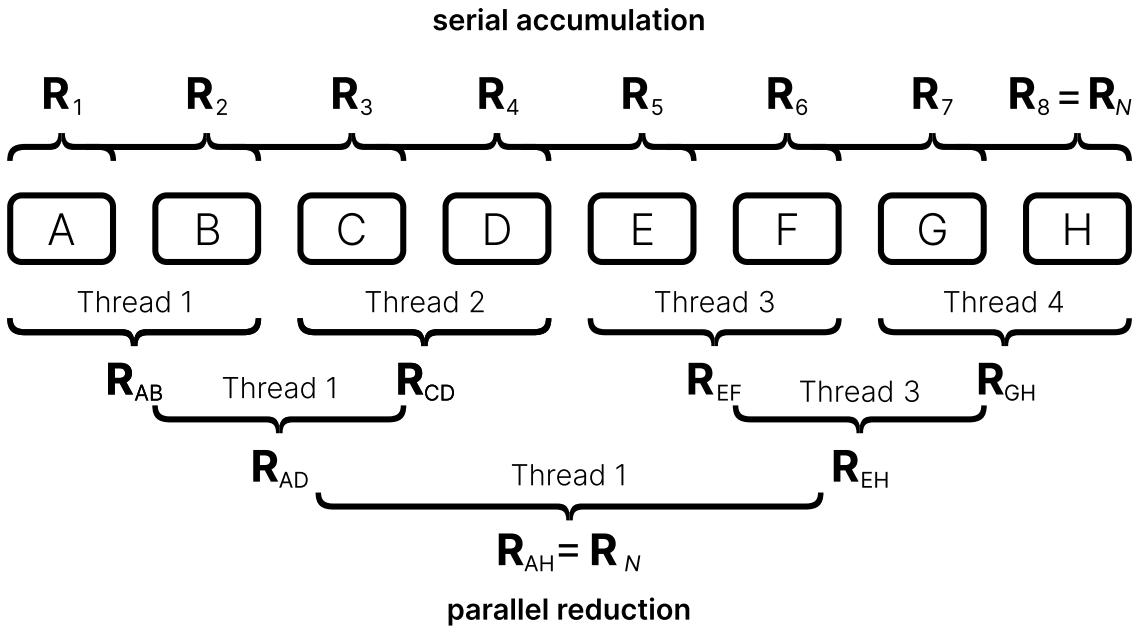

Figure 3.1: The calculation of the accumulated transfer matrices has to be serial (top). Alternatively the one-turn-matrix $\mathbf{R}_N$ can be calculated in parallel.

The calculation of the Twiss parameters is inherently a not fully parallelizable problem because either step 4 or step 6 have to be computed sequentially, leading to two options:

In the first case, calculating the accumulated transfer matrices $\mathbf{R}_n$ for every orbit position makes step 6 parallelizable. However, as each accumulated transfer matrix $\mathbf{R}_n$ depends on the previous transfer matrix $\mathbf{R}_{n-1}$, step 4 must be sequential, shown in the upper part of Figure 3.1.

Alternatively, the calculation of the accumulated transfer matrices could be skipped, and the one-turn-matrix $\mathbf{R}_N$ could be computed in parallel as shown in the bottom part of Figure 3.1. But, this would make it necessary to calculate the Twiss product using the individual transfer matrices, for example

$$\mathbf{B}_3 = \mathbf{R}_\mathrm{C} \cdot \mathbf{B}_2 \cdot \mathbf{R}_\mathrm{C}^\mathrm{T}, \tag{3.5}$$

which would require step 6 to be sequential.

There are two reasons to choose the first option over the second: First, the Twiss product contains two matrix multiplications, while the accumulation of the transfer matrix contains only one. Consequently, the first option performs more work in parallel. Secondly, in option 1, the Twiss product is fully parallelizable with a step complexity of 1 and limited only by the number of threads. On the other hand, the calculation of the one-turn-matrix using parallel reduction has a step complexity of $\log_2(N)$.

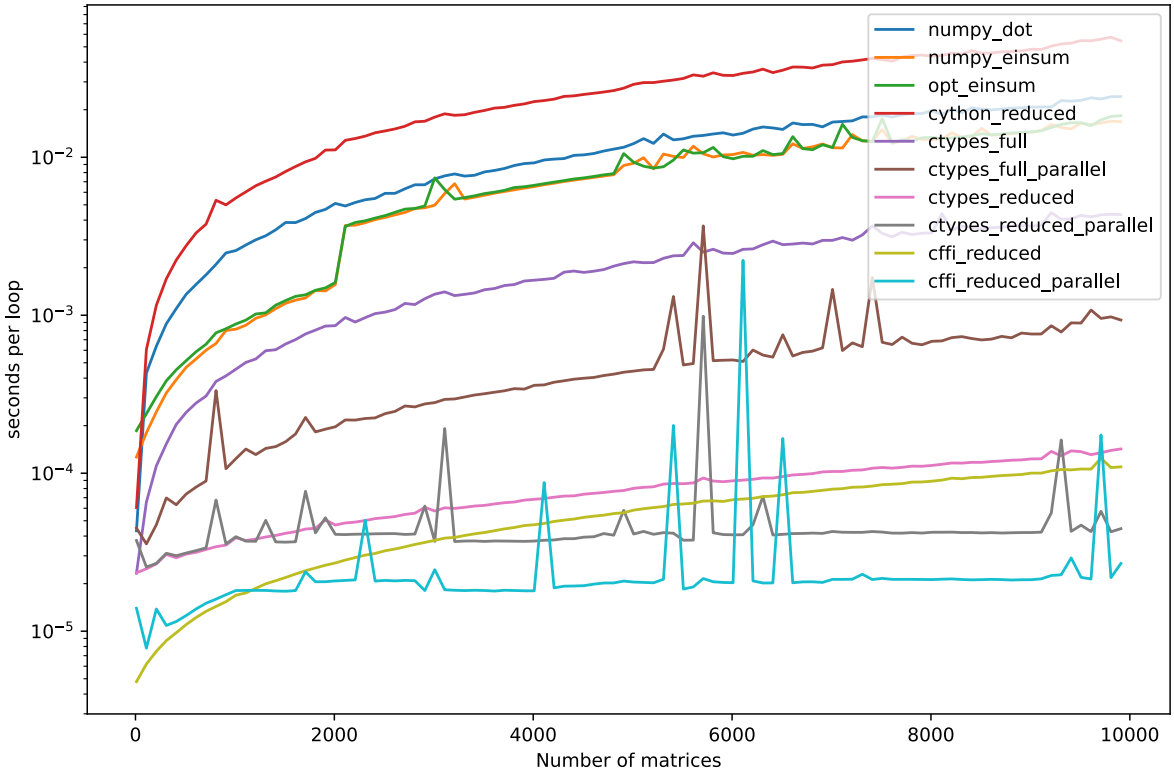

Figure 3.2: Performance comparison of different implemenations of the Twiss product

As most of the time is spent in steps 4 and 6, it is critical to make them as fast as possible. Figure 3.2 shows a benchmark of different implementations of the Twiss product (step 6). `numpy_dot` corresponds to the naive NumPy implementation where the Twiss product is calculated using `numpy.dot`, leading to $2N$ function calls. The `numpy.einsum` or `opt_einsum` functions allow calculating any arbitrary operation expressible in index notation within a single function call. For the Twiss product

$$\mathbf{B}_{nij} = \sum_{kl} \mathbf{R}_{nik} \mathbf{B}_{0kl} \mathbf{R}_{njl} \tag{3.6}$$

this corresponds to:

```
numpy.einsum("nik,kl,njl→nij", matrices, b0, matrices)
```

The Twiss product was also implemented in C. Once as the full matrix product corresponding to Equation 2.36 and once as a reduced version corresponding to Equation 2.37, which takes advantage of the symmetry of the transfer matrices. Furthermore, the C versions were implemented as sequential and parallel versions, taking advantage of all CPU cores. Finally, the C code was called once using the ctypes package, included in the standard library, and once with the aforementioned CFFI package.

The implementations vastly vary in execution speed. The `numpy.einsum` version is below $N = 2000$ significantly faster than the naive NumPy implementation `numpy_dot`. This can probably be attributed to the much fewer function calls. Surprisingly, there is a large performing decrease at about $N \approx 2000$, making the `numpy.einsum` as about as fast as the `numpy_dot` version.

The custom C versions are significantly faster. One reason for that is that the C compiler knows the size of the matrix array is $(N, 6, 6)$ and can therefore produce optimized code for this operation. Furthermore, modern C compilers can automatically take advantage of the *Single Instruction, Multiple Data* (SIMD) instructions shipped with modern CPUs. SIMD instructions allow performing vectorized operations on up to 512 bits at once. By looking at the generated code, it was verified that these SIMD instructions were present in the resulting binary.

Two of the parallel C versions are below $N \approx 1500$ slower than their sequential counterpart. There is some cost in setting up the different threads. If the time spent within the thread is too short, this initial cost limits the executions speed, explaining why the sequential versions are faster for a lower number of matrices N. Looking at the graphs of the parallel version, one can see multiple spikes. This is because the slowest core limits the total runtime of a parallel operation. If the CPU is fully utilized, the operating system might briefly stop one core to schedule a background process, leading one thread to be significantly slower than the others.

Finally, the CFFI package has a lower overhead in calling a C function than the ctypes packages. The difference of this overhead is significant up to $N \approx 5000$. For this reason and because CFFI is compatible with more Python implementations, it was decided to use the CFFI package instead of the built-in ctypes package.

3.3 Representation of the Magnetic Lattice

Besides implementing the physical simulation methods, a major difficulty is the representation of the particle accelerator within the simulation software. The goal is to facilitate the usage of existing optimization and learning algorithms. A good conceptual approach is to model the programming interface after the real machine, in a sense, by creating a digital representation of the accelerator. Different simulation methods like tracking or Twiss calculation implementations are built on top of this digital representation.

Together with a simulation method, the interaction of digital representation of the accelerator and the optimizer can be considered an *environment-agent* relationship, visualized in Figure 3.3. In that sense, the optimizer is an *agent* changing configuration of the *environment*, the digital representation of the accelerator, and observes the new state corresponding to the result of the simulation method. An event system powers the underlying implementation. Supposing the optimizer changes the strength of a quadrupole. Then, the digital representation of the accelerator informs the simulation method that it must recompute a new state. Next, the optimizer accesses the updated state and accordingly takes another action on the environment. These steps repeat until they reach a satisfactory condition.

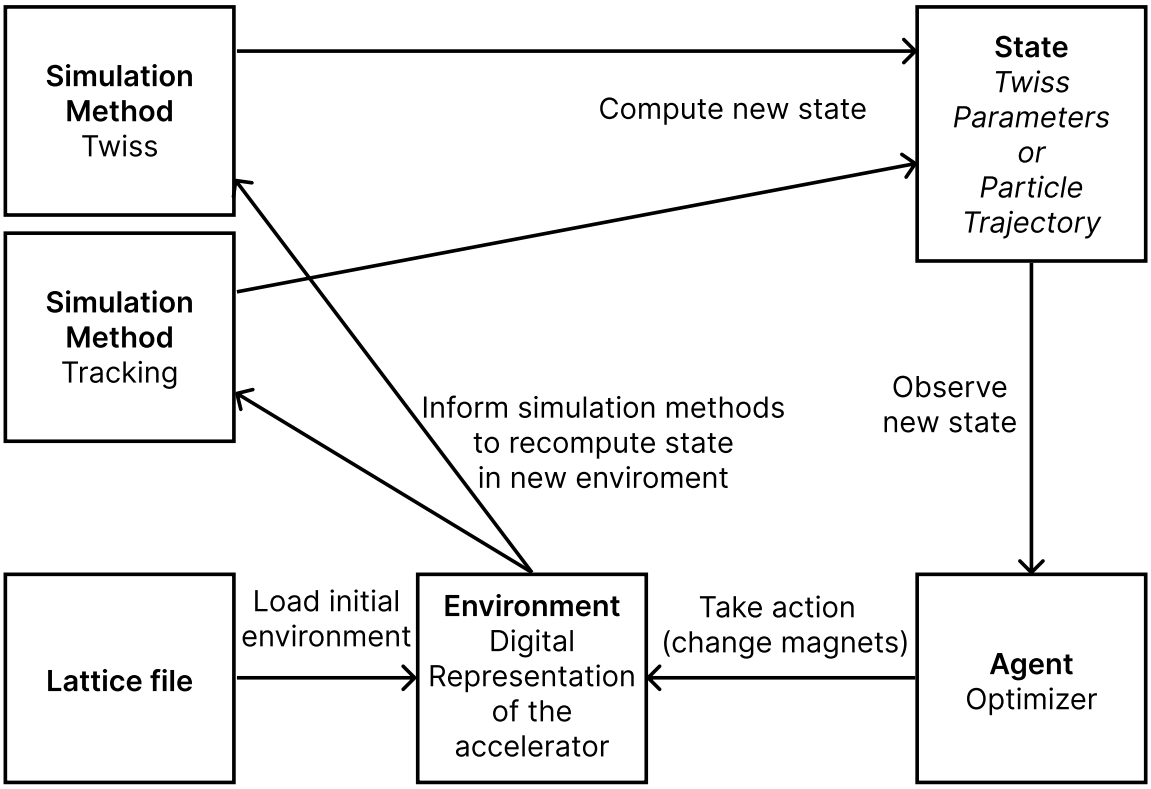

Figure 3.3: Relationship between the lattice file, the digital representation of the accelerator, the simulation methods, and the optimizer.

The description of the accelerator is stored in a so-called lattices file. This file has to be parsed by the simulation software. The different constituents of the accelerator have to be represented by data types available in the given programming language. To provide a convenient interface, this was done in an object-oriented way, primarily using two base classes:

1. `Element` class
2. `Lattice` class

An instance of the `Element` class is a fundamental building block of the particle accelerator. This could be, for example, a `Drift` space, a `Dipole` magnet, or a `Cavity` object. On the other hand, an instance of the `Lattice` class is a sequence of `Element` objects or even other (sub-) `Lattice` objects.

Each element type is a subclass of the `Element` class, which provides common attributes like a `length` or a `name` that uniquely identifies the element. Furthermore, each subclass

implements additional attributes to describe the given element type.

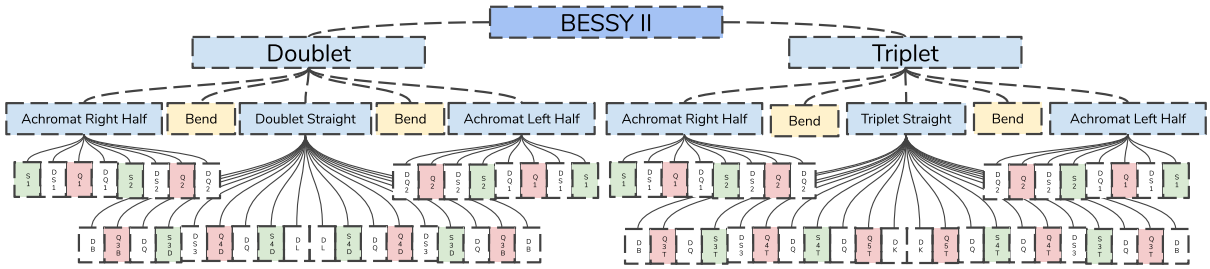

Figure 3.4: A representation of the BESSY II storage ring design lattice. At the top tree is the *BESSY II* lattice, which consists out of the *Doublet* and *Triplet* sub-lattices. The *Doublet* and *Triplet* again consist out of two *Achromat* sub-lattices, one *Straight* sub-lattice and two dipole elements.

The representation of the particle accelerator can be thought of as a tree-like structure, as shown in Figure 3.4, where the `Element` objects are the leaves, the `Lattice` objects are the nodes, and the main `Lattice` object is the root of the tree.

As this object-orient way comes at a price, the simulation methods should use more primitive data structures like arrays. Only the interface to the digital representation of the accelerator should use these higher-level data structures. Moreover, the simulation methods should be implemented as fast as possible.

3.4 Usage

This section gives a small overview of the usage of the developed code. The reader is encouraged to start an interactive Python shell or IPython shell, which has multiline support, and follow along by copying the following code snippets.

The only requirements are Python 3.6 or higher and a C compiler. Then, the most recent version of the developed code, which is 0.1.0 at the moment, can be installed with:

```
pip install -U apace
```

The first step is to import the library:

```
import apace as ap
```

Next, we will build a FODO-ring consisting out of 8 FODO cells. Therefore we create a drift, a bending magnet, one horizontal and one vertical focusing quadrupole.

```
d1 = ap.Drift("d1", length=0.55)
b1 = ap.Dipole("b1", length=1.5, angle=0.392699, e1=0.1963505, e2=0.1963505)
q1 = ap.Quadrupole("q1", length=0.2, k1=1.2)
q2 = ap.Quadrupole("q2", length=0.4, k1=-1.2)
```

The first argument corresponds to the name of the element, while the other parameters are self-explaining. By arranging the just created elements, we can build a FODO cell. Copying this cell eight times, we end up with a ring with 8-fold symmetry.

```
fodo = ap.Lattice("FODO", [q1, d1, b1, d1, q2, d1, b1, d1, q1])
ring = ap.Lattice("RING", 8 * [fodo])
```

To get more information on a specific element, we can use the `print` function to output a table of attributes:

```
>>> print(b1)

type            : Dipole
angle           : 0.392701
e1              : 0.1963505
e2              : 0.1963505
k0              : 0.2618006666666667
length          : 1.5
name            : b1
parent_lattices : {FODO}
radius          : 3.8196999752992733
```

These attributes can be accessed using Python's usual `.` -operator.

```
>>> d1.length
0.55
>>> fodo.length
6.0
```

What is notable is that attributes like the length of a lattice, which depends on its elements' lengths, are automatically updated when one of its elements changes its length.

```
>>> d1.length = 0.6
>>> fodo.length
6.2
```

Next, our goal is to plot the Twiss parameters of the FODO ring. Therefore, we create an instance of the `Twiss` class, which takes a `Lattice` object as its argument.

```
twiss = ap.Twiss(ring)
```

We can use matplotlib to plot the beta and dispersion functions:

```
import apace.plot as aplot
import matplotlib.pyplot as plt

fig, ax = plt.subplots()
ax.plot(twiss.s, twiss.beta_x, label=r"$\beta_\mathrm{x}$ / m")
ax.plot(twiss.s, twiss.beta_y, label=r"$\beta_\mathrm{x}$ / m")
ax.plot(twiss.s, twiss.eta_x, label=r"$\eta_\mathrm{x}$ / m")
ax.set(xlabel="orbit position $s$ / m")
ax.legend(loc="upper center", bbox_to_anchor=(0.5, 1.1), ncol=3, frameon=False)
fig.show()
```

The developed code comes with a set of convenience functions, which allows us to draw the magnetic lattice and annotate the locations of the individual FODO cells.

```
aplot.draw_elements(ax, ring, location="bottom")
aplot.draw_sub_lattices(ax, ring, location="bottom")
```

After calling these functions, we end up with the plot shown in Figure 3.5.

