Extraction of the Scalar Wave in $J/\psi \to \gamma\pi^0\pi^0$ using the ComPWA Framework

Dissertation
zur Erlangung des Grades
"Doktor
der Naturwissenschaften"
am Fachbereich Physik, Mathematik und Informatik
der Johannes Gutenberg-Universität
in Mainz

Mathias Michel

geb. in Bad Schwalbach
Mainz, den 29.03.2016

1. Gutachter:	-
2. Gutachter:	-
Datum der mündlichen Prüfung:	15.07.2016

Abstract

The origin of the restmass of the proton is still a mystery and thus a central question of hadron physics: The valance quarks of the proton cover only 2% of its mass, the remaining 98% are created by the strong coupling which consists of exchange particles (gluons) and quantum fluctuations (seaquarks). However, the coupling of the strong interaction has characteristics which are not yet fully understood. The question of the nature of the strong interaction is equal to the question if other, so called exotic particles exist, which have a different inner structure than the conventional hadrons (mesons and baryons).
Thus, the main topic of the physics program of the PANDA experiment at FAIR and BESIII in Bejing deals with the search for new conventional and exotic hadronic states like e.g. hybrids, tetraquark states, and glueballs. For many investigations in the field of hadron spectroscopy a complex amplitude analysis, e.g. a partial wave analysis, is necessary in order to identify possible candidates and for the classification of known states. As an example of how elaborate analyses can get with the increasing precision and amount of data available, a model-independent extraction method for the individual amplitude contribution is presented and tested with Monte Carlo generated $J/\psi \to \gamma\pi^0\pi^0$ data.
For this, and many more analysis techniques, a new, agile, and efficient amplitude analysis framework named *ComPWA* is being developed. It is modularized to allow an easy extension with arbitrary models and formalisms as well as simultaneous fitting of multiple datasets, even from different experiments. Experience from existing amplitude analysis software was used to define the requirements of the framework and in particular to prevent it from restricting its functionality to certain analysis methods. The challenges involve parallelization, fitting with a high number of free parameters, managing complex meta-fits, and quality assurance / comparability of fits. In this work, the design of the framework, the implementation of the wave extraction method, and the results from the $J/\psi \to \gamma\pi^0\pi^0$ analysis are presented.

Kurzfassung

Die zentrale Frage der Hadronenphysik, die bislang noch nicht vollständig geklärt werden konnte, ist die Frage, woher das Proton seine Ruhemasse erhält. Nur etwa 2% seiner Ruhemasse stammen von den Valenzquarks, die restlichen 98% werden von der sehr energiereichen Bindung erzeugt, welche aus Austauschteilchen (Gluonen) und Quantenfluktuationen (Seequarks) besteht. Allerdings konnten bisher nicht alle Eigenschaften der Bindung in der starken Wechselwirkung ergründet werden. Die Frage nach der Natur der starken Wechselwirkung ist gleichbedeutend mit der Frage, ob es andere, sogenannte exotische Teilchen gibt, die eine andere innere Struktur besitzen als die konventionellen Hadronen (Mesonen und Baryonen).
Ein Großteil des Physikprogramms der Experimente PANDA an FAIR und BESIII in Peking umfasst daher die Suche nach bislang unentdeckten konventionellen und exotischen Zuständen wie z.B. Hybriden, Tetraquarkzuständen und Gluon-Bällen. Für die meisten Analysen im Rahmen der Hadronspektroskopie ist eine aufwendige, sogenannte Amplitudenanalyse wie z.B. eine Partialwellenanalyse notwendig, um mögliche Kandidaten zu identifizieren und bekannte Zustände zu klassifizieren. Als Beispiel dafür, wie die erhöhte Genauigkeit und die Verfügbarkeit größerer Datenmengen neue Anforderungen an Analysemethoden stellt, wird hier eine modellunabhängige Extraktion von Wellen vorgestellt und an $J/\psi \to \gamma\pi^0\pi^0$ Monte-Carlo-Daten getestet.
Für diese und weitere zukünftige Methoden wird ein neues, flexibles und effizientes Amplitudenanalyse-Framework namens *ComPWA* entwickelt. Es ist modular aufgebaut, um eine möglichst einfache Erweiterungen durch neue Modelle und Formalismen zu ermöglichen. Darüber hinaus erlaubt es die simultane Analyse mehrerer Datensätze, sogar wenn diese von verschiedenen Experimenten stammen. Um Einschränkungen bezüglich der Anwendung sowie für die Weiterentwicklung der Software zu verhindern, wurde die Erfahrung aus den Softwarepaketen bisheriger Analyseprogramme anderer Experimente für die Entwicklung von ComPWA genutzt. Die Anforderungen umfassen die Parallelisierung, Modelle mit vielen freien Parametern, Verwaltung von komplexen Analysen, das Erreichen von besserer Vergleichbarkeit zwischen Analysen und die einfache Kontrolle systematischer Unsicherheiten. In dieser Arbeit werden der Entwurf des ComPWA-Frameworks, die Entwicklung der Wellenextraktion und die Ergebnisse der $J/\psi \to \gamma\pi^0\pi^0$ Analyse vorgestellt.

Danksagung

Das Universum ist kalt, einsam und unwirtlich. Leben, so weit wir wissen, wird darin einzig durch das Licht von Sternen ermöglicht. Unsere Erde besitzt lediglich eine Lichtquelle, die Sonne, in ihrer Nachbarschaft. Mir geht es anders. Ohne die vielen Lichter, die mir in meinem Leben Wege aufzeigen und meine Schritte leiten, hätte ich mich vermutlich schon oft verlaufen. Ich musste nie herausfinden, wie weit ich alleine kommen würde, da ich immer das maßlose Glück hatte von vielen Lichtern umgeben zu sein. Dabei kommen sie in allen Farben vor, verändern sich, sind mal nah, mal fern, mal groß, mal klein. Manche sind beständig, ihr helles Licht ist mir wohl bekannt und sie begleiten mich schon so lange, dass ich manchmal vergesse was für ein Glück ich mit ihnen habe. Einige sind mir sehr nah, verändern sich mit mir und fühlen sich an als wären sie ein Teil von mir. Andere leuchten in der Ferne, aber sehe ich sie immer im Augenwinkel und weiß, ich kann jederzeit zu ihnen wenn ich mich verloren fühle. Es gibt auch Lichter die mir nur kurz begegnen, weil sie mir auf einem komplizierten Pfad helfen oder einfach glücklicherweise in dieselbe Richtung unterwegs sind. Aber egal wie kurz dieser Abschnitt auch sein mag, ihr Licht beeinflusst mich und lässt mich wachsen, sodass ich mich immer an sie erinnern werde. Ab und zu verliere ich auch lang vertraute Lichter, weil sich Wege trennen oder ihr Licht in eine andere Richtung leuchtet. Manchmal verlöschen Lichter auch einfach so, unerwartet und endgültig. Aber auch sie hinterlassen Spuren, dadurch, wie sie mich verändert haben und durch die anderen Lichter, die durch den Verlust noch heller leuchten.
Danke, Euch allen.

Contents

Motivation **1**

1 Hadron Spectroscopy **5**
1.1 Charmonium . . . 8
1.2 Light Mesons . . . 12

2 Experimental Setups **17**
2.1 The PANDA Experiment . . . 17
2.2 The BESIII Experiment . . . 22
2.3 Common Objectives . . . 24

3 ComPWA Framework **25**
3.1 Concept . . . 25
3.2 General Layout . . . 26
3.3 Existing Software Packages . . . 27
3.4 Requirements . . . 28
3.5 Software Design . . . 31
3.6 Modules and Interfaces . . . 32
3.7 Function Tree . . . 35
3.8 Implementation of the Modules . . . 38
3.9 Expert Systems . . . 44

4 Analysis of $J/\psi \to \gamma\pi^0\pi^0$ **47**
4.1 Reaction Channel . . . 47
4.2 Toy Data . . . 50
4.3 Dalitz Plot Fit . . . 53
4.4 Model Independent Fit (Slice Fit) . . . 55
4.4.1 Slice Fit with Binned Likelihood . . . 57
4.4.2 Slice Fit with Unbinned Likelihood . . . 61
4.4.3 Slice Fit Comparison . . . 71
4.5 Evaluation of Fit Methods . . . 72
4.6 Function Tree for Breit-Wigner Sum Model . . . 81
4.7 Optimization Tests . . . 84

Summary **85**

List of Tables **87**

List of Figures **89**

Bibliography **91**

A Appendix **95**
A.1 Helicity Amplitude Calculation . . . 95
A.2 Additional Toy Data Studies . . . 109
A.3 Pull Distributions . . . 115
A.4 CLIPS Expert System Test . . . 118

Motivation

The nuclei of the atoms in our universe are made out of neutrons and protons which themselves are bound states of fundamental particles called quarks. Quarks are the objects underlying the strong interaction, which is described by a theory called Quantum Chromodynamics (QCD). Since the gluons (g), the mediator gauge bosons of the strong interaction, are self-interacting, the strength of the coupling constant α_s decreases at high energies. Therefore, different methods of calculating QCD are needed for different energy regimes, because there is no quantitative description of the whole energy regime present yet. Perturbation theory can be used with great success for high energies where α_s becomes very small, but at lower energies it sharply rises in the strong coupling strength causing the quark confinement and thus rendering perturbation approaches inapplicable. In this regime, predictions of hadronic states come from either lattice QCD, effective field theories, or potential model calculations. The most prominent experimental results in this field are the discoveries of states. However, to better understand the hadron structure, creation processes and ordering, new measurements and analyses have to provide more accurate and detailed properties of these states. In order to deduce quantum numbers of resonances like the spin, typically techniques such as amplitude analysis methods have to be applied.

The constituent quark model describes hadrons as bound states of their valence quark content, which also determines the quantum numbers of the hadron. Although this information explains the quantum properties, it does not provide a quantitative description of the hadrons' mass. The proton's quantum numbers for example indicate a composition of two u and one d quark, however, the sum of the bare masses of these three constituents being 9.4 MeV/c^2 underestimates the actual proton mass of 938.27 MeV/c^2 by roughly two orders of magnitude. The main fraction of the proton's mass necessarily comes from other sources: Virtual quark-antiquark pairs and a gluon field, caused by the self-interacting nature of the gluons, are formed inside the hadrons.

To date, there are three generations of quarks established. Table M.1 shows an overview of the properties of all known quarks, called up (u), down (d), strange (s), charm (c), bottom (b), and top (t). The names represent the three generations of oppositely charged quark pairs. Not shown are the antiquarks ($\overline{u}, \overline{d}, \overline{s}, \overline{c}, \overline{b}, \overline{t}$) of all six quark types, which have opposite charge but the same mass as their counterparts. The u, d, and s masses are so-called "current-quark masses" in the $\overline{\text{MS}}$ scheme (modified minimal subtraction scheme, a renormalization to absorb the infinities that arise in perturbative calculations beyond leading order [1]). The c, b, and t masses are the "running" masses in the $\overline{\text{MS}}$ scheme, see [2] for other mass estimations and further explanations.

Table M.1: Properties of the quark generations according to PDG [2].

Generation	I		II		III	
Name	u	d	s	c	b	t
Mass [MeV/c^2]	$2.3^{+0.7}_{-0.5}$	$4.8^{+0.7}_{-0.3}$	95 ± 5	1275 ± 25	4180 ± 30	160000^{+5000}_{-4000}
$I(J^P)$	$\frac{1}{2}(\frac{1}{2}^+)$	$\frac{1}{2}(\frac{1}{2}^+)$	$0(\frac{1}{2}^+)$	$0(\frac{1}{2}^+)$	$0(\frac{1}{2}^+)$	$0(\frac{1}{2}^+)$
Charge	$\frac{2}{3}\,e$	$-\frac{1}{3}\,e$	$-\frac{1}{3}\,e$	$\frac{2}{3}\,e$	$-\frac{1}{3}\,e$	$\frac{2}{3}\,e$

Beyond the masses and electrical charges, the table also shows quantum numbers of the quarks. The $I(J^{PC})$ nomenclature is widely used in particle physics and summarizes the quantum numbers I (isospin, which was introduced to treat particles with similar masses coupling to the strong force as different states of one particle), J (total angular momentum, which combines spin and orbital angular momentum), P (parity, which represents the behavior under sign change of spatial coordinates), and C (C-parity, which represents the behavior under charge conjugation of all inner quantum numbers, e.g. the electrical charge).
The simplest configurations of quarks allowed by QCD principles are called mesons, consisting of a valence quark antiquark pair ($q\overline{q}$), and baryons, consisting of three quarks (qqq) and their antiparticles. The mass eigenstates of the three lightest quarks up, down and strange, can be described by SU(3) group if the mass-perturbation is neglected [3] [4]. This approach is a good approximation due to the small mass differences between u, d, and s. Figure M.1 and M.2 show the SU(3) multiplets of possible quark combinations, ordered by isospin and strangeness: the strangeness quantum number is zero for all quarks except for strange quarks, with $S = -1$ or $S = 1$ for the corresponding antiquark. The quark antiquark pair $u\overline{d}$, for example, is labeled as the charged pseudo scalar meson π^+ in figure M.1 which has no strange content. If one replaces the antidown quark with an anti strange quark, creating the quark pair $u\overline{s}$, one ends up with the K^+ meson. The strange antiquark contributes a strangeness of $S = +1$ to the K^+ meson, but it does not carry isospin, which means the K^+ meson has only half the isospin of the π^+ meson. Therefore, the K^+ is positioned left (less isospin) and above (more strangeness) the π^+ in figure M.1.
The same ordering applies to the baryons shown in figure M.2: The proton p consists of uud and therefore has $S = 0$. The $S = -1$ hyperon Σ^+ differs from the proton only by having a strange quark instead of the down quark, it consists of uus. Its position in figure M.2 is right and below of the proton: the strange quark adds negative strangeness but has no isospin charge, therefore the isospin of the two up quarks add up to a higher value than the proton's isospin, where the down quark has no negative isospin contribution.

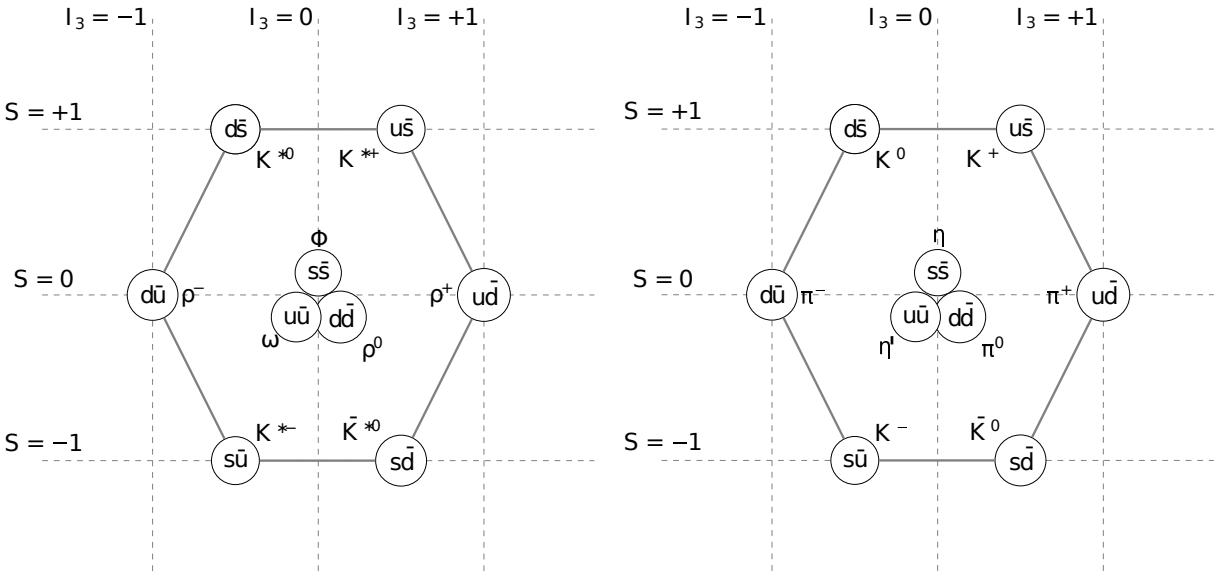

Figure M.1: The SU(3) meson multiplets ordered by isospin I_3 and strangeness S. On the left side, the vector meson ($J^P = 1^-$) nonet is shown. On the right side, the pseudo scalar meson ($J^P = 0^-$) nonet is shown.

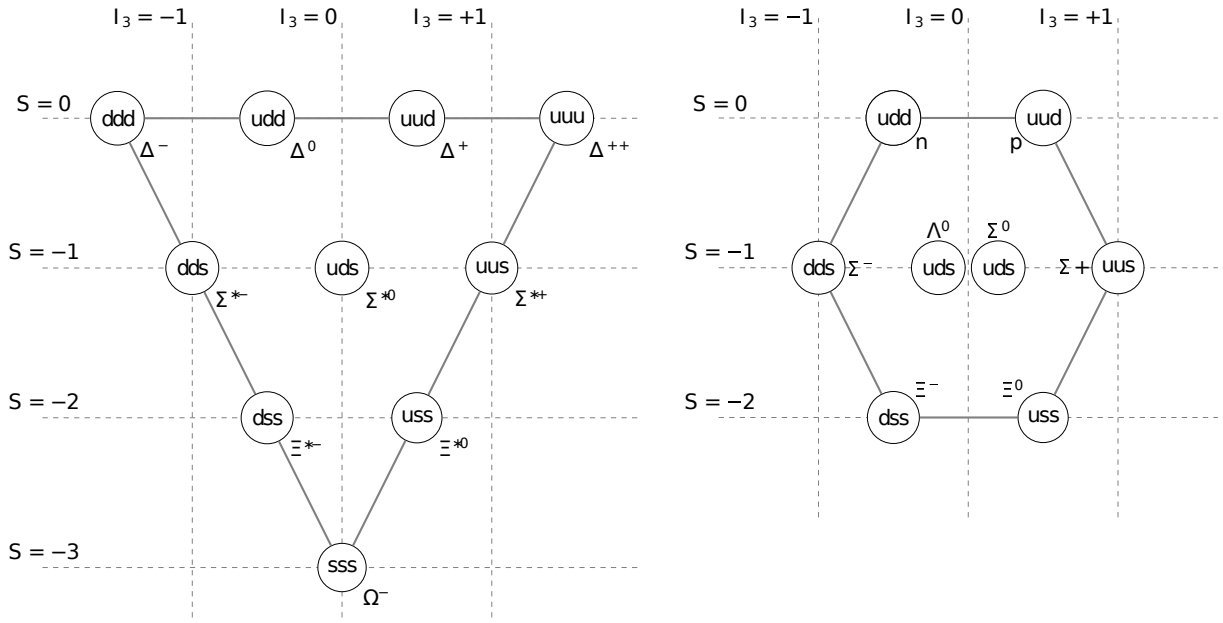

Figure M.2: The SU(3) baryon multiplets ordered by isospin I_3 and strangeness S. On the left side, the $J^P = \frac{3}{2}^+$ baryon decuplet is shown. On the right side, the $J^P = \frac{1}{2}^+$ baryon octet is shown. Only baryonic states, consisting purely of quarks, are shown here. Antibaryons consisting of antiquarks form equivalent multiplets.

It is getting more complicated when considering hadronic states with more complex configurations. The results of previous and recent measurements show that there are more states observed than expected by simple combinations of quarks, and the masses (one of the main ordering criteria) of the states are far off the various model predictions. To explain this situation, theoretical models include additional glue content, additional quarks and antiquarks, or propose molecular states to deliver possible interpretations for these unpredicted hadronic states. The measurement of their properties and quantum numbers as well as the extraction of complete multiplets of states is crucial for the understanding of the nature of these hadronic states and to learn about the underlying principles.

The next generation of multi-purpose high-energy physic experiments, e.g. the PANDA experiment [5] at FAIR, will provide precise data using accelerators with high luminosities. The increase in available data and accuracy makes detailed hadron physics analyses possible, such as measuring properties of observed states and determining their quantum numbers as well as branching fractions in different channels. This induces new demands on software tools which are needed to match models to the data, e.g. to perform a partial wave analysis [6]. In a partial wave analysis, the amplitude to describe scattering processes by decomposing waves into constituent components, e.g. into angular momentum contributions. Amplitudes can be described in various ways, thus the more general term amplitude analysis is also common.

The goal of this work is to set up and test a model-independent approach to extract waves and thereby learn more about the nature of the resonances under investigation. As the development of this method requires a certain final state configuration, the channel $J/\psi \to \gamma\pi^0\pi^0$ was chosen for the Monte Carlo studies presented in this work. From the physics point of view, this channel is interesting for this kind of analysis because of the scalar resonances appearing in the $\pi^0\pi^0$ system. Since those are still a large source of debate, a model-independent answer could be given by extracting the wave instead of single resonances. To demonstrate the method it is important that this final state provides access to a range of mesonic resonances decaying to $\pi^0\pi^0$, while the few resonances in the $\pi^0\gamma$ final state are well understood. Although this particular channel was chosen for some first studies, the method shown might be used for other analyses, e.g. at PANDA, as well. J/ψ data from BESIII [7] is already available for testing. The next step should be an analysis of the data with this software and a comparison to

the already performed analyses of this channel, see [8] and [9]. Elaborate techniques like the extraction method require flexible software tools which are not restricted to specific methods, therefore the common PWA framework (ComPWA) described in this thesis was developed to enable these techniques.
The first chapter summarizes the present knowledge of hadronic resonances relevant for the investigated reaction. The established resonances and properties are shown as well as open questions and possible theoretical interpretations. Chapter 2 introduces the experimental setups PANDA and BESIII, which are able to provide the necessary measurements to perform new analysis methods. Their physics programs and detector setups are briefly presented but especially highlighted is the ability of the experiments to provide hadron spectroscopy measurements. The third chapter introduces and describes the ComPWA framework. Starting with the concept and design of ComPWA in the scope of the present amplitude analysis methods, this chapter covers the present status of the framework and introduces all modules implemented to perform and compare the wave extraction method. Also, unique features of ComPWA are presented. In chapter four the wave extraction method is discussed. The channel of interest is presented, the model to generate the Monte Carlo data is introduced and the analysis is performed. The features and challenges of the method are discussed and results of the Monte Carlo analysis are presented. Afterwards, a summary of the gained insights and an outlook to further steps important to enhance the wave extraction method as well as ComPWA is shown.

1 Hadron Spectroscopy

One important tool to understand the principles of hadronic particles is the method of spectroscopy. Spectroscopy in general is the method of extracting information from an observed spectrum, e.g. an energy spectrum, which reflects the internal structure of a system. A classical example is the splitting of light into a spectrum of wavelengths. This can be done by using a prism, as shown in figure 1.1. The white light is a superposition of light with different wavelengths. When entering the prism, the light changes its speed because of the different medium, which causes refraction. The amount of refraction depends on the wavelength of the light, so the wavelengths of the light are separated into a spectrum of different colors.

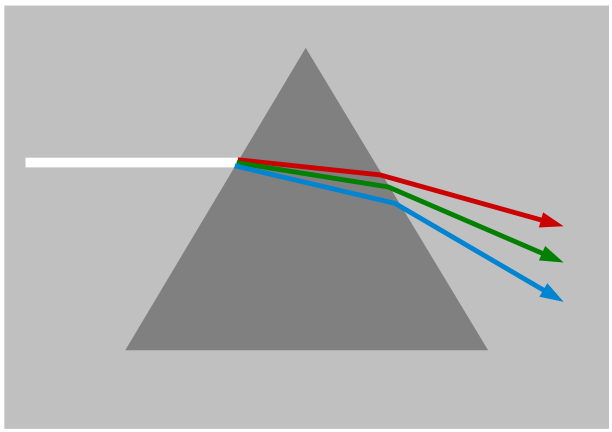

Figure 1.1: Dispersion of white light in a prism. White light is a mixture of the wavelengths of the visible spectrum. Waves of different wavelengths undergo different amounts of refraction.

Studying the corresponding spectrum of a natural light source like the sun allows the deduction of its chemical composition and gas properties: Since electrons populate distinct energy levels depending on the species of the atom, only light with the wavelengths corresponding to the transition between these levels are absorbed or emitted. For example, figure 1.2 shows the energy levels and transitions of the hydrogen atom. The observation of these series gave rise to a further understanding of the atom, e.g. with the orbital model which explained the stability of the electron orbits by introducing angular momentum as a quantum number. This marks a milestone towards the exact description of electromagnetic interaction by using quantum electrodynamics.

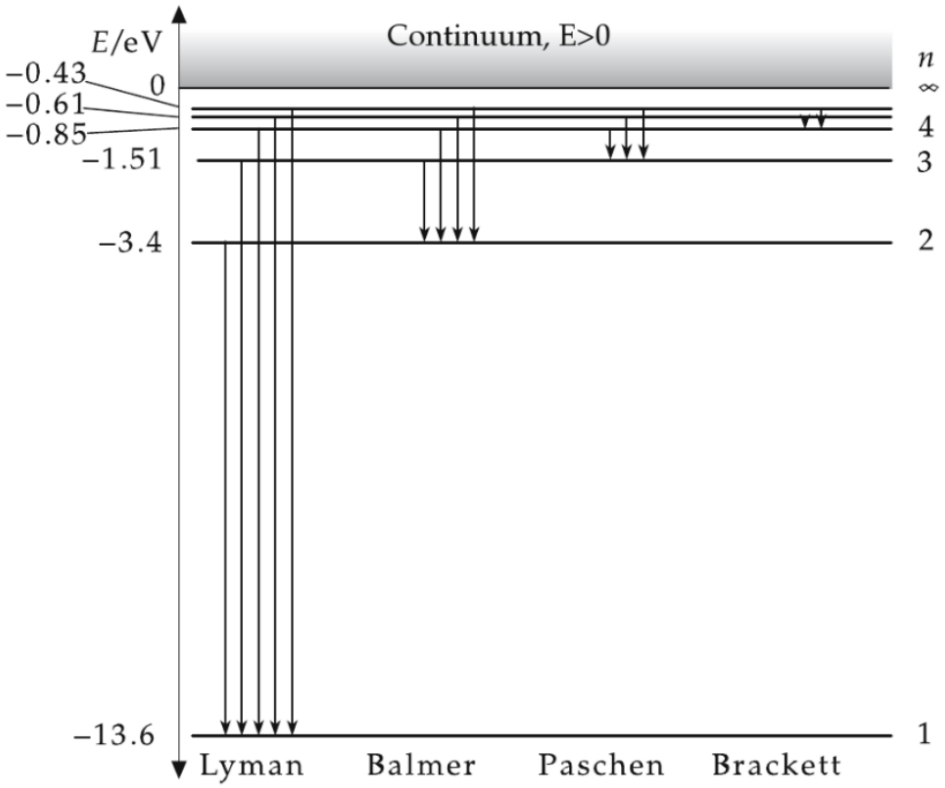

Figure 1.2: Energy level scheme and the transition series of the hydrogen atom [10].

In hadron spectroscopy, the spectrum of interest shows hadronic resonant states e.g. produced in scattering experiments. A typical hadronic state is the J/ψ resonance as shown as a peaking structure in the invariant mass spectrum of its decay particles. Figure 1.3 shows one of the discovery plots of the first charm anticharm resonant state observed, the vector state. Hadronic states are characterized by their properties: the lineshape, which is reflected in the invariant mass spectrum of e.g. the decay particles, but also quantum numbers, decay products and behavior at decay thresholds. The states in the spectrum of figure 1.4 are ordered by quantum number versus mass. The investigation of the hadron spectrum and the measurement of the properties of the hadronic resonances can be performed in two ways: in scan experiments, where the resonance of interest is formed directly from the incoming particles and its lineshape is determined by variation of the beam energy, or in an indirect way via the decay products, when also additional particles are produced. In this case the line shape of the resonance of interest is measured directly, but the precision is limited by the detector resolution. Scan experiments are only possible if the initial reaction allows for the direct production of the hadrons of interest.

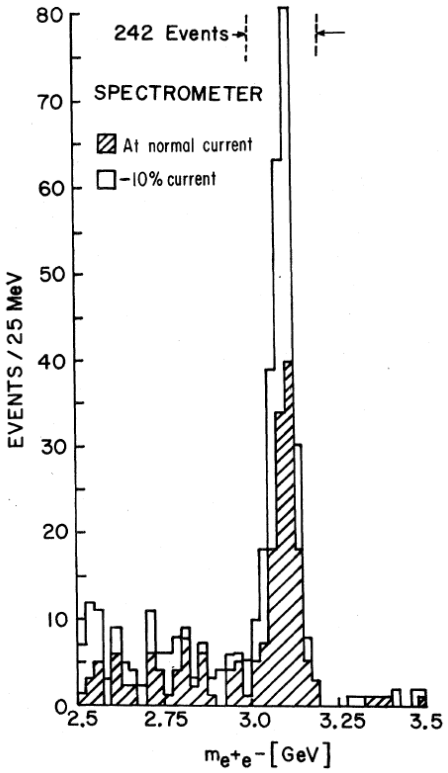

Figure 1.3: Mass spectrum of the original measurement at the Brookhaven National Laboratory discovering the J/ψ in 1974 [11]. Two spectrometer settings are plotted to show that the peak is independent of spectrometer currents.

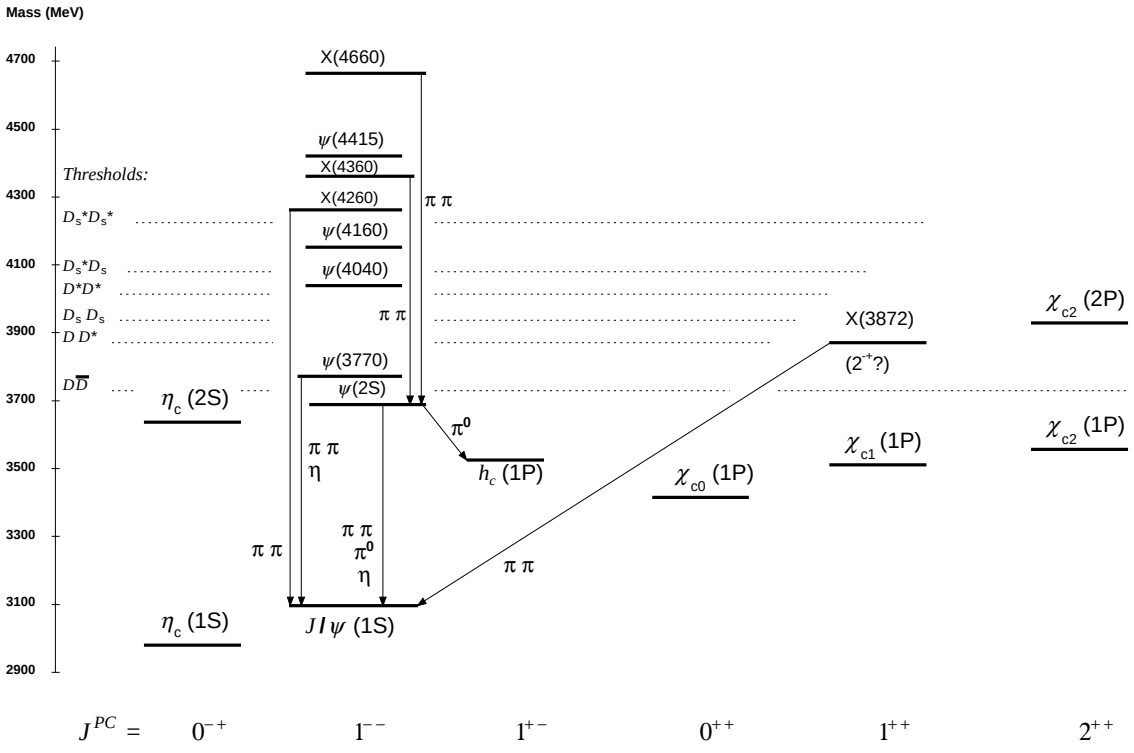

Figure 1.4: The level scheme of experimentally established charmonium states according to the Particle Data Group [2]. Unassigned states are labeled with X. The observed hadronic transitions are indicated.

1.1 Charmonium

In 1974, the J/ψ resonance was discovered independently at two experimental setups and confirmed soon from several more. This, and the following discoveries of additional narrow resonances, mark the beginning of the charm and charmonium spectroscopy, as these states are interpreted as bound states of the fourth quark, the charm quark. Its existence was already predicted, as *u* and *d* form a generation of quarks, indicated by their names, but the *s* quark, which makes the *K* meson living ”strangely” long was lacking a similar partner. In addition, the GIM-mechanism (Glashow–Iliopoulos–Maiani mechanism) requires a fourth quark: To explain the unexpectedly small branching fraction of the reaction $K^0 \rightarrow \mu^+\mu^-$, Glashow Iliopoulus and Maiani introduced a hypothetical fourth quark to postulate an additional, similar decay which is destructively interfering. As the suppression of flavor-changing neutral currents, which is explained by the GIM-mechanism, was observed, the existence of a fourth quark was also supported [12]. Therefore, the *c* quark was named ”charm”, as it was evidence for the charming idea of another generation of quarks.
To describe charmonium resonances within the quark model, one could think of extending the description of quark-pairs to SU(4) the same way it was done for the *s* states. But the charm quark is so heavy, that mixing of states with charm content with the light quark states can be neglected. The $c\bar{c}$ system can be compared to positronium, which is also the reason it was named charmonium. Positronium is a bound system consisting of a positron and an electron. It behaves similar to the hydrogen atom, but with the reduced mass of the two equally light particles. They can only interact via electroweak interactions, making the Positronium an interesting probe for the electromagnetic force. Similarly, the exited charmonium spectrum allows for the study of binding forces between the two heavy quarks through the excitation spectrum of the charmonium.

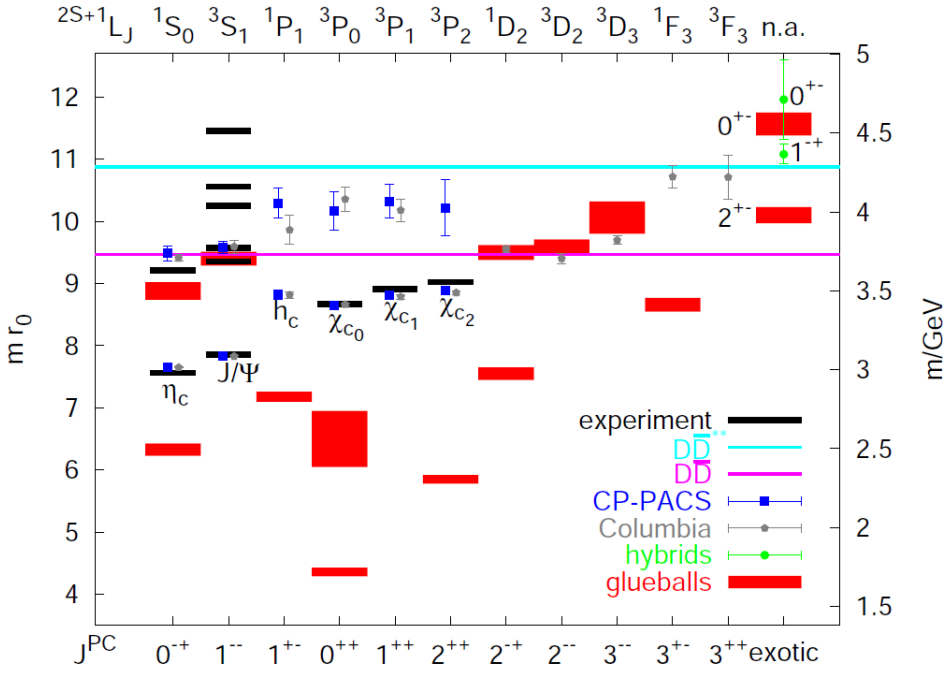

Figure 1.5: Quenched lattice QCD predictions for the spectrum of the charmonium, glueballs and the spin-exotic $\bar{c}cg$ hybrids [5].

Below Threshold

Below the open charm threshold (3.73 GeV/c^2), the energy to create $D\overline{D}$ meson pairs, which contain only one charm quark each, the states are well understood within potential models and match the predictions of e.g. lattice QCD calculations as can be seen in figure 1.5. The spectrum consists of eight narrow and well-established states (masses and widths are PDG averages [2]):

- $J/\psi(1S)$: $J^{PC} = 1^{--}$, $m = (3096.916 \pm 0.011)$ MeV/c^2, $\Gamma = (0.0929 \pm 0.0028)$ MeV
  The first particle found, the $J/\psi(1S)$, is the triplet S charmonium state. Due to its quantum number $J^{PC} = 1^{--}$, it can be directly formed at e^+e^- colliders and therefore a lot of experimental data is available from various facilities. It decays mainly ($87.7 \pm 0.5\%$) via hadronic reactions, but also radiative decays are observed as well as pure leptonic decays which are often used as an unique signature for the event selection. The reaction of interest, $J/\psi \to \gamma\pi^0\pi^0$ involving hadronic resonances, is discussed later in detail.
  It was first observed in 1974 at SLAC and at BNL.

- $\psi(2S)$: $J^{PC} = 1^{--}$, $m = 3686.109^{+0.012}_{-0.014}$ MeV/c^2, $\Gamma = (0.299 \pm 0.008)$ MeV
  The $\psi(2S)$ is the first radial excitation of the J/ψ. Due to the same reasons as mentioned for the J/ψ, it is well measured and many final states are established. Its main decay modes are naturally involving the J/ψ. Still, the branching ratio of hadronic decays to the ground state are still puzzling [13].
  It was first observed in 1974 at SLAC, as were the $J/\psi(1S)$ measurements.

- $\eta_c(1S)$: $J^{PC} = 0^{-+}$, $m = (2983.6 \pm 0.7)$ MeV/c^2, $\Gamma = (32.2 \pm 0.9)$ MeV
  The $\eta_c(1S)$ is the charmonium singlet ground state and pseudoscalar partner of the J/ψ. Due to its quantum numbers, it can not be produced directly in e^+e^- reactions. The indirect production via radiative decays of the J/ψ or ψ' suffers from small branching fractions [2]. It can be directly formed in $p\overline{p}$ reactions. It decays mainly to kaons and pions, sometimes also involving hadronic resonances, but parity conservation suppresses the decay in kaon or pion pairs. Otherwise, these pairs would be expected to be the main decay channel as they need the least energy to be formed.
  It was first observed at SLAC in 1980.

- $\eta_c(2S)$: $J^{PC} = 0^{-+}$, $m = (3639.4 \pm 1.3)$ MeV/c^2, $\Gamma = 11.3^{+3.2}_{-2.9}$ MeV
  The $\eta_c(2S)$ is the first radial excitation of the η_c. Concerning production and decay modes, the same points as for its ground state hold. In addition, it has higher mass and a smaller cross section in the indirect production at e^+e^- colliders, wherefore the experimental data is way more sparse.
  It was first observed at SLAC in 1982.

- $h_c(1P)$: $J^{PC} = 1^{+-}$, $m = (3525.38 \pm 0.11)$ MeV/c^2, $\Gamma = (0.7 \pm 0.4)$ MeV
  The singlet P state $h_c(1P)$ is lacking data as the measurement is more complicated compared to the J/ψ because of the missing unique signature. It is an interesting probe for the spin-dependence of the quark confinement as the spin-spin potential gives rise to hyperfine splitting between the triplet and singlet states only for S-wave states and not for P-wave [14]. In $p\overline{p}$ collisions, it can be directly formed by coherent annihilation of the quarks of the proton with their counterparts in the antiproton into hard gluons. In massless QCD, this process would be ruled out by helicity conservation, but this rule is strongly violated in other processes.
  First indications were found at CERN in 1986 and at Fermilab in 1992.

- $\chi_{c0}(1P) : J^{PC} = 0^{++}, m = (3414.75 \pm 0.31)\,\text{MeV}/c^2, \Gamma = (10.5 \pm 0.6)\,\text{MeV}$
  $\chi_{c1}(1P) : J^{PC} = 1^{++}, m = (3510.66 \pm 0.07)\,\text{MeV}/c^2, \Gamma = (0.84 \pm 0.04)\,\text{MeV}$
  $\chi_{c2}(1P) : J^{PC} = 2^{++}, m = (3556.20 \pm 0.09)\,\text{MeV}/c^2, \Gamma = (1.93 \pm 0.11)\,\text{MeV}$
  The triplet P state is split in mass by orbital spin and tensor interactions, forming the $\chi_{c0}(1P)$, $\chi_{c1}(1P)$, and the $\chi_{c2}(1P)$. First evidence was observed in radiative $\psi(2S)$ decays at SLAC and at DESY in 1975.

Above Threshold

Above the $D\overline{D}$ threshold only few states are established and many states, which are labeled as X, Y, and Z, require further measurements and analyses to deduce their properties (see the X labeled states in figure 1.4). The predominant decay of states above the $D\overline{D}$ threshold is into two D mesons, which makes experimental observation more difficult. D mesons are the lightest mesons containing charmed quarks. As there are several combinations of a charm quark with u, d and s quarks possible, there are several D mesons formed by these quark combinations, D^+, D^-, D_s^+, D_s^-, D^0, and $\bar{D}^0$, and their respective excitations. Depending on the initial energy various decay channels are possible for the charmonium. The thresholds of these final states are indicated in figure 1.4. As below threshold, only the spin triplet states with $J^{PC} = 1^{--}$ can be produced directly at e^+e^- facilities where most scan experiments were performed. Therefore, four of the five established states summarized below are spin triplet states and also most of the confirmed XYZ resonances are found there (figure 1.4). The PDG [2] lists the following averaged properties for the established states:

- $\psi(3770) : J^{PC} = 1^{--}, m = (3773.15 \pm 0.33)\,\text{MeV}/c^2, \Gamma = (27.2 \pm 1.0)\,\text{MeV}$
- $\chi_{c2}(2P) : J^{PC} = 2^{++}, m = (3927.2 \pm 2.6)\,\text{MeV}/c^2, \Gamma = (24 \pm 6)\,\text{MeV}$
- $\psi(4040) : J^{PC} = 1^{--}, m = (4039 \pm 1)\,\text{MeV}/c^2, \Gamma = (80 \pm 10)\,\text{MeV}$
- $\psi(4160) : J^{PC} = 1^{--}, m = (4153 \pm 3)\,\text{MeV}/c^2, \Gamma = (103 \pm 8)\,\text{MeV}$
- $\psi(4415) : J^{PC} = 1^{--}, m = (4421 \pm 4)\,\text{MeV}/c^2, \Gamma = (62 \pm 20)\,\text{MeV}$

Beside these states and the numerous X resonances indicated in figure 1.4, there are even more states seen in various experiments. Some are not confirmed yet by another experiment and none is well understood. The lack of measurements, both quantitatively to confirm states and qualitatively to determine properties, allows a great deal of latitude interpreting the sparse data and makes a solid determination of the states nature difficult. New measurements will provide additional information in order to constrain theoretical models for the description of these states. As they were not expected when considering pure quark-antiquark content, several possible explanations are described in the following.

Search for Gluonic Excitations and Exotic States

In QCD there is no restriction on the combination of quarks and gluons to hadrons as long as the principles of QCD are conserved. Indeed, already Gell-Mann expected baryons and mesons with additional Quark content in his proposal of the quark model: ”Baryons can now be constructed from quarks by using the combinations (qqq), $(qqqq\overline{q})$, etc., while mesons are made out of $(q\overline{q})$, $(qq\overline{qq})$, etc.” [15]. These and other so called exotic internal structures are considered in theoretical models as explanation for the XYZ states. The simplest realizations of these kind of states would be:

- Tetra-quark states:
  Adding a quark antiquark pair to a meson $(qq\overline{qq})$.

- Penta-quark states
  Adding a quark antiquark pair to a baryon ($qqqq\bar{q}$).
- Molecular states:
  E.g. two mesons ($q\bar{q}$)($q\bar{q}$) or a meson and a baryon ($q\bar{q}$)(qqq) forming a bound state.
- Hybrid states:
  E.g. mesons with an exited gluon ($q\bar{q}g$).
- Glueballs:
  States with only gluons as content (gg, ggg).

Figure 1.6 schematically presents how these exotic states can be imagined compared to conventional mesons and baryons. Due to a peculiarity of the relationship between the quantum numbers J, P and C, not all combinations of J^{PC} can be realized for conventional $q\bar{q}$ bound states. Therefore, the combinations $J^{PC} = 0^{--}$, 0^{+-}, 1^{-+}, 2^{+-}, etc., being inaccessible for quark antiquark pairs, are labeled as exotic quantum numbers. Figure 1.5 shows lattice QCD predictions of hadronic states including exotic states as glueballs and hybrids with conventional and exotic quantum numbers. The experimental results described before are shown in black. Most of them have the quantum numbers $J^{PC} = 1^{--}$ because historically, many states were discovered in e^+e^- collisions via an intermediate virtual photon. Many predictions are made for glueballs which can appear with any J^{PC} quantum numbers and over a significant energy range. As there are many more predicted glueball states than measured states which could possibly have glue content, it is still an open question whether the calculations need correction or the measurements are insufficient. Hybrid states with exotic quantum numbers are also shown in the plot, but again the predictions are uncertain here. This overview illustrates the open discrepancy between model predictions and measurements as well as the need for the extraction of quantum numbers of observed resonances.
The discovery of spin exotic states and thus an internal exotic structure would provide important input for the underlying theories. In addition, there could be states with conventional quantum numbers but exotic internal structure which thus also mix with $q\bar{q}$ and qqq states. To identify these kind of exotic contributions is an even bigger challenge with an higher demand on the experimental setup, the amplitude analyses procedure and the used amplitude models. In order to be sure the observed resonances are e.g. hybrids, it is helpful when the whole pattern of hybrid candidates proposed by a certain theoretical model is provided.

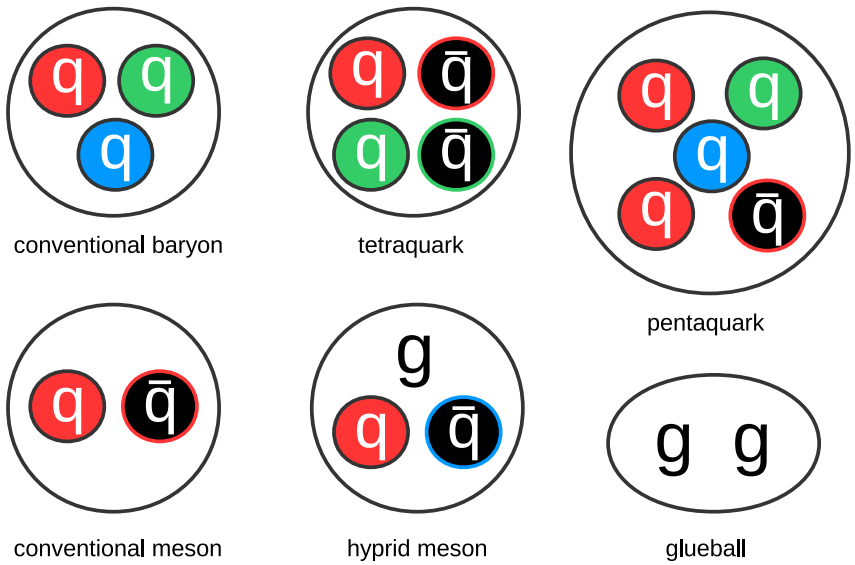

Figure 1.6: Sketch of some possible bound state configurations in the quark model.

1.2 Light Mesons

The quark model prediction of the spectrum of the light mesons was shown in figure M.1. This scheme offers a nice description of the light quark hadrons, but measurements confirm that the mass eigenstates differ significantly from this pure SU(3) description. The mass eigenstates being observed in nature might be mixtures of the flavor eigenstates of SU(3). For example, the ϕ meson can not be considered the $s\bar{s}$ eigenstate by SU(3), as a small contribution of $u\bar{u}$ and $d\bar{d}$ is mixed into its content. Therefore, the physical states are mixtures of the SU(3) wave functions ω_8 and ω_1 with the nonet mixing angle Θ [2]:

$$\phi = \omega_8 cos\Theta - \omega_0 sin\Theta \tag{1.1}$$

$$\omega = \omega_8 sin\Theta + \omega_0 cos\Theta \tag{1.2}$$

$$\omega_8 = \frac{1}{\sqrt{6}}(u\bar{u} + d\bar{d} - 2s\bar{s}) \tag{1.3}$$

$$\omega_0 = \frac{1}{\sqrt{3}}(u\bar{u} + d\bar{d} + s\bar{s}) \tag{1.4}$$

Table 1.1 shows a summary of observed light mesons and their quark model assignments as well as mixing angles.