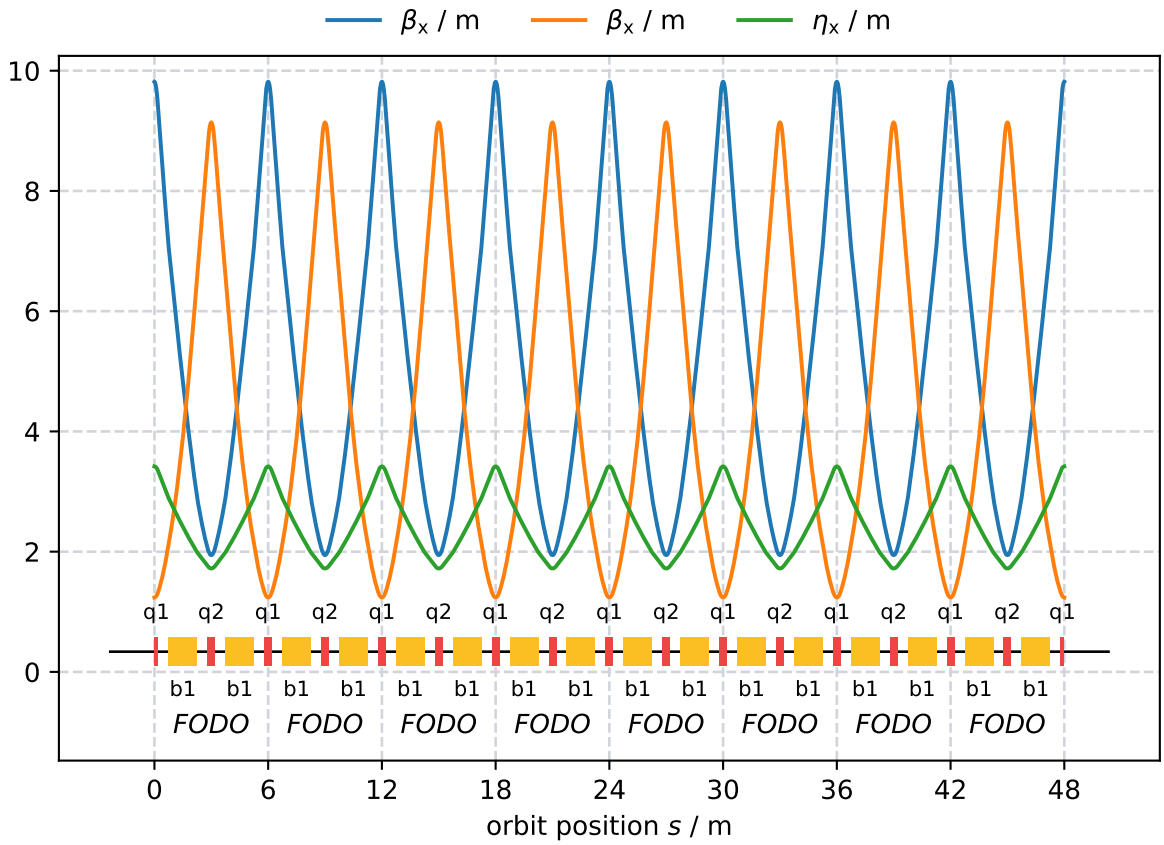

Figure 3.5: Twiss parameters for a FODO ring with 8-fold symmetry

In the following example, we will create the famous necktie plot, which marks the regions of quadrupole configurations leading to a stable lattice. We will use an interval $0 < k < 2$ with a sample size of 100. The results will be stored in a 2-dimensional NumPy array.

```
import numpy as np


samples = 100
start, end = 0, 2
results = np.empty((samples, samples))
interval = np.linspace(start, end, samples)
```

Next, we loop over the quadrupole strength of the horizontal focusing Q1 and vertical focusing Q2 quadrupoles. In the body of the loop we, store the average beta function if a stable solution exists. Otherwise, we store the *not a number* (nan) value.

```
for i, q1.k1 in enumerate(interval):
    for j, q2.k1 in enumerate(-interval):
        try:
            results[i, j] = np.mean([twiss.beta_x, twiss.beta_y])
        except ap.UnstableLatticeError:
            results[i, j] = np.nan
```

Again, we can use matplotlib to visualize the results.

```
import matplotlib.pyplot as plt


fig, ax = plt.subplots()
extent = start, end, start, -end
image = ax.imshow(results.T, extent=extent, origin="lower", vmin=0, vmax=30)
ax.set(xlabel=r"$k_\mathrm{q1}$ / m$^{-2}$", ylabel=r"$k_\mathrm{q2}$ /
        m$^{-2}$")
fig.colorbar(image, ax=ax).ax.set_title(r"$\beta_\mathrm{mean}$")
fig.show()
```

Executing this code gives the plot in Figure 3.6.

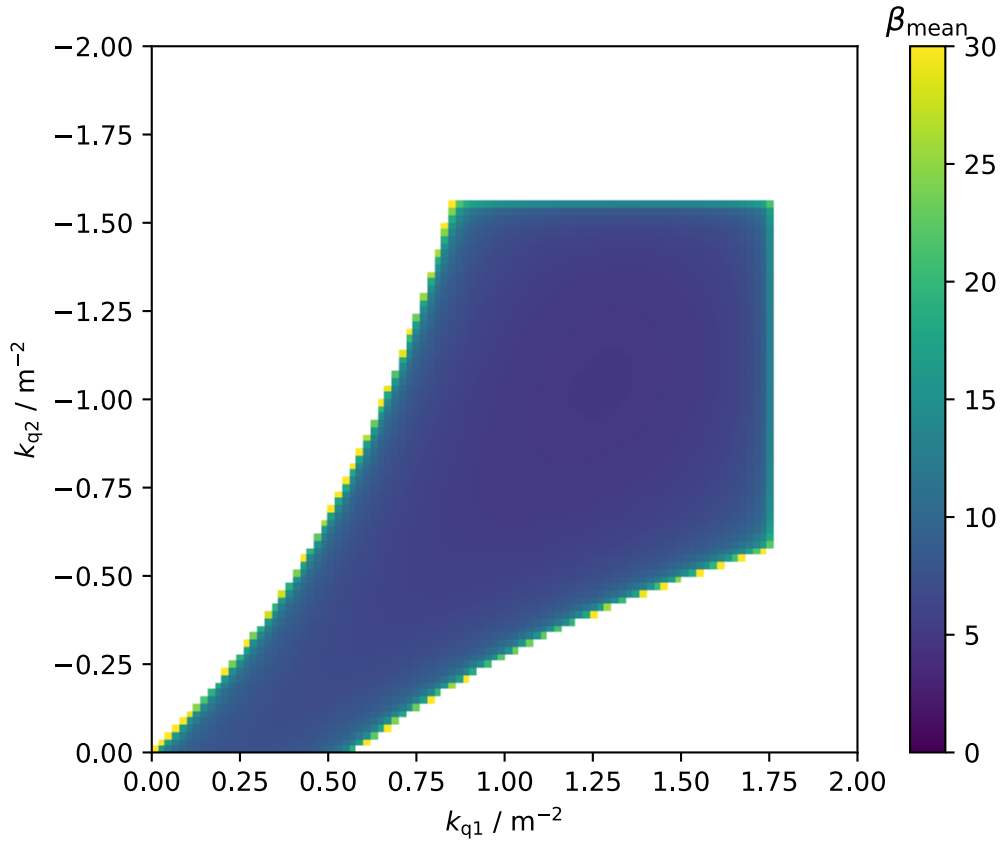

Figure 3.6: Necktie plot - The region of quadrupole configurations leading to a stable lattice has the shape of a necktie

In the last example, we will use a more complex lattice. Lattices can not only be constructed manually as shown above but also loaded from a lattice file. The developed code can load lattice files in the LatticeJSON format, discussed in more detail in the next section, and all formats where converters to the LatticeJSON format exist. At the time of this thesis, these are the elegant and MAD-X lattice file formats.

The `ap.Lattice.from_file` method can be used to create a `Lattice` object from a file. The argument can be a local path or the URL to the lattice file. In our case, we use this function to load the BESSY II standard user lattice from a URL.

```
url = "https://raw.githubusercontent.com/andreasfelix/master-
        thesis/main/code/demo/accelerator-model/bessy2_stduser.json"

bessy2 = ap.Lattice.from_file(url)

twiss = ap.Twiss(bessy2)
```

We obtain a reference to an individual element by passing its name to the BESSY II `Lattice` object. In this case, we want a reference to the Q5 quadrupole in the T2 section.

```
q5t2 = bessy2["Q5T2"]
```

Besides the beta and dispersion functions, the `Twiss` has attributes for the tunes, alpha and gamma functions, emittance, natural chromaticity, and synchrotron radiation integrals. For example, the horizontal and vertical tunes of the BESSY II standard user lattices are given:

```
>>> twiss.tune_x, twiss.tune_y
(17.830955250872567, 6.725123956032014)
```

Also, attributes like the tunes, which depend on the strength of the quadrupoles, are automatically updated when a magnet is changed. For example, reducing the strength of the Q5T2 quadrupole leads to a slightly lower horizontal but slightly larger vertical tune:

```
>>> q5t2.k1 -= 0.1
>>> twiss.tune_x, twiss.tune_y
(17.80617444219928, 6.745445862447529)
```

Finally, the `stable` attribute indicates if a lattice fulfills the stability condition, defined by Equation 2.40.

```
>>> twiss.stable
True
```

Turning off the Q5T2 quadrupole leads to an unstable lattice.

```
>>> q5t2.k1 = 0
>>> twiss.stable
False
```

How to achieve a working machine with the Q5T2 quadrupoles turned off is discussed in Section 4.1.

3.5 LatticeJSON: An Attempt towards a Universal Lattice File Format

The definition of the magnetic lattice is stored in a so-called lattice file. There are several of these data formats, and often only their corresponding simulation software can process them. The variety of lattice file formats is such a big issue of the accelerator physics community that it even has its dedicated Wikipedia section [37]. Several converters between these files can do the heavy lifting but often still require manual adjustments made by a human. Nevertheless, even if the lattice file is available in all common formats, there is still no straightforward way to load the data into an arbitrary programming language. As the most mature accelerator simulation codes come bundled with a scripting language and optimization routines, this was a lesser problem in the past. However, there has been an uprise of powerful scripting languages and modern optimization libraries in the last several years. To leverage these tools, it would be is necessary to have convenient access to the lattice data.

One solution would be to write a robust parser for an existing lattice file format. Candidates would be the MAD-X [8] or elegant [9] lattice file formats, which are very similar but not compatible. Both file formats have the problem that their syntax is ambiguous. Without further context, the parser cannot infer an attribute type. For example in

```
TWISS, FILE=optics;
```

`optics` is a string and not a variable name. In another case

```
RFC: RFCAVITY, HARMON=num;
```

`num` refers to a variable and not a string. In these cases, the parser has to know the type of `FILE` and `HARMON` to parse the file correctly, making implementing a parser non-trivial. Both lattice file formats also support variables and arithmetic expressions. MAD-X even supports more advanced constructs like loops, macros, and if-else statements, making implementing such a parser even more complex and error-prone. If the goal is to be 100% compatible, one would have to reimplement a whole programming language to load the lattice data. A more

feasible solution would be to define a restricted subset of the MAD-X input file, which would contain only data and no logic. Appendix E.1 and Appendix E.2 contain grammar files allowing to load simple elegant and a subset of MAD-X lattice files into Python. They work for simple cases but fail for more advanced lattice files because of the reasons mentioned above.

A better solution for a plain lattice data format would be to define a schema for an existing data format, which already has parsers implemented for the most popular programming languages. One attempt of such a universal lattice file format was the Accelerator Markup language (AML) [38] and the Universal Accelerator Parser (UAP) [39]. The AML is based on the eXtensible Markup Language (XML), which is syntactically similar to HTML, the standard markup language for web documents. AML's goal was to support the evaluation of arithmetic expressions and two different representations of the magnetic lattice: An *unevaluated representation* containing nested sub-lattices and variables and a *flat representation*, where the sub-lattices are expanded into a flattened array of elements. Furthermore, it aimed to support information beyond the lattice description, like control system configurations, magnet history, or other documentary data. This way, AML could be used as an all-in-one solution for an accelerator facility database. Unfortunately, as of March 2020, AML and the UAP are no longer maintained.

The AML had high ambitions, which made it challenging to implement: The number of features, like variables and a complex representation of the magnetic lattice, made it, even though it is based on XML, necessary to implement a custom parser for every programming language.

This thesis takes a simpler approach: The lattice file should not implement variables or other dependency relations. This functionality is already available in programming languages and should not be reimplemented in the lattice file. The definition of a lattice file should not rely on different representations of the magnetic lattice. The lattice file should be easy to load and should not require an additional parser. The primary target of this lattice file should be simulation codes, and the implementation should not be made more complicated by making it suitable for control system usage.

Further requirements are: The lattice file format should be independent of the simulation tool and programming language. The underlying file format should be commonly used, so there is no need to implement it for different programming languages. Furthermore, it should be easy to generate elegant and MAD-X files from this intermediate format. Finally, shown in Figure 3.4, the magnetic lattice has a tree-like structure: It consists of different elements, like drifts, magnets, cavities, and sub-lattices, which in turn consist of other elements. A universal lattice file format should take this structure into account.

Storing the description of the magnetic lattice raises effectively two questions (Figure 3.7):

1. How to store the lattice file persistently on disk?
2. How to represent the lattice file within a programming language?

As data structures and object types vary between programming languages, it is impossible to choose arbitrary data types. However, practically every programming language supports primitives like *strings*, *numbers*, *bools*, *null*, and some containers structures like *arrays* and *key-value stores*. So the goal should be to provide a consistent way to represent the information stored in the lattice files using these generic data types. This representation should not try to invent the next scripting language, but it should be a pure data format. That way, the lattice data can be used by any programming language.

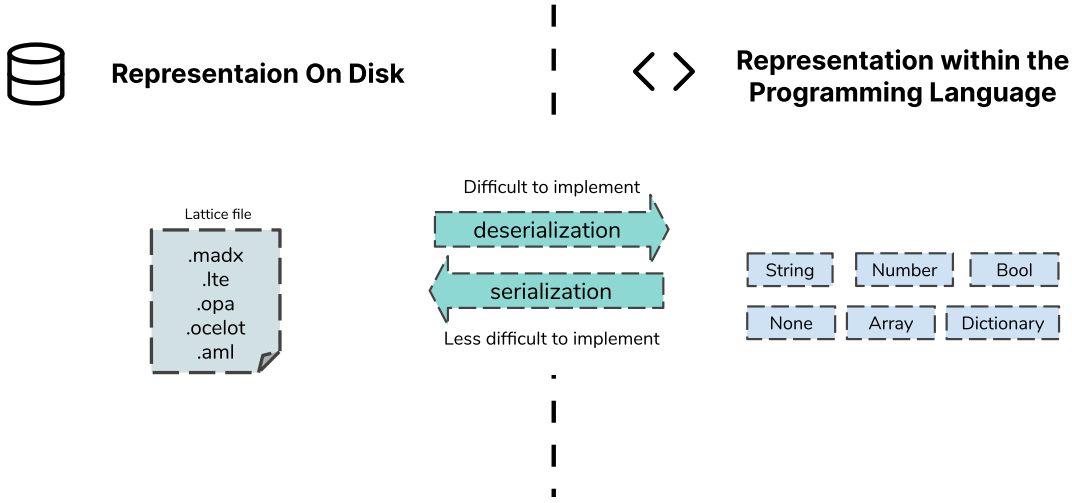

Figure 3.7: The lattice file has to be stored on disk. Therefore the representation of the lattice file within the programming language has to be serialized into a format that can be stored permanently. Conversely, this persistent data format has to be deserialized into a data structure when the lattice is loaded into a program.

Fortunately, these issues are already solved by JSON [40], which is an acronym for JavaScript Object Notation. Even though it has its origins in JavaScript, it is a language-independent and open data file format. It can describe complex data and is the de-facto standard for exchanging data between web applications. There already exist a huge amount of tools for validating and working with JSON.

JSON answers both questions as the is a 1:1 correspondence between JSON data types and the generic data types available in almost all programming languages. Once parsed, the lattice description is represented by the same abstraction of data types as when the lattice information was stored on disk. One problem of AML was, that the parsed lattice data, returned by the

UAP, had a different structure than the stored XML file. So there were effectively two different representations of the same information.

JSON provides a convenient way to serialize and deserialize generic data types. To make use of JSON, the description of the magnetic lattice has still to be defined in terms of these generic data types. Such a definition can be achieved with a so-called JSON schema [41]. A first definition of such a JSON-based lattice file format was done and called LatticeJSON. The LatticeJSON schema file and more information on LatticeJSON are available at [42].

For example, the definition of a FODO lattice as LatticeJSON file is given by:

```
{
  "version": "2.2",
  "title": "FODO Lattice",
  "info": "This is the simplest possible strong focusing lattice.",
  "root": "ring",
  "elements": {
    "d1": ["Drift", {"length": 0.55}],
    "q1": ["Quadrupole", {"length": 0.2, "k1": 1.2}],
    "q2": ["Quadrupole", {"length": 0.4, "k1": -1.2}],
    "b1": ["Dipole", {"length": 1.5, "angle": 0.392701, "e1": 0.1963505, "e2":
        0.1963505}]
  },
  "lattices": {
    "cell": ["q1", "d1", "b1", "d1", "q2", "d1", "b1", "d1", "q1"],
    "ring": ["cell", "cell", "cell", "cell", "cell", "cell", "cell", "cell"]
  }
}
```

To read the quadrupole strength of the Q1 quadrupole in Python, one can use:

```
import json

with open("/path/to/lattice.json") as file:
    lattice = json.load(file)

type_, attributes = lattice["elements"]["Q1"]
print("Quadrupole strength", attributes["k1"])
```

Alternatively, one can use the LatticeJSON library, which validates the lattice file and can load lattices files from URLs.

```
import latticejson
lattice = latticejson.load("https://lattice-database.org/bessy2?mode=stduser")
```

The URL *lattice-database.io* is chosen as an illustrative filler in this example. However, it should be a goal to establish a central database of magnetic lattice files, where the operation mode or file format can be passed as a URL parameter. This would give physicists access to a large number of lattice files without the overhead of maintaining their own set of lattices. Furthermore, as it is straightforward to generate different versions of a JSON file automatically, LatticeJSON could simplify the implementation of such a database significantly.

Chapter 4

Applications

This chapter covers the applications of the developed optics simulation program. The first section comprises the development and optimization of the Q5T2-off optics of the BESSY II storage ring, which enlargens the available installation length of the cavity module of the BESSY-VSR project. The second section shows how the beta functions within a straight section of the BESSY II storage ring were adjusted for an emittance exchange experiment. Finally, the third section presents the development of a framework to automatically generate summaries for a given set of lattice files.

4.1 Modifying the BESSY II Optics for the VSR project

The motivation of enlarging the installation length for the cavity module was already set out in Section 1.2.1. One solution is to turn off the Q5 quadrupoles in the T2 section of the storage ring to gain about 0.7 meters of installation length. The first steps towards such a Q5T2off-optics are described in my bachelor's thesis [5]. With the best solution found, it was possible to achieve a working machine with reasonable injection efficiency and lifetime. The goal should be to improve upon this solution.

The following three subsections provide a summarization of my bachelor's thesis. The first subsection gives an overview of the lattice development of the BESSY II storage ring. The second subsection discusses the constraints for the development of an optics with a turned-off Q5T2 magnet. In Section 4.1.3, the results of the bachelor's thesis are presented. The fourth subsection shows how the Q5T2off-optics was further optimized using the developed code of this thesis. The last subsection summarizes the results of a user operation acceptance test of the improved Q5T2off-optics.

4.1.1 Lattice Design of the BESSY Storage Ring

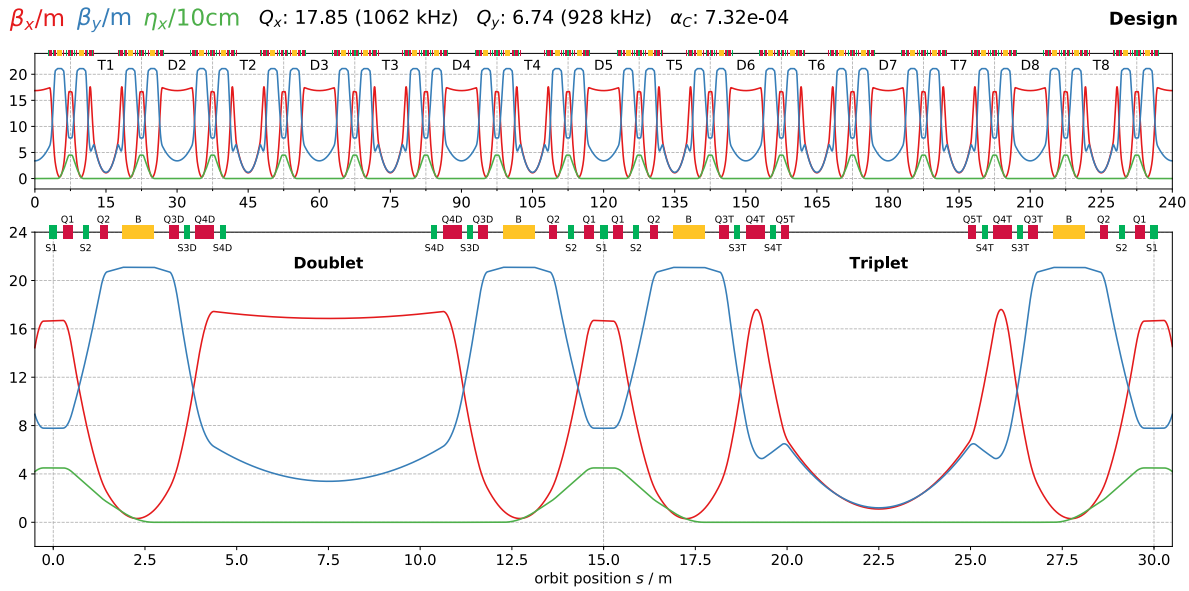

Figure 4.1: The design lattice of the BESSY II storage ring 1996. Full lattice (top), unit cell (bottom) - (extracted from [5])

Table 4.1: The quadrupole strength of the BESSY II design lattice

Magnet	Quadrupole strength k / m²
Q1	+2.45190
Q2	-1.89757
Q3D	-2.02025
Q4D	+1.40816
Q3T	-2.46319
Q4T	+2.62081
Q5T	-2.60000

The BESSY II storage ring has a double bend achromat lattice with 16 straight sections. The injection requires high horizontal beta functions. At the same time, the superconducting wavelength shifter needs a small horizontal beta function. That led to the decision to develop a lat-

tice with alternating high and low horizontal beta straights [43]. As a result, the 240 m storage ring has an 8-fold symmetry with the unit cell shown in Figure 4.1. The seven quadrupole families of the design-lattice are listed in Table 4.1.

The two horizontal focusing Q1 quadrupoles in the center of the DBA reverse the gradient of the dispersion function as discussed in Section 2.6.2. The two vertical focusing Q2 quadrupoles are necessary to limit the vertical beam size. The high beta straights have the quadrupole doublet Q3D and Q4D, which are vertical and horizontal focusing, respectively. The low beta straights have the vertical focusing Q3T, horizontal focusing Q4T, and vertical focusing Q5T quadrupole families, which form a triplet. In order to achieve the low horizontal beta function, the Q4T quadrupoles must be significantly stronger than the Q4D quadrupoles. The additional Q5T quadruples are needed to compensate for the stronger vertical defocusing introduced by the Q4T quadrupoles.

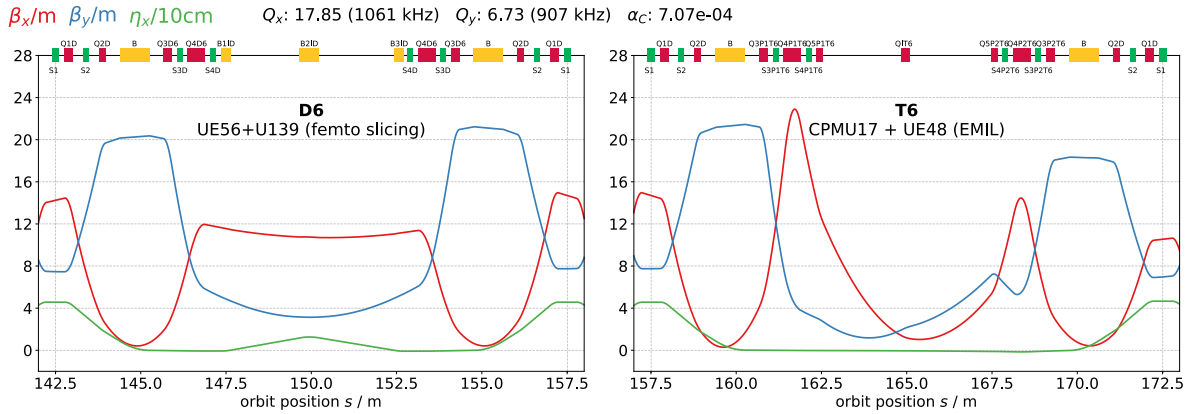

Figure 4.2: The Twiss parameters in D6 (Emil) and T6 (femto slicing) straights (extracted from [5])

Turning off the Q5T2 quadrupoles means turning the T2 triplet section into another doublet section. As the BESSY II storage ring has undergone several modifications since the design lattices from 1996, there are additional constraints that have to be taken into account:

Since 2005, a femto-slicing facility, producing ultra-short x-ray pulses, is commissioned in the D6 section of the BESSY II storage ring [44], [45]. A femtosecond laser pulse modulates the electron energy in a wiggler called the modulator. The off-momentum electrons are extracted by the transverse displacement of a bending magnet, and the synchrotron radiation is emitted in the following undulator called the radiator. This femto-slicing experiment required the installation of three additional bending magnets, which together form a dipole chicane, the wiggler U139, and the undulator UE56. Another significant alteration of the BESSY II storage ring was carried out as part of the EMIL project [46], which is dedicated to researching materials for

renewable energy. It provides multiple beamlines with simultaneous access to soft and hard x-rays. These are provided by the two UE-48 and CPMU-17 undulators, which are canted to separate the generated beam cones. In order to support this setup of the two canted undulators, the vertical focusing quadrupole QIT6 was installed in the center of the T6 straight, which shifts the vertical beam waist to the center of the CPMU-17 device. The Twiss parameters in D6 (Emil) and T6 (femto-slicing) straights are shown in Figure 4.2.

Another modification of the BESSY II storage ring was the introduction of the so-called injection optics [47]: The horizontal beta function β_x was increased in the injection-straight D1 and decreased in all other doublet straights to improve the injection efficiency.

4.1.2 Constraints for the Development of the Q5T2off-optics

The Twiss parameters of the BESSY II storage ring were carefully tuned to support the mentioned changes. New modifications should not conflict with the lattice development of the last several years. Therefore, when turning off the Q5T2 quadrupoles, it should be made sure that the changes are as local as possible with respect to the T2 section, and perturbations of the Twiss parameters outside of the T2 should be as small as possible.

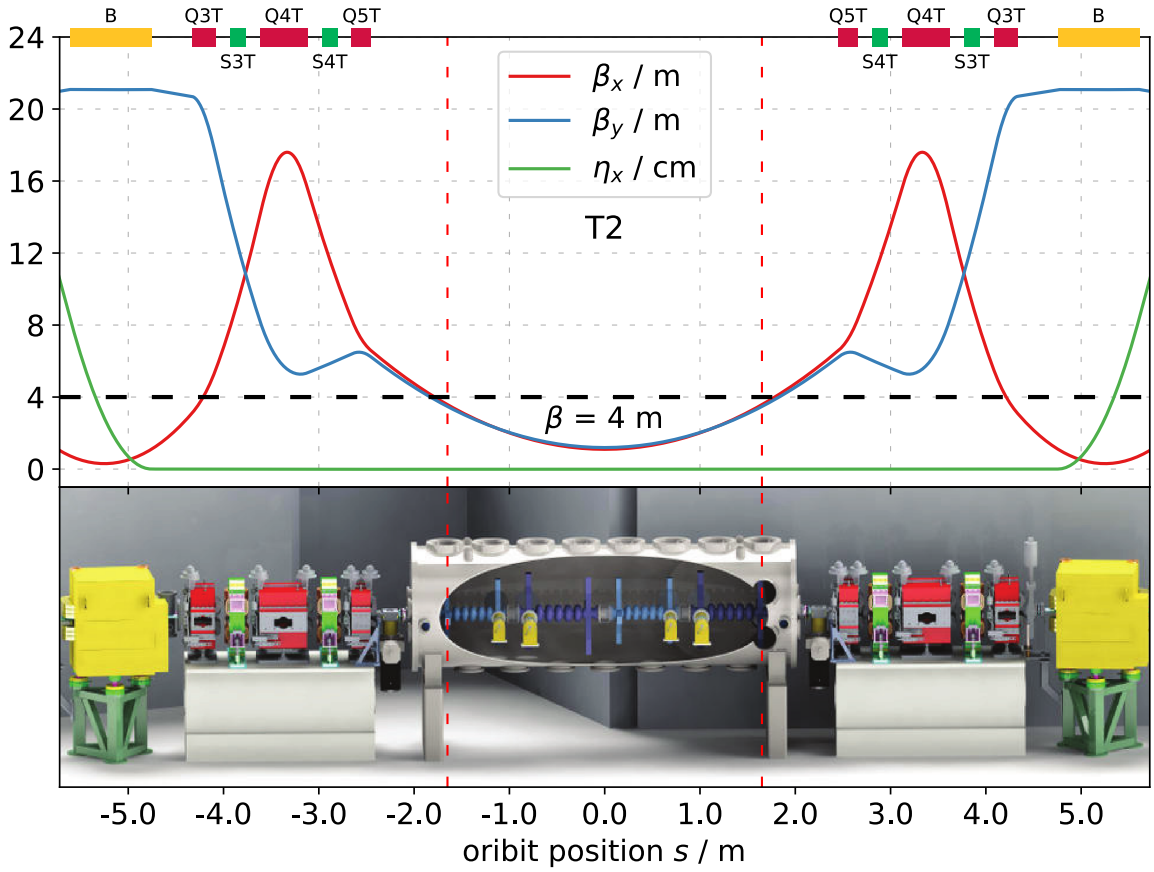

Figure 4.3: The horizontal and vertical beta functions $\beta_{x,y}$ of the current standard user lattice. The value of has to be kept below 4 m to avoid coupled bunch instabilities (based on [1] and extracted from [5])

The VSR cavities set another requirement for the Q5T2off-optics. As addressed by [1] beam-cavity interactions can cause coupled bunch instabilities, which spoil the beam quality. For a fixed threshold transverse cavity impedance

$$Z_{\mathrm{th}}^{\perp}(\tau_{\mathrm{d}}^{-1}) = \frac{\tau_{\mathrm{d}}^{-1}}{\beta} \frac{4\pi E/e}{\omega_{\mathrm{rev}} I_{\mathrm{DC}}} \tag{4.1}$$

the average beam current I_{DC} is determined by the energy E, the damping rate τ_{d}^{-1}, the angular beam revolution frequency ω_{rev} and the value of the beta function β within the cavity. As the requirements of the users of the synchrotron radiation restrain the energy E, the circumference of the ring defines the angular beam revolution frequency ω_{rev} and increasing the damping rate τ_{d}^{-1} might conflict with aspects of the machine operation, the value of the beta function within the cavity module effectively limits the average beam current I_{DC}. By the estima-

tions of [1, p. 83], a beta function of below 4 m within the cavity module would be sufficient to store the required current.

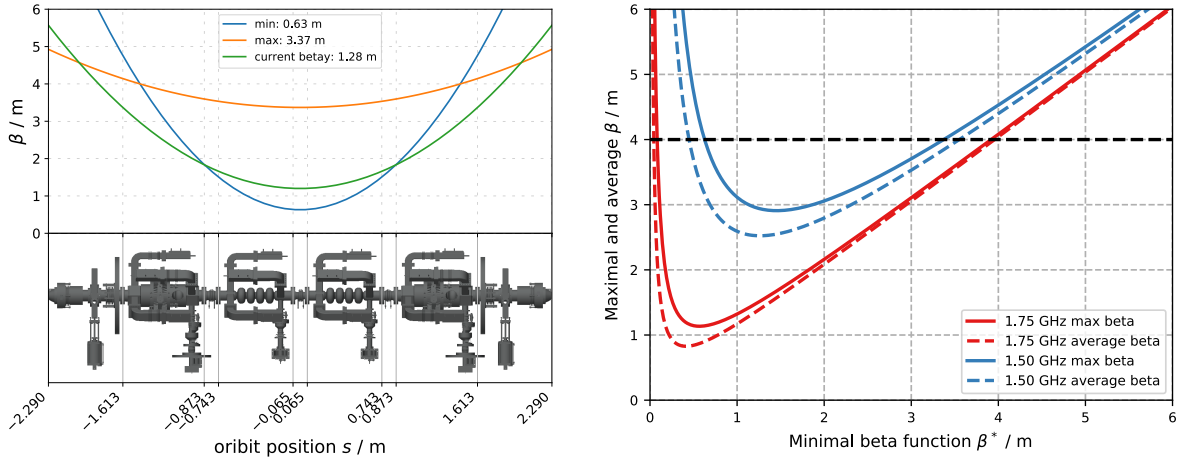

Figure 4.4: Maxium and average beta functions $\beta_{x,y}$ at the 1.50 GHz and 1.75 GHz cavites as a function of the minimum beta function β_{min} (based on [M. Ries, priv. comm., 2017] and extracted from [5])

Assuming a symmetry point, the beta function $\beta(s)$ at the distance from the center of T2 is fully defined by the minimum beta function β_{min} at symmetry point:

$$\beta(s) = \beta_{\text{min}} + \frac{s^2}{\beta_{\text{min}}} \tag{4.2}$$

Therefore it is possible to calculate the required minimum beta function β_{min} to keep the beta function belows 4 m within the cavity moduel. Figure 4.4 shows the maximum and average beta function at the 1.50 GHz and 1.75 GHz cavities as a function of the minimum beta function β_{min}. A minimum beta function β_{min} between 0.6 m and 3.4 m is sufficient to keep the average beta below 4 m. A minimum beta function β_{min} between 1 m and 2 m would be optimal to keep the average beta function within the cavity module as small as possible.

4.1.3 Turning off the Q5T2 Quadrupoles

As stated above, the goal is to minimize the beta functions' perturbation compared to the current standard user optics and keep the changes as local as possible. This can be accomplished by an optimizer that minimizes a fitness function chosen to satisfy the stated objective. However, the fact that not all quadruple settings result in a stable lattice poses a unique chal-

lenge: As an unstable quadrupole setting has no solution, all of them are equally bad. Therefore, choosing an objective function that provides a practical value to the optimizer in such a case is not easy. It seems reasonable to return a high constant or infinity. However, this leads to a problem when starting an optimizer in a region of unstable quadrupole settings. Usually, all other quadrupole settings in configuration space in the direct vicinity of an unstable lattice configuration are unstable as well. Therefore, the objective function always returns the same value regardless of which direction the optimizer chooses in the configuration space. Thus, the optimizer is practically left in the dark, and the possibility to converge is left to chance.

Just switching off the Q5T2 quadrupoles in the current standard user optics leads to an unstable lattice. That means that a quadrupole setting with only the Q5T2 quadrupoles turned off cannot be used as a starting configuration for optimization. So first, a stable lattice configuration has to be found, which can then be further improved by an optimizer.

A reasonable approach seems to start by only using quadrupoles in the T2 section to find a stable quadrupole setting. This was directly tested at the machine and also afterward checked by doing a quadrupole scan in simulations. The stability condition of the Twiss parameters

$$2 - R_{11}^2 - 2R_{12}R_{21} - R_{22}^2 > 0 \tag{4.3}$$

was derived in Section 2.3.3. The quadrupole scan in Figure 4.5 shows the stable regions for different settings of the Q3T2, Q4T2, and Q5T2 magnets.

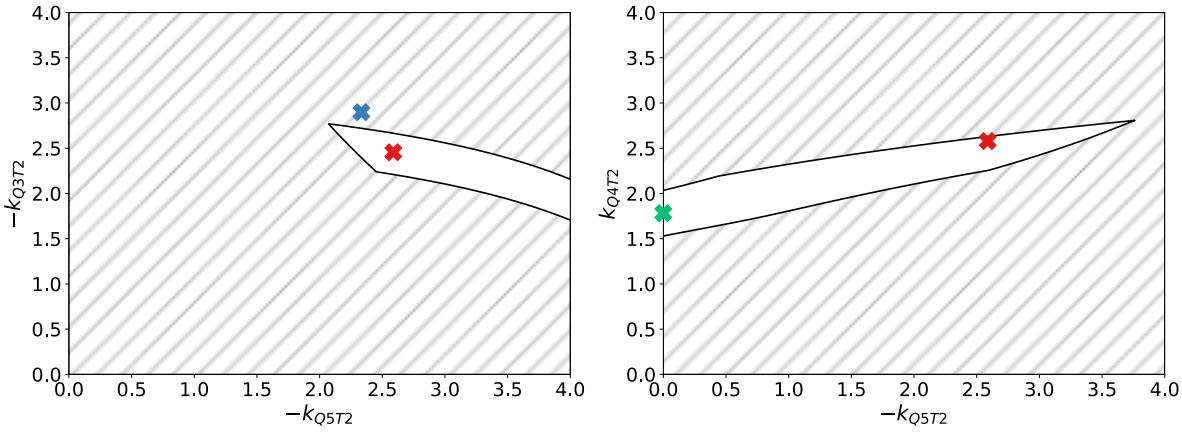

Figure 4.5: The stability of the BESSY II storage ring lattices for different configurations of the quadrupoles in the T2 section. Areas where no stable solution of the Twiss parameters exists, are shaded with diagonal lines. The left plot shows the different quadrupole configurations of the Q5T2 and the Q3T2 quadrupoles. The right plot shows the different quadrupole configurations of the Q5T2 and the Q4T2 quadrupoles. The current standard user optics is marked in red. The blue cross marks the attempt to compensate the turn-off solely with the Q3T2 quadrupoles, which eventually led to an unstable lattice. A stable solution, compensated by the decrease of the Q4T2s quadrupole strength, is shown in green. (extracted from [5])

At the machine, the first idea was to use the other vertical focusing Q3T2 quadrupoles to compensate for the turn-off of the Q5T2 quadrupoles. Increasing the Q3T2 quadrupoles made it possible to slightly more decrease the Q5T2 quadrupoles. However, at about 94 % of the initial strength of the Q5T2 quadrupoles, the beam was lost. That seems to be consistent with the results of the quadrupole scan. The blue cross in Figure 4.5 marks this unstable quadrupole setting.