Table 1.1: Suggested $q\bar{q}$ quark-model assignments for observed light meson resonances. K_{1A} and K_{1B} are nearly equal 45° mixtures of the $K_1(1270)$ and $K_1(1400)$ resonances. The full table can be found in [2], table 15.2.

$n^{2S+1}L_J$	J^{PC}	$I = 1$ $u\bar{d}, \bar{u}d,$ $\frac{1}{\sqrt{2}}(d\bar{d} - u\bar{u})$	$I = 1/2$ $u\bar{s}, d\bar{s},$ $d\bar{s}, -\bar{u}s$	$I = 0$	
1^1S_0	0^{-+}	π	K	η	$\eta'(958)$
1^3S_1	1^{--}	$\rho(770)$	$K^*(892)$	$\phi(1020)$	$\omega(782)$
1^1P_1	1^{+-}	$b_1(1235)$	K_{1B}	$h_1(1380)$	$h_1(1170)$
1^3P_0	0^{++}	$a_0(1450)$	$K_0^*(1430)$	$f_0(1710)$	$f_0(1370)$
1^3P_1	1^{++}	$a_1(1260)$	K_{1A}	$f_1(1420)$	$f_1(1285)$
1^3P_2	2^{++}	$a_2(1320)$	$K_2^*(1430)$	$f_2'(1525)$	$f_2(1270)$

An interesting case of mixing happens in the case of the neutral kaons, $K^0(s\bar{d})$ and $\overline{K}^0(\bar{s}d)$. As they do not consist of a quark and its own antiquark, they cannot be the antiparticle to each other. They can mix by the weak interaction, which does not conserve flavor. Assuming CP symmetry, the CP eigenstates are:

$$|K_1\rangle = \frac{1}{\sqrt{2}}(|K^0\rangle - |\overline{K}^0\rangle) \tag{1.5}$$

$$|K_2\rangle = \frac{1}{\sqrt{2}}(|K^0\rangle + |\overline{K}^0\rangle) \tag{1.6}$$

with $|K_1\rangle$ decaying to two pions but $|K_2\rangle$ not (or vice versa depending on the phase convention). This makes the $|K_2\rangle$ decay take significantly longer, which leads to the assignment of the eigenstates as K-short $K_S = |K_1\rangle$ and K-long $K_L = |K_2\rangle$. Obviously, this is a nice probe for observing CP violation.
Figure 1.7 shows a level scheme of observed light mesons considered to have mainly u and d quark pair content. The mesons are ordered by their total angular momentum on the ab-

scissa. A color and symbol code is used to indicate mesons with same isospin I and parity P. The smallest spin of $J = 0$ features also the mesons with the smallest mass square, which is shown on the ordinate. The minimal mass for mesons rises with the total angular momentum J, leaving the lower right part of the plot empty. A separation in masses is visible and indicated by red bands for all spins, which emphasizes an interpretation of the higher mass mesons as the radial excitation of the lowest state. But the absence or overpopulation of mesons with certain quantum numbers within the clusters mark the limits of this level scheme and interpretation. Also, the quark model suggests multiplets of states with similar quantum numbers which does not contradict observing excitations, but complicates the interpretation of the observed resonances. E.g. in the case of $J = 0$ of the plot in figure 1.7, the f_0 resonances seem to follow rather nicely the indicated level scheme. But from the quark model, one would expect two ground states from the scalar nonet, which both could have excitations, and it is unclear which resonances can be assigned to the nonet states. This shows that the light meson spectrum is not yet fully understood and shows degeneration which lacks validated theoretical description.

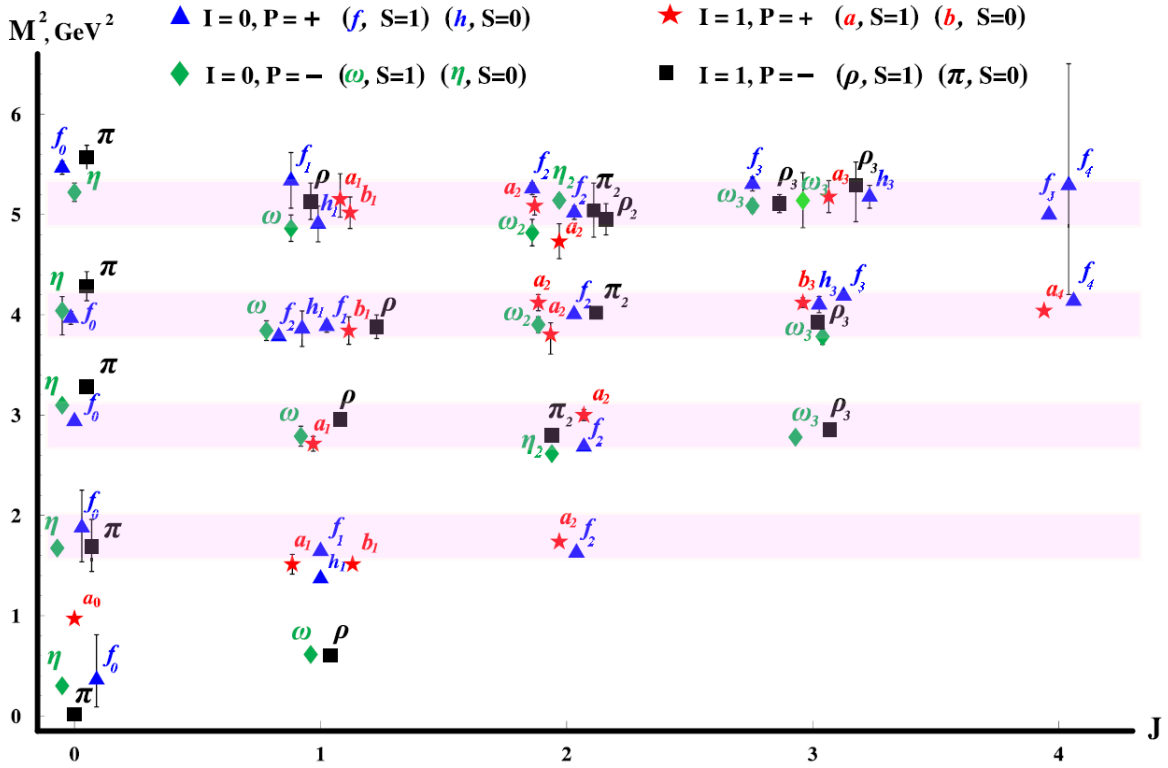

Figure 1.7: Overview of measured light mesons with presumably u and d quark content [16]. To better distinguish between mesons with similar masses and identical spin, they are shifted slightly on the abscissa.

In addition to the unclear interpretation of the various observed mesons, lattice QCD calculations predict the existence of glueballs, hybrids, and other particles with exotic internal structure at the energies of observed light mesons. An overview of glueball predictions was already shown in figure 1.5. As an example, the lightest $J^{PC} = 0^{++}$ glueball has an energy range close to the observed $f_0(1500)$ resonance with the same quantum numbers. The interpretation of the $f_0(1500)$ is still an open topic.

Light Isoscalar Meson Spectrum

A particular interesting playground when looking for excitations and exotics with conventional quantum numbers is the spectrum of isoscalar mesons. Particles are called isoscalar when they underlie the scalar transformation under the SU(2) group of isospin, thus they are singlet states with total isospin $I = 0$. As e.g. glueballs are necessarily isoscalar, the structure and spectrum of isoscalar mesons is highly interesting.
Experiments measuring isoscalar states have found more states then predicted by quark models assuming two quarks forming the mesons. In figure 1.8, the gray areas of the spectrum are confidence intervals of the measured states while the black bars are the quark model predictions. Below 1500 MeV/c^2, four overlapping f-resonances were measured were only two states are expected. Table 1.2 shows a list of observed isoscalar resonances according to the PDG [2].

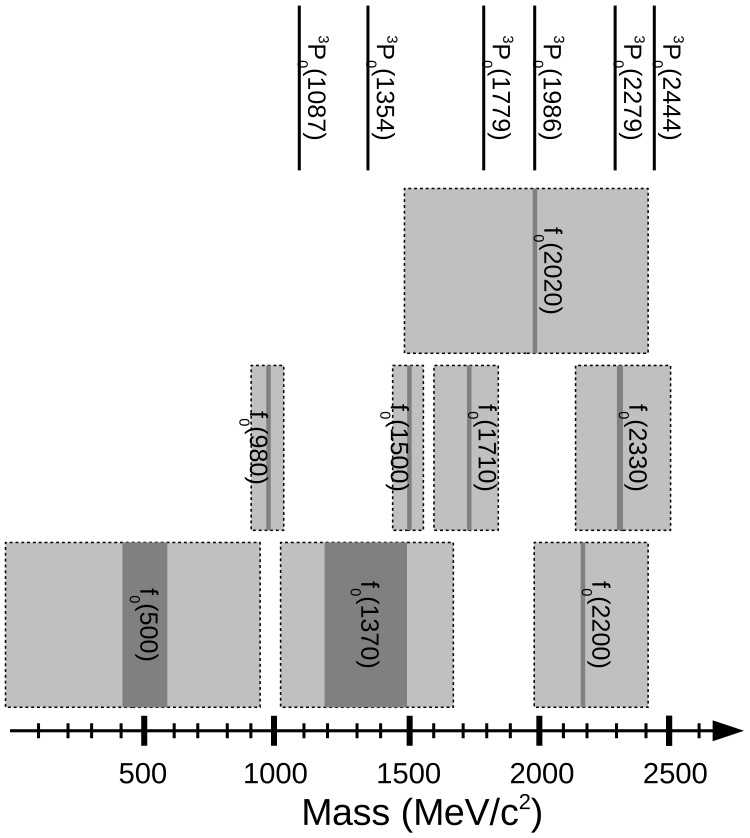

Figure 1.8: The spectrum of isoscalar mesons with $J^{PC} = 0^{++}$. The black bars represent quark model predictions [17], while the gray shaded areas and bars show the experimental data [2], see table 1.2, with the masses in dark and widths in light gray.

Table 1.2: Observed isoscalar resonances and PDG averages [2]. For the $f_0(2330)$, no averages or fits were performed by the PDG.

Name	Mass [MeV/c^2]	Width [MeV]	T-Matrix Pole [MeV]	Dominant Decay
$f_0(500)$	$400-550$	$400-700$	$(400-550)-\mathrm{i}(200-350)$	$\pi\pi$
$f_0(980)$	990 ± 20	$40-100$		$\pi\pi$
$f_0(1370)$	$1200-1500$	$200-500$	$(1200-1500)-\mathrm{i}(150-250)$	$\pi\pi, 4\pi$
$f_0(1500)$	1505 ± 6	109 ± 7		$\pi\pi, 4\pi$
$f_0(1710)$	1722^{+6}_{-5}	135 ± 7		$K\overline{K}$
$f_0(2020)$	1992 ± 16	442 ± 60		
$f_0(2200)$	2189 ± 13	238 ± 50		
$f_0(2330)$	$\approx 2310-2340$	$\approx 140-230$		

The isoscalar f_0 resonances are difficult to extract as they are rather broad, overlapping, and the Breit-Wigner description is sometimes insufficient to describe the consequential interference. E.g. in the case of the $f_0(980)$, the coupling to $K\overline{K}$ plays also a role and a cross-channel analysis is necessary. Especially the $\pi\pi$ S-wave is difficult to model due to the richly populated spectrum.
There are a lot of interpretations of the observed isoscalar resonances. The $f_0(980)$ seems to have a large $s\bar{s}$ contribution as it is observed in the decay $J/\psi \to \phi\pi^+\pi^-$, but has a very small cross-section in $J/\psi \to \omega\pi^+\pi^-$. Thus, the $f_0(980)$ is often interpreted as a multiquark or $K\overline{K}$ bound state. In distinction to the other heavier resonances, the $f_0(1710)$ tends to decay more likely to $K\overline{K}$, so it is assumed to be a $s\bar{s}$ state. The $f_0(1500)$ has a surprisingly small width and a very small $K\overline{K}$ branching fraction, so $s\bar{s}$ contribution seems unlikely. Its upper limit from $\pi^+\pi^-$ decays however suggests a small contribution from the lightest quarks. Glue contribution could explain this contradiction [2].
Although there are many possible explanations for the situation in the isoscalar meson system, the question arises how to improve the measurements to allow better testing of the different theoretical explanations. If measurements are precise enough, additional properties of resonances might be used to understand their internal structure or to sort them into groups which hopefully simplifies assigning them to predictions. One of the properties which need better measurements and analysis is the dynamical function, the shape of the resonant behavior of a state. As example for the variation of the resonance shape in models, figure 1.9 shows several calculations for the shape of the $f_0(980)$ in $\gamma\gamma \to \pi^0\pi^0$ reactions, the definition of each can be found in [18].
Due to the significant overlap of these broad resonances, interference between them needs to be correctly accounted for. Therefore, model dependency can be a limiting factor in the measurement in this regime, especially if it is unclear, how many resonances contribute to the interference.
With this in mind, a model-independent method to extract the dynamical function separated by spins is presented in this work. For the development of the wave extraction method of the scalar wave the reaction channel $J/\psi \to \gamma\pi^0\pi^0$ was chosen. Here, resonances decaying to $\pi^0\pi^0$ are only possible with the quantum numbers $J^{PC} = 0^{++}, 2^{++}, ...$ The usage of the model-independent ansatz is only possible because of the clean $\gamma\pi^0$ final state, where the well understood ω-resonance is the sole expected contribution.

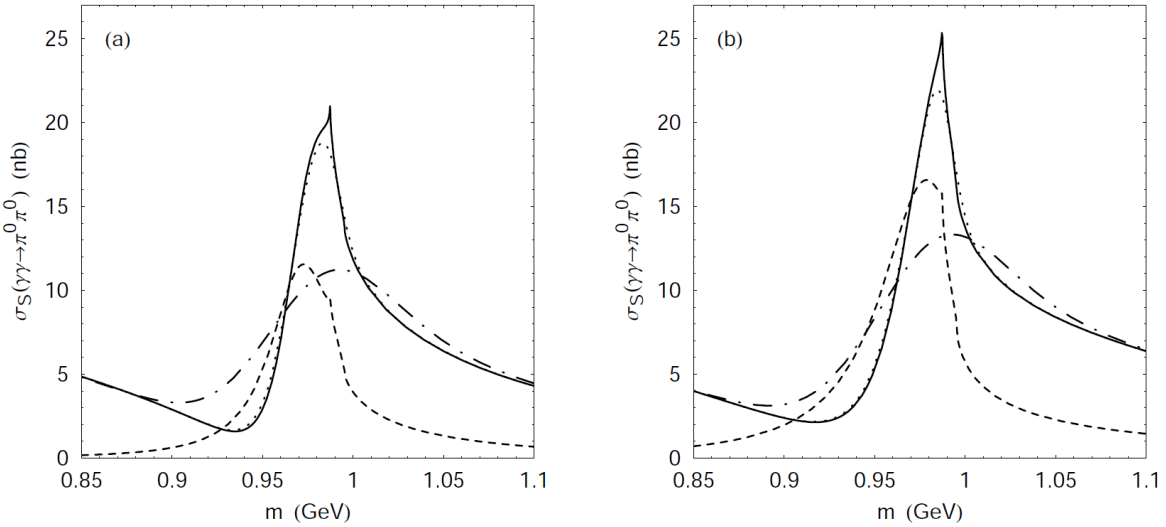

Figure 1.9: (a) The solid curve shows the cross section $\sigma_S(\gamma\gamma \to \pi^0\pi^0)$ calculated with a certain set of parameters. The dotted and dot-dashed curves show the same cross section but smoothed with a Gaussian mass distribution accounting for a resolution of 5 and 30 MeV/c^2 respectively. The dashed curve shows the contribution caused by the $f_0(980)$ resonance production via the K^+K^- loop mechanism only. (b) The same as in (a) but for another set of model parameters [18].

2 Experimental Setups

Clean and precise experimental data are crucial for amplitude analyses. ComPWA was designed for existing and future experiments which will offer new levels of data quantity and quality. The two most promising experiments are presented in the following: the future PANDA experimental setup and the already operating BESIII experimental setup. At the moment, BESIII performs some of the most interesting measurements for hadron spectroscopy and it is therefore the reasonable choice for the first analyses performed with ComPWA. In addition, performing cross-checks for recent sophisticated analyses of BESIII data will help to verify new analysis methods. PANDA, however, will use an antiproton beam impinging on a proton target which offers unique possibilities for physics questions. The increased resolution of the beam energy and the reconstructed particles gives access to finer structures in the data which requires higher complexity in the models and the fit procedure, thus raising the software demands as well.

2.1 The PANDA Experiment

The PANDA Experiment [5] is going to be one of the future experiments aiming at measurements needed to increase our understanding of the strong interaction. It is going to be one of the major experiments at FAIR (Facility for Antiproton and Ion Research) [19].

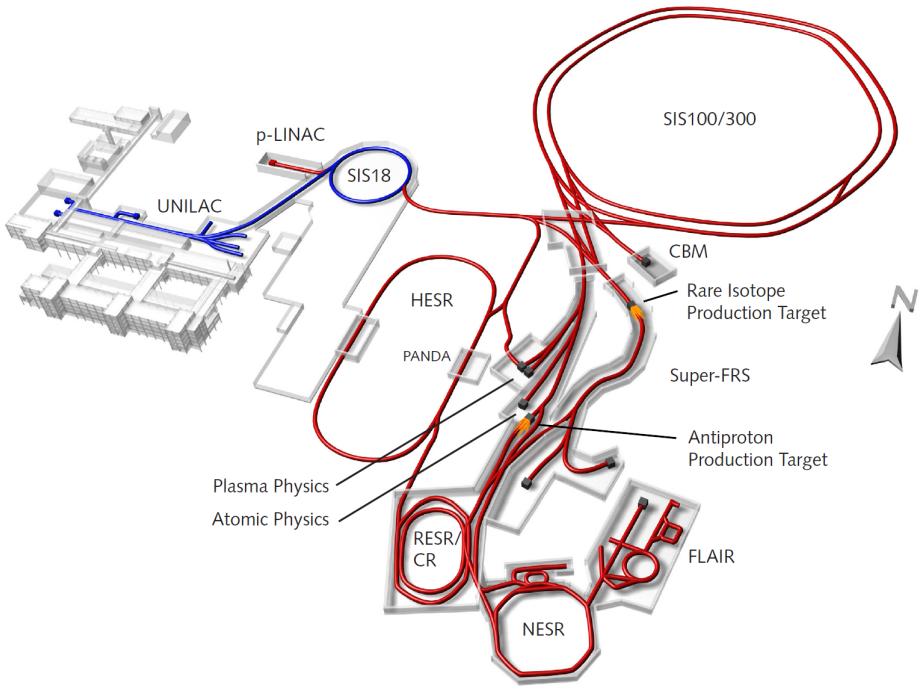

Figure 2.1: Overview of the FAIR accelerators and experiments [19].

Figure 2.1 shows an overview of the existing facilities at the GSI Helmholtzzentrum für Schwerionenforschung GmbH in Darmstadt, Germany, and the planned FAIR accelerators and experiments. The existing accelerators are shown in blue while the future accelerators are shown in red. After pre-acceleration with the present UNILAC or with the new p-LINAC for antiproton production and SIS18, the new, 1100 meter long SIS100 heavy ion synchrotron will provide high energy protons and ions for various experiments and additional accelerators. For the production of the antiprotons a high intense proton beam will be guided to a production target (e.g. nickel). Then the antiprotons are collected and precooled in the Collector Ring (CR) before transferred to the HESR (High Energy Storage Ring, see figure 2.2). In a later stage, also the RESR (Recycled Experimental Storage Ring) will accumulate the precooled antiprotons in order to allow a ten times higher luminosity. At HESR, they are stored and further accelerated or deaccelerated for PANDA. The antiprotons can also be transferred directly from the CR to the FLAIR experiments.

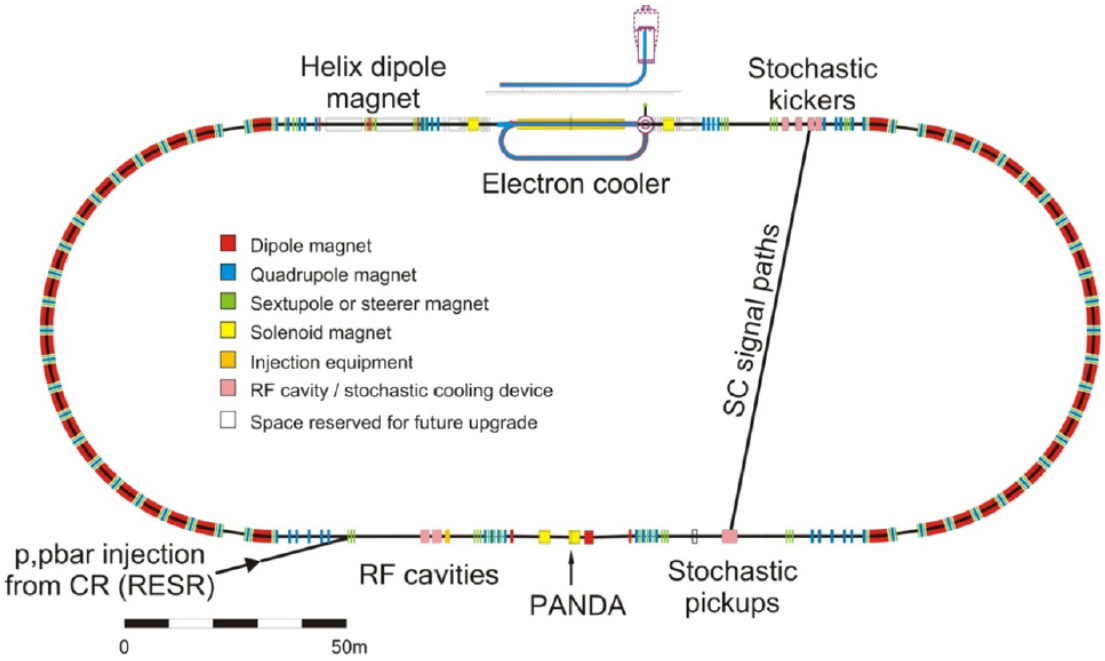

Figure 2.2: Overview of the HESR accelerator [20].

Located at the HESR (High Energy Storage Ring) of FAIR, PANDA will be provided with an antiproton beam of 1.5 GeV/c to 15 GeV/c momentum. The accelerator is designed for two modes: In the high resolution mode, the momentum spread is reduced down to $\Delta p/p < 4 \cdot 10^{-5}$ by stochastic and electron cooling for $p_{\mathrm{beam}} < 8.9\,\mathrm{GeV}/c$, in the high luminosity mode, the mean design interaction rate is $2 \cdot 10^7$ annihilations per second with a momentum spread of $\Delta p/p \approx 10^{-4}$ [5]. Table 2.1 shows the parameters of both modes.

Table 2.1: Antiproton beam parameters for PANDA according to [20] assuming a target density of $4 \cdot 10^{15}$ *atoms*/cm^2.

	HL mode	HR mode
p [GeV/c]	$1.5 - 15$	$1.5 - 8.9$
$\frac{\Delta p}{p}$ (RMS)	$\approx 10^{-4}$	$< 4 \cdot 10^{-5}$
number of antiprotons	10^{11}	10^{10}
peak luminosity [/cm^2/s]	$2 \cdot 10^{32}$	$2 \cdot 10^{31}$

PANDA will be the only experiment using antiproton-proton reactions at FAIR. The main reasons why this initial state was chosen are that it offers a gluon-rich environment with less restrictions on its quantum numbers by formation and that the expected formation cross sections are large. In comparison, positron-electron experiments have an initial state in first order limited to the quantum number $J^{PC} = 1^{--}$ of the virtual photon created by the e^+e^- annihilation in almost all reactions. States with different quantum numbers can only be produced in subsequent decays with smaller cross sections. Compared to proton-proton scattering, the antiproton-proton annihilation process offers effectively more energy for particle production. In quark-antiquark annihilation as in $p\bar{p}$, each quark or antiquark carries about 12% of the corresponding momentum of the initial state particle as both are valence quarks [21]. In pp reactions, the antiquark is a sea quark and therefore carries only about 4% of the momentum of the proton. Therefore, the $q\bar{q}$ annihilation has higher energy in $p\bar{p}$ as in pp reactions given the same momenta of protons and antiprotons.
Another advantage of antiproton proton scattering in comparison to positron electron scattering lies in the better beam adjustment. This leads to a better momentum resolution $\Delta p/p$ ratio which allows for resonance scans of directly produced resonances by varying the beam momentum.

PANDA Instrumentation

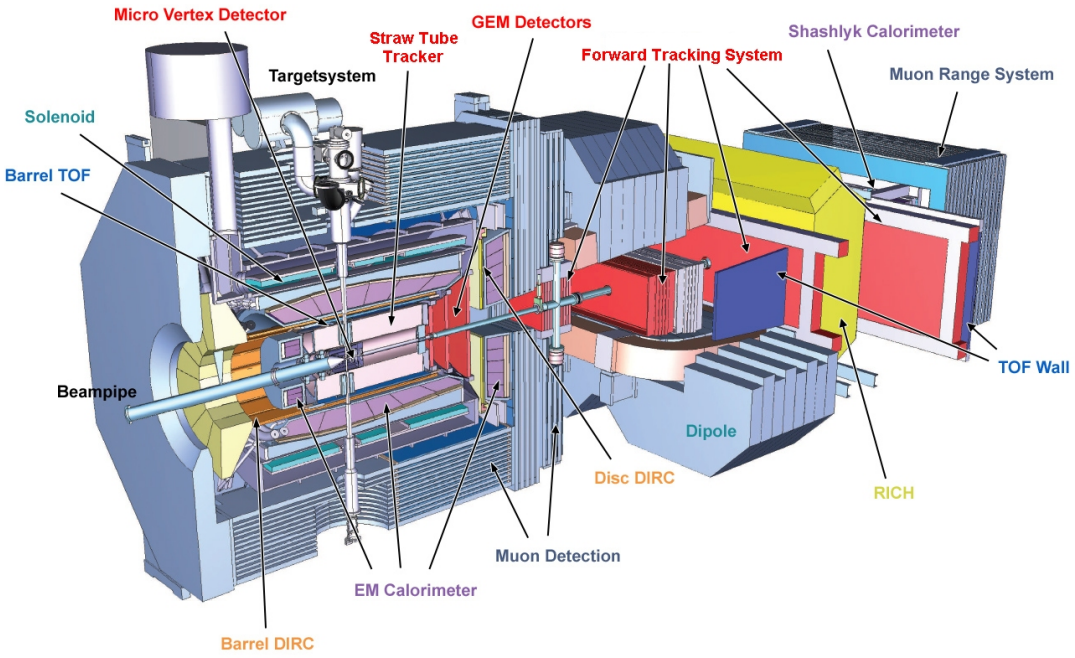

Figure 2.3: Overview of the detector components of the PANDA experimental setup [22].

The physics program of the PANDA experimental setup makes high demands on the detector components. A universal, modularized detector setup is being developed which provides optimal solid angle coverage, tolerates high influx rates, measures a broad spectrum of particle momenta and provides efficient particle identification. As a fixed target experiment, this is realized using a target spectrometer (TS) and a forward spectrometer (FS) setup (figure 2.3).
In the target spectrometer, charged tracks are reconstructed with the cooperation of several components: The innermost detector, the Micro-Vertex-Detector (MVD) consists of several layers of semiconductor detectors which provide positions and energy loss information. The Straw Tube Tracker (STT), representing the main tracking detector in the TS, is placed around the MVD for additional tracking and energy loss information. For small angle scattering, a GEM-Detector in the forward direction performs track reconstruction. The target spectrometer

is placed inside a 2 T solenoid magnet which bends the tracks of charged particles depending on their momentum. This allows for a momentum reconstruction with precision up to a few percent [5]. For the research of hypernuclei, the MVD can be replaced by a complex system of primary and secondary targets and a germanium based detector.
To identify charged particles with momenta above 1 GeV/c, Cherenkov detectors are used in addition to the energy loss information from the tracking detectors. Particles with momenta below 1 GeV/c are identified by a Time of Flight detector (ToF). Two DIRC (Detection of Internal Reflected Cherenkov Light) detectors are being developed, the Barrel-DIRC and the Disc-DIRC, for particles scattered in forward direction. Photons are measured with three electromagnetic calorimeters (EMC), which are placed in the forward end cap, backward end cap, and as a barrel around the tracking and PID detectors. The solonoid magnet is instrumented to act as a muon detector.
The forward spectrometer is placed inside and behind a dipole magnet and operates basically in the same way. It consists of another STT, a RICH detector (Ring Imaging Cherenkov), a scintillator wall for Time-of-Flight measurements, a shashlyk EMC and muon detectors. Behind the forward spectrometer, there will be a tracking station inside the beam pipe for luminosity measurements.
The EMCs of PANDA are tested to have a relative energy resolution which will be below 2.5% at 1 GeV photon energy to 13% for 20 MeV photon energy. Simulation studies of the Micro-Vertex-Detector group using a complete PANDA target spectrometer setup of the benchmark channel $\bar{p}p \to \psi(2S) \to J/\psi\pi^+\pi^- \to \mu^+\mu^-\pi^+\pi^-$ yield a vertex resolution of $\sigma_{pz} \approx 65\,\mu$m in beam direction and $\sigma_{px/py} \approx 45\,\mu$m in x and y direction [23]. The mass resolution of the selected J/ψ and $\psi(2S)$ candidates in the same analysis was below 2% for both.

Search for Exotic States with PANDA

The energy range of the antiproton beam at PANDA allows the search for exotic states, e.g. hybrids, glueballs and multiquark states in the charm and the light meson region. At low masses ($< 2.5\,\text{GeV}/c^2$), there are a lot of conventional $q\bar{q}$ mesons present. Although experimentally easier accessible, this complicates the identification of exotic contributions and exotic states due to overlapping and mixing of the broad states.
In the charm region, the heavy quark states offer a better opportunity to understand the QCD spectrum, as it has a lower resonance density and less interference is present as states are rather narrow. In addition, all conventional quantum numbers can be produced directly in $p\bar{p}$ reactions, which makes PANDA unique for the search for supernumerous states and the measurement of their properties as e.g. line shapes very precisely. In order to identify exotic internal structure, states with exotic quantum numbers can be produced with an additional recoil particle.
PANDA will provide measurements tackling several interesting topics [5]:

- Hybrids
  In the light quark region, there are more then ten candidates whose nature is yet to be determined, e.g. the $f_0(1500)$, which means also conventional or exotic contributions might be possible. In the charm region, there are eight charmed hybrids ($c\bar{c}$ states with gluonic degrees of freedom) with masses below $5\,\text{GeV}/c^2$ expected. The lightest one at $4.15\,\text{GeV}/c^2$ has the conventional quantum numbers $J^{PC} = 0^{-+}$. The lightest hybrid with exotic quantum numbers is named $\widetilde{\eta}_{c1}$ and expected at $4.3\,\text{GeV}/c^2$ with $J^{PC} = 1^{-+}$. To confirm the theoretical predictions, it would be ideal to find patterns of several charmed hybrids.

- Glueballs
  Glueballs with conventional quantum numbers are hard to identify due to possible mixing with normal mesons. In contrast, glueballs with exotic quantum numbers cannot mix with conventional mesons, so they are expected to be rather narrow and easy to identify. Heavy glueballs are predicted at 4.14 GeV/c^2 ($J^{PC} = 0^{+-}$) and 4.74 GeV/c^2 ($J^{PC} = 2^{+-}$) which are accessible at PANDA energies. At lower energies, in the light quark sector, the same is true as for the hybrid candidates: the large number of broad light quark states makes the identification of all kind of glueballs very challenging.

- Multiquark-States
  In the light quark spectrum, the most promising multiquark candidates are the $a_0(980)$ and the $f_0(980)$ resonances, where overlapping, mixing and threshold effects complicate a certain interpretation. Again, the situation looks more promising in the charm/charmonium sector due to less mixing and overlapping where the first candidate of a multiquark-state was discovered by the BESIII experiment in 2013 ($Z_c(3900)$) [24]. For the search for multiquark states, PANDA will be the ideal experiment since it can provide a larger amount of data.

2.2 The BESIII Experiment

The BESIII experiment [7] is performed at the Beijing Electron-Positron Collider (BEPC) II [25] in Beijing, China, which is running since 2008. It has already produced the world's largest sample of more than 1.3 billion J/ψ events. Among other physics topics, J/ψ decays offer a hadron-rich environment for spectroscopy as hadronic decays are possible with small background. The experiment has a design luminosity of $1 \cdot 10^{33}\,/\mathrm{cm}^2/\mathrm{s}$ (table 2.2) at center of mass energy $\sqrt{s}$ = 3.78 GeV while the accelerator is able to operate at an energy range of 2 to 4.6 GeV. Figure 2.4 shows a sketch of the BEPCII collider layout. It is built of two almost 240 m long rings for accelerating positrons and electrons with a crossing angle between the two beams of 11 mrad at the interaction point.

Table 2.2: Parameter of the BEPCII collider [25].

Beam Energy	1.89 GeV
Circumference	237.53 m
Bunch Number	93
Beam Currents per Ring	1116 mA
Design Luminosity	$1 \cdot 10^{33}\,/\mathrm{cm}^2/\mathrm{s}$

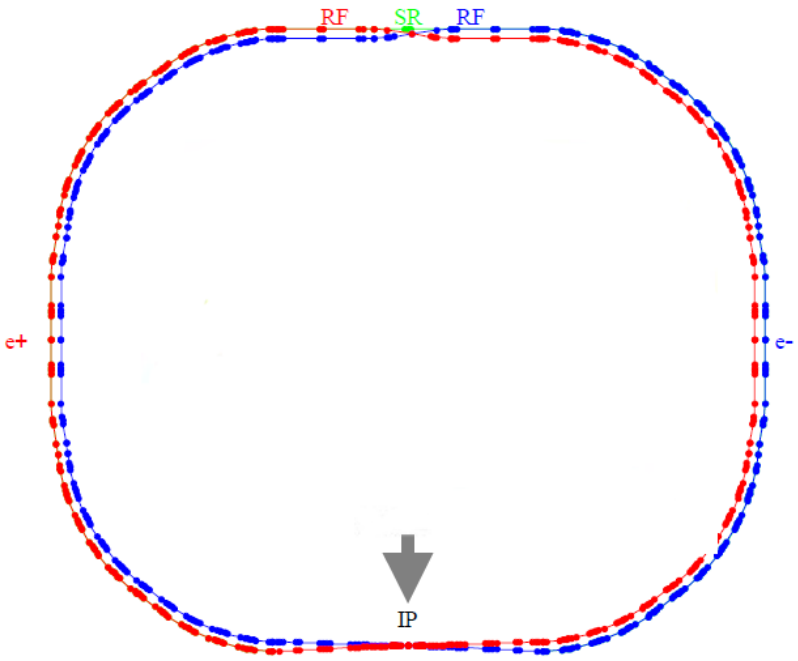

Figure 2.4: Layout of the double ring of BEPCII [25].

BESIII Instrumentation

To measure the final states of the proposed physics topics with good efficiency, a versatile detector setup was built. Figure 2.5 shows a schematic profile of the detector. An instrumentalized 1 T solonoid magnet is used to bend the trajectories of charged particle and thus allows momentum measurements and particle identification. Inside the magnetic field, four components

are installed: a multilayer drift chamber (MDC) for tracking and momentum determination and for π/K separation up to p = 770 MeV/c, a Time of Flight (ToF) detector for particle identification with time resolution of about 100 ps, an electromagnetic calorimeter (EMC) measuring energies of electromagnetic showers above 20 MeV and a muon chamber system (MUC) for identifying muons. The momentum resolution of the MDC is expected to be better than $\Delta p/p$ = 0.5% at 1 GeV/c, the EMC provides an energy resolution of $\Delta E/E$ = 2.5% to 5% at 1 GeV. The MUC has a reconstruction efficiency of more than 90% for muons with momenta at and above 0.5 GeV/c.

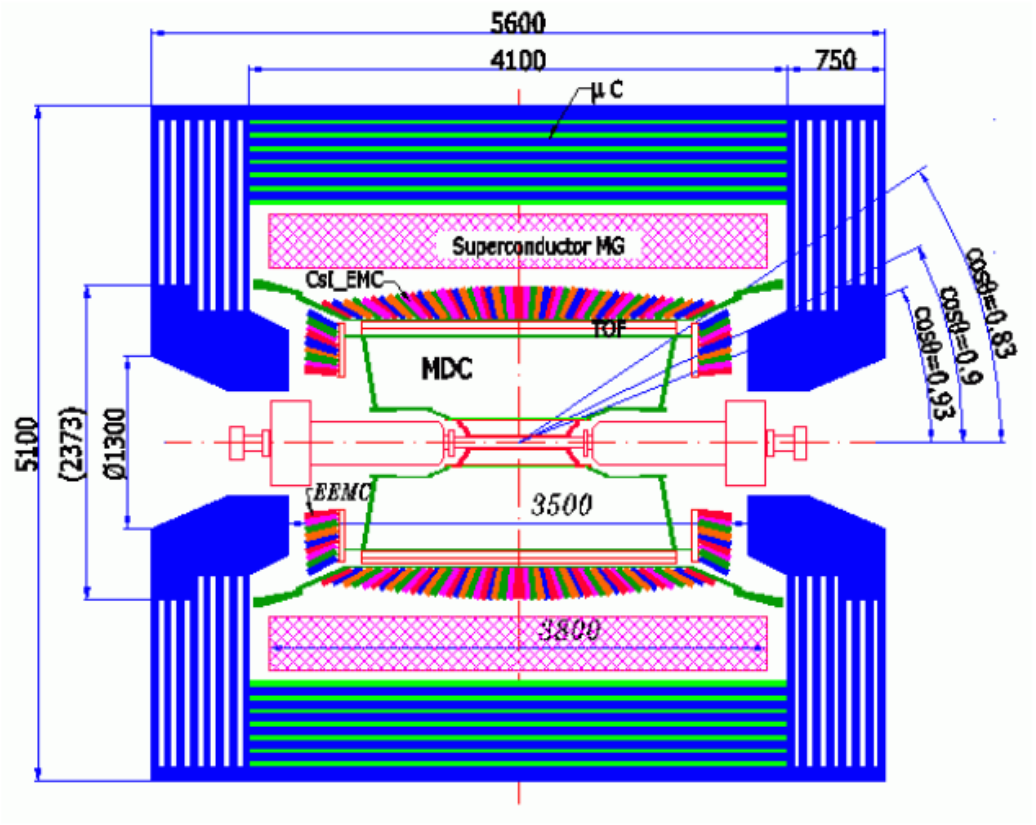

Figure 2.5: Layout of the BESIII detector [7].

Search for Exotic States at BESIII

The BESIII collaboration made some unexpected discoveries recently. In spring 2013, they published the observation of a charged resonance called the $Z_c(3900)^+$ subsequently decaying into a charged pion and and a J/ψ [24]. The resonance was confirmed by the Belle experiment [26]. As the J/ψ consists of a $c\bar{c}$ quark pair, the Z_c very likely has charmonium content. But to explain its charge, additional quarks are needed, leading to a minimum quark content of $c\bar{c}q\bar{q}'$. Later, a partner of the $Z_c(3900)^+$ was found, the $Z_c(4020)^+$ [27]. It decays into a charged pion and a h_c, which is a charmonium state with $J^{PC} = 1^{+-}$. Therefore, the same interpretation concerning the minimal quark content as for the $Z_c(3900)^+$ holds true. Furthermore, the neutral partners to the Z_c states were observed as well, e.g. the $Z_c(4020)^0$ in $e^+e^- \to \pi^0\pi^0 h_c$ reactions at BESIII [28] or the $Z_c(3900)^0$ in $e^+e^- \to \pi^0\pi^0 J/\psi$ in an analysis of CLEOc data [29] and also very recently by BESIII, which suggests the presence of two isospin triplets.
BESIII also observed charmonium-like structures in $e^+e^- \to D^*\bar{D}^{*\pm}\pi^{\mp}$ as well [30], although its nature still needs to be determined. This supports the interpretation of the Z_c's as bound states of two charmed D mesons each, as the masses of a $D\bar{D}^*$ pair and a $D^*\bar{D}^*$ pair are

slightly below the masses of the current Z_c mass measurements. The search continues in other channels involving η_c and χ_{c1} mesons. Belle found the $Z_1(4050)^+$ and the $Z_2(4250)^+$ states decaying to $\chi_{c1}\pi$ [31], though this is not confirmed by BaBar [32].
Another goal of BESIII is the search for glueballs in the light isoscalar meson spectrum (I^GJ^{PC} = 0^+0^{++}). One reaction of interest is $J/\psi \to \pi^0\pi^0\gamma$ as only the quantum numbers $J^{PC} = 0^{++}$, 2^{++}, 4^{++}, etc are accessible in radiative J/ψ decays in this channel [8]. The absence of a large background contribution promises clean data of the resonances with possible glue content expected in $\pi^0\pi^0$, e.g. the $f_2(1270)$, $f_0(1500)$, and $f_0(1710)$.

2.3 Common Objectives

Due to the different initial states, both experiments have different access to the resonances of interest. Nevertheless, the resonances observed and the measured parameters should by consistent. However, the direct comparison of results from different experiments can be difficult when elaborate analysis methods such as amplitude analyses are used. For example, the use of different models leads to different interpretation of the model parameters and thus, they cannot be compared. But in addition also technical aspects must be considered when comparing results, like different implementations of the same models in the respective analysis software, different optimization criteria, or different estimation methods. The comparison and confirmation of hadron resonance observations and measurements among experiments is very complicated and sometimes nearly impossible.
A first step to improve the situation would be to use common models and analysis techniques in order to compare the same parameters and physics interpretation. In contrary to a single, inflexible software tool, a framework could provide common features as far as possible while still allowing special implementations if necessary. Using a modular design and well defined interfaces makes introducing experiment-specific code as easy as possible while at the same time it will help to document the differences in analyses. The common PWA Framework (ComPWA) is the approach to provide such a framework.
In addition, a framework could further increase the quality assurance and acceptance of hadron spectroscopy results by providing the possibility of combined fits. Up to now, it has been nearly impossible to perform combined fits of data from different experiments as the understanding of the detectors and the analysis techniques are strongly interwoven with the software tools used by the various experiments. If this information is provided by modules within the same framework, it will be much easier to set up a model describing both datasets at once and extract resonance parameters directly by analyzing them simultaneously. This could be a huge improvement over comparing lists of results from different experiments after each independent fit was performed.

3 ComPWA Framework

3.1 Concept

Beside the difficulties in comparing the results from various experiments, as described in chapter 2.3, there are several additional requirements for a new amplitude analysis framework not being met by existing tools and cannot easily be reached by extending them. One is that more precise data of higher statistics might need more complex models to describe the data as more details and also small contributions of certain resonances become visible and need to be treated properly. Therefore, the framework should be extendable by different models and formalisms and the interfacing should be as easy as possible. There might also be the need to use different models in the same analysis, e.g. to describe complex final states or to directly compare models. Another requirement is induced by the large datasets of new experiments striving for higher statistics. To analyze these, the software must be able to efficiently handle large datasets which implies caching, distributing, and parallelization. Also, detailed analyses might include data from multiple channels or even data from various experiments. A simultaneous fit with the same is preferrable in that respect. In addition, complex decay chains or initial states lead to intricate amplitudes. This leads to models with many parameters which need to be fitted to the data. New optimization algorithms will be necessary to deal with high dimensional parameter spaces so the framework should be open for existing and future optimization libraries. Another point to stress here concerns the quality assurance for this kind of analyses as the documentation serving the purpose of reproducibility needs a high level of detail. The functionality to provide comparable and reproducible documentation of different analyses is therefore an important goal of this project.
Hadron spectroscopy analysis offers large variability when it comes to fitting a model to the experimental data:

- How to prepare the data: cuts, background suppression, etc.?
- Which model(s) describes the data best, can a model be adapted for this problem?
- What set of resonances is needed, which are fixed, variable or even optional?
- What is a good measure of how well the model describes the data and is binning needed?
- What optimization algorithm is suitable and efficient for the chosen model(s)?
- What is the required precision?

When trying to compare results obtained in different analyses, all these points need to be considered and reasons for inconsistency are hard to track down. Having a common framework would reduce these complications as most of the analyses are done with the exact same code except when performing cross-checks. In addition, a common bookkeeping makes it easier to identify necessary differences between analyses.
As shown in the following, corresponding software packages currently available are not able to be extended to meet these demands, therefore the new common PWA framework ComPWA was especially designed in order to overcome these shortcomings.

3.2 General Layout

In order to prepare the discussion of existing tools and the requirements of ComPWA, a short overview of the layout of amplitude analysis software is given in the following. In general, an amplitude analysis is an extraction of physically relevant parameters by fitting a suitable amplitude model defined by a certain parameter set to experimental data. Figure 3.1 shows a sketch of the basic, mandatory components needed for any amplitude analysis.

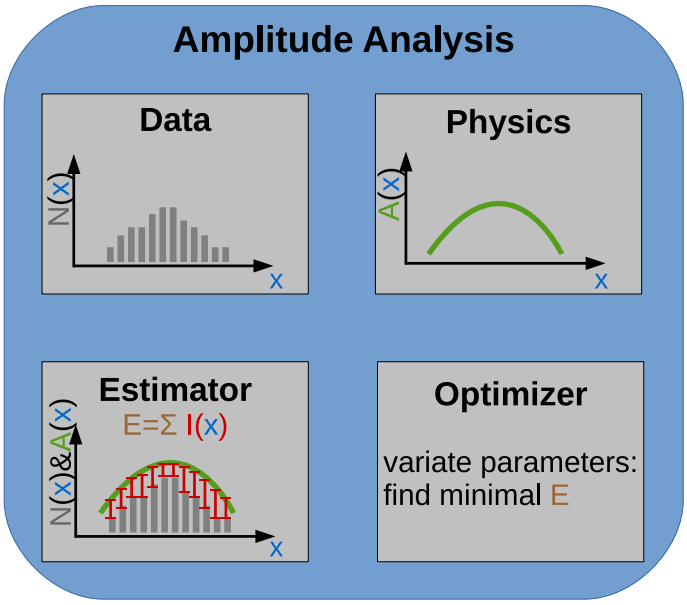

Figure 3.1: Sketch of the basic components of an amplitude analysis software. $I(x)$ describes the deviation between the amplitude model and the data at the position of one event.

An amplitude analysis software needs to provide access to and control of the experimental data, the amplitude model, and the fitting procedure. The **Data** handling often depends on the experiment and usually consists of event based information (e.g. four momentum vectors) and global information (e.g. beam energy). It might come either in a unbinned or binnned representation or could be transformed into the latter during preparation.
The **Physics** amplitude is usually calculated according to a physics model and a formalism. The amplitude is either provided inside the software package or is described with mathematical toolkits beforehand and interfaced as a whole into the framework. It needs to be set up by the analyst according to his understanding of the physics of the experimental data. During the fit procedure, the **Physics** module usually provides an intensity value for a given event based on a given set of parameters. The setup procedure and the parameter set strongly depends on the physics model used.
The fitting procedure relies on two parts: the **Estimator** function which calculates an estimate of the discrepancy between the **Physics** model and **Data** and an **Optimizer** routine which varies the parameter in order to find the setting with the minimal discrepancy. Common examples for **Estimator** functions are the maximum likelihood estimator or the χ^2 estimator which will be discussed later in detail. A widely used optimization package is the Minuit2 library which provides e.g. gradient descent based strategies [33].
The workflow of the analysis is the following: The **Optimizer** derives an improved parameter setting. The **Estimator** calculates a goodness of fit value, the discrepancy between the amplitude and the data by evaluating the **Physics** amplitude at the given **Data** points. The goodness of fit value is fed back to the **Optimizer** and it varies the parameters accordingly. Once a convergence criteria like the estimated distance to minimum is reached, the fit procedure is

stopped with the best parameter setting found.
In addition, a toolkit can provide much more than these mandatory parts, e.g. simple user interfaces, documentation and visualization tools or means for parallelization are present in nearly all toolkits as well. Widely used are fitting frameworks which strictly combine parts of the analysis, e.g. the fitter provided by ROOT [34] which combines **Estimator** and **Optimizer**. However, one then relies on the provided functionality and has to conform to the given interfaces.
For a dedicated analysis it is possible to combine certain parts of this layout in order to optimize the runtime of the fit procedure. The amplitude and estimation function for instance can be calculated and optimized together in case this allows for symmetrization or other modifications of the amplitude.