The next idea was to compensate for the turn-off of the Q5T2 quadrupoles by reducing the strength of the vertical defocusing Q4T2 quadrupoles. That led to a working machine but resulted in a very low injection efficiency. Figure 4.6 shows the Twiss parameters where only the Q3T2 and Q4T2 quadrupoles were changed to compensate for the turn-off of the Q5T2 magnets. Compared to the standard user optics, the perturbation of the beta functions is significant all around the ring, which probably led to the low injection efficiency. In particular, the vertical beta function increases substantially in the T1, T3, and T6 sections.

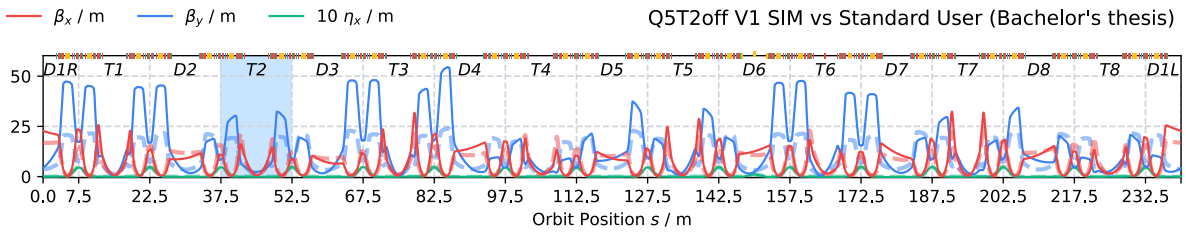

Figure 4.6: Beta functions with turned off Q5T2 quadrupoles compensated with the Q3T and Q4T quadrupoles of the T2 section (highlighted in blue). The Twiss parameters of the current standard user are shown in comparison with a dashed line.

The first and most local solution provides a starting point to improve the optics further using an optimizer. The average relative change of the beta function

$$F(\beta) = \frac{1}{L} \int_0^L \frac{\beta(s)}{\beta_{\mathrm{ref}}(s)} \mathrm{d}s \tag{4.4}$$

was chosen as the objective function. Different combinations of quadrupoles were used to compensate for the turn-off of the Q5T2 quadrupoles. Motivation for selecting the different quadrupole sets and a detailed overview of the solutions can be found in my bachelor thesis [5]. Figure 4.7 shows the best solution obtained within the scope of the bachelor thesis. The optics was tested at the storage ring, and a high current with reasonable lifetime and injection efficiency was stored.

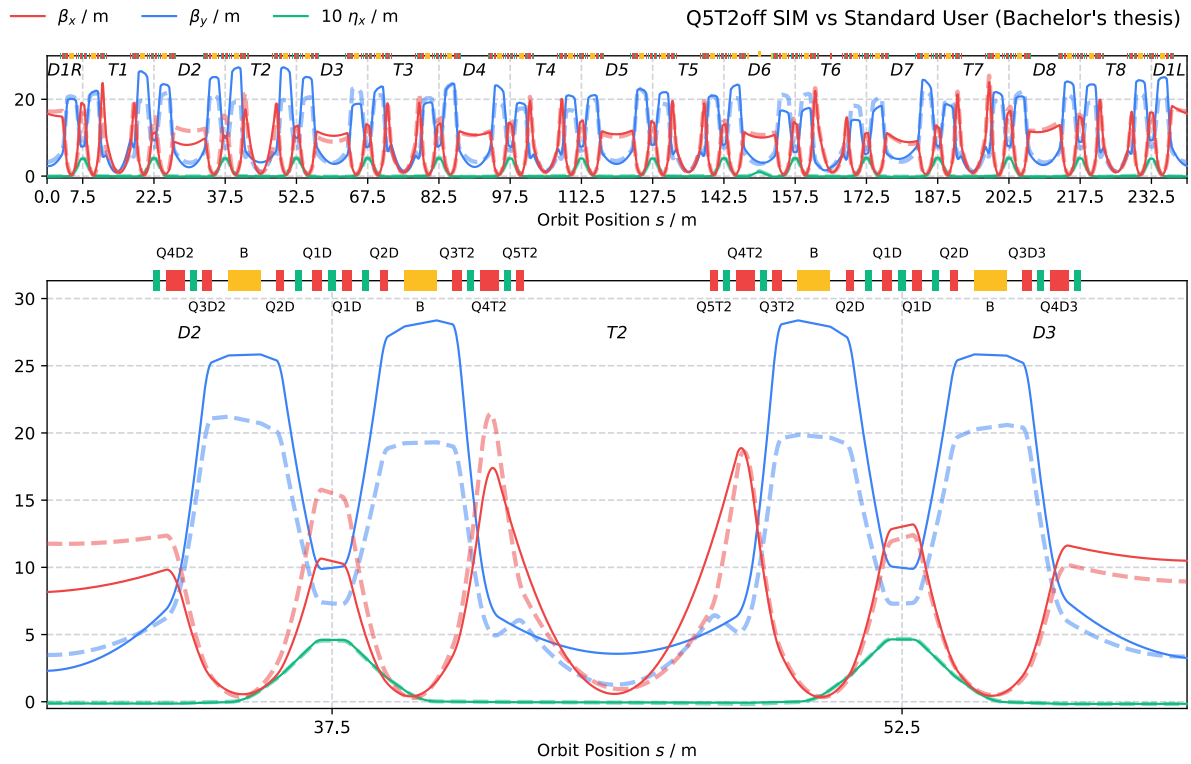

Figure 4.7: Twiss parameters of the best solution of my bachelor's thesis (solid) in comparison to the standard user optics (dashed).

4.1.4 Optimizing the Q5T2off-optics in Simulations

As stated in the conclusion of my bachelor's thesis, further improvements to the optics are necessary: First, the presented solution has a non-negligible β-beat outside of the T2 section. The standard user optics is fine-tuned in the injection straight, EMIL and, femto-slicing sections. Therefore, changes introduced by the Q5T2off optics should be as local as possible to avoid conflicting with previously made considerations. The goal is to limit the β-beat to the T2 and its adjacent neighboring sections. Another issue is that the vertical beta function in the center of the T2 section is with 3.6 m still too high. As stated before, the minimum beta function must be a least 3.4 m and would optimally be between one and two meters. Furthermore, the optics have to be further optimized regarding the non-linear dynamics. Different sextupole settings can be used to optimize the momentum acceptance.

Adjusting the objective function can reduce the β-beat. My bachelor's thesis used the mean relative residual of the beta function. That meant that a change from 4 m to 2 m corresponding to -50 % had the same weight as a change from 20 m to 30 m (+50 %). Therefore, this incen-

tivized the optimizer to reduce the beta function below the value of the standard user optics at certain positions, resulting in a larger β-beat. This can be solved by composing the relative change of the beta function $\frac{\beta(s)}{\beta_{\mathrm{ref}}(s)}$ with a rectified linear unit (ReLU) function $R(x) = \max(1, x)$, which maps every value below one to one. This has the effect of a lower threshold, which disincentivizes to lower the beta function below the reference value at the expense of making it larger somewhere else. Larger changes can be more disincentivized by squaring the relative change. To ensure the vertical beta function β_{y} is small enough at the center of the T2 section s_{T2}, it can be included in the objective function.

Taking the mentioned considerations into account, leads to the new objective function:

$$F(\beta) = \frac{1}{L}\int_0^L R\left(\frac{\beta(s)}{\beta_{\mathrm{ref}}(s)}\right)^2 \mathrm{d}s + \beta_{\mathrm{y}}(s_{\mathrm{T2}}) \text{ with } R(x) = \begin{cases} 1, & \text{for } x < 1 \\ x, & \text{else} \end{cases} \tag{4.5}$$

With the developed code, this roughly translates to:

```
def objective_function(values, quads):
    for quad, value in zip(quads, values):
        quad.k1 = value

    if not twiss.stable:
        return np.inf

    beta_beat_x = np.maximum(1, twiss.beta_x / twiss_ref.beta_x) ** 2
    beta_beat_y = np.maximum(1, twiss.beta_y / twiss_ref.beta_y) ** 2
    return np.mean([beta_beat_x, beta_beat_y]) + twiss.beta_y[t2_center]
```

The optimization procedure was run for different combinations of quadrupoles. Due to the improvements made to the Twiss calculation code, it was now possible to do much more iterations in a much shorter time. Furthermore, it was possible to run the optimization procedure multiple times in a row, using the last optimization result as a starting point for the next. Figure 4.8 shows the Twiss parameters of the best obtained Q5T2off optics compared to the current standard user optics. A complete code snippet that reproduces the Q5T2off optics and explanation is included in the Appendix B.

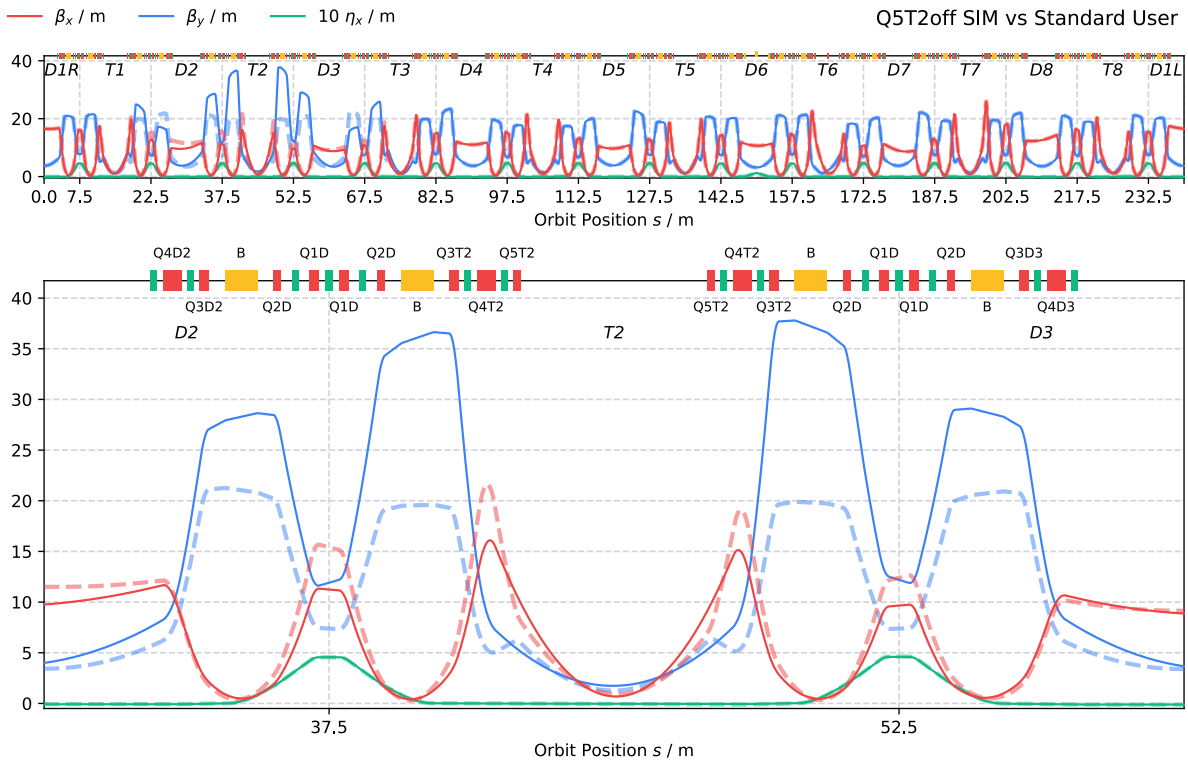

Figure 4.8: Twiss parameters of the best Q5T2off optics (solid) in comparison to the current standard user optics (dashed)

Figure 4.9 shows the β-beat of the new Q5T2off optics compared to the old Q5T2off optics. The old Q5T2off optics has a high horizontal β_x-beat in the injection straight and a high vertical β_y-beat all around the ring, including the femto-slicing section D6 and EMIL section T6. With the improvements to the Q5T2off optics, the β-beat outside of T2 and its adjacent sections is effectively negligible. For the horizontal plane, the changes within the T2 and its adjacent sections are primarily negative. For the vertical plane, there are two peaks at the beginning of the adjacent DBA. Furthermore, the vertical beta function at the center of the T2 section was reduced from 3.6 m to 1.8 m.

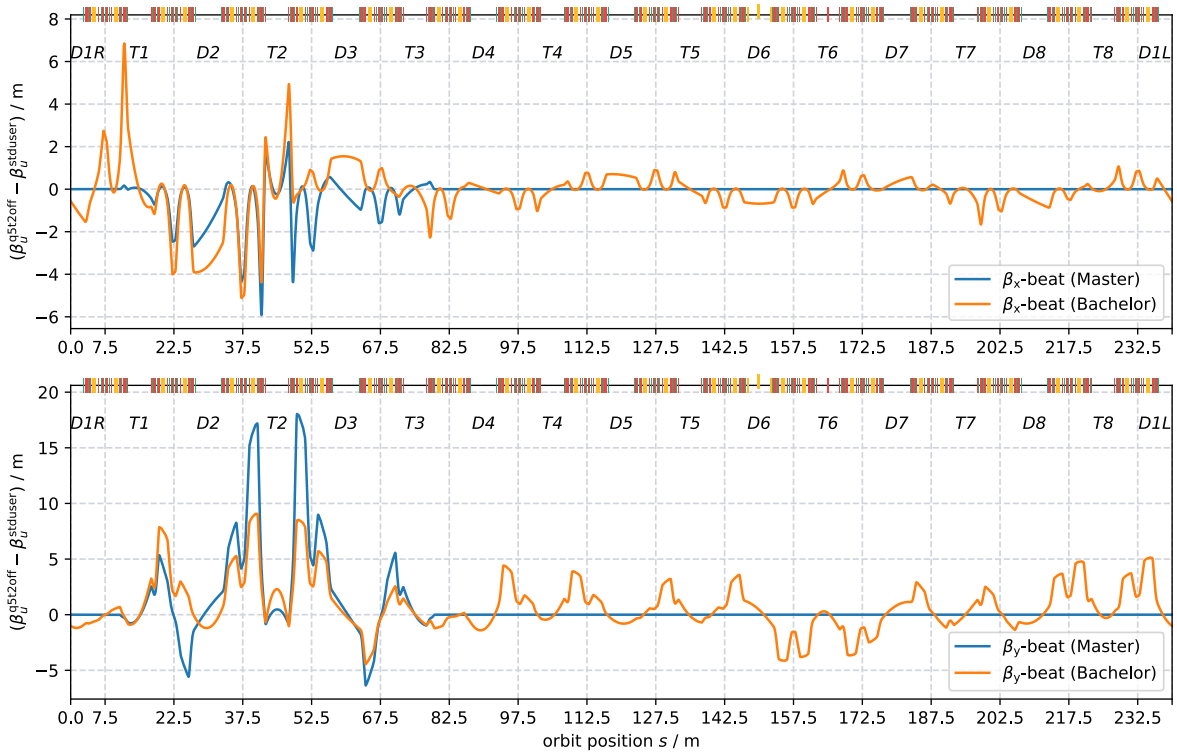

Figure 4.9: Beta beat of the Q5T2off optics with the standard user optics (blue) compared to the beta beat of the best solution of the bachelors thesis (orange)

4.1.5 User Operation Acceptance Test of the Q5T2 Optics

With improvements made to the Q5T2off optics, the following steps are to transfer the optics to the machine and test if it is ready for standard user operation. According to [18], the quadrupole strength is proportional to the current I

$$k \approx 2\frac{\mu_0 n I}{a^2}\frac{q}{p} \propto I, \tag{4.6}$$

where μ_0 is the vacuum permeability, a is the aperture radius and n corresponds to the winding number. Therefore, the new power supply values can be calculated by the ratio of the new and old quadrupole strength times the old power supply value:

$$I_{\text{new}} = \frac{k_{\text{new}}}{k_{\text{old}}} I_{\text{old}} \tag{4.7}$$

To make sure that the optics is transferred correctly to the machine, a precise measurement of the current quadrupole strengths k_{old} is necessary. Therefore, the Linear Optics From Closed Orbits (LOCO) method [48], [49] from the MatLab Middle Layer [50] can be used to determine the linear optics. The LOCO method first measures the orbit response matrix and the dispersion function. Afterward, the data is fitted to a lattice model to calculate the individual quadrupole strengths.

LOCO-measuring the current standard user optics started of the machine commissioning. Then a new Q5T2off optics was fitted based on the just LOCO-measured standard user optics. Next, the new power supply values were transferred to the machine using a program developed for this purpose, shown in Appendix C. After the machine was set to the new Q5T2off optics, another LOCO measurement was carried out to ensure the optics was transferred to the machine correctly. Figure 4.10 shows the Twiss parameters of the LOCO measured Q5T2off optics compared to the Twiss parameters of the Q5T2off optics obtained from the optimization. There seem to be some deviations in the horizontal beta function. Especially within some of the doublet sections, the beta function is slightly asymmetric to the center. However, the vertical beta function matches very closely with the one from the simulation. Overall, it can be said that the optics was transferred correctly. Finally, the quadrupole setting of the Q5T2off optics was saved in the control software to make it available for future machine commissions. Figure 4.11 shows the LOCO-measured Q5T2off optics compared to the current standard user optics.