3.3 Existing Software Packages

As mentioned already, there are many software packages available performing amplitude analyses. The most common tools used in hadron physics are shortly described in the following to demonstrate their strengths but also their limitations. The experiences of both, strengths and limitations, were used to form the requirements and goals of the new amplitude analysis framework ComPWA.

- PWA2000
  The PWA2000 [35] environment is one of the first object-orientated, experiment-independent approaches for amplitude calculations. It is designed to provide a flexible user interface to easily build amplitudes for general physics cases rather than to provide a complete fitting framework. Nevertheless, test analyses have been performed with Minuit [33] as the fitting environment. The amplitude is calculated using the helicity formalism and the isobar model and the implementation of other methods should be possible due to its object-oriented design, but it is not completely modularized.
- RootPWA
  RootPWA [36] is used for the analyses of the COMPASS experiment [37]. As its name suggests, it is based on the ROOT framework [34], which offers a lot of functionality and tools for RootPWA but also means strict dependencies. Initially, its amplitude calculation was based on PWA2000, which means it was limited to helicity amplitudes and diffractive production only. Later, RootPWA was updated with an amplitude generator which is independent from distinct formalisms and models which is being tested at the time of writing this thesis. Data formats and optimization algorithms are still based on and limited by the ROOT functionality. As it is used solely for COMPASS analyses, the available amplitudes are tailored to and optimized for the corresponding physics e.g. by combining amplitude and estimation function for faster calculations. For PANDA physics or other experiments, e.g. electron-positron collisions, new amplitudes need to be implemented first.
- Tara
  The Tara software [38] was written for analyses of antiproton-proton collisions at the Crystal Barrel experiment [39]. Tara is an independent tool which contains interesting features, e.g. switching of optimization libraries, a tree structure for the amplitude representation, abstract parameters, and many more. The physics of the Crystal Barrel experiment is similar to the PANDA experiment, as it produced the same inital state of antiproton-proton in flight as the PANDA experiment and its energy range overlaps with the low energies of the PANDA range. Although the code is not compatible with current compilers and the design of Tara is not modular, parts of the implementations and user interfaces might be of interest to be used as modules for ComPWA, as the C++ code should be portable.

- GPUPWA
  GPUPWA [40] is an amplitude analysis framework optimized for fast fit procedures using graphic processing units (GPU). Both the amplitude model and the goodness of fit calculation can be calculated on GPUs, which in combination with clever caching mechanisms allows for very fast analyses. As PANDA and ComPWA are long term projects, it was decided to trade the specialization on GPUs and the speedup for more general parallelization methods. The development of GPUs and especially the availability of future GPU farms is not predictable enough to rely on it. Therefore, specialized estimation functions or amplitude modules using GPUs are not part of the framework but a possible future extension of ComPWA.

- Laura++
  Laura++ [41] is a ROOT dependent framework developed for BaBar [42] analyses and now reactivated for the LHCb experiment [43]. It uses the isobar model and offers efficient calculation of the amplitude model. However, the means to speed up the calculation are tailored to the specific amplitude model by optimizing it to the likelihood calculation and it has to be checked case by case if the method can be reused for other amplitude models. It is still under development and new features might emerge in the future.

- AmpTools
  The AmpTools [44] package is a collection of C++ classes for amplitude analyses, which makes its design similar to ComPWA's modules. It is in development by a group of the Indiana University and is used for the BESIII [7] and the GlueX [45] experiment. For fast calculation, AmpTools likelihood calculation can be run on GPUs or be parallelized on a farm. Its design is partly as modular as ComPWA's, but with a different set of modules e.g. the likelihood function and optimization are mostly fixed. The AmpTools package is a good example of how a modular design also leads to few dependencies on other packages. Its development is still ongoing and it will be used for cross checking of BESIII results in the future.

To summarize and generalize, there are two major difficulties with existing amplitude analysis software: built-in, fixed amplitude models are tuned for certain physics cases making it hard to generalize them. Strong third-party software dependencies lead to unclear future portability and maintainability. In addition to the physics calculation, there are often also other parts which are restrictive, e.g. the estimation method, which is not crucial but is in contradiction to the flexibility which might be needed in the future. Nevertheless, there are many amplitude analysis tools available with various interesting features, from which ComPWA will learn and also profit by porting features and performing cross checks.

3.4 Requirements

In the process of designing the analysis framework and writing down all the requirements, the importance of separating the four main tasks in four completely detached modules became obvious: Data, Physics, Estimator and Optimizer (figure 3.2). The modular design allows an easy implementation of new amplitude models or new experimental setups without the need to change other parts of the software.

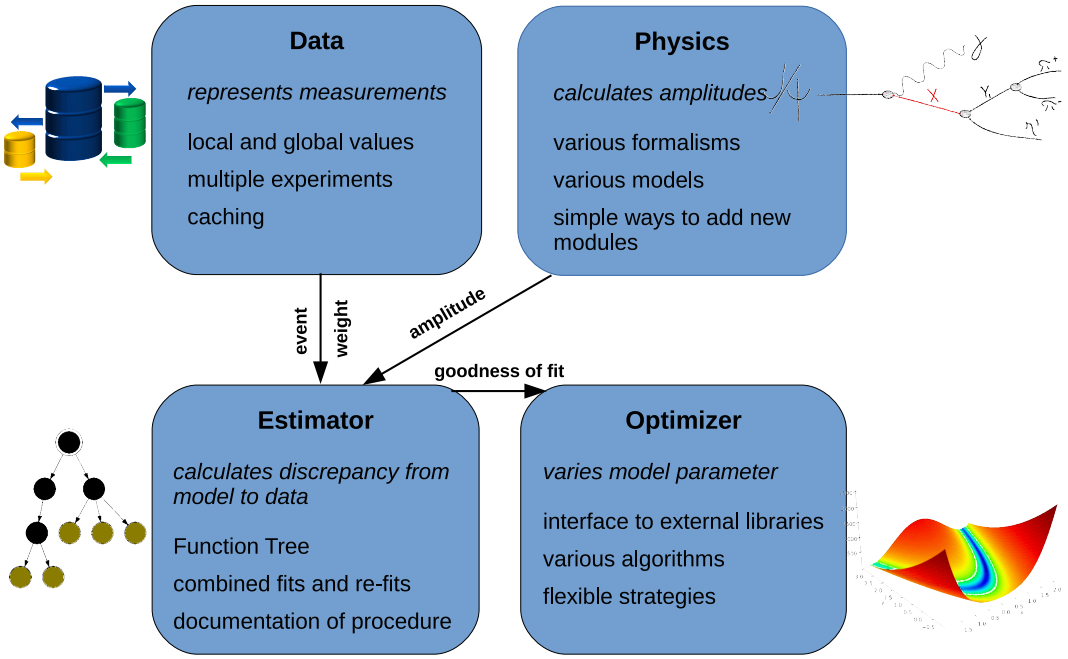

Figure 3.2: Modular design of ComPWA and key features of the modules.

The most important design feature, to reach the goal of being usable for various experiments and analyses, is the separation of the experimental information from the amplitude calculation. Therefore it is possible, right after the implementation of a new amplitude model, to directly test it without the need of changing the fit procedure and especially the access to experimental data. On the other hand, if a set of amplitude models is already available, experiments only need to provide their data in the correct format via a data reader and the fitting of amplitudes to their data can start without reimplementing models and formalism.
The scope of the Physics module (upper right in figure 3.2) is to cover everything necessary to describe the data via an amplitude. One can expect that there will be various implementations of the Physics module, as there are many amplitude models thinkable and not all models and formalisms are suited to describe all available data. The Data module (upper left in figure 3.2) covers everything related to the experiment, e.g. providing event based and general information. One of its challenge will be the handling of big datasets by caching or parallelization methods. There will be various implementations of the Data module, as different experiments usually use different data formats and provide different information. The two other modules (lower left and right in figure 3.2) are dedicated to the fit procedure, which should also be separated from the data handling and the amplitude modeling: the Estimator module, which calculates a measure of the discrepancy between the amplitude model and the data, given for a certain set of model parameters, and the Optimizer module, which takes care of the variation of the parameter. There might be some Estimator modules depending on binned or unbinned estimation or dedicated to special fit procedures, but there are only a few common implementations for most of the standard amplitude analyses. The modularization allows for Estimator modules which perform combined fits. The estimation can be performed efficiently with a Function Tree, which is described in section 3.7. The optimization is usually performed by external libraries, as there are quite a number of well tested and elaborated algorithms. Still, separating the optimization as an own module is important as it simplifies the switching between optimization algorithms, allows to add further external algorithms and also to implement own algorithms.
For the fit procedure, the modules provide the necessary information via interfaces defined by ComPWA. In the setup phase, the Physics module informs the Optimizer about the parameters of the amplitude and the Data module about kinematic parameters needed for the evaluation. In the fit phase, the Optimizer varies the parameters and asks the Estimator for an estimation value. The Estimator calculates it by evaluating the amplitude provided by the Physics module

at the position of the measured events or Monte Carlo events provided by the Data module.
The key to maintain this modular design is to use simple, non-restrictive interfaces to allow different implementations and therefore any kind of analysis. Then, reusing, adapting, and switching between modules will be easily possible. The concept is illustrated in figure 3.3 where e.g. the completely independent treatment of data from different experiments is shown by exchanging the Data module without changing anything else.

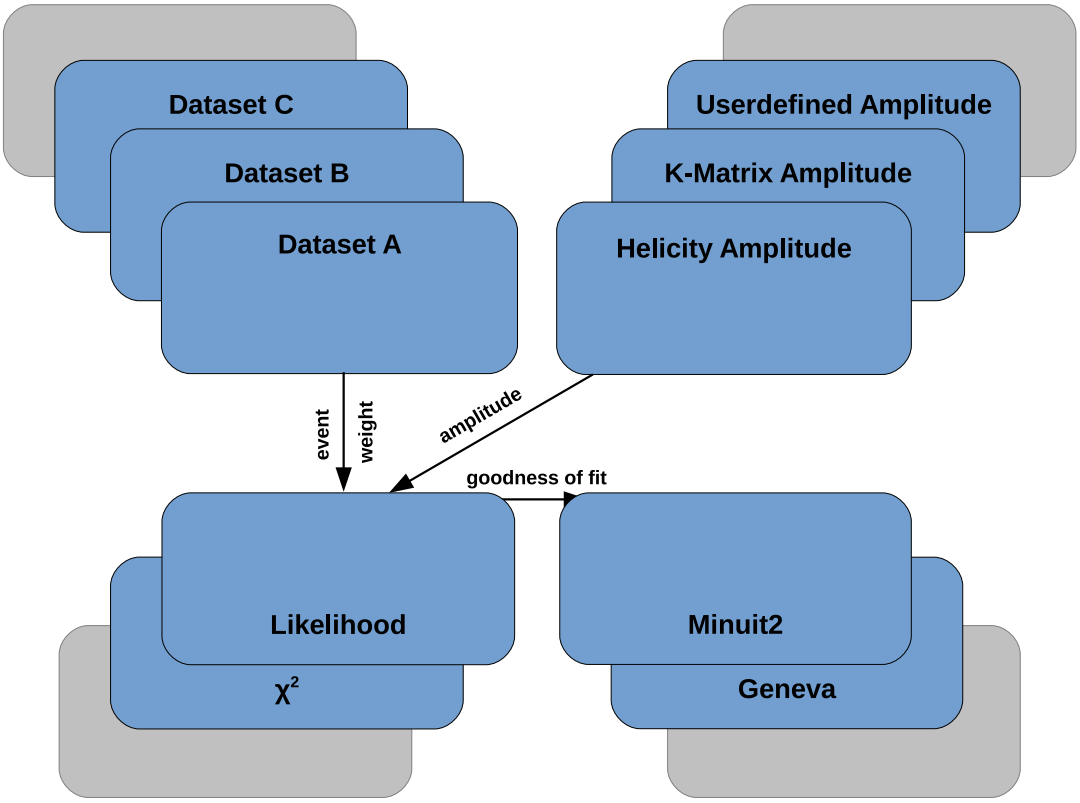

Figure 3.3: Modular design of ComPWA with a small set of modules. Geneva [46] and Minuit2 [33] as optimization modules refer to the respective libraries.

For the user interface, ComPWA is designed to make scripting in various languages possible. This can be achieved by providing a library with the ComPWA interfaces which can be accessed within the scripts. Input is provided by configuration files in an human-readable format.
The output, logging and fit documentation is crucial for usability and, even more important, for quality assurance: An amplitude analysis often has many options, e.g. mathematical formulations within the model, approximations, number of waves, limits of model parameters, parameters of the optimization, etc. Some of these choices do not influence the result of the analysis but they have to be documented nevertheless, e.g. to verify this assumption. All information necessary to exactly redo the analysis has to be documented in order to allow for cross checks and comparisons. Therefore, the ComPWA logging and fit documentation is designed as a module within the framework which gathers all information provided by the modules without providing feedback. In order to achieve this, the module stores all configuration files, user input, and fit results. The visualization of the output, e.g. by generation of plots, is a separate, independent tool which can run in parallel but might use part of the ComPWA modules. Furthermore, it is planned to persist the complete application, which will allow to restore a ComPWA analysis e.g. to continue a fit after a crash.
For the best compatibility with existing tools and libraries and for the possibility to use the extended standard template library C++11 was chosen as programming language. The framework core and the interfaces do not depend on external libraries to ensure long term support.

The only exception are the Boost libraries [47] which are designed closely to the C++ standard template library (STL). Boost provides potentially useful containers, e.g. graphs, corresponding algorithms, file parser for various formats, the platform-independent built system 'boost build' and more.
Also, a team working environment for the ComPWA framework was established including version control using Git [48], wiki pages and in code documentation using Doxygen [49]. It is also publicly available on GitHub [50] using the GNU General Public License [51]. This builds the basis for future developments and the usage across experiments.
The main design features of the software package, the modules, the interfaces, and the Function Tree are described in the following sections before first implementations are presented.

3.5 Software Design

First of all, the decision had to be taken how to design the ComPWA framework. In 1988, Ralph E. Johnson and Brian Foote described the definition of a framework in context of object-oriented programming like this [52]: "An object-oriented abstract design, also called a framework, consists of an abstract class for each major component [...] The framework often plays the role of the main program in coordinating and sequencing application activity." One can differentiate between different types of frameworks depending on their use case although there is not always a clear separation possible [53]. ComPWA can be described by the following two types:

- Application Frameworks
  build the programming conditions for a certain type of applications. Functions and structures used in this type of applications are provided. In case of the ComPWA framework, the application type is amplitude analyses. ComPWA provides general tools, e.g. container for experimental information, fit parameter management or fit documentation, and a structure for the fit procedure. One can imagine the framework as an incomplete software which works as a basis for multiple applications and therefore simplifies their implementation.
- Component Frameworks
  abstract of the object-oriented level and offer an environment for the development and integration of software components. Software components are often viewed as bundles of classes with clearly defined interfaces. The components in ComPWA are represented by the core modules, which provide abstraction, decoupling of dependencies, and an inherent workflow. In addition to the features of an application framework, as described above, the implementation of an amplitude analysis application using ComPWA is further assisted by separating the needed functionality in independent components. This modularization leads to higher reusability of modules and reduces the amount of reimplementation in future analyses. But even in case of a completely new application which needs new implementations of all modules, ComPWA simplifies this by providing the interfaces to the modules, the structure of the fit procedure, and useful tools.

This shows that the definition of the module interfaces is a very important step in order to provide a useful and adaptable framework. The interfaces to the modules also act as wrappers for possibly complex implementations, simplifying the reusage of module implementations inside and outside of ComPWA.
The interfaces define the interaction between modules and therefore the methodology of ComPWA. In principle, once designed the interfaces should stay fixed for all usages of ComPWA, but extending and remanufacturing them is foreseen and possible. As the design of the module interfaces of ComPWA is most important they are described in the following in detail.

3.6 Modules and Interfaces

Physics Module

This module provides the theoretical model aiming to describe the data. This means it must set up and provide decay trees, build amplitudes based on models and formalisms, and handle transformations of physical constraints and parameters. It should also offer access to external libraries or tools in order to calculate amplitudes. Possible formalisms which are commonly used are the helicity formalism [54] and the tensor formalism [55] to calculate decay angles or the *K*-Matrix approach [56] for a complete amplitude. It is also foreseen to use other decay descriptions than the commonly used isobar model [57] to handle effects like rescattering [58]. Therefore, the interface for this module is implemented as:

```
bool fillStartParVec(ParameterList)
double intensity(ParameterList, PhspPoint)
FcnTree getFcnTree()
```

The *fillStartParVec* function is used when setting up the fit procedure. After the setup, the Physics module provides a list of model parameters to the other modules. This list also contains the information which parameters are floating in the fit, whether they have limits etc. The *intensity* function is used by the other modules to request the calculation of the intensity for a set of parameters at a given phasespace position. The returned intensity value can then be used by the Estimator module to calculate how good the model fits to the data. The *getFcnTree* function provides an alternative way to perform the same intensity calculation, namely by providing a Function Tree (see section 3.7) instead of performing the complete calculation every time.

Data Module

The Data module is taking care of all experiment related information. It is used to read, write and manipulate experimental data as well as generate and store phasespace or amplitude distributed toy Monte Carlo data. It provides event based data and global values, like metadata from the experiment. Since it is foreseen to perform parallelization on data level, reading and caching of a certain part of the data should be possible.
The duty of this module is mainly to handle input and output of event information, therefore the interface is implemented as:

```
void pushEvent(Event)
Event getEvent(integer)
double getBin(integer)
```

The *pushEvent* function is used to write back data, e.g. when data is generated with ComPWA (phasespace distributed or according to a model). The *getEvent* function provides access to the data on event level and *getBin* on binned data. In addition, functions are provided to get basic information like event size and global information from the experiment, e.g. beam energy or target polarization.

Estimator Module

The Estimator module requests information from the Physics and the Data module and provides a measure on how good the model describes the data for the current set of parameter. The most common estimation functions are the least squares method and the binned and unbinned likelihood [59] functions. Unlike the Data and the Physics modules being highly dependent on

the analysis, the number of expected estimator modules might be rather small. Most likely, new implementations for this module are needed only when it is necessary to perform special analysis techniques. If the Physics module provides a Function Tree (see section 3.7), the Estimator module can extend and manage the Function Tree instead of letting the Physics module perform the calculation every time. The interface is implemented as the following function:

```
ParameterList controlParameter(ParameterList)
```

It is quite simple as the input, a set of parameters the model needs, and the output, a goodness of fit estimate, are well defined. The goodness of fit estimate can be a simple value, but one can imagine that also more complex information being provided, e.g. a set of likelihood values or complex numbers. Therefore, it is provided in form of a parameter container which can hold more information.

Optimizer Module

The Optimizer module organizes the variation of the model parameters in order to find the best fit to the data. Optimization algorithms for these tasks are widely available so that most implementations for the Optimizer module will be wrapper to external libraries providing the algorithms. Own implementations are also possible and might be used in special analyses or for meta fits (automatized performing of multiple fits, e.g. to scan systematic effects).
The optimization algorithm gets the necessary information for the optimization process (e.g. a sum of residuals between model and data) directly from the Estimator module and performs the optimization process until a predefined criterion has been reached, therefore the interface is implemented as a trigger function to start the parameter optimization:

```
FitResult exec(ParameterList)
```

As the values of the parameters are accessible from the outside during the optimization, it is not necessary for the optimizer module to return them. The return value consists of the fit result after the optimization stopped, this can be used to cross-check the final goodness of fit value which was found by the optimization. Manipulations during the fit process depend on the capabilities of the optimization library and have to be implemented specifically.

Fit Control

The setup and control of the fit procedure is managed via configuration files and scripts using a script-language of choice. The ComPWA modules and interfaces will be made available via libraries which can easily be loaded in different script-languages, allowing users to use their preferred language. In future, this scheme can also be extended by a graphical user interface. However, ComPWA analyses are performed via C++ executables and XML configuration files as long as it is in the development phase.
The XML configuration files are human readable but also easy to create, manipulate, and read in applications and scripts. Figure 3.4 shows part of the XML configuration file for the $J/\psi \to \gamma\pi^0\pi^0$ amplitude. This is important as it is planed to use an expert system (see section 3.9) to aid the setup of the physics models and formalisms. They can be used to provide the rules of a formalism or to gather information, e.g. quantum numbers, from different sources. Ideally, the user provides the basic information, the expert system adds information of a particle database or applies the rules of the formalism and model, and if this is not sufficient, it asks the user to provide certain additional information.

```
< amplitude_setup >
  < filename > JPSI_ypipi.xml < /filename >
    < resonance >
    < enable > true < /enable >
    < name > f0_980 < /name >
    < type > relBW < /type >
    < reference > f0_980 < /reference >
    < mass > 0.99 < /mass >
    < mass_min > 0.5 < /mass_min >
    < mass_max > 2.0 < /mass_max >
    < mesonRadius > 1. < /mesonRadius >
    < width > 0.05 < /width >
    < width_min > 0. < /width_min >
    < width_max > 2. < /width_max >
    ...
  < /resonance >
  < resonance >
    < enable > true < /enable >
    < name > f0_1500 < /name >
    < type > relBW < /type >
...
```

Figure 3.4: Extract of $J/\psi \to \gamma\pi^0\pi^0$ amplitude XML configuration file.

Bookkeeping

A very important task is to monitor the fit procedure by documenting the information of every optimization step. The Bookkeeping module will passively collect all information provided by the modules in each fit iteration. Reporting tools, which run parallel to the fit itself, can be configured to show the information e.g. by plots. The stored information has to be sufficiently detailed for the ability to restart fits at any iteration in case of computer failures and to be able to redo or change only a part of the fit procedure.
Up to now, there is no implementation of the bookkeeping scheme available within ComPWA although it is part of its requirements and a very important issue. The feasibility of the framework design and first analyses have a higher priority tough. Results are obtained as ROOT or Ascii encoded files and are analyzed by hand.

3.7 Function Tree

As the goal of the project is to have a very flexible and generally usable amplitude analysis framework, there is also the necessity to provide a general approach to perform efficient calculations independent on the physics model. Common to all physics models is their functionality to calculate the amplitude, most likely a complex number, for each data point and for a set of model parameters via a more or less complicated formula. With this formula, it is possible to build a tree like representation where parts of the formulation are cached and only recalculated if necessary, else the stored values can be reused. In particular for the fitting procedure there is a large gain in performance, as the amplitude needs to be recalculated for every data point every time the optimizer varies the parameters and needs to check the estimation function. For the most common models, most parts of the tree stay constant while only a small part needs recalculation: e.g. in case only one resonance is varied, while the rest stays fixed and does not need to be recalculated. In ComPWA, the tool to provide this tree like representation of formulas is called the Function Tree.

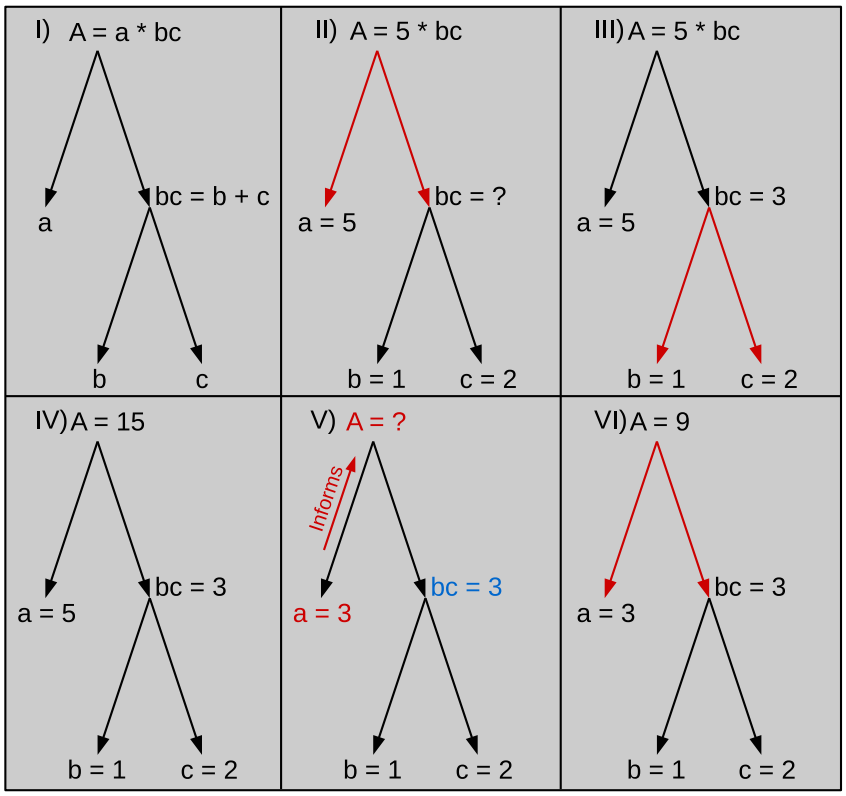

Figure 3.5: Example of a Function Tree in several stages: I) Setup II) Calculation starts III) Calculation of bc IV) Calculation finished V) Input changes VI) Faster recalculation.

The basic idea of the Function Tree is shown in figure 3.5. It illustrates the distribution of a simple calculation (e.g. $A = a \cdot (b + c)$) in a treelike structure which caches part of the calculation. The calculation of this formula is constructed by two nodes, *A* and *bc*, and three input parameter, *a*, *b*, and *c*. During setup, the node *bc* is created, linked with the input *b* and *c*, and the operation " + ". The second node *A* is created and linked to *a* and the node *bc*, and the operation " $\cdot$ ". When initialized, the top node *A* tries to calculate its value by multiplying its two input values (figure 3.5 II). *a* is a parameter with a value, but *bc* has no value yet, so recursively *bc* calculates its value by summing up the values of its two input parameters (figure 3.5 III). The result is passed to *A*, which then can calculate its final value (figure 3.5 IV).
The caching feature of the Function Tree is important when only part of the calculation changes

and the top node is recalculated. If, in the given example, the value of parameter *a* is changed (figure 3.5 V), *a* informs its parent, node *A*, about the change, so *A* knows it is not up to date anymore. This would go on recursively if *A* had any parents. No recalculation is performed yet as additional parameters could change as well. The recalculation is triggered only when asking *A* for its value, as it was informed that at least one child is changed. It asks *a* and *bc* for its values. As *b* and *c* did not change, *bc* is still valid and does not trigger a recalculation but returns the cached result. Only *A* performs a calculation and the tree is up to date again (figure 3.5 VI).
For amplitude calculations, the trees grow way more complex and the time for calculations which is saved by caching intermediate results is enormous. Measurements of the achieved speedups will be shown later with the corresponding analysis.
For the implementation two goals have to be considered. The first one is to provide the internal representation of the Function Tree with the described feature of caching parts of the calculation. The second one is to provide tools to set up a Function Tree for new models, flexible enough for the different kinds of sub-calculations needed by the various models.

Internal Structure

There are already software libraries on the market to create and use a Function Tree, e.g. the boost graph library [47]. Unfortunately most graph libraries focus on algorithms on the graphs like rearrangement and searching, which are not the features the Function Tree of ComPWA needs. Therefore, the overhead in implementation based on one of these libraries without actually using the algorithms was considered too large. Instead, a new, very slim and specialized Function Tree was set up by only using C++11 and the usual containers of the STL. Most of the features of the tree are implemented in the **TreeNode** class, which has basically the following members:

- Parents: list of pointers to the parent nodes.
- Children: list of pointers to the child nodes.
- Strategy: pointer to the function calculating this node.
- Parameter: list of pointers to the calculated or cached values.
- Recalculate: flag to indicate necessity of recalculation.

The list of parent and child nodes is used to set up the tree structure as they present the connections of the nodes. The node itself is either storing the result of the calculation performed via the strategy, or it is a leaf which holds a parameter as input for the tree. This can either be a fixed parameter or a link to a parameter outside the tree which can be varied by the optimizer. The so-called visitor pattern [60], is used to give note in case these external parameters change. If a change occurs, the node will inform its parents (and they recursively do so to their parents) that they need to be recalculated. If the head node is then asked for its value and is flagged to be recalculated, it will recursively ask for recalculation downward the tree until all parts which have changed are recalculated. This two-staged procedure is used to prevent unnecessary recalculation of parts of the amplitude. Finally, each node which is not a leaf needs a strategy which it uses to calculate its value out of the input values from its children. These strategies can be simple mathematical operations as the sum of all child values, or more complicated functions like a Breit-Wigner description of a resonance.

User Interface

In addition to the bare functionality, a Function Tree should also be easy to use when implementing a new Physics module. For the setup, the manager class **FunctionTree** provides the linking of the nodes, access to the head, and it offers the functionality to check and manipulate the tree. Its methods and member variables are shown in figure 3.6.
A model might also need specific functions to calculate the amplitudes, like the description of the dynamic behavior, angular distributions etc. This can be achieved by the implementation of user-defined so-called strategies, based on the standard strategy pattern [60]: The abstract interface **Strategy** is provided for specific implementations, see figure 3.6.
When setting up the tree, the user creates all necessary strategies first. Then, he creates an instance of **FunctionTree** as manager and uses its functions to add nodes. At this point, it is possible to specify a **Strategy** for a node. The **FunctionTree** creates instances of **TreeNode** accordingly and takes care of the linking. When all nodes have been added and the final checks are successful, the tree is ready to use.

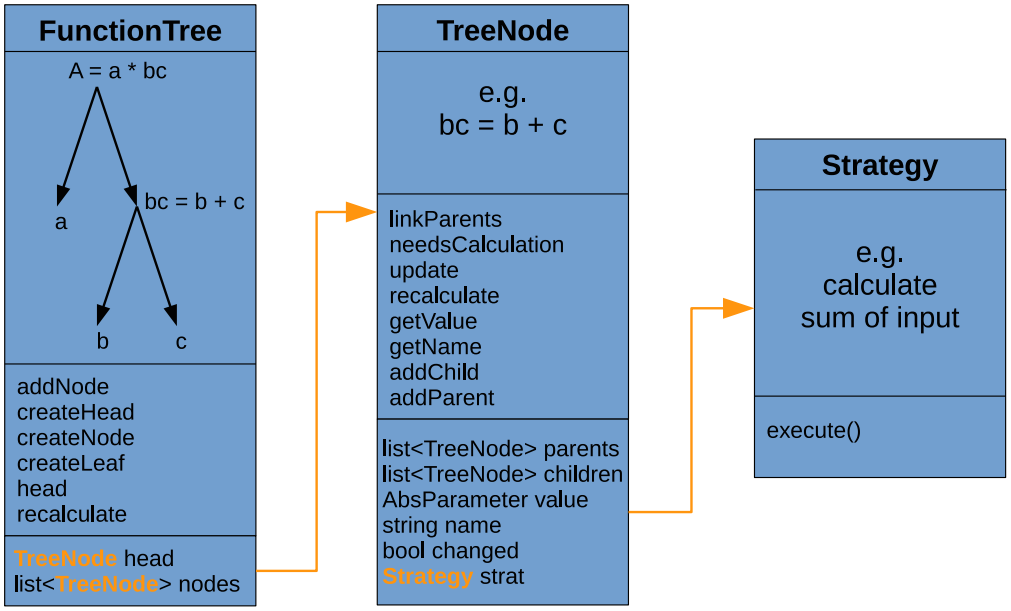

Figure 3.6: Overview of the main classes of the Function Tree, showing methods and members of the interfaces. The example tree refers to figure 3.5.

Further Development

The **TreeNode** and **Strategy** classes together with the **FunctionTree** manager build a fully functional directed graph structure as representation of a formula. It is capable of efficiently performing repetitive calculations, e.g. optimization processes. As shown later, the Function Tree is ready to use and has already performed very well in first tests. The user interface, however, still needs improvement. Automatic user interface tools, e.g. a setup assistant, would simplify the creation of a tree, which until now needs to be performed by using the methods provided by the **FunctionTree** class.
The memory consumption of the tree could be reduced by collapsing static branches of the tree. Also, nodes with direct dependencies (e.g. a parent with only one child) make caching of the children unnecessary (e.g. the parent recalculates only in case the only child changes). Both could be done by some checks after the tree is set up and the list of parameters to be fitted is chosen by the user.

3.8 Implementation of the Modules

For the analyses being described in chapter 4, several modules are already implemented: a Breit-Wigner sum model, a four-momentum data reader using ROOT format files [34], three estimator functions (χ^2, an unbinned and a binned likelihood estimator), and a generator for Monte Carlo data using the hit-or-miss method. Wrapper for two optimization libraries (Minuit2 and Geneva) are implemented as well. This section contains the description of each module. They are later combined for the complete analyses. All modules fulfill the corresponding ComPWA interface. This ensures that they will work with other modules and provides minimal overhead in the implementation.

Physics Module

The purpose of the Physics module is the calculation of an amplitude value at a given phase-space point. To do this, it also has to provide a set of parameters needed for the calculation to be varied by an optimizer module. Optionally, it makes use of a Function Tree for a faster calculation of the amplitude.

Breit-Wigner-Sum Model

This amplitude A of the Breit-Wigner-Sum model consists of the coherent sum of complex relativistic Breit-Wigner functions T_n for n resonances:

$$A(m) = \sum_n c_n T_n(m) D_n(m) \tag{3.1}$$

with c_n being complex parameters for the strength and the phase of the resonance and D_n being spin-dependent Wigner D-functions [61]. Thus, the interference between the resonances is taken into account. The Breit-Wigner function describes the dynamical behaviour of the resonance, while the Wigner D-function describes the spin dynamics.
The following Breit-Wigner formulation (see appendix A.1 for details) is implemented:

$$T(m) = \frac{1}{m_0^2 - m^2 - \mathrm{i} m_0 \Gamma} \tag{3.2}$$

with m_0 being the mass of the resonance, m the invariant mass of the decay products and Γ the dynamical width of the resonance. The dynamical width depends on the total spin J and the mass m:

$$\Gamma(m) = \Gamma_0 \left(\frac{q(m)}{q(m_0)}\right)^{2J+1} \left(\frac{m_0}{m}\right) B_J(q(m), q(m_0)) \tag{3.3}$$

with the width of the resonance Γ_0 and the break-up momenta $q(m)$ and $q(m_0)$. They represent the momenta of the decay product particles with mass m_a and m_b in the rest frame of the decaying resonance and are calculated according to the equation:

$$q(m) = \sqrt{\frac{(m^2 - (m_a + m_b)^2)(m^2 - (m_a - m_b)^2)}{4m^2}} \cdot c. \tag{3.4}$$

In addition, the width contains the spin-dependent Blatt-Weiskopf form factors B_J. The maximum angular momentum in a strong decay is limited by the linear momentum q. Decay particles moving slowly with an impact parameter (meson radius) d of the order of 1 fm have difficulty to generate sufficient angular momentum to conserve the spin of the resonance. The Blatt-Weiskopf factors give weight to the resonance in order to account for this effect. The factors for

the lowest three spins are implemented according to [62] (in natural units):

$$
\begin{aligned}
B_0 &= 1 \\
B_1 &= \frac{1 + z_0}{1 + z} \\
B_2 &= \frac{(z_0 - 3)^2 + 9z_0}{(z - 3)^2 + 9z} \\
z_0 &= q(m_0)^2 d^2 \\
z &= q(m)^2 d^2
\end{aligned}
$$

again using the linear momenta $q(m)$ and $q(m_0)$.
Figure 3.7 shows different visualizations of the Breit-Wigner function $T(m)$ with mass $m_0 = 1.25\,\mathrm{GeV}/c^2$. A single Breit-Wigner amplitude forms a circle in an Argand plot (a). In (b), a phase transition of π around the resonance mass is visible. (c) shows the asymmetric intensity $|T|^2$ of the Breit-Wigner function, which determines the production probability of the resonance.

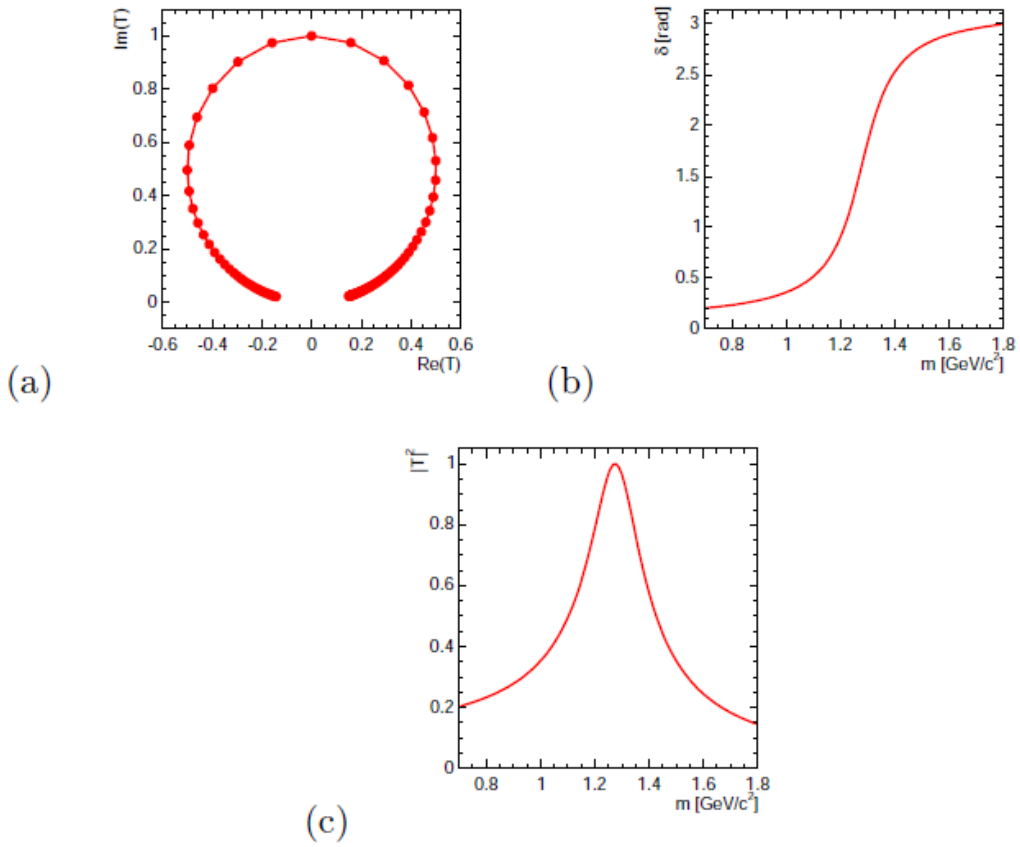

Figure 3.7: Relativistic Breit-Wigner function T (see formula 3.2) [6].
(a) Argand plot: imaginary part versus real part,
(b) phase as a function of the invariant mass,
(c) intensity $|T|^2$ as a function of the invariant mass.

The implementation of the Wigner-D functions were taken from the qft++ [63] library:

$$
\begin{aligned}
D = (-1)^{J+M} \sqrt{(J+M)!(J-M)!(J+N)!(J-N)!} \cdot \sum_{k=MAX(0,M+N)}^{MIN(J+M,J+N)} (-1)^k \\
\cdot \frac{\cos(\frac{\beta}{2})^{k-M-N} \sin(\frac{\beta}{2})^{2J+M+N-2k}}{k!(J+M-k)!(J+N-k)!(k-M-N)!}
\end{aligned}
\tag{3.5}
$$

with J the spin of the resonance, M and N the spin difference to the daughter particles. β is the helicity angle as described in [6]. Three examples of the Wigner-D functions are shown in figure 3.8.

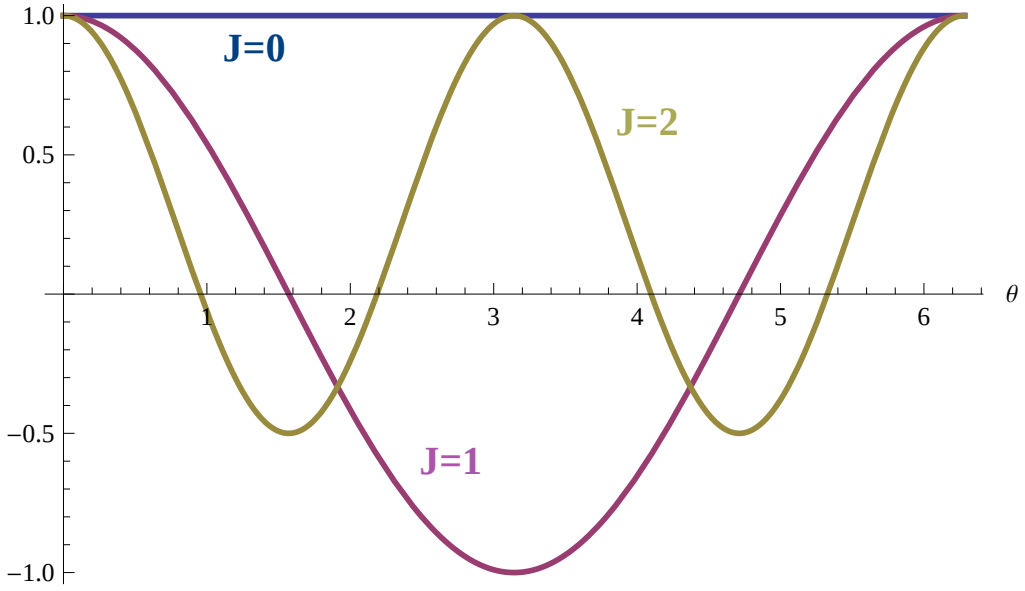

Figure 3.8: Wigner-D functions in dependence of the decay angle θ for the decay into two Spin 0 particles. Shown are three Spin J values of the initial particle.

Outlook

The Physics modules are the crucial part of ComPWA, as a large variety of models is needed for the interpretation of the data from new experiments and to make cross-checks easier. At the moment, only the Breit-Wigner sum model for three-body decays is implemented. Although it could be extended to be more general, it is planned to write a new, more complex helicity formalism generator module using expert systems for better separation of the physics model and the amplitude calculation. Alternatively, the porting of an existing generator allows direct cross checks by reperforming analyses done with the existing software. These options underline the basic concept of ComPWA, which lies in the reusing and maintaining of software packages and aid quality assurance. Therefore, the goal to be achieved is the porting of many different amplitude models to ComPWA for the use in various experiments.

Data Module

A Data module provides experimental data in an either unbinned or binned format. In addition, it should handle Monte Carlo generated data e.g. by generating them according to an amplitude given by the Physics module. This module is also responsible for storing the complete dataset.

ROOT Data IO and Generation

For now, a simple event based four-momentum reader is implemented as a Data module based on ROOT. It is extended with the possibility to generate so-called toy Monte Carlo data for testing purposes.
For the generation of toy data events first of all phasespace distributed four-momentum vectors

are produced with the ROOT phasespace generator *TGenPhasespace*. This generates homogeneously distributed four-momentum vectors according to the energy of the system and the masses of all final-state particles. In order to generate events according to a certain model, the corresponding amplitude has to be provided and the data is modeled from phase space distributed events by the hit-or-miss method [64]. To apply the hit-or-miss method, the maximal value A_m of the amplitude in the given phasespace has to be determined, either analytically or by scanning the amplitude values over sufficiently enough phasespace points. Since an analytic solution depends on the model and might not be available in all cases, the second option was chosen and performed using random phasespace positions with a high number of events. This A_m sets the limit for the random number generator as described in the pseudo-code below:

while *events* $<$ *MaxEvents* ***do***
  generate phasespace event p
  randomly choose $x \in (0, A_m)$
  if $x < A(p)$ ***then***
    keep event
  else
    reject event
  end if
end while

$A(phsp)$ represents the amplitude value at the tested phasespace point p. Since the probability for keeping a random event at position p is proportional to $A(p)/A_{max}$, the accepted distribution resembles the input amplitude model.
The events produced with this method can be written to disk and read again as toy data for any kind of analysis. For convenience, the ROOT file format is used for this implementation as it allows simple checks of the in- and output data by using ROOT tools. It also provides compression of the stored data to save disk space.

Outlook

The Data modules implementations depend on the way the experiment provides the data. In order to properly handle experiment specific information, it might be more effective to implement an individual data module for each data format than a format converter to adapt the data structure.

Estimator Module

Estimation functions are used to calculate the discrepancy between the amplitude model and the data. In that respect two main categories exists to treat either unbinned or binned data. Estimation functions for both cases are implemented.

Unbinned Fits

For unbinned fits, the estimation function operates on event level by evaluating the amplitude for every measured event. For these analyses, the following likelihood function is available in ComPWA [59]:

$$-\log(\mathcal{L}) = -\sum_{i=1}^{N} \log(I(m_i)) + N \log\left(\frac{1}{MV} \cdot \sum_{j=1}^{M} (I(m_j))\right) \tag{3.6}$$

The intensity (the square of the amplitude) $I(m)$ is evaluated at a given position. The intensities are summed over a number of N data events m_i and M phasespace distributed events m_j.

$V = \int d\bar{y}$ is the phasespace volume of the decay channel. The phasespace distributed events are necessary to normalize the amplitude and to calculate the acceptance and efficiency of an experimental setup, e.g. by using accepted toy data from a detector simulation.

Binned Fits

For fits on binned data, the estimation function evaluates the amplitude at the bin position and compares it with the bin content. Two methods for binned data are implemented in ComPWA, a least-squares method and a binned likelihood function. The least-squares estimation is calculated using

$$\chi^2 = \sum_{i=1}^{N_{bins}} (N_i - I(x_i))^2 \tag{3.7}$$

for data distributed in a number of N_{bins} bins, where N_i is the number of events in bin i and $I(x_i)$ is the intensity at the position of the center of the bin.
The binned likelihood estimation is calculated using

$$\mathcal{L} = \sum_{i=1}^{N_{bins}} (-N_i \cdot \log(I(x_i)) + I(x_i)) \tag{3.8}$$

again using the same number of events of the binned data and the calculated intensity $I(x_i)$ at the position x_i of bin i.
In the future, the method can be extended by integrating the intensity over the bin range or taking the average of some more sample points in the bin range, leading to a better expectation value from the model.

Outlook

With the likelihood calculation, the most common estimation function is provided. Still, it can be expected that specialized likelihood functions might be implemented in the future. Also, automatic procedures which perform multiple fits e.g. to sample over sets of initial parameter values of a fit could be realized by providing a correspondent Estimator module.

Optimizer Module

The optimization routine varies parameters in order to find an optimal estimation value, e.g. a minimal discrepancy, between the amplitude model and the data. As there are well tested and widely used optimization libraries on the market, it is constructive to use these packages by writing wrappers for them which fulfill the ComPWA optimization interfaces. The libraries used here are the Minuit2 [33] library and the rather new Geneva (grid enabled evolutionary algorithms) [46] library, as both offer a complementary set of optimization algorithms suitable for various problems.

Minuit2

The Minuit2 library offers a physics analysis tool for function minimization. It is part of the CERN library and used in the ROOT framework. It offers various optimization algorithms (gradient descent based, simplex) and error analysis tools (Hesse matrix, asymmetric errors). With the gradient descent algorithm, Minuit2 provides a stable optimization method with well tested correlation and error estimations. The gradient descent locates the local minimum of a function by varying parameters proportional to the negative of the gradient (or of numerical approximated

gradient) of the function. Thus, it varies only one parameter at each iteration.
In addition, it offers Monte Carlo based methods for optimization problems with complicated parameter spaces featuring lots of local minima. In contrast to the gradient descent, they randomly create a new sets of parameters in each step and it is possible to overcome local maxima to find other minima.
To enhance its runtime, Minuit2 can be run in a multithreaded mode using the full power of multicore machines. For some optimization algorithms, Minuit2 provides covariance matrices for the optimized set of parameters, which is provided by the interface to analyze the correlation of the parameters.

Geneva

The Geneva library offers highly variable genetic and other Monte Carlo based algorithms as well as a gradient descent method, but for now lacks the error estimation Minuit2 offers. It allows the complete control over the algorithms, parameter variation methods and convergence criteria. Since the principle of a genetic algorithm bases on the simultaneous handling of multiple parameter sets, it is optimized for parallelization on farms and grids. In the parallelization mode, an arbitrary number of slave processes could run on a farm, each evaluating a different parameter set on the same data. A master process gathers the results and generates new parameter settings (so-called offspring) based on a set of the best N parameter settings to submit new slave processes for a next iteration. A beneficial factor of this method is that the master does not need to wait for all results but can also operate with a smaller set and use the delayed results in the next iteration. In contrast, when splitting the data and calculating part of the estimation function at each node, the master needs to wait for the slowest node to continue. On the downside, the parallelization Geneva offers is only efficient for calculation of estimation functions which take long enough that the overhead created by the needed network communication is small enough in comparison.

Outlook

With access to two libraries each offering a set of different optimization algorithms, ComPWA already offers solutions for many fit procedures. The usage of the various algorithms of Geneva needs testing and tuning for hadron spectroscopy problems and due to the active development, it will provide even more options in the future. There are no short term plans to add further modules, but since the interface is already working with two very different libraries, nothing speaks against the implementation of additional wrappers for future optimization algorithms. It is also possible to implement own optimization routines.

3.9 Expert Systems

In order to provide a simple user interface, it is planned to use an expert system to help setting up decay trees and amplitudes for ComPWA. Expert systems use knowledge databases and a set of rules for decision making. They can either be implemented by using general rule-based programs like Mathematica [65] or by using dedicated languages. Although there are no such features available yet, first tests were performed to illustrate the capabilities of such systems.

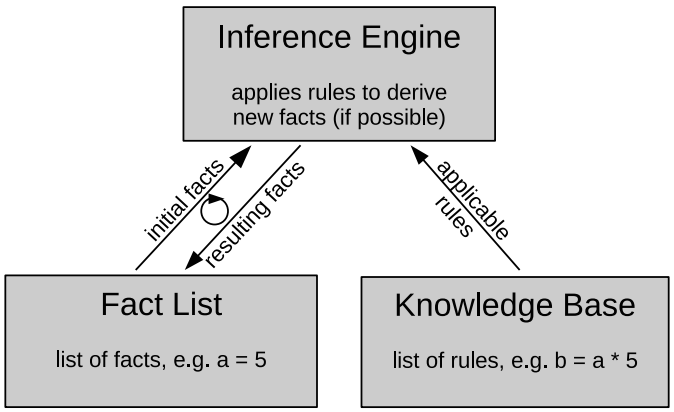

Figure 3.9: Basic components and workflow of an CLIPS expert system [66]. The circle indicates recursive application of the growing Fact List to the rules.

Figure 3.9 shows the three key elements of an expert system: the Knowledge Base, the Fact List and the Inference Engine which operates on the former two. The Inference Engine infers all deducible information based on the rules from the Knowledge Base and the existing Fact List and adds them to the later. If new facts were introduced, the process continues as now these rule conditions might be fulfilled too. Two iterations of this process are shown in figure 3.10 for a very simple set of rules and facts.
For a test, the CLIPS (C Language Integrated Production System) package [66] was chosen. CLIPS provides the mechanics of the Inference Engine and a language to easily define rules and facts. The goal was the generation of an amplitude using the rules of the helicity formalism [54] and the facts of an example decay (figure 3.11).

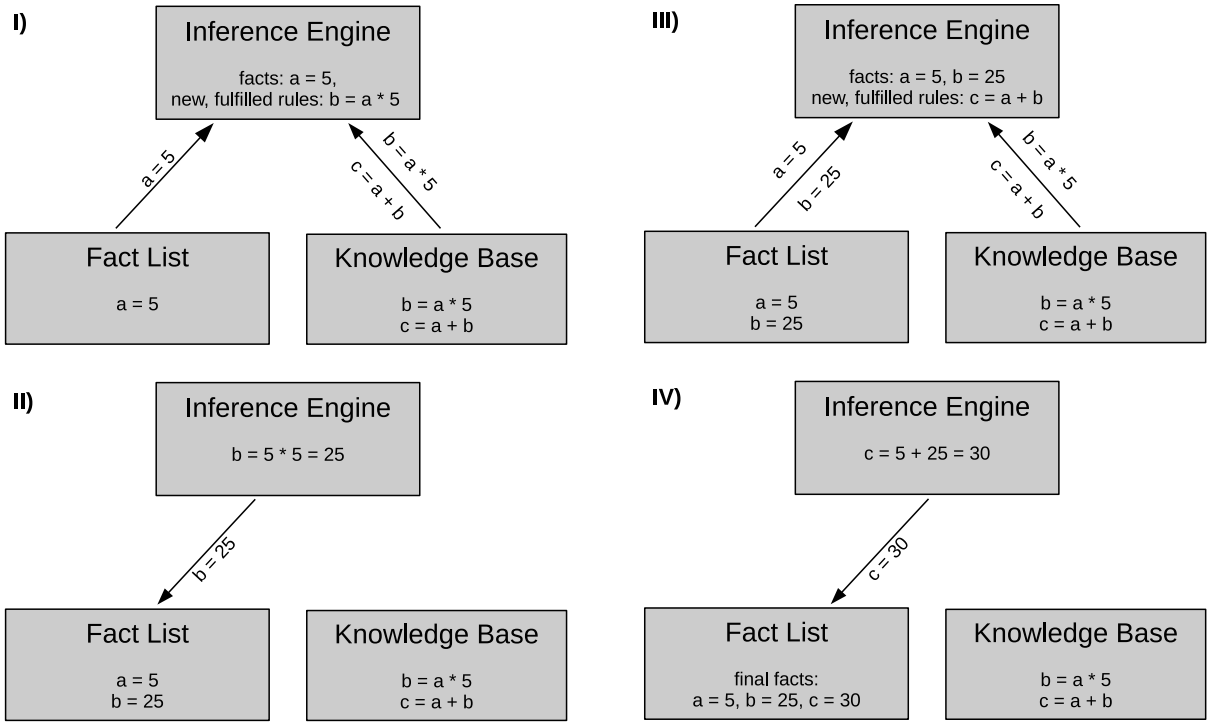

Figure 3.10: Example workflow of an expert system. In step I) the Inference Engine gathers information. In step II) it applies the fulfilled rules and writes new facts to the Fact List. As now a new rule ($c = a+b$) is fulfilled, a second iteration starts and is shown in III) and IV).

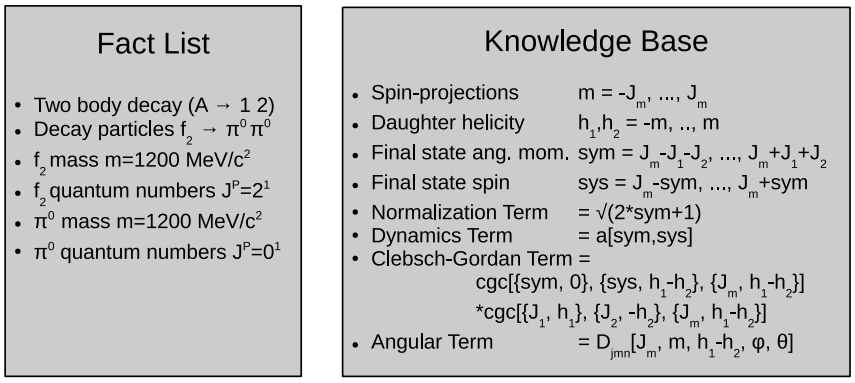

Figure 3.11: Fact List and rules for a two body decay helicity amplitude.

Example: Amplitude Generation

To work with the information in figure 3.11, it has to be provided in a format CLIPS understands. Therefore, three defined templates, a Fact List, and several rules were used, see appendix A.4 for the CLIPS input files. The templates cover the complete information of a decay in the helicity formalism, of the final state information, and of the amplitude terms, the latter being the desired output. Another template, the decay template, contains the input information.
An expert system builder like CLIPS does not need to explicitly trigger the rules. Instead, if the described Knowledge Base (with the rules) is loaded and some input facts are provided, CLIPS automatically initiates the inference process to create output once started. When new rules are introduced, the process starts again until no new facts can be concluded.
In this example, the first iteration can only infer the first three rules of figure 3.11, as the others are not fulfilled yet. But the three valid rules create new facts which triggers the next iteration.

Specifically, for the given example decay of a spin $J = 2$ particle, CLIPS first generates five final state templates with a mother spin projections of $J_z = -2, -1, 0, 1, 2$. Then, amplitude templates where the respective spin is put as a parameter into the terms of the helicity amplitude are generated. As an example, the $J_z = 2$ outcome would be the following terms:

(AmpTerms (djmnterm "Djmn[2.0,0.0,0.0,Phi,Theta]]") ·
(cgcterm "ClebschGordan[0.0,0,2,0.0,2.0,0.0] ·
ClebschGordan[0.0,0.0,0.0,-0.0,2.0,0.0]") ·
(dynamicterm "a[0.0,2]") (normterm "Sqrt[1.0]"))

The complete output is listed in appendix A.4. The output consists of plain text placeholder for functions with different quantum number and helicity value combinations. Some terms have Clebsch-Gordan coefficients of zero, but this is not yet known by the expert system. In order to calculate an intensity, the placeholders can be interpreted as calls to functions which calculate the components and a summation of all amplitudes is needed.
It is important to note here that the Knowledge Base of this example is by far not a complete helicity amplitude generator. It only uses parts of the helicity formalism to illustrate the idea on how to use an expert system and why this is beneficial.

Adoption to ComPWA

For the usage in ComPWA, CLIPS offers an interface to the C programming language, an object-orientated extension called COOL [66] and is available via package managers on many linux distributions. It is a public domain software with full documentation, tutorials and examples online. Also, classical textbooks are available. So far, it seems the expert systems planned for ComPWA are realizable with CLIPS. If so, the amplitude generation would be easy to maintain and assure a better quality as the Knowledge Base is by far easier to document, adapt, and fix as comparable code where all steps of building the amplitude have to be implemented by hand.

4 Analysis of $J/\psi \to \gamma\pi^0\pi^0$

In this chapter, the features of ComPWA are utilized to compare different methods for an analysis of the light scalar meson sector. The scalar wave at low energies consists of many broad and thus overlapping resonances which are not yet fully understood. The interference of the resonances and threshold effects make it difficult to model the dynamics of the resonances. Therefore, an unbiased extraction of the scalar wave without the systematic effects introduced by the dependence of the resonance models is highly desirable. To understand the benefits of an approach like this, a conventional Dalitz plot analysis is performed in addition to the model-independent approach for comparison.

4.1 Reaction Channel

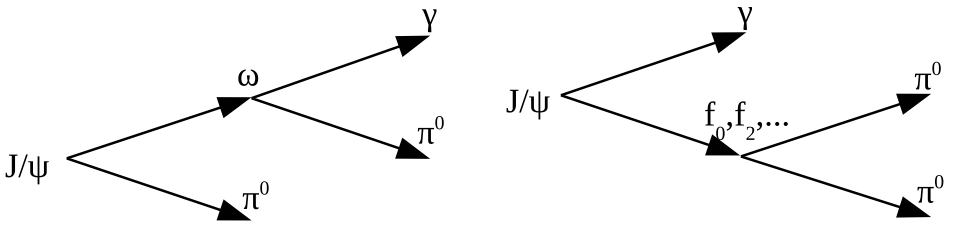

Figure 4.1: Diagrams of the considered isobar decays of J/ψ to the final state $\gamma\pi^0\pi^0$. The *f* represents the various scalar and tensor *f*-resonances as listed in table 4.1.

A suitable reaction for studying the scalar wave is the decay $J/\psi \to \gamma\pi^0\pi^0$:

- $\pi^0\pi^0$ limits the possible quantum numbers:
  With two identical, permutable particles, symmetry of the wavefunction and parity conservation limit the quantum numbers of resonances in the $\pi^0\pi^0$ subsystem to $J^{PC} = 0^{++}, 2^{++}$, *etc*. with the higher spins suppressed by the angular momentum barrier. In the $\pi^+\pi^-$ subsystem, also additional spin parity numbers like $J^{PC} = 0^{-+}, 1^{++}, 1^{-+}$, *etc*. are possible. Thus, $\pi^0\pi^0$ provides less complicated access to the wave of interest.
- $\pi^0\gamma$ is rather uncomplicated:
  The only visible expected resonance in this subsystem is the narrow, well-known $\omega(782)$ resonance. This means there is only a small region where interference with the resonances in $\pi^0\pi^0$ is present and little systematic effects are introduced by modeling the resonance. An alternative would be to remove the data in the $\omega(782)$ region from the analysis.
- J/ψ is a narrow resonance:
  The small width of the J/ψ resonance has two beneficial implications. First, there is a large and clean dataset available as the J/ψ resonance is easy to produce and separate and second, the energy of the system is well defined. This is a big advantage when reconstructing the final state particles and therefore introduces less uncertainties to the amplitude analysis. In particular in a binned Dalitz plot fit there is no need to correct it if the binning is wider than the reconstruction accuracy .

Figure 4.1 illustrates the relevant decay modes. For the development of the ComPWA framework and especially for the model-independent fit, toy Monte Carlo data were simulated according to the resonances and parameters listed in the PDG [2] (Table 4.1). Not all resonances were included in the simulation study. Also, for first tests only a subset of the resonances of table 4.1 was used. This freedom of inserting, excluding, or changing parameters of certain resonances in the toy data generation is one of the advantages of ComPWA.
Amplitude analyses often rely on precise data and high statistics. Although only Monte Carlo tests are presented in this work, considerations of data rates are important to set the systematic studies into relationship. Table 4.2 shows the PDG branching fractions of the considered processes.

Table 4.1: PDG parameters used in the event generation of the toy MC datasets [2].

Final State Particles	Bachelor Particle	Resonance	PDG Mass [MeV/c^2]	PDG Width [MeV]
$\gamma\pi^0$	π^0	$\omega(782)$	782.7	8.5
$\pi^0\pi^0$	γ	$f_0(980)$	980.0	50.0
$\pi^0\pi^0$	γ	$f_2(1270)$	1274	185
$\pi^0\pi^0$	γ	$f_0(1500)$	1505	109
$\pi^0\pi^0$	γ	$f_2'(1525)$	1525	73
$\pi^0\pi^0$	γ	$f_0(1710)$	1720	135

Table 4.2: Branching ratios (B) with $\gamma\pi^0\pi^0$ final state according PDG [2].

J/ψ decay	B $[10^{-4}]$	Subsequent decay	B $[10^{-2}]$	$J/\psi \to \gamma\pi^0\pi^0$ B $[10^{-5}]$
$J/\psi \to \omega\pi^0$	4.5 ± 0.5	$\omega \to \gamma\pi^0$	8.28 ± 0.28	3.7
$J/\psi \to \rho^0\pi^0$	56 ± 7	$\rho^0 \to \gamma\pi^0$	0.060 ± 0.008	0.3
$J/\psi \to \gamma f_0(980)$	-	$f_0(980) \to \pi^0\pi^0$	$\approx 0.3 - 0.7$	-
$J/\psi \to \gamma f_0(1500)$	1.01 ± 0.32	$f_0(1500) \to \pi^0\pi^0$	$\frac{1}{3} \cdot (34.9 \pm 2.3)$	2.3
$J/\psi \to \gamma f_0(1710)$	-	$f_0(1710) \to \pi^0\pi^0$	-	40.0 ± 10.0
$J/\psi \to \gamma f_2(1270)$	1.43 ± 0.11	$f_2(1270) \to \pi^0\pi^0$	$\frac{1}{3} \cdot (84.8^{+2.4}_{-1.2})$	4.1
$J/\psi \to \gamma f_2'(1525)$	$4.5^{+0.7}_{-0.4}$	$f_2'(1525) \to \pi^0\pi^0$	$\frac{1}{3} \cdot (8.2 \pm 2.2)$	1.2
$J/\psi \to \gamma f_4(2050)$	27 ± 7	$f_4(2050) \to \pi^0\pi^0$	$\frac{1}{3} \cdot (17.0 \pm 1.5)$	15.3
$J/\psi \to \gamma f_J(2200)$	-	$f_J(2200) \to \pi^0\pi^0$	-	8 ± 4
$J/\psi \to \gamma\pi^0\pi^0$	-	-	-	≈ 75

With these numbers, one can estimate that a fraction of about 0.1% of the produced J/ψ will decay via the channels of interest. This would mean one billion recorded J/ψ would provide about one million events for this final state of interest, not taking into account the reduced detection efficiency by the event selection and background suppression. The $\rho^0\pi^0$ was not taken into account in these studies up to now, since the combined branching fraction is about ten times smaller than that of $J/\psi \to \omega\pi^0$. In order to show the separation capabilities of the extraction method, the simulation excluded the heavier f resonances and focuses on the lighter f_0 and f_2 resonances.
There is no branching fraction available for $J/\psi \to \gamma f_0(980)$, presumably as the nature of this resonance is not well understood and measurements of its properties rely heavily on the underlying model. Previous $J/\psi \to \gamma\pi^0\pi^0$ amplitude analyses often relied on fitting only data above $m(\pi^0\pi^0) = 1\,\text{GeV}/c^2$, e.g. the analysis shown in figure 4.2, which shows the mass distribution and fit of an analysis performed by the BESIII Collaboration [67]. The plot shows no visible enhancement close to $1\,\text{GeV}/c^2$ where the right tail of the $f_0(980)$ could be visible, but this might

be clearer in an efficiency corrected plot. Although its nature and mixing is poorly understood, a resonance based on the PDG averages of the $f_0(980)$ measurements was still simulated in order to study the sensitivity for it.

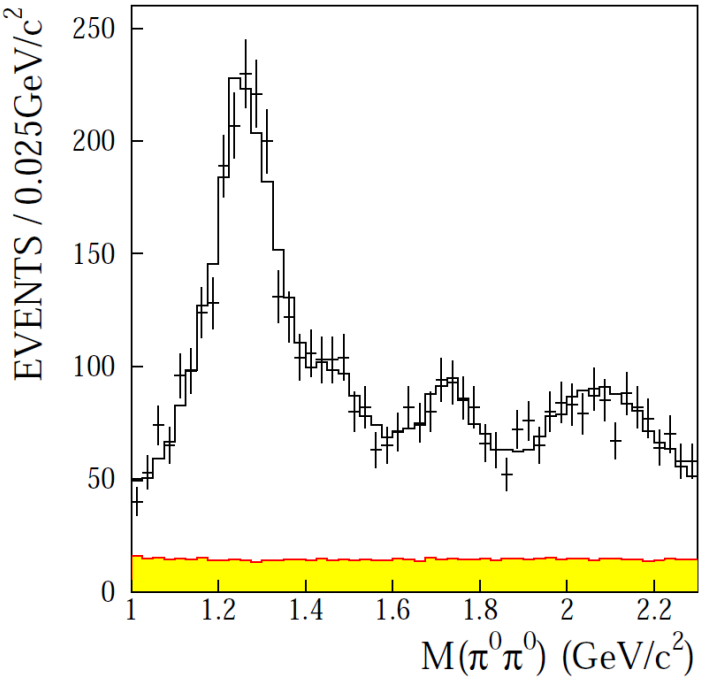

Figure 4.2: The $\pi^0\pi^0$ invariant mass distribution from $J/\psi \to \gamma\pi^0\pi^0$ without efficiency correction, measured at BESIII [67]. The crosses are data, the full histogram shows the maximum likelihood fit to the data, and the shaded histogram corresponds to the background.

4.2 Toy Data

To test the analysis techniques and the framework, toy Monte Carlo data were produced. Since all input properties like contributing resonances, relative strengths and interferences, are exactly known for these kind of data, which allows a proper functionality check of the software. It also allows to vary parameters to perform systematic studies. The toy data used in this work were generated using the Breit-Wigner sum model (formula 3.1). The reaction channel $J/\psi \to \gamma\pi^0\pi^0$ is produced with the J/ψ at its rest mass $m_{J/\psi}$, the π^0 mesons and the photon have the masses m_{π^0} and m_γ:

$$m_{J/\psi} = 3.0969\,\text{GeV}/c^2$$
$$m_\gamma = 0\,\text{GeV}/c^2$$
$$m_{\pi^0} = 0.135\,\text{GeV}/c^2$$

Many characteristics of the analysis method are difficult to observe in the rich environment of the *f* resonances. Therefore, an simplified set of artificial *f* resonances was chosen as a starting point and a cross-check environment for the analysis, e.g. can it be used to understand how much a limiting factor of a method depends on the resonance parameters. Table 4.3 lists the intermediate resonances simulated and their Breit-Wigner parameters.
Figure 4.3 shows the generated data in a Dalitz plot, with three broad *f* resonances and a broad ω. On the $\pi^0\pi^0$ axis below 5 $(\text{GeV}/c^2)^2$ the two scalar resonances and the tensor resonance are visible with their angular distributions parallel to the $\pi^0\gamma$ axis. In addition, two bands of the $\omega(782)$ meson are visible in both $\pi^0\gamma$ combinations. One is close and parallel to the $\pi^0\pi^0$ axis, the other is a diagonal band close to the upper phasespace border in this representation. They are crossing and interfering with the *f* resonances and crossing each other at around 8.5 $(\text{GeV}/c^2)^2$.
To demonstrate the effects of interference between the fictive f_0 and f_2 resonance and the broad ω meson more clearly, the plot on the right shows a zoom into the lower left corner of the Dalitz plot on the left, showing the intersection of the horizontal ω band and the two bands of the f_0 and f_2, respectively. In the lower plots, the same setup is shown with a slight change: the phase of the f_2 resonance has changed from 1.6 rad to 0.0 rad (table 4.3). This causes a change of the interference between the resonances which can be seen when comparing the plots on the right of figure 4.3.

Table 4.3: Resonance parameters for simple dataset generation.

Resonance	Mass [GeV/c^2]	Width [GeV]	Magnitude	Phase [rad]
$f_0(A)$	1.0	0.2	1.0	0.0
$f_0(B)$	2.1	0.2	1.0	0.0
f_2	1.5	0.2	1.0	1.6 (0.0)
$\omega(782)$	0.783	0.05	1.0	0.8

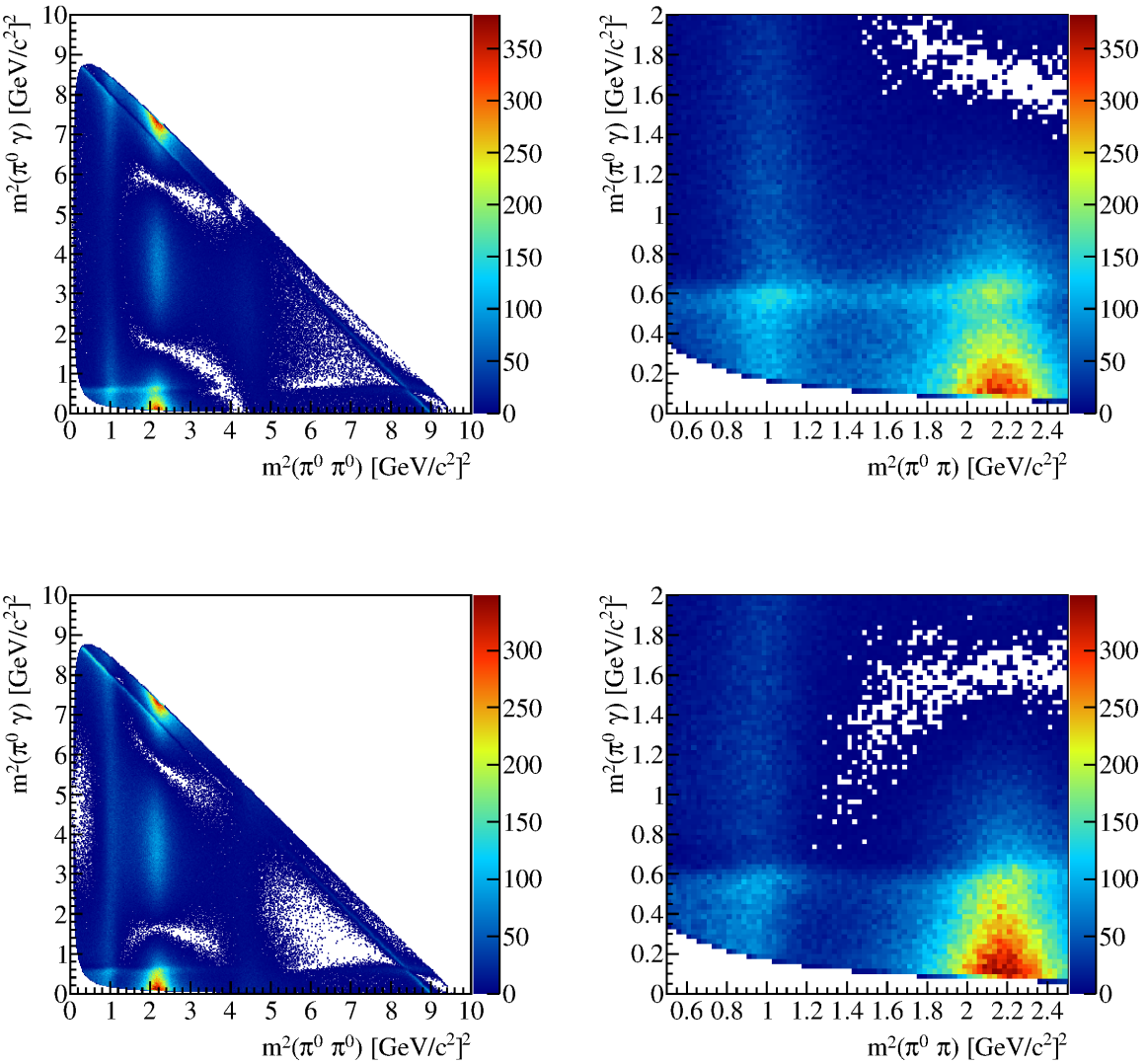

Figure 4.3: Dalitz plot of 10^6 $J/\psi \to \gamma\pi^0\pi^0$ events.
Upper Left: Distributed according to Breit-Wigner sum model with the parameters from table 4.3.
Lower Left: As above but with alternative phase of f_2 shown in table 4.3.
Right: Crop of the lower left region.

For the more realistic scenario, table 4.4 shows the intermediate resonances simulated and their Breit-Wigner parameters. The parameters are realistic assumptions for the masses and widths of the resonances, but they are not normalized nor weighted according to their branching fractions and the phases are set to arbitrary values.

Table 4.4: Resonance parameters for Monte Carlo data generation.

Resonance	Mass [GeV/c^2]	Width [GeV]	Magnitude	Phase [rad]
$f_0(980)$	0.99	0.05	1.0	0.0
$f_0(1500)$	1.505	0.109	1.0	0.0
$f_0(1710)$	1.720	0.135	1.0	0.0
$f_2(1270)$	1.274	0.185	1.0	0.0
$f_2'(1525)$	1.525	0.073	1.0	0.0
$\omega(782)$	0.783	0.0085	1.0	0.0

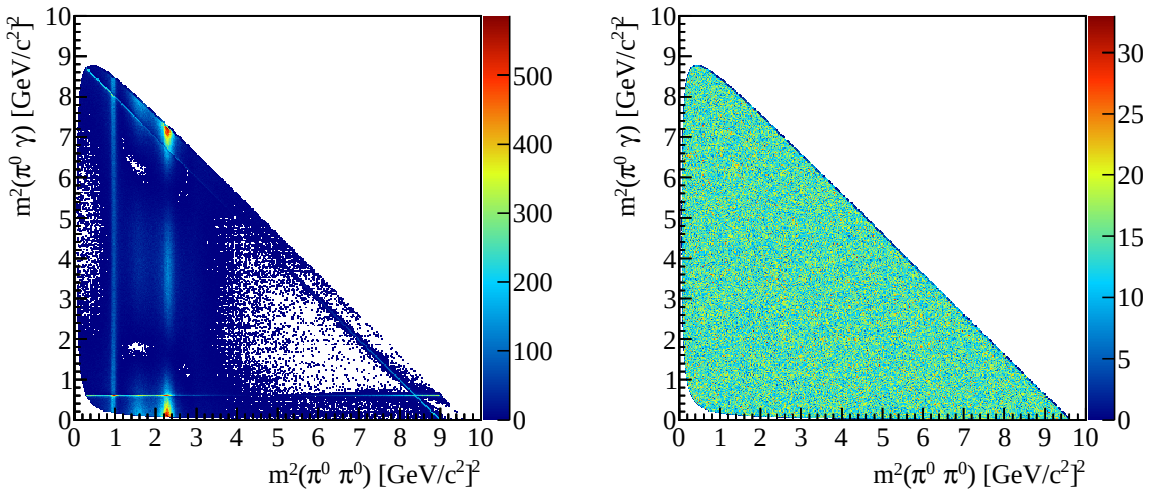

Figure 4.4: Dalitz plots of 10^6 $J/\psi \to \gamma\pi^0\pi^0$ toy data events.
Left: Distributed according to Breit-Wigner sum model and parameters from table 4.4.
Right: Phasespace distributed.

Figure 4.4 shows the Dalitz plots of the datasets generated with the Breit-Wigner sum model on the left and using phasespace distribution on the right. Due to the distinct angular distribution, the two f_2 resonances clearly stand out, as well as the $f_0(980)$, appearing as a narrow vertical band at 1 $(\mathrm{GeV}/c^2)^2$ due to its small width. The $f_0(1500)$ and $f_0(1710)$ are barely visible due to the overlap with the f_2 resonances. The right tail of the $f_0(1710)$ can be noticed, as the region above 3 $(\mathrm{GeV}/c^2)^2$ is rather homogeneously populated along the $\pi^0\gamma$ axis. Again, the two bands of the $\omega(782)$ meson are visible in both $\pi^0\gamma$ combinations.
The plot on the right of figure 4.4 shows the Dalitz plot of the toy Monte Carlo data which were generated without any intermediate resonances of the final state particles. It is uniformly populated, as can be expected of the phasespace distributed Monte Carlo data if no detector efficiency and acceptance is simulated. The blue colored bins at the upper right edge of the phasespace border have less entries due to the chosen binning of the histogram. These bins cover areas which are partly kinematically not allowed, thus they have less events. The phasespace distributed data are used for the unbinned logarithmic likelihood fits, as this method relies on a normalization of the likelihood function being calculated via a Monte Carlo integration.

4.3 Dalitz Plot Fit

The model-dependent amplitude analysis of the full phasespace using an unbinned logarithmic likelihood estimator is called Dalitz plot fit. As the same model is used for the generation of the toy data and for the fit, this analysis works as a measure how precise the input model can be extracted within the limits of statistical effects.
For this test, the complete set of resonances (table 4.4) was chosen. As described in section 3.8 (formula 3.1), the amplitude model is:

$$|A|^2 = |\sum_i c_i T_i D_i|^2 \quad \text{with } c = r \cdot e^{i\phi},\ i = \omega, f_0(980), ... \tag{4.1}$$

Table 4.5: Parameter settings used for generation, extracted by the Dalitz plot fit to the toy data, and pull value (difference/error). m: mass of resonance in GeV/c^2, Γ: width of resonance in GeV, r: magnitude relative to r_ω and ϕ: phase relative to ϕ_ω in rad.

Resonance		Generated	Fit Result	Pull
$f_0(980)$	m	0.99	0.9899 ± 0.0001	1.00
	Γ	0.05	0.0501 ± 0.0002	0.50
	r	1.0	1.0056 ± 0.0031	1.81
	ϕ	0.0	0.0096 ± 0.0065	1.48
$f_0(1500)$	m	1.505	1.5037 ± 0.0005	2.60
	Γ	0.109	0.1116 ± 0.0011	2.36
	r	1.0	1.0323 ± 0.0108	2.99
	ϕ	0.0	-0.0229 ± 0.0093	2.46
$f_0(1710)$	m	1.720	1.7200 ± 0.0008	0.00
	Γ	0.135	0.1334 ± 0.0015	1.07
	r	1.0	0.9821 ± 0.0133	1.35
	ϕ	0.0	0.0035 ± 0.0133	0.26
$f_2(1270)$	m	1.274	1.2738 ± 0.0004	0.50
	Γ	0.185	0.1838 ± 0.0009	1.33
	r	1.0	1.0043 ± 0.0040	1.08
	ϕ	0.0	0.0152 ± 0.0065	2.34
$f_2'(1525)$	m	1.525	1.5247 ± 0.0001	3.00
	Γ	0.073	0.0727 ± 0.0002	1.50
	r	1.0	0.9949 ± 0.0029	1.76
	ϕ	1.6	1.5986 ± 0.0060	0.23
$\omega(782)$	m	0.786	fixed	
	Γ	0.0085	fixed	
	r_ω	1.0	fixed	
	ϕ_ω	0.0	fixed	

All masses, widths, magnitudes and phases of the five f resonances in the $\pi^0\pi^0$ system were allowed to vary during the fit. All four parameters of the ω resonance were fixed, as the mass and width of the ω meson is measured very precisely [2]. The fixed magnitude and fixed phase of the ω resonance is needed as reference for all other magnitudes and phases because the description of the amplitude is only sensitive to the differences of these values. The starting values for all parameters were arbitrary displaced to 105% of their values used for generating the toy data. For the optimization Minuit2 was chosen. Table 4.5 shows the result of the fit including the uncertainties estimated by Minuit2 and the pull value, which is the deviation of the fit result divided by the statistical error. Pull values above ≈ 3 are very unlikely in ideal cases

since 99, 7% of the results are lying within $3 \cdot error$ if the statistical error represents 1 σ of a normal distribution. Otherwise, the pull indicates problems with the error estimation or systematic problems. Complete pull distributions are discussed in 4.5.
The fit result is very close to the generated values and is compatible with it within three sigma of the statistical uncertainties. Due to the high number of free parameters and therefore a high dimensional parameter space, correlations between parameters occur and are discussed later.
The plot on the left side of figure 4.5 shows a dataset generated using the parameters of the fit results as model parameters. The plot on the right shows the ratio of the toy data generated with the fit result to these generated with the original parameters (table 4.4 and figure 4.4 left). Two identical histograms would result in a uniform ratio plot with all bins set to 1. A deviation by factor 2 at a certain bin position would result in a value of 0.5 or 2 in the ratio. The ratio plot in figure 4.5 shows that the deviations between the two plots are very small, hence the unbinned likelihood fit is compatible with original parameter set very well. One exception is the region of the ω resonance: The small width makes it difficult to fit as can be seen in the right part ratio plot (red bins).

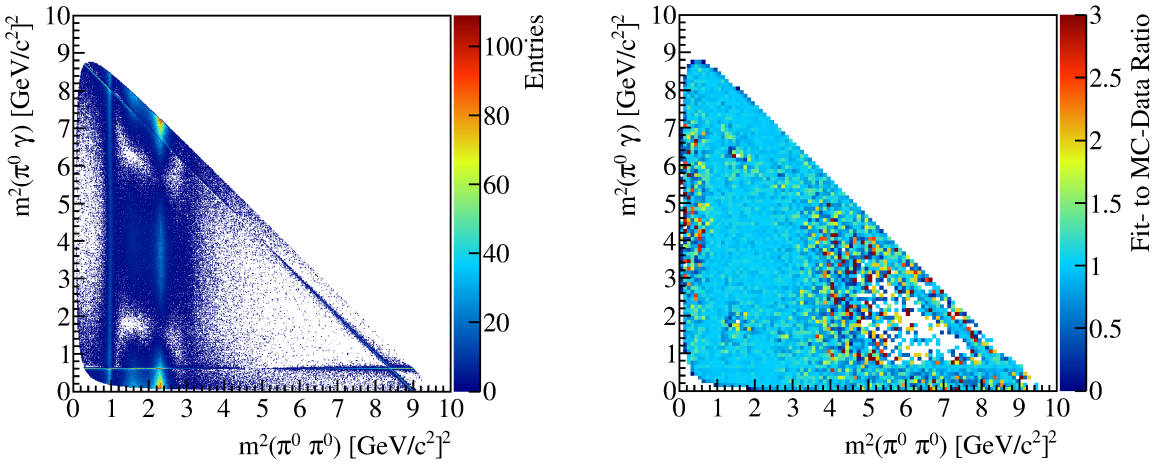

Figure 4.5: Left: Dalitz plot of 10^6 $J/\psi \to \gamma\pi^0\pi^0$ toy MC data events generated by using the parameters of the Dalitz plot fit result in table 4.5.
Right: Ratio of the Dalitz plot of figure 4.3 and the Dalitz plot on the left side.

4.4 Model Independent Fit (Slice Fit)

The second method implemented in ComPWA is a model-independent fit also called *Slice Fit*. It involves independent fits of ”slices” in a Dalitz plot: In figure 4.6 one can see the Dalitz plot of the same toy data as used for the Dalitz plot fit with some illustrated slices along the $\pi^0\pi^0$ axis. One slice corresponds to a one dimensional histogram, as plotted in figure 4.7 and 4.8, where the spin of the dominant resonance can be seen directly.
In the light quark sector, the scalar resonances are not well determined in terms of masses, widths and shape. This procedure allows to independently extract the contribution of the scalar wave to this final states in each bin without assuming a certain model. The scalar resonances decay into the $\pi^0\pi^0$ final state, therefore the slices are chosen along the axis of the invariant $\pi^0\pi^0$ mass.

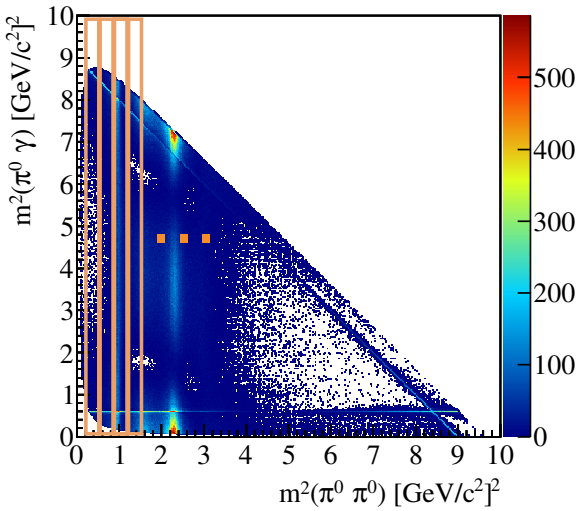

Figure 4.6: Dalitz plot of 10^6 $J/\psi \to \gamma\pi^0\pi^0$ toy MC data events with parameters from table 4.4. The slicing along the $\pi^0\pi^0$ axis is illustrated with orange boxes.

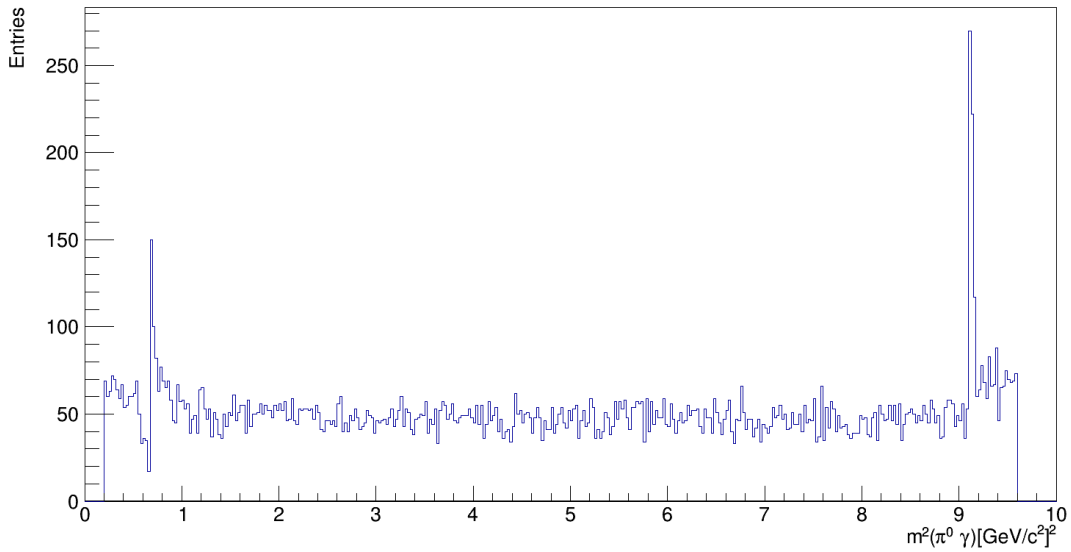

Figure 4.7: Slice in $f_0(980)$ region of Dalitz plot 4.6.

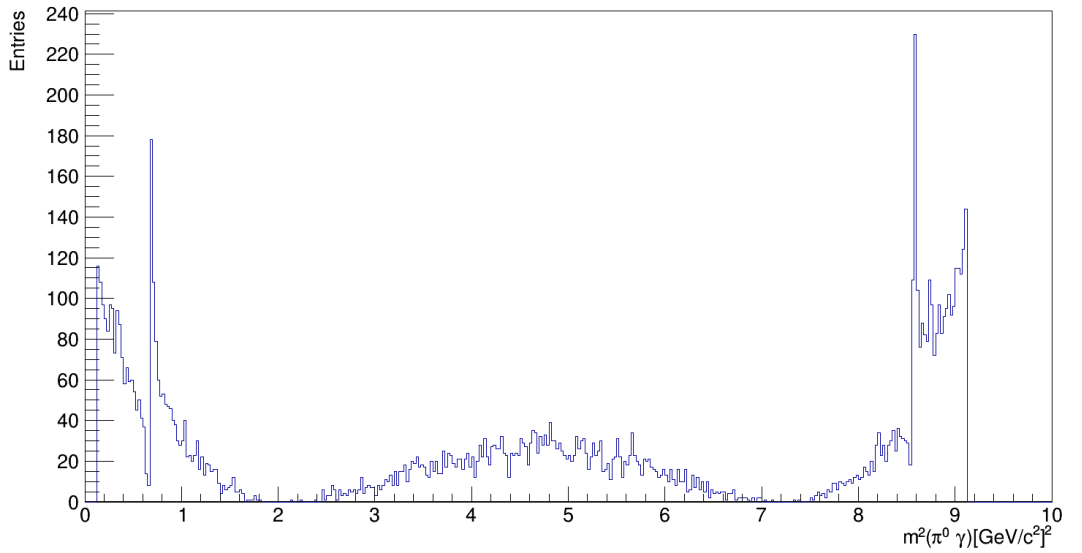

Figure 4.8: Slice in $f_2(1270)$ region of Dalitz plot 4.6.

As one can see in the distributions of the two slices (figure 4.7 and 4.8), the spin of the respective resonance determines the distribution of the decay angle of the final state particles and therefore the structure seen in the particular slice. In addition, there are two dips and peaks visible close to both phasespace limits caused by the two bands of the crossing ω resonance. As one slice corresponds to a fixed position in the $\pi^0\pi^0$ invariant mass domain, the shape of the $\pi^0\pi^0$ system itself is not subject of the fit to a single slice. Each resonance is only reflected in the contribution of each wave and the scaling of the slice. This effect requires to fit a modified amplitude model, where the dynamical behavior of the f resonances perpendicular to the direction of the slices is not fixed to a certain model like Breit-Wigner functions. Therefore only one term for all resonances with the same spin decaying to $\pi^0\pi^0$ is sufficient:

$$I = |T_\omega p_\omega D_\omega + c_0 D_0 + c_2 D_2|^2 \tag{4.2}$$

Compared to equation 4.1 of the Dalitz plot fit, only the spin-dependent information of the f resonances is present in the formula. In addition, a model for resonances decaying to other final states is still required, which in this case means a Breit-Wigner function T_ω for the ω resonance is needed. The complex parameters c_0 and c_2 are determined by the fit and describe the magnitude and the phase of the spin 0 and spin 2 resonances, respectively, while the magnitude and phase of the ω model were fixed. Since the resonances were simulated using Breit-Wigner functions, c_0 and c_2 represent the weighted sum of the Breit-Wigner functions of resonances with the respective spin at the slice position in the $\pi^0\pi^0$ invariant mass.

4.4.1 Slice Fit with Binned Likelihood

Simple Setup

To test the performance of the Slice Fit the data with the simple set of resonances, see figure 4.3, is a reasonable choice. The results of the binned likelihood fit of each slice as a function of $m^2(\pi^0\pi^0)$ is shown in figure 4.9. The fit results resemble the dynamical behavior of the resonances, only separated by spins, with the Breit-Wigner shapes of the resonances clearly visible. Although not perfectly separated, the extracted phases do show some phase motion at the position of the resonances.
Figure 4.9 also shows the result of the second step of this method, the fitting of the resonance model to the extracted waves. For a comparison of this method with the previously described Dalitz plot fit, the shape of the resonances needs to be modeled. This was done by fitting a sum of Breit-Wigner functions T (equation 4.3) for the f_0 resonances and a single Breit-Wigner functions (equation 4.4) for the f_2 resonance to the respective extracted parameters in figure 4.9.

$$F_0(m^2_{\pi^0\pi^0}) = T_{0A}(m^2_{\pi^0\pi^0}) + e^{i\phi_{0B}} \cdot T_{0B}(m^2_{\pi^0\pi^0}) \tag{4.3}$$

$$F_2(m^2_{\pi^0\pi^0}) = T_2(m^2_{\pi^0\pi^0}) \tag{4.4}$$

The fit results of the second step are shown as blue lines in figure 4.9 and the parameters after the fit are listed in table 4.6. The red dashed line shows the fit functions 4.3 and 4.4 with the parameter values used for the generation with the complete amplitude model. The fit was performed on the extracted magnitudes only by using the magnitude of the Breit-Wigner sum as the fit function. This was done because of the problems with the extracted phases which are discussed later. To illustrate the result, the phase shown in figure 4.9 is calculated with the Breit-Wigner sum using the parameter after the magnitude fit. The phase of the spin 0 wave (lower left plot of figure 4.9) shows good agreement with the Breit-Wigner sum fit on the rising slope of $f_0(A)$ and at the $f_0(B)$ resonance. In the center of $f_0(A)$ the phase is more chaotic and not described by the Breit-Wigner sum. Here, the separation of the phases was not working since structure of the $f_0(A)$ is visible in the phase of the spin 2 wave.
Repeating the study with more events and more bins yields smaller statistic errors but does not show clearer separation of the phases, see table A.2 and figure A.8 in appendix A.2. Since this points to a systematic problem with determining the relative phases, an additional test was performed where only the phase difference between the f resonances was fitted instead of relative phases to the ω resonance. Table A.3 and figure A.9 show that a relative phase between the scalar and tensor resonances is not sufficient even in case of broad resonances, thus the ω is used as the phase reference in the following. A simulation with a more dominant and even broader ω resonance ($\Gamma = 0.2\,\text{GeV}$), see table A.4 and figure A.10, gives an improvement in the extraction of the phases. Thus, it can be concluded that the method is working but is limited depending on the characteristics of the decay channel analyzed.