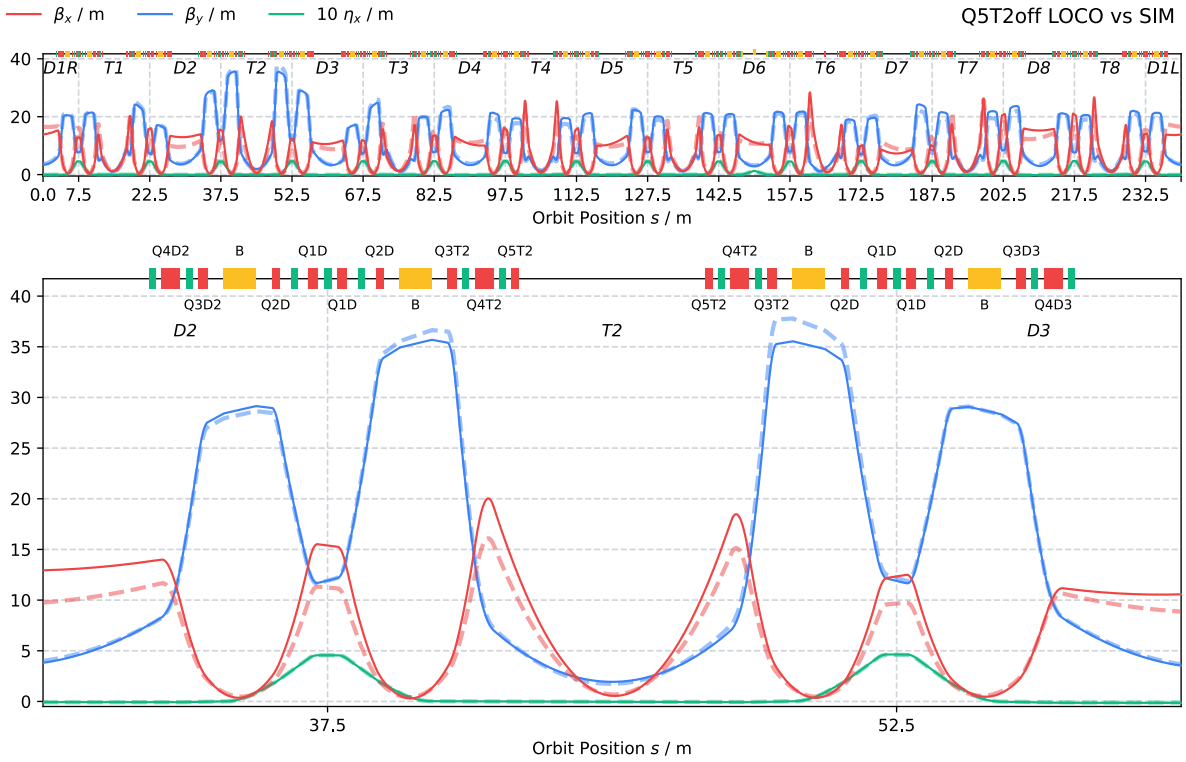

Figure 4.10: Twiss parameters of loco measured Q5T2off optics (solid) in comparison to the Q5T2off optics obtained from the optimization (dashed)

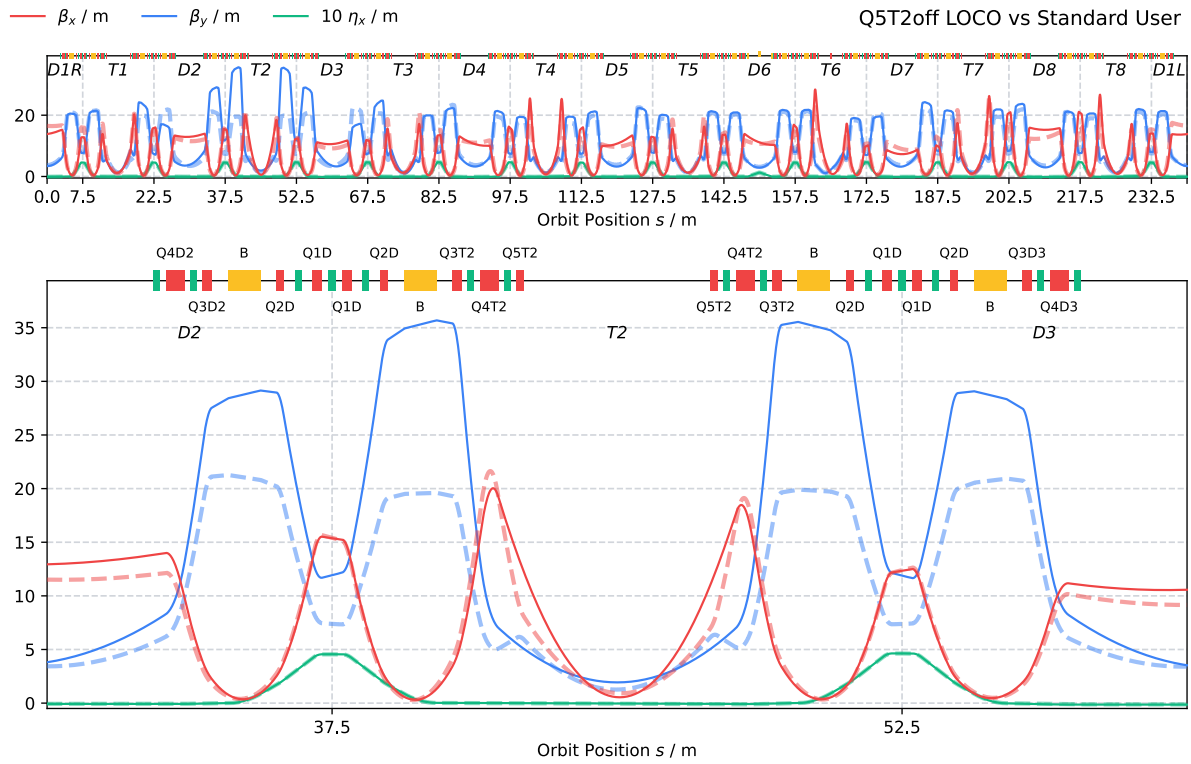

Figure 4.11: Twiss parameters of loco measured Q5T2off optics (solid) in comparison to the current standard user optics (dashed)

In another machine commissioning session, the Q5T2off optics was audited for standard user operation. During this user acceptance test, several smaller checks were performed. First, a small injection with low current was done to verify that the optics was correctly restored from the control software. Then, the quadrupoles were cycled to eliminate possible hysteresis effects. The orbit correction was used to improve the orbit. Next, a high current test was carried out. The current was injected up to 250 mA with injection efficiencies between 90% to 95%.

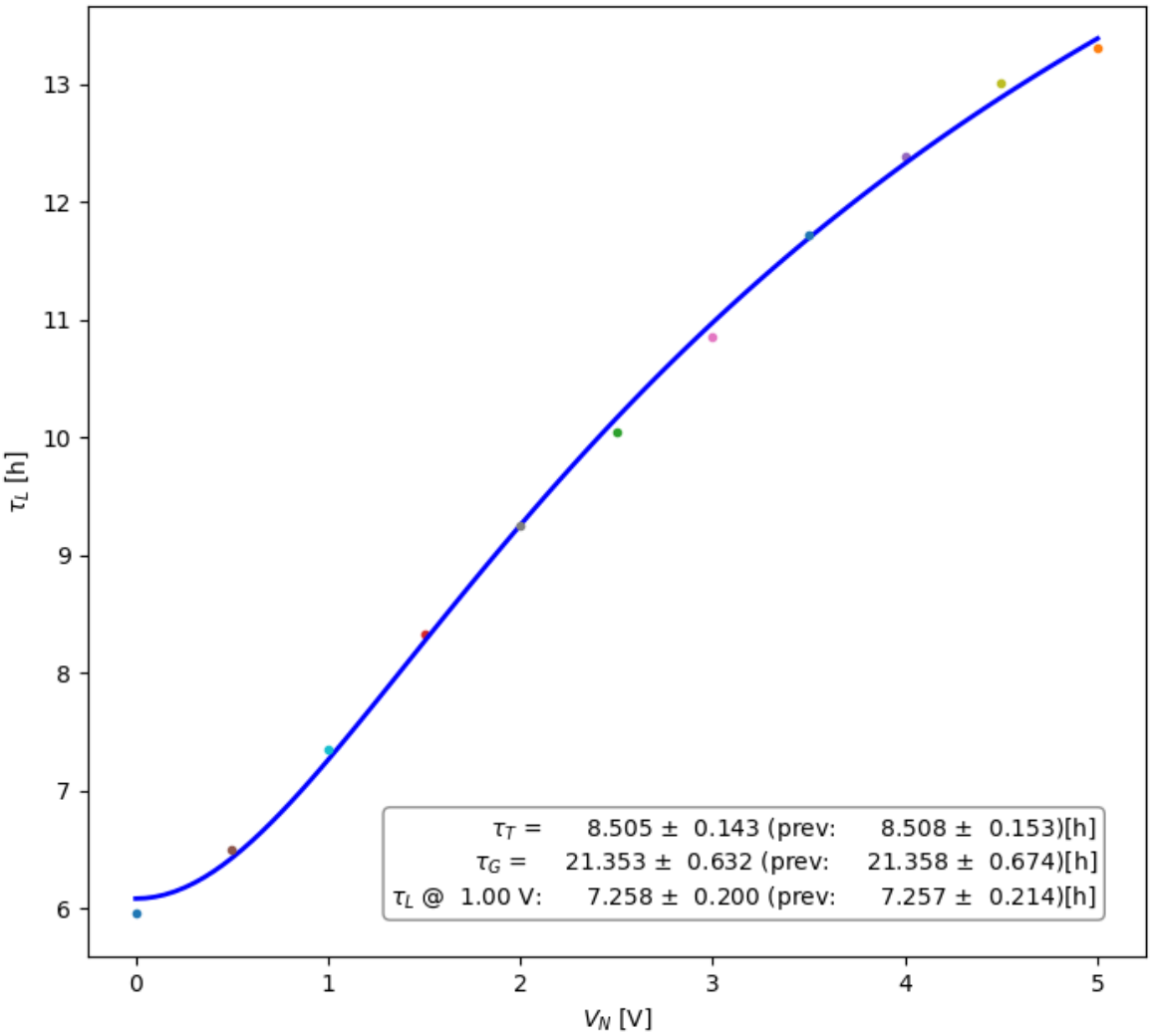

Figure 4.12: Lifetime as a function of the vertical noise V_{N}

Figure 4.12 shows the lifetime measurement. A vertical noise is used to inflate the vertical beam size, which reduces the probability of electron collisions within a bunch and therefore increases the lifetime. Touschek lifetime τ_{T} and gas lifetime τ_{G} are 8.5 and 21.4 hours, respectively. All in all, the measured lifetimes are close to the lifetimes of standard user optics.

Up next was the measurement of the kicker lifetimes. Here the kicker magnets of the orbit bump are fired rapidly without actually injecting a beam. If the lifetime is decreased during this process, electrons are lost at the septum, which would limit the possible injection efficiency. However, no effect on lifetime could be detected, which indicates an infinite kicker lifetime.

Table 4.2 list the results of the chromaticity measurement, containing the values of the tune, first and second-order chromaticity for the Q5T2off optics compared to the standard user optics. The horizontal, vertical, and longitudinal tunes are unchanged. However, the vertical

and longitudinal chromaticities of the Q5T2off optics are slightly smaller than in the standard user optics. Also, the 2nd order chromaticities are slightly smaller in the horizontal and vertical plane.

Table 4.2: Tune, first and second order chromaticity of the Q5T2off optics (left) and the standard user optics (right)

	Tune	**Chroma**	**Chroma 2nd**	**Tune**	**Chroma**	**Chroma 2nd**
x	0.8485	1.9245	-43.2045	0.8427	2.0929	-47.2121
y	0.7259	3.1846	-63.0949	0.7264	3.1098	-67.7113
z	0.0061	0.0034	0.2384	0.0061	0.0058	-0.7480

The streak camera was used to measure the length of the different bunches of the BESSY II fill pattern. Single bunch, PPRE bunch, slicing bunch, and multi-bunches had the same length as in the standard user optics. That was expected as the Q5T2off optics did not change the momentum compaction factor α_c.

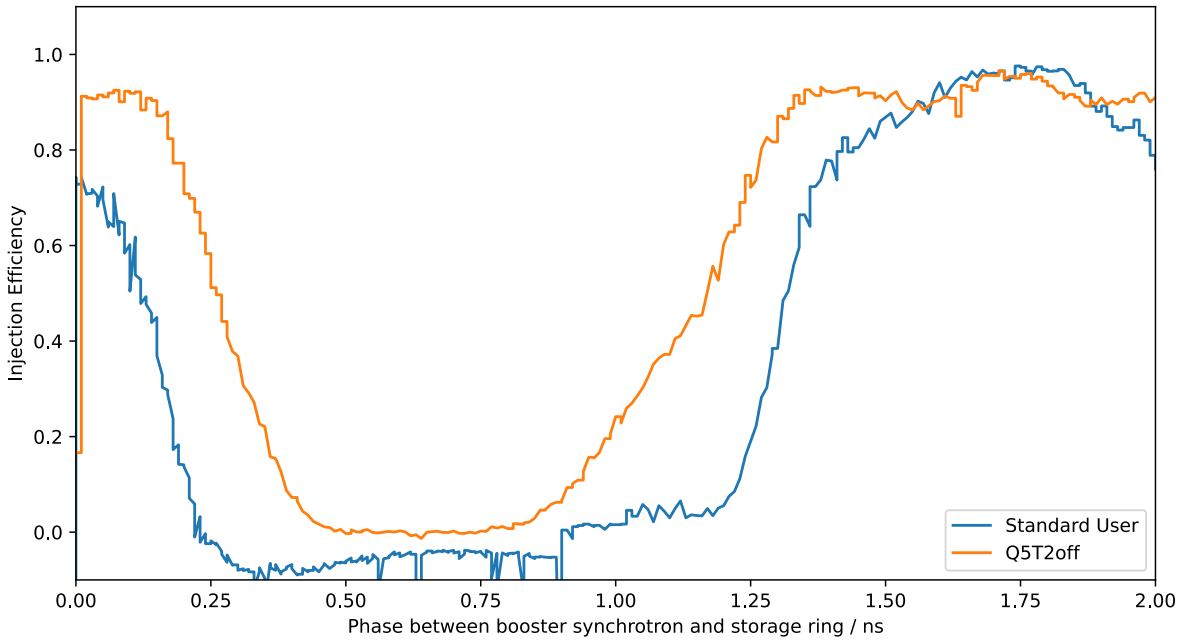

Figure 4.13: Phase acceptance scan off the Q5T2off optics compared to the standard user optics.

As demonstrated in [47], improving the phase acceptance can increase the momentum acceptance, injection efficiency, and Touschek lifetime. At BESSY II, the longitudinal phase between the booster synchrotron and the storage ring can be varied. This allows measuring the injection

efficiency as a function of the phase offset, also called a phase acceptance scan. The resulting curve's full width at half maximum (FWHM) can then define the phase acceptance. Figure 4.13 shows the phase acceptance of Q5T2off optics compared to the standard user optics. Surprisingly, even with non-optimized sextuples, the phase acceptance seems wider but slightly lower than in the standard user optics. The FWHM of the curves are 1.2 ns and 0.9 ns, respectively. The fact that the phase acceptance of the standard optics was so small leaves the suspicion that the sextupole setting was not fully optimized during machine commissioning. Also demonstrated in [47], a phase acceptance of 1.5 ns FWHM for standard user optics is possible with an optimized sextupole setting.

The optics was run overnight as a long-term test. One thing that remains is to optimize the harmonic sextupole setting. However, all in all, the user acceptance test can be considered a success. Important beam parameters like lifetime, injection efficiency, phase acceptance, chromaticity, and kicker lifetime are as good as in the standard user optics.

4.2 Emittance Exchange Experiment

Besides optimizing the Q5T2-off optics, the developed code was also used to modify the BESSY II storage ring optics for an emittance exchange experiment. The experiment required a high horizontal beta function of 15 m at the center of the T2 straight while keeping the perturbation of the beta function small around the rest of the storage ring. The goal was to study if raising the beta function at the position of a skew quadrupole, set up by four striplines, would increase its resonant excitation. In theory, the higher amplitude due to the higher beta function should lead to a stronger skew quadrupole kick, increasing the transverse emittance exchange.

The standard user optics was modified using the objective function

$$F(\beta) = \frac{1}{L}\int_0^L R\left(\frac{\beta(s)}{\beta_{\mathrm{ref}}(s)}\right)^2 \mathrm{d}s + (15\mathrm{m} - \beta_{\mathrm{x}}(s_{\mathrm{T2}}))^2 + |\frac{\mathrm{d}\beta_{\mathrm{x}}}{\mathrm{d}s}(s_{\mathrm{T2}})|, \tag{4.8}$$

where $R(x) = \max(1, x)$ is the ReLU function thresholding at one. Similar to optimizing the Q5T2off optics, the first term is used to reduce the β_{x}-beat around the rest of the ring. Moreover, the region between 40 m and 50 m is excluded from the integral as this would be in conflict with the second term. The ReLU function R(x) is composed with the relative residual of the beta function $\frac{\beta(s)}{\beta_{\mathrm{ref}}(s)}$ to disincentivize lowering the beta function below the reference value at the expense of making it larger somewhere else. The second term is used to raise the beta

function to 15 m. The first derivative of the beta function $\frac{\mathrm{d}\beta_x}{\mathrm{d}s}(s_{T2})$ ensures the resulting optics is symmetrical at the center of T2 section.

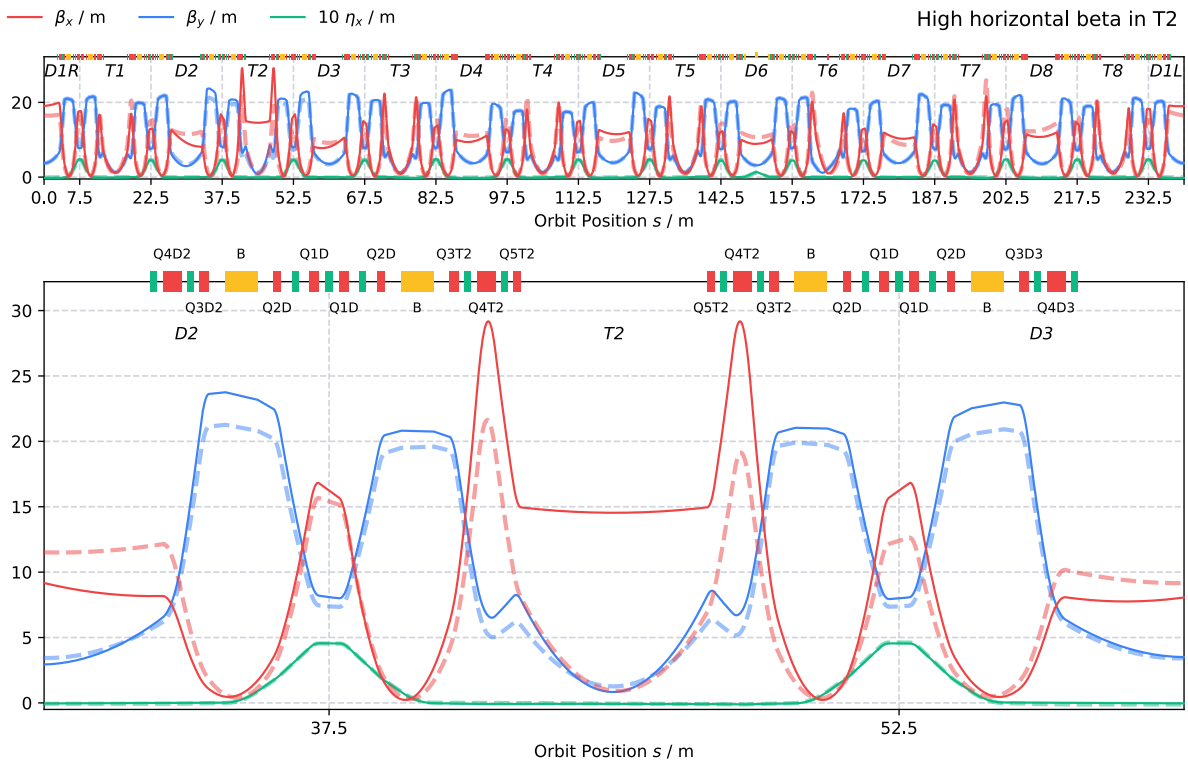

Figure 4.14: Optics 1 - High horizontal and low vertical beta and function in the T2 straight.