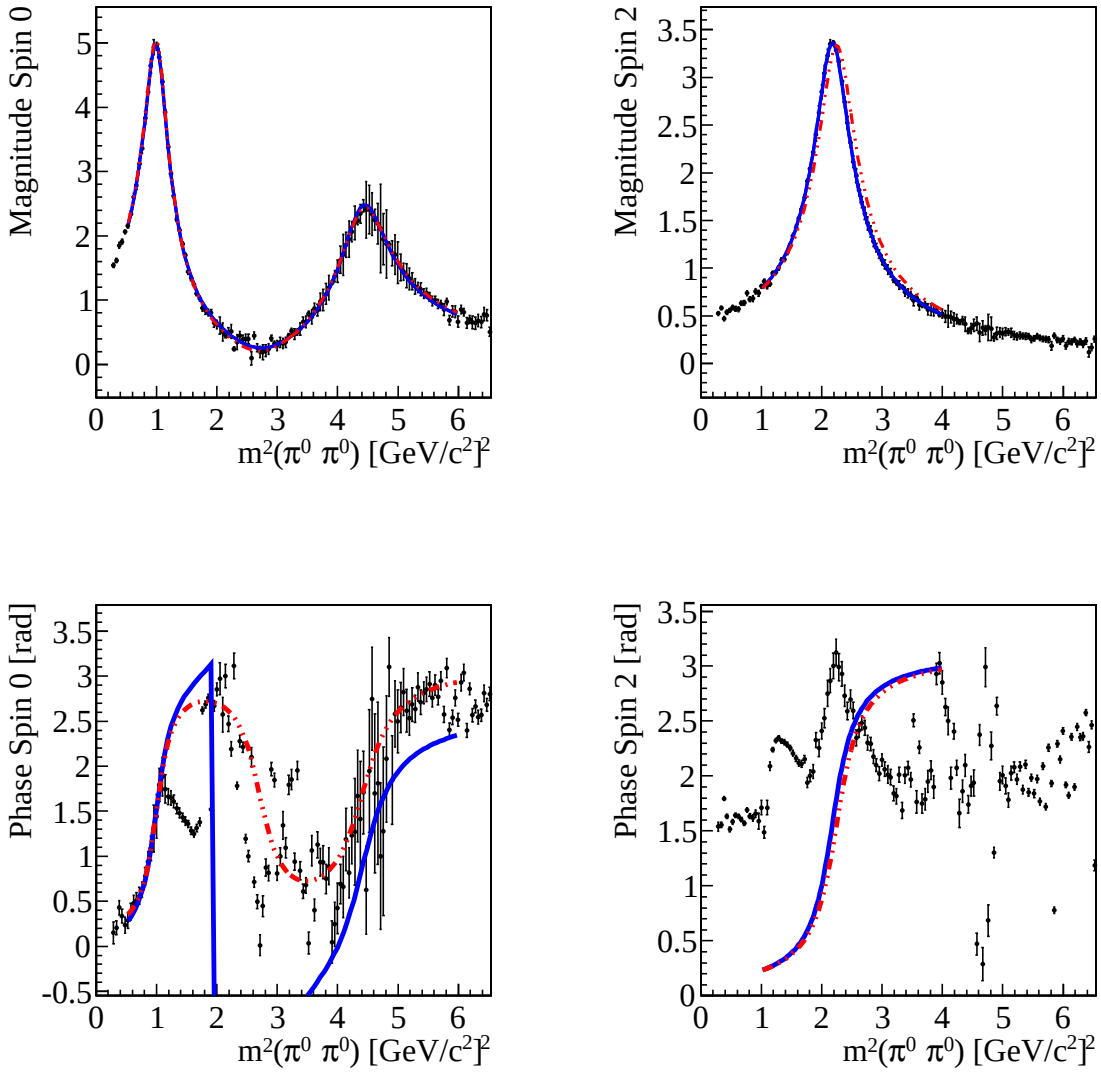

Figure 4.9: Mass dependent fit coefficients c_0 and c_2 (top: magnitudes; bottom: phases) from equation 4.2 as the result of the Slice Fit of the simple toy data. Also, the Breit-Wigner fit result (blue line), see table 4.6, and the Breit-Wigner sum with the ideal parameter setting (red dashed line) are plotted.

Table 4.6: Parameter settings used for generation and extracted by the Slice Fit of the simple toy data. m: mass of resonance in GeV/c^2, Γ: width of resonance in GeV.

Resonance		Generated	Fit Result	Pull
$f_0(A)$	m	1.0	0.9991 ± 0.0009	1.00
	Γ	0.2	0.2012 ± 0.0024	0.50
$f_0(B)$	m	2.1	2.1003 ± 0.0023	0.15
	Γ	0.2	0.1860 ± 0.0036	3.89
	ϕ	0.0	-0.7778 ± 0.0236	32.96
f_2	m	1.5	1.4761 ± 0.0008	29.88
	Γ	0.2	0.1910 ± 0.0012	7.50
ω	m	0.786	fixed	
	Γ	0.05	fixed	

Realistic Setup

The Slice Fit of the realistic setup was performed by using the same settings, binning, and estimation functions as for the previously discussed fit to the simple setup. The results of the binned likelihood fit of each slice as a function of $m^2(\pi^0\pi^0)$ is shown in figure 4.10. The shape of the three scalar and the two tensor resonances is clearly visible, but the phases show much more fluctuations.

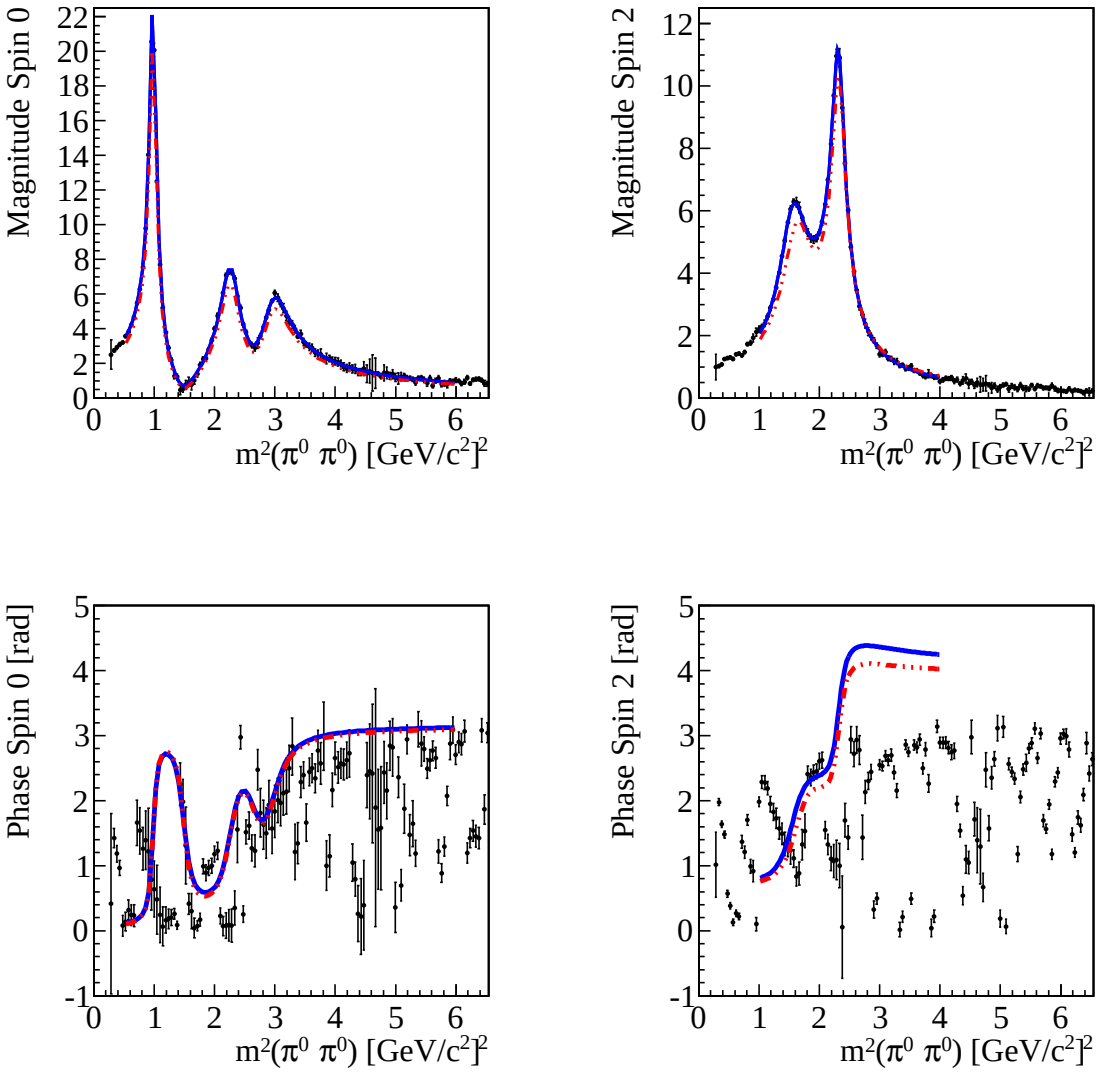

Figure 4.10: Mass dependent fit coefficients c_0 and c_2 (top: magnitudes; bottom: phases) from equation 4.2 as the result of the Slice Fit. Also, the Breit-Wigner fit result (blue line), see table 4.7, and the Breit-Wigner sum with the ideal parameter setting (red dashed line) are plotted.

Figure 4.10 also shows the fit of the resonance model to the magnitudes after separation as blue lines. For the realistic scenario, this was done by fitting a sum of three Breit-Wigner functions T (equation 4.5) for the f_0 resonances and a sum of two Breit-Wigner functions (equation 4.6) for the f_2 resonances to the respective extracted parameters in figure 4.10. The red dashed line shows the fit functions 4.5 and 4.6 with the parameter values used for the generation with the complete amplitude model.

$$F_0(m^2_{\pi^0\pi^0}) = T_{908}(m^2_{\pi^0\pi^0}) + e^{i\phi_{1500}} \cdot T_{1500}(m^2_{\pi^0\pi^0}) + e^{i\phi_{1710}} \cdot T_{1710}(m^2_{\pi^0\pi^0}) \quad (4.5)$$

$$F_2(m^2_{\pi^0\pi^0}) = T_{1270}(m^2_{\pi^0\pi^0}) + e^{i\phi_{1525}} \cdot T_{1525}(m^2_{\pi^0\pi^0}) \quad (4.6)$$

The results of the two separate fit procedures, Slice Fit and Breit-Wigner sum fit, are shown in table 4.7. The statistical errors are bigger than the errors of the Dalitz plot fit (table 4.5) which is expected due to binning and the two-stage approach, but the separation of the waves works well and the extracted parameters deviate from the simulated values by maximum of a few percent.

Table 4.7: Parameter settings used for generation and extracted by the Slice Fit of the toy data. m: mass of resonance in GeV/c^2, Γ: width of resonance in GeV.

Resonance		Generated	Fit Result	Pull
$f_0(980)$	m	0.99	0.9902 ± 0.0005	0.40
	Γ	0.05	0.0513 ± 0.0010	1.30
$f_0(1500)$	m	1.505	1.5052 ± 0.0011	0.18
	Γ	0.109	0.1088 ± 0.0013	0.15
	ϕ	0.0	0.0298 ± 0.0441	0.68
$f_0(1710)$	m	1.720	1.7199 ± 0.0020	0.05
	Γ	0.135	0.1363 ± 0.0022	0.59
	ϕ	0.0	0.0534 ± 0.0435	1.23
$f_2(1270)$	m	1.274	1.2478 ± 0.0009	29.11
	Γ	0.185	0.1825 ± 0.0020	1.25
$f_2'(1525)$	m	1.525	1.5230 ± 0.0004	5.00
	Γ	0.073	0.0731 ± 0.0010	0.10
	ϕ	1.6	1.8670 ± 0.0271	9.85
$\omega(782)$	m	0.786	fixed	
	Γ	0.0085	fixed	

With the lack of an alternative to provide a reference for the phases, this limiting factor can only be addressed by a high number of events and a good reconstruction precision, as can be seen in section 4.5. This will allow the fine binning needed to describe the interference of the small ω with the resonances of interest. As this approach is also limited, an alternative method for performing the Slice Fit unbinned is presented next to test it's sensitivity for the small interference effects.

4.4.2 Slice Fit with Unbinned Likelihood

The loss of information by quantization of the measured information can cause problems which can lead to the so-called binning effects. They are often visible at phasespace borders where bins are only partly covered by the accessible phasespace. The bin content then represents the fraction of it which lies inside the phasespace, but the behaviour of the phasespace border is lost. When comparing the bin content to a model, the trivial approach would be to evaluate the model in the center of the bin. In a bin at the phasespace border, this either leads to a value of zero when the center lies outside of the phasespace or a model value too high when it is inside. Either way, the bin content lies somewhere in between and is not properly described. This can also happen in other, less obvious cases, e.g. in regions featuring steep intensity gradients.
There are several ways to address binning effects: smaller bins, adaptive binning at borders, evaluating the model on several positions instead of evaluating it at only one point, integrating the model over the bin size, etc. All of them reduce the binning effects in exchange for more calculation time. For the Slice Fit, which is inherently binned, it is still possible to perform a part unbinned to see if binning has a significant effect on the accuracy of the fit results. To accommodate that, an unbinned likelihood fit was performed for each individual slice. Since the second step, i.e. the fit to distribution gained by the Slice Fit, is inherently binned by the slicing technique itself, there is no change. However, if the unbinned approach of the first step yields better results and error estimates, the second step should noticeably profit as well.

Simple Setup

Again, the Slice Fit was performed on the data of the simple setup, see table 4.3, first in order to get a starting point for this method. The fit is identical to the previously performed binned fit except for the use of an unbinned log likelihood estimator for the first step. Figure 4.11 shows the fit result of the Slice Fit as data points. In the second step, the magnitude of a Breit-Wigner sum was fitted to the extracted spin 0 magnitude values and the magnitude of a single Breit-Wigner function to the extracted spin 2 values. Phase and magnitude of the respective fit functions with the parameters after the fit, listed in table 4.8, are shown in figure 4.11 as blue lines, the same function with the parameter values used for the generation are shown as red dashed lines.
In the binned case, the simple setup showed a very clean extraction of the magnitude and phase values with only some mixing effects in the phases. In the unbinned case, outliers and big uncertainties at higher energies are present already in the simple setup, indicating less stability of the fits. As the $f_0(B)$ is partly in this region (4 to 5 $(\mathrm{GeV}/c^2)^2$ in the left plots of figure 4.11) a lack of data should not be the reason, but the slices are short. Also, there appears to be some jumps between the points from slice to slice which is bigger than the uncertainties and seems to be too regular for being a random effect. Still, the magnitudes are reproduced quite good and the Breit-Wigner sum phases show a good qualitative agreement with the extracted values, except for the region around 1.5 $(\mathrm{GeV}/c^2)^2$, which shows the same effects as the binned approach, and an $\approx \pi/2$ offset in the spin 2 resonance phase.
The parameters of the Breit-Wigner sum fit (listed in table 4.8) show that the problems at high energies cause the $f_0(B)$ parameters to be significantly off the ideal values. Also, the statistical uncertainties are bigger than in the binned alternative case (listed in table 4.6) due to the bad approximation of the data with the extraction model which is discussed later, see figure 4.13.
However, the extraction of the phase motion of the f_2 looks more reasonable in case of the unbinned Slice Fit than in the binned case shown in figure 4.9. In unclear cases, the phase motion can help identify resonances. Therefore, albeit quantitatively worse than the binned Slice Fit, the better extraction of the phases with the unbinned Slice Fit can be useful as a first look to a new channel or a cross-check for the resonance models.

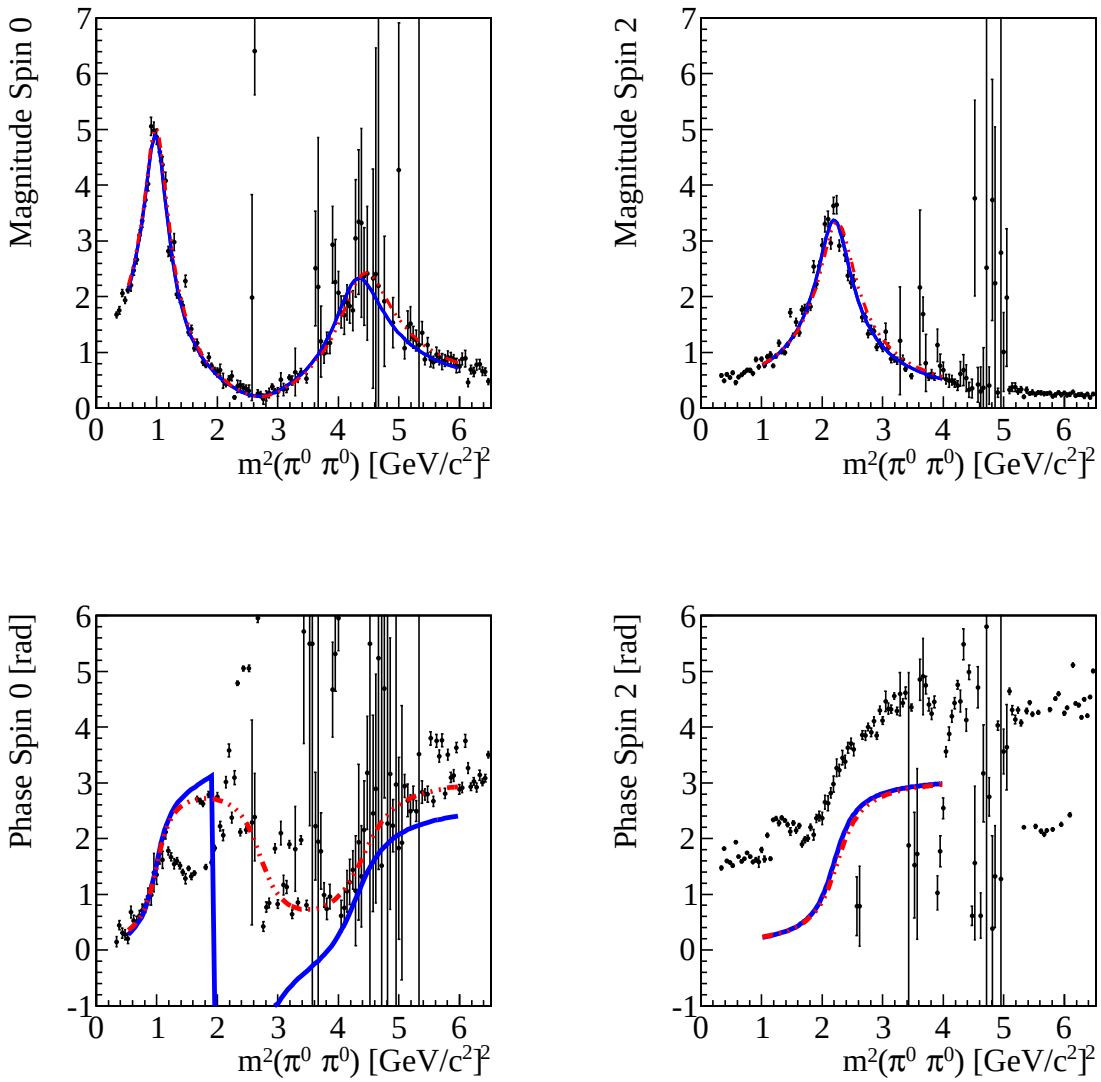

Figure 4.11: Mass dependent fit coefficients c_0 and c_2 (top: magnitudes; bottom: phases) from equation 4.2 as the result of the unbinned Slice Fit of the simple toy data. Also, the Breit-Wigner fit result (blue line), see table 4.8, and the Breit-Wigner sum with the ideal parameter setting (red dashed line) are plotted.

Table 4.8: Parameter settings used for generation and extracted by the unbinned Slice Fit of the simple toy data. m: mass of resonance in GeV/c^2, Γ: width of resonance in GeV.

Parameter		Generated	Fit Result	Pull
$f_0(A)$	m	1.	0.9962 ± 0.0013	2.92
	Γ	0.2	0.1977 ± 0.0046	0.50
$f_0(B)$	m	2.1	2.0729 ± 0.0079	3.43
	Γ	0.2	0.1931 ± 0.0148	0.47
	ϕ	0.0	-0.7246 ± 0.0304	24.30
f_2	m	1.5	1.4823 ± 0.0017	10.41
	Γ	0.2	0.1884 ± 0.0037	3.14
ω	m	0.786	fixed	
	Γ	0.05	fixed	

Realistic Setup

The configuration of this analysis is identical to the binned case, except for the use of an unbinned log likelihood estimator for the first step. Figure 4.12 shows the fit result of the Slice Fit as points and the Breit-Wigner sum fit as blue lines, the same function with the parameter values used for the generation are shown as red dashed lines. Only the magnitudes of the Breit-Wigner sums were fitted to the extracted magnitudes, like in the binned fit, but the resulting phases not used in the fit are shown for illustration as well.
As in the simple setup, the outliers and high statistical uncertainties at higher energies are present. The resulting Breit-Wigner sum after the fits to the extracted magnitudes (upper plots of figure 4.12) show a good agreement with the data. Outliers and larger uncertainties of the extracted values are visible above 3 $(\,\mathrm{GeV}/c^2)^2$ which match with similar effects in the extracted phases. At high energies, the slices are shorter and fewer events are present, as the toy Monte Carlo model solely consists of the lighter *f* resonances and the narrow ω resonance. Thus, the large uncertainties and the outliers are most likely caused by ambiguous solutions and too little data to support a certain solution. However, the extracted phases (lower plots of figure 4.12) seem to be extracted slightly better and show more agreement with the phases of the Breit-Wigner sum fits. As the phases are extracted relative to the narrow ω, whose width is about the size of 3 bins in the respective binned Slice Fit, the unbinned Slice Fit has more information and can fit the interference better since small changes in the intensity are not lost to the binning.
Table 4.9 lists the result of the model dependent fits of the equations 4.5 and 4.6 to the extracted waves. The unbinned fit results show a good agreement with the binned results although the parameters after fit show higher statistical uncertainties, see table 4.11 in 4.4.3. This can be expected, since the model for the unbinned Slice Fit is not ideal: Due to the approximation of the behavior of the *f* resonances with a certain spin with a single complex number in each slice the model is not describing the data along the width of a slice correctly. Figure 4.13 illustrates this effect. Albeit the unbinned fit should handle effects along the slice better, e.g. the interference with the ω meson, it can not describe the data along the $m^2(\pi^0\pi^0)$ axis perfectly within a slice due to the reduced model which leads to higher statistical uncertainties when performing an unbinned fit. In the binned case, this doesn't have an effect since the binning already reduces the information of the single events down to a level which suits the approximation.

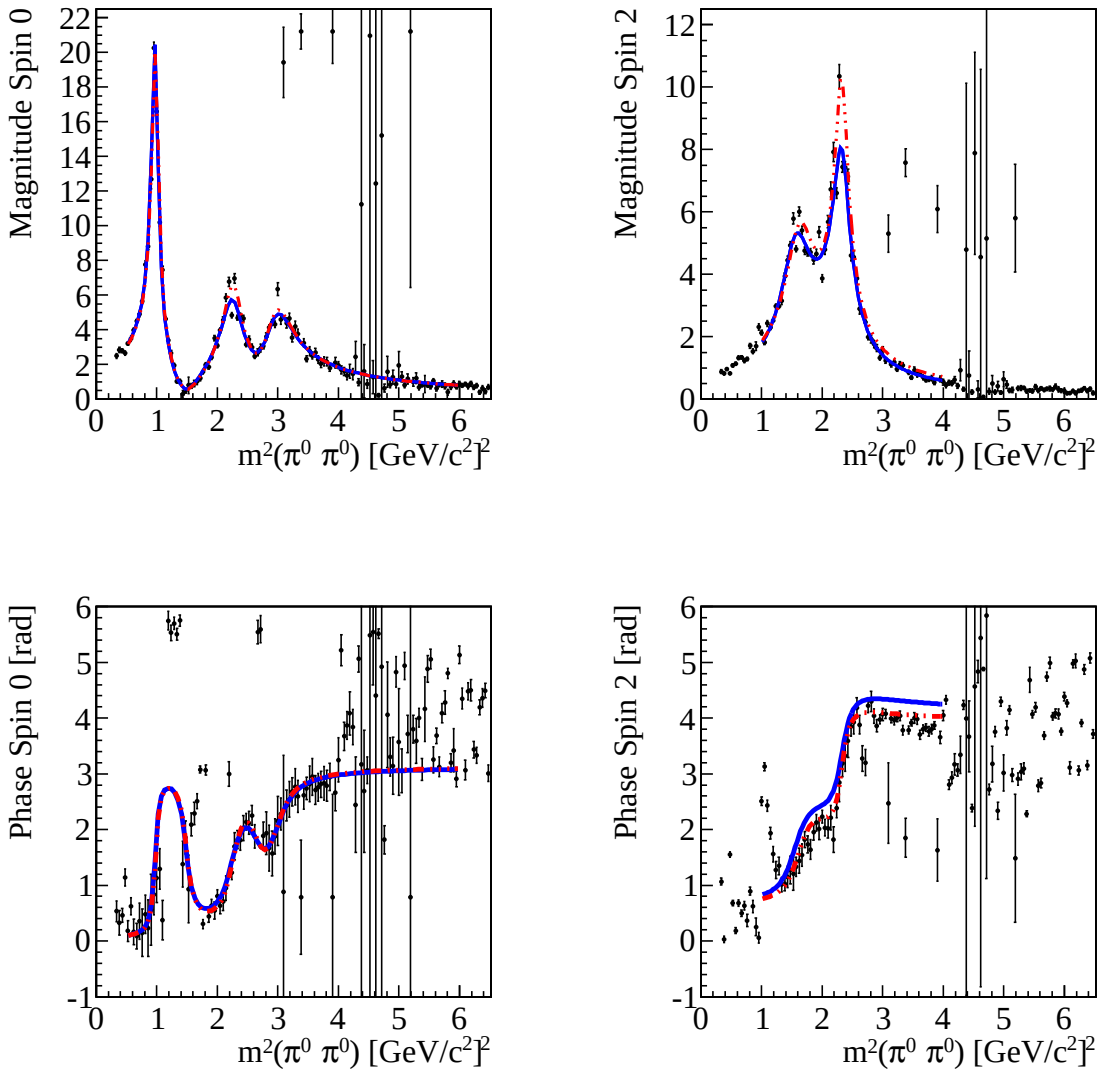

Figure 4.12: Mass dependent fit coefficients c_0 and c_2 (top: magnitudes; bottom: phases) from equation 4.2 as the result of the unbinned Slice Fit. Also, the Breit-Wigner fit result (blue line), see table 4.9, and the Breit-Wigner sum with the ideal parameter setting (red dashed line) are plotted.

Table 4.9: Parameter settings used for generation and extracted by the unbinned Slice Fit of the toy data. m: mass of resonance in GeV/c^2, Γ: width of resonance in GeV.

Parameter		Generated	Fit Result	Pull
$f_0(980)$	m	0.99	0.9863 ± 0.0005	7.40
	Γ	0.05	0.0502 ± 0.0010	0.20
$f_0(1500)$	m	1.505	1.4997 ± 0.0021	2.52
	Γ	0.109	0.1278 ± 0.0026	7.23
	ϕ	0.0	-0.0222 ± 0.0484	0.46
$f_0(1710)$	m	1.720	1.7210 ± 0.0039	0.26
	Γ	0.135	0.1441 ± 0.0043	2.12
	ϕ	0.0	-0.0001 ± 0.0557	0.00
$f_2(1270)$	m	1.274	1.2474 ± 0.0020	13.30
	Γ	0.185	0.1925 ± 0.0043	1.74
$f_2(1525)$	m	1.525	1.5254 ± 0.0014	0.29
	Γ	0.073	0.0926 ± 0.0020	9.80
	ϕ	1.6	1.8840 ± 0.0478	5.94
$\omega(782)$	m	0.786	fixed	
	Γ	0.0085	fixed	

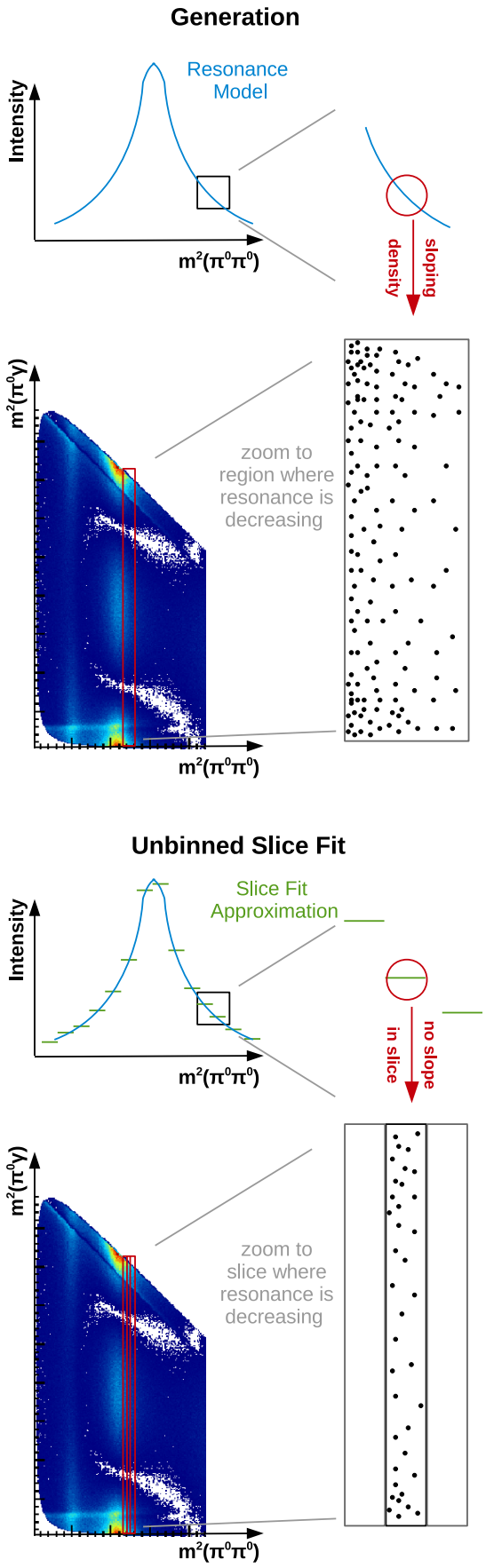

Figure 4.13: Illustration of the approximation introduced by the Slice Fit model. The rise in intensity introduced by a resonance model (blue) along the sliced axis is approximated by a constant within a slice (green), so in the unbinned fit the data is not perfectly described.

Complex Fit

With the better extraction of the phase motion, the unbinned Slice Fit allows to also perform a combined fit of the models magnitude and phase to the extracted magnitudes and phases of the waves. In the previously discussed binned and unbinned scenarios, only the extracted magnitudes were fitted since the phases were not separated clearly. Figure 4.14 shows that by trying to match the extracted phases as well as the magnitudes more deviations from the magnitudes are introduced, e.g. in case of the $f_2(1270)$ which is barely visible in the fit (blue line) compared to the fit function with the parameter values from the generation (red dashed lines). The results listed in table 4.10 confirm that the complex fit yields bigger discrepancies to the true values while overestimating the errors, since it operates on twice as much data points, which leads to the high pull values. Remarkably, the phase ϕ of the $f_2'(1525)$ is better reconstructed than in all previous scenarios. Nevertheless, the complex Breit-Wigner sum fit is not improving the analysis therefore only the magnitudes are fitted for now.

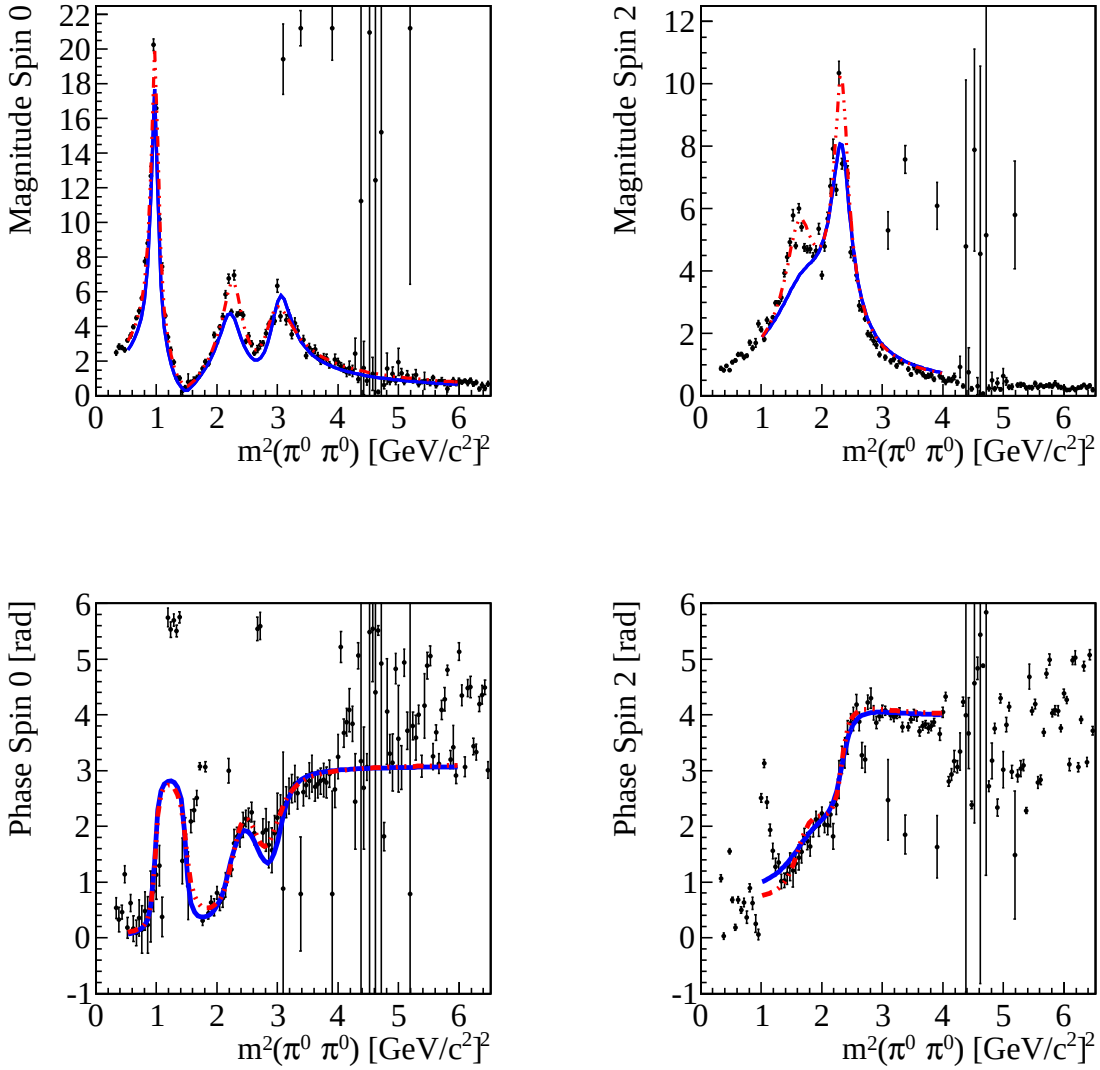

Figure 4.14: Mass dependent fit coefficients c_0 and c_2 (top: magnitudes; bottom: phases) from equation 4.2 as the result of the unbinned Slice Fit. Also, the complex Breit-Wigner fit result (blue line), see table 4.10, and the Breit-Wigner sum with the ideal parameter setting (red dashed line) are plotted.

Table 4.10: Parameter settings used for generation and extracted by the complex unbinned Slice Fit of the toy data. m: mass of resonance in GeV/c^2, Γ: width of resonance in GeV.

Parameter		Generated	Fit Result	Pull
$f_0(980)$	m	0.99	0.9875 ± 0.0001	25.00
	Γ	0.05	0.0475 ± 0.0006	4.17
$f_0(1500)$	m	1.505	1.4882 ± 0.0017	9.88
	Γ	0.109	0.1269 ± 0.0021	8.52
	ϕ	0.0	-0.2341 ± 0.0169	13.85
$f_0(1710)$	m	1.720	1.7424 ± 0.0018	12.44
	Γ	0.135	0.0949 ± 0.0024	16.71
	ϕ	0.0	-0.1096 ± 0.0236	4.64
$f_2(1270)$	m	1.274	1.2875 ± 0.0026	5.19
	Γ	0.185	0.3510 ± 0.0144	11.53
$f_2'(1525)$	m	1.525	1.5270 ± 0.0008	2.50
	Γ	0.073	0.1022 ± 0.0022	13.27
	ϕ	1.6	1.6093 ± 0.0116	0.80
$\omega(782)$	m	0.786	fixed	
	Γ	0.0085	fixed	

Start Parameter

The results of the unbinned fit exhibit a strange pattern for slices at the smallest invariant masses. Figure 4.15 shows a crop for $m^2(\pi^0\pi^0) < 1.7\,(\mathrm{GeV}/c^2)^2$ of figure 4.11 top left, with the Breit-Wigner sum fitted to the extracted magnitudes of the simple toy data Slice Fit. The phase shown on the right of figure 4.15 shows an even more pronounced and corresponding pattern. Some of the slices show a magnitude which fits rather well to the Breit-Wigner shape, but about half of them seems to be systematically shifted to higher magnitude values and show bigger uncertainties. This is due to the optimizer finding a second solution very close to the seemingly better one. Figure 4.16 shows the results of multiple fits to an arbitrary slice (sixth point in figure 4.15) with different, randomly varied start parameters. The fit finds two kinds of solutions, the lower ones with the spin 0 magnitude ≈ 3 which correspond to the fit result in figure 4.15, and another solution with a higher value of the spin 0 magnitude ≈ 3.6. The higher points have slightly bigger uncertainties and a little worse likelihood value, indicating a competing and rather shallow local minimum close to the optimal result, which the optimizer is likely to get stuck in.
The two nearly identical solutions probably are a result of an underestimation of the ω in the cases with a slightly worse likelihood value which can be seen when comparing the solutions to the slice data. Figure 4.17 a) shows the distribution of the example slice which was fitted with several, random start parameter sets. Histogram b) shows the model with the fit results with smaller uncertainties and smaller likelihood value which fits the data quite good. Histogram c) shows the model with the fit result of the fits which likely ran in a local minimum. There, one can notice a discrepancy between the data in the ω region.
This effect only occurred in the first few slices. To avoid these kind of local minima in future analyses, the fit of every slice can be performed several times with random start parameter values in order to then choose the result with best likelihood value. An alternative approach could be using another optimizer which is less prone to local minima e.g. one based on evolutionary algorithms.

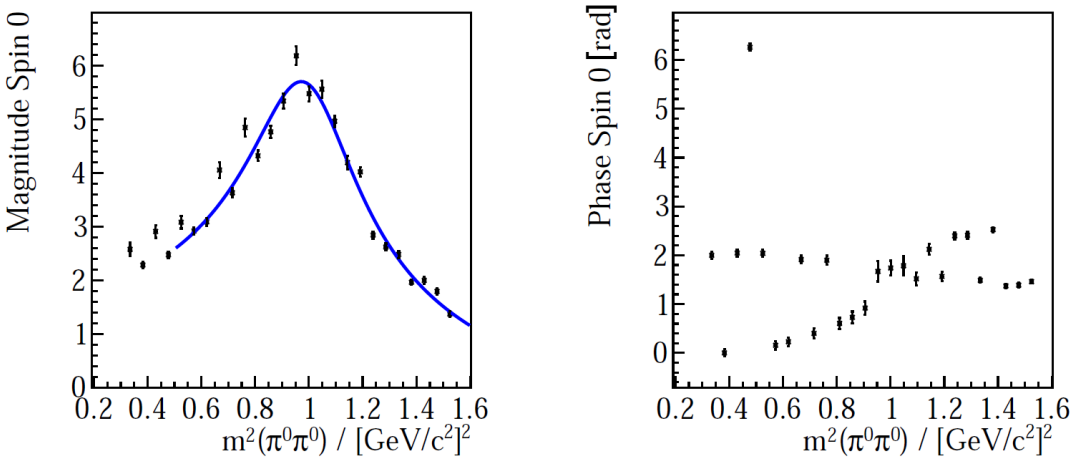

Figure 4.15: Slice Fit parameter (black points) after unbinned fit of each slice and Breit-Wigner fit result (blue) of the simple model toy data, see table 4.8, for the lower energy slices.

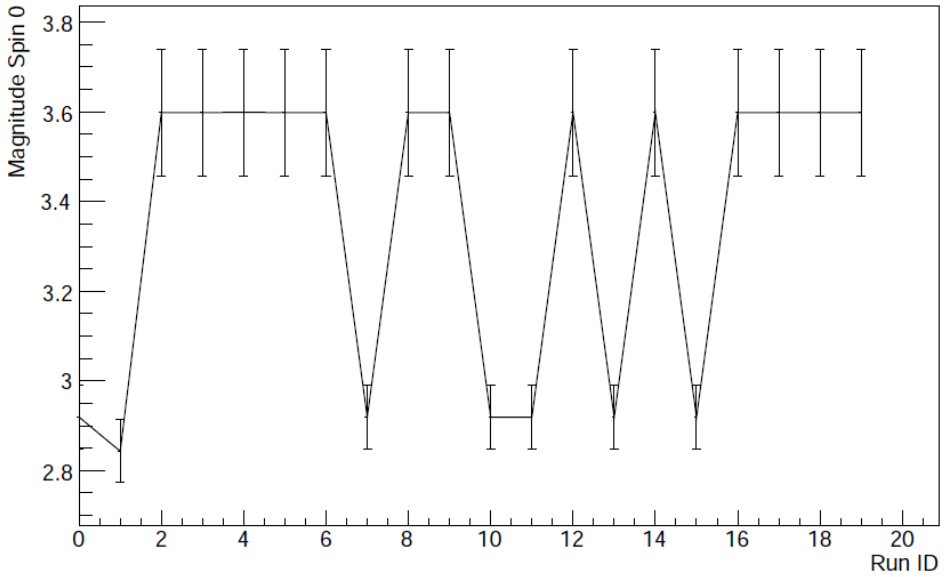

Figure 4.16: Spin 0 magnitude of the slice at $m^2(\pi^0\pi^0) = 0.5\,(\mathrm{GeV}/c^2)^2$ extracted with the unbinned Slice Fit for 20 different sets of randomly created start parameters.

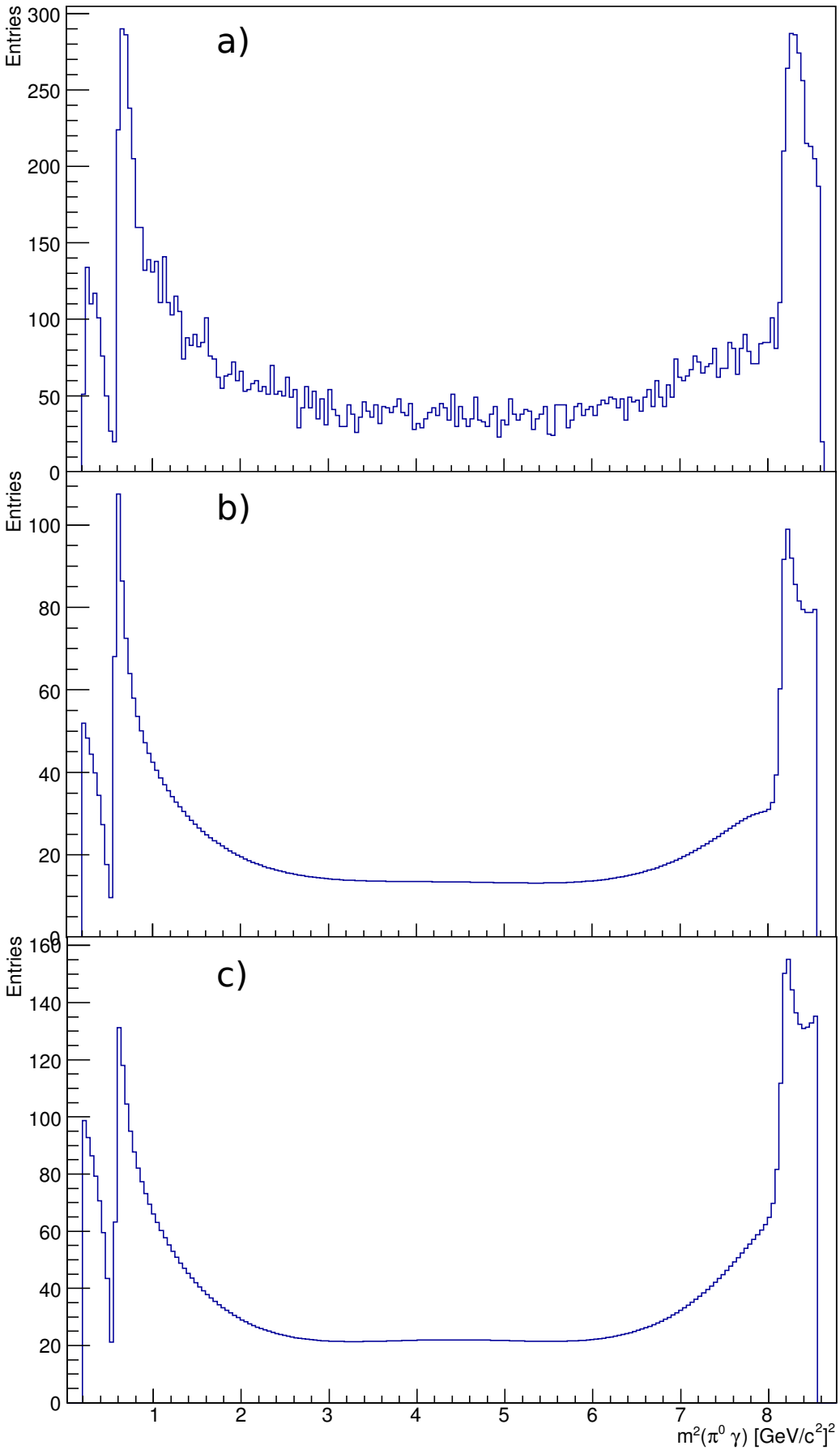

Figure 4.17: a) Histogram of the data of the slice at $m^2(\pi^0\pi^0) = 0.5\,(\,\mathrm{GeV}/c^2)^2$.
b) Histogram of the model values with parameters after fit of the same slice at global minimum.
c) Histogram of the model values with parameters after fit of the same slice at local minimum.

4.4.3 Slice Fit Comparison

Table 4.11 lists the results of the binned and unbinned Slice Fit of the realistic scenario in order to directly compare the results. The deviations from the generated values are smaller in case of the binned Slice Fit, ranging from below per mil to a few percent, e.g. for the $f_2(1270)$ parameters. The pulls show some large values for the tensor resonance parameters, indicating systematic effects. Complete pull distributions are discussed in 4.5. The unbinned fit shows larger deviations, ranging from per mil to more than 10% in case of the widths of the $f_0(1500)$ and the $f_2'(1525)$. Albeit the statistical uncertainties are higher, there are more pull values with unlikely high values in the unbinned case and they are not limited to the f_2 parameters.
This direct comparison strengthens the impression from the unbinned Slice Fit results which did show that this method can not compete with the binned Slice Fit in this scenario. The better separation of the phase motions of the resonances remains a small advantage of the unbinned method, but the binned Slice Fit is more stable and should be used for the extraction of the parameters when a model independent approach is preferred.

Table 4.11: Deviation from simulated values of parameters extracted by the binned and unbinned Slice Fit method of the toy data and pull values (difference/error). Δm: relative deviation from the simulated mass, $\Delta\Gamma$: relative deviation from the simulated width.