Figure 4.14 shows an optics where all quadrupoles from the D2 to the D3 section were used for the optimization. Already described in Section 4.1.5, the optics was transferred to the machine by calculating the new power supply values from the old and new quadrupole strength ratio, using the tool shown in Appendix C. Afterward, the optics was LOCO-measured to ensure it was transferred correctly.

Table 4.3: Beam size depending on resonant skew excitations for the different optics

Optics	**Skew Excitation**	σ_x **/ µm**	$\sigma_\mathrm{x}/\sigma_\mathrm{x,off}$	σ_y **/ µm**	$\sigma_\mathrm{y}/\sigma_\mathrm{y,off}$
Standard User	off	62.4		37.0	
Standard User	on	57.0	0.91	169.0	4.57
—	—	—	—	—	—
Optics 1	off	58.0		41.3	
Optics 1	on	52.8	0.91	227.0	5.54
—	—	—	—	—	—
Optics 2	off	58.8		44.3	
Optics 2	on	54.2	0.92	267.0	6.03

The stripline in the center of the T2 section was set up as a skew quadrupole to excite the beam resonantly. A pinhole at the dipole at the end of the D5 section was used to measure the beam size. Table 4.3 list the beam sizes for the standard user optics and the optics with the high horizontal beta function with the skew excitation turned on and off. Without any excitation, the standard user optics' horizontal and vertical beam sizes at the pinhole are 62.4 µm and 37.0 µm, respectively. Despite the small coupling of below 2%, the vertical beam size is σ_y comparable to the horizontal beam size σ_x, since the vertical beta function β_y is many times larger than the horizontal beta function β_x at the pinhole position. With the resonant skew excitation turned on, the horizontal beam size decreases to 57.0 µm while the vertical beam size increases to 169.0 µm. As the beam size is defined by $\sigma_\mathrm{u} = \sqrt{\epsilon_\mathrm{u}\beta_\mathrm{u}}$, this change corresponds to a decrease of the horizontal emittance to $0.91^2 = 0.83$.

Without the excitation, the optics with the high horizontal beta function has a slightly smaller horizontal beam size σ_x of 58.0 µm and a slightly larger vertical beam size σ_y of 41.3 µm. While the Twiss parameters seem to have been unaffected in the D5 section in simulation, this change can probably still be attributed to a change of the beta functions when the optics was transfered to the machine. Surprisingly, with the skew excitation turned on, the horizontal emittance only decreased to 52.8 µm, corresponding to 91 %, which is the same value as in the standard user optics. Thus, in contradiction to the expected results, the additional horizontal amplitude at the skew quadrupole did not seem influence the emittance exchange.

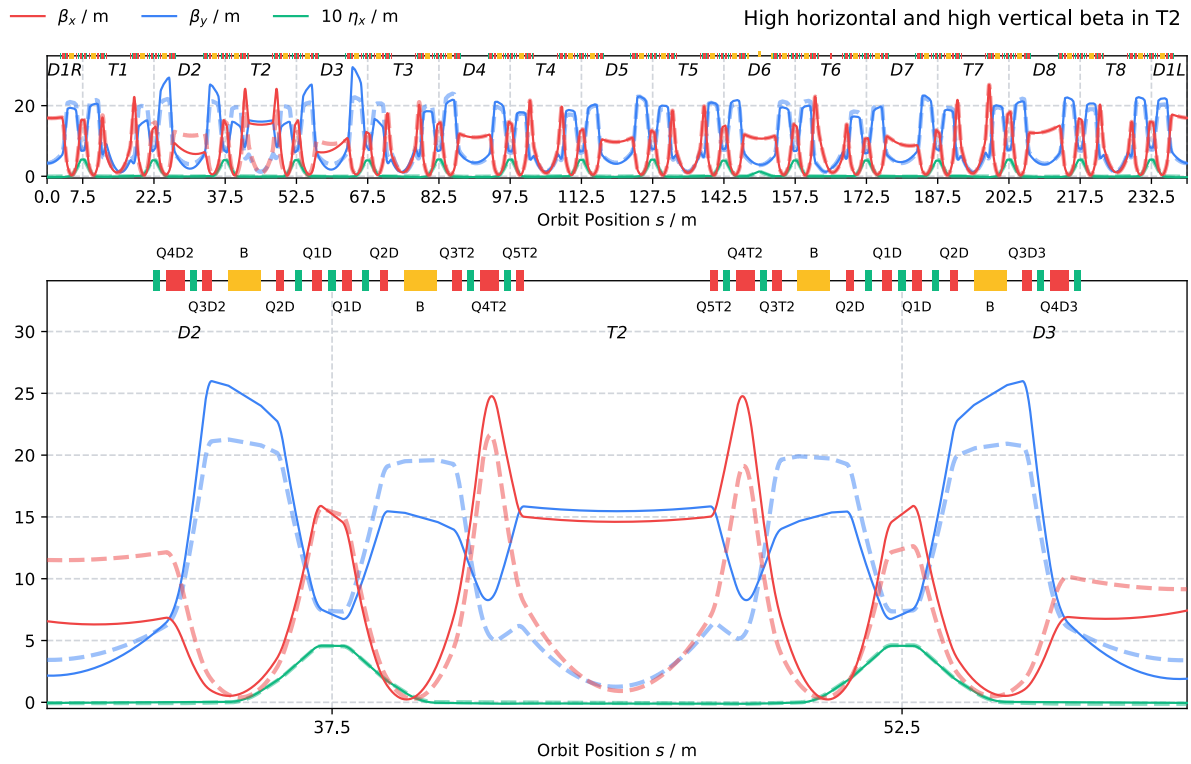

Figure 4.15: Optics 2 - High horizontal and high vertical function in the T2 straight.

Therefore, during the same machine commissioning session, it was decided to fit a second optics where also the vertical beta function is raised to 15 m. For this, the objective function was extended by the vertical analog of the second and third terms of Equation 4.8. Figure 4.15 shows the Twiss parameters of the optics with both beta functions raised to 15 m in the center of the T2 section. The optics was again LOCO-measured to ensure it was transferred to the machine correctly. Again, the horizontal and vertical beam sizes were measured with the skew excitation on and off. Also, this time, the horizontal beam size was decreased to 0.92 of its size without excitation.

While it requires more investigation in simulations as well as in experiments why the larger amplitude at the skew quadrupole did not increase the emittance exchange, the experiment showed that the developed code is flexible enough to support various lattice development tasks.

4.3 Automated Lattice Summaries

Section 1.2.3 outlined the need for a shared database of lattice files and automatically generated summaries: Physicists often repeat the cumbersome work of manually translating between different lattice file formats. In addition, simulations results are often difficult to reproduce or transfer to similar lattice development tasks. Finally, a standardized visualization of the simulation results would make lattices easier to compare.

An infrastructure that makes not only the lattices files sharable but also the simulation routines and visualization of the results could solve these problems. Ideally, a user only has to add a lattices file to the database. Then, a set of routines leveraging different simulation codes would be run locally or in the cloud. Finally, the simulation results would be summarized by a standardized lattice report. Due to the lattice summarizes being contentwise and visually consistent, the user could conveniently compare different lattices.

Such a framework could be valuable in different scenarios: It could facilitate collaborative lattice development within a facility. For example, in the case of HZB, it could be used to benchmark BESSY III lattice candidates. Furthermore, it could also facilitate the exchange of lattice files between different facilities. One tangible idea would be to create an updated version of the Synchrotron Light Source Data Book [7]. That could be realized by a website where every facility could contribute to. Even in cases of smaller lattice development tasks, which do not involve several people, a standardized interface to set up routines and organize lattice files is still beneficial. Using the same framework for local lattice development incentives collaboration as no extra work is required to share custom simulation routines. Reproducing the simulation result and plots of a colleague would then come down to running one command with the name of the routine and lattice file as arguments:

```
lattice-summaries <name-of-routine> <name-of-lattice>
```

This section presents the prove-of-concept framework *lattice-summaries*, developed based on the lattice file format LatticeJSON. Currently, it provides routines to calculate the Twiss parameters using elegant [9], MAD-X [8], or the simulation code developed for this thesis. Section 4.3.1 outlines the architecture of the *lattice-summaries* framework. The following subsections provide a more detailed overview of the different components of the framework.

4.3.1 Architecture

The source code of the *lattice-summaries* framework can be found on GitHub under the *nobeam* organization (https://github.com/nobeam).

The framework is split up into three different repositories:

1. lattice-summaries-data: A database of lattices files and optional run parameters
2. lattice-summaries: A set of routines leveraging different simulation codes
3. lattice-summaries-website: A website to inspect the simulation results

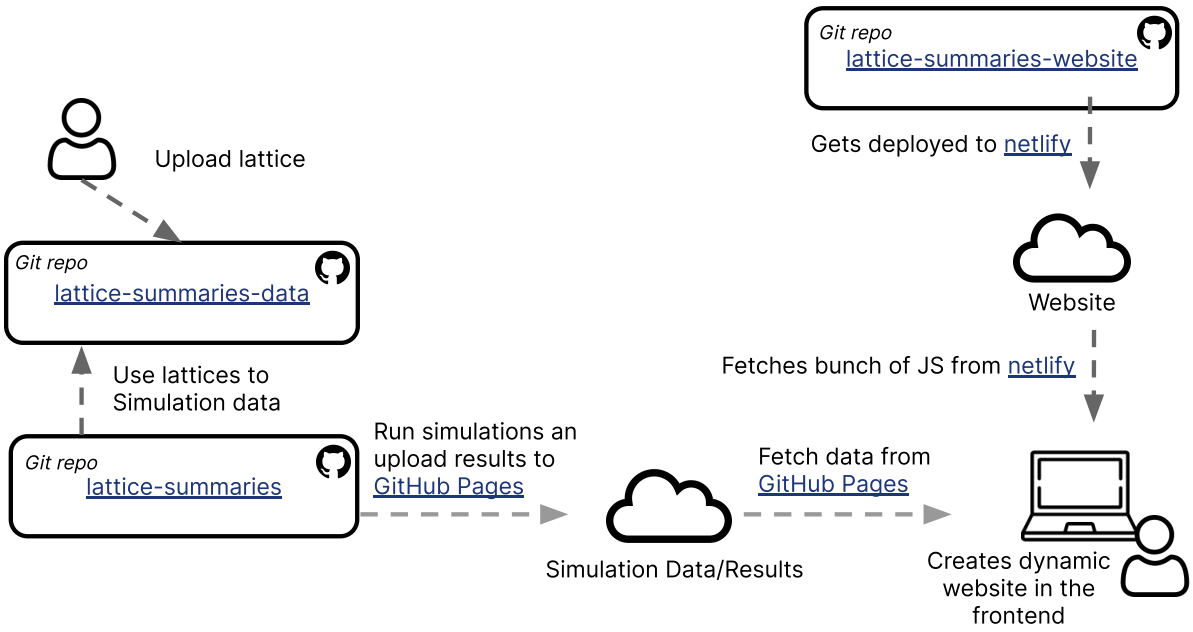

Figure 4.16: Schematic overview of the lattice summaries architecture

Figure 4.16 gives a schematic overview of the lattice summaries architecture. A user uploads a new lattice file and optional run files to the *lattice-summaries-data* repository, which functions as the database. Then routines defined in the *lattice-summaries* repository are run for all selected lattices. Next, the simulation results are uploaded and served by a static web server. Now the simulation results can be viewed through a website, which dynamically creates the summary views for the given simulation data. The *lattice-summaries-website* repository contains the source code of the website.

The presented architecture was chosen to offer maximum flexibility: Keeping the data repository and repository containing the simulation routines separate makes it possible to support multiple instances. This way, the data repository can be switched out so that different facilities can create their own instances for private lattice development. A public instance, for example, could be used to create an updated version of the Synchrotron Light Source Data Book. Furthermore, this allows the framework to be used for local lattice development, as a local instance of the data repository can be used.

The repository containing the website and the simulations results are also kept separate. That has the advantage that the website does not have to be updated when new simulation results are computed. That is convenient in the case of local lattice development, where the simulation results change frequently.

4.3.2 Database of Lattice Files

As most lattice file formats are plain text files, using Git seemed reasonable to build the database. The alternative would have been to use some NoSQL database and built a small API on top of it so that people could contribute and review new lattices. However, this seemed like unnecessary work as most of the functionality is already provided by the Git ecosystem. For example, hosting systems for Git repositories like GitHub or GitLab already provide the functionality to contribute, review contributions and run custom code in the event of a new contribution. Furthermore, Git has the advantage that it is ubiquitous, and many physicists are already familiar with how to use it.

The lattices are split into different namespaces so that multiple physicists can work on similar lattices without creating a name collision. In general, the lattice files follow the naming schema:

```
<namespace> / <machine>_<familiy>_v_<version>
```

An exemplary folder structure of a lattice database looks like:

```
database
├── namespace-1
|   ├── bessy2_design-1996_v_1.json
|   ├── bessy2_stduser-2019-05-07-v_1.json
|   └── info.toml
├── namespace-2
|   ├── bessy3_5ba-20p_v_long-bend-tgrb.lte
|   ├── bessy3_5ba-20p_v_reference.lte
|   └── info.toml
...
```

Every namespace contains an `info.toml` file, which contains additional metadata. That includes the human-readable title, an optional description, a list of authors, a list of labels, and a list of simulation routines that should be run for the given lattice. The list of simulation routines is necessary because a lattice file might contain special elements unique to a given simulation, not available for the other simulation routines. Possible labels, for example, could be if the lattice contains anti-bends, combined functions magnets, or longitudinal gradient bends. A detailed explanation of adding a lattice to the database can be found in the `README.md` file of the lattice-summaries-data repository.

4.3.3 Set of Routines

The set of routines are responsible for generating the simulation results. This is arguably the most complex part. These routines can be dependent on each other. For example, to run the routine that plots the Twiss parameters, a routine that calculates them must be run first. If the lattice is not available in the necessary format, another routine has to be run before translating the lattice file to the needed format.

The goal is that different simulation codes can power these routines, with more routines being added over time. Currently, the *lattice-summaries* framework provides the following routines: One routine that translates lattice files between the LatticeJSON format and the lattice file formats used by elegant and MAD-X. One routine extracts general information about the lattice, such as the circumference, energy, number of sections, section length, or bends per section. Three routines that compute the Twiss parameters using either elegant, MAD-X, or the

developed code for this thesis. Furthermore, three routines that plot the Twiss parameters, a floor plan, or the higher-order chromaticity.

For local lattice development, the users change a config file to use their private version of the lattice database if they need a lattice not available in the shared database. Another convenience feature is that the dependency relation between the routines is lazy evaluated, which means that simulation is not rerun if a user only adjusts code, which is part of a visualization routine. Detailed instructions on the setup and usage are provided in `README.md` file of the lattice-summaries repository.

4.3.4 Lattice Summaries Website

This website provides a frontend to view the simulation results. The website is developed as a Single Page Application (SPA) using the JavaScript Framework Vue.js 3 [51] and the build tool Vite [52]. A SPA is a website that, on user interaction, dynamically rewrites the current web page instead of fetching a pre-rendered page from a server. The advantage is that the website does not have to be updated if new simulation results are uploaded. Instead, new pages are created dynamically in the frontend. Furthermore, it makes it possible to use the website for local development. By changing the `DATA_URL` environment variable, the user can display local simulation results.

Currently, the website consists out of three views:

1. A landing page that lists all lattices
2. A lattice view, which lists an overview of the available summaries for that lattice
3. Different types of summary views depending on the simulation routine

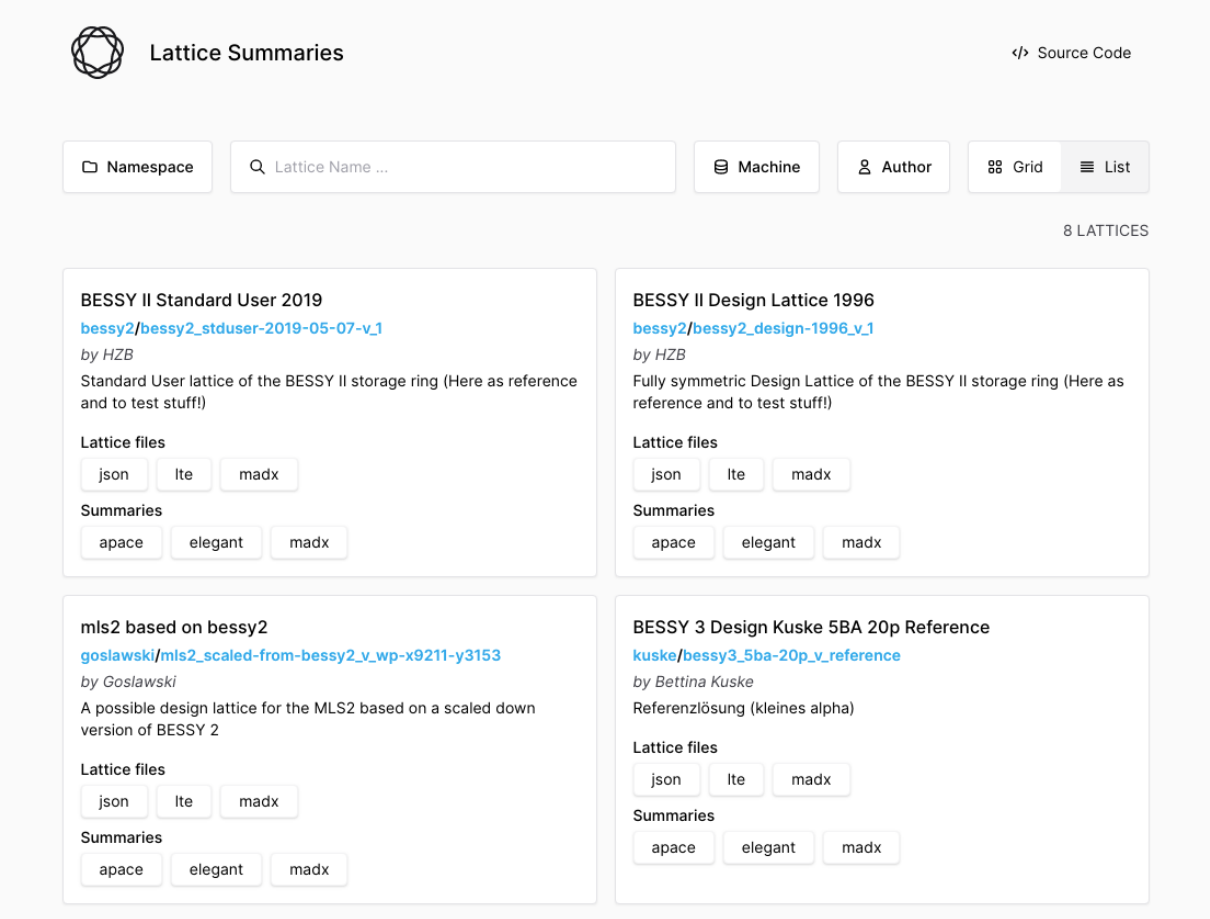

Figure 4.17: Screenshot of the landing page of the lattice summaries website

Figure 4.17 shows a screenshot of the landing page. It provides a search bar where lattices can be filtered by name. Furthermore, lattices can be filtered by *Namespace*, *Machine*, and *Author*. The search results are displayed as cards. These cards provide a short description, links to the corresponding lattice file in different formats, and links to the different types of summaries available for the given lattice.