Parameter		Binned [%]	Pull	Unbinned [%]	Pull
$f_0(980)$	Δm	0.02 ± 0.05	0.40	0.37 ± 0.05	7.40
	$\Delta\Gamma$	2.56 ± 2.02	1.72	0.40 ± 1.90	0.21
$f_0(1500)$	Δm	0.01 ± 0.07	0.18	-0.35 ± 0.14	2.52
	$\Delta\Gamma$	-0.18 ± 1.19	0.15	17.25 ± 2.39	7.23
$f_0(1710)$	Δm	-0.01 ± 0.12	0.05	0.06 ± 0.23	0.26
	$\Delta\Gamma$	0.96 ± 1.63	0.59	6.74 ± 3.19	2.12
$f_2(1270)$	Δm	-2.06 ± 0.07	29.11	-2.09 ± 0.16	13.3
	$\Delta\Gamma$	-1.35 ± 1.08	1.25	4.05 ± 2.32	1.74
$f_2'(1525)$	Δm	-0.13 ± 0.03	5.00	0.03 ± 0.09	0.29
	$\Delta\Gamma$	0.11 ± 1.33	9.85	26.81 ± 2.77	9.69

4.5 Evaluation of Fit Methods

Systematic Studies

In order to show the systematic behavior of the three methods, depending on statistics and bin size, different scenarios were studied based on the realistic toy Monte Carlo dataset, see figure 4.4. The dataset size was varied in scenarios from ten thousand to four million events, changing the bin and slice size accordingly. The phasespace sample for the Monte Carlo integration has always the same number of events as the toy dataset. In addition, the scenario with one million events was tested with different bin sizes. Table 4.12 shows the settings and the corresponding fit result of the $f_0(980)$ parameters as an example.

Table 4.12: Fit results of the mass (m, in GeV/c^2) and the width (Γ, in GeV) of the $f_0(980)$ resonance by varying settings and methods. Simulated mass = 0.99 GeV/c^2, simulated width = 0.05 GeV.

Dalitz Plot Fit		$m(f_0(980))$		$\Gamma(f_0(980))$	
Events		Fit Result	Pull	Fit Result	Pull
10 000		0.9882 ± 0.0008	2.25	0.0435 ± 0.0016	4.06
100 000		0.9899 ± 0.0003	0.33	0.0512 ± 0.0006	2.00
1 000 000		0.9899 ± 0.0001	1.00	0.0502 ± 0.0002	1.00
4 000 000		0.9901 ± 0.0001	1.00	0.0501 ± 0.0001	1.00
Binned Slice Fit					
Events	Binning				
10 000	50^2	1.0002 ± 0.0042	2.43	0.0235 ± 0.0366	0.72
100 000	160^2	0.9889 ± 0.0014	0.79	0.0516 ± 0.0017	0.94
1 000 000	200^2	0.9901 ± 0.0009	0.11	0.0517 ± 0.0009	1.89
1 000 000	400^2	0.9901 ± 0.0007	0.14	0.0508 ± 0.0007	1.14
1 000 000	800^2	0.9900 ± 0.0004	0.00	0.0503 ± 0.0007	0.43
4 000 000	800^2	0.9901 ± 0.0002	0.50	0.0500 ± 0.0003	0.00
Uninned Slice Fit					
Events	Slices				
10 000	50	1.0064 ± 0.0071	2.31	0.0953 ± 0.0222	2.04
100 000	160	0.9943 ± 0.0039	1.10	0.0587 ± 0.0039	2.23
1 000 000	400	0.9885 ± 0.0005	3.00	0.0439 ± 0.0005	12.2
4 000 000	800	0.9911 ± 0.0003	3.67	0.0582 ± 0.0005	16.4

Comparing the different fit results from table 4.12, the observations are:

- The Dalitz plot fit shows the best overall final parameters and error estimates, especially in the case with a small number of events.
- The unbinned Slice Fit shows the worst results and looks less stable when looking at the resulting parameters.
- All methods show the expected $\propto 1/\sqrt{N}$ dependency of the statistical uncertainty from the number of events used.
- The pull values are reasonable in case of the Dalitz plot fit, indicate an underestimation of the uncertainties by the binned Slice Fit, and show a problem with the estimation of the width in case of the unbinned Slice Fit.

- A finer binning leads to only slightly better results. The scenario with 800^2 bins and one million events has nearly the same statistical uncertainty like the scenario with the same dataset but only 400^2 bins, but increasing the number of events to four million and using 800^2 bins achieves significantly better results. In the case with wider bins and smaller event numbers, the scenario with one million events and 200^2 bins shows smaller uncertainties than the scenario with ten times less events but very close bin size (160^2 bins).

Figures 4.18 and 4.19 show the relative deviation of the final fit parameter and error of all resonances to the true value depending on the number of events and the method used. As in the case of the $f_0(980)$, also the other resonance parameters show similar behavior and a convergence to the ideal values. Only in the case of the two f_2 resonances, the Slice Fit show some systematic discrepancy (figure 4.19). One can see that the unbinned Slice Fit has the biggest statistical uncertainties and shows problems in the case of low event numbers. In most cases, the different bin sizes of the one million event scenario do not change the result much but are only reflected in the statistical uncertainties. When comparing the 100 000 event scenario results with the 1 000 000 events and wide bin (200^2 bins) scenario, the results of the scenario with less events are quite close, indicating the high number of events per slice does not compensate for the loss of information induced by the wide bins.
The large uncertainties of the unbinned Slice Fit result originating from the underlying model are already discussed. The dependence of the statistical uncertainties on the bin size shows that the uncertainties have to be treated carefully in the binned case. On a quantitative level, both Slice Fit methods do not reach the precision of the unbinned Dalitz plot fit except for the case with a very high number of events. However, the Dalitz plot fit was performed with a perfect model while the Slice Fits can only get close to this by definition of the staged procedure.

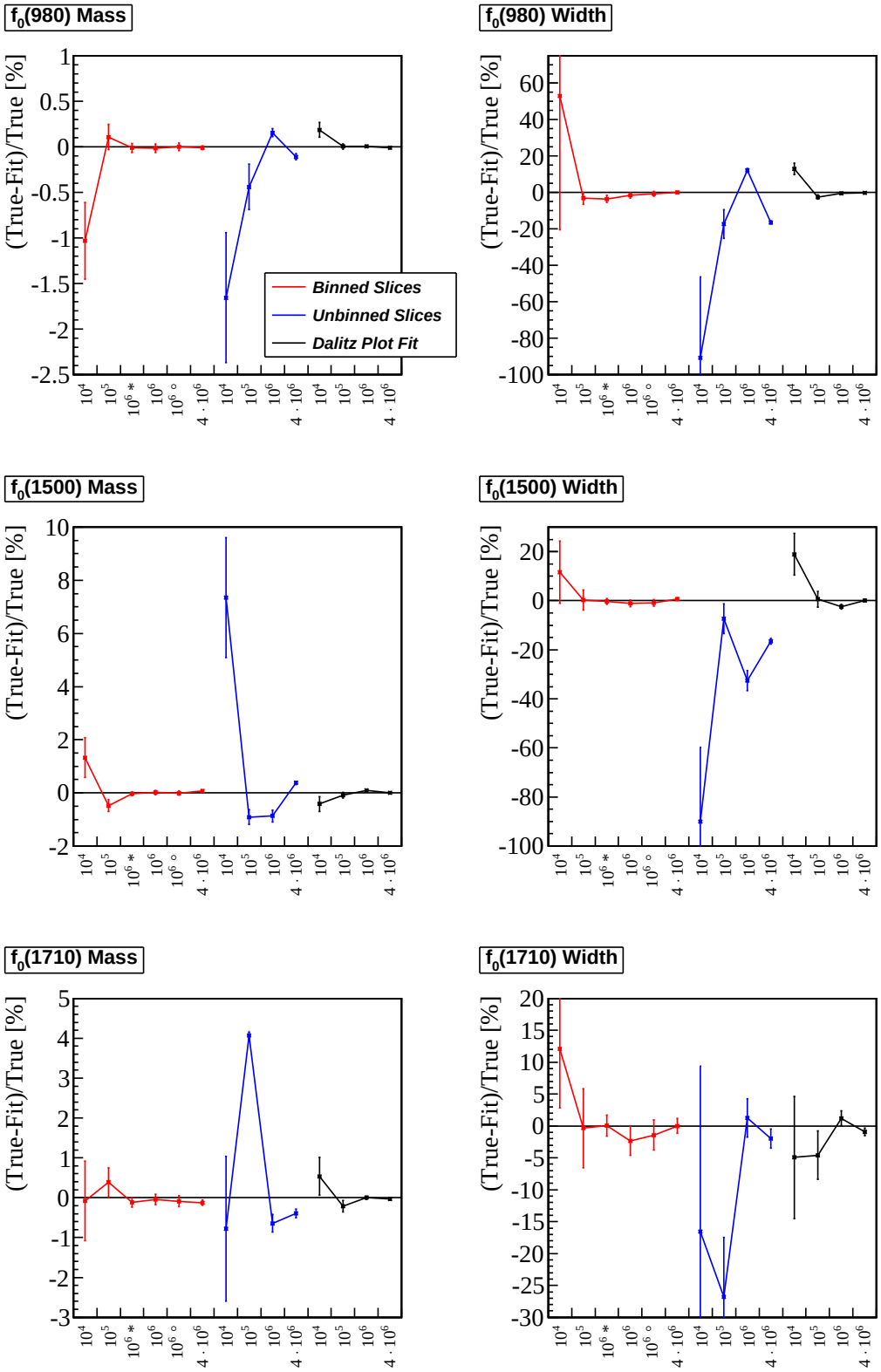

Figure 4.18: Relative Deviation of fitted parameter values to true value of Spin 0 resonances after fit with different number of events and with different techniques, see table 4.12. On the abscissa, the number of events is shown. In the binned slices case, the fit of 10^6 events was performed with 400 bins and with two additional binnings: *) 200 bins and ○) 800 bins.

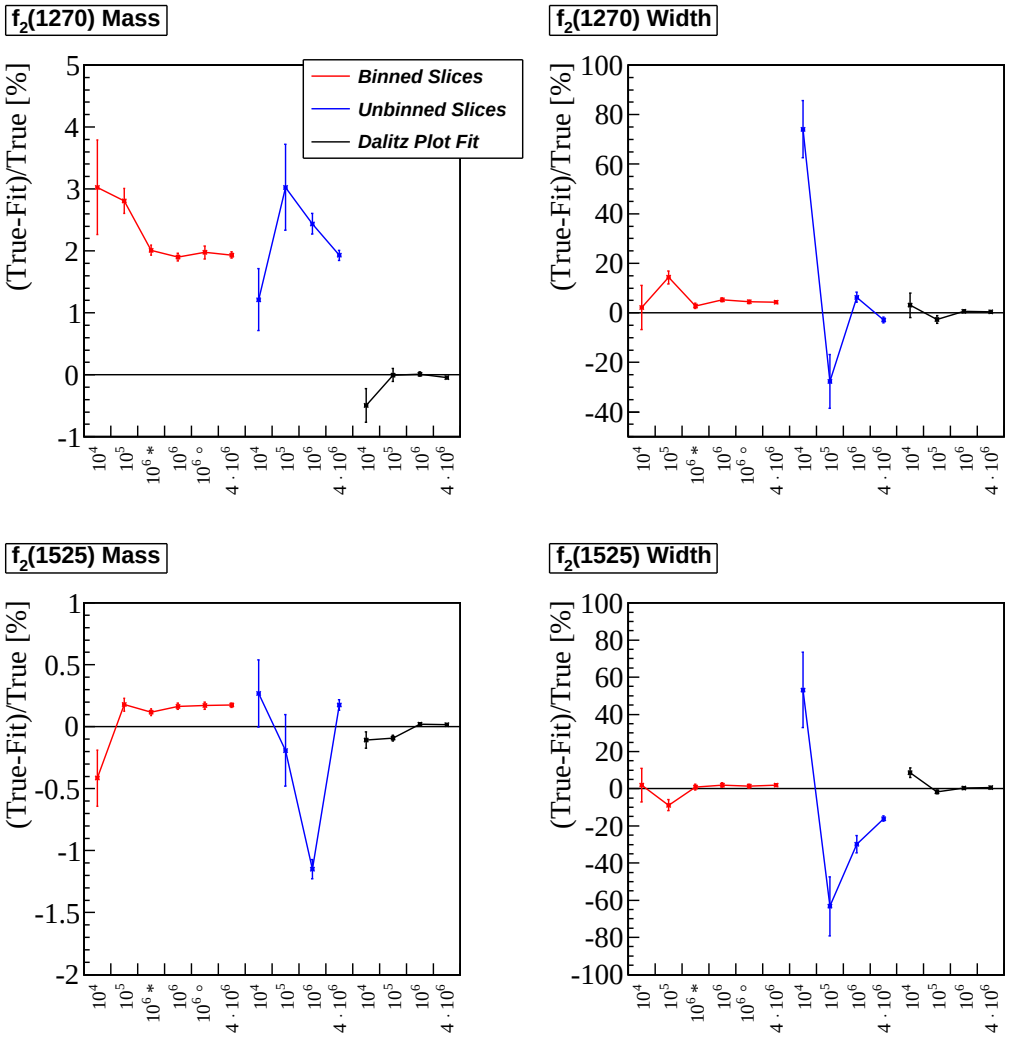

Figure 4.19: Relative Deviation of fitted parameter values to true value of Spin 2 resonances after fit with different number of events and with different techniques, see table 4.12. On the abscissa, the number of events is shown. In the binned slices case, the fit of 10^6 events was performed with 400 bins and with two additional binnings: *) 200 bins and ○) 800 bins.

Error Estimations

In order to test the reliability of the resultant statistical uncertainties of the three different methods, 100 independently generated sets of 100 000 events each using the parameter settings given in table 4.3 were fitted with all methods. The pull value of a parameter is the difference between the parameter value after the fit to the generated value, normalized to the error estimation from the minimizer. If the fitted values are unbiased with gaussian uncertainties, the pull distribution follows a gaussian distributions with mean value $\mu = 0$ and standard deviation $\sigma = 1$. A different μ value indicates a bias of the fit parameter, whereas a different σ indicates over- or underestimation of the uncertainties. Pull distributions of all simulated resonances can be found in the appendix section A.3.
As an example, the resulting parameters for the $f_0(B)$ resonance are shown as pull distributions in figure 4.20. They show a different behavior in the pulls for all three fit methods. In case of the Dalitz plot fit, figure 4.20 a), the parameter uncertainties are slightly underestimated but show a good agreement with a Gaussian distribution and have no bias. As the fitted model is the same that was used for generating the data and there are no binning effects when using the unbinned log likelihood estimator, the most likely reason for the underestimation can lie in the correlation of parameters which is discussed later.
The binned Slice Fit, figure 4.20 b), resembles the ideal expectation quite well. There is no significant bias and the error estimation is even better than in the case of the Dalitz plot fit. Nevertheless, the pull values seem to be a bit segregated which prevents a better fit of the pull distribution.
In contrast, the unbinned Slice Fit, figure 4.20 c), show a prominent bias. As the uncertainties are underestimated, the significance of the shifts is diminished a bit. The width of a slice is about 8 MeV/c^2 in this scenario. The uncertainties are of the same size, so a relative shift of 5 in the pull distributions corresponds to roughly 5 slices covered by the shift. A shift of about the size of the slice width would not be unexpected, but in this case the shift is too big to be a binning effect of the slicing. Also, the unbinned Slice Fit shows some bigger deviations from the Gaussian distributed results which shows that this method is less stable.
In addition, as the second step of the two Slice Fit methods is identically, the under- and overestimation of uncertainties respectively is caused by the first step, the fit to each slice independently. Not only the extracted wave parameters but also the uncertainties per slice are passed to the second step. As the fit results are quite close, the different behavior of the pull distribution must be caused by the uncertainty estimations of the fits of the slices. In the binned case, this works as expected. In the unbinned case, the difference between the approximated Slice Fit model (the change in intensity caused by the shape of a resonance within a slice is not described, as illustrated in figure 4.13) and the unbinned events is problematic. It is difficult to break down the different effects and correlations in a staged procedure, so a sophisticated analysis has to be conducted when the results of this procedure are to be compared quantitatively with common methods.

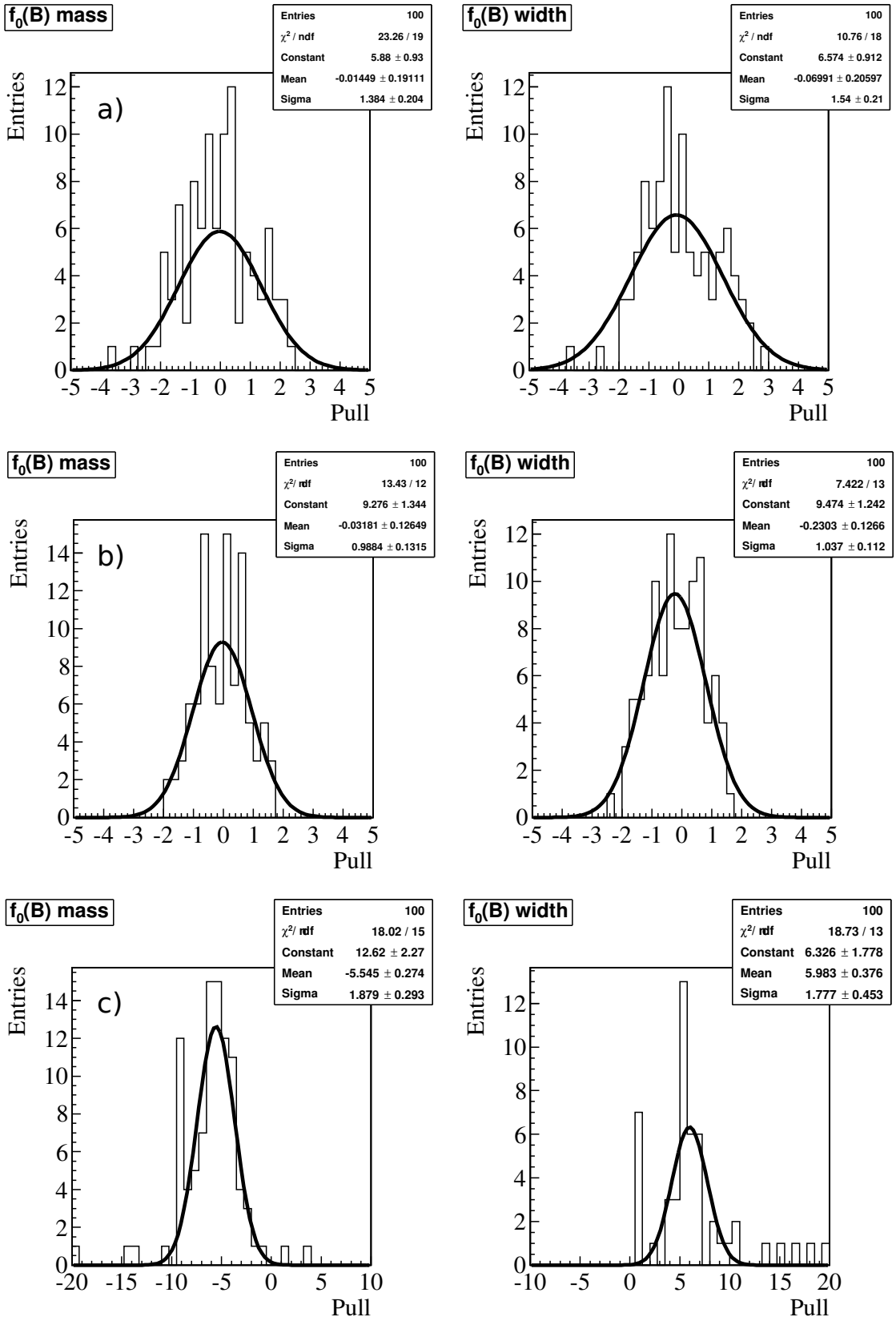

Figure 4.20: Pull distribution of $f_0(B)$ mass and width for 100 fits to 100 000 events each generated with the simple setup, see table 4.3, with: a) unbinned Dalitz plot fit, b) binned Slice Fit and c) unbinned Slice Fit.

One possible source of deviation from an ideal pull distribution is the correlation of parameters. Already in the simple setup, there are 16 parameters to be fitted which tends to correlations. Figure 4.21 shows a visualization of the covariance matrix of the Dalitz plot fit. The strongest correlations involve the strength and phases of all resonances, e.g. the phase of $f_0(A)$ is correlated strongly with the phase of the f_2. The phases are defined relative to each other and the resonances are all overlapping, thus the correlation can be expected. Another expected correlation can be seen within the parameters of the $f_0(A)$, there is a strong correlation of its width and its intensity. Since the resonances are not normalized, a change in the width changes the intensity of the resonance. The strongest anti-correlation appears between the relative phase of $f_0(B)$ and the width and intensity of $f_0(A)$. The latter are strongly correlated, so the anti-correlation effects both. That the relative phase of the $f_0(B)$ influences the intensity of another resonance is not intuitive, but this can be an effect of the interference.
In case of the model-independent Slice Fits, both binned and unbinned, the correlation of parameters is difficult to estimate. In the first step each slice is fitted independently and afterwards independent fits of the extracted waves are performed. In order to scan the parameter space for correlations, one possibility would be to create an estimation function to perform all steps within one fit procedure, which is not trivial.
A more detailed analysis of the error estimation should be performed in future when ComPWA is more advanced. E.g. a cross-check of an existing analysis can be used to better handle the error estimations of the three presented methods, as it provides a reference for the results.

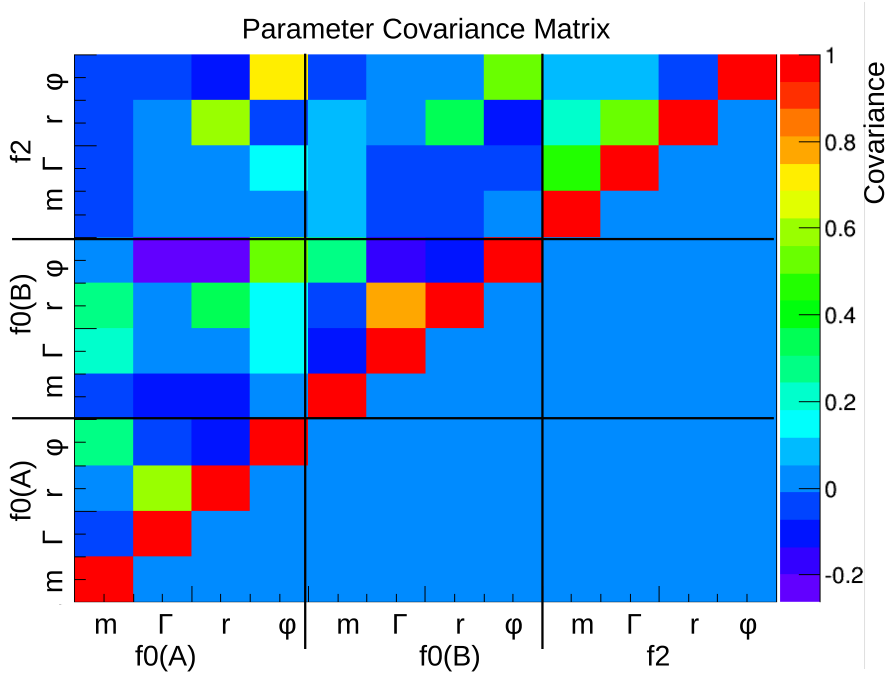

Figure 4.21: Covariance matrix of parameters after an unbinned Dalitz plot fit of 100 000 events generated with the simple setup, see table 4.3, visualized in a two dimensional histogram.

Use Cases

The comparison of the fit methods has shown that both Slice Fit methods are not able to reproduce the model parameters as good as the unbinned Dalitz plot fit. The binned Slice Fit has a good error estimation and gets close to the results of the Dalitz plot fit in case of a high number of events but lacks a clear separation of the phases. The unbinned Slice Fit is unstable and has problems with the error estimations but shows an improvement in the extraction of the phases compared to the binned Slice Fit. Therefore, if the resonance models are known, the unbinned Dalitz plot fit is the most reliable choice to extract resonance parameters.
However, the Slice Fit methods can help to identify resonances and to choose the correct model for a resonance before performing an unbinned Dalitz plot fit which depends on a complete model of the reaction. A comparison of the Dalitz plot projection to the extracted waves is shown in figure 4.22. Since the resonance terms are not normalized and the angular distributions are different, there is a factor to be considered between the spin 0 and the spin 2 waves when comparing the contributions. Albeit the projection is certainly not used for a fit, it represents the first impression an analyst gets before choosing a model for the Dalitz plot fit. The waves are extracted without the use of a model for resonances in $\pi^0\pi^0$ and clearly build a better basis for identifying resonances and choosing the resonance model if necessary. In addition, the extracted phases can help distinguishing kinematic or detector effects from actual resonances when phase motions are visible. The unbinned Slice Fit showed a qualitatively better separation of the extracted phases and thus can be considered a helpful tool when searching for resonances in a suitable channel.
In a scenario where the resonance of interest is overlain by an unknown, hard to model resonance with a different spin the binned Slice Fit might even be a better choice for the fit than the unbinned Dalitz plot fit. The Dalitz plot fit is not reliable if the interference with the overlying resonance can not be modeled correctly. With the Slice Fit, the waves are separated without the need to model the overlying resonance at all. Thus, only the model for the resonance of interest and maybe close resonances with the same spin is necessary in the second step of the Slice Fit and systematic uncertainties introduced by the hard to model resonance are avoided at the expense of the binning. If this is beneficial has to be tested from case to case.
Another interesting use for the wave extraction could be in channels with a higher number of final state particles. Since there is no visualization of the complete phasespace possible in this case, one has to rely on projections or mass-dependent subsets in order to verify the modelation. Kinematic and interference effects are harder to observe without the complete information, but if part of the decay is well enough understood the Slice Fit can be used on one final state particle pair to extract the waves without the overlying effects.

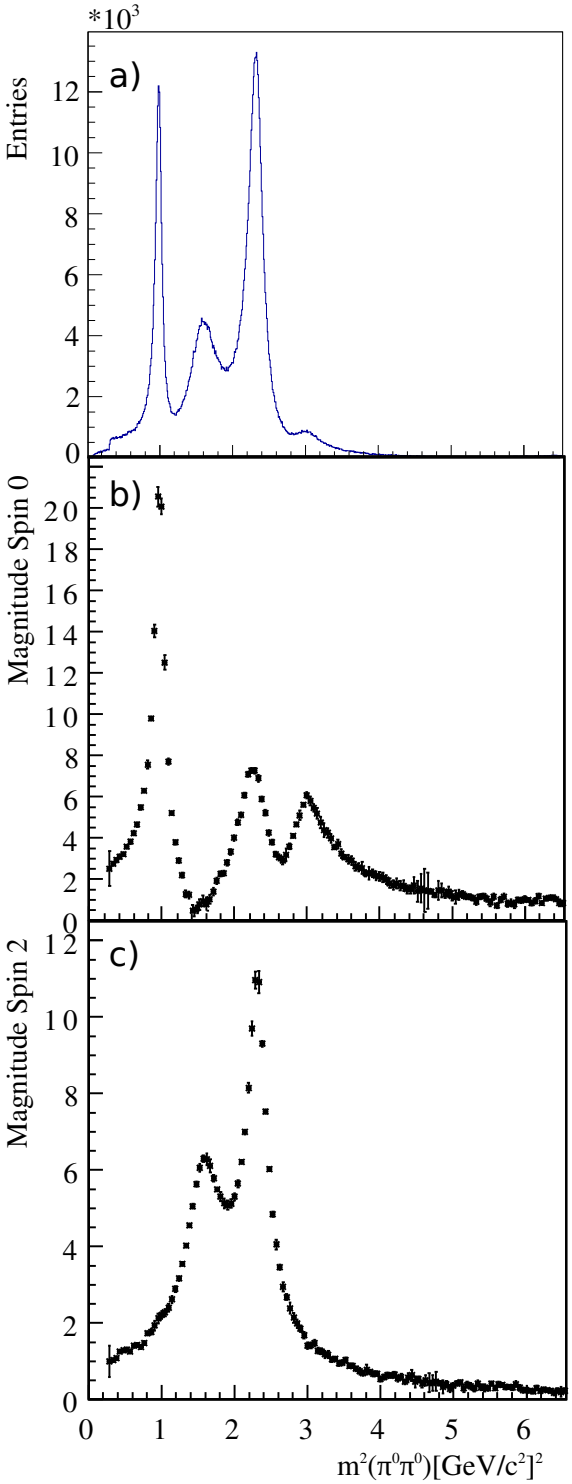

Figure 4.22: a) Projection of the Dalitz plot of the $J/\psi \to \gamma\pi^0\pi^0$ toy data to the $m^2(\pi^0\pi^0)$ axis.
b) Extracted spin 0 wave with the binned Slice Fit.
c) Extracted spin2 wave with the binned Slice Fit.

4.6 Function Tree for Breit-Wigner Sum Model

In chapter 3.8 the usage of the ComPWA Function Tree was motivated. In the following, the gain in speed by calculating the likelihood value of the amplitude of the Breit-Wigner sum model (equation 4.1) for a certain set of parameters was investigated and will be presented. Figure 4.23 shows a schematic of the Function Tree which calculates the amplitude for a number of resonances in the Breit-Wigner sum model.
For the likelihood calculation, the amplitude is evaluated at the position of every data event and the logarithms of the intensities are summed up.

$$\mathcal{L}_{data} = \sum_{data} \log(|c_\omega T_\omega D_\omega + c_{980} T_{980} D_0 + c_{1270} T_{1270} D_2 + ...|^2), \quad c = r \cdot e^{i\phi} \tag{4.7}$$

For the normalization, the sum of intensities at the position of phasespace distributed Monte Carlo events is calculated.

$$\mathcal{L}_{norm} = \sum_{phasespace} |c_\omega T_\omega D_\omega + c_{980} T_{980} D_0 + c_{1270} T_{1270} D_2 + ...|^2, \quad c = r \cdot e^{i\phi} \tag{4.8}$$

These two terms are sufficient since there are no background models or other incoherent terms present for this toy data analysis. Two Function Trees are set up by the Physics module for the calculation of $\mathcal{L}_{data}$ and $\mathcal{L}_{norm}$ and are then passed to the Estimator module. The final calculation of the likelihood value is then performed by the Estimator module as described in section 3.8.
The calculation of the Breit-Wigner and Wigner D functions were implemented as Function Tree strategies, so their calculations take place in one node instead of splitting the calculations into subtrees using basic algebraic operations. The other strategies needed for the amplitude calculation are basic algebraic operations.

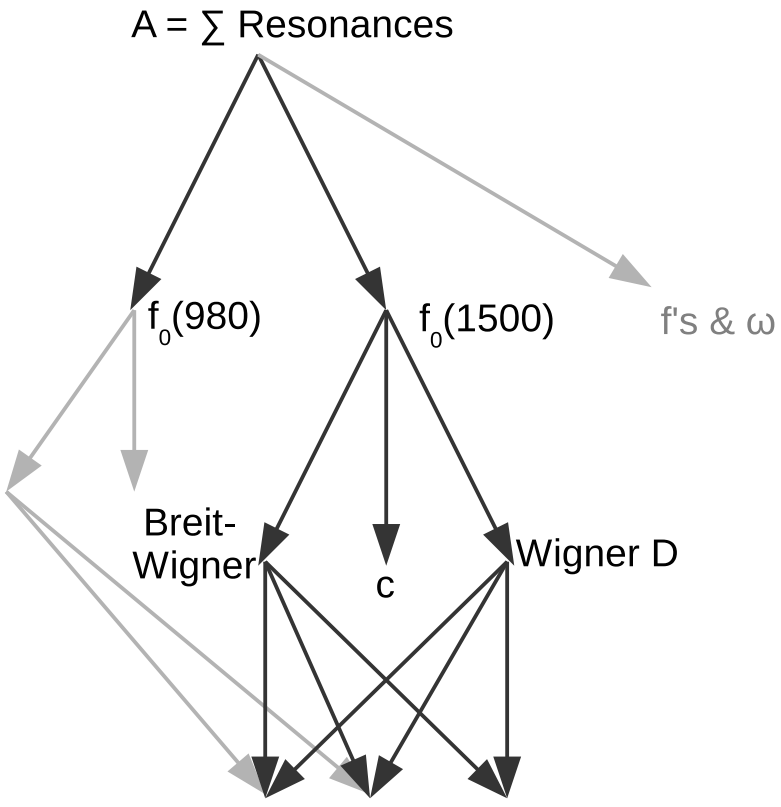

Figure 4.23: Sketch of the Function Tree setup for the Breit-Wigner sum model. The full branch of one resonance is shown and the other branches are partly indicated in gray.

The Function Tree means additional effort for the users, as they need to set up the tree for their amplitude calculation. The main benefit justifying this additional effort is the significant speedup when fitting the model to data. To test if and how much this implementation can save time when performing the fit, the amplitude was set up as a Function Tree (see Figure 4.23) and 100 000 toy Monte Carlo data events of the channel $J/\psi \to \gamma\pi^0\pi^0$ were fitted.
As the speedup of the Function Tree strongly depends on the way of how parameters of the model change during the fit procedure, it is important to understand how the optimization algorithm works. For this test, the gradient descent based migrad algorithm of Minuit2 was chosen, as it is commonly used in high energy physics analyses. A gradient descent algorithm only varies one parameter in each optimization step and needs recalculation afterwards although it was only partly changed. There are also other algorithms available, e.g. provided by Geneva [46], changing multiple parameters at once between calculations, which means there are less static parts in the Function Tree between two calculations and therefore less speedup is expected.
There were three analysis setups chosen with a different amount of fit parameters and different impact on the calculation to compare the run-times:

- 1 Magnitude: The magnitude of one *f* resonance was fitted
- 5 Magnitudes: The magnitudes of all five *f* resonances were fitted
- 5 Widths: The widths of all five *f* resonances were fitted

All other magnitudes or widths and all additional parameters, e.g. all phases and the parameters of the ω resonance, were fixed during the fit procedure. The set of free parameters in each scenario was not chosen by physical meaning but solely to provide scenarios with a different level of complexity for the measurement of the run-times. All fits are set up to converge nicely. To prevent the results to be dominated by machine conditions, they were repeated twenty times in case of the direct calculation and ten times in case of the Function Tree. Table 4.13 shows the average run-time of the three different scenarios with and without using a Function Tree.

Table 4.13: Average run-time of six toy Monte Carlo analyses.

Scenario	Iterations	Avg. Run-Time without Tree [s]	Avg. Run-Time with Tree [s]	Speedup
1 Magnitude	7	3.50 ± 0.03	0.067 ± 0.001	> 50
5 Magnitudes	37	29.44 ± 0.17	1.017 ± 0.001	> 25
5 Widths	31	22.83 ± 0.20	2.587 ± 0.007	> 8

For the case without using the Function Tree, the optimization converges fastest when only one parameter needs to be fitted to the data, as expected. The two cases with five parameters have a higher dimensional parameter-space of equal size, but the fitting of the widths converged faster. In all three cases, the calculation performed after changing one of the parameters is always the same, the complete amplitude needs to be recalculated for all events. Thus, the time needed to perform the fit is directly correlated with the numbers of iteration.
Looking at the run-times and speedup when using the Function Tree, the time saving with the cached calculations is obvious. The biggest speedup is visible for the simplest case with just one free magnitude parameter. Here, only one of the resonance terms needed recalculation during the hole fit. In addition, the dynamics function and the angular distribution in the recalculated branch of the fitted resonance were constant and thus cached, since the magnitude just acts as a scaling factor. Nevertheless, the likelihood needs recalculation for every data point every time the parameter changes. In case of fitting the magnitudes of five resonances, the speedup is reduced due to the higher number of affected branches. When fitting the widths of five resonances, the Function Tree needs even more time, since the Function Tree needs more recalculation because the dynamical parts of the resonances change as well, not only

the coefficients. With every change of a width, one of the seven Breit-Wigner functions in the Function Tree needs to be recalculated which was not the case before. This leads to the smallest speedup of all test cases, as expected. The fitting of the widths converges faster than the fitting of the magnitudes, which can be noticed when comparing the run-times of the fits without the Function Tree used. When comparing the run-times of these fits with the Function Tree, this means the speedup per iteration is even higher than the overall run-time suggests since the faster fit needed more iterations.
The three scenarios already demonstrate significant speedups which are paid by memory consumption and configuration overhead. The biggest part of the additional memory consumption scales with the number of events N_{events} and can be easily estimated via

$$Memory(tree) \approx N_{events} \cdot N_{nodes} \cdot Memory(node) \tag{4.9}$$

with the number of nodes N_{nodes} and the memory consumption of the data type which is cached in a node $Memory(node)$. In this setup, there are seven nodes for the resonances which have two sub nodes each which also differ per event, all caching a complex number in double precision. Thus, the memory consumption is calculated with $N_{nodes} = 21$, $N_{events} = 100\,000$, and $Memory_{node} = 16$ byte giving an estimate of $Memory(tree) = 33\,600\,000$ byte ≈ 35 MB which is uncritical. The other parts of the tree, like the structure itself and the leafs, e.g. the parameters of the resonances, do not significantly to the memory consumption as they do not scale with N_{events}. In a real world example the memory consumption needs to be kept in mind tough, since a factor of ten more nodes and ten times more events would already result in a memory consumption close to the limits of modern computers: $Memory(tree) \approx 3.5$ GB.
Also, the Function Tree shows the strong dependency on the algebraic expression of the physics model and on the optimization routine. The final state and the model used for this test were rather simple. In case of more complex models, more difficult decays and more resonances occurring, larger Function Trees arise. There, more branches are constant during the fitting procedure or changes occur less often which means even more computing time can be saved by caching the permanently or temporary constant terms. On the other hand, other optimization routines, e.g. genetic algorithms which change multiple parameters at once, will benefit less of the caching as several branches change at once. Therefore, although the Function Tree can be used with any model or optimizer, it must be considered individually how speedup and memory consumption behave.

4.7 Optimization Tests

One of the major advantages of the modular design of ComPWA is that various optimization algorithms can not only be freely added and be chosen of, but in addition multiple algorithms can be used in one fit. Thus, it is possible to combine different algorithms in order to benefit from the respective features. For instance, one can start a fit using the genetic algorithms of Geneva, which have a fast convergence in the beginning (figure 4.24 left side) and the better stability of the Monte Carlo approach against local minima. Afterwards, the resulting set of parameters can be used as start parameters for the gradient descent algorithm of Minuit2 which provides better convergence close to the minimum (figure 4.24 right side) and has better convergence criteria using covariance matrices and uncertainty estimations. In ComPWA, such or similar procedures can be easily achieved as the interface to both libraries is the same and the algorithms operate on the same set of parameters.

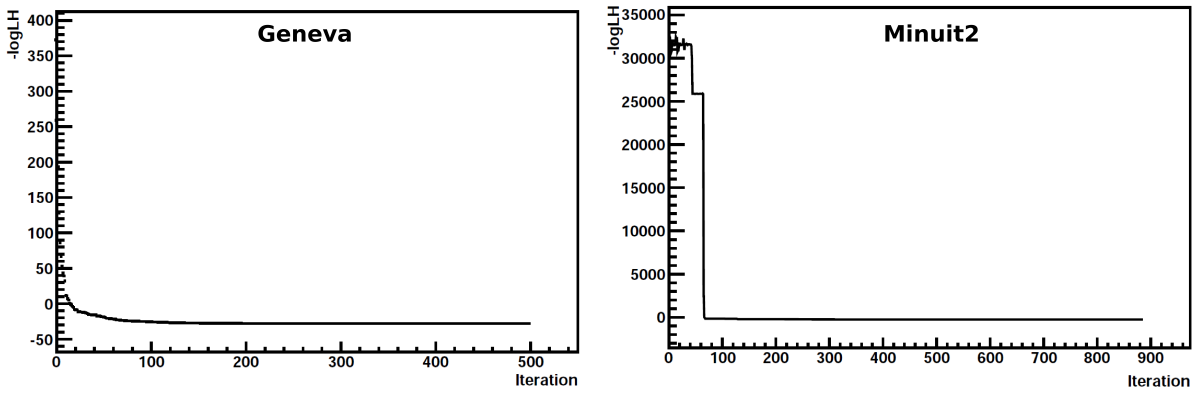

Figure 4.24: Convergence of likelihood using a Geneva [46] genetic algorithm (left side) in comparison to the Minuit2 [33] gradient descent algorithm (right side).

Summary

The light quark region is populated with many resonances which are not yet fully understood. Responsible for this lack of understanding are the resonances being quite broad and therefore overlapping which makes a good modelation of close resonances and of the interference effects most important. As there are more resonances than expected when assuming quark-antiquark states only, some of these resonances observed are candidates for having exotic internal structure like gluonic degrees of freedom or extra quarks. To verify or falsify this assumption, the quantum numbers and resonance shape parameters of the resonances need to be extracted and analyzed in multiple channels and experiments to get a clearer picture. As the internal structure and the interference of resonances are demanding for models which describe the data, reducing model-dependency is a consequential and worthwhile approach.
Therefore, a model-independent analysis technique (Slice Fit) for extracting waves using a two staged binned approach was presented. Tests were performed on Monte Carlo data for the channel $J/\psi \to \gamma\pi^0\pi^0$ using the newly developed ComPWA amplitude analysis framework. ComPWA has a modular structure in order to prevent limitations when an extension to additional cases becomes necessary. The implementation of this model-independent ansatz was one of the first test cases. The core of this analysis technique is the extraction of the various spin-waves separately in bins of the invariant mass of the subsystem of interest without assuming a model for the single resonances as e.g. the Breit-Wigner function with a dedicated mass and width.
It is advantageous for this model-independent method when a reference phase can be provided, e.g. by means of a well-described resonance in the crossing subsystem of the same channel. Unfortunately, the only suitable candidate for this test channel, $\omega \to \gamma\pi^0$, is quite narrow which limits the extraction of the phases. Nevertheless, it can be concluded that the model-independent fit can be used to extract the dynamical behavior of the $\pi^0\pi^0$ resonances before a model needs to be chosen to describe them. The separation of waves simplifies the interpretation of the extracted structures because the set of contributing resonances is limited to a certain set of quantum numbers and the interference with other waves is already taken into account. This technique can also be used to better judge the amount of resonances needed per spin-wave, to choose the correct model for certain resonances, or to test various resonance models before an unbinned Dalitz plot fit of the complete amplitude is performed.
Three different analysis methods were tested to compare their sensitivity and stability: the unbinned Dalitz plot fit, the Slice Fit with binned slices, and the Slice Fit with unbinned fit of the slices. Quantitatively, the best results were obtained with the common Dalitz-Plot fit, which can be expected since the model perfectly resembles the input. The Monte Carlo data generation used the exact same model in order to have an idealized reference point. It is followed by the binned Slice Fit which works stable and fast but does only come near the ideal precision of the Dalitz plot fit with a high number of events available due to the binning and the limits of the method for this channel. It does however show the benefits of extracting and separating the waves in a qualitative way, as the waves are easier to model and interpret than in the Dalitz plot. The unbinned Slice Fit is unstable in this setup and shows effects like degenerated results depending on the start parameters chosen. The reason is a local minimum with underestimated ω resonance which is quite close to the better solution in the parameter space. The unbinned approach also shows the bigger uncertainty estimates due to the usage of the extraction model which does not describe the data in a slice perfectly: by design it only approximates the resonances in a slice. However, it succeeded in the extraction and the phases seem to be even better separated than in the binned case. So the Slice Fit with unbinned fit of the slices can

still be considered as a cross-check for resonance models in a Dalitz plot fit or to distinguish between resonances and other effects.
All tests were performed within ComPWA which minimizes systematic differences in the comparison of the methods. This is possible because the framework offers flexibility for new analysis techniques while also being open for any amplitude model and formalism as well as data from any experimental setup. As all components of the various analyses are identical except for the necessarily different parts, a comparison is more simple than in the case of stand-alone tools where differences can have multiple sources. For example, the unbinned likelihood estimation function for the Slice Fit is the same as the one used for the complete Dalitz plot fit, which makes it easier to ensure it works correctly. The framework also allows to switch and compare parts of the analysis, so tests could be performed with different optimization algorithms from the Minuit2 and Geneva libraries. The ComPWA Function Tree shows huge potential for accelerating amplitude analysis while being independent from formalisms or models.

Next Steps of ComPWA Development

The ComPWA framework is still under development, nevertheless first analyses are ongoing, e.g. a Dalitz plot analysis of $D^0 \to K^0_S K^+ K^-$ data measured at BESIII [68]. Beside using it for more experiments and different physics analyses, the ComPWA framework still lacks some of its basic features and usability. The lack of amplitude generators is the most important issue to be tackled in order to be able to redo analyses to test ComPWA. A sophisticated helicity amplitude generator for ComPWA is in preparation [69]. The other modules provide enough functionality for first tests and are suitable for common analysis methods.
Especially the user interfaces, in particular the fit management and the generation of reports, need to be implemented in order to make it usable for a broader user base. Expert systems are planned to simplify the generation of decay trees and amplitude models. To rerun analyses partly, it will be possible to persist the whole fit in every stage. This will allow the reload of the exact fit procedure in every single step of the fit and will be useful when generating reports while fitting.

Next Steps of Analysis

The Monte Carlo studies have shown the potential and limitations of the model-independent approach and the benefits of extracting spin waves. Now, the method needs to be applied on experimental data, e.g. to the huge J/ψ dataset from BESIII. Therefore, the treatment of possible background channels, combinatorial background, and detector resolution effects have to be taken into account which will extend the capabilities of ComPWA.
The extraction of the spin-waves of a $\pi^0\pi^0$ final state with the Slice Fit method will give new insight on the light quark resonances since interference effects between different waves are removed and the shapes of the resonances can be observed without modeling it. Compared to the common Dalitz plot fit, this raises the sensitivity for unexpected shapes and different behavior between the resonances which can be an useful additional information in order to assign the observed resonances to physical states.

List of Tables

M.1 Quark properties 1

1.1 Assignment of light mesons 12
1.2 Isoscalar meson parameters 15

2.1 PANDA beam parameter 18
2.2 BEPCII parameter 22

4.1 Parameters for the event generation 48
4.2 Branching ratios of $\gamma\pi^0\pi^0$ final state 48
4.3 Parameters for alternative MC data generation 50
4.4 Parameters for Monte Carlo data generation 51
4.5 Dalitz plot fit parameters 53
4.6 Parameters after binned Slice Fit to simple toy data 58
4.7 Parameters after binned Slice Fit to toy data 60
4.8 Parameters after unbinned Slice Fit to simple toy data 62
4.9 Parameters after unbinned Slice Fit to toy data 65
4.10 Parameters after complex unbinned Slice Fit to toy data 68
4.11 Parameters after binned and unbinned Slice Fit 71
4.12 Parameters of systematic study 72
4.13 Run-time comparison with and without Function Tree 82

A.1 Quantum numbers and helicities 97
A.2 Parameters after binned Slice Fit to simple toy data (high statistics) 110
A.3 Parameters after binned Slice Fit to simple toy data (alternative phase) 112
A.4 Parameters after binned Slice Fit to simple toy data (broader ω) 114

List of Figures

M.1 Meson multiplets . . . 2
M.2 Baryon multiplets . . . 3

1.1 Sketch of a prism . . . 5
1.2 Hydrogen spectrum . . . 6
1.3 J/ψ discovery plot . . . 7
1.4 Charmonium states . . . 7
1.5 LQCD charmonium spectrum . . . 8
1.6 Quarkmodel bound states . . . 11
1.7 Meson Resonances . . . 13
1.8 Isoscalar meson spectrum . . . 14
1.9 Models for the $f_0(980)$ shapes in $\gamma\gamma \to \pi^0\pi^0$ . . . 16

2.1 FAIR overview . . . 17
2.2 HESR overview . . . 18
2.3 The PANDA detector . . . 19
2.4 The BEPCII collider . . . 22
2.5 The BESIII detector . . . 23

3.1 Basic components of amplitude analysis software . . . 26
3.2 ComPWA design . . . 29
3.3 Sketch of ComPWA modules . . . 30
3.4 XML configuration example . . . 34
3.5 Example of a Function Tree . . . 35
3.6 Function Tree classes . . . 37
3.7 Relativistic Breit-Wigner Function . . . 39
3.8 Wigner-*D* plots . . . 40
3.9 Expert system sketch . . . 44
3.10 Expert system example . . . 45
3.11 Helicity formalism expert system . . . 45

4.1 Subsequent decays of $J/\psi \to \gamma\pi^0\pi^0$ . . . 47
4.2 BESIII $J/\psi \to \gamma\pi^0\pi^0$ data . . . 49
4.3 Dalitz plot of 10^6 $J/\psi \to \gamma\pi^0\pi^0$ toy data events . . . 51

4.4 Dalitz plots of $J/\psi \to \gamma\pi^0\pi^0$ toy data . . . 52
4.5 Dalitz plot and ratio plot of 10^6 $J/\psi \to \gamma\pi^0\pi^0$ events using fitted parameters . . . 54
4.6 Dalitz plot of 10^6 $J/\psi \to \gamma\pi^0\pi^0$ events showing the slices . . . 55
4.7 Example slice in f_0 region . . . 55
4.8 Example slice in f_2 region . . . 56
4.9 Result of binned Slice Fit to simple toy data . . . 58
4.10 Result of binned Slice Fit to toy data . . . 59
4.11 Result of unbinned Slice Fit to simple toy data . . . 62
4.12 Result of unbinned Slice Fit to toy data . . . 64
4.13 Illustration of Slice Fit resonance approximation . . . 66
4.14 Result of complex unbinned Slice Fit to toy data . . . 67
4.15 Slice Fit parameter at low energies and Breit-Wigner fit . . . 69
4.16 Slice Fit parameter of one slice with different start parameters . . . 69
4.17 Histograms of data and model at slice $m^2(\pi^0\pi^0) = 0.5\,GeV^2/c^4$ . . . 70
4.18 Systematic of parameters of Spin 0 resonances . . . 74
4.19 Systematic of parameters of Spin 2 resonances . . . 75
4.20 Pulls of $f_0(B)$ with different methods . . . 77
4.21 Covariance matrix of Dalitz plot fit . . . 78
4.22 Projection of Dalitz plot compared to extracted waves . . . 80
4.23 Sketch of the function tree setup . . . 81
4.24 Convergence behaviour of Minuit2 and Geneva optimizer . . . 84

A.1 Relativistic Breit-Wigner . . . 96
A.2 Sketch of decays in the $J/\psi \to \gamma\pi^0\pi^0$ channel. . . . 97
A.3 Distribution of f_2 decay angle . . . 98
A.4 Distribution of ω decay angle . . . 98
A.5 θ distribution of spin 2 resonances . . . 104
A.6 ϕ distribution of spin 2 resonances . . . 105
A.7 θ distribution of ω resonance . . . 108
A.8 Result of binned Slice Fit to simple toy data (high statistics) . . . 109
A.9 Result of binned Slice Fit to simple toy data (alternative phase) . . . 111
A.10 Result of binned Slice Fit to simple toy data (broader ω) . . . 113
A.11 Pull distribution of Dalitz plot fit . . . 115
A.12 Pull distribution of binned Slice Fit . . . 116
A.13 Pull distribution of unbinned Slice Fit . . . 117

Bibliography

[1] J. C. Collins, *Renormalization: an introduction to renormalization, the renormalization group, and the operator-product expansion*. Cambridge monographs on mathematical physics, Cambridge: Cambridge Univ. Press, 1984.

[2] K. A. Olive *et al.*, "Review of particle physics," *Chin. Phys. C*, vol. C38, p. 090001, 2014.

[3] M. Gell-Mann, "A Schematic Model of Baryons and Mesons," *Phys. Lett.*, vol. 8, pp. 214–215, 1964.

[4] G. Zweig, "An SU(3) model for strong interaction symmetry and its breaking. Version 1," *CERN-TH-401*, 1964.

[5] The PANDA Collaboration, "Physics Performance Report for: PANDA," tech. rep., GSI, Mar 2009.

[6] K. J. Peters, "A Primer on partial wave analysis," *Int. J. Mod. Phys.*, vol. A21, pp. 5618–5624, 2006.

[7] D. M. Asner *et al.*, "Physics at BES-III," *arXiv:0809.1869*, Sep 2008.

[8] M. Ablikim *et al.*, "An amplitude analysis of the $\pi^0\pi^0$ system produced in radiative J/ψ decays," *arXiv:1506.00546*, 2015.

[9] J. Bennett, *An Amplitude Analysis of the $\pi^0\pi^0$ System produced in radiative J/ψ decays*. Ph.D. Thesis, Indiana University, 2014.

[10] W. Benenson and H. Stöcker, *Handbook of Physics*. Handbook of Physics, Springer, 2002.

[11] J. J. Aubert *et al.*, "Experimental observation of a heavy particle *j*," *Phys. Rev. Lett.*, vol. 33, pp. 1404–1406, Dec 1974.

[12] S. L. Glashow, J. Iliopoulos, and L. Maiani, "Weak interactions with lepton-hadron symmetry," *Phys. Rev. D*, vol. 2, pp. 1285–1292, Oct 1970.

[13] S. S. Pinsky, "The ψ' to J/ψ Hadronic Decay Puzzle," *Phys. Lett.*, vol. B236, p. 479, 1990.

[14] E. Eichten and F. Feinberg, "Spin-dependent forces in quantum chromodynamics," *Phys. Rev. D*, vol. 23, pp. 2724–2744, Jun 1981.

[15] M. Gell-Mann, "A schematic model of baryons and mesons," *Phys. Lett.*, vol. 8, no. 3, pp. 214 – 215, 1964.

[16] M. Shifman and A. Vainshtein, "Highly excited mesons, linear Regge trajectories, and the pattern of the chiral symmetry realization," *Phys. Rev. D*, vol. 77, p. 034002, Feb 2008.

[17] M. DeWitt, *The Spectrum and Decays of Scalar Mesons in the Light–Front Quark Model*. Ph.D. Thesis, North Carolina State University, 2008.

[18] N. N. Achasov and G. N. Shestakov, "The Shape of the $f_0(980)$ in $\gamma\gamma \to \pi^+\pi^-$," *Phys. Rev.*, vol. D72, p. 013006, 2005.

[19] H. H. Gutbrod et al., *FAIR Baseline Technical Report*. Darmstadt GSI, 2006.

[20] D. M. Welsch, *Investigation and Optimization of Transverse Non-Linear Beam Dynamics in the High-Energy Storage Ring HESR*. Ph.D. Thesis, University of Bonn, 2009.

[21] B. Povh, K. Rith, C. Scholz, and F. Zetsche, *Teilchen und Kerne: Eine Einführung in die*

physikalischen Konzepte. Physics and astronomy online library, Springer, 2004.

[22] P. Collaboration, "Homepage of the PANDA experimental setup." `http://www-panda.gsi.de/`, Feb. 2016.

[23] W. Erni *et al.*, "Technical Design Report for the: PANDA Micro Vertex Detector," *arXiv:1207.6581*, 2012.

[24] M. Ablikim *et al.*, "Observation of a Charged Charmoniumlike Structure in $e^+e^- \to \pi^+\pi^- J/\psi$ at $\sqrt{s}$ = 4.26 *GeV*," *Phys. Rev. Lett.*, vol. 110, p. 252001, Jun 2013.

[25] C. Zhang, "Status of BEPC and plan of BEPCII," in *High luminosity e+e- colliders. Proceedings, 23rd Advanced ICFA Beam Dynamics Workshop*, 2001.

[26] Z. Q. Liu *et al.*, "Study of $e^+e^- \to \pi^+\pi^- J/\psi$ and Observation of a Charged Charmoniumlike State at Belle," *Phys. Rev. Lett.*, vol. 110, p. 252002, Jun 2013.

[27] M. Ablikim *et al.*, "Observation of a Charged Charmoniumlike Structure $Z_c(4020)$ and Search for the $Z_c(3900)$ in $e^+e^- \to \pi^+\pi^- h_c$," *Phys. Rev. Lett.*, vol. 111, no. 24, p. 242001, 2013.

[28] M. Ablikim *et al.*, "Observation of $e^+e^- \to \pi^0\pi^0 h_c$ and a Neutral Charmoniumlike Structure $Z_c(4020)^0$," *Phys. Rev. Lett.*, vol. 113, p. 212002, Nov 2014.

[29] T. Xiao, S. Dobbs, A. Tomaradze, and K. K. Seth, "Observation of the charged hadron $Z_c(3900)^\pm$ and evidence for the neutral $Z_c(3900)^0$ in $e^+e^- \to \pi^0\pi^0 J/\Psi$ at $\sqrt{s}$ = 4170 *MeV*," *Phys. Lett. B*, vol. 727, no. 4–5, pp. 366 – 370, 2013.

[30] M. Ablikim *et al.*, "Observation of a Charged Charmoniumlike Structure in $e^+e^- \to (D^*\overline{D}^*)^\pm\pi^\mp$ at $\sqrt{s}$ = 4.26 GeV," *Phys. Rev. Lett.*, vol. 112, p. 132001, Apr 2014.

[31] R. Mizuk *et al.*, "Observation of two resonancelike structures in the $\pi^+\chi_{c1}$ mass distribution in exclusive $\overline{B}^0 \to K^-\pi^+\chi_{c1}$ decays," *Phys. Rev. D*, vol. 78, p. 072004, Oct 2008.

[32] J. P. Lees *et al.*, "Search for the $Z_1(4050)^+$ and $Z_2(4250)^+$ states in $\overline{B}^0 \to \chi_{c1}K^-\pi^+$ and $B^+ \to \chi_{c1}K^0_S\pi^+$," *Phys. Rev. D*, vol. 85, p. 052003, Mar 2012.

[33] F. James and M. Roos, "Minuit - a system for function minimization and analysis of the parameter errors and correlations," *Computer Physics Communications*, vol. 10, no. 6, pp. 343 – 367, 1975.

[34] R. Brun and F. Rademakers, "{ROOT} — An object oriented data analysis framework," *Nucl. Instrum. Meth.*, vol. A389, no. 1–2, pp. 81 – 86, 1997. New Computing Techniques in Physics Research V.

[35] J. P. Cummings and D. P. Weygand, "An Object-Oriented Approach to Partial Wave Analysis," *arXiv:physics/0309052*, Sep 2003.

[36] S. Neubert and B. Grube, "Hompage of the ROOTPWA software." `https://github.com/ROOTPWA-Maintainers/ROOTPWA`, Feb. 2016.

[37] P. Abbon *et al.*, "The COMPASS experiment at CERN," *Nucl. Instrum. Meth.*, vol. A577, pp. 455–518, 2007.

[38] T. Degener, *Analyse von Endzuständen der Antiproton-Proton Annihilation im Fluge*. Ph.D. Thesis, University of Bochum, 1999.

[39] E. Aker *et al.*, "The crystal barrel spectrometer at LEAR," *Nucl. Instrum. Meth.*, vol. A321, no. 1–2, pp. 69 – 108, 1992.

[40] N. Berger, "Partial wave analysis at BES III harnessing the power of GPUs," *AIP Conf.Proc.*, vol. 1374, pp. 553–556, 2011.

[41] Latham, T. and Back, J. and Harrison P., "Laura++ homepage." `https://laura.hepforge.`

org/, Feb. 2016.

[42] B. Aubert *et al.*, "The BABAR detector," *Nucl. Instrum. Meth.*, vol. A479, no. 1, pp. 1 – 116, 2002. Detectors for Asymmetric B-factories.

[43] The LHCb Collaboration, "The LHCb Detector at the LHC," *JINST*, vol. 3, no. 08, p. S08005, 2008.

[44] M. Shepherd, "Amptools homepage." http://sourceforge.net/projects/amptools/, Feb 2016.

[45] M. Dugger *et al.*, "A study of meson and baryon decays to strange final states with GlueX in Hall D (A proposal to the 39th Jefferson Lab Program Advisory Committee)," *arXiv:1210.4508*, 2012.

[46] R. Berlich and M. Kunze, "Parallel evolutionary algorithms," *Nucl. Instrum. Meth.*, vol. 502, no. 2–3, pp. 467 – 470, 2003. Proceedings of the {VIII} International Workshop on Advanced Computing and Analysis Techniques in Physics Research.

[47] J. G. Siek, L.-Q. Lee, and A. Lumsdaine, *Boost Graph Library, The: User Guide and Reference Manual*. Addison-Wesley Professional, Dec 2001.

[48] K. Ram, "Git can facilitate greater reproducibility and increased transparency in science," *Source Code for Biology and Medicine*, vol. 8, no. 1, 2013.

[49] D. van Heesch, "Doxygen." http://www.stack.nl/~{}dimitri/doxygen/index.html, Oct 2012.

[50] "Github URL." https://github.com, Jul 2014.

[51] "GNU General Public License." http://www.gnu.org/licenses/gpl.html, Jul 2007.

[52] R. E. Johnson and B. Foote, "Designing Reusable Classes," *Journal of Object-Oriented Programming*, vol. 1, pp. 22–35, 1988.

[53] W. Pree, *Komponentenbasierte Softwareentwicklung mit Frameworks*. dpunkt, 1997.

[54] M. Jacob and G. Wick, "On the general theory of collisions for particles with spin," *Annals of Physics*, vol. 7, no. 4, pp. 404 – 428, 1959.

[55] B. Zou and D. Bugg, "Covariant tensor formalism for partial wave analyses of psi decay to mesons," *Eur. Phys. J.*, vol. A16, pp. 537–547, 2003.

[56] S. Chung, J. Brose, R. Hackmann, E. Klempt, S. Spanier, *et al.*, "Partial wave analysis in K matrix formalism," *Annalen Phys.*, vol. 4, pp. 404–430, 1995.

[57] D. J. Herndon, P. Söding, and R. J. Cashmore, "Generalized isobar model formalism," *Phys. Rev. D*, vol. 11, pp. 3165–3182, Jun 1975.

[58] X. Liu, B. Zhang, and S.-L. Zhu, "The hidden charm decay of and final state interaction effects," *Phys. Lett. B*, vol. 645, no. 2–3, pp. 185 – 188, 2007.

[59] A. W. F. Edwards, *Likelihood*. Cambridge [Eng.]University Press, 1972.

[60] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, *Design Patterns – Elements of Reusable Object-Oriented Software*. Amsterdam: Addison-Wesley Longman, 1 ed., 1995. 37. Reprint (2009).

[61] E. Wigner, *Gruppentheorie und ihre Anwendung auf die Quantenmechanik der Atomspektren*. Wissenschaft (Braunschweig, Germany), J.W. Edwards, 1931.

[62] J. Blatt and V. Weisskopf, *Theoretical nuclear physics*. Wiley, 1952.

[63] M. Williams, "Numerical Object Oriented Quantum Field Theory Calculations," *Comput. Phys. Commun.*, vol. 180, pp. 1847–1852, 2009.

[64] D. Landau and K. Binder, *A Guide to Monte Carlo Simulations in Statistical Physics*. Cambridge University Press, 2013.

[65] Wolfram Research, Inc., "Mathematica." `https://www.wolfram.com/mathematica/`, Jul 2012.

[66] J. C. Giarratano and G. Riley, *Expert systems: principles and programming*. Thomson Course Technology, Oct 2005.

[67] M. Ablikim, J. Bai, Y. Ban, J. Bian, X. Cai, *et al.*, "Partial wave analyses of $J/\psi \to \gamma\pi^+\pi^-$ and $\gamma\pi^0\pi^0$," *Phys. Lett.*, vol. B642, pp. 441–448, 2006.

[68] P. Weidenkaff, *Analysis of the decay $D^0 \to K^0_S K^+ K^-$ with the BESIII experiment*. Ph.D. Thesis in preparation, University of Mainz, 2016.

[69] S. Pflüger, *A Luminosity Fit for Panda and Development of an Helicity Amplitude Generator for ComPWA*. Ph.D. Thesis in preparation, University of Mainz, 2016.

A Appendix

A.1 Helicity Amplitude Calculation

Following "A primer on partial wave analysis" [6], a short recap is given on the shape of resonant behavior in two-body decays. In an scattering process, one can introduce partial waves by expanding the initial state Φ_i in terms of Legendre polynomials P_l to separate angular from radial wave function U_l:

$$\Phi_i = e^{\mathrm{i}kz} = \sum_{l=0}^{\infty} U_l(r) P_l(\cos\theta) \tag{A.1}$$

U_l cab be parametrized in terms of a phase δ_l and an inelasticity η_l. The scattering wave function, which is the difference between outgoing and incoming wave, gives then:

$$\Phi_S = \Psi_f - \Psi_i = \frac{1}{k} \sum_{l=0}^{\infty} (2l+1) \frac{\eta_l e^{2\mathrm{i}\delta_l} - 1}{2\mathrm{i}} P_l(\cos\theta) \frac{e^{\mathrm{i}kr}}{r} \tag{A.2}$$

Applying Fermi's Golden Rule, the total differential cross section can be written as:

$$\frac{d\sigma}{d\phi d\cos\theta} = \frac{1}{k^2} \left| \sum_{l=0}^{\infty} (2l+1) \frac{\eta_l e^{2\mathrm{i}\delta_l} - 1}{2\mathrm{i}} P_l(\cos\theta) \right|^2 \tag{A.3}$$

The dynamical part of the Amplitude is the T-Matrix T_l:

$$T_l = \frac{\eta_l e^{2\mathrm{i}\delta_l} - 1}{2\mathrm{i}} \tag{A.4}$$

As example, in elastic scattering of spinless particles via a resonance of spin $J = l$ this simplifies to:

$$T = \frac{1}{cot\delta - \mathrm{i}} \tag{A.5}$$

Close to the resonance energy E_R, $\delta \approx \frac{\pi}{2}$ holds and one can expand $cot\delta$:

$$cot\delta(E) = cot\delta(E_R) + (E - E_R) \left[\frac{d}{dE} \cot\delta(E) \right]_{E=E_R} \tag{A.6}$$

When defining $\frac{2}{\Gamma}$ as the first derivative of $cot\delta$, one gets:

$$cot\delta(E) = 0 + (E - E_R)(-\frac{2}{\Gamma}) \tag{A.7}$$

Therefore, one obtains the Breit-Wigner shape for the dynamical behavior at $|E - E_R| \approx \Gamma \ll E_R$:

$$T = \frac{\frac{\Gamma}{2}}{(E_R - E) - \mathrm{i}\frac{\Gamma}{2}} \tag{A.8}$$

Figure 3.7 shows different visualizations of a Breit-Wigner function. One can see a phase shift of π around the resonance pole and the asymmetry of the shape of $|T|^2$.

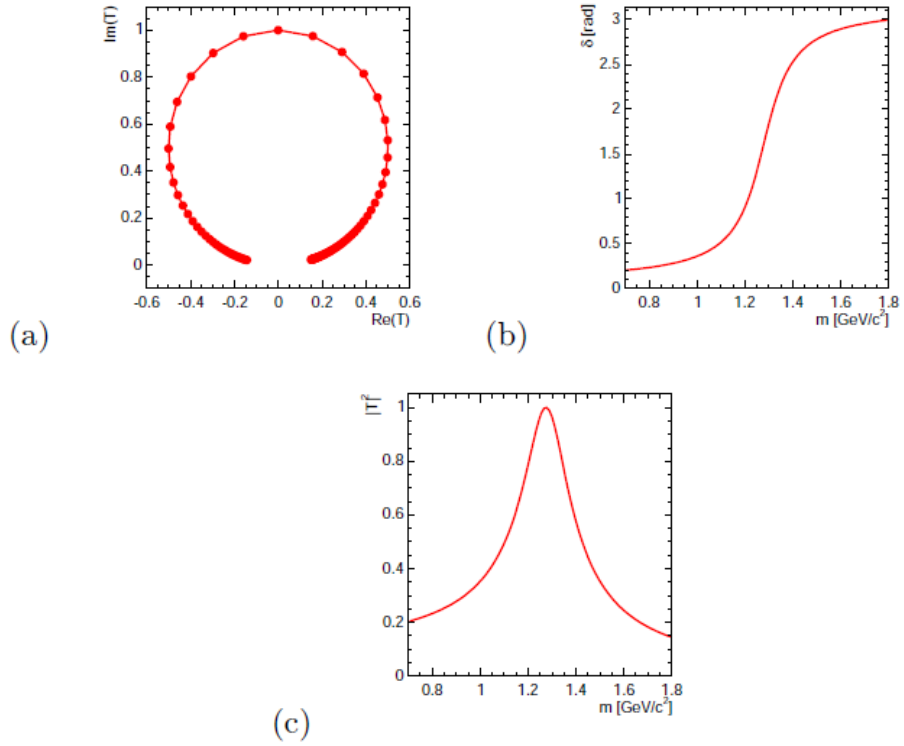

Figure A.1: Simple relativistic Breit-Wigner, from [6]. (a) Argand plot: imaginary versus real part, (b) phase versus mass, (c) absolute value squared versus mass.

A good summary of formalisms used in amplitude analyses and their characteristics can be found in [6] as well. Amplitude calculations on this work were based solely on the helicity formalism. As described in [6], one obtains the following helicity state

$$|JM\lambda_s\lambda_t >= N_J \int d\Omega D^{J*}_{M,\lambda_s-\lambda_t}|\Omega, s\lambda_s t\lambda_t > \tag{A.9}$$

for two particles s and t with spin projections λ in a rest system with spin J and projection M. Ω is the decay angle and D the Wigner D-functions. The normalization is chosen to fulfill completeness:

$$N_J = \sqrt{\frac{2J+1}{4\pi}} \tag{A.10}$$

To construct the amplitude, one uses two-particle states and sums over all unobservable spin-projections

$$A = \sum_{\lambda_s,\lambda_t} < \vec{p}_s, s\lambda_s| < -\vec{p}_s, t\lambda_t|\mathcal{M}|JM > \tag{A.11}$$

using the helicity states, one obtains

$$A^{JM}_{\lambda_s\lambda_t} = N_J f_{\lambda_s\lambda_t} D^{J*}_{M,\lambda_s-\lambda_t}(\Omega_s) \tag{A.12}$$

with the decay operator $\mathcal{M}$ contained in the helicity amplitude:

$$N_J f_{\lambda_s\lambda_t} = \sqrt{\frac{4\pi}{\rho_s}(2J+1)} < JM\lambda_s m_t|\mathcal{M}|JM > \tag{A.13}$$

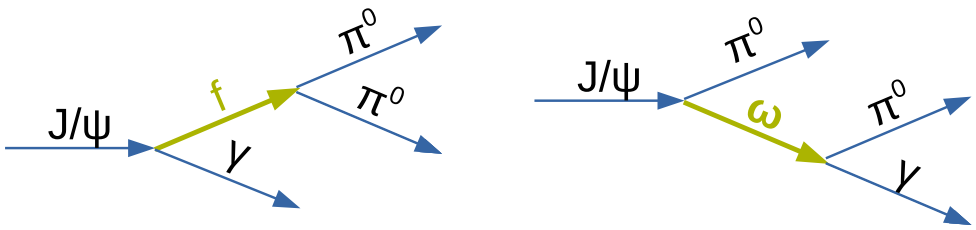

Figure A.2: Sketch of decays in the $J/\psi \to \gamma\pi^0\pi^0$ channel.

In order to understand the nature of the $J/\psi \to \gamma\pi^0\pi^0$ channel, the helicity amplitudes for the possible waves were calculated. Table A.1 shows the quantum numbers and helicities λ of all contributing particles and resonances.

Table A.1: Quantum numbers and helicities

	Spin J	Parity P	C-Parity C	helicities
J/ψ	1	-	-	$0, \pm 1$
f_0	0	+	+	0
f_2	2	+	+	$0, \pm 1, \pm 2$
ω	1	-	-	$0, \pm 1$
π^0	0	-	+	0
γ	1	-	-	± 1

The helicity formalism [54] was used to describe the behavior of these resonances in the simulation. As the resonances with same spin in the same state behave the same, this leaves three helicity amplitudes A which need to be to be calculated:

$$A_0 = A_{J/\Psi} \cdot A_{f_0} = A^{1M}_{\lambda f_0 \lambda\gamma} \cdot A^{0\lambda f_0}_{00} \tag{A.14}$$

$$A_2 = A_{J/\Psi} \cdot A_{f_2} = A^{1M}_{\lambda f_2 \lambda\gamma} \cdot A^{2\lambda f_2}_{00} \tag{A.15}$$

$$A_\omega = A_{J/\Psi} \cdot A_\omega = A^{1M}_{\lambda\omega\lambda\gamma} \cdot A^{1\lambda\omega}_{00} \tag{A.16}$$

Using equation A.12 and equation A.13 gives:

$$A_0 = N_1 \cdot F^1_{\lambda f_0 \lambda\gamma} \cdot D^{1*}_{M(\lambda f_0 - \lambda\gamma)}(\psi_a, \theta_a) \cdot \sqrt{1} \cdot a_{00} \cdot D^{0*}_{M0}(\psi_b, \theta_b) \tag{A.17}$$

$$A_2 = N_1 \cdot F^1_{\lambda f_2 \lambda\gamma} \cdot D^{1*}_{M(\lambda f_2 - \lambda\gamma)}(\psi_a, \theta_a) \cdot \sqrt{5} \cdot a_{20} \cdot D^{2*}_{M0}(\psi_b, \theta_b) \tag{A.18}$$

$$A_\omega = N_1 \cdot F^1_{\lambda\omega\lambda\pi^0} \cdot D^{1*}_{M0}(\psi_a, \theta_a) \cdot N_1 * F^1_{0\lambda\gamma} \cdot D^{1*}_{M(0-\lambda\gamma)}(\psi_b, \theta_b) \tag{A.19}$$

The decay amplitude of the f's is one dimensional, as the two π^0 have fixed helicities which leaves only one decay amplitude. Else, all possible combinations of helicities, see table A.1, have to be taken in to account and a matrix of equations is formed for the amplitude. The matrix can be collapsed when calculating the intensity $I = |A^2|$. As these matrices manipulations are quite extensive, the Mathematica [65] package was used. The complete notebooks can be found at the end of this section. The intensity of the resonances is calculated to:

$$I_0 = a^2_{00} \cdot (2a^2_{01} + 2.828 a_{01} a_{21} + a^2_{21}) \tag{A.20}$$

$$I_\omega = a^4_{11} \cdot (3.375 + 1.125 cos(2\theta)) \tag{A.21}$$

Where a_{ls} stands for the dynamical behavior. The formula for intensity I_0 is not shown as it is simply too long. To extract the decay-angle θ of the resonance one integrates over the

remaining parameter. This yields:

$$I_0(\theta) = a_{00}^2 \cdot (39.48a_{01}^2 + 55.83a_{01}a_{21} + 19.74a_{21}^2) \tag{A.22}$$

$$\begin{aligned} I_2(\theta) = a_{20}^2 \cdot ((&34.37a_{23}^2 + 21.06a_{23}a_{43} + 31.26a_{40}^2) \\ &+ cos(2\theta)(23.26a_{23}^2 + 40.29a_{23}a_{43} + 17.45a_{43}^2) \\ &+ cos(4\theta)(10.05a_{23}^2 + 55.86a_{23}a_{43} + 1.982a_{43}^2) \end{aligned} \tag{A.23}$$

$$I_\omega(\theta) = 66.62a_{11}^4 + 22.21cos(2\theta) \tag{A.24}$$

As one can see, the decay angle distribution of the f_0 resonances is flat. For the f_2 we get the behavior shown in figure A.3, for the ω the behavior shown in figure A.4.

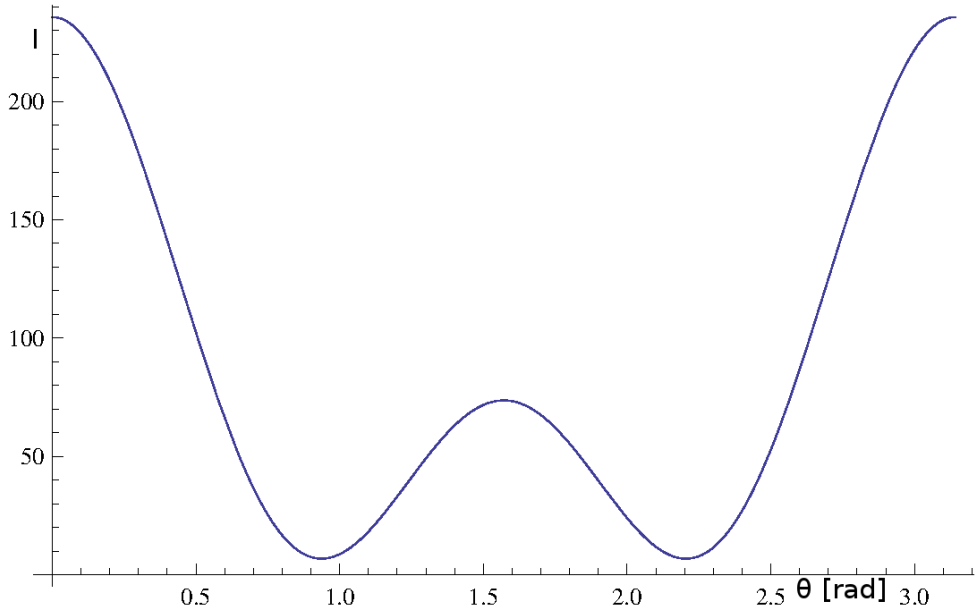

Figure A.3: Distribution of f_2 decay angle.

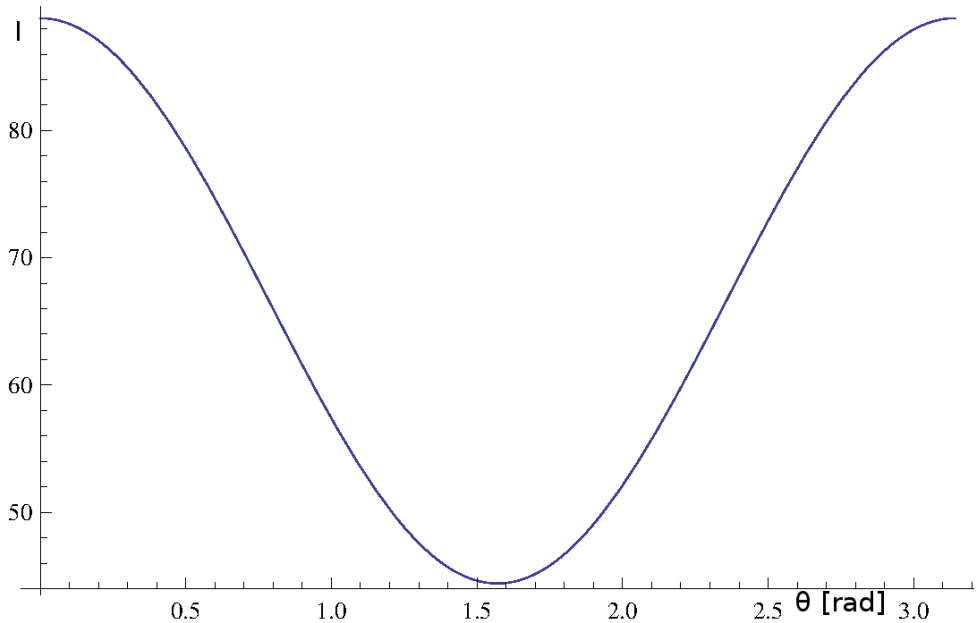

Figure A.4: Distribution of ω decay angle.

This information was used as input and cross-check for the model implemented in ComPWA,

see section 3.8. As described in the previous chapter, the model and therefore these calculated distributions were also used to generate the toy Monte Carlo data the analyses in this chapters are performed on.
On the following pages, the complete Mathematica [65] notebooks used to calculate the helicity amplitudes of the $J/\psi \to \gamma\pi^0\pi^0$ example are shown. Some of the output matrices of the formalism are to long to be shown, but can be reproduces using the shown Mathematica operations. For the plots of the angular distributions at the end of each amplitude calculation only the biggest contributions were used.

Decay Amplitude of f_0

```
SetDirectory["\\\\fs02\\mamichel$\\Dokumente\\JPSI"]
<< djmn.m

\\\\fs02\\mamichel$\\Dokumente\\JPSI

...djmn-pack by jl,kp Version 2.0

Af0 = {Times[Heli[0, 0, 0, 0, 0, +1, −1, −1], Djmn[0, 0, 0, Phi2, Theta2]]}

(*AJ[M_, h1_, h2_] = Times[Heli[1, 0, 1, 0, h2, −1], Djmn[1, M, −h2, Phi, Theta]]*)
AJ = {
{{Times[Heli[1, 0, 1, 0, −1, −1, −1, +1], Djmn[1, −1, 1, Phi, Theta]]},
{0},
{Times[Heli[1, 0, 1, 0, 1, −1, −1, +1], Djmn[1, −1, −1, Phi, Theta]]}},

{{Times[Heli[1, 0, 1, 0, −1, −1, −1, +1], Djmn[1, 0, 1, Phi, Theta]]},
{0},
{Times[Heli[1, 0, 1, 0, 1, −1, −1, +1], Djmn[1, 0, −1, Phi, Theta]]}},

{{Times[Heli[1, 0, 1, 0, −1, −1, −1, +1], Djmn[1, 1, 1, Phi, Theta]]},
{0},
{Times[Heli[1, 0, 1, 0, 1, −1, −1, +1], Djmn[1, 1, −1, Phi, Theta]]}}
}

$Assumptions = Element[Phi, Reals]&&Element[Phi2, Reals]&&Element[Theta, Reals]
&&Element[Theta2, Reals]&&Element[a20, Reals]&&Element[a01, Reals]
&&Element[a02, Reals]&&Element[a03, Reals]&&Element[a21, Reals]
&&Element[a22, Reals]&&Element[a23, Reals];

Atot = AJ.Af0

TA = Simplify[ComplexExpand[ConjugateTranspose[TrigToExp[Atot]]]]
Imat = Dot[Atot, TA]

Intens = TrigToExp[Imat[[1, 1]] + Imat[[2, 2]] + Imat[[3, 3]]]//FullSimplify
```

a[0, 0]^2 ((2.0)a[0, 1]^2 + (2.82843)a[0, 1]a[2, 1]+
(1.0)a[2, 1]^2) + a[0, 0]^2 ((5.551115123125783*^-17)a[0, 1]^2+
(5.551115123125783*^-17)a[0, 1]a[2, 1]+
(2.7755575615628914*^-17)a[2, 1]^2) Cos[2Theta]

Itheta = Integrate[Integrate[Intens, {Phi2, 0, 2 * π}], {Theta, 0, π}]//FullSimplify

Iphi = Integrate[Integrate[Intens, {Theta2, 0, π}], {Theta, 0, π}]//FullSimplify

a[0, 0]^2 ((39.4784)a[0, 1]^2 + (55.8309)a[0, 1]a[2, 1]+
(19.7392)a[2, 1]^2)

a[0, 0]^2 ((19.7392)a[0, 1]^2 + (27.9155)a[0, 1]a[2, 1]+
(9.8696)a[2, 1]^2)

Decay Amplitude of f_2

```
SetDirectory["\\\\fs02\\mamichel$\\Dokumente\\JPSI"]
<< djmn.m
```

```
\\\\fs02\\mamichel$\\Dokumente\\JPSI
...djmn-pack by jl,kp Version 2.0
```

```
Af2 = {
Times[Heli[2, 0, 0, 0, 0, +1, -1, -1], Djmn[2, -2, 0, Phi2, Theta2]],
Times[Heli[2, 0, 0, 0, 0, +1, -1, -1], Djmn[2, -1, 0, Phi2, Theta2]],
Times[Heli[2, 0, 0, 0, 0, +1, -1, -1], Djmn[2, 0, 0, Phi2, Theta2]],
Times[Heli[2, 0, 0, 0, 0, +1, -1, -1], Djmn[2, 1, 0, Phi2, Theta2]],
Times[Heli[2, 0, 0, 0, 0, +1, -1, -1], Djmn[2, 2, 0, Phi2, Theta2]]
}

(*AJ[M_, h1_, h2_] = Times[Heli[1, 2, 1, h1, h2, -1], Djmn[1, M, h1 - h2, Phi, Theta]]*)
AJ = {
{{Times[Heli[1, 1, 2, -1, -2, -1, -1, +1], Djmn[1, -1, -1 + 2, Phi, Theta]],
Times[Heli[1, 1, 2, -1, -1, -1, -1, +1], Djmn[1, -1, -1 + 1, Phi, Theta]],
Times[Heli[1, 1, 2, -1, 0, -1, -1, +1], Djmn[1, -1, -1 - 0, Phi, Theta]],
Times[Heli[1, 1, 2, -1, 1, -1, -1, +1], Djmn[1, -1, -1 - 1, Phi, Theta]],
Times[Heli[1, 1, 2, -1, 2, -1, -1, +1], Djmn[1, -1, -1 - 2, Phi, Theta]]},
{0, 0, 0, 0, 0},
{Times[Heli[1, 1, 2, 1, -2, -1, -1, +1], Djmn[1, -1, 1 + 2, Phi, Theta]],
Times[Heli[1, 1, 2, 1, -1, -1, -1, +1], Djmn[1, -1, 1 + 1, Phi, Theta]],
Times[Heli[1, 1, 2, 1, 0, -1, -1, +1], Djmn[1, -1, 1 - 0, Phi, Theta]],
Times[Heli[1, 1, 2, 1, 1, -1, -1, +1], Djmn[1, -1, 1 - 1, Phi, Theta]],
Times[Heli[1, 1, 2, 1, 2, -1, -1, +1], Djmn[1, -1, 1 - 2, Phi, Theta]]}},

{{Times[Heli[1, 1, 2, -1, -2, -1, -1, +1], Djmn[1, 0, -1 + 2, Phi, Theta]],
Times[Heli[1, 1, 2, -1, -1, -1, -1, +1], Djmn[1, 0, -1 + 1, Phi, Theta]],
Times[Heli[1, 1, 2, -1, 0, -1, -1, +1], Djmn[1, 0, -1 - 0, Phi, Theta]],
Times[Heli[1, 1, 2, -1, 1, -1, -1, +1], Djmn[1, 0, -1 - 1, Phi, Theta]],
Times[Heli[1, 1, 2, -1, 2, -1, -1, +1], Djmn[1, 0, -1 - 2, Phi, Theta]]},
```

```
{0, 0, 0, 0, 0},
{Times[Heli[1, 1, 2, 1, −2, −1, −1, +1], Djmn[1, 0, 1 + 2, Phi, Theta]],
Times[Heli[1, 1, 2, 1, −1, −1, −1, +1], Djmn[1, 0, 1 + 1, Phi, Theta]],
Times[Heli[1, 1, 2, 1, 0, −1, −1, +1], Djmn[1, 0, 1 − 0, Phi, Theta]],
Times[Heli[1, 1, 2, 1, 1, −1, −1, +1], Djmn[1, 0, 1 − 1, Phi, Theta]],
Times[Heli[1, 1, 2, 1, 2, −1, −1, +1], Djmn[1, 0, 1 − 2, Phi, Theta]]}},

{{Times[Heli[1, 1, 2, −1, −2, −1, −1, +1], Djmn[1, 1, −1 + 2, Phi, Theta]],
Times[Heli[1, 1, 2, −1, −1, −1, −1, +1], Djmn[1, 1, −1 + 1, Phi, Theta]],
Times[Heli[1, 1, 2, −1, 0, −1, −1, +1], Djmn[1, 1, −1 − 0, Phi, Theta]],
Times[Heli[1, 1, 2, −1, 1, −1, −1, +1], Djmn[1, 1, −1 − 1, Phi, Theta]],
Times[Heli[1, 1, 2, −1, 2, −1, −1, +1], Djmn[1, 1, −1 − 2, Phi, Theta]]},
{0, 0, 0, 0, 0},
{Times[Heli[1, 1, 2, 1, −2, −1, −1, +1], Djmn[1, 1, 1 + 2, Phi, Theta]],
Times[Heli[1, 1, 2, 1, −1, −1, −1, +1], Djmn[1, 1, 1 + 1, Phi, Theta]],
Times[Heli[1, 1, 2, 1, 0, −1, −1, +1], Djmn[1, 1, 1 − 0, Phi, Theta]],
Times[Heli[1, 1, 2, 1, 1, −1, −1, +1], Djmn[1, 1, 1 − 1, Phi, Theta]],
Times[Heli[1, 1, 2, 1, 2, −1, −1, +1], Djmn[1, 1, 1 − 2, Phi, Theta]]}}
}

$Assumptions = Element[Phi, Reals]&&Element[Phi2, Reals]&&Element[Theta, Reals]
&&Element[Theta2, Reals]&&Element[a20, Reals]&&Element[a01, Reals]
&&Element[a02, Reals]&&Element[a03, Reals]&&Element[a21, Reals]
&&Element[a22, Reals]&&Element[a23, Reals];

Atot = AJ.Af2

TA = Simplify[ComplexExpand[ConjugateTranspose[TrigToExp[Atot]]]]
Imat = Dot[Atot, TA]

Intens = TrigToExp[Imat[[1, 1]] + Imat[[2, 2]] + Imat[[3, 3]]]//FullSimplify
```

Result not shown, too long

```
Itheta = Integrate[Integrate[Intens, {Phi2, 0, 2 ∗ π}], {Theta, 0, π}]//FullSimplify
```

Iphi = Integrate[Integrate[Intens, {Theta2, 0, π}], {Theta, 0, π}]//FullSimplify

a[2, 0]^2 ((34.3674)a[2, 3]^2 + (21.0631)a[2, 3]a[4, 3]+
(31.3272)a[4, 3]^2) + a[2, 0]^2 (((23.2641)a[2, 3]^2 + (40.2945)
a[2, 3]a[4, 3] + (17.4481)a[4, 3]^2) Cos[2Theta2] + ((10.0458)
(55.8629)a[2, 3]a[4, 3] + (1.98273)a[4, 3]^2) Cos[4Theta2])

a[2, 0]^2 ((17.1837)a[2, 3]^2 + (10.5315)a[2, 3]a[4, 3]+
(15.6636)a[4, 3]^2) + a[2, 0]^{2}a[2, 3]Cos[Phi2](−5.1363103359521173*^-17
a[2, 3] − (8.560517226586862*^-18)a[4, 3] + ((8.560517226586862*^-17i)
a[2, 3] − (1.5653517214330262*^-16i)a[4, 3])Sin[Phi2]) + a[2, 0]^2
(((2.5681551679760587*^-17)a[2, 3]^2 + (7.826758607165131*^-17)
a[2, 3]a[4, 3] + (3.424206890634745*^-17)a[4, 3]^2) Cos[2Phi2]+
((0.@ − 3.424206890634745*^-17i)a[2, 3]^2 − (8.560517226586862*^-18i)
a[2, 3]a[4, 3] + (6.84841378126949*^-17i)a[4, 3]^2) Sin[Phi2])

(* Full Amplitude Theta *)
Plot[(34.36737246807901 + 21.063058622449272 + 31.327181826672025)+
((23.264067516853487 + 40.29454692990296 + 17.448050637640115)Cos[2Theta2]+
(10.045847336823096 + 55.862894607365476 + 1.9827330270045582)Cos[4Theta2])
, {Theta2, 0, π}]

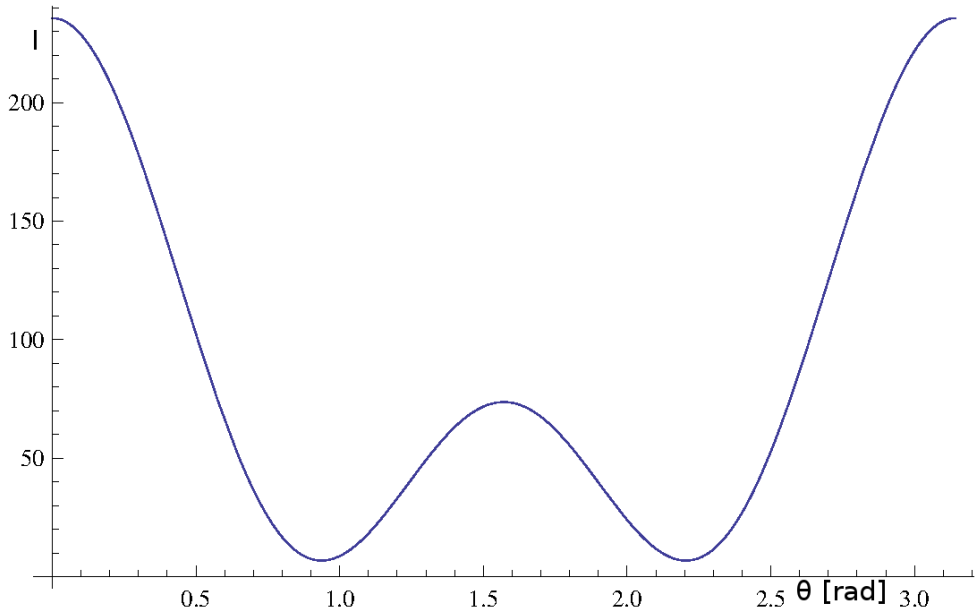

Figure A.5: θ angle distribution of spin 2 resonances in $J/\psi \to \gamma f_2 \to \gamma\pi\pi$

```
(* Full Amplitude Phi *)
Plot[(17.183686234039506 + 10.531529311224636 + 15.663590913336012),
{Phi2, 0, 2 * π}]
```

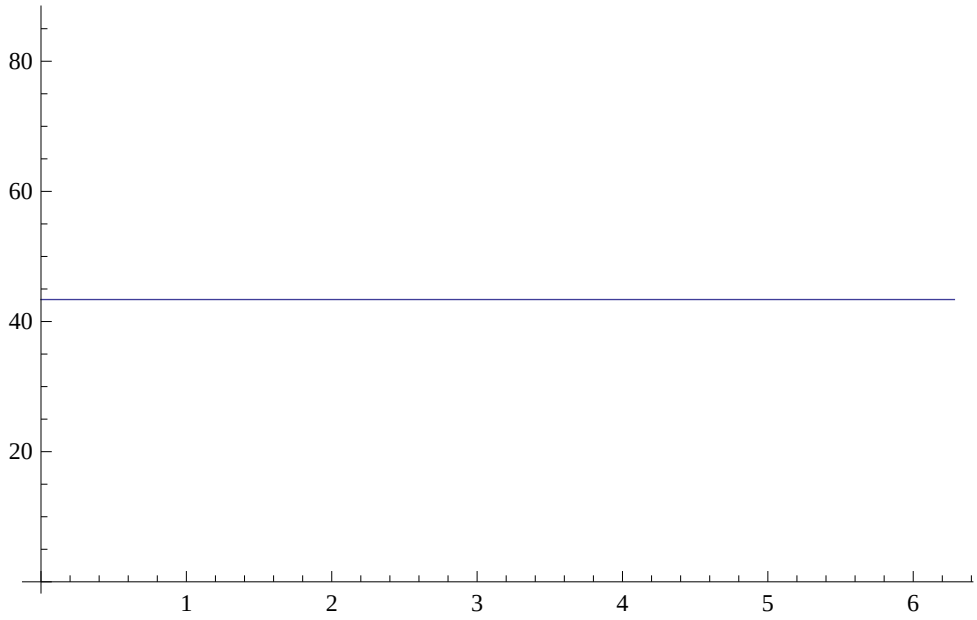

Figure A.6: ϕ angle distribution of spin 2 resonances in $J/\psi \to \gamma f_2 \to \gamma\pi\pi$

Decay Amplitude of ω

```
SetDirectory["\\\\fs02\\mamichel$\\Dokumente\\JPSI"]
<< djmn.m

\\\\fs02\\mamichel$\\Dokumente\\JPSI

...djmn-pack by jl,kp Version 2.0

Ao = {
{Times[Heli[1, 0, 1, 0, −1, −1, −1, −1], Djmn[1, 1, 1, Phi2, Theta2]],
Times[Heli[1, 0, 1, 0, 1, −1, −1, −1], Djmn[1, 1, −1, Phi2, Theta2]]},
{Times[Heli[1, 0, 1, 0, −1, −1, −1, −1], Djmn[1, 0, 1, Phi2, Theta2]],
Times[Heli[1, 0, 1, 0, 1, −1, −1, −1], Djmn[1, 0, −1, Phi2, Theta2]]},
{Times[Heli[1, 0, 1, 0, −1, −1, −1, −1], Djmn[1, −1, 1, Phi2, Theta2]],
Times[Heli[1, 0, 1, 0, 1, −1, −1, −1], Djmn[1, −1, −1, Phi2, Theta2]]}
}

(*AJ = Times[Heli[1, 1, 0, h1, 0, −1], Djmn[1, M, h1, Phi, Theta]]*)
AJ = {
{Times[Heli[1, 1, 0, −1, 0, −1, −1, −1], Djmn[1, −1, −1, Phi, Theta]],
Times[Heli[1, 1, 0, 0, 0, −1, −1, −1], Djmn[1, −1, 0, Phi, Theta]],
Times[Heli[1, 1, 0, 1, 0, −1, −1, −1], Djmn[1, −1, 1, Phi, Theta]]},

{Times[Heli[1, 1, 0, −1, 0, −1, −1, −1], Djmn[1, 0, −1, Phi, Theta]],
Times[Heli[1, 1, 0, 0, 0, −1, −1, −1], Djmn[1, 0, 0, Phi, Theta]],
Times[Heli[1, 1, 0, 1, 0, −1, −1, −1], Djmn[1, 0, 1, Phi, Theta]]},

{Times[Heli[1, 1, 0, −1, 0, −1, −1, −1], Djmn[1, 1, −1, Phi, Theta]],
Times[Heli[1, 1, 0, 0, 0, −1, −1, −1], Djmn[1, 1, 0, Phi, Theta]],
Times[Heli[1, 1, 0, 1, 0, −1, −1, −1], Djmn[1, 1, 1, Phi, Theta]]}
}

$Assumptions = Element[Phi, Reals]&&Element[Phi2, Reals]&&Element[Theta, Reals]
&&Element[Theta2, Reals]&&Element[a20, Reals]&&Element[a01, Reals]
&&Element[a02, Reals]&&Element[a03, Reals]&&Element[a21, Reals]
```

```
&&Element[a22, Reals]&&Element[a23, Reals];

Atot = AJ.Ao

TA = Simplify[ComplexExpand[ConjugateTranspose[TrigToExp[Atot]]]]
Imat = Dot[Atot, TA]

Intens = TrigToExp[Imat[[1, 1]] + Imat[[2, 2]] + Imat[[3, 3]]]//FullSimplify
```

$$a[1,1]^4\,((3.375) + ((1.125) + (1.1102230\text{*^-16})\,\text{Cos}[2\text{Phi2}])\text{Cos}[2\text{Theta2}] + \text{Cos}[2\text{Theta}]\,((2.7755576\text{*^-16}) + (5.5511151\text{*^-17})e^{-2i\text{Phi2}} + (2.2204460\text{*^-16})\text{Cos}[2\text{Theta2}]\text{Sin}[\text{Phi2}]^2) + (1.110223025\text{*^-16}i)\text{Cos}[\text{Theta}]\text{Sin}[\text{Theta}] - (2.2204460\text{*^-16})\,e^{i\text{Phi2}}\text{Cos}[\text{Phi2}]\text{Cos}[\text{Theta2}]\text{Sin}[\text{Theta}]^2)$$

```
Itheta = Integrate[Integrate[Intens, {Phi2, 0, 2 * π}], {Theta, 0, π}]//FullSimplify
Iphi = Integrate[Integrate[Intens, {Theta2, 0, π}], {Theta, 0, π}]//FullSimplify
```

$$+(66.6198)a[1,1]^4 + a[1,1]^4((-1.0957462\text{*^-15})\text{Cos}[\text{Theta2}] + (22.2066)\text{Cos}[2\text{Theta2}])$$

$$(33.3099)a[1,1]^4$$

```
Plot[66.62 + ((−1.1 * 10^ − 15)Cos[Theta2] + (22.21)Cos[2Theta2]), {Theta2, 0, π}]
```

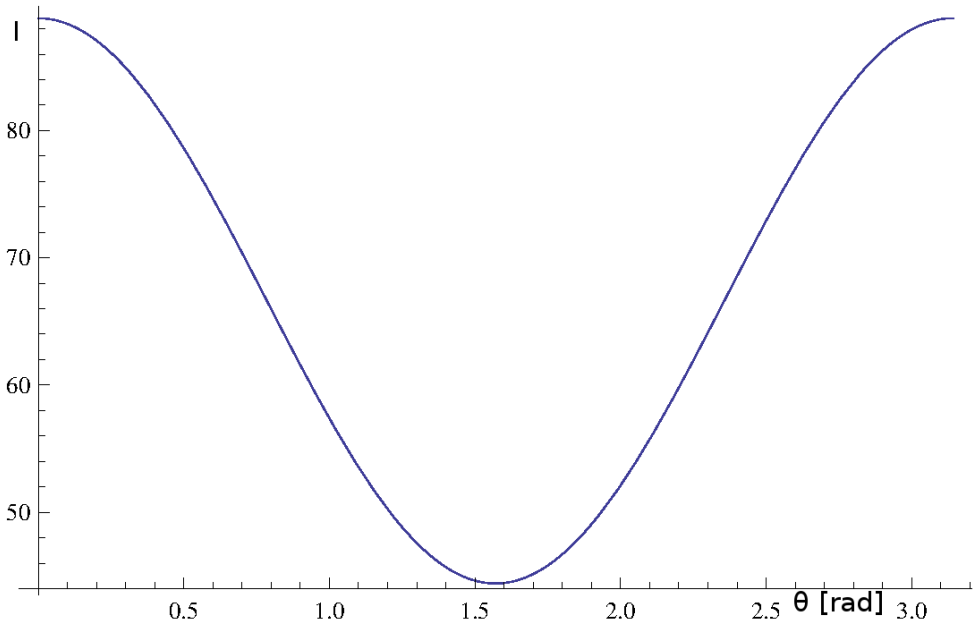

Figure A.7: θ angle distribution of ω resonance in $J/\psi \to \omega\pi \to \gamma\pi\pi$

A.2 Additional Toy Data Studies

Binned Slice Fit with more Data

Figure A.8 and table A.2 show the results of the study of the simple setup as described in 4.4.1 repeated with 4 000 000 events, 800 slices, and 800 bins as a comparison.