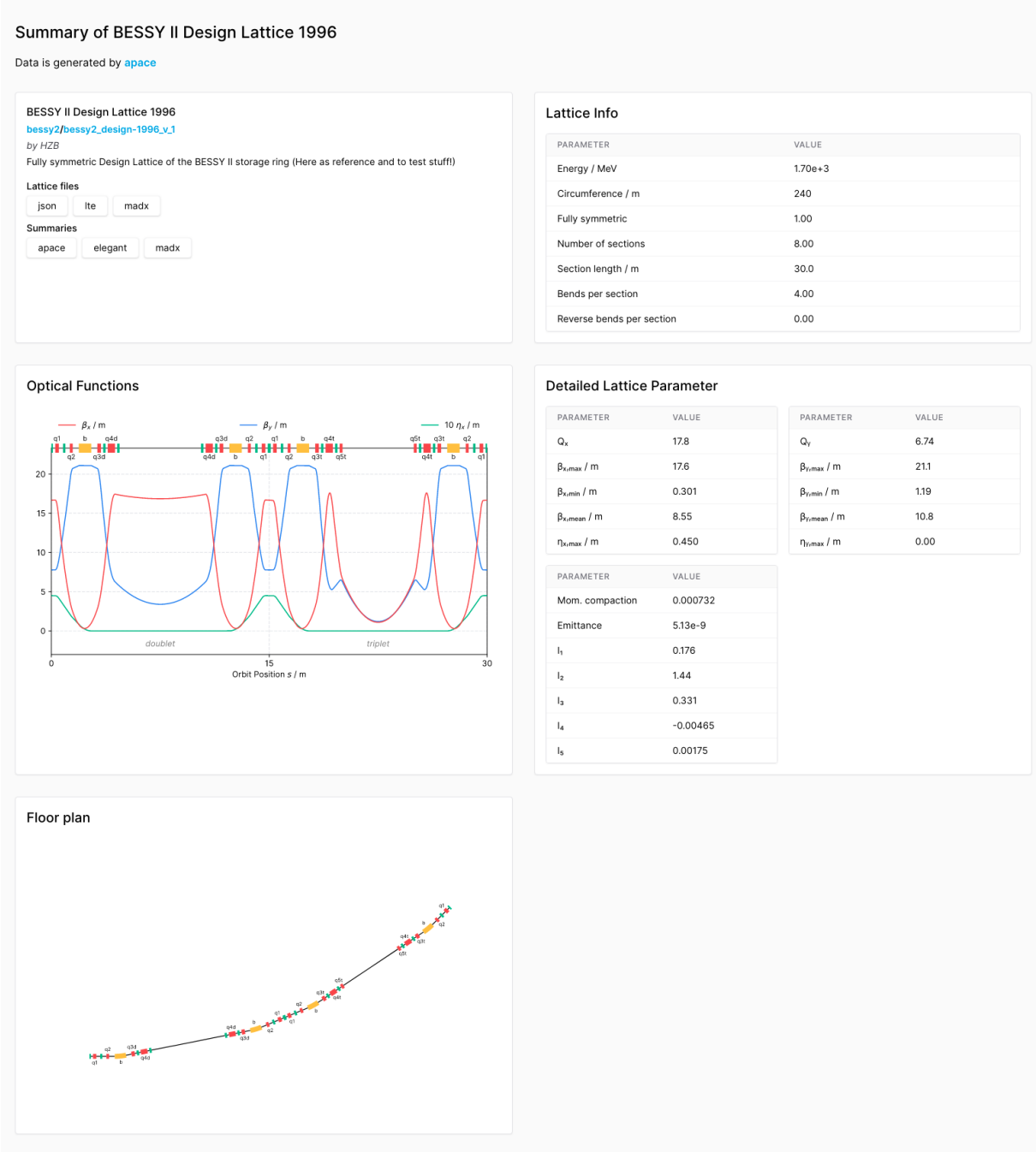

Figure 4.18: Screenshot of an examplary lattice summary of the BESSY II design lattice

Figure 4.18 shows an exemplary lattice summary of the BESSY II design lattice generated by the apace simulation code. The lattice summary consists out of different cards which provide different information. Independent of the type of summary, the first card is an info-card identical to the card shown in the search results. All other cards are specific to the type of summary. For example, in the case of the Twiss summary generated by the apace code, the second card provides some general information on the lattice. That includes energy, circumference, number of

bends, number of sections or, section length. The third card provides a plot with the beta function and horizontal dispersion function for the unit cell. Other lattice parameters like the momentum compaction factor, emittance, or the synchrotron radiation integrals are listed in a table on the fourth card. The last card shows a floor plan of the unit cells.

Chapter 5

Conclusion

This thesis presents the development of a new lattice design tool developed in the Python language and a new JSON-based lattice file format. The introduction outlines the technical obstacles physicists have to overcome during lattice development. Ranging from the problems of exchanging lattice files to the challenges of interfacing existing accelerator codes from a modern programming language, forcing physicists to write complicated and error-prone wrapper scripts.

Since Python has evolved to the lingua franca of the scientific community, it is an obvious candidate to form the basis of a new lattice design tool. In addition, the native interface to Python's ecosystem gives particle accelerator physics access to a large pool of scientific libraries, easy to integrate into lattice development workflows. While the capabilities at the moment are limited to linear beam dynamics, which corresponds only to a fraction of what mature accelerator physics codes like elegant or MAD-X are capable of, the developed code has been successfully used for several applications:

The Q5T2off optics presented in my bachelor's thesis was significantly improved. The β-beat outside the T2 section and its adjacent sections is negligible. The reduced vertical beta function in the center of the T2 section now clearly fulfills the requirements to avoid coupled bunch instabilities. Finally, a successful user acceptance test showed that the optics is ready for standard user operation.

In the context of an emittance exchange experiment, the BESSY II storage ring's optics was modified. First, the horizontal beta function was raised in the center of a triplet section to increase the influence of resonant skew excitation. Then, during the experiment, a second optics change was needed. The developed code was used throughout the same machine commissioning session to fit another optics where both beta functions were raised to 15 m in the center of the triplet section. While the experiment itself did not lead to the expected effect, it showed that the developed code is flexible enough to support a variety of lattice development tasks.

The developed code *apace* and the *LatticeJSON* lattice file format form the basis of a proof-of-concept framework to generate standardized lattice reports for a given set of lattice files.

While much work still has to be done, automated routines to calculate the Twiss parameters using elegant [9], MAD-X [8], or code developed for this thesis have been set up. The simulation results can be inspected via a website. The framework could be used to facilitate the lattice development of the BESSY II successor BESSY III, where many lattice candidates arise. Also, the framework could be used to create an updated version of the Synchrotron Light Source Data Book.

The new JSON-based lattice file format LatticeJSON can be considered a byproduct of this work. The goal was to create the most simple and straightforward lattice file format possible to make it easy to load the lattice data into every programming language. In contrary to other lattice files formats, the LatticeJSON format is a pure data format and does not support any constructs like variables, loops, or conditions. While, at the moment, only basic objects like drift sections, multipoles, cavities, or sub-lattices structures are supported, this is already sufficient to describe many accelerators. The simplicity of the format might not make it useable for every scenario, like for control system software, where additional configuration or the history of the magnet values is stored. However, its simplicity facilitates it to integrate it into other tasks. For example, the LatticeJSON format has proven helpful in transferring a new optics to the machine. To automate the process of calculating the new power supply values, the quadrupole strengths have to be extracted from the lattices files. Because the new lattice file format is JSON-based, which makes it trivial to extract values out of it, it was much easier than with the existing lattice file formats.

In conclusion, the code developed as part of this thesis will not replace the existing more mature and full-featured particle accelerator codes. However, it has shown how a modern lattice development workflow can look like and extends the ecosystem of accelerator tools by enabling fundamental lattice development using the Python language.

Appendix A

Developed Code

---

apace is yet **a**nother **p**article **a**ccelerator **c**od**e** designed for the optimization of beam optics. It is available as Python package and aims to provide a convenient and straightforward API to make use of Python's numerous scientific libraries.

The source code is available at:

https://github.com/andreasfelix/apace

A.1 Installation

Install and update using pip:

```
pip install -U apace
```

A.2 Requirements

- Python 3.6 or higher (CPython or PyPy)
- CFFI 1.0.0 or higher
- NumPy/SciPy
- Matplotlib
- C Compiler

A.3 Links

- Documentation: https://apace.readthedocs.io
- API Reference: https://apace.readthedocs.io/en/stable/reference/apace/index.html
- Examples: https://apace.readthedocs.io/en/stable/examples/index.html
- Releases: https://pypi.org/project/apace/
- Code: https://github.com/andreasfelix/apace
- Issue tracker: https://github.com/andreasfelix/apace/issues

A.4 License

Appendix B

Code to Reproduce the Q5T2off Optics

The following code snippets reproduce the Q5T2off optics presented in Section 4.1.4.

```
import apace as ap
import numpy as np
from scipy import optimize

bessy2 = ap.Lattice.from_file("bessy2_stduser_2019_05_07.json")
twiss = ap.Twiss(bessy2, steps_per_meter=4)
t2_center = np.searchsorted(twiss.s, 45)
beta_ref_x = twiss.beta_x.copy()
beta_ref_y = twiss.beta_y.copy()
```

First, the `apace`, `numpy` and, `scipy.optimize` libraries are imported. Then, a new `Lattice` and `Twiss` object for the current BESSY II standard user optics is created. The `np.searchsorted` is used to obtain the index of the center of the T2 section, located at 45 meters. Finally, the current horizontal and vertical beta functions are copied to be later used as reference values:

```
t1 = ["Q3P1T1", "Q3P2T1", "Q4P1T1", "Q4P2T1", "Q5P1T1", "Q5P2T1"]
d2 = ["Q3D2", "Q4D2"]
t2 = ["Q3T2", "Q4T2"]
d3 = ["Q3D3", "Q4D3"]
t3 = ["Q3T3", "Q4T3", "Q5T3"]
q5t2 = bessy2["Q5T2"]
quads_turnoff = [bessy2[name] for name in t2]
quads_optimize = [bessy2[name] for name in t1 + d2 + t2 + d3 + t3]
```

The optimization procedure consists of two parts, which use two different sets of quadrupoles defined in this snippet. The first part uses the `quads_turnoff` quadrupoles, corresponding to the quadrupoles in the T2 section. Here, each iteration lowers the value of the Q5T2 quadrupole step by step, while the optimizer tries to compensate the turn-off using the `quads_turnoff` quadrupoles. Then, in the next step, the optimizer tries to optimize the Twiss parameter for similarity with the reference optics. It uses the `quad_optimize` quadrupoles, corresponding to all quadrupoles from the T1 section to the T3 section.

```
def objective_function(values, quads):
    for quad, value in zip(quads, values):
        quad.k1 = value

    if not twiss.stable:
        return np.inf

    beta_beat_x = np.maximum(1, twiss.beta_x / beta_ref_x) ** 2
    beta_beat_y = np.maximum(1, twiss.beta_y / beta_ref_y) ** 2
    beat_mean = np.trapz(beta_beat_x + beta_beat_y, x=twiss.s) / bessy2.length
    return beat_mean + max(twiss.beta_y[t2_center], 1.6)
```

The snippet above corresponds to the objective function defined in Equation 4.5. The values of `twiss.beta_x` are not equidistant but depend on how long a given element is and how often it is sliced. Integrating the beta beat using the trapezoidal rule `np.trapz` takes the correct weighting factor into account. The `max(twiss.beta_y[t2_center], 1.6)` is used to incentivize a small vertical beta function in the center of the T2 section $\beta_\mathrm{y}(s_\mathrm{T2})$, but it is capped at 1.6 m as these should be enough to fulfill the requirements stated in Section 4.1.2.

```
def run(quads):
    result = optimize.minimize(
        objective_function,
        np.array([x.k1 for x in quads]),
        args=quads,
        method="Nelder-Mead",
    )
    print(f"k1: {q5t2.k1:+.3f}, objective: {result.fun:+.3f}")
    if not twiss.stable:
        raise Exception("Result must be stable!")
```

This snippet defines a run function, which takes one parameter, `quads`. That is the list of quadrupoles used for the optimization. If the Twiss parameters are not stable, the program stops with an error message.

```
for q5t2.k1 in np.linspace(q5t2.k1, 0, 10):
    run(quads_turnoff)

for _ in range(5):
    run(quads_optimize)

bessy2.as_file("bessy2_q5t2off.json")
```

In the last snippet, the optimization is executed. First, optimization is run using the quadrupoles `quads_turnoff`. For each iteration, the `k1` value of `q5t2` quadrupole is lowered until it reaches zero. In the second loop, the obtained optics is optimized using all quadrupoles. Finally, the optimized Q5T2off optics is saved to disk.

Appendix C

GUI to Calculate and Set new Power Supply Values

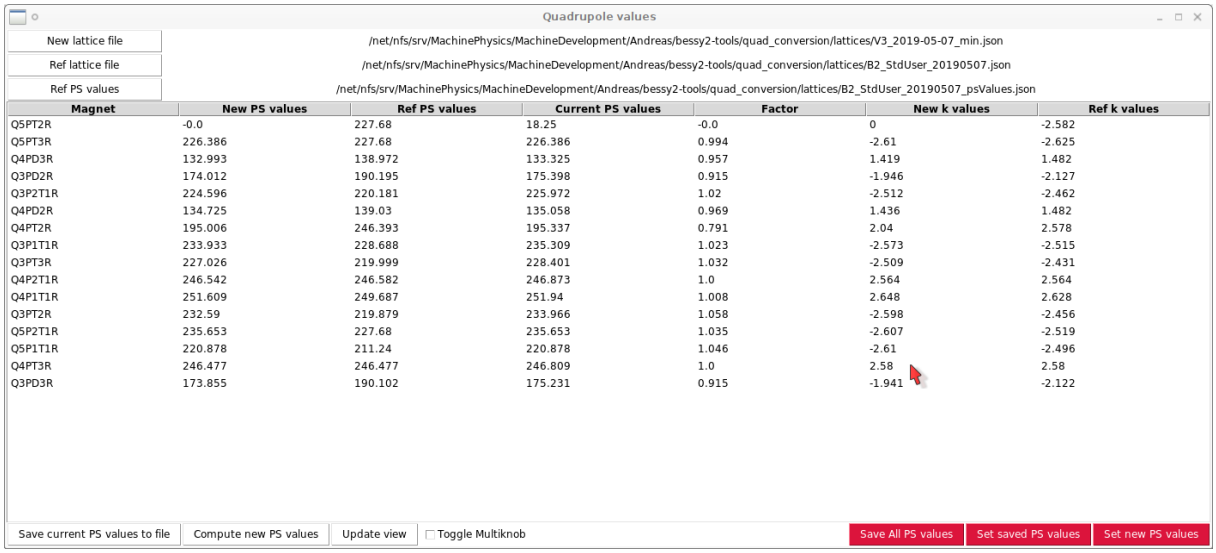

Figure 8.1: Screenshot of a simple application to calculate and set new power supply values

Transfering a new quadrupole setting to the machine by manually entering individual power supply values into the control system can be cumbersome and error-prone. Therefore, a simple application with a graphic user interface (GUI) was developed to automate this process. It is written in Python and uses the EPICS interface to interact with the machine. Figure C.1 shows a screenshot of the application. The application is visually split into three parts:

The first two buttons in the upper part load the quadrupole values of the *new lattice* and *reference lattice*. The files must be in the LatticeJSON format. The third button loads the power supply values of the *reference lattice*.

Below, a table shows the reference, current and, new power supply values. The new power supply values

$$I_{\text{new}} = \frac{k_{\text{new}}}{k_{\text{old}}} I_{\text{old}} \tag{C.1}$$

are calculated by the ratio of the new and old quadruple strength times the old power supply values.

The bottom frame contains multiple buttons: The first button stores all quadrupoles' current power supply values to a JSON file. During the LOCO measurement, that should be done to make sure the quadrupole values correspond to the power supply value. The second button computes the new power supply values. The multi-knob-toggle in the middle allows interpolating between two lattices. The user can choose a second lattice file and use a slider to interpolate between the quadrupole values of these two lattices. Finally, the three buttons in the right corner can set the new power supply values or set and restore the current quadrupole setting.

Appendix D

Transfer Matrices

Solving Equation 2.20 yields the matrix entries for the transversal offset $u(s)$ and slope $u'(s)$.

Two effects influence the longitudinal offset $l(s)$: First, the path length within an element can vary from the path length of the ideal orbit. Following from Equation 2.12, in linear approximation the total path length within an element of the length L is given by:

$$Z = \int_0^L (1 + \kappa_{\mathrm{x0}}(s)x)\mathrm{d}s \tag{D.1}$$

Secondly, the time scales linearly with the relative velocity error $\frac{\Delta v}{v_0}$. Using an approximation from relativistic kinematics

$$\frac{\mathrm{d}p}{\mathrm{d}v} = m_0\gamma + m_0\frac{v^2}{c^2}\gamma^3 = m_0\gamma^3(\frac{1}{\gamma^2} + \frac{v^2}{c^2}) \approx m_0\gamma^3 = \frac{p}{v}\gamma^2, \tag{D.2}$$

we can combine both effects and obtain an expression for the longitudinal offset

$$\begin{aligned} l(s+L) &= l(s) - (Z - L) + L\frac{\Delta v}{v_0} \\ &\approx l(s) - \int_s^{s+L} \kappa_{\mathrm{x0}}(s')x\mathrm{d}s' + \frac{L}{\gamma^2}\delta_0. \end{aligned} \tag{D.3}$$

Assuming no radiation, the relative momentum deviation

$$\delta(s) = \delta(0) = \delta_0 \tag{D.4}$$

stays constant, resulting in only one nonzero entry in the bottom row of the transfer matrices.

D.1 Drift Space

In a drift space of the length L, where $\kappa_{\mathrm{x0}} = k = 0$, Equation 2.20 simplifies to:

$$\begin{aligned} x'' &= 0 \\ y'' &= 0 \end{aligned} \tag{D.5}$$

The longitudinal offset $l(s)$ simplifies to:

$$l(s+L) = l(s) + \frac{L}{\gamma^2}\delta_0 \tag{D.6}$$

Thus, the transfer matrix of a drift space is:

$$\mathbf{R}_{\mathrm{drift}} = \begin{pmatrix} 1 & L & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & L & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & L/\gamma^2 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \tag{D.7}$$

D.2 Dipole Magnet

In a dipole magnet of the length L, where $\kappa_{\mathrm{x0}} \neq 0$ and $k = 0$, Equation 2.20 simplifies to

$$\begin{aligned} x'' + \kappa_{\mathrm{x0}}^2 x &= \kappa_{\mathrm{x0}}\delta \\ y'' &= 0, \end{aligned} \tag{D.8}$$

which is an inhomogeneous second-order differential equation. The transfer matrix for a bending magnet, solving Equation D.8, is

$$\mathbf{R}_{\mathrm{b}} = \begin{pmatrix} c & \frac{s}{\kappa_{\mathrm{x0}}} & 0 & 0 & 0 & \frac{1-c}{\kappa_{\mathrm{x0}}} \\ -\kappa_{\mathrm{x0}} s & c & 0 & 0 & 0 & s \\ 0 & 0 & 1 & L & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ -s & \frac{c-1}{\kappa_{\mathrm{x0}}} & 0 & 0 & 1 & \frac{L}{\gamma^2} - \frac{\varphi_0 - s}{\kappa_{\mathrm{x0}}} \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}, \tag{D.9}$$

where $\varphi_0 = L\kappa_{\mathrm{x0}}$, $s = \sin\varphi_0$, and $c = \cos\varphi_0$. The expression above corresponds to a sector magnet, where the entrance and exit angles are zero. The entrance and exit angles α, introduced by a rectangular dipole magnet, lead to an effect known as *edge focusing* and can be described by the matrix

$$\mathbf{R}_{\mathrm{e}} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ \kappa_{\mathrm{x0}} \tan\alpha & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & -\kappa_{\mathrm{x0}} \tan\alpha & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}. \tag{D.10}$$

Consequently, the transfer matrix for a rectangular dipole magnet is

$$\mathbf{R}_{\mathrm{b,r}} = \mathbf{R}_{\mathrm{e}} \mathbf{R}_{\mathrm{b}} \mathbf{R}_{\mathrm{e}}. \tag{D.11}$$

D.3 Quadrupole Magnet

In a quadrupole magnet of the length L, where $\kappa_{\mathrm{x0}} = 0$ and $k \neq 0$, Equation 2.20 simplifies to:

$$\begin{aligned} x'' + kx &= 0 \\ y'' - ky &= 0 \end{aligned} \tag{D.12}$$

The longitudinal offset $l(s)$ changes identical as in the drift space, leading to the transfer matrices for a horizontal focusing quadrupole

$$\mathbf{R}_{\mathrm{q,h}} = \begin{pmatrix} c & s/\sqrt{k} & 0 & 0 & 0 & 0 \\ -s\sqrt{k} & c & 0 & 0 & 0 & 0 \\ 0 & 0 & ch & sh/\sqrt{k} & 0 & 0 \\ 0 & 0 & sh\sqrt{k} & ch & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & L/\gamma^2 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \tag{D.13}$$

and for the vertical focusing quadrupole

$$\mathbf{R}_{\mathrm{q,v}} = \begin{pmatrix} ch & sh/\sqrt{k} & 0 & 0 & 0 & 0 \\ sh\sqrt{k} & ch & 0 & 0 & 0 & 0 \\ 0 & 0 & c & s/\sqrt{k} & 0 & 0 \\ 0 & 0 & -s\sqrt{k} & c & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & L/\gamma^2 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}, \tag{D.14}$$

where $s = \sin\sqrt{k}L$, $c = \cos\sqrt{k}L$, $sh = \sinh\sqrt{k}L$, and $ch = \cosh\sqrt{k}L$.

Appendix E

Grammar Files of Common Lattice File Formats

The grammar files are given in the Lark grammar language, which is based on the Extended Backus–Naur Form.

E.1 Elegant Grammar File

This grammar is not 100% consistent with elegants parser:

- Elegant's parser allows tokens to be split by the line continuation character "&". For example, it parses ANGLE=0.123& without an error. However, this is non-trivial to express with grammar rules and is therefore omitted.
- Elegant's parser allows a trailing " in attribute definitions. This means L=1.23" is parsed without an error. It seems like a bug and is left out.
- Elegant's parser allows unlimited trailing ",", which also seems like a bug.

```
%ignore /!.*/            // ignore comments
%ignore /[ \t\f]/+       // ignore whitespace
%ignore /&[ \t\f]*\r?\n/ // line continuation
%import common (SIGNED_INT, SIGNED_FLOAT, SIGNED_NUMBER, ESCAPED_STRING, CNAME)

int         : SIGNED_INT
float       : SIGNED_FLOAT
string      : ESCAPED_STRING
word        : /\w+/
name        : /\w+/ | "\"" /[\w:]+/ "\""
start        : _NEWLINE* (statement _NEWLINE+)*
_NEWLINE    : /[ \t\f]*\r?\n[ \t\f]*/
?statement  : element | lattice | command | "%" assignment
element     : name ":" [name] ("," attribute)* ","?
attribute   : word "=" (int | float | string | word)
lattice     : name ":" "LINE"i "=" arrangement
arrangement : [int "*"] [/-/] "(" object ("," object)* ")"
?object     : ref_name | arrangement
ref_name    : [int "*"] [/-/] ["\""] /[\w:]+/ ["\""]
command     : name ["," word]
```

Elegant used the so called reverse polish notation (RPN) for its arithmetic expressions. As there is no syntactic distinction between an escaped string and a variable, it is possible that a collision can happen. In this case a variable is wrongly identified as string.

```
assignment  : expr "sto" CNAME
?expr       : SIGNED_NUMBER -> number
            | CNAME         -> variable
            | function
            | binary
!function   : expr ("exp" | "sin" | "cos" | "tan" | "asin" | "acos" | "atan")
?binary     : expr expr "+" -> add
            | expr expr "-" -> sub
            | expr expr "*" -> mul
            | expr expr "/" -> div
?start_rpn  : assignment | expr // used to tested the rpn parser
```

E.2 MAD-X Grammar File

The following grammar is only a subset of MAD-X grammar, but it is sufficient to parse basic MAD-X lattice files.

```
%ignore /\s+/  // whitespace
%ignore "&" // backwards compatiable line continuation
%ignore /(!|\/\/).*/  // single line comments
%ignore /\/\*(\*(?!\/)|[^*])*\*\//  // multiline comment
%import common (SIGNED_INT, NUMBER, ESCAPED_STRING)

int         : SIGNED_INT
string      : ESCAPED_STRING
word        : /[\w\.]+/
start       : (_statement ";")*
_statement  : element | lattice | command | assignment
element     : word ":" [word] ("," attribute)* ","?
attribute   : word ("=" | ":=") (expr | string)
lattice     : word ":" "LINE"i "=" arrangement
arrangement : [int "*"] [/-/] "(" object ("," object)* ")"
?object     : ref_name | arrangement
ref_name    : [int "*"] [/-/] word
command     : word ("," (word | string | attribute))*
```

As there is no syntactic distinction between a non-escaped word and a variable, we must parse words as variables and test afterward if it is a variable or not.

```
assignment  : word ("=" | ":=") expr        → assignment
?expr       : item
            | "{" expr ("," expr)* ","? "}" → array
?item       : term
            | expr "+" term                 → add
            | expr "-" term                 → sub
?term       : factor
            | term "*" factor               → mul
            | term "/" factor               → div
?factor     : power
            | "+" factor                    → identity
            | "-" factor                    → neg
?power      : atom
            | power ("^" | "**") power      → pow
?atom       : NUMBER                        → number
            | word                          → variable // see 1.
            | word "(" expr ")"             → function
            | "(" expr ")"
?start_artih : assignment | expr  // used to tested the arithmetic parser
```

Appendix F

Documentation

The documentation is available as a Web version

https://apace.readthedocs.io

or as PDF

https://apace.readthedocs.io/_/downloads/en/latest/pdf/.

Bibliography

[1] M. Ruprecht, “Calculation of coupled bunch effects in the synchrotron light source BESSY VSR”, PhD thesis, Humboldt-Universität zu Berlin, 2016. doi: 10.18452/17446. (cit. in sec. 1.1, 1.2.1, 4.1.2, 4.1.2, 4.1.2)

[2] T. Atkinson *et al.*, “Commissioning of the 50 MeV preinjector linac for the BESSY II facility”, in *Proc. 2nd int. Particle accelerator conf. (IPAC’11)*, San Sebastian, Spain, 2011, pp. 3140–3142. (cit. in sec. 1.1)

[3] M. Abo-Bakr *et al.*, “Brilliant, coherent far-infrared (THz) synchrotron radiation”, *Phys. Rev. Lett.*, vol. 90, p. 094801, Mar. 2003, doi: 10.1103/PhysRevLett.90.094801. (cit. in sec. 1.1)

[4] A. Jankowiak *et al.*, *Technical design study BESSY VSR*. Helmholtz-Zentrum Berlin für Materialien und Energie, 2015. doi: 10.5442/R0001. (cit. in sec. 1.2.1, 1.2.1)

[5] F. Andreas, “Optimization of the BESSY II optics for the VSR project: Increase the installation length for the VSR cryomodule in the T2 section of the BESSY II storage ring”, Bachelor's thesis, Humboldt-Universität zu Berlin, 2017. (cit. in sec. 1.2.1, 2.3, 2.3.2, 2.6.2, 4.1, 4.1.1, 4.1.1, 4.1.2, 4.1.2, 4.1.3, 4.1.3)

[6] P. Kuske and F. Kramer, “Transverse emittance exchange for improved injection efficiency”, in *Proc. 7th int. Particle accelerator conf. (IPAC’16)*, Busan, Korea, May 2016, pp. 2028–2031. doi: doi:10.18429/JACoW-IPAC2016-WEOAA01. (cit. in sec. 1.2.2, 1.2.2)

[7] J. B. Murphy, *Synchrotron light source data book: Version 4, revision 05/96*. Brookhaven National Lab., 1996. doi: 10.2172/383670. (cit. in sec. 1.2.3, 4.3)

[8] H. Grote, F. Schmidt, L. Deniau, *et al.*, *Methodical accelerator design*. CERN, 2021. (cit. in sec. 1.3, 2.5, 3.5, 4.3, 5)

[9] M. Borland, *Elegant: A flexible SDDS-compliant code for accelerator simulation*. Argonne National Laboratory, 2021. (cit. in sec. 1.3, 3.5, 4.3, 5)

[10] Wikipedia, “Accelerator physics codes”. https://en.wikipedia.org/wiki/Accelerator_physics_codes (accessed Sep. 26, 2021). (cit. in sec. 1.3)

[11] M. Borland, *SDDS and SDDS-compliant programs: A modular system for accelerator design, simulation, control, and analysis*. Argonne National Laboratory. Available: https://ops.aps.anl.gov/SDDSInfo.shtml (cit. in sec. 1.3)

[12] K. Fuchsberger and Y. I. Levinsen, "PYMAD – integration of MADX in Python", in *Proc. 2nd int. Particle accelerator conf. (IPAC'11)*, San Sebastian, Spain, 2011, pp. 2289–2291. (cit. in sec. 1.3)

[13] T. Gläßle, R. D. Maria, Y. Levinsen, and K. Fuchsberger, *Cpymad: Cython binding to MAD-X*. Heidelberg Ion-Beam Therapy Center (HIT): Zenodo, 2021. doi: 10.5281/zenodo.5364513. (cit. in sec. 1.3)

[14] G. van Rossum *et al.*, *CPython*. GitHub. Available: https://github.com/python/cpython (cit. in sec. 1.3, 3.1)

[15] H. Wiedemann, *Particle Accelerator Physics*. Berlin, Germany: Springer, 2015. doi: 10.1007/978-3-319-18317-6. (cit. in sec. 2)

[16] A. Wolski, *Introduction to beam dynamics in high-energy electron storage rings*. Morgan & Claypool Publishers, 2018. doi: 10.1088/978-1-6817-4989-1. (cit. in sec. 2, 2.5.2)

[17] K. Wille, *The physics of particle accelerators: An introduction*. Clarendon Press, 2001. (cit. in sec. 2)

[18] F. Hinterberger, *Physik der Teilchenbeschleuniger und Ionenoptik*, 2. Aufl. Springer, 2008. doi: 10.1007/978-3-540-75282-0. (cit. in sec. 2, 4.1.5)

[19] E. D. Courant and H. S. Snyder, "Theory of the alternating-gradient synchrotron", *Annals of Physics*, pp. 1–49, 1954, doi: 10.1016/0003-4916(58)90012-5. (cit. in sec. 2.3, 2.3.2)

[20] W. Magnus and S. Winkler, *Hill's equation*. 1966. (cit. in sec. 2.3.2)

[21] W. Nolting, *Theoretical physics 8 - statistical physics*. Berlin, Heidelberg: Springer, 2018. (cit. in sec. 2.3.2, 2.5)

[22] M. Sands, "The physics of electron storage rings: An introduction", 1970, doi: 10.2172/4064201. (cit. in sec. 2.5, 2.5.1, 2.5.2)

[23] R. H. Helm, M. J. Lee, P. L. Morton, and M. Sands, "Evaluation of synchrotron radiation integrals", in *IEEE Transactions on Nuclear Science*, 1973, vol. 20, pp. 900–901. doi: 10.1109/TNS.1973.4327284. (cit. in sec. 2.5)

[24] R. Chasman, G. K. Green, and E. M. Rowe, "Preliminary design of a dedicated synchro-

tron radiation facility", in *IEEE Transactions on Nuclear Science*, 1975, vol. 22, pp. 1765–1767. doi: 10.1109/tns.1975.4327987. (cit. in sec. 2.6.2)

[25] L. Farvacque, J. L. Laclare, P. Nghiem, J. Peyet, H. Tanaka, and A. Tkatchenko, "Possible retuning of the ESRF storage ring lattice for reducing the beam emittance", in *Proc. 4th european particle accelerator conf. (EPAC'94)*, London, UK, 1994, pp. 612–615. (cit. in sec. 2.6.2)

[26] G. van Rossum *et al.*, *Faster CPython*. GitHub. Available: https://github.com/faster-cpython (cit. in sec. 3.1)

[27] J. Lehtosalo, M. J. Sullivan, *et al.*, *Mypyc: Compile type annotated Python to fast C extensions*. GitHub. Available: https://github.com/mypyc/mypyc

[28] Facebook, *Cinder: Instagram's performance oriented fork of CPython*. GitHub. Available: https://github.com/facebookincubator/cinder (cit. in sec. 3.1)

[29] A. Rigo *et al.*, *PyPy: A fast, compliant alternative implementation of Python*. Available: https://www.pypy.org/ (cit. in sec. 3.1)

[30] K. Hayen *et al.*, *Nuitka: A Python compiler written in Python*. GitHub. Available: https://github.com/Nuitka/Nuitka (cit. in sec. 3.1)

[31] C. R. Harris *et al.*, "Array programming with NumPy", *Nature*, vol. 585, no. 7825, pp. 357–362, Sep. 2020, doi: 10.1038/s41586-020-2649-2. (cit. in sec. 3.1)

[32] S. K. Lam, A. Pitrou, and S. Seibert, "Numba: A LLVM-based Python JIT compiler", in *Proceedings of the second workshop on the LLVM compiler infrastructure in HPC*, 2015, pp. 1–6. (cit. in sec. 3.1)

[33] S. Guelton, P. Brunet, M. Amini, A. Merlini, X. Corbillon, and A. Raynaud, "Pythran: Enabling static optimization of scientific Python programs", *Computational Science & Discovery*, vol. 8, no. 1, p. 014001, 2015. (cit. in sec. 3.1)

[34] A. Rigo *et al.*, *CFFI: C foreign function interface for Python*. Available: https://cffi.readthedocs.io/en/latest (cit. in sec. 3.1)

[35] R. Chandra, L. Dagum, D. Kohr, R. Menon, D. Maydan, and J. McDonald, *Parallel programming in OpenMP*. Morgan kaufmann, 2001. (cit. in sec. 3.1)

[36] J. D. Hunter, "Matplotlib: A 2D graphics environment", *Computing in Science & Engineering*, vol. 9, no. 3, pp. 90–95, 2007, doi: 10.1109/MCSE.2007.55. (cit. in sec. 3.1)

[37] Wikipedia, "Lattice file format and data interchange issues". https://en.wikipedia.org/w-

iki/Accelerator_physics_codes#Lattice_file_format_and_data_interchange_issues (accessed Sep. 26, 2021). (cit. in sec. 3.5)

[38] D. Sagan *et al.*, "The accelerator markup language and the universal accelerator parser", Edinburgh, UK, 2006. (cit. in sec. 3.5)

[39] D. Sagan, D. A. Bates, and A. Wolski, "The universal accelerator parser", in *Proc. 9th int. Computational accelerator physics conf. (ICAP'06)*, Chamonix, Switzerland, 2006, pp. 303–307. (cit. in sec. 3.5)

[40] D. Crockford, "JSON (JavaScript object notation)", Accessed: Sep. 26, 2021. [Online]. Available: https://www.json.org (cit. in sec. 3.5)

[41] "JSON schema", Accessed: Sep. 26, 2021. [Online]. Available: https://json-schema.org/ (cit. in sec. 3.5)

[42] F. Andreas, *LatticeJSON: A JSON based lattice file format*. Accessed: Sep. 26, 2021. [Online]. Available: https://github.com/nobeam/latticejson (cit. in sec. 3.5)

[43] E. Jaeschke *et al.*, "Lattice design for the 1.7-GeV light source BESSY II", in *Proc. 15th particle accelerator conf. (PAC'93)*, Washington D.C., USA, 1993, pp. 1474–1477. (cit. in sec. 4.1.1)

[44] H. Bäcker *et al.*, "Status of the BESSY II femtosecond x-ray source", Lucerne, Switzerland, 2004. Available: http://accelconf.web.cern.ch/e04/papers/THPKF014.pdf (cit. in sec. 4.1.1)

[45] K. Holldack, T. Kachel, S. Khan, R. Mitzner, and T. Quast, "Characterization of laser-electron interaction at the BESSY II femtoslicing source", in *Physical review special topics - Accelerators and beams*, 2005, vol. 8, p. 040704. doi: 10.1103/PhysRevSTAB.8.040704. (cit. in sec. 4.1.1)

[46] J. Bahrdt *et al.*, "Modifications to the machine optics of BESSY II necessitated by the EMIL project", in *Proc. 3rd int. Particle accelerator conf. (IPAC'12)*, New Orleans, LA, USA, May 2012, pp. 1614–1616. (cit. in sec. 4.1.1)

[47] P. Kuske and J. Li, "Performance improvements of the BESSY II storage ring by optimizing the phase acceptance", May 2017. doi: 10.18429/JACoW-IPAC2017-WEPAB006. (cit. in sec. 4.1.1, 4.1.5, 4.1.5)

[48] J. Safranek, "Linear optics from closed orbits (LOCO): An introduction", in *ICFA Beam Dyn. Newslett.*, 2007, vol. 44, pp. 43–49. (cit. in sec. 4.1.5)

[49] J. Safranek, G. Portmann, and A. Terebilo, "MATLAB-BASED LOCO", in *Proc. 8th euro-*

pean particle accelerator conf. (EPAC'02), Paris, France, 2002, pp. 1184–1186. (cit. in sec. 4.1.5)

[50] G. J. Portmann, W. J. Corbett, and A. Terebilo, "An accelerator control middle layer using matlab", in *Proc. 21st particle accelerator conf. (PAC'05)*, Knoxville, TN, USA, May 2005, pp. 4009–4011. (cit. in sec. 4.1.5)

[51] E. You *et al.*, *Vue.js: The progressive JavaScript framework*. 2021. Available: https://vuejs.org/ (cit. in sec. 4.3.4)

[52] E. You *et al.*, *Vite: Next generation frontend tooling*. 2021. Available: https://vitejs.dev/ (cit. in sec. 4.3.4)

Declaration of Authorship

I, Felix Andreas, declare that this thesis and the work presented in it are my own. I confirm that where I have quoted from the work of others, the source is always given.

Signature

Place, Date Berlin, December 20, 2021