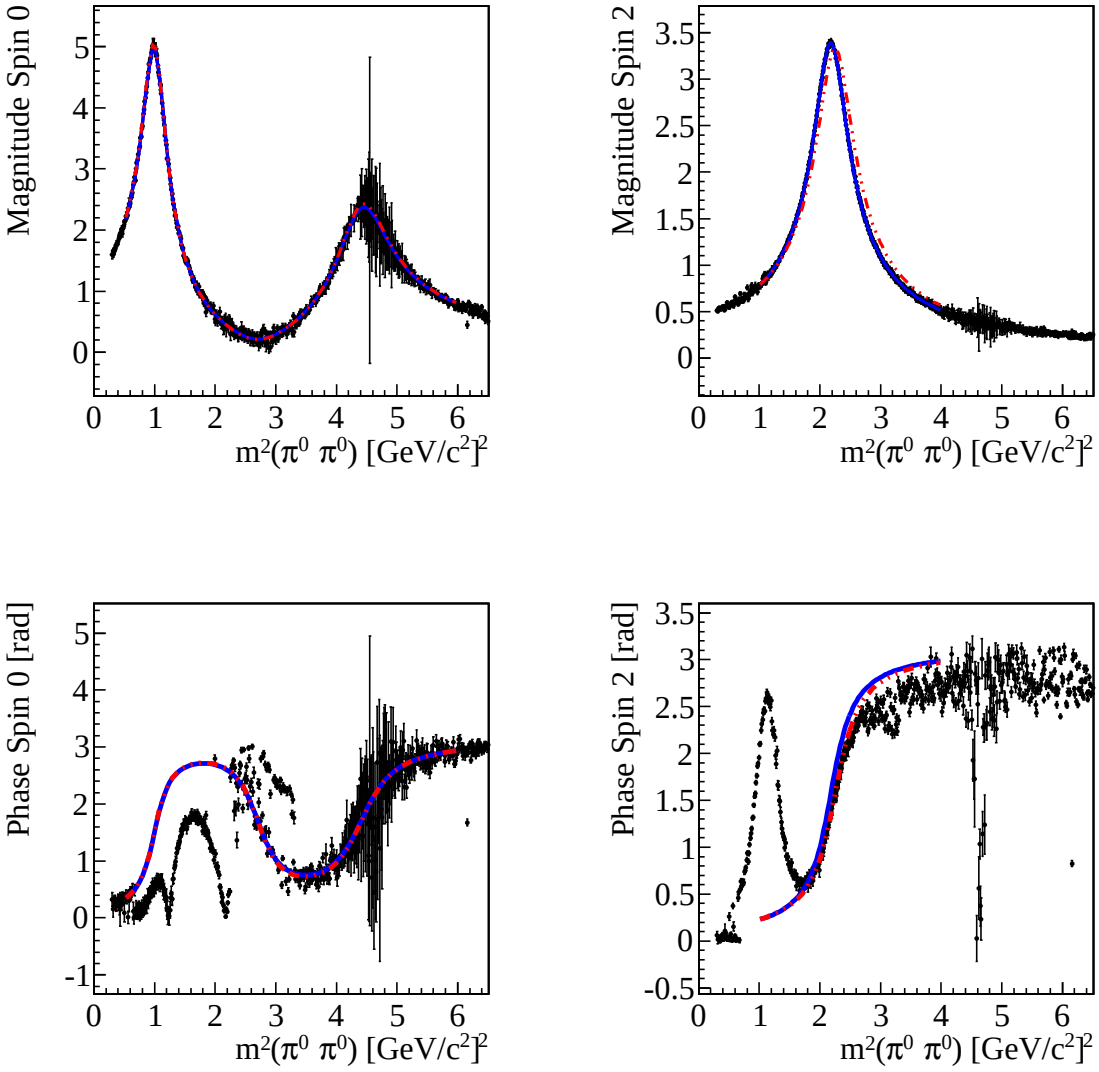

Figure A.8: Mass dependent fit coefficients c_0 and c_2 (top: magnitudes; bottom: phases) from equation 4.2 as the result of the Slice Fit of the simple toy data with 4 000 000 events. Also, the Breit-Wigner fit result (blue line), see table 4.6, and the Breit-Wigner sum with the ideal parameter setting (red dashed line) are plotted.

Table A.2: Parameter settings used for generation and extracted by the Slice Fit of the simple toy data with $4M$ events. m: mass of resonance in GeV/c^2, Γ: width of resonance in GeV.

Resonance		Generated	Fit Result	Pull
$f_0(A)$	m	1.	1.0012 ± 0.0004	3.00
	Γ	0.2	0.2013 ± 0.0013	1.00
$f_0(B)$	m	2.1	2.0997 ± 0.0013	0.23
	Γ	0.2	0.2042 ± 0.0021	2.00
	ϕ	0.0	0.0092 ± 0.0167	0.11
f_2	m	1.5	1.4764 ± 0.0004	59.00
	Γ	0.2	0.1888 ± 0.0006	18.67
ω	m	0.786	fixed	
	Γ	0.05	fixed	

Binned Slice Fit with other Phase Reference

Figure A.9 and table A.3 show the results of the study of the simple setup as described in 4.4.1 repeated with a relative phase between the scalar and tensor resonances only.

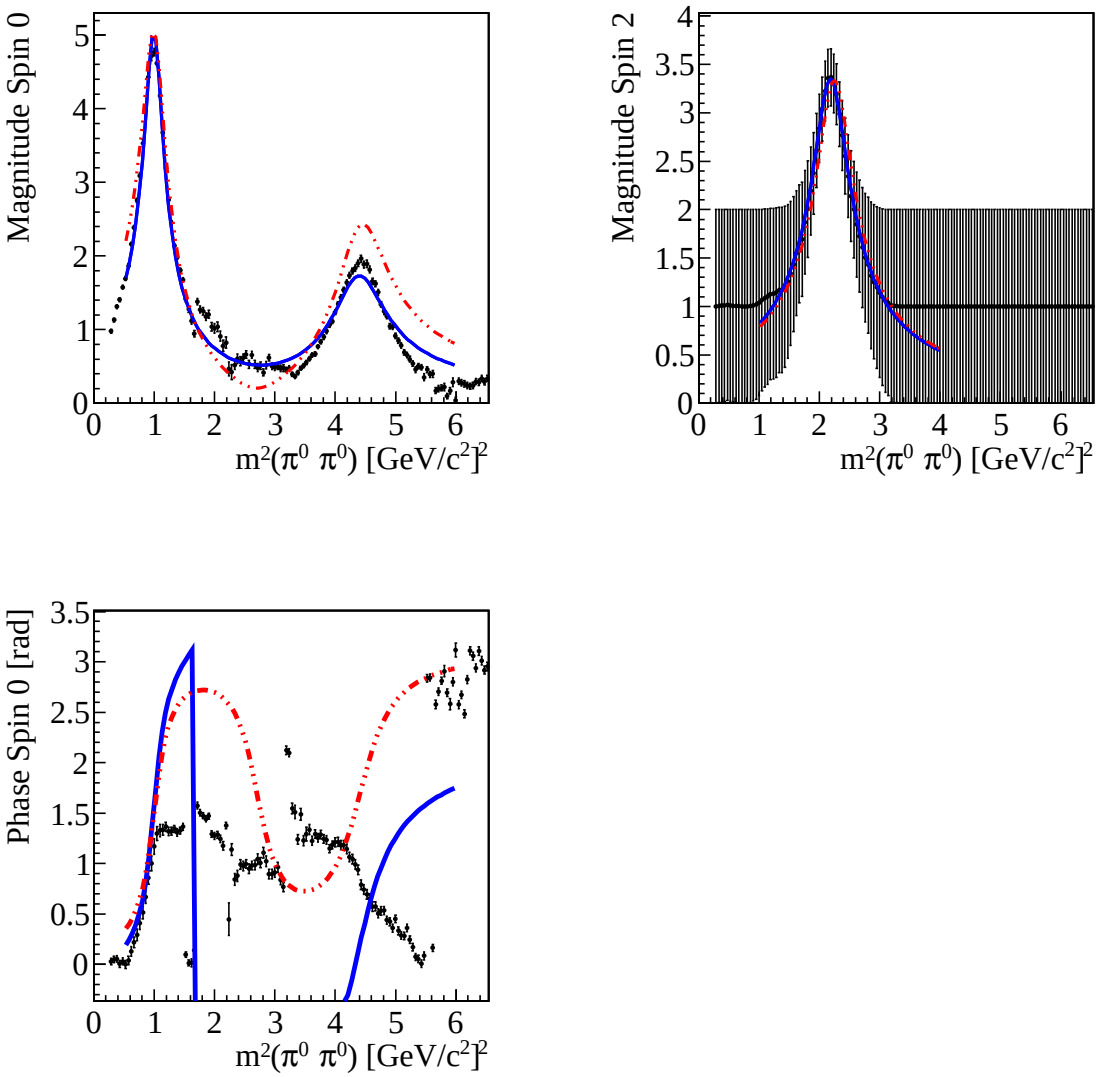

Figure A.9: Mass dependent fit coefficients c_0 and c_2 (top: magnitudes; bottom: phases) from equation 4.2 as the result of the Slice Fit of the simple toy data with only a relative phase between the resonances. Also, the Breit-Wigner fit result (blue line), see table 4.6, and the Breit-Wigner sum with the ideal parameter setting (red dashed line) are plotted.

Table A.3: Parameter settings used for generation and extracted by the Slice Fit of the simple toy data with only a relative phase between the resonances,. m: mass of resonance in GeV/c^2, Γ: width of resonance in GeV.

Resonance		Generated	Fit Result	Pull
$f_0(A)$	m	1.	0.9976 ± 0.0006	4.00
	Γ	0.2	0.1608 ± 0.0012	32.67
$f_0(B)$	m	2.1	2.0933 ± 0.0013	5.15
	Γ	0.2	0.2041 ± 0.0013	3.15
	ϕ	1.6	1.6905 ± 0.0185	4.89
f_2	m	1.5	1.4756 ± 0.0081	3.01
	Γ	0.2	0.2019 ± 0.0239	0.08
ω	m	0.786	fixed	
	Γ	0.05	fixed	

Binned Slice Fit with broader ω

Figure A.10 and table A.4 show the results of the study of the simple setup as described in 4.4.1 repeated with a very broad ω resonance.

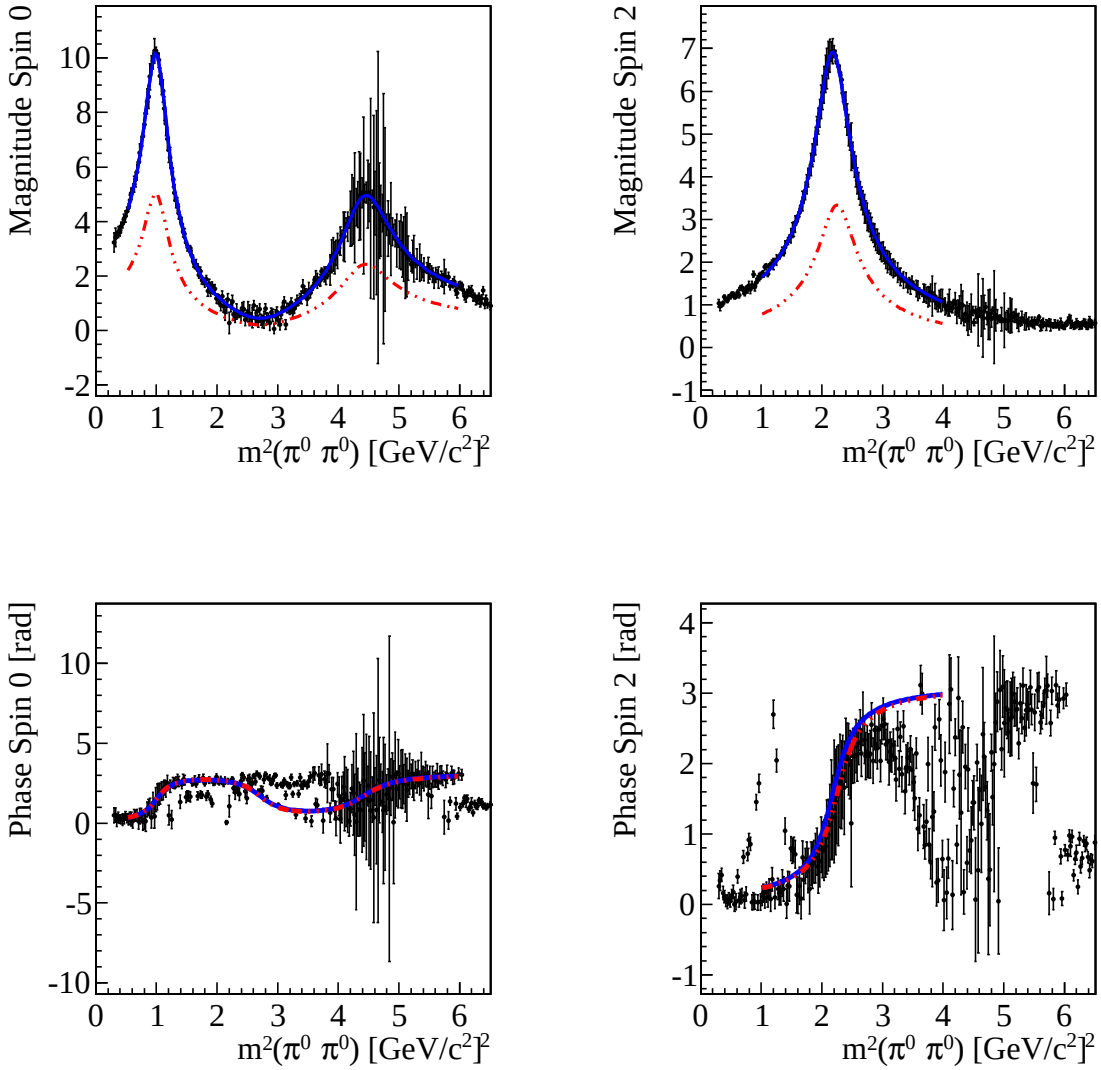

Figure A.10: Mass dependent fit coefficients c_0 and c_2 (top: magnitudes; bottom: phases) from equation 4.2 as the result of the Slice Fit of the simple toy data with a broader ω resonance. Also, the Breit-Wigner fit result (blue line), see table 4.6, and the Breit-Wigner sum with the ideal parameter setting (red dashed line) are plotted.

Table A.4: Parameter settings used for generation and extracted by the Slice Fit of the simple toy data with a broader ω resonance. m: mass of resonance in GeV/c^2, Γ: width of resonance in GeV.

Resonance		Generated	Fit Result	Pull
$f_0(A)$	m	1.	0.9976 ± 0.0006	4.00
	Γ	0.2	0.1608 ± 0.0012	32.67
$f_0(B)$	m	2.1	2.0933 ± 0.0013	5.15
	Γ	0.2	0.2041 ± 0.0013	3.15
	ϕ	1.6	1.6905 ± 0.0185	4.89
f_2	m	1.5	1.4756 ± 0.0081	3.01
	Γ	0.2	0.2019 ± 0.0239	0.08
ω	m	0.786	fixed	
	Γ	0.2	fixed	

A.3 Pull Distributions

The following plots show the pull distributions of the resonance parameters of the alternative setup, see table 4.3, performed with three different methods: the Dalitz plot fit, the binned Slice Fit and the unbinned Slice Fit.

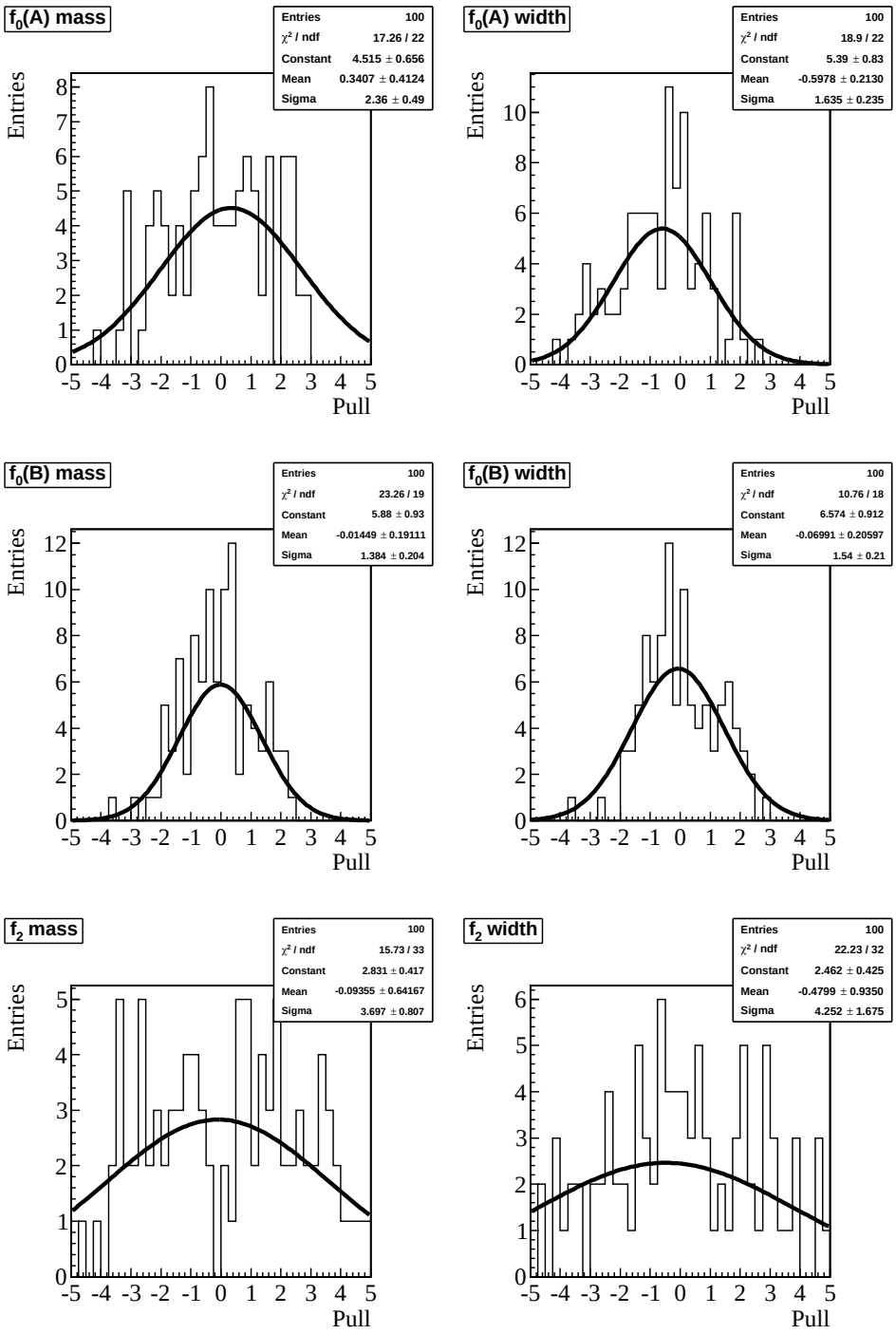

Figure A.11: Pull distribution of resonance parameters after 100 unbinned Dalitz plot fits to 100 000 events each generated with the simple setup, see table 4.3.

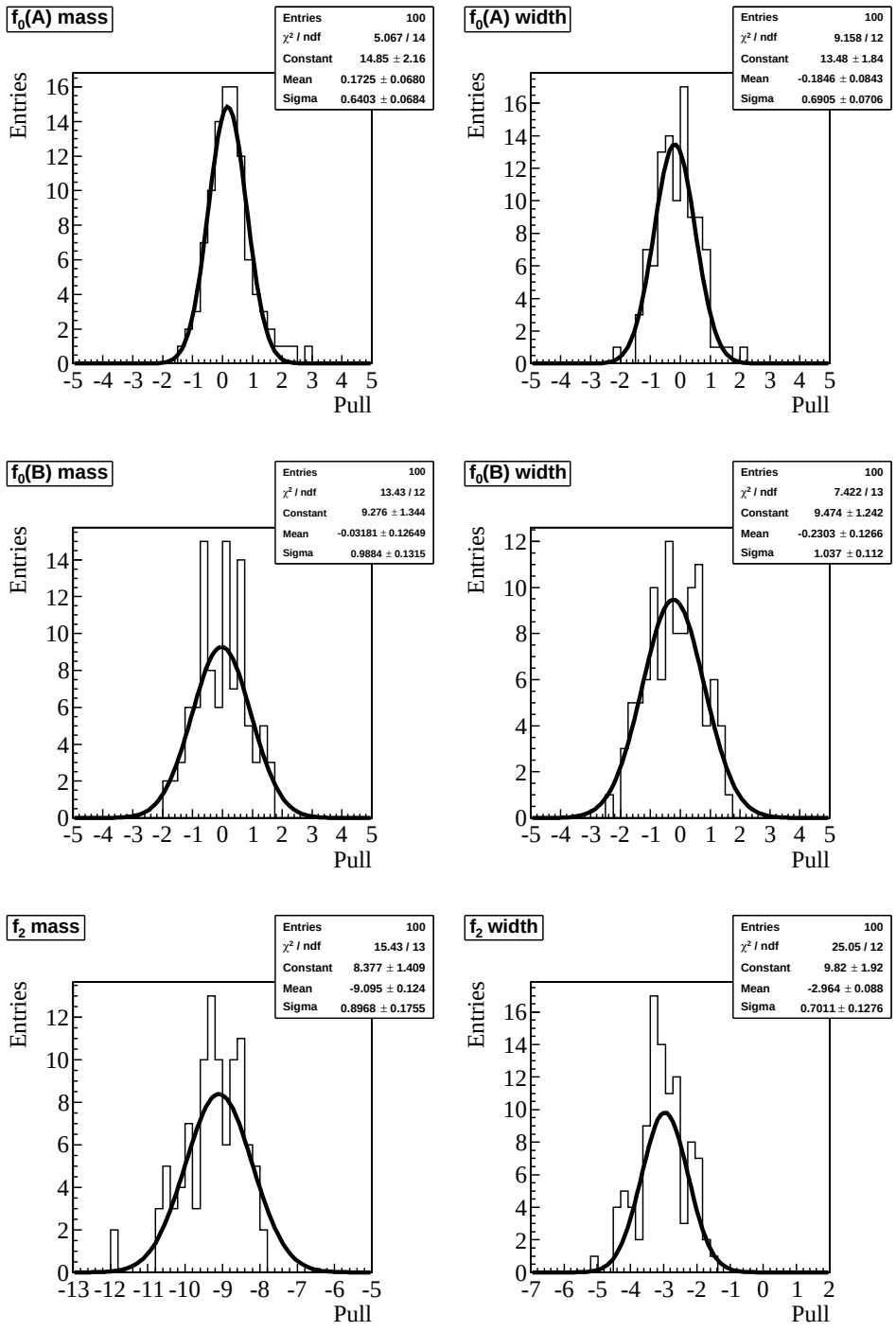

Figure A.12: Pull distribution of resonance parameters after 100 binned Slice Fits to 100 000 events each generated with the simple setup, see table 4.3.

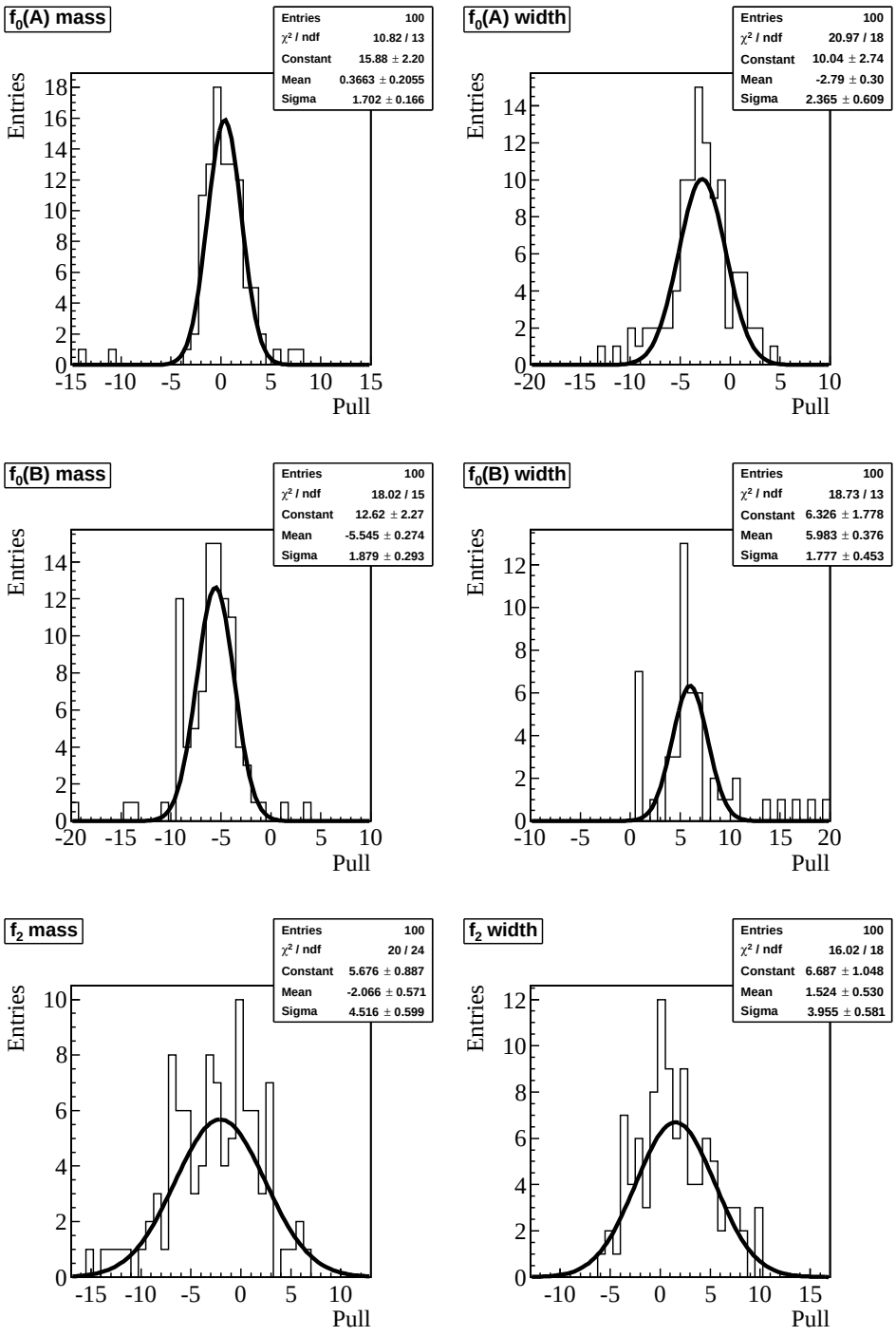

Figure A.13: Pull distribution of resonance parameters after 100 unbinned Slice Fits to 100 000 events each generated with the simple setup, see table 4.3.

A.4 CLIPS Expert System Test

On the following pages one can find the knowledge base used to demonstrate the capabilities of an expert system build using CLIPS [66] and the facts generated by using it.

Helicity like Knowledge Base

```
;;;======================================================
;;;   Helicity Formalism Knowledgebase
;;;
;;;     To execute, merely load, reset and run.
;;;======================================================

;;;****************
;;;*  TEMPLATES   *
;;;****************

;;;Input

(deftemplate Decay
(slot mothername (type SYMBOL) (default none))
(slot daugther1name (type SYMBOL) (default none))
(slot daugther2name (type SYMBOL) (default none))
(slot mothermass (type FLOAT) (default 0.))
(slot daugther1mass (type FLOAT) (default 0.))
(slot daugther2mass (type FLOAT) (default 0.))
(slot motherspin (type FLOAT) (default 0.))
(slot daugther1spin (type FLOAT) (default 0.))
(slot daugther2spin (type FLOAT) (default 0.))
(slot motherparity (type INTEGER) (default 1))
(slot daugther1parity (type INTEGER) (default 1))
(slot daugther2parity (type INTEGER) (default 1)) )

;;;Output

(deftemplate Sys
   (slot motherm (type FLOAT) (default 0.))
   (slot heli1 (type FLOAT) (default 0.))
   (slot heli2 (type FLOAT) (default 0.))
   (slot syss (type FLOAT) (default 0.))
   (slot sysl (type FLOAT) (default 0.)))

(deftemplate AmpTerms
   (slot djmnterm)
   (slot cgcterm)
   (slot dynamicterm)
   (slot normterm))


;;;****************
;;;*    RULES     *
;;;****************
```

```
;;; Calculate sys variables

(defrule SpinProj
   (Decay (motherspin ?myj))
   =>
   (loop-for-count (?m (* -1. ?myj) ?myj) do
   (assert (Sys (motherm ?m))) ) )

(defrule Heli1
   (Decay (daugther1spin ?myl))
   =>
   (loop-for-count (?m (* -1. ?myl) ?myl) do
   (assert (Sys (heli1 ?m))) ) )

(defrule Heli2
   (Decay (daugther2spin ?myl))
   =>
   (loop-for-count (?m (* -1. ?myl) ?myl) do
   (assert (Sys (heli2 ?m))) ) )

;;; {s,Max[j-l,Abs[s1-s2]],Min[j+l,s1+s2]}],
;;; {l,Max[j-s1-s2+(1-p)/2,(1-p)/2],j+s1+s2,2}])];

(defrule SysAngMom
   (Decay (motherspin ?myspin) (daugther1spin ?myspind1)
     (daugther2spin ?myspind2) (motherparity ?myp))
   =>
   (loop-for-count (?m (- (- ?myspin ?myspind1) ?myspind2)
     (+ (+ ?myspin ?myspind1) ?myspind2)) do
   (assert (Sys (sysl ?m))) ) )

(defrule SysSpin
   (Decay (motherspin ?myspin) (daugther1spin ?myspind1)
     (daugther2spin ?myspind2) (motherparity ?myp))
   (Sys (sysl ?myl))
   =>
   (loop-for-count (?m (- ?myspin ?myl) (+ ?myspin ?myl)) do
   (assert (Sys (syss ?m))) ) )


;;; Calculate Amplitudes

(defrule Norm
   (Sys (sysl ?myl))
   =>
   (assert (AmpTerms (normterm (str-cat "Sqrt[" (+ (* ?myl 2) 1) "]"))))
   (printout t "NormTerm calculated" crlf))

(defrule Dynamics
   (Sys (sysl ?myl) (syss ?mys))
   =>
   (assert (AmpTerms (dynamicterm (str-cat "a[" ?myl "," ?mys "]"))))
   (printout t "DynamicTerm calculated" crlf))
```

```
(defrule Cgc
   (Decay (motherspin ?myspin) (daugther1spin ?myspind1)
     (daugther2spin ?myspind2) )
   (Sys (sysl ?myl) (syss ?mys) (heli1 ?myl1) (heli2 ?myl2))
   =>
   (assert (AmpTerms (cgcterm (str-cat "ClebschGordan[{" ?myl ",0},
     {" ?mys "," (- ?myl1 ?myl2) "},{" ?myspin "," (- ?myl1 ?myl2) "}]
     * ClebschGordan[{" ?myspind1 "," ?myl1 "},{" ?myspind2 "," (* ?myl2 -1.) "}
     ,{" ?myspin "," (- ?myl1 ?myl2) "}]"))))
   (printout t "CgcTerm calculated" crlf))

(defrule Djmn
   (Decay (motherspin ?myspin))
   (Sys (motherm ?mym) (heli1 ?myl1) (heli2 ?myl2))
   =>
   (assert (AmpTerms (djmnterm (str-cat "Djmn[" ?myspin "," ?mym ","
     (- ?myl1 ?myl2)",Phi,Theta]]"))))
   (printout t "DjmnTerm calculated" crlf))


;;;  Heli[j_Integer,s1_Integer,s2_Integer,l1_Integer,l2_Integer,p_Integer]
;;;     :=Module[{l,s},(
;;; Sum[Sum[Times[Sqrt[2l+1],a[l,s],Cgc[j,l,s,s1,s2,l1,l2]],
;;; {s,Max[j-l,Abs[s1-s2]],Min[j+l,s1+s2]}],
;;; {l,Max[j-s1-s2+(1-p)/2,(1-p)/2],j+s1+s2,2}])];

;;;  Cgc[j_Integer,l_Integer,s_Integer,s1_Integer,
;;;    s2_Integer,l1_Integer,l2_Integer]:=0 /;
;;; (j>(l+s)) || (j<Abs[l-s]) || (j<Abs[l1-l2]) || (s<Abs[l1-l2]) ||
;;;        Abs[l1]>s1 || Abs[l2]>s2 || (s1+s2)>s || Abs[s1-s2]>s;

;;;  Cgc[j_Integer,l_Integer,s_Integer,s1_Integer,s2_Integer
:::    ,l1_Integer,l2_Integer]:=(
;;; Times[ClebschGordan[{l,0},{s,l1-l2},{j,l1-l2}],
;;; ClebschGordan[{s1,l1},{s2,-l2},{s,l1-l2}]])


;;;****************
;;;*    FACTS     *
;;;****************

(deffacts Input
  (Decay (mothername f2)
 (daugther1name pi0)
 (daugther2name pi0)
 (mothermass 1200.)
 (daugther1mass 140.)
 (daugther2mass 140.)
 (motherspin 2.)
 (daugther1spin 0.)
 (daugther2spin 0.)
 (motherparity 1)
```

```
 (daugther1parity 1)
 (daugther2parity 1)))

;;;(deffacts mother
;;;  (mass 1200)
;;;  (spin 2)
;;;  (parity 1))

;;;(deffacts daugther1
;;;  (mass 140)
;;;  (spin 0)
;;;  (parity 1))

;;;(deffacts daugther2
;;;  (mass 140)
;;;  (spin 0)
;;;  (parity 1))
```

Output of Helicity Test

```
         CLIPS (Quicksilver Beta 5/31/08)
CLIPS> (load "heli.clp")
Defining deftemplate: Decay
Defining deftemplate: Sys
Defining deftemplate: AmpTerms
Defining defrule: SpinProj +j+j
Defining defrule: Heli1 =j+j
Defining defrule: Heli2 =j+j
Defining defrule: SysAngMom =j+j
Defining defrule: SysSpin =j+j+j
Defining defrule: Norm +j+j
Defining defrule: Dynamics =j+j
Defining defrule: Cgc =j=j+j
Defining defrule: Djmn =j=j+j
Defining deffacts: Input
TRUE
CLIPS> (reset)
CLIPS> (run)
CgcTerm calculated
DjmnTerm calculated
NormTerm calculated
DynamicTerm calculated
CgcTerm calculated
DjmnTerm calculated
NormTerm calculated
DynamicTerm calculated
CgcTerm calculated
DjmnTerm calculated
NormTerm calculated
DynamicTerm calculated
CgcTerm calculated
DjmnTerm calculated
NormTerm calculated
DynamicTerm calculated
CgcTerm calculated
DjmnTerm calculated
NormTerm calculated
DynamicTerm calculated
CgcTerm calculated
DjmnTerm calculated
NormTerm calculated
DynamicTerm calculated
CgcTerm calculated
DjmnTerm calculated
NormTerm calculated
DynamicTerm calculated
CgcTerm calculated
DjmnTerm calculated
NormTerm calculated
DynamicTerm calculated
CgcTerm calculated
```

```
DjmnTerm calculated
NormTerm calculated
DynamicTerm calculated
CgcTerm calculated
DjmnTerm calculated
NormTerm calculated
DynamicTerm calculated
CgcTerm calculated
DjmnTerm calculated
NormTerm calculated
DynamicTerm calculated
CgcTerm calculated
DjmnTerm calculated
NormTerm calculated
DynamicTerm calculated
CgcTerm calculated
DjmnTerm calculated
NormTerm calculated
DynamicTerm calculated
CLIPS> (facts)
f-0     (initial-fact)
f-1     (Decay (mothername f2) (daugther1name pi0) (daugther2name pi0)
  (mothermass 1200.0) (daugther1mass 140.0) (daugther2mass 140.0)
  (motherspin 2.0) (daugther1spin 0.0) (daugther2spin 0.0) (motherparity 1)
  (daugther1parity 1) (daugther2parity 1))
f-2     (Sys (motherm -2) (heli1 0.0) (heli2 0.0) (syss 0.0) (sysl 0.0))
f-3     (Sys (motherm -1) (heli1 0.0) (heli2 0.0) (syss 0.0) (sysl 0.0))
f-4     (Sys (motherm 0) (heli1 0.0) (heli2 0.0) (syss 0.0) (sysl 0.0))
f-5     (Sys (motherm 1) (heli1 0.0) (heli2 0.0) (syss 0.0) (sysl 0.0))
f-6     (Sys (motherm 2) (heli1 0.0) (heli2 0.0) (syss 0.0) (sysl 0.0))
f-7     (Sys (motherm 0.0) (heli1 0.0) (heli2 0.0) (syss 2) (sysl 0.0))
f-8     (AmpTerms (djmnterm nil) (cgcterm "ClebschGordan[{0.0,0},{2,0.0},
  {2.0,0.0}] * ClebschGordan[{0.0,0.0},{0.0,-0.0},{2.0,0.0}]")
  (dynamicterm nil) (normterm nil))
f-9     (AmpTerms (djmnterm "Djmn[2.0,0.0,0.0,Phi,Theta]]") (cgcterm nil)
  (dynamicterm nil) (normterm nil))
f-10    (AmpTerms (djmnterm nil) (cgcterm nil) (dynamicterm nil)
  (normterm "Sqrt[1.0]"))
f-11    (AmpTerms (djmnterm nil) (cgcterm nil) (dynamicterm "a[0.0,2]")
  (normterm nil))
f-12    (AmpTerms (djmnterm nil) (cgcterm "ClebschGordan[{0.0,0},
  {0.0,0.0},{2.0,0.0}] * ClebschGordan[{0.0,0.0},{0.0,-0.0},{2.0,0.0}]")
  (dynamicterm nil) (normterm nil))
f-13    (AmpTerms (djmnterm "Djmn[2.0,2,0.0,Phi,Theta]]") (cgcterm nil)
  (dynamicterm nil) (normterm nil))
f-14    (AmpTerms (djmnterm nil) (cgcterm nil) (dynamicterm "a[0.0,0.0]")
  (normterm nil))
f-15    (AmpTerms (djmnterm "Djmn[2.0,1,0.0,Phi,Theta]]") (cgcterm nil)
  (dynamicterm nil) (normterm nil))
f-16    (AmpTerms (djmnterm "Djmn[2.0,0,0.0,Phi,Theta]]") (cgcterm nil)
  (dynamicterm nil) (normterm nil))
f-17    (AmpTerms (djmnterm "Djmn[2.0,-1,0.0,Phi,Theta]]") (cgcterm nil)
  (dynamicterm nil) (normterm nil))
```

```
f-18    (AmpTerms (djmnterm "Djmn[2.0,-2,0.0,Phi,Theta]]") (cgcterm nil)
  (dynamicterm nil) (normterm nil))
f-19    (Sys (motherm 0.0) (heli1 0) (heli2 0.0) (syss 0.0) (sysl 0.0))
f-20    (AmpTerms (djmnterm nil) (cgcterm "ClebschGordan[{0.0,0},
  {0.0,0.0},{2.0,0.0}] * ClebschGordan[{0.0,0},{0.0,-0.0},{2.0,0.0}]")
  (dynamicterm nil) (normterm nil))
f-21    (Sys (motherm 0.0) (heli1 0.0) (heli2 0) (syss 0.0) (sysl 0.0))
f-22    (Sys (motherm 0.0) (heli1 0.0) (heli2 0.0) (syss 0.0) (sysl 2))
f-23    (Sys (motherm 0.0) (heli1 0.0) (heli2 0.0) (syss 0) (sysl 0.0))
f-24    (Sys (motherm 0.0) (heli1 0.0) (heli2 0.0) (syss 1) (sysl 0.0))
f-25    (Sys (motherm 0.0) (heli1 0.0) (heli2 0.0) (syss 3) (sysl 0.0))
f-26    (Sys (motherm 0.0) (heli1 0.0) (heli2 0.0) (syss 4) (sysl 0.0))
f-27    (AmpTerms (djmnterm nil) (cgcterm "ClebschGordan[{0.0,0},
  {4,0.0},{2.0,0.0}] * ClebschGordan[{0.0,0.0},{0.0,-0.0},{2.0,0.0}]")
  (dynamicterm nil) (normterm nil))
f-28    (AmpTerms (djmnterm nil) (cgcterm nil) (dynamicterm "a[0.0,4]")
  (normterm nil))
f-29    (AmpTerms (djmnterm nil) (cgcterm "ClebschGordan[{0.0,0},
  {3,0.0},{2.0,0.0}] * ClebschGordan[{0.0,0.0},{0.0,-0.0},{2.0,0.0}]")
  (dynamicterm nil) (normterm nil))
f-30    (AmpTerms (djmnterm nil) (cgcterm nil) (dynamicterm "a[0.0,3]")
  (normterm nil))
f-31    (AmpTerms (djmnterm nil) (cgcterm "ClebschGordan[{0.0,0},
  {1,0.0},{2.0,0.0}] * ClebschGordan[{0.0,0.0},{0.0,-0.0},{2.0,0.0}]")
  (dynamicterm nil) (normterm nil))
f-32    (AmpTerms (djmnterm nil) (cgcterm nil) (dynamicterm "a[0.0,1]")
  (normterm nil))
f-33    (AmpTerms (djmnterm nil) (cgcterm "ClebschGordan[{0.0,0},
  {0,0.0},{2.0,0.0}] * ClebschGordan[{0.0,0.0},{0.0,-0.0},{2.0,0.0}]")
  (dynamicterm nil) (normterm nil))
f-34    (AmpTerms (djmnterm nil) (cgcterm nil) (dynamicterm "a[0.0,0]")
  (normterm nil))
f-35    (AmpTerms (djmnterm nil) (cgcterm "ClebschGordan[{2,0},
  {0.0,0.0},{2.0,0.0}] * ClebschGordan[{0.0,0.0},{0.0,-0.0},{2.0,0.0}]")
  (dynamicterm nil) (normterm nil))
f-36    (AmpTerms (djmnterm nil) (cgcterm nil) (dynamicterm nil)
  (normterm "Sqrt[5]"))
f-37    (AmpTerms (djmnterm nil) (cgcterm nil) (dynamicterm "a[2,0.0]")
  (normterm nil))
For a total of 38 facts.
CLIPS>
```